aws

Entwicklerhandbuch

Amazon Keyspaces (für Apache Cassandra)



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

Amazon Keyspaces (für Apache Cassandra): Entwicklerhandbuch

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

Die Handelsmarken und die Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist Amazon Keyspaces? ... 1
Funktionsweise ... 2
High-level Architektur ... 2
Cassandra-Datenmodell ... 5
Zugreifen auf Amazon Keyspaces ... 6
Anwendungsfälle ... 7
Was ist CQL? ... 8
Vergleichen Sie Amazon Keyspaces mit Cassandra ... 10
Funktionelle Unterschiede zu Apache Cassandra ... 11
APIs, Operationen und Datentypen von Apache Cassandra ... 12
Asynchrone Erstellung und Löschung von Schlüsselräumen und Tabellen ... 12
Authentifizierung und Autorisierung ... 12
Batch ... 12
Erfassung von Datenänderungen (Change Data Capture, CDC) ... 13
Cluster-Konfiguration ... 13
Verbindungen ... 13
`IN`Schlüsselwort ... 14
`FROZEN`Sammlungen ... 15
Einfache Transaktionen ... 15
Lastausgleich ... 15
Paginierung ... 16
Partitionierer ... 16
Vorbereitete Aussagen ... 16
Bereich löschen ... 17
Systemtabellen ... 17
Zeitstempel ... 18
User-defined Typen (UDTs) ... 18
Unterstützte Cassandra-APIs, Operationen, Funktionen und Datentypen ... 18
Cassandra-API-Unterstützung ... 19
API-Unterstützung für die Cassandra-Steuerebene ... 20
API-Unterstützung für Cassandra-Datenebene ... 21
Unterstützung der Cassandra-Funktion ... 21
Unterstützung des Cassandra-Datentyps ... 22
Unterstützte Cassandra-Konsistenzstufen ... 24

Konsistenzstufen für Schreibvorgänge ... 24
Lesen Sie die Konsistenzstufen ... 25
Konsistenzstufen werden nicht unterstützt ... 26
Migration zu Amazon Keyspaces ... 27
Migration von Cassandra ... 28
Kompatibilität ... 29
Schätzen Sie den Preis ... 30
Migrationsstrategie ... 40
Online-Migration ... 41
Offline-Migration ... 54
Hybridmigration ... 56
Migrationstools ... 61
Daten mit cqlsh laden ... 63
Daten mit DSBulk laden ... 77
Zugreifen auf Amazon Keyspaces ... 95
Melden Sie sich an für ein AWS-Konto ... 95
Amazon Keyspaces einrichten ... 95
Verwenden der Konsole ... 96
CloudShell Konsolenintegration ... 96
Benutzen AWS CloudShell ... 98
AWS CloudShell Integrationsoptionen ... 99
Voraussetzungen ... 99
Erste Schritte mit AWS CloudShell ... 99
Einholen von IAM-Berechtigungen für AWS CloudShell ... 100
Den Cqlsh-Client installieren ... 101
Konfiguration von TLS-Zertifikaten ... 101
Verbindung zu Amazon Keyspaces herstellen ... 101
Verwenden von CQLSH mit Amazon Keyspaces ... 102
Dauerhafte Daten in AWS CloudShell ... 102
Best Practices ... 102
Fehlerbehebung ... 102
Beispiel: So interagieren Sie mit Amazon Keyspaces mithilfe von AWS CloudShell ... 103
Erstellen Sie programmatische Zugangsdaten ... 105
Erstellen Sie dienstspezifische Anmeldeinformationen ... 106
Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ... 108
Service-Endpunkte ... 118

Ports und Protokolle ... 118
Globale Endpunkte ... 119
AWS GovCloud (US) Region FIPS-Endpunkte ... 125
Endpunkte der China Regionen ... 126
Streaming-Endpunkte ... 126
Verbindung zu Dual-Stack-Endpunkten herstellen ... 131
IPv6-Unterstützung in Amazon Keyspaces ... 132
DNS-Auflösung in Amazon Keyspaces ... 132
Verwenden von `cqlsh` ... 133
Verwendung der `cqlsh-expansion` ... 134
Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell ... 140
Mit dem AWS CLI ... 141
Herunterladen und Konfigurieren des AWS CLI ... 142
Verwenden von AWS CLI mit Amazon Keyspaces ... 142
Verwenden der API ... 147
Verwenden eines Cassandra-Client-Treibers ... 147
Verwenden eines Cassandra-Java-Client-Treibers ... 148
Verwenden eines Cassandra-Python-Client-Treibers ... 166
Verwenden eines Cassandra-Client-Treibers Node.js ... 170
Verwenden eines Cassandra.NET Core-Client-Treibers ... 175
Verwenden eines Cassandra Go-Client-Treibers ... 177
Verwenden eines Cassandra Perl-Client-Treibers ... 183
Konfigurieren Sie den kontoübergreifenden Zugriff ... 184
Kontenübergreifenden Zugriff in einer gemeinsam genutzten VPC konfigurieren ... 185
Kontenübergreifenden Zugriff ohne gemeinsam genutzte VPC konfigurieren ... 189
Erste Schritte ... 194
Voraussetzungen ... 195
Erstellen Sie einen Schlüsselraum ... 196
Überprüfen Sie den Status der Keyspace-Erstellung ... 199
Erstellen einer Tabelle ... 200
Überprüfen Sie den Status der Tabellenerstellung ... 208
CRUD-Operationen ... 209
Create ... 210
Lesen ... 214
Aktualisierung ... 219
Delete ... 220

Löschen einer Tabelle ... 222
Löschen Sie einen Schlüsselraum ... 225
Tutorials und Lösungen ... 229
Verbindung mit VPC-Endpunkten herstellen ... 229
Voraussetzungen ... 230
Schritt 1: Starten einer Amazon-EC2-Instance ... 230
Schritt 2: Konfigurieren Ihrer Amazon-EC2-Instance ... 232
Schritt 3: Erstellen Sie einen VPC-Endpunkt für Amazon Keyspaces ... 235
Schritt 4: Berechtigungen für die VPC-Endpunktverbindung konfigurieren ... 241
Schritt 5: Überwachung konfigurieren ... 245
Schritt 6: (Optional) Bewährte Methoden für Verbindungen ... 246
Schritt 7: (Optional) Aufräumen ... 249
Integration mit Apache Spark ... 250
Voraussetzungen ... 251
Schritt 1: Amazon Keyspaces konfigurieren ... 252
Schritt 2: Konfigurieren Sie den Apache Cassandra Spark Connector ... 254
Schritt 3: Erstellen Sie die App-Konfigurationsdatei ... 256
Schritt 4: Bereiten Sie die Quelldaten und die Zieltabelle vor ... 259
Schritt 5: Amazon Keyspaces-Daten schreiben und lesen ... 260
Fehlerbehebung ... 263
Verbindung über Amazon EKS herstellen ... 265
Voraussetzungen ... 266
Schritt 1: Den Amazon EKS-Cluster konfigurieren ... 269
Schritt 2: Konfigurieren Sie die Anwendung ... 274
Schritt 3: Anwendungs-Image erstellen ... 277
Schritt 4: Bereitstellen der Anwendung auf Amazon EKS ... 279
Schritt 5: (Optional) Aufräumen ... 284
Daten nach Amazon S3 exportieren ... 287
Voraussetzungen ... 287
Schritt 1: Bootstrap-Infrastruktur und Jobs AWS Glue ... 289
Schritt 2: Führen Sie den Exportauftrag aus ... 292
Schritt 3: (Optional) Planen Sie den Exportauftrag ... 294
Schritt 4: (Optional) Bereinigung ... 297
Verwaltung serverloser Ressourcen ... 300
Schätzen Sie die Zeilengröße ... 301
Schätzen Sie die kodierte Größe von Spalten ... 302

Schätzen Sie die kodierte Größe von Datenwerten auf der Grundlage des Datentyps 303
Berücksichtigen Sie die Auswirkungen der Funktionen von Amazon Keyspaces auf die Zeilengröße 305
Wählen Sie die richtige Formel, um die kodierte Größe einer Zeile zu berechnen 306
Beispiel für die Berechnung der Zeilengröße 307
Schätzen Sie den Kapazitätsverbrauch 308
Schätzen Sie den Kapazitätsverbrauch von Bereichsabfragen 309
Schätzen Sie den Lesekapazitätsverbrauch von Grenzabfragen 310
Schätzen Sie den Lesekapazitätsverbrauch von Tabellenscans ab 311
Schätzen Sie den Kapazitätsverbrauch von LWT 312
Schätzen Sie den Kapazitätsverbrauch statischer Spalten 313
Schätzen Sie die Kapazität für eine Tabelle mit mehreren Regionen 318
Schätzen Sie den Kapazitätsverbrauch mit CloudWatch 319
read/write Kapazitätsmodi konfigurieren 320
Konfigurieren Sie den Kapazitätsmodus auf Anforderung 320
Konfigurieren Sie den Modus für bereitgestellte Kapazität 323
Den Kapazitätsmodus einer Tabelle anzeigen 325
Ändern Sie den Kapazitätsmodus 327
Vorwärmen konfigurieren 330
Verwalten Sie die Durchsatzkapazität mit Auto Scaling 345
So funktioniert die automatische Skalierung von Amazon Keyspaces 346
So funktioniert Auto Scaling für Tabellen mit mehreren Regionen 347
Nutzungshinweise 348
Auto Scaling-Richtlinien konfigurieren und aktualisieren 349
Verwenden Sie Burst-Kapazität 364
Arbeiten mit Amazon Keyspaces-Funktionen 365
Systemschlüsselräume 367
`system` 367
`system_schema` 369
`system_schema_mcs` 370
`system_multiregion_info` 373
User-defined Typen (UDTs) 376
Konfigurieren von -Berechtigungen 377
Erstellen Sie ein UDT 381
UDTs anzeigen 385
Löschen Sie eine UDT 389

Arbeiten mit CQL-Abfragen ... 390
Verwenden IN SELECT ... 390
Verwenden Sie Batch-Anweisungen ... 394
Ergebnisse bestellen ... 402
Ergebnisse paginieren ... 403
Arbeiten mit CDC-Streams ... 404
Wie funktionieren CDC-Streams ... 405
Verwenden Sie CDC-Streams ... 410
Mit Partitionierern arbeiten ... 447
Ändern Sie den Partitionierer ... 447
Client-side Zeitstempel ... 449
AWS Integration mit Diensten ... 451
Erstellen Sie eine Tabelle mit clientseitigen Zeitstempeln ... 451
Konfigurieren Sie clientseitige Zeitstempel ... 454
Verwenden Sie clientseitige Zeitstempel in Abfragen ... 457
Multi-Region Replikation ... 458
Vorteile ... 459
Kapazitätsmodi und Preisgestaltung ... 460
Funktionsweise ... 461
Nutzungshinweise ... 466
Multi-Region-Replikation konfigurieren ... 469
Backup und Wiederherstellung mit Point-in-Time-Recovery ... 499
Funktionsweise ... 499
Verwenden Sie die Point-in-Time-Wiederherstellung ... 505
Daten mit Time to Live ablaufen lassen ... 519
Integration mit Diensten AWS ... 521
Erstellen Sie eine Tabelle mit dem TTL-Standardwert ... 521
Aktualisieren Sie den TTL-Standardwert der Tabelle ... 525
Tabelle mit benutzerdefinierter TTL erstellen ... 529
Tabelle mit benutzerdefinierter TTL aktualisieren ... 531
Wird verwendetINSERT, um benutzerdefinierte TTL für neue Zeilen festzulegen ... 533
Wird verwendetUPDATE, um benutzerdefinierte TTL für Zeilen und Spalten festzulegen ... 534
Arbeiten mit AWS SDKs ... 536
Mit Tags arbeiten ... 537
Tagging-Einschränkungen ... 538
Taggen Sie Schlüsselräume, Tabellen und Streams ... 539

Erstellen Sie Berichte zur Kostenverteilung ... 557
Ressourcen erstellen CloudFormation ... 559
Amazon Keyspaces und Vorlagen CloudFormation ... 559
Erfahren Sie mehr über CloudFormation ... 559
NoSQL Workbench ... 560
Download ... 561
Erste Schritte ... 561
Visualisieren Sie ein Datenmodell ... 563
Erstellen Sie ein Datenmodell ... 567
Bearbeiten Sie ein Datenmodell ... 570
Übernehmen Sie ein Datenmodell ... 572
Beispieldatenmodelle ... 583
Versionsverlauf ... 584
Codebeispiele ... 586
Grundlagen ... 587
Hello Amazon Keyspaces ... 587
Kennenlernen der Grundlagen ... 592
Aktionen ... 655
Bibliotheken und Tools ... 707
Bibliotheken und Beispiele ... 707
Amazon Keyspaces (für Apache Cassandra) Entwickler-Toolkit ... 707
Beispiele für Amazon Keyspaces (für Apache Cassandra) ... 707
AWS Authentifizierungs-Plugins für Signature Version 4 (Sigv4) ... 707
Hervorgehobene Beispiel- und Entwickler-Tool-Repos ... 708
Amazon Keyspaces-Protokollpuffer ... 708
CloudFormation Vorlage zur Erstellung eines CloudWatch Amazon-Dashboards für Amazon Keyspaces-Metriken (für Apache Cassandra) ... 708
Verwenden von Amazon Keyspaces (für Apache Cassandra) mit AWS Lambda ... 709
Verwenden von Amazon Keyspaces (für Apache Cassandra) mit Spring ... 709
Verwenden von Amazon Keyspaces (für Apache Cassandra) mit Scala ... 709
Verwenden von Amazon Keyspaces (für Apache Cassandra) mit AWS Glue ... 709
Amazon Keyspaces (für Apache Cassandra) Cassandra Query Language (CQL) zum Konverter CloudFormation ... 710
Amazon Keyspaces (für Apache Cassandra) Helfer für den Apache Cassandra-Treiber für Java ... 710
Schnelle Komprimierungsdemo von Amazon Keyspaces (für Apache Cassandra) ... 710

Amazon Keyspaces (für Apache Cassandra) und Amazon S3 S3-Codec-Demo 710
Bewährte Methoden 711
NoSQL-Design 712
NoSQL gegenüber RDBMS 713
Zwei Schlüsselkonzepte 713
Allgemeiner Ansatz 714
Verbindungen 715
Wie funktionieren sie 716
Wie konfiguriert man Verbindungen 717
Wie konfiguriert man Richtlinien für Wiederholungsversuche 719
VPC-Endpunktverbindungen 719
Wie überwacht man Verbindungen 721
Wie gehe ich mit Verbindungsfehlern um 722
Datenmodellierung 722
Entwerfen von Partitionsschlüsseln 723
Kostenoptimierung 726
Auswerten Ihrer Kosten auf Tabellenebene 727
Bewerten Sie den Kapazitätsmodus Ihrer Tabelle 729
Evaluieren Sie die Application Auto Scaling Scaling-Einstellungen Ihrer Tabelle 733
Ermitteln von ungenutzten Ressourcen 741
Auswerten Ihrer Tabellennutzungsmuster 747
Auswerten Ihrer bereitgestellten Kapazität, um eine angemessene Bereitstellung zu erzielen 748
Fehlerbehebung 758
Allgemeine Fehler 759
Allgemeine Fehler 759
Verbindungsfehler 761
Fehler beim Herstellen einer Verbindung zu einem Amazon Keyspaces-Endpunkt 761
Fehler bei der Kapazitätsverwaltung 774
Kapazitätsfehler bei serverlosen Servern 774
Sprachfehler bei der Datendefinition 780
Sprachfehler bei der Datendefinition 781
Überwachung von Amazon Keyspaces 786
Überwachung mit CloudWatch 787
Verwenden von -Metriken 788
Metriken und Dimensionen 790

Anzeigen von -Metriken ... 814
Erstellen von -Alarmen ... 815
Protokollierung mit CloudTrail ... 816
Konfiguration von Protokolldateieinträgen in CloudTrail ... 816
DDL-Informationen in CloudTrail ... 817
DML-Informationen in CloudTrail ... 818
Grundlagen zu -Protokolldateieinträgen ... 819
Sicherheit ... 831
Datenschutz ... 832
Verschlüsselung im Ruhezustand ... 833
Verschlüsselung während der Übertragung ... 856
Richtlinie für den Datenverkehr zwischen Netzwerken ... 856
AWS Identity and Access Management ... 857
Zielgruppe ... 858
Authentifizierung mit Identitäten ... 858
Verwalten des Zugriffs mit Richtlinien ... 860
So funktioniert Amazon Keyspaces mit IAM ... 862
Identity-based Beispiele für Richtlinien ... 867
AWS verwaltete Richtlinien ... 876
Fehlerbehebung ... 889
Verwenden von servicegebundenen Rollen ... 893
Compliance-Validierung ... 903
Ausfallsicherheit ... 904
Sicherheit der Infrastruktur ... 905
Verwendung von -Schnittstellen-VPC-Endpunkten ... 906
Verwenden von Schnittstellen-VPC-Endpunkten mit Streams ... 913
Konfiguration und Schwachstellenanalyse für Amazon Keyspaces ... 917
Bewährte Methoden für die Gewährleistung der Sicherheit ... 917
Bewährte Methoden für vorbeugende Sicherheitsmaßnahmen ... 917
Bewährte Methoden für aufdeckende Sicherheitsmaßnahmen ... 919
CQL-Sprachreferenz ... 922
Sprachelemente ... 923
IDs ... 923
Konstanten ... 923
Bedingungen ... 924
Datentypen ... 924

JSON-Kodierung von Amazon Keyspaces-Datentypen 928
DDL-Anweisungen 932
Keyspaces 933
Tabellen 937
Typen 952
DML-Anweisungen. 955
SELECT 955
INSERT 958
UPDATE 960
DELETE 961
Built-in Funktionen 962
Skalarfunktionen 962
Kontingente 965
Amazon Keyspaces-Servicekontingente 965
Erhöhen oder Verringern des Durchsatzes (für Tabellen im Modus bereitgestellter Kapazität) .. 973
Erhöhen des bereitgestellten Durchsatzes 973
Senken des bereitgestellten Durchsatzes 973
Amazon Keyspaces-Verschlüsselung im Ruhezustand 973
Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces ... 974
Amazon Keyspaces UDT-Kontingente und Standardwerte 974
Dokumentverlauf 976
..... cmxcii

Was ist Amazon Keyspaces (für Apache Cassandra)?

Amazon Keyspaces (für Apache Cassandra) ist ein skalierbarer, hochverfügbarer und verwalteter Apache Cassandra-kompatibler Datenbankservice. Mit Amazon Keyspaces müssen Sie keine Server bereitstellen, patchen oder verwalten und Sie müssen keine Software installieren, warten oder betreiben.

Amazon Keyspaces ist serverlos, sodass Sie nur für die Ressourcen zahlen, die Sie nutzen, und der Service skaliert Tabellen automatisch als Reaktion auf den Anwendungsdatenverkehr nach oben und unten. Sie können Anwendungen erstellen, die Tausende von Anfragen pro Sekunde mit praktisch unbegrenztem Durchsatz und Speicherplatz verarbeiten.

> **Note**
>
> Apache Cassandra ist ein Open-Source-Datenspeicher mit breiten Spalten, der für die Verarbeitung großer Datenmengen konzipiert ist. Weitere Informationen finden Sie unter Apache Cassandra. http://cassandra.apache.org/

Amazon Keyspaces macht es einfach, Cassandra-Workloads in der zu migrieren, auszuführen und zu skalieren. AWS Cloud Mit nur wenigen Klicks auf der AWS Management Console oder ein paar Codezeilen können Sie Schlüsselbereiche und Tabellen in Amazon Keyspaces erstellen, ohne Infrastruktur bereitstellen oder Software installieren zu müssen.

Mit Amazon Keyspaces können Sie Ihre vorhandenen Cassandra-Workloads mit demselben Cassandra-Anwendungscode und denselben Entwicklertools ausführen, die Sie heute AWS verwenden.

Mit dem Preisrechner für Amazon Keyspaces (für Apache Cassandra), der auf Github verfügbar ist, können Sie Ihre monatlichen Kosten für Amazon Keyspaces auf der Grundlage Ihrer vorhandenen Apache Cassandra-Arbeitslast schätzen. Geben Sie Kennzahlen aus Ihrer Cassandra-Nodetool-Statusausgabe und der geplanten serverlosen Konfiguration für Amazon Keyspaces ein, um die direkten Kosten zwischen den beiden Lösungen zu vergleichen. Beachten Sie, dass sich dieser Rechner nur auf die Betriebskosten von Amazon Keyspaces im Vergleich zu Ihrer bestehenden Cassandra-Bereitstellung konzentriert. Faktoren der Gesamtbetriebskosten (TCO) wie Infrastrukturwartung, Betriebskosten oder Supportkosten für Cassandra sind nicht enthalten.

Eine Liste der verfügbaren AWS-Regionen Endpunkte finden Sie unter Service-Endpunkte für Amazon Keyspaces.

Wir empfehlen Ihnen, zunächst die folgenden Abschnitte zu lesen:

Themen

- Amazon Keyspaces: So funktioniert es
- Anwendungsfälle von Amazon Keyspaces
- Was ist Cassandra Query Language (CQL)?

Amazon Keyspaces: So funktioniert es

Amazon Keyspaces beseitigt den Verwaltungsaufwand für die Verwaltung von Cassandra. Um zu verstehen, warum, ist es hilfreich, mit der Cassandra-Architektur zu beginnen und sie dann mit Amazon Keyspaces zu vergleichen.

Themen

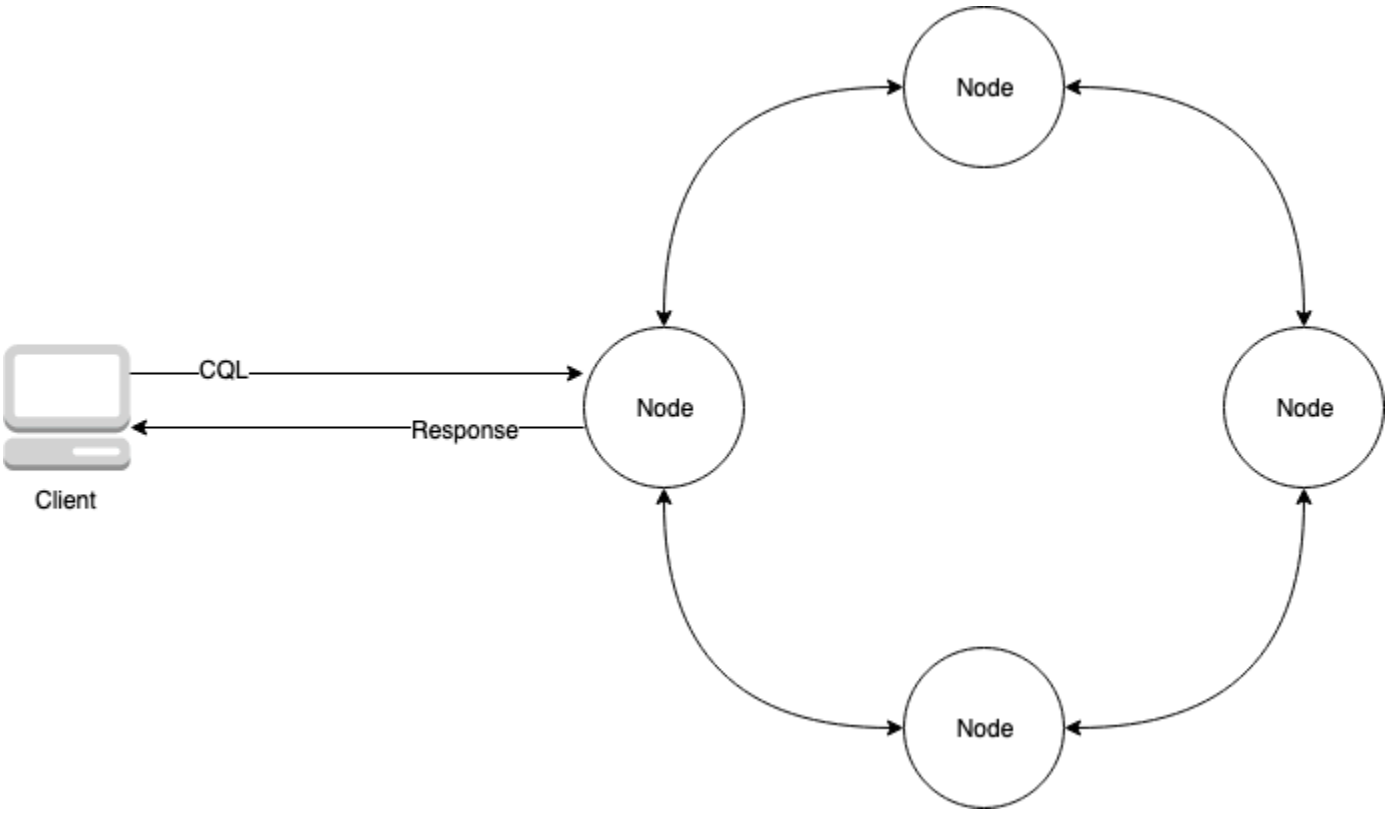
- High-level Architektur: Apache Cassandra im Vergleich zu Amazon Keyspaces
- Cassandra-Datenmodell
- Zugreifen auf Amazon Keyspaces von einer Anwendung aus

High-level Architektur: Apache Cassandra im Vergleich zu Amazon Keyspaces

Traditionell wird Apache Cassandra in einem Cluster eingesetzt, der aus einem oder mehreren Knoten besteht. Sie sind dafür verantwortlich, jeden Knoten zu verwalten und Knoten hinzuzufügen und zu entfernen, wenn Ihr Cluster skaliert.

Ein Client-Programm greift auf Cassandra zu, indem es eine Verbindung zu einem der Knoten herstellt und Cassandra Query Language (CQL) -Anweisungen ausgibt. CQL ähnelt SQL, der beliebten Sprache, die in relationalen Datenbanken verwendet wird. Obwohl Cassandra keine relationale Datenbank ist, bietet CQL eine vertraute Oberfläche zum Abfragen und Bearbeiten von Daten in Cassandra.

Das folgende Diagramm zeigt einen einfachen Apache Cassandra-Cluster, der aus vier Knoten besteht.



Eine Cassandra-Produktionsbereitstellung kann aus Hunderten von Knoten bestehen, die auf Hunderten von physischen Computern in einem oder mehreren physischen Rechenzentren ausgeführt werden. Dies kann für Anwendungsentwickler, die neben der Installation, Wartung und dem Betrieb von Software auch Server bereitstellen, patchen und verwalten müssen, eine betriebliche Belastung darstellen.

Mit Amazon Keyspaces (für Apache Cassandra) müssen Sie keine Server bereitstellen, patchen oder verwalten, sodass Sie sich auf die Entwicklung besserer Anwendungen konzentrieren können. Amazon Keyspaces bietet zwei Durchsatzkapazitätsmodi für Lese- und Schreibvorgänge: bei Bedarf und bereitgestellt. Sie können den Durchsatzkapazitätsmodus Ihrer Tabelle wählen, um den Preis für Lese- und Schreibvorgänge auf der Grundlage der Vorhersagbarkeit und Variabilität Ihrer Arbeitslast zu optimieren.

Im On-Demand-Modus zahlen Sie nur für die Lese- und Schreibvorgänge, die Ihre Anwendung tatsächlich ausführt. Sie müssen die Durchsatzkapazität Ihrer Tabelle nicht im Voraus angeben. Amazon Keyspaces nimmt Ihren Anwendungsdatenverkehr fast sofort auf, wenn er hoch- oder heruntergeht, und ist daher eine gute Option für Anwendungen mit unvorhersehbarem Datenverkehr.

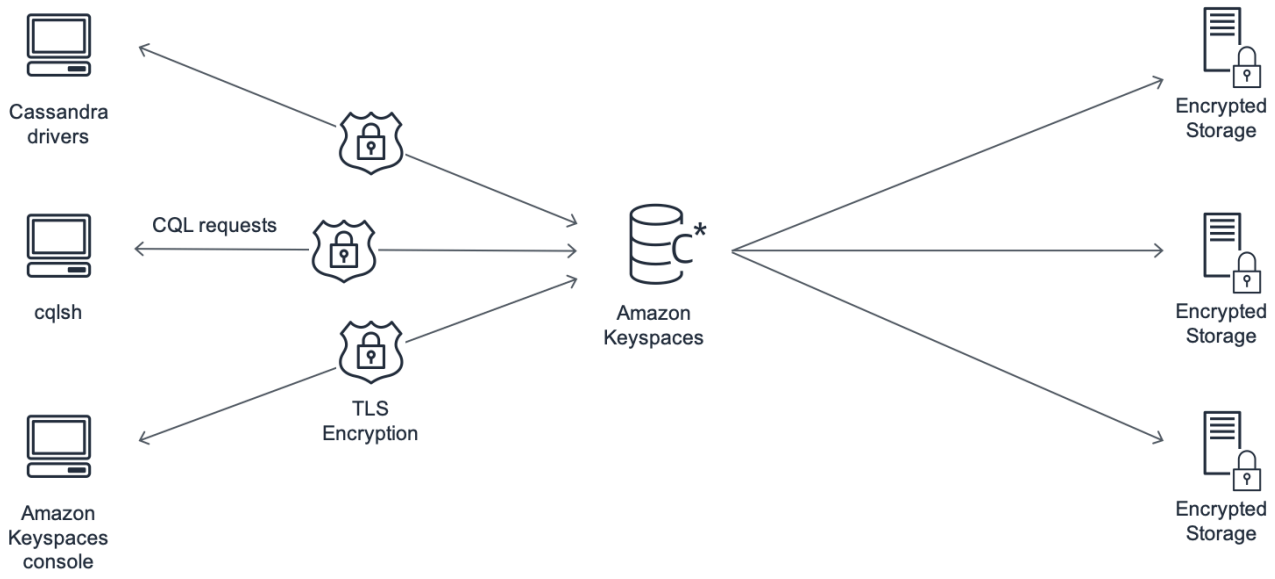
Der Modus „Bereitgestellte Kapazität“ hilft Ihnen dabei, den Durchsatzpreis zu optimieren, wenn Sie über einen vorhersehbaren Anwendungsdatenverkehr verfügen und die Kapazitätsanforderungen

Ihrer Tabelle im Voraus prognostizieren können. Im Modus für bereitgestellte Kapazität geben Sie die Anzahl der Lese- und Schreibvorgänge pro Sekunde an, die Ihre Anwendung voraussichtlich ausführen wird. Sie können die bereitgestellte Kapazität für Ihre Tabelle automatisch erhöhen und verringern, indem Sie die automatische Skalierung aktivieren.

Sie können den Kapazitätsmodus Ihrer Tabelle einmal täglich ändern, wenn Sie mehr über die Datenverkehrsmuster Ihres Workloads erfahren, oder wenn Sie einen starken Anstieg des Datenverkehrs erwarten, z. B. aufgrund eines wichtigen Ereignisses, von dem Sie erwarten, dass es zu einem hohen Tabellenverkehr führen wird. Weitere Informationen zur Bereitstellung von Lese- und Schreibkapazität finden Sie unterthe section called “ read/write Kapazitätsmodi konfigurieren”.

Amazon Keyspaces (für Apache Cassandra) speichert drei Kopien Ihrer Daten in mehreren Availability Zones, um Haltbarkeit und Hochverfügbarkeit zu gewährleisten. Darüber hinaus profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die auf die Anforderungen der sicherheitsrelevantesten Organisationen zugeschnitten ist. Die Verschlüsselung im Ruhezustand wird automatisch aktiviert, wenn Sie eine neue Amazon Keyspaces-Tabelle erstellen und für alle Client-Verbindungen Transport Layer Security (TLS) erforderlich ist. Zu den zusätzlichen AWS Sicherheitsfunktionen gehören Überwachung und Virtual Private Cloud (VPC) -Endpunkte. AWS Identity and Access Management Eine Übersicht über alle verfügbaren Sicherheitsfunktionen finden Sie unter. Sicherheit

Das folgende Diagramm zeigt die Architektur von Amazon Keyspaces.



Ein Client-Programm greift auf Amazon Keyspaces zu, indem es eine Verbindung zu einem vorbestimmten Endpunkt (Hostname und Portnummer) herstellt und CQL-Anweisungen ausgibt. Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Service-Endpunkte”

Cassandra-Datenmodell

Wie Sie Ihre Daten für Ihren Geschäftsfall modellieren, ist entscheidend, um eine optimale Leistung von Amazon Keyspaces zu erzielen. Ein schlechtes Datenmodell kann die Leistung erheblich beeinträchtigen.

Obwohl CQL SQL ähnlich sieht, sind die Backends von Cassandra und relationalen Datenbanken sehr unterschiedlich und müssen unterschiedlich angegangen werden. Im Folgenden sind einige der wichtigsten Aspekte aufgeführt, die es zu berücksichtigen gilt:

Speicherung

Sie können Ihre Cassandra-Daten in Tabellen visualisieren, wobei jede Zeile einen Datensatz und jede Spalte ein Feld innerhalb dieses Datensatzes darstellt.

Tabellendesign: Zuerst abfragen

Es gibt keine `JOIN` s in CQL. Daher sollten Sie Ihre Tabellen mit der Form Ihrer Daten und der Art und Weise, wie Sie für Ihre geschäftlichen Anwendungsfälle darauf zugreifen müssen, entwerfen. Dies kann zu einer Deinormalisierung bei duplizierten Daten führen. Sie sollten jede Ihrer Tabellen speziell für ein bestimmtes Zugriffsmuster entwerfen.

Partitionen

Ihre Daten werden in Partitionen auf der Festplatte gespeichert. Die Anzahl der Partitionen, in denen Ihre Daten gespeichert werden, und wie sie auf die Partitionen verteilt werden, wird durch Ihren Partitionsschlüssel bestimmt. Wie Sie Ihren Partitionsschlüssel definieren, kann erhebliche Auswirkungen auf die Leistung Ihrer Abfragen haben. Bewährte Methoden finden Sie unter the section called “Entwerfen von Partitionsschlüsseln”.

Primärschlüssel

In Cassandra werden Daten als Schlüssel-Wert-Paar gespeichert. Jede Cassandra-Tabelle muss einen Primärschlüssel haben, der der eindeutige Schlüssel für jede Zeile in der Tabelle ist. Der Primärschlüssel setzt sich aus einem erforderlichen Partitionsschlüssel und optionalen Clusterspalten zusammen. Die Daten, aus denen der Primärschlüssel besteht, müssen für alle Datensätze in einer Tabelle eindeutig sein.

- Partitionsschlüssel — Der Partitionsschlüsselteil des Primärschlüssels ist erforderlich und bestimmt, in welcher Partition Ihres Clusters die Daten gespeichert sind. Der Partitionsschlüssel kann eine einzelne Spalte oder ein zusammengesetzter Wert sein, der aus zwei oder mehr Spalten besteht. Sie würden einen zusammengesetzten Partitionsschlüssel verwenden, wenn ein einspaltiger Partitionsschlüssel dazu führen würde, dass eine einzelne Partition oder nur sehr wenige Partitionen die meisten Daten enthalten und somit die meisten I/O Festplattenoperationen ausführen.
- Clustering-Spalte — Die optionale Cluster-Spalte Ihres Primärschlüssels bestimmt, wie die Daten innerhalb der einzelnen Partitionen gruppiert und sortiert werden. Wenn Sie eine Cluster-Spalte in Ihren Primärschlüssel aufnehmen, kann die Cluster-Spalte eine oder mehrere Spalten enthalten. Wenn die Cluster-Spalte mehrere Spalten enthält, wird die Sortierreihenfolge durch die Reihenfolge bestimmt, in der die Spalten in der Cluster-Spalte von links nach rechts aufgeführt sind.

Weitere Informationen zum NoSQL-Design und zu Amazon Keyspaces finden Sie unter. the section called “NoSQL-Design” Weitere Informationen zu Amazon Keyspaces und Datenmodellierung finden Sie unter. the section called “Datenmodellierung”

Zugreifen auf Amazon Keyspaces von einer Anwendung aus

Amazon Keyspaces (für Apache Cassandra) implementiert die Apache Cassandra Query Language (CQL) API, sodass Sie CQL- und Cassandra-Treiber verwenden können, die Sie bereits verwenden. Das Aktualisieren Ihrer Anwendung ist so einfach wie das Aktualisieren Ihres Cassandra-Treibers oder Ihrer `cqlsh` Konfiguration, sodass sie auf den Amazon Keyspaces-Service-Endpunkt verweisen. Weitere Informationen zu den erforderlichen Anmeldeinformationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

> **Note**
>
> Um Ihnen den Einstieg zu erleichtern, finden Sie im Amazon Keyspaces-Codebeispiel-Repository unter Verwendung verschiedener Cassandra-Client-Treiber umfassende Codebeispiele für die Verbindung zu Amazon Keyspaces. GitHub

Stellen Sie sich das folgende Python-Programm vor, das eine Verbindung zu einem Cassandra-Cluster herstellt und eine Tabelle abfragt.

```
from cassandra.cluster import Cluster
#TLS/SSL configuration goes here

ksp = 'MyKeyspace'
tbl = 'WeatherData'

cluster = Cluster(['NNN.NNN.NNN.NNN'], port=NNNN)
session = cluster.connect(ksp)

session.execute('USE ' + ksp)

rows = session.execute('SELECT * FROM ' +  tbl)
for row in rows:
    print(row)
```

Um dasselbe Programm mit Amazon Keyspaces auszuführen, müssen Sie:

- Cluster-Endpunkt und Port hinzufügen: Beispielsweise kann der Host durch einen Service-Endpunkt ersetzt werden, z. B. `cassandra.us-east-1.amazonaws.com` und die Portnummer durch:`9142`.
- TLS/SSL Konfiguration hinzufügen: Weitere Informationen zum Hinzufügen der TLS/SSL Konfiguration für die Verbindung zu Amazon Keyspaces mithilfe eines Cassandra-Client-Python-Treibers finden Sie unter. Verwenden eines Cassandra-Python-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

Anwendungsfälle von Amazon Keyspaces

Im Folgenden sind nur einige der Möglichkeiten aufgeführt, wie Sie Amazon Keyspaces verwenden können:

- Erstellen Sie Anwendungen, die eine niedrige Latenz benötigen — Verarbeiten Sie Daten mit hoher Geschwindigkeit für Anwendungen, die eine Latenz im einstelligen Millisekundenbereich erfordern, wie z. B. die Wartung von Industrieanlagen, Handelsüberwachung, Flottenmanagement und Routenoptimierung.
- Erstellen Sie Anwendungen mithilfe von Open-Source-Technologien — Erstellen Sie Anwendungen AWS mithilfe von Open-Source-Cassandra-APIs und -Treibern, die für eine Vielzahl von Programmiersprachen wie Java, Python, Ruby, Microsoft.NET, PHP, C++, Perl und Go Node.js verfügbar sind. Codebeispiele finden Sie unter Bibliotheken und Tools.

- Verschieben Sie Ihre Cassandra-Workloads in die Cloud — Cassandra-Tabellen selbst zu verwalten ist zeitaufwändig und teuer. Mit Amazon Keyspaces können Sie Cassandra-Tabellen einrichten, sichern und skalieren, ohne die Infrastruktur verwalten zu müssen. AWS Cloud Weitere Informationen finden Sie unter Verwaltung serverloser Ressourcen.

Was ist Cassandra Query Language (CQL)?

Cassandra Query Language (CQL) ist die Hauptsprache für die Kommunikation mit Apache Cassandra. Amazon Keyspaces (für Apache Cassandra) ist mit der CQL 3.x-API kompatibel (abwärtskompatibel mit Version 2.x).

In CQL werden Daten in Tabellen, Spalten und Zeilen gespeichert. In diesem Sinne ähnelt CQL der Structured Query Language (SQL). Dies sind die Schlüsselkonzepte in CQL.

- CQL-Elemente — Die grundlegenden Elemente von CQL sind Bezeichner, Konstanten, Begriffe und Datentypen.
- Data Definition Language (DDL) — DDL-Anweisungen werden zur Verwaltung von Datenstrukturen wie Schlüsselräumen und Tabellen verwendet, bei denen es sich um Ressourcen in Amazon Keyspaces handelt. AWS DDL-Anweisungen sind Operationen auf Steuerungsebene in. AWS
- Data Manipulation Language (DML) — DML-Anweisungen werden zur Verwaltung von Daten in Tabellen verwendet. DML-Anweisungen werden zum Auswählen, Einfügen, Aktualisieren und Löschen von Daten verwendet. Dies sind Operationen auf Datenebene in. AWS
- Built-in Funktionen — Amazon Keyspaces unterstützt eine Vielzahl integrierter Skalarfunktionen, die Sie in CQL-Anweisungen verwenden können.

Weitere Informationen zu CQL finden Sie unter. CQL-Sprachreferenz für Amazon Keyspaces (für Apache Cassandra) Informationen zu funktionalen Unterschieden mit Apache Cassandra finden Sie unter. the section called “Funktionelle Unterschiede zu Apache Cassandra”

Um CQL-Abfragen auszuführen, können Sie einen der folgenden Schritte ausführen:

- Verwenden Sie den CQL-Editor im. AWS-Managementkonsole
- Verwenden Sie AWS CloudShell und die CQLSH-Erweiterung.
- Benutze einen Client. `cqlsh`
- Verwenden Sie einen für Apache 2.0 lizenzierten Cassandra-Client-Treiber.

Zusätzlich zu CQL können Sie DDL-Operationen (Data Definition Language) in Amazon Keyspaces mithilfe der SDKs und der AWS ausführen. AWS Command Line Interface

Weitere Informationen zur Verwendung dieser Methoden für den Zugriff auf Amazon Keyspaces finden Sie unter. Zugreifen auf Amazon Keyspaces (für Apache Cassandra)

Wie ist Amazon Keyspaces (für Apache Cassandra) im Vergleich zu Apache Cassandra?

Um eine Verbindung zu Amazon Keyspaces herzustellen, können Sie entweder einen öffentlichen oder einen AWS privaten Endpunkt mithilfe von Interface VPC-Endpunkten (AWS PrivateLink) in der Amazon Virtual Private Cloud verwenden. Je nach verwendetem Endpunkt kann Amazon Keyspaces dem Client auf eine der folgenden Arten angezeigt werden.

AWS Verbindung zum Service-Endpunkt

> Dies ist eine Verbindung, die über einen beliebigen öffentlichen Endpunkt hergestellt wird. In diesem Fall erscheint Amazon Keyspaces dem Client als Apache Cassandra 3.11.2-Cluster mit neun Knoten.

Schnittstelle VPC-Endpunktverbindung

> Dies ist eine private Verbindung, die über einen VPC-Schnittstellen-Endpunkt hergestellt wird. In diesem Fall erscheint Amazon Keyspaces dem Client als Apache Cassandra 3.11.2-Cluster mit drei Knoten.

Unabhängig vom Verbindungstyp und der Anzahl der Knoten, die für den Client sichtbar sind, bietet Amazon Keyspaces praktisch unbegrenzten Durchsatz und Speicherplatz. Zu diesem Zweck ordnet Amazon Keyspaces die Knoten Load Balancern zu, die Ihre Abfragen an eine der vielen zugrunde liegenden Speicherpartitionen weiterleiten. Weitere Informationen zu Verbindungen finden Sie unter the section called “Wie funktionieren sie”.

Amazon Keyspaces speichert Daten in Partitionen. Eine Partition ist eine Speicherzuweisung für eine Tabelle, die von Solid-State-Laufwerken (SSDs) unterstützt wird. Amazon Keyspaces repliziert Ihre Daten automatisch innerhalb mehrerer Availability Zones, AWS-Region um Haltbarkeit und hohe Verfügbarkeit zu gewährleisten. Wenn Ihr Durchsatz- oder Speicherbedarf steigt, übernimmt Amazon Keyspaces die Partitionsverwaltung für Sie und stellt automatisch die erforderlichen zusätzlichen Partitionen bereit.

Amazon Keyspaces unterstützt alle häufig verwendeten Cassandra-Datenebenenoperationen, z. B. das Erstellen von Schlüsselräumen und Tabellen, das Lesen von Daten und das Schreiben von Daten. Amazon Keyspaces ist serverlos, sodass Sie keine Server bereitstellen, patchen oder verwalten müssen. Sie müssen auch keine Software installieren, warten oder betreiben. Daher

müssen Sie in Amazon Keyspaces nicht die API-Operationen der Cassandra-Steuerungsebene verwenden, um Cluster- und Knoteneinstellungen zu verwalten.

Amazon Keyspaces konfiguriert automatisch Einstellungen wie den Replikationsfaktor und das Konsistenzniveau, um Ihnen hohe Verfügbarkeit, Beständigkeit und Leistung im einstelligen Millisekundenbereich zu bieten. Für noch mehr Stabilität und lokale Lesevorgänge mit geringer Latenz bietet Amazon Keyspaces Replikation in mehreren Regionen.

Themen

- Funktionale Unterschiede: Amazon Keyspaces im Vergleich zu Apache Cassandra
- Unterstützte Cassandra-APIs, Operationen, Funktionen und Datentypen
- Unterstützte Lese- und Schreibkonsistenzstufen von Apache Cassandra und damit verbundene Kosten

Funktionale Unterschiede: Amazon Keyspaces im Vergleich zu Apache Cassandra

Im Folgenden sind die funktionalen Unterschiede zwischen Amazon Keyspaces und Apache Cassandra aufgeführt.

Themen

- APIs, Operationen und Datentypen von Apache Cassandra
- Asynchrone Erstellung und Löschung von Schlüsselräumen und Tabellen
- Authentifizierung und Autorisierung
- Batch
- Erfassung von Datenänderungen (Change Data Capture, CDC)
- Cluster-Konfiguration
- Verbindungen
- INSchlüsselwort
- FROZENSammlungen
- Einfache Transaktionen
- Lastausgleich
- Paginierung

- Partitionierer
- Vorbereitete Aussagen
- Bereich löschen
- Systemtabellen
- Zeitstempel
- User-defined Typen (UDTs)

APIs, Operationen und Datentypen von Apache Cassandra

Amazon Keyspaces unterstützt alle häufig verwendeten Cassandra-Datenebenenoperationen, z. B. das Erstellen von Schlüsselräumen und Tabellen, das Lesen von Daten und das Schreiben von Daten. Informationen zu den zurzeit unterstützten Komponenten finden Sie unter Unterstützte Cassandra-APIs, Operationen, Funktionen und Datentypen.

Asynchrone Erstellung und Löschung von Schlüsselräumen und Tabellen

Amazon Keyspaces führt DDL-Operationen (Data Definition Language) wie das asynchrone Erstellen und Löschen von Schlüsselräumen, Tabellen und Typen durch. Informationen zur Überwachung des Erstellungsstatus von Ressourcen finden Sie unter und. the section called “Überprüfen Sie den Status der Keyspace-Erstellung” the section called “Überprüfen Sie den Status der Tabellenerstellung” Eine Liste der DDL-Anweisungen in der CQL-Sprachreferenz finden Sie unter. the section called “DDL-Anweisungen”

Authentifizierung und Autorisierung

Amazon Keyspaces (für Apache Cassandra) verwendet AWS Identity and Access Management (IAM) für die Benutzerauthentifizierung und -autorisierung und unterstützt die entsprechenden Autorisierungsrichtlinien wie Apache Cassandra. Daher unterstützt Amazon Keyspaces die Sicherheitskonfigurationsbefehle von Apache Cassandra nicht.

Batch

Amazon Keyspaces unterstützt die folgenden Batch-Befehle:

- Protokollierter Batch — unterstützt bis zu 100 Befehle pro Batch.
- Nicht protokollierter Stapel — unterstützt bis zu 30 Befehle pro Batch.

In einem Batch sind nur INSERT DELETE BefehleUPDATE, oder zulässig.

Erfassung von Datenänderungen (Change Data Capture, CDC)

In Apache Cassandra kopiert Change Data Capture (CDC) Commit-Log-Segmente, die Tabellenaktualisierungen erfassen, in ein bestimmtes Verzeichnis auf dem Knoten. Jeder Änderungsdatensatz im CDC-Protokoll kann Aktualisierungen in mehreren Schlüsselräumen und Tabellen enthalten. Wenn eine Zeile aktualisiert oder gelöscht wird, zeigt das Apache Cassandra CDC-Protokoll nur die geänderten Spalten an. Das Protokoll enthält nicht den vorherigen Status der Daten vor der Änderung. Um die vollständigen Details der Datenänderungen zu verstehen, müssen Sie diese Protokolle mit speziellen Cassandra-Tools deserialisieren.

Mit Amazon Keyspaces CDC-Streams können Sie die Art der Informationen auswählen, die der CDC-Stream für jede Zeile sammelt. Standardmäßig erfassen Amazon Keyspaces CDC-Streams die Version der Zeile vor und nach der Änderung. Amazon Keyspaces CDC stellt deduplizierte und geordnete Änderungsdatensätze auf Tabellenebene bereit.

In Amazon Keyspaces ist jeder CDC-Stream eine AWS Ressource mit einem Amazon-Ressourcennamen (ARN), und Ihre Anwendung kann Amazon Keyspaces-CDC-Streams mithilfe der Amazon Keyspaces CDC-API über einen der verfügbaren Endpunkte konsumieren.

Weitere Informationen zu Amazon Keyspaces CDC finden Sie unter. the section called “Arbeiten mit CDC-Streams”

Cluster-Konfiguration

Amazon Keyspaces ist serverlos, sodass keine Cluster, Hosts oder Java Virtual Machines (JVMs) konfiguriert werden müssen. Cassandras Einstellungen für Komprimierung, Komprimierung, Caching, Garbage-Collection und Bloom-Filterung gelten nicht für Amazon Keyspaces und werden ignoriert, falls angegeben.

Verbindungen

Sie können vorhandene Cassandra-Treiber für die Kommunikation mit Amazon Keyspaces verwenden, müssen die Treiber jedoch anders konfigurieren. Amazon Keyspaces unterstützt bis zu 3.000 CQL-Abfragen pro TCP-Verbindung pro Sekunde, aber die Anzahl der Verbindungen, die ein Treiber herstellen kann, ist unbegrenzt.

Die meisten Open-Source-Cassandra-Treiber richten einen Verbindungspool zu Cassandra ein und verteilen Abfragen über diesen Verbindungspool. Amazon Keyspaces stellt Treibern 9 Peer-

IP-Adressen zur Verfügung, und das Standardverhalten der meisten Treiber besteht darin, eine einzige Verbindung zu jeder Peer-IP-Adresse herzustellen. Daher beträgt der maximale CQL-Abfragedurchsatz eines Treibers, der die Standardeinstellungen verwendet, 27.000 CQL-Abfragen pro Sekunde.

Um diese Zahl zu erhöhen, empfehlen wir, die Anzahl der Verbindungen pro IP-Adresse zu erhöhen, die Ihr Treiber in seinem Verbindungspool verwaltet. Wenn Sie beispielsweise die maximale Anzahl an Verbindungen pro IP-Adresse auf 2 festlegen, verdoppelt sich der maximale Durchsatz Ihres Treibers auf 54.000 CQL-Abfragen pro Sekunde.

Als bewährte Methode empfehlen wir, Treiber so zu konfigurieren, dass sie 500 CQL-Abfragen pro Sekunde und Verbindung verwenden, um Mehraufwand zu vermeiden und die Verteilung zu verbessern. In diesem Szenario sind für die Planung von 18.000 CQL-Abfragen pro Sekunde 36 Verbindungen erforderlich. Die Konfiguration des Treibers für 4 Verbindungen an 9 Endpunkten ermöglicht 36 Verbindungen, die 500 Anfragen pro Sekunde ausführen. Weitere Informationen zu bewährten Methoden für Verbindungen finden Sie unterthe section called “Verbindungen”.

Bei der Verbindung mit VPC-Endpunkten sind möglicherweise weniger Endpunkte verfügbar. Das bedeutet, dass Sie die Anzahl der Verbindungen in der Treiberkonfiguration erhöhen müssen. Weitere Informationen zu bewährten Methoden für VPC-Verbindungen finden Sie unterthe section called “VPC-Endpunktverbindungen”.

`IN`Schlüsselwort

Amazon Keyspaces unterstützt das `IN` Schlüsselwort in der `SELECT` Anweisung. `IN`wird mit `UPDATE` und `DELETE` nicht unterstützt. Wenn das `IN` Schlüsselwort in der `SELECT` Anweisung verwendet wird, werden die Ergebnisse der Abfrage in der Reihenfolge zurückgegeben, in der die Schlüssel in der `SELECT` Anweisung dargestellt werden. In Cassandra sind die Ergebnisse lexikografisch geordnet.

Bei Verwendung `ORDER BY` wird eine vollständige Neuanordnung mit deaktivierter Paginierung nicht unterstützt und die Ergebnisse werden innerhalb einer Seite angeordnet. Slice-Abfragen werden mit dem Schlüsselwort nicht unterstützt. `IN TOKEN`Swerden mit dem `IN` Schlüsselwort nicht unterstützt. Amazon Keyspaces verarbeitet Abfragen mit dem `IN` Schlüsselwort, indem es Unterabfragen erstellt. Jede Unterabfrage zählt als Verbindung im Rahmen der Obergrenze von 3.000 CQL-Abfragen pro TCP-Verbindung pro Sekunde. Weitere Informationen finden Sie unter the section called “Verwenden `IN SELECT`”.

FROZENSammlungen

Das FROZEN Schlüsselwort in Cassandra serialisiert mehrere Komponenten eines Sammlungsdatentyps zu einem einzigen unveränderlichen Wert, der wie ein behandelt wird. BLOB INSERTund UPDATE Anweisungen überschreiben die gesamte Sammlung.

Amazon Keyspaces unterstützt standardmäßig bis zu 8 Verschachtelungsebenen für eingefrorene Sammlungen. Weitere Informationen finden Sie unter the section called “Amazon Keyspaces-Servicekontingente”.

Amazon Keyspaces unterstützt keine Ungleichheitsvergleiche, bei denen die gesamte eingefrorene Sammlung in einer Bedingung UPDATE oder SELECT Anweisung verwendet wird. Das Verhalten für Sammlungen und eingefrorene Sammlungen ist in Amazon Keyspaces dasselbe.

Wenn Sie eingefrorene Sammlungen mit clientseitigen Zeitstempeln verwenden und der Zeitstempel eines Schreibvorgangs dem Zeitstempel einer vorhandenen Spalte entspricht, die nicht abgelaufen oder als veraltet markiert ist, führt Amazon Keyspaces keine Vergleiche durch. Stattdessen kann der Server den neuesten Writer ermitteln, und der letzte Writer gewinnt.

Weitere Informationen zu eingefrorenen Sammlungen finden Sie unterthe section called “Sammlungstypen”.

Einfache Transaktionen

Amazon Keyspaces (für Apache Cassandra) unterstützt vollständig die Funktionen zum Vergleichen und Einstellen von DELETE Befehlen, und INSERTUPDATE, die in Apache Cassandra als Lightweight Transactions (LWTs) bezeichnet werden. Als serverloses Angebot bietet Amazon Keyspaces (für Apache Cassandra) gleichbleibende Leistung in jeder Größenordnung, auch für einfache Transaktionen. Mit Amazon Keyspaces gibt es keine Leistungseinbußen bei der Verwendung einfacher Transaktionen.

Lastausgleich

Die system.peers Tabelleneinträge entsprechen den Load Balancern von Amazon Keyspaces. Um optimale Ergebnisse zu erzielen, empfehlen wir, eine Round-Robin-Load-Balancing-Richtlinie zu verwenden und die Anzahl der Verbindungen pro IP an die Anforderungen Ihrer Anwendung anzupassen.

Paginierung

Amazon Keyspaces paginiert Ergebnisse auf der Grundlage der Anzahl der Zeilen, die es liest, um eine Anfrage zu verarbeiten, und nicht auf der Anzahl der Zeilen, die in der Ergebnismenge zurückgegeben werden. Daher können einige Seiten weniger Zeilen enthalten, als Sie unter SEITENGRÖSSE für gefilterte Abfragen angeben. Darüber hinaus paginiert Amazon Keyspaces die Ergebnisse nach dem Lesen von 1 MB Daten automatisch, um Kunden eine konsistente Leseleistung im einstelligen Millisekundenbereich zu bieten. Weitere Informationen finden Sie unter the section called “Ergebnisse paginieren”.

In Tabellen mit statischen Spalten legen sowohl Apache Cassandra als auch Amazon Keyspaces den statischen Spaltenwert der Partition am Anfang jeder Seite in einer mehrseitigen Abfrage fest. Wenn eine Tabelle aufgrund des Paginierungsverhaltens von Amazon Keyspaces große Datenzeilen enthält, ist die Wahrscheinlichkeit höher, dass das Ergebnis eines Bereichslesevorgangs für Amazon Keyspaces mehr Seiten zurückgibt als für Apache Cassandra. Folglich besteht in Amazon Keyspaces eine höhere Wahrscheinlichkeit, dass gleichzeitige Aktualisierungen der statischen Spalte dazu führen können, dass der Wert der statischen Spalte auf verschiedenen Seiten des Ergebnissatzes für den Bereich Read unterschiedlich ist.

Partitionierer

Die Standardpartitionierung in Amazon Keyspaces ist die. Cassandra-compatible `Murmur3Partitioner` Darüber hinaus haben Sie die Wahl, entweder die Amazon Keyspaces `DefaultPartitioner` oder die Cassandra-compatible `RandomPartitioner` zu verwenden.

Mit Amazon Keyspaces können Sie den Partitionierer für Ihr Konto sicher ändern, ohne Ihre Amazon Keyspaces-Daten neu laden zu müssen. Nach Abschluss der Konfigurationsänderung, die etwa 10 Minuten dauert, sehen Kunden die neue Partitionierungseinstellung automatisch, wenn sie das nächste Mal eine Verbindung herstellen. Weitere Informationen finden Sie unter the section called “Mit Partitionierern arbeiten”.

Vorbereitete Aussagen

Amazon Keyspaces unterstützt die Verwendung von vorbereiteten Anweisungen für DML-Operationen (Data Manipulation Language), z. B. das Lesen und Schreiben von Daten. Amazon Keyspaces unterstützt derzeit nicht die Verwendung von vorbereiteten Anweisungen für DDL-Operationen (Data Definition Language), wie das Erstellen von Tabellen und Schlüsselräumen. DDL-Operationen müssen außerhalb von vorbereiteten Anweisungen ausgeführt werden.

Bereich löschen

> **Important**
>
> Ab dem 1. Juli 2026 werden Vorgänge zum Löschen von Bereichen von der synchronen zur asynchronen Verarbeitung übergehen. Nach dieser Änderung bestätigt eine erfolgreiche Antwort, dass Ihre Löschanforderung akzeptiert wurde, der Löschvorgang jedoch möglicherweise noch im Gange ist. Um zu überprüfen, ob der Löschvorgang abgeschlossen ist, fragen Sie nach Elementen im angegebenen Bereich ab. Wenn die Abfrage keine Ergebnisse zurückgibt, ist der Löschvorgang abgeschlossen.

Amazon Keyspaces unterstützt das Löschen von Zeilen im Bereich. Ein Bereich ist ein zusammenhängender Satz von Zeilen innerhalb einer Partition. Sie geben einen Bereich in einer DELETE-Operation mithilfe einer WHERE-Klausel an. Sie können den Bereich als gesamte Partition angeben.

Darüber hinaus können Sie einen Bereich als Teilmenge zusammenhängender Zeilen innerhalb einer Partition angeben, indem Sie relationale Operatoren verwenden (z. B. '>', '<') oder indem Sie den Partitionsschlüssel angeben und eine oder mehrere Clusterspalten weglassen. Mit Amazon Keyspaces können Sie bis zu 1.000 Zeilen innerhalb eines Bereichs in einem einzigen Vorgang löschen.

Bereichslöschungen sind nicht isoliert. Das Löschen einzelner Zeilen ist für andere Vorgänge sichtbar, während ein Bereich gelöscht wird.

Systemtabellen

Amazon Keyspaces füllt die Systemtabellen, die für die Open-Source-Cassandra-Treiber von Apache 2.0 erforderlich sind. Die Systemtabellen, die für einen Client sichtbar sind, enthalten Informationen, die für den authentifizierten Benutzer eindeutig sind. Die Systemtabellen werden vollständig von Amazon Keyspaces gesteuert und sind schreibgeschützt. Weitere Informationen finden Sie unter the section called “Systemschlüsselräume”.

Read-only Zugriff auf Systemtabellen ist erforderlich, und Sie können ihn mithilfe von IAM-Zugriffsrichtlinien steuern. Weitere Informationen finden Sie unter the section called “Verwalten des Zugriffs mit Richtlinien”. Sie müssen tagbasierte Zugriffskontrollrichtlinien für Systemtabellen unterschiedlich definieren, je nachdem, ob Sie die AWS SDK- oder Cassandra Query Language (CQL) -API-Aufrufe über Cassandra-Treiber und Entwicklertools verwenden. Weitere Informationen

zur tagbasierten Zugriffskontrolle für Systemtabellen finden Sie unter. the section called “Amazon Keyspaces-Ressourcenzugriff basierend auf Tags”

Wenn Sie über Amazon VPC-Endpunkte auf Amazon Keyspaces zugreifen, werden in der `system.peers` Tabelle Einträge für jeden Amazon VPC-Endpunkt angezeigt, für den Amazon Keyspaces Zugriffsberechtigungen besitzt. Infolgedessen gibt Ihr Cassandra-Treiber möglicherweise eine Warnmeldung über den Kontrollknoten selbst in der Tabelle aus. `system.peers` Sie können diese Warnung getrost ignorieren.

Zeitstempel

In Amazon Keyspaces sind Zeitstempel auf Zellenebene, die mit den Standardzeitstempeln in Apache Cassandra kompatibel sind, eine optionale Funktion.

Die `USING TIMESTAMP` Klausel und die `WRITETIME` Funktion sind nur verfügbar, wenn clientseitige Zeitstempel für eine Tabelle aktiviert sind. Weitere Informationen zu clientseitigen Zeitstempeln in Amazon Keyspaces finden Sie unter. the section called “Client-side Zeitstempel”

User-defined Typen (UDTs)

Der Ungleichheitsoperator wird für UDTs in Amazon Keyspaces nicht unterstützt.

Informationen zum Arbeiten mit UDTs in Amazon Keyspaces finden Sie unter. the section called “User-defined Typen (UDTs)”

Informationen darüber, wie viele UDTs pro Schlüsselraum unterstützt werden, welche Verschachtelungsebenen unterstützt werden und welche anderen Standardwerte und Kontingente für UDTs gelten, finden Sie unter. the section called “Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces”

Unterstützte Cassandra-APIs, Operationen, Funktionen und Datentypen

Amazon Keyspaces (für Apache Cassandra) ist mit der Cassandra Query Language (CQL) 3.11-API (abwärtskompatibel mit Version 2.x) kompatibel.

Amazon Keyspaces unterstützt alle häufig verwendeten Cassandra-Datenebenenoperationen, z. B. das Erstellen von Schlüsselräumen und Tabellen, das Lesen von Daten und das Schreiben von Daten.

In den folgenden Abschnitten sind die unterstützten Funktionen aufgeführt.

Themen

- Cassandra-API-Unterstützung
- API-Unterstützung für die Cassandra-Steuerebene
- API-Unterstützung für Cassandra-Datenebene
- Unterstützung der Cassandra-Funktion
- Unterstützung des Cassandra-Datentyps

Cassandra-API-Unterstützung

API-Operation	Unterstützt
`CREATE KEYSPACE`	Ja
`ALTER KEYSPACE`	Ja
`DROP KEYSPACE`	Ja
`CREATE TABLE`	Ja
`ALTER TABLE`	Ja
`DROP TABLE`	Ja
`CREATE INDEX`	Nein
`DROP INDEX`	Nein
`UNLOGGED BATCH`	Ja
`LOGGED BATCH`	Ja
`SELECT`	Ja
`INSERT`	Ja
`DELETE`	Ja

API-Operation	Unterstützt
UPDATE	Ja
USE	Ja
CREATE TYPE	Ja
ALTER TYPE	Nein
DROP TYPE	Ja
CREATE TRIGGER	Nein
DROP TRIGGER	Nein
CREATE FUNCTION	Nein
DROP FUNCTION	Nein
CREATE AGGREGATE	Nein
DROP AGGREGATE	Nein
CREATE MATERIALIZED VIEW	Nein
ALTER MATERIALIZED VIEW	Nein
DROP MATERIALIZED VIEW	Nein
TRUNCATE	Nein

API-Unterstützung für die Cassandra-Steuerebene

Da Amazon Keyspaces verwaltet wird, sind die API-Operationen der Cassandra-Steuerungsebene zur Verwaltung der Cluster- und Knoteneinstellungen nicht erforderlich. Daher sind die folgenden Cassandra-Funktionen nicht anwendbar.

Feature	Grund
Umschalten auf Dauerhafte Schreibvorgänge	Alle Schreibvorgänge sind dauerhaft
Lesen Sie die Reparatureinstellungen	Nicht zutreffend
GC Grace Seconds	Nicht zutreffend
Bloom-Filter-Einstellungen	Nicht zutreffend
Einstellungen für die Komprimierung	Nicht zutreffend
Compression settings (Kompressionseinstellungen)	Nicht zutreffend
Einstellungen für das Zwischenspeichern	Nicht zutreffend
Sicherheitseinstellungen	Ersetzt durch IAM

API-Unterstützung für Cassandra-Datenebene

Funktionsmerkmal	Unterstützt
JSON-Unterstützung für SELECT- und INSERT-Anweisungen	Ja
Statische Spalten	Ja
Time to Live (TTL)	Ja

Unterstützung der Cassandra-Funktion

Weitere Hinweise zu den unterstützten Funktionen finden Sie unterthe section called “Built-in Funktionen”.

Funktion	Unterstützt
`Aggregate` -Funktionen	Nein
`Blob`Konvertierung	Ja
`Cast`	Ja
`Datetime`-Funktionen	Ja
Funktionen zur Zeitumrechnung	Ja
`TimeUuid`-Funktionen	Ja
`Token`	Ja
`User defined functions (UDF)`	Nein
`Uuid`	Ja

Unterstützung des Cassandra-Datentyps

In der folgenden Tabelle sind die Apache Cassandra-Datentypen aufgeführt, die in Amazon Keyspaces unterstützt werden. Weitere Informationen zu Datentypen in Amazon Keyspaces finden Sie unterthe section called “Datentypen”.

Datentyp	Unterstützt
`ascii`	Ja
`bigint`	Ja
`blob`	Ja
`boolean`	Ja
`counter`	Ja
`date`	Ja

Datentyp	Unterstützt
decimal	Ja
double	Ja
float	Ja
frozen	Ja
inet	Ja
int	Ja
list	Ja
map	Ja
set	Ja
smallint	Ja
text	Ja
time	Ja
timestamp	Ja
timeuuid	Ja
tinyint	Ja
tuple	Ja
user-defined types (UDTs)	Ja
uuid	Ja
varchar	Ja
varint	Ja

Unterstützte Lese- und Schreibkonsistenzstufen von Apache Cassandra und damit verbundene Kosten

Die Themen in diesem Abschnitt beschreiben, welche Apache Cassandra-Konsistenzstufen für Lese- und Schreibvorgänge in Amazon Keyspaces (für Apache Cassandra) unterstützt werden.

Themen

- Konsistenzstufen für Schreibvorgänge
- Lesen Sie die Konsistenzstufen
- Konsistenzstufen werden nicht unterstützt

Konsistenzstufen für Schreibvorgänge

Amazon Keyspaces repliziert alle Schreibvorgänge dreimal in mehreren Availability Zones, um Haltbarkeit und hohe Verfügbarkeit zu gewährleisten. Schreibvorgänge werden dauerhaft gespeichert, bevor sie anhand der Konsistenzstufe bestätigt werden. `LOCAL_QUORUM` Für jeden Schreibvorgang mit 1 KB wird Ihnen 1 Schreibkapazitätseinheit (WCU) für Tabellen im Bereitstellungsmodus oder 1 Schreibanforderungseinheit (WRU) für Tabellen im On-Demand-Modus in Rechnung gestellt.

Sie können `cqlsh` damit die Konsistenz für alle Abfragen in der aktuellen Sitzung so einstellen, dass sie den folgenden Code `LOCAL_QUORUM` verwenden.

```
CONSISTENCY LOCAL_QUORUM;
```

Um die Konsistenzstufe programmgesteuert zu konfigurieren, können Sie die Konsistenz mit den entsprechenden Cassandra-Clienttreibern festlegen. Mit den Java-Treibern der Version 4.x können Sie beispielsweise die Konsistenzstufe in der `app config` Datei wie unten gezeigt festlegen.

```
basic.request.consistency = LOCAL_QUORUM
```

Wenn Sie einen Java-Cassandra-Treiber der Version 3.x verwenden, können Sie die Konsistenzstufe für die Sitzung angeben, indem Sie sie hinzufügen, `.withQueryOptions(new QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)` wie im folgenden Codebeispiel gezeigt.

```
Session session = Cluster.builder()
                      .addContactPoint(endPoint)
                      .withPort(portNumber)
                      .withAuthProvider(new SigV4AuthProvider("us-east-1"))
                      .withSSL()
                      .withQueryOptions(new
 QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
                      .build()
                      .connect();
```

Um die Konsistenzstufe für bestimmte Schreiboperationen zu konfigurieren, können Sie die Konsistenz definieren, wenn Sie bei der Verwendung des Java-Treibers `QueryBuilder.insertInto` mit einem `setConsistencyLevel` Argument aufrufen.

Lesen Sie die Konsistenzstufen

Amazon Keyspaces unterstützt drei Lesekonsistenzstufen: `ONE`, `LOCAL_ONE` und `LOCAL_QUORUM`. Während eines `LOCAL_QUORUM` Lesevorgangs gibt Amazon Keyspaces eine Antwort zurück, die die neuesten Aktualisierungen aller vorherigen erfolgreichen Schreibvorgänge wiedergibt. Die Verwendung der Konsistenzstufe `ONE` oder `LOCAL_ONE` kann die Leistung und Verfügbarkeit Ihrer Leseanfragen verbessern, aber die Antwort spiegelt möglicherweise nicht die Ergebnisse eines kürzlich abgeschlossenen Schreibvorgangs wider.

Pro 4 KB Lesegeschwindigkeit `ONE` oder `LOCAL_ONE` Konsistenz werden Ihnen 0,5 Lesekapazitätseinheiten (RCUs) für Tabellen mit bereitgestellter Kapazität oder 0,5 Leseanforderungseinheiten (RRUs) für Tabellen, die den On-Demand-Modus verwenden, in Rechnung gestellt. Für jeden 4 KB Lesevorgang im `LOCAL_QUORUM` Konsistenzmodus wird Ihnen 1 Lesekapazitätseinheit (RCU) für Tabellen im Bereitstellungskapazitätsmodus oder 1 Leseanforderungseinheit (RRU) für Tabellen im On-Demand-Modus in Rechnung gestellt.

Die Abrechnung basiert auf Lesekonsistenz und Lesekapazitätsdurchsatzmodus pro Tabelle für jeweils 4 KB Lesevorgänge

Konsistenzgrad	Bereitgestellt	On-demand
`ONE`	0,5 CCUs	0,5 RUs
`LOCAL_ONE`	0,5 RCUs	0,5 RUs
`LOCAL_QUORUM`	1 RCU	1 RRU

Um eine andere Konsistenz für Lesevorgänge anzugeben, rufen Sie `QueryBuilder.select` mit einem `setConsistencyLevel` Argument auf, wenn Sie den Java-Treiber verwenden.

Konsistenzstufen werden nicht unterstützt

Die folgenden Konsistenzstufen werden von Amazon Keyspaces nicht unterstützt und führen zu Ausnahmen.

Konsistenzstufen werden nicht unterstützt

Apache Cassandra	Amazon Keyspaces
`EACH_QUORUM`	Nicht unterstützt
`QUORUM`	Nicht unterstützt
`ALL`	Nicht unterstützt
`TWO`	Nicht unterstützt
`THREE`	Nicht unterstützt
`ANY`	Nicht unterstützt
`SERIAL`	Nicht unterstützt
`LOCAL_SERIAL`	Nicht unterstützt

Migration zu Amazon Keyspaces (für Apache Cassandra)

Die Migration zu Amazon Keyspaces (für Apache Cassandra) bietet eine Reihe überzeugender Vorteile für Unternehmen und Organisationen. Hier sind einige wichtige Vorteile, die Amazon Keyspaces zu einer attraktiven Wahl für die Migration machen.

- Skalierbarkeit — Amazon Keyspaces ist darauf ausgelegt, enorme Workloads zu bewältigen und sich nahtlos skalieren zu lassen, um wachsenden Datenmengen und Datenströmen gerecht zu werden. Bei herkömmlichem Cassandra erfolgt die Skalierung nicht bei Bedarf und erfordert die Planung zukünftiger Spitzenwerte. Mit Amazon Keyspaces können Sie Ihre Tabellen je nach Bedarf einfach nach oben oder unten skalieren und so sicherstellen, dass Ihre Anwendungen plötzliche Datenverkehrsspitzen ohne Leistungseinbußen bewältigen können.
- Leistung — Amazon Keyspaces bietet Datenzugriff mit niedriger Latenz, sodass Anwendungen Daten mit außergewöhnlicher Geschwindigkeit abrufen und verarbeiten können. Die verteilte Architektur stellt sicher, dass Lese- und Schreibvorgänge auf mehrere Knoten verteilt werden, sodass selbst bei hohen Anforderungsraten konsistente Reaktionszeiten im einstelligen Millisekundenbereich erzielt werden.
- Vollständig verwaltet — Amazon Keyspaces ist ein vollständig verwalteter Service von AWS. Das bedeutet, dass er sich um die betrieblichen Aspekte der Datenbankverwaltung AWS kümmert, einschließlich Bereitstellung, Konfiguration, Patching, Backups und Skalierung. Auf diese Weise können Sie sich stärker auf die Entwicklung von Anwendungen als auf Datenbankverwaltungsaufgaben konzentrieren.
- Serverlose Architektur — Amazon Keyspaces ist serverlos. Sie zahlen nur für die verbrauchte Kapazität, ohne dass eine Vorauszahlung der Kapazität erforderlich ist. Sie müssen weder Server verwalten noch Instanzen auswählen. Dieses Pay-per-Request-Modell bietet Kosteneffizienz und minimalen Betriebsaufwand, da nur die verwendeten Ressourcen berechnet werden, ohne dass Kapazitäten bereitgestellt und überwacht werden müssen.
- NoSQL-Flexibilität mit Schema — Amazon Keyspaces folgt einem NoSQL-Datenmodell, das Flexibilität beim Schemadesign bietet. Mit Amazon Keyspaces können Sie strukturierte, halbstrukturierte und unstrukturierte Daten speichern, sodass sie sich gut für den Umgang mit unterschiedlichen und sich weiterentwickelnden Datentypen eignen. Darüber hinaus führt Amazon Keyspaces beim Schreiben eine Schemavalidierung durch, was eine zentralisierte Weiterentwicklung des Datenmodells ermöglicht. Diese Flexibilität ermöglicht schnellere Entwicklungszyklen und eine einfachere Anpassung an sich ändernde Geschäftsanforderungen.

- Hohe Verfügbarkeit und Beständigkeit — Amazon Keyspaces repliziert Daten über mehrere Availability Zones innerhalb einer und gewährleistet so eine AWS-Region hohe Verfügbarkeit und Datenbeständigkeit. Replikation, Failover und Wiederherstellung von Daten werden automatisch übernommen, wodurch sich das Risiko von Datenverlusten oder Serviceunterbrechungen minimiert. Amazon Keyspaces bietet eine Verfügbarkeits-SLA von bis zu 99,999% Für noch mehr Stabilität und lokale Lesevorgänge mit geringer Latenz bietet Amazon Keyspaces Replikation in mehreren Regionen.
- Sicherheit und Compliance — Amazon Keyspaces lässt sich AWS Identity and Access Management für eine differenzierte Zugriffskontrolle integrieren. Es bietet Verschlüsselung im Ruhezustand und bei der Übertragung und trägt so zur Verbesserung der Sicherheit Ihrer Daten bei. Amazon Keyspaces wurde von externen Prüfern auf Sicherheit und Einhaltung bestimmter Programme wie HIPAA, PCI DSS und SOC geprüft, sodass Sie die gesetzlichen Anforderungen erfüllen können. Weitere Informationen finden Sie unter the section called “Compliance-Validierung”.
- Integration mit dem AWS Ökosystem — Als Teil des AWS Ökosystems lässt sich Amazon Keyspaces nahtlos in andere AWS-Services Systeme integrieren AWS CloudFormation, beispielsweise Amazon CloudWatch und AWS CloudTrail. Diese Integration ermöglicht es Ihnen, Serverless-Architekturen zu erstellen, Infrastruktur als Code zu nutzen und datengesteuerte Echtzeitanwendungen zu entwickeln. Weitere Informationen finden Sie unter Überwachung von Amazon Keyspaces.

Themen

- Erstellen Sie einen Migrationsplan für die Migration von Apache Cassandra zu Amazon Keyspaces
- So wählen Sie das richtige Tool für den Massen-Upload oder die Migration von Daten zu Amazon Keyspaces aus

Erstellen Sie einen Migrationsplan für die Migration von Apache Cassandra zu Amazon Keyspaces

Für eine erfolgreiche Migration von Apache Cassandra zu Amazon Keyspaces empfehlen wir eine Überprüfung der geltenden Migrationskonzepte und Best Practices sowie einen Vergleich der verfügbaren Optionen.

In diesem Thema wird beschrieben, wie der Migrationsprozess funktioniert, indem mehrere wichtige Konzepte sowie die Ihnen zur Verfügung stehenden Tools und Techniken vorgestellt werden. Sie

können die verschiedenen Migrationsstrategien evaluieren, um diejenige auszuwählen, die Ihren Anforderungen am besten entspricht.

Themen

- Funktionale Kompatibilität
- Schätzen Sie die Preise für Amazon Keyspaces
- Wählen Sie eine Migrationsstrategie
- Online-Migration zu Amazon Keyspaces: Strategien und bewährte Verfahren
- Offline-Migrationsprozess: Apache Cassandra zu Amazon Keyspaces
- Verwendung einer hybriden Migrationslösung: Apache Cassandra zu Amazon Keyspaces

Funktionale Kompatibilität

Beachten Sie vor der Migration sorgfältig die funktionalen Unterschiede zwischen Apache Cassandra und Amazon Keyspaces. Amazon Keyspaces unterstützt alle häufig verwendeten Cassandra-Datenebenenoperationen, z. B. das Erstellen von Schlüsselräumen und Tabellen, das Lesen von Daten und das Schreiben von Daten.

Es gibt jedoch einige Cassandra-APIs, die Amazon Keyspaces nicht unterstützt. Weitere Informationen zu unterstützten APIs finden Sie unter. the section called “Unterstützte Cassandra-APIs, Operationen, Funktionen und Datentypen” Einen Überblick über alle funktionalen Unterschiede zwischen Amazon Keyspaces und Apache Cassandra finden Sie unter. the section called “Funktionelle Unterschiede zu Apache Cassandra”

Um die von Ihnen verwendeten Cassandra-APIs und das Cassandra-Schema mit den unterstützten Funktionen in Amazon Keyspaces zu vergleichen, können Sie ein Kompatibilitätsskript ausführen, das im Amazon Keyspaces-Toolkit unter verfügbar ist. GitHub

Wie verwende ich das Kompatibilitätsskript

1. Laden Sie das Kompatibilitäts-Python-Skript von herunter GitHub und verschieben Sie es an einen Ort, der Zugriff auf Ihren vorhandenen Apache Cassandra-Cluster hat.
2. Das Kompatibilitätsskript verwendet ähnliche Parameter wie`CQLSH`. `--port`Geben Sie die IP-Adresse `--host` und den Port ein, über den Sie eine Verbindung herstellen und Abfragen an einen der Cassandra-Knoten in Ihrem Cluster ausführen.

Wenn Ihr Cassandra-Cluster die Authentifizierung verwendet, müssen Sie auch und angeben`-username`. `-password` Um das Kompatibilitätsskript auszuführen, können Sie den folgenden Befehl verwenden.

```
python toolkit-compat-tool.py --host hostname or IP -u "username" -p "password" --
port native transport port
```

Schätzen Sie die Preise für Amazon Keyspaces

Dieser Abschnitt bietet einen Überblick über die Informationen, die Sie aus Ihren Apache Cassandra-Tabellen abrufen müssen, um die geschätzten Kosten für Amazon Keyspaces zu berechnen. Jede Ihrer Tabellen benötigt unterschiedliche Datentypen, muss unterschiedliche CQL-Abfragen unterstützen und sorgt für einen bestimmten Datenverkehr. read/write

Wenn Sie Ihre Anforderungen auf der Grundlage von Tabellen berücksichtigen, entspricht dies den Ressourcenisolations- und Durchsatzkapazitätsmodi von Amazon Keyspaces auf Tabellenebene. read/write Mit Amazon Keyspaces können Sie read/write Kapazitäts- und automatische Skalierungsrichtlinien für Tabellen unabhängig voneinander definieren.

Wenn Sie die Tabellenanforderungen verstehen, können Sie Tabellen für die Migration auf der Grundlage von Funktionalität, Kosten und Migrationsaufwand priorisieren.

Erfassen Sie vor einer Migration die folgenden Messwerte der Cassandra-Tabelle. Diese Informationen helfen bei der Schätzung der Kosten Ihres Workloads auf Amazon Keyspaces.

- Tabellenname — Der Name des vollqualifizierten Schlüsselraums und der Tabellenname.
- Beschreibung — Eine Beschreibung der Tabelle, z. B. wie sie verwendet wird oder welche Art von Daten darin gespeichert sind.
- Durchschnittliche Lesevorgänge pro Sekunde — Die durchschnittliche Anzahl der Lesevorgänge für die Tabelle auf Koordinatenebene über ein großes Zeitintervall.
- Durchschnittliche Schreibvorgänge pro Sekunde — Die durchschnittliche Anzahl von Schreibvorgängen auf Koordinatenebene für die Tabelle über ein großes Zeitintervall.
- Durchschnittliche Zeilengröße in Byte — Die durchschnittliche Zeilengröße in Byte.
- Speichergröße in GB — Die unformatierte Speichergröße für eine Tabelle.

- Aufschlüsselung der Lesekonsistenz — Der Prozentsatz der Lesevorgänge, bei denen die Endkonsistenz (`LOCAL_ONE`oder`ONE`) verwendet wird, im Vergleich zu starker Konsistenz (`LOCAL_QUORUM`).

Diese Tabelle zeigt ein Beispiel für die Informationen zu Ihren Tabellen, die Sie bei der Planung einer Migration zusammenstellen müssen.

Name der Tabelle	Description	Durchschnittliche Lesevorgänge pro Sekunde	Durchschnittliche Schreibvorgänge pro Sekunde	Durchschnittliche Zeilengröße in Byte	Speichergröße in GB	Lesen Sie die Konsistenzübersicht
mykeyspace.mytable	Wird verwendet, um den Verlauf des Einkaufswagens zu speichern	10.000	5,000	2.200	2.000	100% `LOCAL_ONE`
mykeyspace.mytable2	Wird verwendet, um die neuesten Profilinformationen zu speichern	20 000	1.000	850	1.000	25% `LOCAL_QUORUM` 75% `LOCAL_ONE`

Wie sammelt man Tabellenkennzahlen

Dieser Abschnitt enthält schrittweise Anweisungen zum Erfassen der erforderlichen Tabellenmetriken aus Ihrem vorhandenen Cassandra-Cluster. Zu diesen Metriken gehören Zeilengröße, Tabellengröße und read/write Anfragen pro Sekunde (RPS). Sie ermöglichen es Ihnen, die

Durchsatzkapazitätsanforderungen für eine Amazon Keyspaces-Tabelle zu beurteilen und die Preise abzuschätzen.

Wie erfasst man Tabellenmetriken in der Cassandra-Quelltabelle

1. Ermitteln Sie die Zeilengröße

   Die Zeilengröße ist wichtig für die Bestimmung der Lese- und Schreibkapazitätsauslastung in Amazon Keyspaces. Das folgende Diagramm zeigt die typische Datenverteilung über einen Cassandra-Token-Bereich.



   Sie können ein Sampler-Skript für die Zeilengröße verwenden, das auf verfügbar ist, GitHub um die Zeilengrößenmetriken für jede Tabelle in Ihrem Cassandra-Cluster zu erfassen.

   Das Skript exportiert Tabellendaten aus Apache Cassandra, indem es die Mindest `cqlsh -awk`, Maximal-, Durchschnitts- und Standardabweichung der Zeilengröße anhand eines konfigurierbaren Stichprobensatzes von Tabellendaten verwendet und berechnet. Der Zeilengrößen-Sampler übergibt die Argumente an`cqlsh`, sodass dieselben Parameter verwendet werden können, um eine Verbindung herzustellen und aus Ihrem Cassandra-Cluster zu lesen.

   Die folgende Aussage ist ein Beispiel dafür.

```
./row-size-sampler.sh 10.22.33.44 9142 \\
   -u "username" -p "password" --ssl
```

   Weitere Informationen zur Berechnung der Zeilengröße in Amazon Keyspaces finden Sie unterthe section called “Schätzen Sie die Zeilengröße”.

2. Ermitteln Sie die Tabellengröße

   Mit Amazon Keyspaces müssen Sie keinen Speicherplatz im Voraus bereitstellen. Amazon Keyspaces überwacht kontinuierlich die fakturierbare Größe Ihrer Tabellen, um Ihre Speichergebühren zu ermitteln. Der Speicherplatz wird pro abgerechnet. GB-month Die Größe der Amazon Keyspaces-Tabelle basiert auf der Rohgröße (unkomprimiert) eines einzelnen Replikats.

   Um die Tabellengröße in Amazon Keyspaces zu überwachen, können Sie die Metrik verwenden`BillableTableSizeInBytes`, die für jede Tabelle in der angezeigt wird. AWS-Managementkonsole

   Um die fakturierbare Größe Ihrer Amazon Keyspaces-Tabelle abzuschätzen, können Sie eine der beiden Methoden verwenden:

   - Verwenden Sie die durchschnittliche Zeilengröße und multiplizieren Sie sie mit der Anzahl der Zeilen.

     Sie können die Größe der Amazon Keyspaces-Tabelle schätzen, indem Sie die durchschnittliche Zeilengröße mit der Anzahl der Zeilen aus Ihrer Cassandra-Quelltabelle multiplizieren. Verwenden Sie das Beispielskript für die Zeilengröße aus dem vorherigen Abschnitt, um die durchschnittliche Zeilengröße zu erfassen. Um die Zeilenanzahl zu erfassen, können Sie Tools verwenden, `dsbulk count` um beispielsweise die Gesamtzahl der Zeilen in Ihrer Quelltabelle zu ermitteln.
   - Verwenden Sie das`nodetool`, um Tabellenmetadaten zu sammeln.

     `Nodetool`ist ein in der Apache Cassandra-Distribution bereitgestelltes Verwaltungstool, das Einblick in den Status des Cassandra-Prozesses bietet und Tabellenmetadaten zurückgibt. Sie können `nodetool` es verwenden, um Metadaten über die Tabellengröße abzufragen und damit die Tabellengröße in Amazon Keyspaces zu extrapolieren.

     Der zu verwendende Befehl lautet. `nodetool tablestats` Tablestats gibt die Größe und das Komprimierungsverhältnis der Tabelle zurück. Die Größe der Tabelle wird als `tablelivespace` für die Tabelle gespeichert und Sie können sie durch die `compression ratio` dividieren. Multiplizieren Sie dann diesen Größenwert mit der Anzahl der Knoten. Dividieren Sie schließlich durch den Replikationsfaktor (normalerweise drei).

     Dies ist die vollständige Formel für die Berechnung, anhand derer Sie die Tabellengröße beurteilen können.

```
((tablelivespace / compression ratio) * (total number of nodes))/ (replication
 factor)
```

Nehmen wir an, Ihr Cassandra-Cluster hat 12 Knoten. Die Ausführung des `nodetool tablestats` Befehls gibt a `tablelivespace` von 200 GB und a `compression ratio` von 0,5 zurück. Der Schlüsselraum hat einen Replikationsfaktor von drei.

So sieht die Berechnung für dieses Beispiel aus.

```
(200 GB / 0.5) * (12 nodes)/ (replication factor of 3)
                        = 4,800 GB / 3
                        = 1,600 GB is the table size estimate for Amazon
 Keyspaces
```

3. Erfassen Sie die Anzahl der Lese- und Schreibvorgänge

Um die Kapazitäts- und Skalierungsanforderungen für Ihre Amazon Keyspaces-Tabellen zu ermitteln, erfassen Sie vor der Migration die Lese- und Schreibanforderungsrate Ihrer Cassandra-Tabellen.

Amazon Keyspaces ist serverlos und Sie zahlen nur für das, was Sie nutzen. Im Allgemeinen basiert der Preis für den read/write Durchsatz in Amazon Keyspaces auf der Anzahl und Größe der Anfragen.

In Amazon Keyspaces gibt es zwei Kapazitätsmodi:

- On-demand— Dies ist eine flexible Abrechnungsoption, mit der Tausende von Anfragen pro Sekunde bedient werden können, ohne dass eine Kapazitätsplanung erforderlich ist. Sie bietet für Lese- und Schreibanfragen eine Abrechnung pro Anfrage, sodass Sie nur für das bezahlen, was Sie tatsächlich nutzen.
- Bereitgestellt — Wenn Sie den Modus für bereitgestellte Durchsatzkapazität wählen, geben Sie die Anzahl der Lese- und Schreibvorgänge pro Sekunde an, die für Ihre Anwendung erforderlich sind. Dies hilft Ihnen dabei, Ihre Amazon Keyspaces-Nutzung so zu verwalten, dass sie bei oder unter einer definierten Anforderungsrate bleibt, um die Vorhersagbarkeit zu gewährleisten.

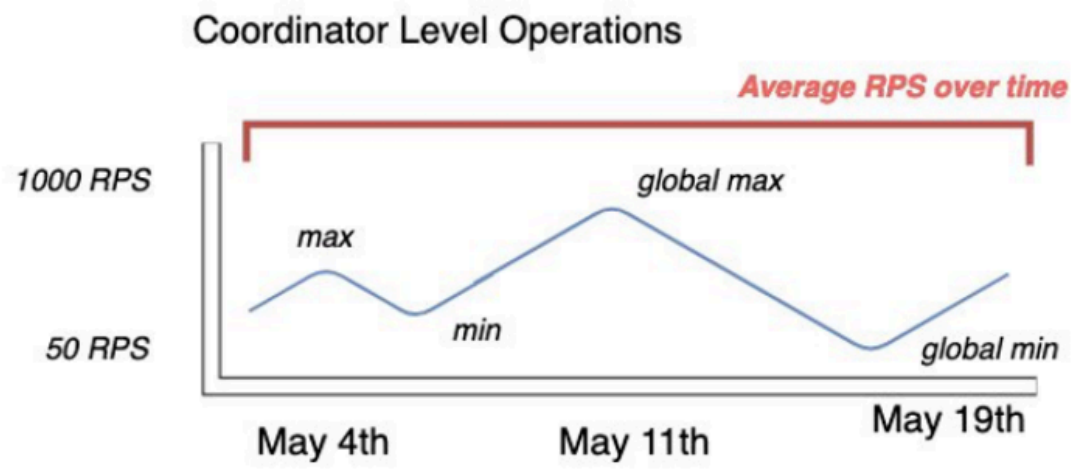
  Der Bereitstellungsmodus bietet eine automatische Skalierung, sodass Ihre bereitgestellte Rate automatisch angepasst wird, um die betriebliche Effizienz zu erhöhen oder zu verringern.

Weitere Informationen zur serverlosen Ressourcenverwaltung finden Sie unter. Verwaltung serverloser Ressourcen

Da Sie die Lese- und Schreibdurchsatzkapazität in Amazon Keyspaces separat bereitstellen, müssen Sie die Anforderungsrate für Lese- und Schreibvorgänge in Ihren vorhandenen Tabellen unabhängig voneinander messen.

Um die genauesten Nutzungsmetriken aus Ihrem vorhandenen Cassandra-Cluster zu ermitteln, erfassen Sie die durchschnittlichen Anforderungen pro Sekunde (RPS) für Lese- und Schreibvorgänge auf Koordinatorenebene über einen längeren Zeitraum für eine Tabelle, die über alle Knoten in einem einzigen Rechenzentrum aggregiert ist.

Durch die Erfassung des durchschnittlichen RPS über einen Zeitraum von mindestens mehreren Wochen werden Spitzen und Täler in Ihren Verkehrsmustern erfasst, wie im folgenden Diagramm dargestellt.



Sie haben zwei Möglichkeiten, die Lese- und Schreibanforderungsrate Ihrer Cassandra-Tabelle zu bestimmen.

- Verwenden Sie das vorhandene Cassandra-Monitoring

  Sie können die in der folgenden Tabelle aufgeführten Metriken verwenden, um Lese- und Schreibanforderungen zu beobachten. Beachten Sie, dass sich die Metriknamen je nach verwendetem Überwachungstool ändern können.

Dimension	Cassandra JMX-Metrik
Writes	`org.apache.cassandra.metrics:type=ClientRequest,scope=Write,name=Latency#Count`
Reads	`org.apache.cassandra.metrics:type=ClientRequest,scope=Read,name=Latency#Count`

- Verwenden der `nodetool`

  Verwenden Sie `nodetool tablestats` und`nodetool info`, um durchschnittliche Lese- und Schreibvorgänge aus der Tabelle zu erfassen. `tablestats`gibt die Gesamtzahl der Lese- und Schreibvorgänge ab dem Zeitpunkt zurück, zu dem der Knoten initiiert wurde. `nodetool info`gibt die Betriebszeit für einen Knoten in Sekunden an.

  Um den Durchschnitt der Lese- und Schreibvorgänge pro Sekunde zu erhalten, dividieren Sie die Anzahl der Lese- und Schreibvorgänge durch die Knotenbetriebszeit in Sekunden. Dann dividieren Sie für Lesevorgänge durch das Konsistenzniveau und für Schreibvorgänge dividieren Sie durch den Replikationsfaktor. Diese Berechnungen werden in den folgenden Formeln ausgedrückt.

  Formel für durchschnittliche Lesevorgänge pro Sekunde:

```
((number of reads * number of nodes in cluster) / read consistency quorum
 (2)) / uptime
```

  Formel für durchschnittliche Schreibvorgänge pro Sekunde:

```
((number of writes * number of nodes in cluster) / replication factor of 3) /
 uptime
```

  Nehmen wir an, wir haben einen Cluster mit 12 Knoten, der seit 4 Wochen aktiv ist. `nodetool info`gibt 2.419.200 Sekunden Betriebszeit zurück und `nodetool`

`tablestats` gibt 1 Milliarde Schreibvorgänge und 2 Milliarden Lesevorgänge zurück. Dieses Beispiel würde zu der folgenden Berechnung führen.

```
((2 billion reads * 12 in cluster) / read consistency quorum (2)) / 2,419,200
 seconds
=  12 billion reads / 2,419,200 seconds
=  4,960 read request per second
                        ((1 billion writes * 12 in cluster) / replication
 factor of 3) / 2,419,200 seconds
=  4 billion writes / 2,419,200 seconds
=  1,653 write request per second
```

4. Ermitteln Sie die Kapazitätsauslastung der Tabelle

   Um die durchschnittliche Kapazitätsauslastung abzuschätzen, beginnen Sie mit den durchschnittlichen Anforderungsraten und der durchschnittlichen Zeilengröße Ihrer Cassandra-Quelltabelle.

   Amazon Keyspaces verwendet Lesekapazitätseinheiten (RCUs) und Schreibkapazitätseinheiten (WCUs), um die bereitgestellte Durchsatzkapazität für Lese- und Schreibvorgänge für Tabellen zu messen. Für diese Schätzung verwenden wir diese Einheiten, um den Lese- und Schreibkapazitätsbedarf der neuen Amazon Keyspaces-Tabelle nach der Migration zu berechnen.

   Später in diesem Thema werden wir erörtern, wie sich die Wahl zwischen bereitgestelltem und bedarfsorientiertem Kapazitätsmodus auf die Abrechnung auswirkt. Für die Schätzung der Kapazitätsauslastung in diesem Beispiel gehen wir jedoch davon aus, dass sich die Tabelle im Bereitstellungsmodus befindet.

   - Lesevorgänge — Eine RCU steht für eine `LOCAL_QUORUM` Leseanforderung oder zwei `LOCAL_ONE` Leseanforderungen für eine Zeile mit einer Größe von bis zu 4 KB. Wenn Sie eine Zeile lesen müssen, die größer als 4 KB ist, verwendet der Lesevorgang zusätzliche RCU. Die Gesamtzahl der benötigten RCUs hängt von der Zeilengröße ab und davon, ob Sie Consistency verwenden `LOCAL_QUORUM` oder `LOCAL_ONE` lesen möchten.

     Zum Lesen einer 8-KB-Zeile sind beispielsweise 2 RCU mit `LOCAL_QUORUM` Lesekonsistenz erforderlich, und 1 RCU, wenn Sie Lesekonsistenz wählen`LOCAL_ONE`.
   - Schreibvorgänge — Eine WCU steht für einen Schreibvorgang für eine Zeile mit einer Größe von bis zu 1 KB. Bei allen Schreibvorgängen wird `LOCAL_QUORUM` Konsistenz verwendet, und

für die Verwendung von Lightweight Transactions (LWTs) fallen keine zusätzlichen Gebühren an.

Die Gesamtzahl der benötigten WCUs hängt von der Zeilengröße ab. Wenn Sie eine Zeile schreiben müssen, die größer als 1 KB ist, verwendet der Schreibvorgang zusätzliche WCUs. Wenn Ihre Zeilengröße beispielsweise 2 KB beträgt, benötigen Sie 2 WCUs, um eine Schreibanforderung auszuführen.

Die folgende Formel kann verwendet werden, um die erforderlichen RCUs und WCUs zu schätzen.

- Die Lesekapazität in RCUs kann bestimmt werden, indem die Lesevorgänge pro Sekunde mit der Anzahl der pro Lesevorgang gelesenen Zeilen multipliziert und mit der durchschnittlichen Zeilengröße geteilt durch 4 KB multipliziert und auf die nächste ganze Zahl aufgerundet werden.
- Die Schreibkapazität in WCUs kann bestimmt werden, indem die Anzahl der Anfragen mit der durchschnittlichen Zeilengröße geteilt durch 1 KB multipliziert und auf die nächste ganze Zahl aufgerundet wird.

Dies wird in den folgenden Formeln ausgedrückt.

```
Read requests per second * ROUNDUP((Average Row Size)/4096 per unit) =  RCUs per
 second

Write requests per second * ROUNDUP(Average Row Size/1024 per unit) = WCUs per
 second
```

Wenn Sie beispielsweise 4.960 Leseanforderungen mit einer Zeilengröße von 2,5 KB in Ihrer Cassandra-Tabelle ausführen, benötigen Sie 4.960 RCUs in Amazon Keyspaces. Wenn Sie derzeit 1.653 Schreibanforderungen pro Sekunde mit einer Zeilengröße von 2,5 KB in Ihrer Cassandra-Tabelle ausführen, benötigen Sie 4.959 WCUs pro Sekunde in Amazon Keyspaces.

Dieses Beispiel wird in den folgenden Formeln ausgedrückt.

```
4,960 read requests per second * ROUNDUP( 2.5KB /4KB bytes per unit)
= 4,960 read requests per second * 1 RCU
= 4,960 RCUs
```

```
1,653 write requests per second * ROUNDUP(2.5KB/1KB per unit)
= 1,653 requests per second * 3 WCUs
= 4,959 WCUs
```

`eventual consistency`Mithilfe können Sie bei jeder Leseanforderung bis zur Hälfte der Durchsatzkapazität einsparen. Jeder letztendlich konsistente Lesevorgang kann bis zu 8 KB verbrauchen. Sie können mögliche konsistente Lesevorgänge berechnen, indem Sie die vorherige Berechnung mit 0,5 multiplizieren, wie in der folgenden Formel dargestellt.

```
4,960 read requests per second * ROUNDUP( 2.5KB /4KB per unit) * .5
= 2,480 read request per second * 1 RCU
= 2,480 RCUs
```

5. Berechnen Sie die monatliche Preisschätzung für Amazon Keyspaces

   Um die monatliche Abrechnung für die Tabelle auf der Grundlage des read/write Kapazitätsdurchsatzes abzuschätzen, können Sie die Preise für den On-Demand- und den Bereitstellungsmodus mithilfe verschiedener Formeln berechnen und die Optionen für Ihre Tabelle vergleichen.

   Bereitgestellter Modus — Der Kapazitätsverbrauch für Lese- und Schreibvorgänge wird auf der Grundlage der Kapazitätseinheiten pro Sekunde auf Stundenbasis abgerechnet. Teilen Sie diese Rate zunächst durch 0,7, um der standardmäßigen Autoscaling-Zielauslastung von 70% zu entsprechen. Multiplizieren Sie dann mit 30 Kalendertagen, 24 Stunden pro Tag und regionaler Preisgestaltung.

   Diese Berechnung wird in den folgenden Formeln zusammengefasst.

   ```
   (read capacity per second / .7) * 24 hours * 30 days * regional rate
                  (write capacity per second / .7) * 24 hours * 30 days * regional
    rate
   ```

   On-demand Modus — Die Lese- und Schreibkapazität wird pro Anforderung abgerechnet. Multiplizieren Sie zunächst die Anforderungsrate mit 30 Kalendertagen und 24 Stunden pro Tag. Teilen Sie dann durch eine Million Anforderungseinheiten. Schließlich multiplizieren Sie mit der regionalen Rate.

   Diese Berechnung ist in den folgenden Formeln zusammengefasst.

```
((read capacity per second * 30 * 24 * 60 * 60) / 1 Million read request units) *
 regional rate
               ((write capacity per second * 30 * 24 * 60 * 60) / 1 Million write
 request units) * regional rate
```

Wählen Sie eine Migrationsstrategie

Bei der Migration von Apache Cassandra zu Amazon Keyspaces können Sie zwischen den folgenden Migrationsstrategien wählen:

- Online — Dies ist eine Live-Migration, bei der duale Schreibvorgänge verwendet werden, um gleichzeitig mit dem Schreiben neuer Daten in Amazon Keyspaces und den Cassandra-Cluster zu beginnen. Dieser Migrationstyp wird für Anwendungen empfohlen, die während der Migration keine Ausfallzeiten und eine konsistente Lese-nach-Schreib-Verarbeitung erfordern.

  Weitere Informationen zur Planung und Implementierung einer Online-Migrationsstrategie finden Sie unterthe section called “Online-Migration”.
- Offline — Bei dieser Migrationstechnik wird ein Datensatz während eines Ausfallzeitfensters von Cassandra nach Amazon Keyspaces kopiert. Eine Offline-Migration kann den Migrationsprozess vereinfachen, da keine Änderungen an Ihrer Anwendung oder eine Konfliktlösung zwischen historischen Daten und neuen Schreibvorgängen erforderlich sind.

  Weitere Informationen zur Planung einer Offline-Migration finden Sie unterthe section called “Offline-Migration”.
- Hybrid — Mit dieser Migrationstechnik können Änderungen nahezu in Echtzeit auf Amazon Keyspaces repliziert werden, jedoch ohne Konsistenz beim Lesen nach dem Schreiben.

  Weitere Informationen zur Planung einer Hybridmigration finden Sie unter. the section called “Hybridmigration”

Nachdem Sie sich mit den in diesem Thema erläuterten Migrationstechniken und bewährten Methoden vertraut gemacht haben, können Sie die verfügbaren Optionen in einer Entscheidungsstruktur platzieren, um eine Migrationsstrategie zu entwerfen, die auf Ihren Anforderungen und verfügbaren Ressourcen basiert.

Online-Migration zu Amazon Keyspaces: Strategien und bewährte Verfahren

Wenn Sie die Anwendungsverfügbarkeit während einer Migration von Apache Cassandra zu Amazon Keyspaces aufrechterhalten müssen, können Sie eine benutzerdefinierte Online-Migrationsstrategie vorbereiten, indem Sie die in diesem Thema erläuterten Schlüsselkomponenten implementieren. Indem Sie diese bewährten Methoden für Online-Migrationen befolgen, können Sie sicherstellen, dass die Anwendungsverfügbarkeit und die Konsistenz beim Lesen nach dem Schreiben während des gesamten Migrationsprozesses erhalten bleiben, wodurch die Auswirkungen auf Ihre Benutzer minimiert werden.

Bei der Entwicklung einer Online-Migrationsstrategie von Apache Cassandra zu Amazon Keyspaces müssen Sie die folgenden wichtigen Schritte berücksichtigen.

1. Neue Daten schreiben
   - Migration von ZDM Dual Write Proxy für Amazon Keyspaces — Verwenden Sie den ZDM Dual Write Proxy, der auf Github verfügbar ist, um eine Migration ohne Ausfallzeiten von Apache Cassandra zu Amazon Keyspaces durchzuführen. Der ZDM-Proxy führt duale Schreibvorgänge durch, ohne dass bestehende Anwendungen umgestaltet werden müssen, und führt doppelte Lesevorgänge zur Abfragevalidierung durch.
   - Doppeltes Schreiben von Anwendungen: Sie können duale Schreibvorgänge in Ihrer Anwendung mithilfe vorhandener Cassandra-Clientbibliotheken und -Treiber implementieren. Benennen Sie eine Datenbank als Leader und die andere als Follower-Datenbank. Fehler beim Schreiben in die Follower-Datenbank werden zur Analyse in einer Dead Letter Queue (DLQ) aufgezeichnet.
   - Doppelte Schreibvorgänge auf Nachrichtenebene: Alternativ können Sie Ihre bestehende Messaging-Plattform so konfigurieren, dass Schreibvorgänge sowohl an Cassandra als auch an Amazon Keyspaces mithilfe eines zusätzlichen Consumer gesendet werden. Dadurch entstehen letztendlich konsistente Ansichten in beiden Datenbanken.
2. Migration historischer Daten
   - Historische Daten kopieren: Sie können historische Daten von Cassandra zu Amazon Keyspaces migrieren, indem Sie unsere benutzerdefinierten AWS Glue ETL-Skripte (Extract, Transform, Load) verwenden. Behandeln Sie die Konfliktlösung zwischen dualen Schreibvorgängen und Massenladungen mithilfe von Techniken wie einfachen Transaktionen oder Zeitstempeln.

- Verwenden Time-To-Live (TTL): Für kürzere Datenaufbewahrungsfristen können Sie TTL sowohl in Cassandra- als auch in Amazon Keyspaces verwenden, um das Hochladen unnötiger historischer Daten zu vermeiden. Wenn alte Daten in Cassandra ablaufen und neue Daten über Dual-Writes geschrieben werden, holt Amazon Keyspaces irgendwann auf.

3. Daten validieren
   - Doppelte Lesevorgänge: Implementieren Sie duale Lesevorgänge sowohl aus Cassandra-Datenbanken (primär) als auch aus Amazon Keyspaces-Datenbanken (sekundär) und vergleichen Sie die Ergebnisse asynchron. Unterschiede werden protokolliert oder an einen DLQ gesendet.
   - Stichprobenlesungen: Verwenden Sie die Λ -Funktionen, um in regelmäßigen Abständen Daten aus beiden Systemen abzufragen und zu vergleichen, wobei alle Abweichungen in einem DLQ protokolliert werden.
4. Die Anwendung wird migriert
   - Blue-green Strategie: Stellen Sie Ihre Anwendung in einem einzigen Schritt so um, dass Amazon Keyspaces als primären und Cassandra als sekundären Datenspeicher behandelt werden. Überwachen Sie die Leistung und führen Sie bei Problemen ein Rollback durch.
   - Bereitstellung auf Canary: Führen Sie die Migration zunächst schrittweise für eine Untergruppe von Benutzern durch und erhöhen Sie dabei schrittweise den Traffic zu Amazon Keyspaces als primärem Bereich, bis die Migration vollständig abgeschlossen ist.
5. Außerbetriebnahme von Cassandra

   Sobald Ihre Anwendung vollständig zu Amazon Keyspaces migriert und die Datenkonsistenz überprüft wurde, können Sie die Außerbetriebnahme Ihres Cassandra-Clusters auf der Grundlage der Datenaufbewahrungsrichtlinien planen.

Wenn Sie eine Online-Migrationsstrategie mit diesen Komponenten planen, können Sie reibungslos und mit minimalen Ausfallzeiten oder Unterbrechungen auf den vollständig verwalteten Amazon Keyspaces-Service umsteigen. In den folgenden Abschnitten wird auf jede Komponente näher eingegangen.

Themen

- Schreiben neuer Daten während einer Online-Migration
- Hochladen historischer Daten während einer Online-Migration
- Überprüfung der Datenkonsistenz während einer Online-Migration

- Migration der Anwendung während einer Online-Migration
- Außerbetriebnahme von Cassandra nach einer Online-Migration

Schreiben neuer Daten während einer Online-Migration

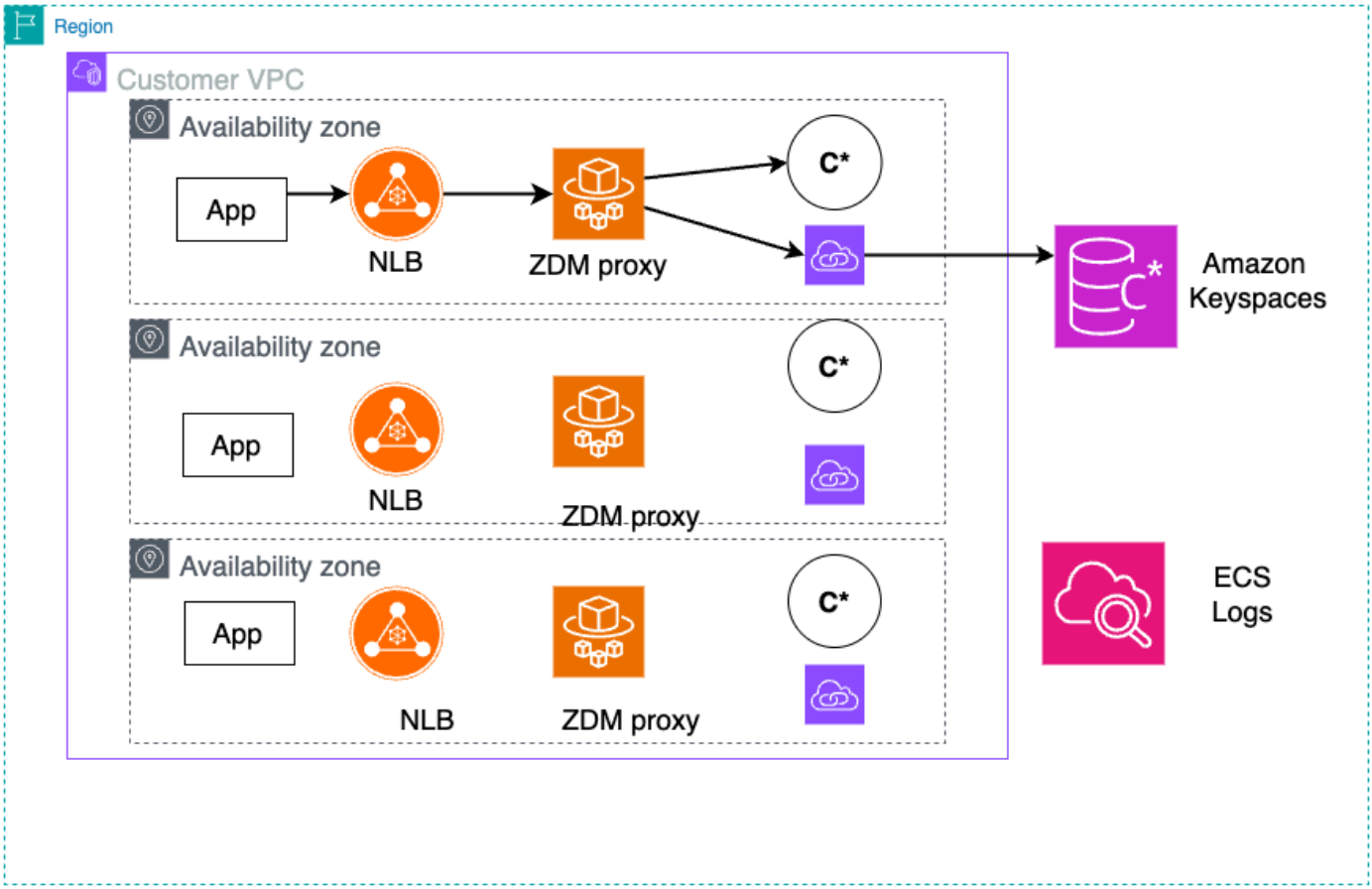
Der erste Schritt in einem Online-Migrationsplan besteht darin, sicherzustellen, dass alle neuen Daten, die von der Anwendung geschrieben werden, in beiden Datenbanken, Ihrem vorhandenen Cassandra-Cluster und Amazon Keyspaces, gespeichert werden. Ziel ist es, einen konsistenten Überblick über die beiden Datenspeicher zu bieten. Sie können dies tun, indem Sie alle neuen Schreibvorgänge auf beide Datenbanken anwenden. Um duale Schreibvorgänge zu implementieren, ziehen Sie eine der folgenden drei Optionen in Betracht.

- Migration von ZDM Dual Write Proxy für Amazon Keyspaces — Mithilfe des auf Github verfügbaren ZDM-Proxy für Amazon Keyspaces können Sie Ihre Apache Cassandra-Workloads ohne Anwendungsausfälle zu Amazon Keyspaces migrieren. Diese verbesserte Lösung implementiert AWS Best Practices und erweitert die offiziellen ZDM-Proxy-Funktionen.
  - Führen Sie Online-Migrationen zwischen Apache Cassandra und Amazon Keyspaces durch.
  - Schreiben Sie Daten gleichzeitig in Quell- und Zieltabellen, ohne die Anwendungen umzugestalten.
  - Überprüfen Sie Abfragen durch duale Lesevorgänge.

  Die Lösung bietet die folgenden Verbesserungen für die Arbeit mit AWS Amazon Keyspaces.

  - Container-Bereitstellung — Verwenden Sie ein vorkonfiguriertes Docker-Image aus Amazon Elastic Container Registry (Amazon ECR) für Bereitstellungen. VPC-accessible
  - Infrastruktur als Code — Bereitstellung mithilfe von AWS CloudFormation Vorlagen für die automatische Einrichtung AWS Fargate
  - Amazon Keyspaces-Kompatibilität — Greifen Sie auf Systemtabellen mit benutzerdefinierten Anpassungen für Amazon Keyspaces zu.

  Die Lösung läuft auf Amazon ECS mit Fargate und bietet serverlose Skalierbarkeit auf der Grundlage Ihrer Workload-Anforderungen. Ein Network Load Balancer verteilt den eingehenden Anwendungsdatenverkehr auf mehrere Amazon ECS-Aufgaben, um eine hohe Verfügbarkeit zu gewährleisten.



- Doppeltes Schreiben von Anwendungen — Sie können duale Schreibvorgänge mit minimalen Änderungen an Ihrem Anwendungscode implementieren, indem Sie die vorhandenen Cassandra-Clientbibliotheken und -Treiber nutzen. Sie können entweder duale Schreibvorgänge in Ihrer vorhandenen Anwendung implementieren oder eine neue Ebene in der Architektur erstellen, um duale Schreibvorgänge zu verarbeiten. Weitere Informationen und eine Kundenfallstudie, die zeigt, wie duale Schreibvorgänge in einer vorhandenen Anwendung implementiert wurden, finden Sie in der Fallstudie zur Cassandra-Migration.

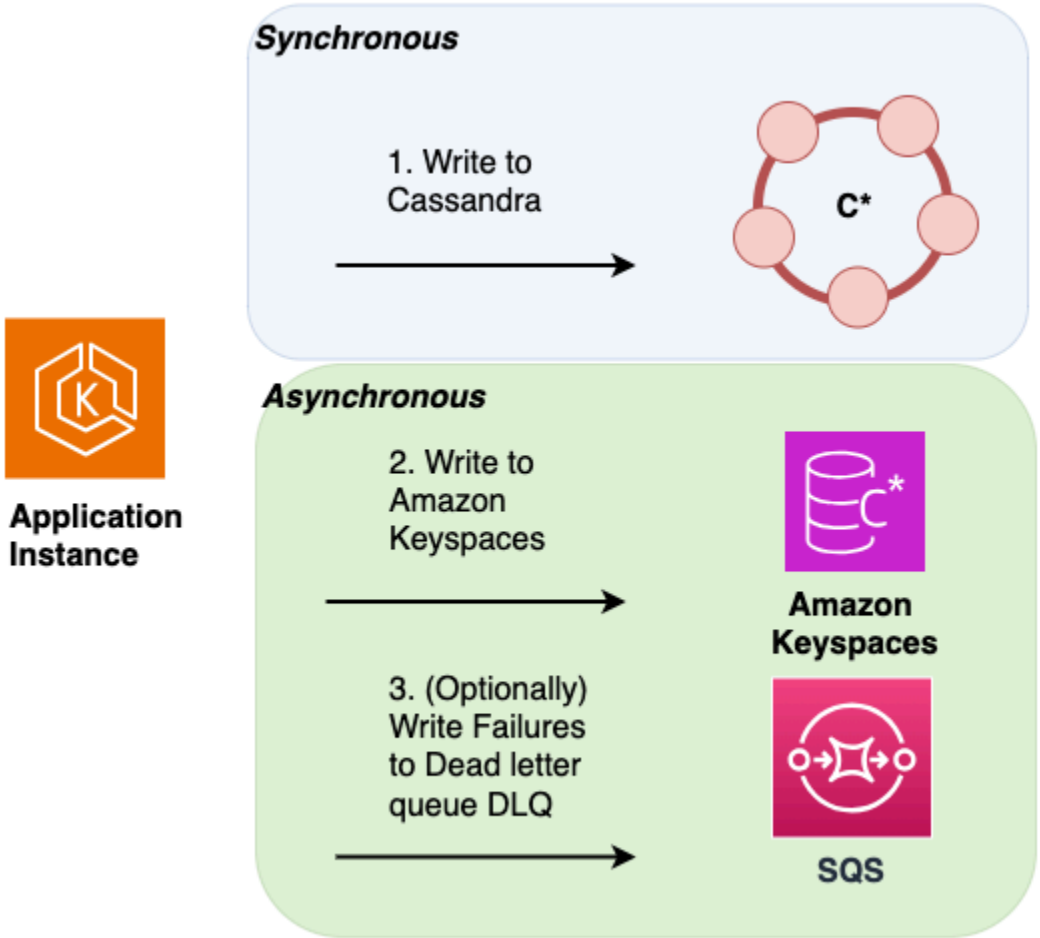
  Bei der Implementierung von Dual-Writes können Sie eine Datenbank als Leader und die andere Datenbank als Follower-Datenbank festlegen. Auf diese Weise können Sie weiterhin in Ihre ursprüngliche Quell- oder Leader-Datenbank schreiben, ohne dass Schreibfehler in die Follower- oder Zieldatenbank den kritischen Pfad Ihrer Anwendung stören.

  Anstatt fehlgeschlagene Schreibvorgänge in den Follower erneut zu versuchen, können Sie Amazon Simple Queue Service verwenden, um fehlgeschlagene Schreibvorgänge in einer Dead Letter Queue (DLQ) aufzuzeichnen. Mit dem DLQ können Sie die fehlgeschlagenen

Schreibvorgänge in den Follower analysieren und feststellen, warum die Verarbeitung in der Zieldatenbank nicht erfolgreich war.

Für eine anspruchsvollere Dual-Write-Implementierung können Sie die AWS bewährten Methoden zum Entwerfen einer Abfolge lokaler Transaktionen mithilfe des Saga-Musters befolgen. Ein Saga-Muster stellt sicher, dass die Saga, wenn eine Transaktion fehlschlägt, kompensierende Transaktionen ausführt, um die durch die vorherigen Transaktionen vorgenommenen Datenbankänderungen rückgängig zu machen.

Wenn Sie Dual-Writes für eine Online-Migration verwenden, können Sie die Dual-Writes nach dem Saga-Muster konfigurieren, sodass jeder Schreibvorgang eine lokale Transaktion ist, um atomare Operationen in heterogenen Datenbanken sicherzustellen. Weitere Informationen zum Entwerfen verteilter Anwendungen mithilfe der empfohlenen Entwurfsmuster für finden Sie unter Cloud-Entwurfsmuster AWS Cloud, -Architekturen und -Implementierungen.



- Doppelte Schreibvorgänge auf Messaging-Ebene — Anstatt duale Schreibvorgänge auf der Anwendungsebene zu implementieren, können Sie Ihre bestehende Messaging-Ebene verwenden, um duale Schreibvorgänge an Cassandra und Amazon Keyspaces durchzuführen.

  Zu diesem Zweck können Sie auf Ihrer Messaging-Plattform einen zusätzlichen Consumer konfigurieren, der Schreibvorgänge an beide Datenspeicher sendet. Dieser Ansatz bietet eine einfache Low-Code-Strategie, bei der die Messaging-Ebene verwendet wird, um zwei Ansichten in beiden Datenbanken zu erstellen, die letztendlich konsistent sind.

Hochladen historischer Daten während einer Online-Migration

Nachdem duale Schreibvorgänge implementiert wurden, um sicherzustellen, dass neue Daten in Echtzeit in beide Datenspeicher geschrieben werden, besteht der nächste Schritt im Migrationsplan darin, zu evaluieren, wie viele historische Daten Sie kopieren oder massenweise von Cassandra auf Amazon Keyspaces hochladen müssen. Dadurch wird sichergestellt, dass sowohl neue als auch historische Daten in der neuen Amazon Keyspaces-Datenbank verfügbar sind, bevor Sie die Anwendung migrieren.

Abhängig von Ihren Anforderungen zur Datenspeicherung, z. B. davon, wie viele historische Daten Sie gemäß den Richtlinien Ihres Unternehmens aufbewahren müssen, können Sie eine der folgenden beiden Optionen in Betracht ziehen.

- Massen-Upload historischer Daten — Die Migration historischer Daten von Ihrer bestehenden Cassandra-Bereitstellung zu Amazon Keyspaces kann durch verschiedene Techniken erreicht werden, z. B. mithilfe von AWS Glue oder benutzerdefinierten Skripten zum Extrahieren, Transformieren und Laden (ETL) der Daten. Weitere Informationen zur Verwendung AWS Glue zum Hochladen historischer Daten finden Sie unter. the section called “Offline-Migration”

  Wenn Sie den Massenupload historischer Daten planen, müssen Sie berücksichtigen, wie Sie Konflikte lösen können, die auftreten können, wenn neue Schreibvorgänge versuchen, dieselben Daten zu aktualisieren, die gerade hochgeladen werden. Es wird erwartet, dass der Massen-Upload letztendlich konsistent ist, was bedeutet, dass die Daten irgendwann alle Knoten erreichen werden.

  Wenn dieselben Daten aufgrund eines erneuten Schreibvorgangs gleichzeitig aktualisiert werden, sollten Sie sicherstellen, dass diese Daten nicht durch den Upload der historischen Daten überschrieben werden. Um sicherzustellen, dass Sie die neuesten Aktualisierungen Ihrer Daten

auch während des Massenimports beibehalten, müssen Sie die Konfliktlösung entweder in die Bulk-Upload-Skripts oder in die Anwendungslogik für duale Schreibvorgänge integrieren.

Beispielsweise können Sie the section called “Einfache Transaktionen” (LWT) verwenden, um Operationen zu vergleichen und festzulegen. Dazu können Sie Ihrem Datenmodell ein zusätzliches Feld hinzufügen, das den Zeitpunkt der Änderung oder den Status darstellt.

Darüber hinaus unterstützt Amazon Keyspaces die `WRITETIME` Cassandra-Zeitstempelfunktion. Sie können clientseitige Zeitstempel von Amazon Keyspaces verwenden, um die Zeitstempel der Quelldatenbank beizubehalten und die Last-Writer-Wins-Konfliktlösung zu implementieren. Weitere Informationen finden Sie unter the section called “Client-side Zeitstempel”.

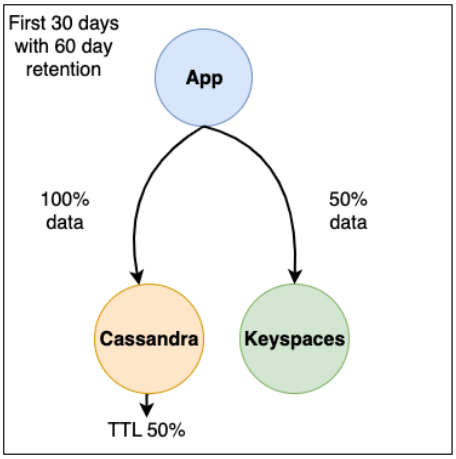
- Verwenden von Time-to-Live (TTL) — Für Datenaufbewahrungszeiträume unter 30, 60 oder 90 Tagen können Sie während der Migration TTL in Cassandra und Amazon Keyspaces verwenden, um zu vermeiden, dass unnötige historische Daten in Amazon Keyspaces hochgeladen werden. TTL ermöglicht es Ihnen, einen Zeitraum festzulegen, nach dem die Daten automatisch aus der Datenbank entfernt werden.

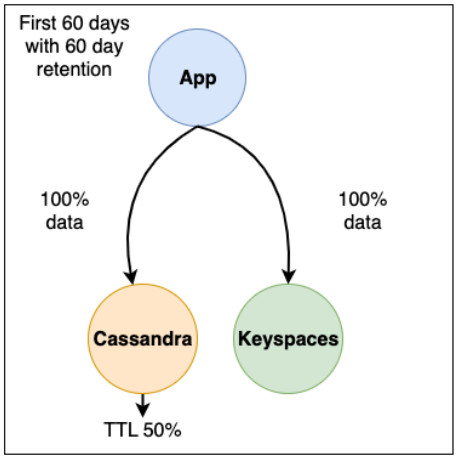
  Während der Migrationsphase können Sie, anstatt historische Daten in Amazon Keyspaces zu kopieren, die TTL-Einstellungen so konfigurieren, dass die historischen Daten im alten System (Cassandra) automatisch ablaufen, während die neuen Schreibvorgänge nur mithilfe der Dual-Write-Methode auf Amazon Keyspaces angewendet werden. Im Laufe der Zeit und da alte Daten im Cassandra-Cluster kontinuierlich ablaufen und neue Daten mit der Dual-Write-Methode geschrieben wurden, holt Amazon Keyspaces automatisch nach und enthält dieselben Daten wie Cassandra.

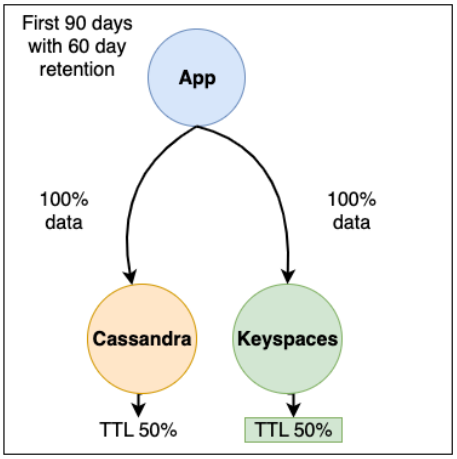
  Dieser Ansatz kann die Menge der zu migrierenden Daten erheblich reduzieren, was zu einem effizienteren und optimierten Migrationsprozess führt. Sie können diesen Ansatz in Betracht ziehen, wenn Sie mit großen Datensätzen mit unterschiedlichen Datenaufbewahrungsanforderungen zu tun haben. Weitere Informationen zu TTL finden Sie unter the section called “Daten mit Time to Live ablaufen lassen”.

  Stellen Sie sich das folgende Beispiel für eine Migration von Cassandra zu Amazon Keyspaces mit TTL-Datenablauf vor. In diesem Beispiel setzen wir TTL für beide Datenbanken auf 60 Tage und zeigen, wie der Migrationsprozess über einen Zeitraum von 90 Tagen abläuft. Beide Datenbanken erhalten in diesem Zeitraum dieselben neu geschriebenen Daten unter Verwendung der Methode mit zwei Schreibvorgängen. Wir werden uns drei verschiedene Phasen der Migration ansehen, jede Phase dauert 30 Tage.

Wie der Migrationsprozess für jede Phase funktioniert, wird in den folgenden Bildern gezeigt.







1. Nach den ersten 30 Tagen haben der Cassandra-Cluster und Amazon Keyspaces neue Schreibvorgänge erhalten. Der Cassandra-Cluster enthält auch historische Daten, für die die Aufbewahrungsdauer von 60 Tagen noch nicht erreicht ist, was 50% der Daten im Cluster ausmacht.

   Daten, die älter als 60 Tage sind, werden im Cassandra-Cluster mithilfe von TTL automatisch gelöscht. Zu diesem Zeitpunkt enthält Amazon Keyspaces 50% der im Cassandra-Cluster gespeicherten Daten, die sich aus den neuen Schreibvorgängen abzüglich der historischen Daten zusammensetzen.
2. Nach 60 Tagen enthalten sowohl der Cassandra-Cluster als auch Amazon Keyspaces dieselben Daten, die in den letzten 60 Tagen geschrieben wurden.
3. Innerhalb von 90 Tagen enthalten sowohl Cassandra als auch Amazon Keyspaces dieselben Daten und die Daten laufen mit derselben Geschwindigkeit ab.

Dieses Beispiel zeigt, wie Sie den Schritt des Hochladens historischer Daten vermeiden können, indem Sie TTL mit einem auf 60 Tage festgelegten Ablaufdatum verwenden.

Überprüfung der Datenkonsistenz während einer Online-Migration

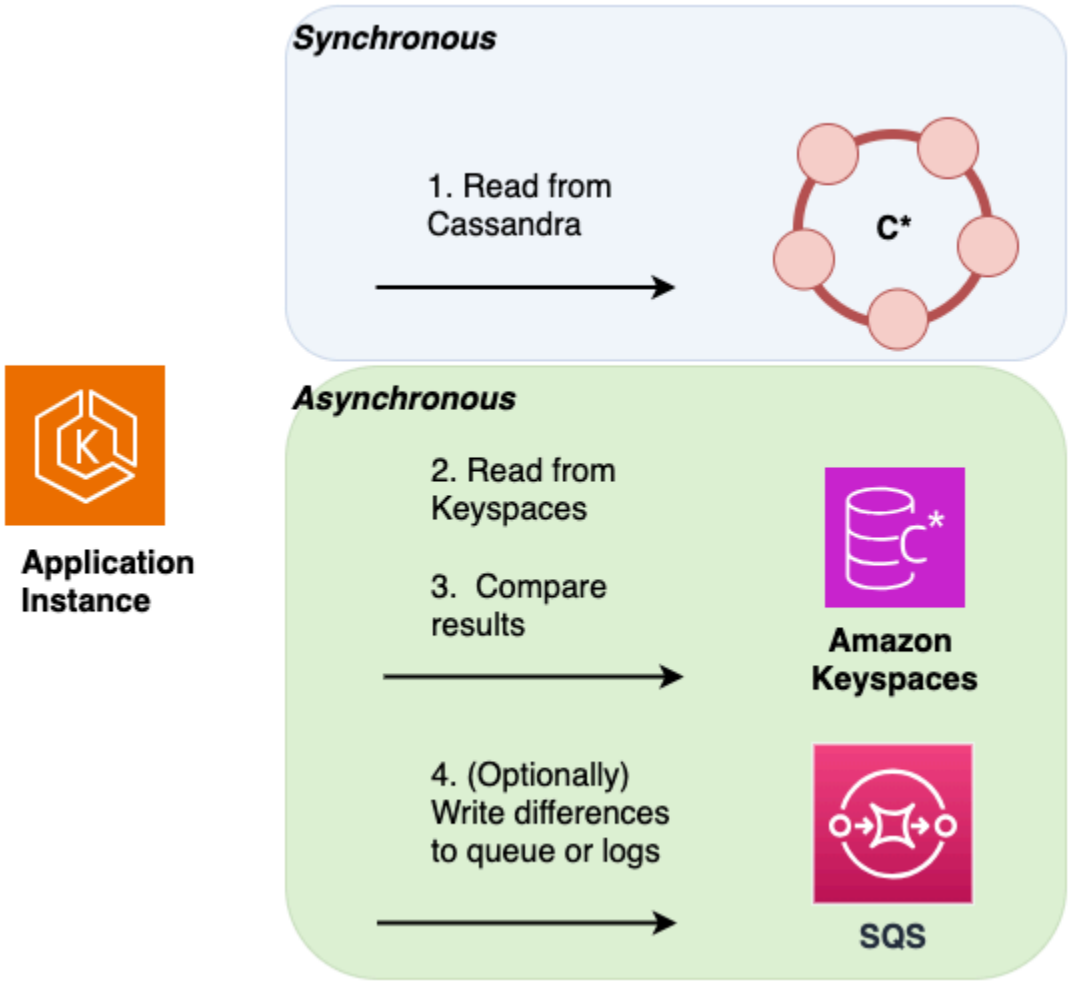
Der nächste Schritt im Online-Migrationsprozess ist die Datenvalidierung. Bei dualen Schreibvorgängen werden Ihrer Amazon Keyspaces-Datenbank neue Daten hinzugefügt, und Sie haben die Migration historischer Daten entweder durch Massen-Upload oder durch Datenablaufen mit TTL abgeschlossen.

Jetzt können Sie die Validierungsphase nutzen, um zu bestätigen, dass beide Datenspeicher tatsächlich dieselben Daten enthalten und dieselben Leseergebnisse zurückgeben. Sie können aus einer der folgenden beiden Optionen wählen, um zu überprüfen, ob Ihre beiden Datenbanken identische Daten enthalten.

- Doppelte Lesevorgänge — Um zu überprüfen, ob sowohl die Quell- als auch die Zieldatenbank denselben Satz neu geschriebener und historischer Daten enthalten, können Sie duale Lesevorgänge implementieren. Dazu lesen Sie ähnlich wie bei der Methode mit dualen Schreibvorgängen sowohl aus Ihrer primären Cassandra- als auch aus Ihrer sekundären Amazon Keyspaces-Datenbank und vergleichen die Ergebnisse asynchron.

  Die Ergebnisse aus der primären Datenbank werden an den Client zurückgegeben, und die Ergebnisse aus der sekundären Datenbank werden verwendet, um sie anhand der primären Ergebnismenge zu validieren. Gefundene Unterschiede können protokolliert oder zur späteren Abstimmung an eine Warteschlange für unzustellbare Briefe (DLQ) gesendet werden.

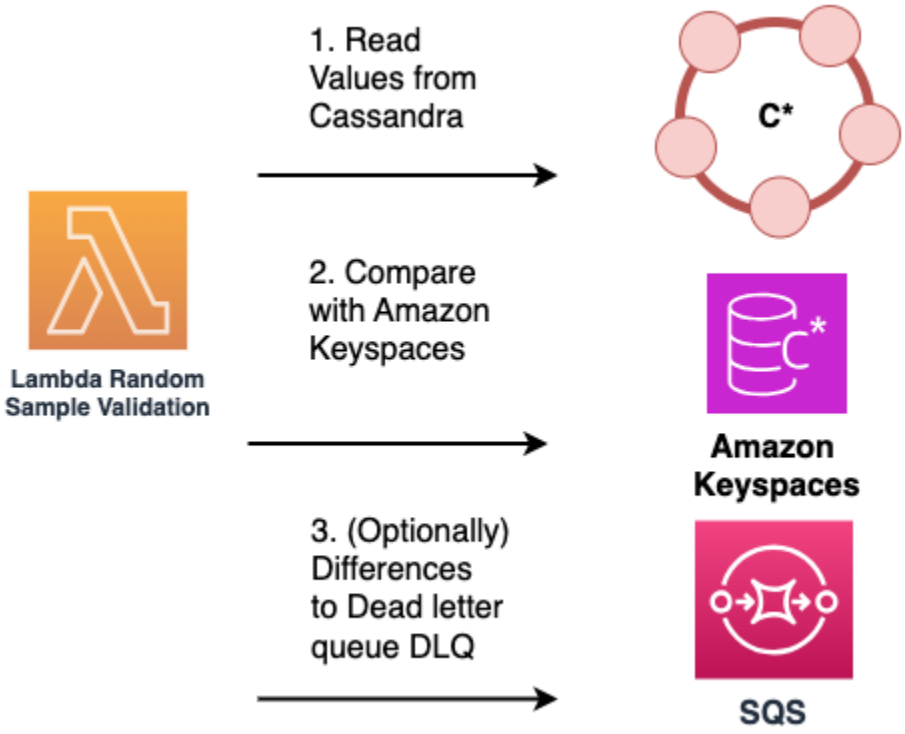
  In der folgenden Abbildung führt die Anwendung einen synchronen Lesevorgang aus Cassandra (dem primären Datenspeicher) und einen asynchronen Lesevorgang aus Amazon Keyspaces, dem sekundären Datenspeicher.



- Lesevorgänge — Eine alternative Lösung, die keine Änderungen am Anwendungscode erfordert, besteht darin, AWS Lambda Funktionen zu verwenden, um regelmäßig und nach dem Zufallsprinzip Daten sowohl aus dem Cassandra-Quell-Cluster als auch aus der Amazon Keyspaces-Zieldatenbank abzufragen.

  Diese Lambda-Funktionen können so konfiguriert werden, dass sie in regelmäßigen Abständen ausgeführt werden. Die Lambda-Funktion ruft eine zufällige Teilmenge von Daten sowohl aus dem Quell- als auch vom Zielsystem ab und führt dann einen Vergleich der Stichprobendaten durch. Jegliche Diskrepanzen oder Nichtübereinstimmungen zwischen den beiden Datensätzen können aufgezeichnet und zur späteren Abstimmung an eine dedizierte Warteschlange für tote Briefe (DLQ) gesendet werden.

  Dieser Prozess wird im folgenden Diagramm veranschaulicht.



Migration der Anwendung während einer Online-Migration

In der vierten Phase einer Online-Migration migrieren Sie Ihre Anwendung und wechseln zu Amazon Keyspaces als primären Datenspeicher. Das bedeutet, dass Sie Ihre Anwendung so umstellen, dass sie direkt von und zu Amazon Keyspaces liest und schreibt. Um sicherzustellen, dass Ihre Benutzer nur minimal gestört werden, sollte dies ein gut geplanter und koordinierter Prozess sein.

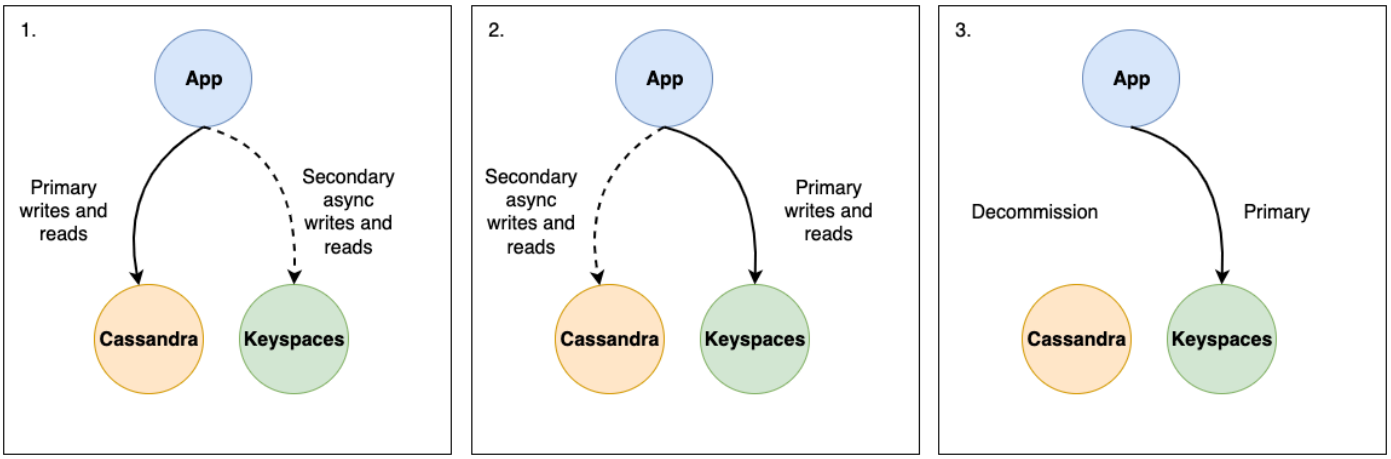
Es gibt zwei verschiedene empfohlene Lösungen für die Anwendungsmigration: die blau-grüne Cut-Over-Strategie und die Canary-Cut-Over-Strategie. In den folgenden Abschnitten werden diese Strategien detaillierter beschrieben.

- Blau-grüne Strategie — Mit diesem Ansatz stellen Sie Ihre Anwendung in einem einzigen Schritt so um, dass Amazon Keyspaces als primären Datenspeicher und Cassandra als sekundären Datenspeicher behandelt werden. Dazu können Sie ein AWS AppConfig Feature-Flag verwenden, um die Auswahl der primären und sekundären Datenspeicher in der Anwendungsinstanz zu steuern. Weitere Informationen zu Feature-Flags finden Sie unter Erstellen eines Feature-Flag-Konfigurationsprofils in AWS AppConfig.

Nachdem Sie Amazon Keyspaces zum primären Datenspeicher gemacht haben, überwachen Sie das Verhalten und die Leistung der Anwendung und stellen so sicher, dass Amazon Keyspaces Ihre Anforderungen erfüllt und die Migration erfolgreich ist.

Wenn Sie beispielsweise Dual-Reads für Ihre Anwendung implementiert haben, übertragen Sie während der Anwendungsmigrationsphase die primären Lesevorgänge von Cassandra auf Amazon Keyspaces und die sekundären Lesevorgänge von Amazon Keyspaces auf Cassandra. Nach der Umstellung überwachen und vergleichen Sie die Ergebnisse weiterhin, wie im Abschnitt zur Datenvalidierung beschrieben, um die Konsistenz beider Datenbanken sicherzustellen, bevor Cassandra außer Betrieb genommen wird.

Wenn Sie Probleme feststellen, können Sie schnell zum vorherigen Zustand zurückkehren, indem Sie zu Cassandra als primären Datenspeicher zurückkehren. Sie fahren nur dann mit der Außerbetriebnahmephase der Migration fort, wenn Amazon Keyspaces alle Ihre Anforderungen als primärer Datenspeicher erfüllt.

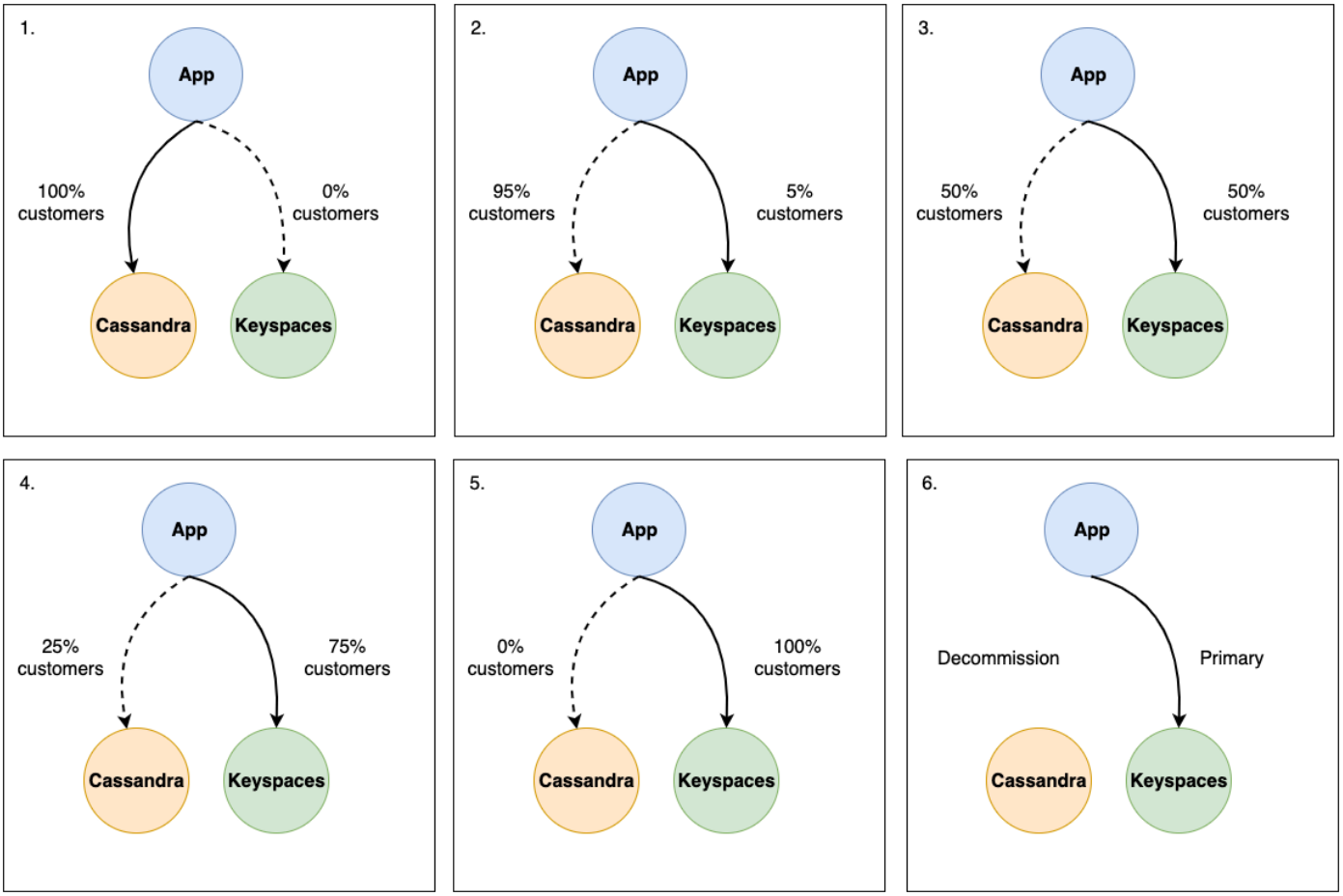


- Canary-Strategie — Bei diesem Ansatz führen Sie die Migration schrittweise auf eine Teilmenge Ihrer Benutzer oder Ihres Datenverkehrs aus. Anfänglich wird ein kleiner Prozentsatz des Datenverkehrs Ihrer Anwendung, beispielsweise 5% des gesamten Datenverkehrs, an die Version weitergeleitet, die Amazon Keyspaces als primären Datenspeicher verwendet, während der Rest des Datenverkehrs weiterhin Cassandra als primären Datenspeicher verwendet.

  Auf diese Weise können Sie die migrierte Version gründlich mit echtem Datenverkehr testen und ihre Leistung und Stabilität überwachen und potenzielle Probleme untersuchen. Wenn Sie keine Probleme feststellen, können Sie den Prozentsatz des Datenverkehrs, der zu Amazon Keyspaces weitergeleitet wird, schrittweise erhöhen, bis er zum primären Datenspeicher für alle Benutzer und den gesamten Datenverkehr wird.

Diese schrittweise Einführung minimiert das Risiko weit verbreiteter Serviceunterbrechungen und ermöglicht einen kontrollierteren Migrationsprozess. Sollten während der Bereitstellung von Canary kritische Probleme auftreten, können Sie schnell zur vorherigen Version zurückkehren, indem Sie Cassandra als primären Datenspeicher für das betroffene Traffic-Segment verwenden. Sie fahren erst mit der Außerbetriebnahmephase der Migration fort, nachdem Sie bestätigt haben, dass Amazon Keyspaces 100% Ihrer Benutzer und Ihres Datenverkehrs erwartungsgemäß verarbeitet.

Das folgende Diagramm veranschaulicht die einzelnen Schritte der Canary-Strategie.



Außerbetriebnahme von Cassandra nach einer Online-Migration

Nachdem die Anwendungsmigration abgeschlossen ist, Ihre Anwendung vollständig auf Amazon Keyspaces läuft und Sie die Datenkonsistenz über einen bestimmten Zeitraum überprüft haben, können Sie die Außerbetriebnahme Ihres Cassandra-Clusters planen. Während dieser Phase können Sie beurteilen, ob die in Ihrem Cassandra-Cluster verbleibenden Daten archiviert werden müssen

oder gelöscht werden können. Dies hängt von den Richtlinien Ihres Unternehmens für den Umgang und die Aufbewahrung von Daten ab.

Wenn Sie bei der Planung Ihrer Online-Migration von Cassandra zu Amazon Keyspaces dieser Strategie folgen und die in diesem Thema beschriebenen empfohlenen Best Practices berücksichtigen, können Sie einen nahtlosen Übergang zu Amazon Keyspaces sicherstellen und gleichzeitig die Lese-nach-Schreib-Konsistenz und Verfügbarkeit Ihrer Anwendung aufrechterhalten.

Die Migration von Apache Cassandra zu Amazon Keyspaces kann zahlreiche Vorteile bieten, darunter einen geringeren Betriebsaufwand, automatische Skalierung, verbesserte Sicherheit und ein Framework, das Ihnen hilft, Ihre Compliance-Ziele zu erreichen. Durch die Planung einer Online-Migrationsstrategie mit doppelten Schreibvorgängen, Upload historischer Daten, Datenvalidierung und schrittweiser Einführung können Sie einen reibungslosen Übergang mit minimalen Unterbrechungen für Ihre Anwendung und ihre Benutzer sicherstellen.

Durch die Implementierung der in diesem Thema erörterten Online-Migrationsstrategie können Sie die Migrationsergebnisse validieren, Probleme identifizieren und beheben und letztendlich Ihre bestehende Cassandra-Bereitstellung zugunsten des vollständig verwalteten Amazon Keyspaces-Service außer Betrieb nehmen.

Offline-Migrationsprozess: Apache Cassandra zu Amazon Keyspaces

Offline-Migrationen eignen sich, wenn Sie sich Ausfallzeiten für die Durchführung der Migration leisten können. In Unternehmen ist es üblich, dass Wartungsfenster für Patches, große Releases oder Ausfallzeiten aufgrund von Hardware-Upgrades oder größeren Upgrades gelten. Bei der Offline-Migration kann dieses Fenster verwendet werden, um Daten zu kopieren und den Anwendungsdatenverkehr von Apache Cassandra auf Amazon Keyspaces umzustellen.

Die Offline-Migration reduziert Änderungen an der Anwendung, da keine gleichzeitige Kommunikation mit Cassandra und Amazon Keyspaces erforderlich ist. Wenn der Datenfluss unterbrochen ist, kann außerdem der genaue Status kopiert werden, ohne dass Mutationen beibehalten werden.

In diesem Beispiel verwenden wir Amazon Simple Storage Service (Amazon S3) als Staging-Bereich für Daten während der Offline-Migration, um Ausfallzeiten zu minimieren. Sie können die Daten, die Sie in Amazon S3 im Parquet-Format gespeichert haben, mithilfe des Spark-Cassandra-Connectors und automatisch in eine Amazon Keyspaces-Tabelle importieren. AWS Glue Der folgende Abschnitt gibt einen allgemeinen Überblick über den Prozess. Codebeispiele für diesen Prozess finden Sie auf Github.

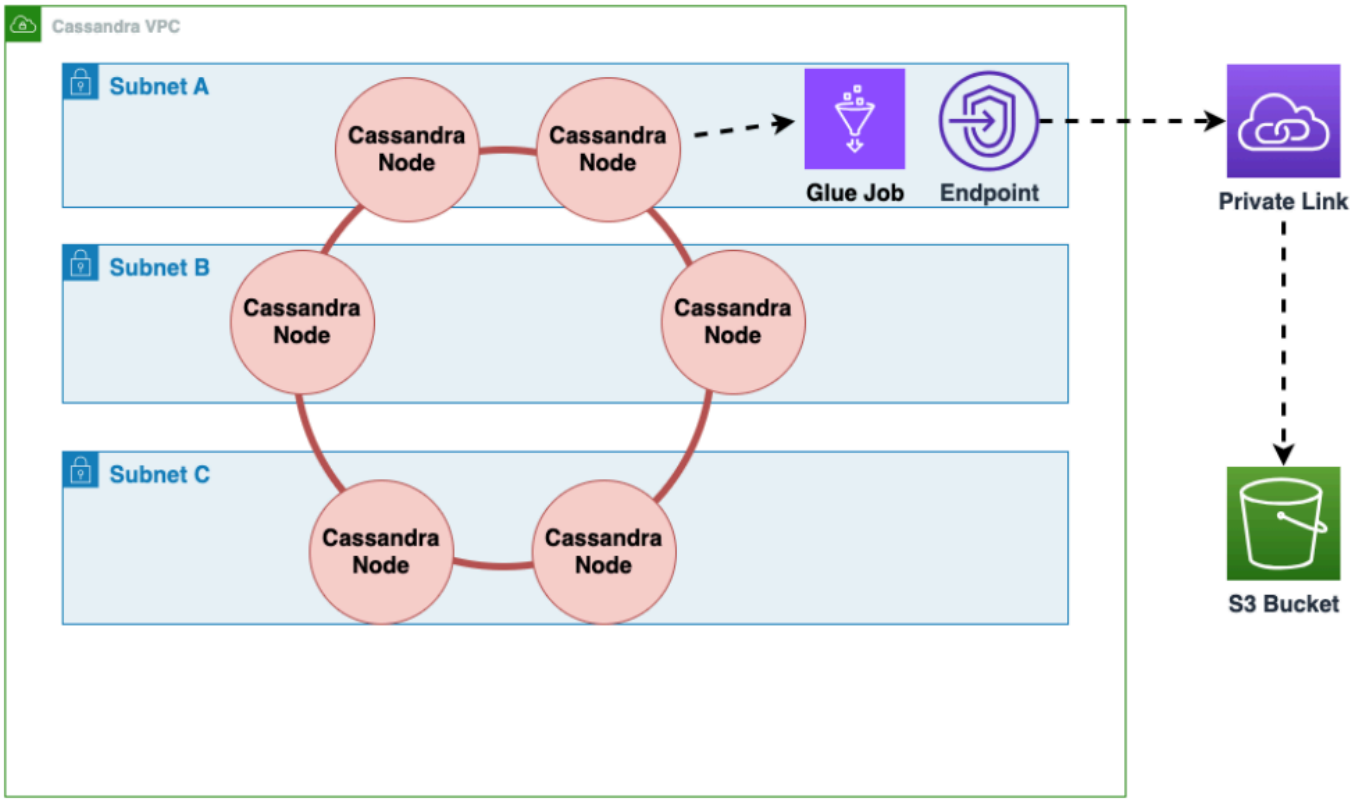
Der Offline-Migrationsprozess von Apache Cassandra zu Amazon Keyspaces verwendet Amazon S3 und AWS Glue erfordert die folgenden AWS Glue Jobs.

1. Ein ETL-Job, der CQL-Daten extrahiert, transformiert und in einem Amazon S3-Bucket speichert.
2. Ein zweiter Job, der die Daten aus dem Bucket in Amazon Keyspaces importiert.
3. Ein dritter Job zum Importieren inkrementeller Daten.

Wie führe ich eine Offline-Migration von Cassandra, die auf Amazon EC2 in einer Amazon Virtual Private Cloud ausgeführt wird, zu Amazon Keyspaces durch

1. Zuerst exportieren Sie AWS Glue Tabellendaten aus Cassandra im Parquet-Format und speichern sie in einem Amazon S3-Bucket. Sie müssen einen AWS Glue Job mithilfe eines AWS Glue Connectors zu einer VPC ausführen, auf der sich die Amazon EC2-Instance befindet, auf der Cassandra ausgeführt wird. Anschließend können Sie mithilfe des privaten Amazon S3-Endpunkts Daten im Amazon S3-Bucket speichern.

   Das folgende Diagramm veranschaulicht diese Schritte.



2. Mischen Sie die Daten im Amazon S3-Bucket, um die Randomisierung der Daten zu verbessern. Gleichmäßig importierte Daten ermöglichen einen stärker verteilten Datenverkehr in der Zieltabelle.

   Dieser Schritt ist erforderlich, wenn Daten aus Cassandra mit großen Partitionen (Partitionen mit mehr als 1000 Zeilen) exportiert werden, um Hotkey-Muster beim Einfügen der Daten in Amazon Keyspaces zu vermeiden. Hotkey-Probleme werden `WriteThrottleEvents` in Amazon Keyspaces verursacht und führen zu einer längeren Ladezeit.



3. Verwenden Sie einen anderen AWS Glue Job, um Daten aus dem Amazon S3-Bucket in Amazon Keyspaces zu importieren. Die gemischten Daten im Amazon S3-Bucket werden im Parquet-Format gespeichert.



Weitere Informationen zum Offline-Migrationsprozess finden Sie im Workshop Amazon Keyspaces mit AWS Glue

Verwendung einer hybriden Migrationslösung: Apache Cassandra zu Amazon Keyspaces

Die folgende Migrationslösung kann als eine Mischung aus Online- und Offline-Migration betrachtet werden. Bei diesem hybriden Ansatz werden Daten nahezu in Echtzeit in die Zieldatenbank geschrieben, ohne dass eine Konsistenz beim Lesen nach dem Schreiben gewährleistet ist. Das bedeutet, dass neu geschriebene Daten nicht sofort verfügbar sind und Verzögerungen zu

erwarten sind. Wenn Sie ein einheitliches Lesen nach dem Schreiben benötigen, finden Sie weitere Informationen unterthe section called “Online-Migration”.

Für eine Migration von Apache Cassandra zu Amazon Keyspaces nahezu in Echtzeit können Sie zwischen zwei verfügbaren Methoden wählen.

- CqlReplicator — (Empfohlen) CqlReplicator ist ein Open-Source-Programm, das auf Github verfügbar ist und Ihnen hilft, Daten nahezu in Echtzeit von Apache Cassandra zu Amazon Keyspaces zu migrieren.

  Um zu ermitteln, welche Schreibvorgänge und Aktualisierungen an die Zieldatenbank weitergegeben werden sollen, scannt CqlReplicator den Apache Cassandra-Token-Bereich und verwendet einen AWS Glue Job, um doppelte Ereignisse zu entfernen und Schreibvorgänge und Aktualisierungen direkt auf Amazon Keyspaces anzuwenden.
- Change Data Capture (CDC) — Wenn Sie mit Cassandra CDC vertraut sind, ist die in Apache Cassandra integrierte CDC-Funktion, die das Erfassen von Änderungen durch Kopieren des Commit-Logs in ein separates CDC-Verzeichnis ermöglicht, eine weitere Option für die Implementierung einer hybriden Migration.

  Sie können dies tun, indem Sie die Datenänderungen auf Amazon Keyspaces replizieren, wodurch CDC zu einer alternativen Option für Datenmigrationsszenarien wird.

Wenn Sie keine Konsistenz beim Lesen nach dem Schreiben benötigen, können Sie entweder den CQLReplicator oder eine CDC-Pipeline verwenden, um Daten von Apache Cassandra zu Amazon Keyspaces zu migrieren, je nach Ihren Präferenzen und Ihrer Vertrautheit mit den in jeder Lösung verwendeten Tools. AWS-Services Die Verwendung dieser Methoden zur Migration von Daten nahezu in Echtzeit kann als hybrider Migrationsansatz betrachtet werden, der eine Alternative zur Online-Migration bietet.

Diese Strategie wird als hybrider Ansatz betrachtet, da Sie zusätzlich zu den in diesem Thema beschriebenen Optionen einige Schritte des Online-Migrationsprozesses implementieren müssen, z. B. das Kopieren historischer Daten und die im Thema Online-Migration erläuterten Strategien zur Anwendungsmigration.

In den folgenden Abschnitten werden die Optionen für die hybride Migration ausführlicher behandelt.

Themen

- Migrieren Sie Daten mit CQLReplicator

- Migrieren Sie Daten mithilfe von Change Data Capture (CDC)

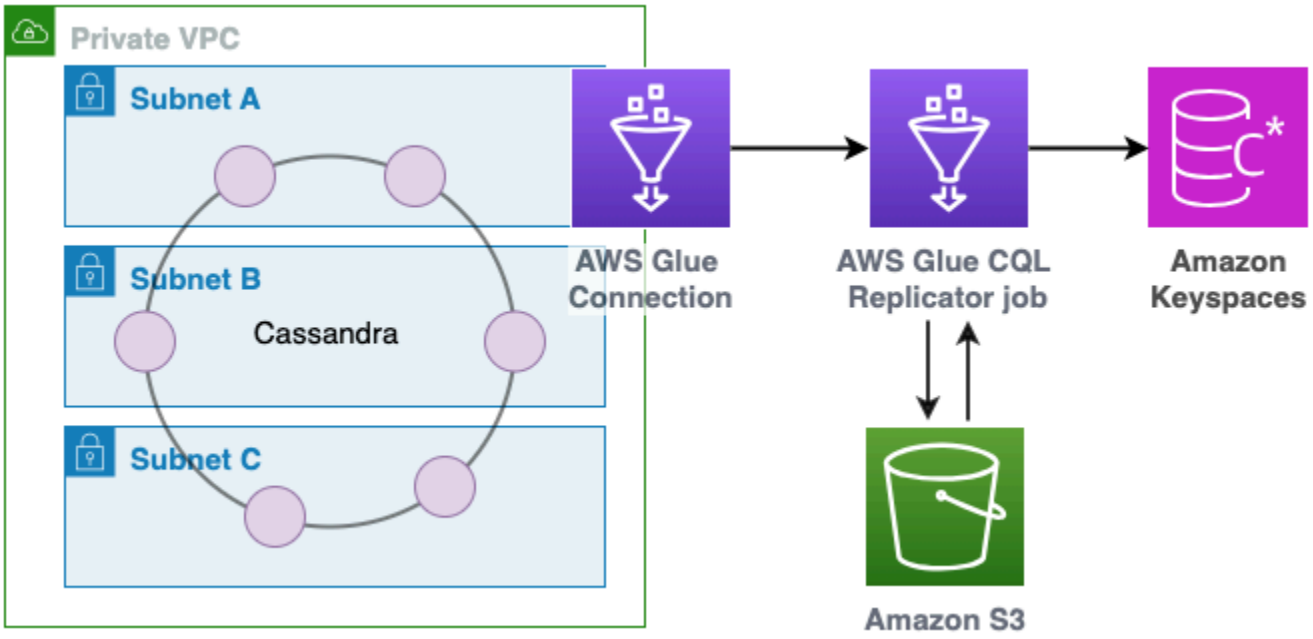
Migrieren Sie Daten mit CQLReplicator

Mit CqlReplicator können Sie Daten von Apache Cassandra nahezu in Echtzeit lesen, indem Sie den Cassandra-Token-Ring mithilfe von CQL-Abfragen intelligent scannen. CqlReplicator verwendet Cassandra CDC nicht und implementiert stattdessen eine Caching-Strategie, um die Leistungseinbußen bei vollständigen Scans zu reduzieren.

Um die Anzahl der Schreibvorgänge auf das Ziel zu reduzieren, entfernt CqlReplicator automatisch doppelte Replikationsereignisse. Mit CqlReplicator können Sie die Replikation von Änderungen von der Quelldatenbank in die Zieldatenbank optimieren und so eine Migration von Daten von Apache Cassandra zu Amazon Keyspaces nahezu in Echtzeit ermöglichen.

Das folgende Diagramm zeigt die typische Architektur eines CqlReplicator-Jobs mit. AWS Glue

1. Um den Zugriff auf Apache Cassandra zu ermöglichen, das in einer privaten VPC ausgeführt wird, konfigurieren Sie eine AWS Glue Verbindung mit dem Verbindungstyp Netzwerk.
2. Um Duplikate zu entfernen und das Schlüssel-Caching mit dem CQLReplicator-Job zu aktivieren, konfigurieren Sie Amazon Simple Storage Service (Amazon S3).
3. Der CqlReplicator-Job streamt verifizierte Änderungen an der Quelldatenbank direkt an Amazon Keyspaces.



Weitere Informationen zum Migrationsprozess mit CqlReplicator finden Sie im folgenden Beitrag im AWS Datenbank-Blog Migrieren Sie Cassandra-Workloads mit CqlReplicator zu Amazon Keyspaces und in der vorgeschriebenen Anleitung Migrieren Sie Apache Cassandra-Workloads mithilfe von Amazon Keyspaces. AWS AWS Glue

Migrieren Sie Daten mithilfe von Change Data Capture (CDC)

Wenn Sie bereits mit der Konfiguration einer Change Data Capture (CDC) -Pipeline mit Debezium vertraut sind, können Sie diese Option verwenden, um Daten als Alternative zur Verwendung von CQLReplicator zu Amazon Keyspaces zu migrieren. Debezium ist eine verteilte Open-Source-Plattform für CDC, die darauf ausgelegt ist, eine Datenbank zu überwachen und Änderungen auf Zeilenebene zuverlässig zu erfassen.

Der Debezium-Connector für Apache Cassandra lädt Änderungen auf Amazon Managed Streaming for Apache Kafka (Amazon MSK) hoch, sodass sie von nachgeschalteten Verbrauchern konsumiert und verarbeitet werden können, die wiederum die Daten in Amazon Keyspaces schreiben. Weitere Informationen finden Sie unter Anleitung zur kontinuierlichen Datenmigration von Apache Cassandra zu Amazon Keyspaces.

Um potenzielle Probleme mit der Datenkonsistenz zu lösen, können Sie mit Amazon MSK einen Prozess implementieren, bei dem ein Verbraucher die Schlüssel oder Partitionen in Cassandra mit denen in Amazon Keyspaces vergleicht.

Um diese Lösung erfolgreich zu implementieren, empfehlen wir, Folgendes zu berücksichtigen.

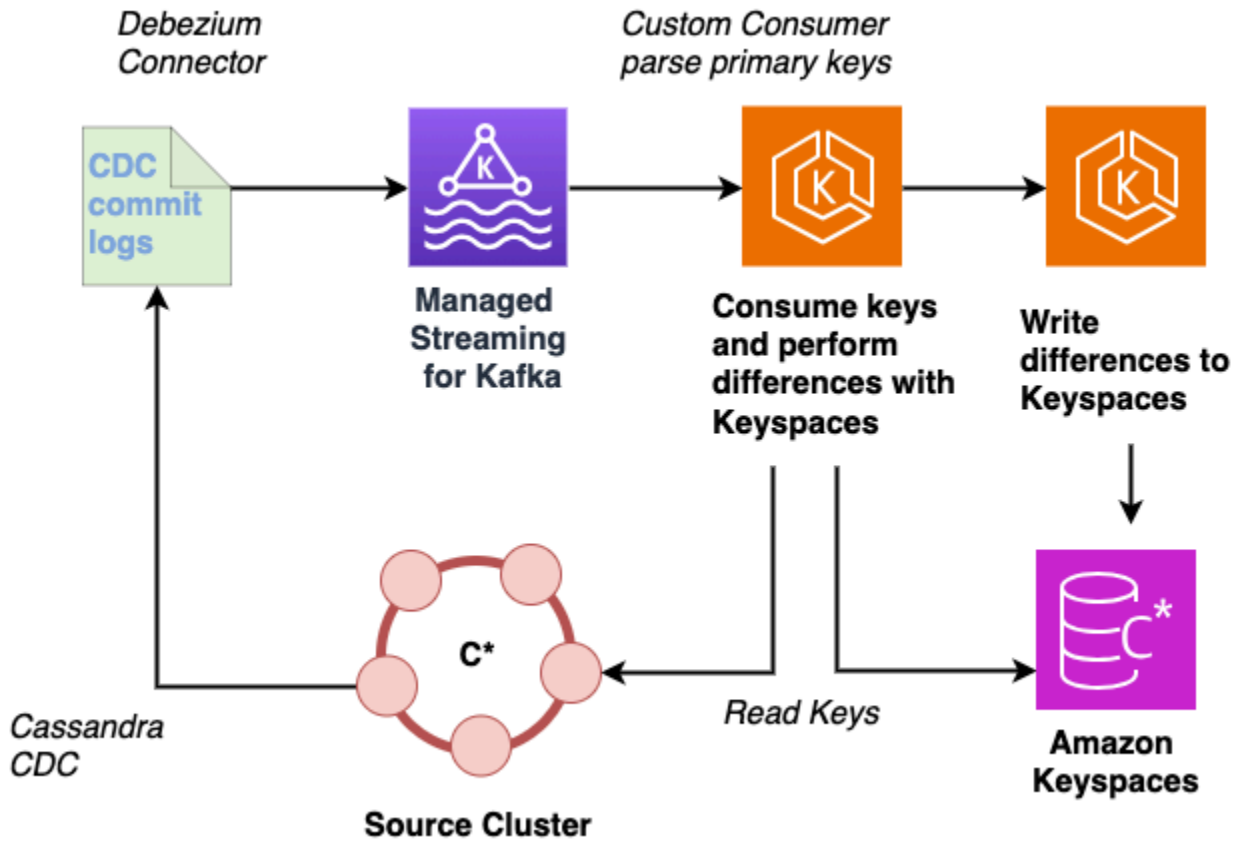
- Wie man das CDC-Commit-Log analysiert, zum Beispiel, wie man doppelte Ereignisse entfernt.
- Wie man das CDC-Verzeichnis verwaltet, zum Beispiel wie man alte Logs löscht.
- Wie man mit Teilfehlern in Apache Cassandra umgeht, zum Beispiel, wenn ein Schreibvorgang nur in einem von drei Replikaten erfolgreich ist.
- Wie mit der Ressourcenzuweisung umgegangen wird, z. B. durch Erhöhen der Größe der Instanz, um zusätzlichen CPU-, Speicher-, DISK- und I/O-Anforderungen für den CDC-Prozess auf einem Knoten Rechnung zu tragen.

Dieses Muster behandelt Änderungen von Cassandra als „Hinweis“ darauf, dass sich ein Schlüssel gegenüber seinem vorherigen Zustand geändert haben könnte. Um festzustellen, ob Änderungen an die Zieldatenbank weitergegeben werden müssen, müssen Sie zuerst aus dem Cassandra-Quell-Cluster lesen, indem Sie eine `LOCAL_QUORUM` Operation ausführen, um die neuesten Datensätze zu empfangen, und sie dann in Amazon Keyspaces schreiben.

Bei Bereichslöschungen oder Bereichsaktualisierungen müssen Sie möglicherweise einen Vergleich mit der gesamten Partition durchführen, um festzustellen, welche Schreib- oder Aktualisierungsereignisse in Ihre Zieldatenbank geschrieben werden müssen.

In Fällen, in denen Schreibvorgänge nicht idempotent sind, müssen Sie Ihre Schreibvorgänge auch mit dem vergleichen, was sich bereits in der Zieldatenbank befindet, bevor Sie in Amazon Keyspaces schreiben.

Das folgende Diagramm zeigt die typische Architektur einer CDC-Pipeline mit Debezium und Amazon MSK.



So wählen Sie das richtige Tool für den Massen-Upload oder die Migration von Daten zu Amazon Keyspaces aus

In diesem Abschnitt können Sie sich mit den verschiedenen Tools vertraut machen, mit denen Sie Daten massenweise auf Amazon Keyspaces hochladen oder migrieren können, und erfahren, wie Sie das richtige Tool für Ihre Bedürfnisse auswählen können. Darüber hinaus bietet dieser Abschnitt einen Überblick und Anwendungsfälle für die verfügbaren schrittweisen Anleitungen, die zeigen, wie Daten in Amazon Keyspaces importiert werden.

Informationen zu den verfügbaren Strategien für die Migration von Workloads von Apache Cassandra zu Amazon Keyspaces finden Sie unter. the section called “Migration von Cassandra”

- Tools für die Migration
  - Mit dem Preisrechner für Amazon Keyspaces (für Apache Cassandra), der auf Github verfügbar ist, können Sie Ihre monatlichen Kosten für Amazon Keyspaces auf der Grundlage Ihrer vorhandenen Apache Cassandra-Arbeitslast schätzen. Geben Sie Metriken aus Ihrer Cassandra-Nodetool-Statusausgabe und der geplanten serverlosen Konfiguration für Amazon Keyspaces

ein, um die direkten Kosten zwischen den beiden Lösungen zu vergleichen. Beachten Sie, dass sich dieser Rechner nur auf die Betriebskosten von Amazon Keyspaces im Vergleich zu Ihrer bestehenden Cassandra-Bereitstellung konzentriert. Dabei sind die Gesamtbetriebskosten (TCO) wie Infrastrukturwartung, Betriebskosten oder Supportkosten für Cassandra nicht berücksichtigt.

- ZDM Dual Write Proxy für Amazon Keyspaces-Migration — Der auf Github verfügbare ZDM Dual Write Proxy unterstützt die Migration ohne Ausfallzeiten von Apache Cassandra zu Amazon Keyspaces.
- CqlReplicator — CqlReplicator ist ein auf Github verfügbares Open-Source-Hilfsprogramm, mit dem Sie Daten nahezu in Echtzeit von Apache Cassandra zu Amazon Keyspaces migrieren können.

  Weitere Informationen finden Sie unter the section called “CqlReplicator”.
- Weitere Informationen zur Verwendung von Amazon Managed Streaming for Apache Kafka zur Implementierung eines Online-Migrationsprozesses mit Dual-Writes finden Sie unter Anleitung für die kontinuierliche Datenmigration von Apache Cassandra zu Amazon Keyspaces.
- Für umfangreiche Migrationen sollten Sie die Verwendung eines ETL-Tool (Extract, Transform and Load) in Betracht ziehen. Sie können es verwenden AWS Glue , um Datentransformationsmigrationen schnell und effektiv durchzuführen. Weitere Informationen finden Sie unter the section called “Offline-Migration”.
- Informationen zur Verwendung des Apache Cassandra Spark-Connectors zum Schreiben von Daten in Amazon Keyspaces finden Sie unter. the section called “Integration mit Apache Spark”
- Beginnen Sie schnell mit dem Laden von Daten in Amazon Keyspaces mithilfe des `COPY FROM` Befehls cqlsh. cqlsh ist in Apache Cassandra enthalten und eignet sich am besten zum Laden kleiner Datensätze oder Testdaten. Schrittweise Anleitungen hierzu finden Sie unter the section called “Daten mit cqlsh laden”.
- Sie können auch den DataStax Bulk Loader für Apache Cassandra verwenden, um Daten mit dem Befehl in Amazon Keyspaces zu laden. `dsbulk` DSBulk bietet robustere Importfunktionen als cqlsh und ist im Repository verfügbar. GitHub Schrittweise Anleitungen hierzu finden Sie unter the section called “Daten mit DSBulk laden”.

Allgemeine Überlegungen zu Datenuploads auf Amazon Keyspaces

- Teilen Sie den Datenupload in kleinere Komponenten auf.

Betrachten Sie die folgenden Migrationseinheiten und ihren potenziellen Platzbedarf in Bezug auf die Rohdatengröße. Das Hochladen kleinerer Datenmengen in einer oder mehreren Phasen kann dazu beitragen, Ihre Migration zu vereinfachen.

- Nach Cluster — Migrieren Sie alle Ihre Cassandra-Daten auf einmal. Dieser Ansatz kann für kleinere Cluster in Ordnung sein.
- Nach Schlüsselraum oder Tabelle — Teilen Sie Ihre Migration in Gruppen von Schlüsselräumen oder Tabellen auf. Dieser Ansatz kann Ihnen dabei helfen, Daten in Phasen zu migrieren, die Ihren Anforderungen für jeden Workload entsprechen.
- Nach Daten — Erwägen Sie die Migration von Daten für eine bestimmte Gruppe von Benutzern oder Produkten, um die Datenmenge noch weiter zu reduzieren.

- Priorisieren Sie anhand der Einfachheit, welche Daten zuerst hochgeladen werden sollen.

  Überlegen Sie, ob Sie Daten haben, die zunächst einfacher migriert werden könnten, z. B. Daten, die sich zu bestimmten Zeiten nicht ändern, Daten aus nächtlichen Batch-Jobs, Daten, die während der Offline-Zeiten nicht verwendet werden, oder Daten aus internen Apps.

Themen

- Tutorial: Daten mit cqlsh in Amazon Keyspaces laden
- Tutorial: Daten mit DSBulk in Amazon Keyspaces laden

Tutorial: Daten mit cqlsh in Amazon Keyspaces laden

Dieses Tutorial führt Sie durch den Prozess der Migration von Daten von Apache Cassandra zu Amazon Keyspaces mithilfe des Befehls. `cqlsh COPY FROM` Der `cqlsh COPY FROM` Befehl ist nützlich, um schnell und einfach kleine Datensätze für akademische Zwecke oder Testzwecke auf Amazon Keyspaces hochzuladen. Weitere Informationen zur Migration von Produktionsworkloads finden Sie unter. the section called “Offline-Migration” In diesem Tutorial führen Sie die folgenden Schritte durch:

Voraussetzungen — Richten Sie ein AWS Konto mit Anmeldeinformationen ein, erstellen Sie eine JKS-Trust-Store-Datei für das Zertifikat und konfigurieren Sie die Verbindung `cqlsh` zu Amazon Keyspaces.

1. Quell-CSV und Zieltabelle erstellen — Bereiten Sie eine CSV-Datei als Quelldaten vor und erstellen Sie den Zielschlüsselraum und die Zieltabelle in Amazon Keyspaces.

2. Daten vorbereiten — Randomisieren Sie die Daten in der CSV-Datei und analysieren Sie sie, um die durchschnittliche und maximale Zeilengröße zu ermitteln.
3. Durchsatzkapazität festlegen — Berechnen Sie die erforderlichen Schreibkapazitätseinheiten (WCUs) auf der Grundlage der Datengröße und der gewünschten Ladezeit und konfigurieren Sie die bereitgestellte Kapazität der Tabelle.
4. Cqlsh-Parameter konfigurieren — Ermitteln Sie optimale Werte für `cqlsh COPY FROM` Parameter wie`INGESTRATE`,, und `NUMPROCESSES`, `MAXBATCHSIZE`, um die Arbeitslast gleichmäßig `CHUNKSIZE` zu verteilen.
5. **`cqlsh COPY FROM`**Befehl ausführen — Führen Sie den `cqlsh COPY FROM` Befehl aus, um die Daten aus der CSV-Datei in die Amazon Keyspaces-Tabelle hochzuladen, und überwachen Sie den Fortschritt.

Fehlerbehebung — Beheben Sie häufig auftretende Probleme wie ungültige Anfragen, Parserfehler, Kapazitätsfehler und Cqlsh-Fehler während des Daten-Upload-Vorgangs.

Themen

- Voraussetzungen: Schritte, die Sie ausführen müssen, bevor Sie Daten mit cqlsh COPY FROM hochladen können
- Schritt 1: Erstellen Sie die CSV-Quelldatei und eine Zieltabelle für den Datenupload
- Schritt 2: Bereiten Sie die Quelldaten für einen erfolgreichen Datenupload vor
- Schritt 3: Stellen Sie die Durchsatzkapazität für die Tabelle ein
- Schritt 4: Konfigurieren Sie die Einstellungen von cqlsh COPY FROM
- Schritt 5: Führen Sie den Befehl cqlsh COPY FROM aus, um Daten aus der CSV-Datei in die Zieltabelle hochzuladen
- Fehlerbehebung

Voraussetzungen: Schritte, die Sie ausführen müssen, bevor Sie Daten mit cqlsh COPY FROM hochladen können

Sie müssen die folgenden Aufgaben erledigen, bevor Sie mit diesem Tutorial beginnen können.

1. Falls Sie dies noch nicht getan haben, melden Sie sich für eine an, AWS-Konto indem Sie den Schritten unter folgenthe section called “Amazon Keyspaces einrichten”.

2. Erstellen Sie dienstspezifische Anmeldeinformationen, indem Sie die Schritte unter the section called “Erstellen Sie dienstspezifische Anmeldeinformationen” befolgen.
3. Richten Sie die Cassandra Query Language Shell (cqlsh) -Verbindung ein und bestätigen Sie, dass Sie eine Verbindung zu Amazon Keyspaces herstellen können, indem Sie die Schritte unter befolgen. the section called “Verwenden von `cqlsh`”

Schritt 1: Erstellen Sie die CSV-Quelldatei und eine Zieltabelle für den Datenupload

Für dieses Tutorial verwenden wir eine Datei mit kommagetrennten Werten (CSV) mit dem Namen `keyspaces_sample_table.csv` als Quelldatei für die Datenmigration. Die mitgelieferte Beispieldatei enthält einige Datenzeilen für eine Tabelle mit dem Namen. `book_awards`

1. Erstellen Sie die Quelldatei. Sie können eine der folgenden Optionen wählen:
   - Laden Sie die CSV-Beispieldatei (`keyspaces_sample_table.csv`) herunter, die in der folgenden Archivdatei samplemigration.zip enthalten ist. Entpacken Sie das Archiv und notieren Sie sich den Pfad zu`keyspaces_sample_table.csv`.
   - Um eine CSV-Datei mit Ihren eigenen Daten zu füllen, die in einer Apache Cassandra-Datenbank gespeichert sind, können Sie die CSV-Quelldatei mit der `cqlsh COPY TO` Anweisung füllen, wie im folgenden Beispiel gezeigt.

     ```
     cqlsh localhost 9042 -u "username" -p "password" --execute
      "COPY mykeyspace.mytable TO 'keyspaces_sample_table.csv' WITH HEADER=true";
     ```

     Stellen Sie sicher, dass die von Ihnen erstellte CSV-Datei die folgenden Anforderungen erfüllt:

     - Die erste Zeile enthält die Spaltennamen.
     - Die Spaltennamen in der CSV-Quelldatei stimmen mit den Spaltennamen in der Zieltabelle überein.
     - Die Daten sind durch ein Komma getrennt.
     - Alle Datenwerte sind gültige Amazon Keyspaces-Datentypen. Siehe the section called “Datentypen”.
2. Erstellen Sie den Zielschlüsselraum und die Zieltabelle in Amazon Keyspaces.
   a. Connect zu Amazon Keyspaces her`cqlsh`, indem Sie den Service-Endpunkt, den Benutzernamen und das Passwort im folgenden Beispiel durch Ihre eigenen Werte ersetzen.

```
cqlsh cassandra.us-east-1.amazonaws.com 9142 -u "111122223333" -p
 "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

b. Erstellen Sie einen neuen Schlüsselraum mit dem Namen, `catalog` wie im folgenden Beispiel gezeigt.

```
CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

c. Wenn der neue Schlüsselraum verfügbar ist, verwenden Sie den folgenden Code, um die Zieltabelle zu erstellen. `book_awards`

```
CREATE TABLE "catalog.book_awards" (
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
   PRIMARY KEY ((year, award), category, rank)
   );
```

Wenn Apache Cassandra Ihre ursprüngliche Datenquelle ist, besteht eine einfache Möglichkeit, die Amazon Keyspaces-Zieltabelle mit passenden Headern zu erstellen, darin, die `CREATE TABLE` Anweisung aus der Quelltabelle zu generieren, wie in der folgenden Anweisung gezeigt.

```
cqlsh localhost 9042  -u "username" -p "password" --execute "DESCRIBE
 TABLE mykeyspace.mytable;"
```

Erstellen Sie dann die Zieltabelle in Amazon Keyspaces mit den Spaltennamen und Datentypen, die der Beschreibung aus der Cassandra-Quelltabelle entsprechen.

Schritt 2: Bereiten Sie die Quelldaten für einen erfolgreichen Datenupload vor

Die Vorbereitung der Quelldaten für eine effiziente Übertragung erfolgt in zwei Schritten. Zunächst randomisieren Sie die Daten. Im zweiten Schritt analysieren Sie die Daten, um die geeigneten `cqlsh` Parameterwerte und erforderlichen Tabelleneinstellungen zu ermitteln, um sicherzustellen, dass der Datenupload erfolgreich ist.

Randomisieren Sie die Daten

Der `cqlsh COPY FROM` Befehl liest und schreibt Daten in derselben Reihenfolge, in der sie in der CSV-Datei erscheinen. Wenn Sie den `cqlsh COPY TO` Befehl verwenden, um die Quelldatei zu erstellen, werden die Daten in schlüsselsortierter Reihenfolge in die CSV-Datei geschrieben. Intern partitioniert Amazon Keyspaces Daten mithilfe von Partitionsschlüsseln. Amazon Keyspaces verfügt zwar über eine integrierte Logik, die beim Lastenausgleich von Anfragen für denselben Partitionsschlüssel hilft, das Laden der Daten ist jedoch schneller und effizienter, wenn Sie die Reihenfolge nach dem Zufallsprinzip festlegen. Dies liegt daran, dass Sie den integrierten Lastenausgleich nutzen können, der auftritt, wenn Amazon Keyspaces auf verschiedene Partitionen schreibt.

Um die Schreibvorgänge gleichmäßig auf die Partitionen zu verteilen, müssen Sie die Daten in der Quelldatei randomisieren. Sie können dafür eine Anwendung schreiben oder ein Open-Source-Tool wie Shuf verwenden. Shuf ist auf Linux-Distributionen, auf macOS (durch Installation von Coreutils in Homebrew) und unter Windows (mithilfe des Windows Subsystems for Linux (WSL)) kostenlos verfügbar. Ein zusätzlicher Schritt ist erforderlich, um zu verhindern, dass die Kopfzeile mit den Spaltennamen in diesem Schritt gemischt wird.

Um die Quelldatei nach dem Zufallsprinzip zu sortieren und gleichzeitig die Kopfzeile beizubehalten, geben Sie den folgenden Code ein.

```
tail -n +2 keyspaces_sample_table.csv | shuf -o keyspace.table.csv && (head
 -1 keyspaces_sample_table.csv && cat keyspace.table.csv ) > keyspace.table.csv1 &&
 mv keyspace.table.csv1 keyspace.table.csv
```

Shuf schreibt die Daten in eine neue CSV-Datei mit dem Namen um. `keyspace.table.csv` Sie können die `keyspaces_sample_table.csv` Datei jetzt löschen — Sie benötigen sie nicht mehr.

Analysieren Sie die Daten

Ermitteln Sie die durchschnittliche und maximale Zeilengröße, indem Sie die Daten analysieren.

Sie tun dies aus den folgenden Gründen:

- Die durchschnittliche Zeilengröße hilft bei der Schätzung der Gesamtmenge der zu übertragenden Daten.
- Sie benötigen die durchschnittliche Zeilengröße, um die für den Datenupload benötigte Schreibkapazität bereitzustellen.

- Sie können sicherstellen, dass jede Zeile weniger als 1 MB groß ist. Dies ist die maximale Zeilengröße in Amazon Keyspaces.

> **Note**
>
> Dieses Kontingent bezieht sich auf die Zeilengröße, nicht auf die Partitionsgröße. Im Gegensatz zu Apache Cassandra-Partitionen können Amazon Keyspaces-Partitionen praktisch unbegrenzt groß sein. Partitionsschlüssel und Clusterspalten benötigen zusätzlichen Speicherplatz für Metadaten, den Sie zur Rohgröße der Zeilen hinzufügen müssen. Weitere Informationen finden Sie unter the section called “Schätzen Sie die Zeilengröße”.

Der folgende Code verwendet AWK, um eine CSV-Datei zu analysieren und die durchschnittliche und maximale Zeilengröße zu drucken.

```
awk -F, 'BEGIN {samp=10000;max=-1;}{if(NR>1){len=length($0);t+=len;avg=t/
NR;max=(len>max ? len : max)}}NR==samp{exit}END{printf("{lines: %d, average: %d bytes,
 max: %d bytes}\n",NR,avg,max);}' keyspace.table.csv
```

Die Ausführung dieses Codes führt zu der folgenden Ausgabe.

```
using 10,000 samples:
{lines: 10000, avg: 123 bytes, max: 225 bytes}
```

Im nächsten Schritt dieses Tutorials verwenden Sie die durchschnittliche Zeilengröße, um die Schreibkapazität für die Tabelle bereitzustellen.

Schritt 3: Stellen Sie die Durchsatzkapazität für die Tabelle ein

Dieses Tutorial zeigt Ihnen, wie Sie cqlsh so einstellen, dass Daten innerhalb eines festgelegten Zeitbereichs geladen werden. Da Sie im Voraus wissen, wie viele Lese- und Schreibvorgänge Sie durchführen, sollten Sie den Modus für bereitgestellte Kapazität verwenden. Nachdem Sie die Datenübertragung abgeschlossen haben, sollten Sie den Kapazitätsmodus der Tabelle so einstellen, dass er den Datenverkehrsmustern Ihrer Anwendung entspricht. Weitere Informationen zur Kapazitätsverwaltung finden Sie unterVerwaltung serverloser Ressourcen.

Im Modus „Bereitgestellte Kapazität“ geben Sie im Voraus an, wie viel Lese- und Schreibkapazität Sie für Ihre Tabelle bereitstellen möchten. Die Schreibkapazität wird stündlich abgerechnet und in

Schreibkapazitätseinheiten (WCUs) gemessen. Jede WCU bietet ausreichend Schreibkapazität, um das Schreiben von 1 KB Daten pro Sekunde zu unterstützen. Wenn Sie die Daten laden, muss die Schreibrate unter den maximalen WCUs (Parameter:`write_capacity_units`) liegen, die in der Zieltabelle festgelegt sind.

Standardmäßig können Sie bis zu 40.000 WCUs für eine Tabelle und 80.000 WCUs für alle Tabellen in Ihrem Konto bereitstellen. Wenn Sie zusätzliche Kapazität benötigen, können Sie in der Service Quotas-Konsole eine Erhöhung des Kontingents beantragen. Weitere Informationen zu Kontingenten finden Sie unter Kontingente.

Berechnen Sie die durchschnittliche Anzahl der WCUs, die für eine Einfügung erforderlich sind

Für das Einfügen von 1 KB Daten pro Sekunde ist 1 WCU erforderlich. Wenn Ihre CSV-Datei 360.000 Zeilen hat und Sie alle Daten in einer Stunde laden möchten, müssen Sie 100 Zeilen pro Sekunde schreiben (360.000 Zeilen/60 Minuten/ 60 Sekunden = 100 Zeilen pro Sekunde). Wenn jede Zeile bis zu 1 KB Daten enthält, müssen Sie 100 WCUs für Ihre Tabelle bereitstellen, um 100 Zeilen pro Sekunde einzufügen. Wenn jede Zeile 1,5 KB Daten enthält, benötigen Sie zwei WCUs, um eine Zeile pro Sekunde einzufügen. Um 100 Zeilen pro Sekunde einzufügen, müssen Sie daher 200 WCUs bereitstellen.

Um zu ermitteln, wie viele WCUs Sie für das Einfügen einer Zeile pro Sekunde benötigen, teilen Sie die durchschnittliche Zeilengröße in Byte durch 1024 und runden Sie auf die nächste ganze Zahl auf.

Wenn die durchschnittliche Zeilengröße beispielsweise 3000 Byte beträgt, benötigen Sie drei WCUs, um eine Zeile pro Sekunde einzufügen.

```
ROUNDUP(3000 / 1024) = ROUNDUP(2.93) = 3 WCUs
```

Berechnen Sie die Ladezeit und Kapazität der Daten

Da Sie nun die durchschnittliche Größe und Anzahl der Zeilen in Ihrer CSV-Datei kennen, können Sie berechnen, wie viele WCUs Sie benötigen, um die Daten in einem bestimmten Zeitraum zu laden, und wie lange es ungefähr dauert, alle Daten in Ihrer CSV-Datei mithilfe verschiedener WCU-Einstellungen zu laden.

Wenn beispielsweise jede Zeile in Ihrer Datei 1 KB groß ist und Sie 1.000.000 Zeilen in Ihrer CSV-Datei haben, müssen Sie für diese Stunde mindestens 278 WCUs für Ihre Tabelle bereitstellen, um die Daten in einer Stunde zu laden.

```
1,000,000 rows * 1 KBs = 1,000,000 KBs
```

```
1,000,000 KBs / 3600 seconds =277.8 KBs / second = 278 WCUs
```

Konfigurieren Sie die Einstellungen für die bereitgestellte Kapazität

Sie können die Schreibkapazitätseinstellungen einer Tabelle festlegen, wenn Sie die Tabelle erstellen oder den `ALTER TABLE` CQL-Befehl verwenden. Im Folgenden finden Sie die Syntax für die Änderung der bereitgestellten Kapazitätseinstellungen einer Tabelle mit der `ALTER TABLE` CQL-Anweisung.

```
ALTER TABLE mykeyspace.mytable WITH custom_properties={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 100,
 'write_capacity_units': 278}} ;
```

Die vollständige Sprachreferenz finden Sie unter. the section called “ALTER TABLE”

Schritt 4: Konfigurieren Sie die Einstellungen von `cqlsh COPY FROM`

In diesem Abschnitt wird beschrieben, wie die Parameterwerte für bestimmt werden. `cqlsh COPY FROM` Der `cqlsh COPY FROM` Befehl liest die CSV-Datei, die Sie zuvor vorbereitet haben, und fügt die Daten mithilfe von CQL in Amazon Keyspaces ein. Der Befehl teilt die Zeilen auf und verteilt die `INSERT` Operationen auf eine Gruppe von Arbeitern. Jeder Mitarbeiter stellt eine Verbindung zu Amazon Keyspaces her und sendet `INSERT` Anfragen über diesen Kanal.

Der `cqlsh COPY` Befehl verfügt nicht über eine interne Logik, um die Arbeit gleichmäßig auf die Mitarbeiter zu verteilen. Sie können ihn jedoch manuell konfigurieren, um sicherzustellen, dass die Arbeit gleichmäßig verteilt wird. Überprüfen Sie zunächst die folgenden wichtigen Cqlsh-Parameter:

- DELIMITER — Wenn Sie ein anderes Trennzeichen als ein Komma verwendet haben, können Sie diesen Parameter festlegen, der standardmäßig auf Komma gesetzt ist.
- INGESTRATE — Die Zielanzahl von Zeilen, die pro Sekunde verarbeitet werden sollen. `cqlsh COPY FROM` Wenn sie nicht gesetzt ist, ist sie standardmäßig auf 100.000 eingestellt.
- NUMPROCESSES — Die Anzahl der untergeordneten Worker-Prozesse, die cqlsh für Aufgaben erstellt. `COPY FROM` Das Maximum für diese Einstellung ist 16, die Standardeinstellung ist`num_cores - 1`, wobei es sich um die Anzahl der Prozessorkerne auf dem Host `num_cores` handelt, auf dem cqlsh ausgeführt wird.
- MAXBATCHSIZE — Die Batchgröße bestimmt die maximale Anzahl von Zeilen, die in einem einzigen Batch in die Zieltabelle eingefügt werden. Falls nicht gesetzt, verwendet cqlsh Stapel von 20 eingefügten Zeilen.

- CHUNKSIZE — Die Größe der Arbeitseinheit, die an den untergeordneten Arbeiter weitergegeben wird. Standardmäßig ist sie auf 5.000 festgelegt.
- MAXATTEMPTS — Die maximale Anzahl von Wiederholungen für einen fehlgeschlagenen Worker-Chunk. Wenn der maximale Versuch erreicht ist, werden die fehlgeschlagenen Datensätze in eine neue CSV-Datei geschrieben, die Sie später erneut ausführen können, nachdem Sie den Fehler untersucht haben.

Dieser Wert `INGESTRATE` basiert auf der Anzahl der WCUs, die Sie in der Zieltabelle bereitgestellt haben. Der `INGESTRATE` Wert des `cqlsh COPY FROM` Befehls ist kein Limit, sondern ein Zieldurchschnitt. Das bedeutet, dass er die von Ihnen festgelegte Zahl überschreiten kann (und tut dies häufig auch). Um Bursts zu berücksichtigen und sicherzustellen, dass genügend Kapazität zur Bearbeitung der Datenladeanforderungen vorhanden ist, sollten Sie einen Wert von 90% der Schreibkapazität der Tabelle festlegen`INGESTRATE`.

```
INGESTRATE = WCUs * .90
```

Stellen Sie als Nächstes den `NUMPROCESSES` Parameter so ein, dass er um eins kleiner ist als die Anzahl der Kerne in Ihrem System. Um herauszufinden, wie viele Kerne Ihr System hat, können Sie den folgenden Code ausführen.

```
python -c "import multiprocessing; print(multiprocessing.cpu_count())"
```

Für dieses Tutorial verwenden wir den folgenden Wert.

```
NUMPROCESSES = 4
```

Jeder Prozess erstellt einen Worker, und jeder Worker stellt eine Verbindung zu Amazon Keyspaces her. Amazon Keyspaces kann bei jeder Verbindung bis zu 3.000 CQL-Anfragen pro Sekunde unterstützen. Das bedeutet, dass Sie sicherstellen müssen, dass jeder Mitarbeiter weniger als 3.000 Anfragen pro Sekunde verarbeitet.

Wie bei der `INGESTRATE` anderen Methode überschreiten die Mitarbeiter häufig die von Ihnen festgelegte Anzahl und sind nicht durch Taktsekunden begrenzt. Um Bursts zu berücksichtigen, sollten Sie Ihre cqlsh-Parameter daher so einstellen, dass jeder Worker 2.500 Anfragen pro Sekunde verarbeitet. Verwenden Sie die folgende Richtlinie, um den Arbeitsaufwand zu berechnen, der auf eine Arbeitskraft verteilt wird.

- Dividiere `INGESTRATE` durch`NUMPROCESSES`.

- WennINGESTRATE/NUMPROCESSES> 2.500, verringern Sie den Wert, INGESTRATE damit diese Formel wahr wird.

```
INGESTRATE / NUMPROCESSES <= 2,500
```

Bevor Sie die Einstellungen zur Optimierung des Uploads unserer Beispieldaten konfigurieren, sollten wir uns zunächst die cqlsh Standardeinstellungen ansehen und herausfinden, wie sich ihre Verwendung auf den Daten-Upload-Prozess auswirkt. Da der cqlsh COPY FROM verwendet wirdCHUNKSIZE, um Arbeitsblöcke (INSERTKontoauszüge) zu erstellen, um sie an die Mitarbeiter zu verteilen, wird die Arbeit nicht automatisch gleichmäßig verteilt. Je nach Einstellung können einige Mitarbeiter untätig sitzen. INGESTRATE

Um die Arbeit gleichmäßig auf die Mitarbeiter zu verteilen und dafür zu sorgen, dass für jeden Mitarbeiter die optimale Rate von 2.500 Anfragen pro Sekunde erreicht wirdCHUNKSIZE, müssen SieMAXBATCHSIZE, und INGESTRATE durch Ändern der Eingabeparameter festlegen. Um die Auslastung des Netzwerkverkehrs während des Datenladens zu optimieren, wählen Sie einen WertMAXBATCHSIZE, der dem Höchstwert von 30 nahe kommt. Wenn CHUNKSIZE Sie den Wert auf 100 und MAXBATCHSIZE auf 25 ändern, werden die 10.000 Zeilen gleichmäßig auf die vier Mitarbeiter verteilt (10.000/2500 = 4).

Das folgende Codebeispiel veranschaulicht dies.

```
INGESTRATE = 10,000
NUMPROCESSES = 4
CHUNKSIZE = 100
MAXBATCHSIZE. = 25
Work Distribution:
Connection 1 / Worker 1 : 2,500 Requests per second
Connection 2 / Worker 2 : 2,500 Requests per second
Connection 3 / Worker 3 : 2,500 Requests per second
Connection 4 / Worker 4 : 2,500 Requests per second
```

Zusammenfassend lässt sich sagen, dass Sie beim Einstellen von cqlsh COPY FROM Parametern die folgenden Formeln verwenden:

- INGESTRATE = write_capacity_units * .90
- NUMPROCESSES = num_cores -1 (Standard)
- INGESTRATE//NUMPROCESSES = 2.500 (Dies muss eine wahre Aussage sein.)

- MAXBATCHSIZE = 30 (Der Standardwert ist 20. Amazon Keyspaces akzeptiert Batches von bis zu 30.)
- CHUNKSIZE = (INGESTRATE//NUMPROCESSES)/MAXBATCHSIZE

Nachdem Sie, und `CHUNKSIZE` berechnet haben, sind Sie `NUMPROCESSES` bereit`INGESTRATE`, Ihre Daten zu laden.

Schritt 5: Führen Sie den Befehl **`cqlsh COPY FROM aus`**, um Daten aus der CSV-Datei in die Zieltabelle hochzuladen

Führen Sie die folgenden Schritte `cqlsh COPY FROM` aus, um den Befehl auszuführen.

1. Stellen Sie mithilfe von cqlsh eine Connect zu Amazon Keyspaces her.
2. Wählen Sie Ihren Keyspace mit dem folgenden Code.

```
USE catalog;
```

3. Stellen Sie die Schreibkonsistenz auf ein. `LOCAL_QUORUM` Um die Datenbeständigkeit zu gewährleisten, erlaubt Amazon Keyspaces keine anderen Einstellungen für die Schreibkonsistenz. Sehen Sie sich den folgenden Code an.

```
CONSISTENCY LOCAL_QUORUM;
```

4. Bereiten Sie Ihre `cqlsh COPY FROM` Syntax anhand des folgenden Codebeispiels vor.

```
COPY book_awards FROM './keyspace.table.csv' WITH HEADER=true
AND INGESTRATE=calculated ingestrate
AND NUMPROCESSES=calculated numprocess
AND MAXBATCHSIZE=20
AND CHUNKSIZE=calculated chunksize;
```

5. Führen Sie die im vorherigen Schritt vorbereitete Anweisung aus. cqlsh gibt alle Einstellungen zurück, die Sie konfiguriert haben.
   a. Stellen Sie sicher, dass die Einstellungen mit Ihrer Eingabe übereinstimmen. Sehen Sie sich das folgende Beispiel an.

```
Reading options from the command line: {'chunksize': '120', 'header': 'true',
 'ingestrate': '36000', 'numprocesses': '15', 'maxbatchsize': '20'}
```

```
Using 15 child processes
```

b. Überprüfen Sie die Anzahl der übertragenen Zeilen und die aktuelle Durchschnittsrate, wie im folgenden Beispiel gezeigt.

```
Processed: 57834 rows; Rate: 6561 rows/s; Avg. rate: 31751 rows/s
```

c. Wenn cqlsh das Hochladen der Daten abgeschlossen hat, überprüfen Sie die Zusammenfassung der Datenladestatistiken (Anzahl der gelesenen Dateien, Laufzeit und übersprungene Zeilen), wie im folgenden Beispiel gezeigt.

```
15556824 rows imported from 1 files in 8 minutes and 8.321 seconds (0 skipped).
```

In diesem letzten Schritt des Tutorials haben Sie die Daten auf Amazon Keyspaces hochgeladen.

> **Important**
>
> Nachdem Sie Ihre Daten übertragen haben, passen Sie die Einstellungen für den Kapazitätsmodus Ihrer Zieltabelle an die regulären Datenverkehrsmuster Ihrer Anwendung an. Es fallen Gebühren zum Stundensatz für Ihre bereitgestellte Kapazität an, bis Sie diese ändern.

Fehlerbehebung

Überprüfen Sie nach Abschluss des Datenuploads, ob Zeilen übersprungen wurden. Navigieren Sie dazu zum Quellverzeichnis der CSV-Quelldatei und suchen Sie nach einer Datei mit dem folgenden Namen.

```
import_yourcsvfilename.err.timestamp.csv
```

cqlsh schreibt alle übersprungenen Datenzeilen in eine Datei mit diesem Namen. Wenn die Datei in Ihrem Quellverzeichnis vorhanden ist und Daten enthält, wurden diese Zeilen nicht in Amazon Keyspaces hochgeladen. Um diese Zeilen erneut zu versuchen, überprüfen Sie zunächst, ob beim Upload Fehler aufgetreten sind, und passen Sie die Daten entsprechend an. Um diese Zeilen erneut zu versuchen, können Sie den Vorgang erneut ausführen.

Häufige Fehler

Die häufigsten Gründe, warum Zeilen nicht geladen werden, sind Kapazitätsfehler und Analysefehler.

Ungültige Anforderungsfehler beim Hochladen von Daten zu Amazon Keyspaces

Im folgenden Beispiel enthält die Quelltabelle eine Zählerspalte, die zu protokollierten Batch-Aufrufen des Befehls `COPY` cqlsh führt. Protokollierte Batch-Aufrufe werden von Amazon Keyspaces nicht unterstützt.

```
Failed to import 10 rows: InvalidRequest - Error from server: code=2200 [Invalid query]
 message="Only UNLOGGED Batches are supported at this time.",  will retry later,
 attempt 22 of 25
```

Um diesen Fehler zu beheben, verwenden Sie DSBulk, um die Daten zu migrieren. Weitere Informationen finden Sie unter the section called "Daten mit DSBulk laden".

Parser-Fehler beim Hochladen von Daten zu Amazon Keyspaces

Das folgende Beispiel zeigt eine übersprungene Zeile aufgrund von a. `ParseError`

```
Failed to import 1 rows: ParseError - Invalid ... –
```

Um diesen Fehler zu beheben, müssen Sie sicherstellen, dass die zu importierenden Daten dem Tabellenschema in Amazon Keyspaces entsprechen. Überprüfen Sie die Importdatei auf Analysefehler. Sie können versuchen, mithilfe einer `INSERT` Anweisung eine einzelne Datenzeile zu verwenden, um den Fehler zu isolieren.

Kapazitätsfehler beim Hochladen von Daten auf Amazon Keyspaces

```
Failed to import 1 rows: WriteTimeout - Error from server: code=1100 [Coordinator node
 timed out waiting for replica nodes' responses]
 message="Operation timed out - received only 0 responses." info={'received_responses':
 0, 'required_responses': 2, 'write_type': 'SIMPLE', 'consistency':
 'LOCAL_QUORUM'}, will retry later, attempt 1 of 100
```

Amazon Keyspaces verwendet die `WriteTimeout` Ausnahmen `ReadTimeout` und, um anzuzeigen, wenn eine Schreibanforderung aufgrund unzureichender Durchsatzkapazität fehlschlägt. Um bei der Diagnose von Ausnahmen bei unzureichender Kapazität zu helfen, veröffentlicht `WriteThrottleEvents` Amazon Keyspaces `ReadThrottledEvents` Statistiken in Amazon CloudWatch. Weitere Informationen finden Sie unter the section called "Überwachung mit CloudWatch".

cqlsh-Fehler beim Hochladen von Daten zu Amazon Keyspaces

Um Cqlsh-Fehler zu beheben, führen Sie den fehlgeschlagenen Befehl erneut mit der Markierung aus. `--debug`

Wenn Sie eine inkompatible Version von cqlsh verwenden, wird der folgende Fehler angezeigt.

```
AttributeError: 'NoneType' object has no attribute 'is_up'
Failed to import 3 rows: AttributeError - 'NoneType' object has no attribute 'is_up',
 given up after 1 attempts
```

Vergewissern Sie sich, dass die richtige Version von cqlsh installiert ist, indem Sie den folgenden Befehl ausführen.

```
cqlsh --version
```

Sie sollten etwa das Folgende als Ausgabe sehen.

```
cqlsh 5.0.1
```

Wenn Sie Windows verwenden, ersetzen Sie alle Instanzen von `cqlsh` durch`cqlsh.bat`. Um beispielsweise die Version von cqlsh in Windows zu überprüfen, führen Sie den folgenden Befehl aus.

```
cqlsh.bat --version
```

Die Verbindung zu Amazon Keyspaces schlägt fehl, nachdem der cqlsh-Client drei aufeinanderfolgende Fehler beliebiger Art vom Server empfangen hat. Der Cqlsh-Client schlägt mit der folgenden Meldung fehl.

```
Failed to import 1 rows: NoHostAvailable - , will retry later, attempt 3 of 100
```

Um diesen Fehler zu beheben, müssen Sie sicherstellen, dass die zu importierenden Daten dem Tabellenschema in Amazon Keyspaces entsprechen. Überprüfen Sie die Importdatei auf Analysefehler. Sie können versuchen, eine einzelne Datenzeile zu verwenden, indem Sie eine INSERT-Anweisung verwenden, um den Fehler zu isolieren.

Der Client versucht automatisch, die Verbindung wiederherzustellen.

Tutorial: Daten mit DSBulk in Amazon Keyspaces laden

Dieses Tutorial führt Sie Schritt für Schritt durch die Migration von Daten von Apache Cassandra zu Amazon Keyspaces mithilfe des DataStax Bulk Loaders (DSBulk), der auf verfügbar ist. GitHub Die Verwendung von DSBulk ist nützlich, um Datensätze für akademische Zwecke oder Testzwecke auf Amazon Keyspaces hochzuladen. Weitere Informationen zur Migration von Produktionsworkloads finden Sie unter. the section called “Offline-Migration” In diesem Tutorial führen Sie die folgenden Schritte aus.

Voraussetzungen — Richten Sie ein AWS Konto mit Anmeldeinformationen ein, erstellen Sie eine JKS-Trust-Store-Datei für das Zertifikat, konfigurieren Sie DSBulk`cqlsh`, laden Sie es herunter und installieren Sie es und konfigurieren Sie eine `application.conf` Datei.

1. Quell-CSV und Zieltabelle erstellen — Bereiten Sie eine CSV-Datei als Quelldaten vor und erstellen Sie den Zielschlüsselraum und die Zieltabelle in Amazon Keyspaces.
2. Daten vorbereiten — Randomisieren Sie die Daten in der CSV-Datei und analysieren Sie sie, um die durchschnittliche und maximale Zeilengröße zu ermitteln.
3. Durchsatzkapazität festlegen — Berechnen Sie die erforderlichen Schreibkapazitätseinheiten (WCUs) auf der Grundlage der Datengröße und der gewünschten Ladezeit und konfigurieren Sie die bereitgestellte Kapazität der Tabelle.
4. DSBulk-Einstellungen konfigurieren — Erstellen Sie eine DSBulk-Konfigurationsdatei mit Einstellungen wie Authentifizierung SSL/TLS, Konsistenzstufe und Größe des Verbindungspools.
5. Den Befehl dsBulk load ausführen — Führen Sie den Befehl dsBulk load aus, um die Daten aus der CSV-Datei in die Amazon Keyspaces-Tabelle hochzuladen und den Fortschritt zu überwachen.

Themen

- Voraussetzungen: Schritte, die Sie ausführen müssen, bevor Sie Daten mit DSBulk hochladen können
- Schritt 1: Erstellen Sie die Quell-CSV-Datei und eine Zieltabelle für den Datenupload mit DSBulk
- Schritt 2: Bereiten Sie die Daten für den Upload mit DSBulk vor
- Schritt 3: Stellen Sie die Durchsatzkapazität für die Zieltabelle ein
- Schritt 4: Konfigurieren Sie die DSBulk-Einstellungen, um Daten aus der CSV-Datei in die Zieltabelle hochzuladen
- Schritt 5: Führen Sie den Befehl dsBulk load aus, um Daten aus der CSV-Datei in die Zieltabelle hochzuladen

Voraussetzungen: Schritte, die Sie ausführen müssen, bevor Sie Daten mit DSBulk hochladen können

Sie müssen die folgenden Aufgaben erledigen, bevor Sie mit diesem Tutorial beginnen können.

1. Falls Sie dies noch nicht getan haben, registrieren Sie sich für ein AWS Konto, indem Sie den Schritten unter folgenthe section called “Amazon Keyspaces einrichten”.
2. Erstellen Sie Anmeldeinformationen, indem Sie den Schritten unter folgenthe section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.
3. Erstellen Sie eine JKS-Trust-Store-Datei.

   a. Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.

      1. AmazonRootCA1
      2. AmazonRootCA 2
      3. AmazonRootCA 3
      4. AmazonRootCA 4
      5. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

      Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

> **Note**
>
> Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2 CA verankert waren. AWS migriert alle AWS-Regionen auf Zertifikate, die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden. Während dieses Übergangs konfigurieren Sie die Clients so, dass sie sowohl den Amazon Root-CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

b. Konvertieren Sie die digitalen Zertifikate in TrustStore-Dateien und fügen Sie sie dem Keystore hinzu.

```
openssl x509 -outform der -in AmazonRootCA1.pem -out temp_file.der
keytool -import -alias amazon-root-ca-1 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA2.pem -out temp_file.der
keytool -import -alias amazon-root-ca-2 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA3.pem -out temp_file.der
keytool -import -alias amazon-root-ca-3 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA4.pem -out temp_file.der
keytool -import -alias amazon-root-ca-4 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in sf-class2-root.crt -out temp_file.der
keytool -import -alias cassandra -keystore cassandra_truststore.jks -file
 temp_file.der
```

Im letzten Schritt müssen Sie ein Passwort für den Keystore erstellen und jedem Zertifikat vertrauen. Der interaktive Befehl sieht so aus.

```
Enter keystore password:
Re-enter new password:
Owner: CN=Amazon Root CA 1, O=Amazon, C=US
Issuer: CN=Amazon Root CA 1, O=Amazon, C=US
Serial number: 66c9fcf99bf8c0a39e2f0788a43e696365bca
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sun Jan 17 00:00:00 UTC 2038
Certificate fingerprints:
     SHA1: 8D:A7:F9:65:EC:5E:FC:37:91:0F:1C:6E:59:FD:C1:CC:6A:6E:DE:16
     SHA256:
 8E:CD:E6:88:4F:3D:87:B1:12:5B:A3:1A:C3:FC:B1:3D:70:16:DE:7F:57:CC:90:4F:E1:CB:97:C6:AE
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:
```

```
#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 84 18 CC 85 34 EC BC 0C   94 94 2E 08 59 9C C7 B2  ....4.......Y...
0010: 10 4E 0A 08                                        .N..
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 2, O=Amazon, C=US
Issuer: CN=Amazon Root CA 2, O=Amazon, C=US
Serial number: 66c9fd29635869f0a0fe58678f85b26bb8a37
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
     SHA1: 5A:8C:EF:45:D7:A6:98:59:76:7A:8C:8B:44:96:B5:78:CF:47:4B:1A
     SHA256:
 1B:A5:B2:AA:8C:65:40:1A:82:96:01:18:F8:0B:EC:4F:62:30:4D:83:CE:C4:71:3A:19:C3:9C:01:1E
Signature algorithm name: SHA384withRSA
Subject Public Key Algorithm: 4096-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]
```

```
#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: B0 0C F0 4C 30 F4 05 58   02 48 FD 33 E5 52 AF 4B  ...L0..X.H.3.R.K
0010: 84 E3 66 52                                        ..fR
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 3, O=Amazon, C=US
Issuer: CN=Amazon Root CA 3, O=Amazon, C=US
Serial number: 66c9fd5749736663f3b0b9ad9e89e7603f24a
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
     SHA1: 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E
     SHA256:
 18:CE:6C:FE:7B:F1:4E:60:B2:E3:47:B8:DF:E8:68:CB:31:D0:2E:BB:3A:DA:27:15:69:F5:03:43:B4
Signature algorithm name: SHA256withECDSA
Subject Public Key Algorithm: 256-bit EC (secp256r1) key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]
```

```
#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: AB B6 DB D7 06 9E 37 AC   30 86 07 91 70 C7 9C C4  ......7.0...p...
0010: 19 B1 78 C0                                        ..x.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 4, O=Amazon, C=US
Issuer: CN=Amazon Root CA 4, O=Amazon, C=US
Serial number: 66c9fd7c1bb104c2943e5717b7b2cc81ac10e
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
     SHA1: F6:10:84:07:D6:F8:BB:67:98:0C:C2:E2:44:C2:EB:AE:1C:EF:63:BE
     SHA256:
 E3:5D:28:41:9E:D0:20:25:CF:A6:90:38:CD:62:39:62:45:8D:A5:C6:95:FB:DE:A3:C2:2B:0B:FB:25
Signature algorithm name: SHA384withECDSA
Subject Public Key Algorithm: 384-bit EC (secp384r1) key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: D3 EC C7 3A 65 6E CC E1   DA 76 9A 56 FB 9C F3 86  ...:en...v.V....
0010: 6D 57 E5 81                                        mW..
```

```
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US
Issuer: OU=Starfield Class 2 Certification Authority, O="Starfield
 Technologies, Inc.", C=US
Serial number: 0
Valid from: Tue Jun 29 17:39:16 UTC 2004 until: Thu Jun 29 17:39:16 UTC 2034
Certificate fingerprints:
     SHA1: AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A
     SHA256:
 14:65:FA:20:53:97:B8:76:FA:A6:F0:A9:95:8E:55:90:E4:0F:CC:7F:AA:4F:B7:C2:C8:67:75:21:FB
Signature algorithm name: SHA1withRSA (weak)
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
[OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US]
SerialNumber: [    00]
]

#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
```

```
]
]


Warning:
The input uses the SHA1withRSA signature algorithm which is considered a
 security risk. This algorithm will be disabled in a future update.

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

4. Richten Sie die Cassandra Query Language Shell (cqlsh) -Verbindung ein und bestätigen Sie, dass Sie eine Verbindung zu Amazon Keyspaces herstellen können, indem Sie die Schritte unter befolgen. the section called “Verwenden von `cqlsh`”
5. Laden Sie DSBulk herunter und installieren Sie es.

> **Note**
>
> Die in diesem Tutorial gezeigte Version ist möglicherweise nicht die neueste verfügbare Version. Bevor Sie DSBulk herunterladen, suchen Sie auf der DataStax Bulk Loader-Downloadseite nach der neuesten Version und aktualisieren Sie die Versionsnummer in den folgenden Befehlen entsprechend.

   a. Um DSBulk herunterzuladen, können Sie den folgenden Code verwenden.

```
curl -OL https://downloads.datastax.com/dsbulk/dsbulk-1.8.0.tar.gz
```

   b. Entpacken Sie dann die TAR-Datei und fügen Sie DSBulk zu Ihrer hinzu, `PATH` wie im folgenden Beispiel gezeigt.

```
tar -zxvf dsbulk-1.8.0.tar.gz
# add the DSBulk directory to the path
export PATH=$PATH:./dsbulk-1.8.0/bin
```

   c. Erstellen Sie eine `application.conf` Datei zum Speichern von Einstellungen, die von DSBulk verwendet werden sollen. Sie können das folgende Beispiel unter speichern. `./dsbulk_keyspaces.conf` `localhost`Ersetzen Sie es durch den Kontaktpunkt Ihres lokalen Cassandra-Clusters, wenn Sie sich nicht auf dem lokalen Knoten befinden, z. B. den

DNS-Namen oder die IP-Adresse. Notieren Sie sich den Dateinamen und den Pfad, da Sie dies später im `dsbulk load` Befehl angeben müssen.

```
datastax-java-driver {
  basic.contact-points = [ "localhost"]
  advanced.auth-provider {
        class = software.aws.mcs.auth.SigV4AuthProvider
        aws-region = us-east-1
  }
}
```

d. Um die SigV4-Unterstützung zu aktivieren, laden Sie die schattierte `jar` Datei von herunter GitHubund platzieren Sie sie im `lib` Ordner DSBulk, wie im folgenden Beispiel gezeigt.

```
curl -O -L https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin/
releases/download/4.0.6-shaded-v2/aws-sigv4-auth-cassandra-java-driver-
plugin-4.0.6-shaded.jar
```

Schritt 1: Erstellen Sie die Quell-CSV-Datei und eine Zieltabelle für den Datenupload mit DSBulk

Für dieses Tutorial verwenden wir eine Datei mit kommagetrennten Werten (CSV) mit dem Namen `keyspaces_sample_table.csv` als Quelldatei für die Datenmigration. Die mitgelieferte Beispieldatei enthält einige Datenzeilen für eine Tabelle mit dem Namen. `book_awards`

1. Erstellen Sie die Quelldatei. Sie können eine der folgenden Optionen wählen:
   - Laden Sie die CSV-Beispieldatei (`keyspaces_sample_table.csv`) herunter, die in der folgenden Archivdatei samplemigration.zip enthalten ist. Entpacken Sie das Archiv und notieren Sie sich den Pfad zu`keyspaces_sample_table.csv`.
   - Um eine CSV-Datei mit Ihren eigenen Daten zu füllen, die in einer Apache Cassandra-Datenbank gespeichert sind, können Sie die CSV-Quelldatei `dsbulk unload` wie im folgenden Beispiel gezeigt auffüllen.

     ```
     dsbulk unload -k mykeyspace -t mytable -f ./my_application.conf
      > keyspaces_sample_table.csv
     ```

     Stellen Sie sicher, dass die von Ihnen erstellte CSV-Datei die folgenden Anforderungen erfüllt:

- Die erste Zeile enthält die Spaltennamen.
- Die Spaltennamen in der CSV-Quelldatei stimmen mit den Spaltennamen in der Zieltabelle überein.
- Die Daten sind durch ein Komma getrennt.
- Alle Datenwerte sind gültige Amazon Keyspaces-Datentypen. Siehe the section called “Datentypen”.

2. Erstellen Sie den Zielschlüsselraum und die Zieltabelle in Amazon Keyspaces.

   a. Connect zu Amazon Keyspaces her`cqlsh`, indem Sie den Service-Endpunkt, den Benutzernamen und das Passwort im folgenden Beispiel durch Ihre eigenen Werte ersetzen.

   ```
   cqlsh cassandra.us-east-1.amazonaws.com 9142 -u "111122223333" -p
    "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
   ```

   b. Erstellen Sie einen neuen Schlüsselraum mit dem Namen, `catalog` wie im folgenden Beispiel gezeigt.

   ```
   CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
   ```

   c. Wenn der neue Schlüsselraum den Status verfügbar hat, verwenden Sie den folgenden Code, um die Zieltabelle zu erstellen. `book_awards` Weitere Informationen zur asynchronen Ressourcenerstellung und zur Überprüfung, ob eine Ressource verfügbar ist, finden Sie unter. the section called “Überprüfen Sie den Status der Keyspace-Erstellung”

   ```
   CREATE TABLE catalog.book_awards (
      year int,
      award text,
      rank int,
      category text,
      book_title text,
      author text,
      publisher text,
      PRIMARY KEY ((year, award), category, rank)
      );
   ```

Wenn Apache Cassandra Ihre ursprüngliche Datenquelle ist, besteht eine einfache Möglichkeit, die Amazon Keyspaces-Zieltabelle mit passenden Headern zu erstellen, darin, die `CREATE TABLE` Anweisung aus der Quelltabelle zu generieren, wie in der folgenden Anweisung gezeigt.

```
cqlsh localhost 9042  -u "username" -p "password" --execute "DESCRIBE
 TABLE mykeyspace.mytable;"
```

Erstellen Sie dann die Zieltabelle in Amazon Keyspaces mit den Spaltennamen und Datentypen, die der Beschreibung aus der Cassandra-Quelltabelle entsprechen.

Schritt 2: Bereiten Sie die Daten für den Upload mit DSBulk vor

Die Vorbereitung der Quelldaten für eine effiziente Übertragung erfolgt in zwei Schritten. Zunächst randomisieren Sie die Daten. Im zweiten Schritt analysieren Sie die Daten, um die geeigneten `dsbulk` Parameterwerte und erforderlichen Tabelleneinstellungen zu ermitteln.

Randomisieren Sie die Daten

`dsbulk`Mit dem Befehl werden Daten in derselben Reihenfolge gelesen und geschrieben, in der sie in der CSV-Datei erscheinen. Wenn Sie den `dsbulk` Befehl verwenden, um die Quelldatei zu erstellen, werden die Daten in schlüsselsortierter Reihenfolge in die CSV-Datei geschrieben. Intern partitioniert Amazon Keyspaces Daten mithilfe von Partitionsschlüsseln. Amazon Keyspaces verfügt zwar über eine integrierte Logik, die beim Lastenausgleich von Anfragen für denselben Partitionsschlüssel hilft, das Laden der Daten ist jedoch schneller und effizienter, wenn Sie die Reihenfolge nach dem Zufallsprinzip festlegen. Dies liegt daran, dass Sie den integrierten Lastenausgleich nutzen können, der auftritt, wenn Amazon Keyspaces auf verschiedene Partitionen schreibt.

Um die Schreibvorgänge gleichmäßig auf die Partitionen zu verteilen, müssen Sie die Daten in der Quelldatei randomisieren. Sie können dafür eine Anwendung schreiben oder ein Open-Source-Tool wie Shuf verwenden. Shuf ist auf Linux-Distributionen, auf macOS (durch Installation von Coreutils in Homebrew) und unter Windows (mithilfe des Windows Subsystems for Linux (WSL)) kostenlos verfügbar. Ein zusätzlicher Schritt ist erforderlich, um zu verhindern, dass die Kopfzeile mit den Spaltennamen in diesem Schritt gemischt wird.

Um die Quelldatei nach dem Zufallsprinzip zu sortieren und gleichzeitig die Kopfzeile beizubehalten, geben Sie den folgenden Code ein.

```
tail -n +2 keyspaces_sample_table.csv | shuf -o keyspace.table.csv && (head
 -1 keyspaces_sample_table.csv && cat keyspace.table.csv ) > keyspace.table.csv1 &&
 mv keyspace.table.csv1 keyspace.table.csv
```

Shuf schreibt die Daten in eine neue CSV-Datei mit dem Namen um. `keyspace.table.csv` Sie können die `keyspaces_sample_table.csv` Datei jetzt löschen — Sie benötigen sie nicht mehr.

Analysieren Sie die Daten

Ermitteln Sie die durchschnittliche und maximale Zeilengröße, indem Sie die Daten analysieren.

Sie tun dies aus den folgenden Gründen:

- Die durchschnittliche Zeilengröße hilft bei der Schätzung der Gesamtmenge der zu übertragenden Daten.
- Sie benötigen die durchschnittliche Zeilengröße, um die für den Datenupload benötigte Schreibkapazität bereitzustellen.
- Sie können sicherstellen, dass jede Zeile weniger als 1 MB groß ist. Dies ist die maximale Zeilengröße in Amazon Keyspaces.

> **Note**
>
> Dieses Kontingent bezieht sich auf die Zeilengröße, nicht auf die Partitionsgröße. Im Gegensatz zu Apache Cassandra-Partitionen können Amazon Keyspaces-Partitionen praktisch unbegrenzt groß sein. Partitionsschlüssel und Clusterspalten benötigen zusätzlichen Speicherplatz für Metadaten, den Sie zur Rohgröße der Zeilen hinzufügen müssen. Weitere Informationen finden Sie unter the section called “Schätzen Sie die Zeilengröße”.

Der folgende Code verwendet AWK, um eine CSV-Datei zu analysieren und die durchschnittliche und maximale Zeilengröße zu drucken.

```
awk -F, 'BEGIN {samp=10000;max=-1;}{if(NR>1){len=length($0);t+=len;avg=t/
NR;max=(len>max ? len : max)}}NR==samp{exit}END{printf("{lines: %d, average: %d bytes,
 max: %d bytes}\n",NR,avg,max);}' keyspace.table.csv
```

Die Ausführung dieses Codes führt zu der folgenden Ausgabe.

```
using 10,000 samples:
{lines: 10000, avg: 123 bytes, max: 225 bytes}
```

Stellen Sie sicher, dass Ihre maximale Zeilengröße 1 MB nicht überschreitet. Ist dies der Fall, müssen Sie die Zeile aufteilen oder die Daten komprimieren, um die Zeilengröße unter 1 MB zu bringen. Im nächsten Schritt dieses Tutorials verwenden Sie die durchschnittliche Zeilengröße, um die Schreibkapazität für die Tabelle bereitzustellen.

Schritt 3: Stellen Sie die Durchsatzkapazität für die Zieltabelle ein

Dieses Tutorial zeigt Ihnen, wie Sie DSBulk so einstellen, dass Daten innerhalb eines festgelegten Zeitbereichs geladen werden. Da Sie im Voraus wissen, wie viele Lese- und Schreibvorgänge Sie durchführen, sollten Sie den Modus für bereitgestellte Kapazität verwenden. Nachdem Sie die Datenübertragung abgeschlossen haben, sollten Sie den Kapazitätsmodus der Tabelle so einstellen, dass er den Datenverkehrsmustern Ihrer Anwendung entspricht. Weitere Informationen zur Kapazitätsverwaltung finden Sie unterVerwaltung serverloser Ressourcen.

Im Modus „Bereitgestellte Kapazität“ geben Sie im Voraus an, wie viel Lese- und Schreibkapazität Sie für Ihre Tabelle bereitstellen möchten. Die Schreibkapazität wird stündlich abgerechnet und in Schreibkapazitätseinheiten (WCUs) gemessen. Jede WCU bietet ausreichend Schreibkapazität, um das Schreiben von 1 KB Daten pro Sekunde zu unterstützen. Wenn Sie die Daten laden, muss die Schreibrate unter den maximalen WCUs (Parameter:`write_capacity_units`) liegen, die in der Zieltabelle festgelegt sind.

Standardmäßig können Sie bis zu 40.000 WCUs für eine Tabelle und 80.000 WCUs für alle Tabellen in Ihrem Konto bereitstellen. Wenn Sie zusätzliche Kapazität benötigen, können Sie in der Service Quotas-Konsole eine Erhöhung des Kontingents beantragen. Weitere Informationen zu Kontingenten finden Sie unter Kontingente.

Berechnen Sie die durchschnittliche Anzahl der WCUs, die für eine Einfügung erforderlich sind

Für das Einfügen von 1 KB Daten pro Sekunde ist 1 WCU erforderlich. Wenn Ihre CSV-Datei 360.000 Zeilen hat und Sie alle Daten in einer Stunde laden möchten, müssen Sie 100 Zeilen pro Sekunde schreiben (360.000 Zeilen/60 Minuten/ 60 Sekunden = 100 Zeilen pro Sekunde). Wenn jede Zeile bis zu 1 KB Daten enthält, müssen Sie 100 WCUs für Ihre Tabelle bereitstellen, um 100 Zeilen pro Sekunde einzufügen. Wenn jede Zeile 1,5 KB Daten enthält, benötigen Sie zwei WCUs, um eine Zeile pro Sekunde einzufügen. Um 100 Zeilen pro Sekunde einzufügen, müssen Sie daher 200 WCUs bereitstellen.

Um zu ermitteln, wie viele WCUs Sie für das Einfügen einer Zeile pro Sekunde benötigen, teilen Sie die durchschnittliche Zeilengröße in Byte durch 1024 und runden Sie auf die nächste ganze Zahl auf.

Wenn die durchschnittliche Zeilengröße beispielsweise 3000 Byte beträgt, benötigen Sie drei WCUs, um eine Zeile pro Sekunde einzufügen.

```
ROUNDUP(3000 / 1024) = ROUNDUP(2.93) = 3 WCUs
```

Berechnen Sie die Ladezeit und Kapazität der Daten

Da Sie nun die durchschnittliche Größe und Anzahl der Zeilen in Ihrer CSV-Datei kennen, können Sie berechnen, wie viele WCUs Sie benötigen, um die Daten in einem bestimmten Zeitraum zu laden, und wie lange es ungefähr dauert, alle Daten in Ihrer CSV-Datei mithilfe verschiedener WCU-Einstellungen zu laden.

Wenn beispielsweise jede Zeile in Ihrer Datei 1 KB groß ist und Sie 1.000.000 Zeilen in Ihrer CSV-Datei haben, müssen Sie für diese Stunde mindestens 278 WCUs für Ihre Tabelle bereitstellen, um die Daten in einer Stunde zu laden.

```
1,000,000 rows * 1 KBs = 1,000,000 KBs
1,000,000 KBs / 3600 seconds =277.8 KBs / second = 278 WCUs
```

Konfigurieren Sie die Einstellungen für die bereitgestellte Kapazität

Sie können die Schreibkapazitätseinstellungen einer Tabelle festlegen, wenn Sie die Tabelle erstellen oder den `ALTER TABLE` Befehl verwenden. Im Folgenden finden Sie die Syntax für das Ändern der bereitgestellten Kapazitätseinstellungen einer Tabelle mit dem `ALTER TABLE` Befehl.

```
ALTER TABLE catalog.book_awards WITH custom_properties={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 100, 'write_capacity_units':
 278}} ;
```

Die vollständige Sprachreferenz finden Sie unter the section called “CREATE TABLE” und. the section called “ALTER TABLE”

Schritt 4: Konfigurieren Sie die **DSBulk-Einstellungen**, um Daten aus der CSV-Datei in die Zieltabelle hochzuladen

In diesem Abschnitt werden die Schritte beschrieben, die erforderlich sind, um DSBulk für das Hochladen von Daten auf Amazon Keyspaces zu konfigurieren. Sie konfigurieren DSBulk mithilfe einer Konfigurationsdatei. Sie geben die Konfigurationsdatei direkt von der Befehlszeile aus an.

1. Erstellen Sie eine DSBulk-Konfigurationsdatei für die Migration zu Amazon Keyspaces. In diesem Beispiel verwenden wir den Dateinamen. `dsbulk_keyspaces.conf` Geben Sie die folgenden Einstellungen in der DSBulk-Konfigurationsdatei an.

   a. *`PlainTextAuthProvider`*— Erstellen Sie den Authentifizierungsanbieter mit der `PlainTextAuthProvider` Klasse. `ServiceUserName`und `ServicePassword` sollte mit dem Benutzernamen und dem Passwort übereinstimmen, die Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben, indem Sie die Schritte unter the section called “Erstellen Sie programmatische Zugangsdaten” ausführen.

   b. *`local-datacenter`*— Setzen Sie den Wert für `local-datacenter` auf den AWS-Region , zu dem Sie eine Verbindung herstellen. Wenn die Anwendung beispielsweise eine Verbindung herstellt`cassandra.us-east-1.amazonaws.com`, stellen Sie das lokale Rechenzentrum auf ein`us-east-1`. Alle verfügbaren AWS-Regionen Informationen finden Sie unterthe section called “Service-Endpunkte”. Um Replikate zu vermeiden, legen Sie den Wert `slow-replica-avoidance` auf `false` fest.

   c. *`SSLEngineFactory`*— Zur Konfiguration initialisieren Sie den SSL/TLS, `SSLEngineFactory` indem Sie der Konfigurationsdatei einen Abschnitt mit einer einzigen Zeile hinzufügen, in der die Klasse mit angegeben wird. `class = DefaultSslEngineFactory` Geben Sie den Pfad `cassandra_truststore.jks` und das Passwort an, die Sie zuvor erstellt haben.

   d. *`consistency`*— Stellen Sie die Konsistenzstufe auf ein`LOCAL_QUORUM`. Andere Schreibkonsistenzstufen werden nicht unterstützt. Weitere Informationen finden Sie unterthe section called “Unterstützte Cassandra-Konsistenzstufen”.

   e. Die Anzahl der Verbindungen pro Pool ist im Java-Treiber konfigurierbar. Stellen Sie in diesem Beispiel `advanced.connection.pool.local.size` den Wert 3 ein.

   Im Folgenden finden Sie die vollständige Beispielkonfigurationsdatei.

```
datastax-java-driver {
```

```
basic.contact-points = [ "cassandra.us-east-1.amazonaws.com:9142"]
advanced.auth-provider {
    class = PlainTextAuthProvider
    username = "ServiceUserName"
    password = "ServicePassword"
}

basic.load-balancing-policy {
    local-datacenter = "us-east-1"
    slow-replica-avoidance = false
}

basic.request {
    consistency = LOCAL_QUORUM
    default-idempotence = true
}
advanced.ssl-engine-factory {
    class = DefaultSslEngineFactory
    truststore-path = "./cassandra_truststore.jks"
    truststore-password = "my_password"
    hostname-validation = false
  }
advanced.connection.pool.local.size = 3
}
```

2. Überprüfen Sie die Parameter für den `load` Befehl DSBulk.

   a. *`executor.maxPerSecond`*— Die maximale Anzahl von Zeilen, die der Ladebefehl pro Sekunde gleichzeitig zu verarbeiten versucht. Wenn sie nicht gesetzt ist, ist diese Einstellung mit -1 deaktiviert.

      Wird auf der `executor.maxPerSecond` Grundlage der Anzahl der WCUs festgelegt, die Sie für die Zieltabelle bereitgestellt haben. Der `executor.maxPerSecond` Wert des `load` Befehls ist kein Limit, sondern ein Zieldurchschnitt. Das bedeutet, dass er die von Ihnen festgelegte Zahl überschreiten kann (und tut dies häufig auch). Um Bursts zu berücksichtigen und sicherzustellen, dass genügend Kapazität zur Bearbeitung der Datenladeanforderungen vorhanden ist, sollten Sie einen Wert von 90% der Schreibkapazität der Tabelle festlegen`executor.maxPerSecond`.

      ```
      executor.maxPerSecond = WCUs * .90
      ```

In diesem Tutorial haben wir den Wert 5 `executor.maxPerSecond` eingestellt.

> **Note**
>
> Wenn Sie DSBulk 1.6.0 oder höher verwenden, können Sie stattdessen verwendendsbulk.engine.maxConcurrentQueries.

b. Konfigurieren Sie diese zusätzlichen Parameter für den `load` DSBulk-Befehl.

- *batch-mode*— Dieser Parameter weist das System an, Operationen nach Partitionsschlüsseln zu gruppieren. Wir empfehlen, den Batch-Modus zu deaktivieren, da dies zu Hotkey-Szenarien und Ursachen führen kannWriteThrottleEvents.
- *driver.advanced.retry-policy-max-retries*— Dies bestimmt, wie oft eine fehlgeschlagene Abfrage wiederholt werden muss. Wenn diese Option nicht gesetzt ist, ist die Standardeinstellung 10. Sie können diesen Wert nach Bedarf anpassen.
- *driver.basic.request.timeout*— Die Zeit in Minuten, in der das System auf die Rückgabe einer Abfrage wartet. Wenn diese Option nicht gesetzt ist, ist die Standardeinstellung „5 Minuten“. Sie können diesen Wert nach Bedarf anpassen.

Schritt 5: Führen Sie den Befehl dsBulk **`load`** aus, um Daten aus der CSV-Datei in die Zieltabelle hochzuladen

Im letzten Schritt dieses Tutorials laden Sie die Daten in Amazon Keyspaces hoch.

Führen Sie die folgenden Schritte aus, um den `load` DSBulk-Befehl auszuführen.

1. Führen Sie den folgenden Code aus, um die Daten aus Ihrer CSV-Datei in Ihre Amazon Keyspaces-Tabelle hochzuladen. Achten Sie darauf, den Pfad zur Anwendungskonfigurationsdatei zu aktualisieren, die Sie zuvor erstellt haben.

```
dsbulk load -f ./dsbulk_keyspaces.conf  --connector.csv.url keyspace.table.csv
 -header true --batch.mode DISABLED --executor.maxPerSecond 5 --
driver.basic.request.timeout "5 minutes" --driver.advanced.retry-policy.max-
retries 10 -k catalog -t book_awards
```

2. Die Ausgabe enthält den Speicherort einer Protokolldatei, in der erfolgreiche und erfolglose Vorgänge aufgeführt sind. Die Datei wird im folgenden Verzeichnis gespeichert.

```
Operation directory: /home/user_name/logs/UNLOAD_20210308-202317-801911
```

3. Die Einträge in der Protokolldatei werden Metriken enthalten, wie im folgenden Beispiel. Stellen Sie sicher, dass die Anzahl der Zeilen mit der Anzahl der Zeilen in Ihrer CSV-Datei übereinstimmt.

```
total | failed | rows/s | p50ms | p99ms | p999ms
   200 |      0 |    200 | 21.63 | 21.89 |  21.89
```

> **Important**
>
> Nachdem Sie Ihre Daten übertragen haben, passen Sie die Einstellungen für den Kapazitätsmodus Ihrer Zieltabelle an die regulären Datenverkehrsmuster Ihrer Anwendung an. Es fallen Gebühren zum Stundensatz für Ihre bereitgestellte Kapazität an, bis Sie diese ändern. Weitere Informationen finden Sie unter the section called “ read/write Kapazitätsmodi konfigurieren”.

Zugreifen auf Amazon Keyspaces (für Apache Cassandra)

Sie können über die Konsole auf Amazon Keyspaces zugreifen, programmgesteuert AWS CloudShell, indem Sie einen `cqlsh` Client, das AWS SDK ausführen oder einen für Apache 2.0 lizenzierten Cassandra-Treiber verwenden. Amazon Keyspaces unterstützt Treiber und Clients, die mit Apache Cassandra 3.11.2 kompatibel sind. Bevor Sie auf Amazon Keyspaces zugreifen können, müssen Sie die Einrichtung abschließen AWS Identity and Access Management und dann Amazon Keyspaces eine IAM-Identität gewähren.

Melden Sie sich an für ein AWS-Konto

Um loszulegen AWS, benötigen Sie eine AWS-Konto. Informationen zum Erstellen eines AWS-Konto finden Sie unter Erste Schritte mit einem AWS-Konto im AWS -Kontenverwaltung Referenzhandbuch.

Amazon Keyspaces einrichten

Der Zugriff auf Amazon Keyspaces-Ressourcen wird mithilfe von IAM verwaltet. Mithilfe von IAM können Sie Richtlinien an IAM-Benutzer, -Rollen und föderierte Identitäten anhängen, die Lese- und Schreibberechtigungen für bestimmte Ressourcen in Amazon Keyspaces gewähren.

Um mit der Erteilung von Berechtigungen für eine IAM-Identität zu beginnen, können Sie eine der AWS verwalteten Richtlinien für Amazon Keyspaces verwenden:

- AmazonKeyspacesFullAccess— Diese Richtlinie gewährt Berechtigungen für den Zugriff auf alle Ressourcen in Amazon Keyspaces mit vollem Zugriff auf alle Funktionen.
- AmazonKeyspacesReadOnlyAccess_v2 — Diese Richtlinie gewährt Amazon Keyspaces nur Leseberechtigungen.

Eine ausführliche Erläuterung der in den verwalteten Richtlinien definierten Aktionen finden Sie unter. the section called “AWS verwaltete Richtlinien”

Um den Umfang der Aktionen, die eine IAM-Identität ausführen kann, oder die Ressourcen einzuschränken, auf die die Identität zugreifen kann, können Sie eine benutzerdefinierte Richtlinie erstellen, die die `AmazonKeyspacesFullAccess` verwaltete Richtlinie als Vorlage verwendet, und alle nicht benötigten Berechtigungen entfernen. Sie können den Zugriff auch auf bestimmte Schlüsselbereiche oder Tabellen einschränken. Weitere Informationen darüber, wie Sie Aktionen

einschränken oder den Zugriff auf bestimmte Ressourcen in Amazon Keyspaces einschränken können, finden Sie unterthe section called “So funktioniert Amazon Keyspaces mit IAM”.

Um auf Amazon Keyspaces zuzugreifen, nachdem Sie die Richtlinie erstellt AWS-Konto und erstellt haben, die einer IAM-Identität Zugriff auf Amazon Keyspaces gewährt, fahren Sie mit einem der folgenden Abschnitte fort:

- Verwenden der Konsole
- Verwenden AWS CloudShell

Zugreifen auf Amazon Keyspaces über die Konsole

Sie können auf die Konsole für Amazon Keyspaces unter https://console.aws.amazon.com/keyspaces/home zugreifen. Weitere Informationen zum AWS-Managementkonsole Zugriff finden Sie unter Steuern des Zugriffs von IAM-Benutzern auf das AWS-Managementkonsole im IAM-Benutzerhandbuch.

Sie können die Konsole verwenden, um in Amazon Keyspaces Folgendes zu tun:

- Schlüsselräume und Tabellen erstellen, löschen und verwalten.
- Überwachen Sie wichtige Tabellenmetriken auf der Registerkarte Überwachen einer Tabelle:
  - Größe der fakturierbaren Tabelle (Byte)
  - Kapazitätskennzahlen
- Führen Sie Abfragen mit dem CQL-Editor oder über die integrierte AWS CloudShell Benutzeroberfläche aus, z. B. Daten einfügen, aktualisieren und löschen.
- Stellen Sie über die integrierte CQLSH-Unterstützung eine direkte Connect AWS CloudShell zu Amazon Keyspaces her.
- Ändern Sie die Partitionierungskonfiguration des Kontos.
- Sehen Sie sich die Leistungs- und Fehlermetriken für das Konto im Dashboard an.

Connect zu Amazon Keyspaces her über AWS CloudShell von der Konsole

AWS CloudShell bietet eine optimierte Möglichkeit, direkt von der Konsole aus eine Verbindung zu Amazon Keyspaces herzustellen. Die AWS CloudShell Integration initialisiert das automatisch `cqlsh-expansion` für Sie und stellt eine Verbindung zu einem bestimmten Schlüsselraum her.

Erste Schritte mit AWS CloudShell Integration

1. Auf die AWS CloudShell Integration zugreifen — Navigieren Sie in der Amazon Keyspaces-Konsole zu dem Keyspace, zu dem Sie Connect möchten, und wählen Sie die Schaltfläche Mit AWS CloudShell Keyspaces verbinden.
2. Pre-filled Befehl setup — Wenn Sie auf die AWS CloudShell Schaltfläche klicken, stellt das System einen vorausgefüllten Befehl bereit, der die erforderlichen Initialisierungs- und Verbindungsparameter enthält:

   ```
   cqlsh-expansion.init ; cqlsh-expansion cassandra.region.amazonaws.com 9142 --ssl
    -my_keyspace
   ```

3. Klicken Sie auf Ausführen
4. Warten Sie, bis Amazon Keyspaces die Verbindung hergestellt hat — Warten Sie, bis der Einrichtungsvorgang abgeschlossen und die Verbindung zu Ihrem ausgewählten Keyspace hergestellt ist.
5. Befehle ausführen — Wenn eine Verbindung hergestellt ist, können Sie CQL-Befehle direkt für den ausgewählten Schlüsselraum schreiben und ausführen.

Verfügbare CQLSH-Befehle

Die AWS CloudShell Integration bietet Zugriff auf die Standardfunktionen von CQLSH, einschließlich der Unterstützung von Befehlen, die über die grundlegende CQL-Funktionalität hinausgehen.

- Beschreiben — Listet detaillierte Beschreibungen von Schlüsselräumen und Tabellen auf und zeigt sie an.
- Kopieren — Exportiert die Abfrageausgabe zur späteren Verarbeitung in eine Datei.
- Quelle — Führt CQL-Anweisungen aus einer Datei aus.
- Navigation im Befehlsverlauf — Scrollen Sie mithilfe von Tastenkombinationen durch vorherige Befehle.
- Ausführung von Abfragen — Führen Sie Abfragen direkt über die Tastaturschnittstelle aus.
- Kopieren von Ausgaben — Kopiert die Ergebnisse als einfachen Text zur Verwendung in anderen Anwendungen.

Überlegungen zu Funktionen

Beachten Sie bei der Verwendung der AWS CloudShell Integration die folgenden Kompromisse.

Die folgenden Funktionen werden nicht unterstützt:

- Vorschläge zur CQL-Syntax und Funktionen zur automatischen Vervollständigung.
- Editor-Tools wie Autovervollständigung anzeigen, QuickInfo anzeigen, Suchen, Suchen und Ersetzen, Wiederherstellen, Umschalten von Blockkommentaren, Umschalten des Codes, Umschalten von Zeilenkommentaren und Rückgängig machen.

Geänderte Funktionalität:

- CSV-Downloads — Um Ergebnisse im CSV-Format herunterzuladen, geben Sie den > Operator an, der die Befehlsausgabe weiterleitet:

```
SELECT * FROM my_table > file_name.csv
```

- JSON-Anzeige — Sie können Elemente im JSON-Format anzeigen, ohne dass eine Datenkonvertierung erforderlich ist.
- Terminal-style Schnittstelle — Das Erlebnis funktioniert eher als Terminalschnittstelle als als Code-Editor mit vollem Funktionsumfang.

Die AWS CloudShell Integration vereinfacht den Verbindungsprozess und bietet gleichzeitig wichtige CQL-Funktionen für die Verwaltung Ihrer Amazon Keyspaces-Daten direkt vom. AWS-Managementkonsole

Informationen dazu, wie Sie einen Amazon Keyspaces-Schlüsselraum und eine Tabelle erstellen und mit Beispielanwendungsdaten einrichten, finden Sie unter. Erste Schritte mit Amazon Keyspaces (für Apache Cassandra)

Für AWS CloudShell den Zugriff auf Amazon Keyspaces verwenden

AWS CloudShell ist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt von der aus starten können. AWS-Managementkonsole Sie können AWS CLI Befehle für AWS Dienste mit Ihrer bevorzugten Shell (Bash PowerShell oder Z-Shell) ausführen. AWS CloudShell wird mit Ihren

Konsolenanmeldedaten vorauthentifiziert geliefert und enthält gängige Tools für die Arbeit mit AWS Diensten AWS CLI, darunter Python und. Node.js

Sie starten AWS CloudShell von der aus AWS-Managementkonsole, und die AWS Anmeldeinformationen, mit denen Sie sich an der Konsole angemeldet haben, sind automatisch in einer neuen Shell-Sitzung verfügbar. Diese Vorauthentifizierung von AWS CloudShell Benutzern ermöglicht es Ihnen, die Konfiguration von Anmeldeinformationen zu überspringen, wenn Sie mit AWS Diensten wie Amazon Keyspaces interagieren, die `cqlsh` oder AWS CLI Version 2 (in der Computerumgebung der Shell vorinstalliert) verwenden.

AWS CloudShell Integrationsoptionen

Amazon Keyspaces bietet zwei AWS CloudShell Nutzungsmöglichkeiten:

1. Integriertes AWS CloudShell Erlebnis — Eine optimierte Verbindungsmethode, die direkt in der Amazon Keyspaces-Konsole verfügbar ist und die automatisch initialisiert `cqlsh-expansion` und eine Verbindung zu einem bestimmten Schlüsselraum herstellt. Vollständige Anweisungen zur Nutzung dieses integrierten Erlebnisses finden Sie unter. the section called “CloudShell Konsolenintegration”
2. Manuelles AWS CloudShell Setup — Manuelles Installieren und Konfigurieren `cqlsh` in AWS CloudShell. Die Schritte werden in den folgenden Abschnitten beschrieben und bieten mehr Kontrolle über den Einrichtungsprozess und ermöglichen benutzerdefinierte Konfigurationen.

Voraussetzungen

Bevor Sie Amazon Keyspaces mit verwenden AWS CloudShell, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein AWS Konto mit Zugriff auf Amazon Keyspaces und AWS CloudShell
- Entsprechende IAM-Berechtigungen zur Nutzung beider Dienste
- Grundkenntnisse in Cassandra Query Language (CQL)

Erste Schritte mit AWS CloudShell

Um darauf zuzugreifen AWS CloudShell:

1. Melden Sie sich an bei AWS-Managementkonsole

2. Wählen Sie zum Starten AWS CloudShell das AWS CloudShell Symbol in der Navigationsleiste oben in der Konsole oder navigieren Sie zu https://console.aws.amazon.com/cloudshell/
3. Warten Sie, bis die AWS CloudShell Umgebung initialisiert ist

Einholen von IAM-Berechtigungen für AWS CloudShell

Mithilfe der von bereitgestellten Ressourcen zur Zugriffsverwaltung können Administratoren IAM-Benutzern Berechtigungen erteilen AWS Identity and Access Management, sodass sie auf die Funktionen der Umgebung zugreifen AWS CloudShell und diese nutzen können.

Am schnellsten kann ein Administrator Benutzern Zugriff gewähren, indem er eine AWS verwaltete Richtlinie verwendet. Bei einer von AWS verwalteten Richtlinie handelt es sich um eine eigenständige Richtlinie, die von AWS erstellt und verwaltet wird. Die folgende AWS verwaltete Richtlinie für CloudShell kann an IAM-Identitäten angehängt werden:

- `AWSCloudShellFullAccess`: Erteilt die Erlaubnis zur Nutzung AWS CloudShell mit vollem Zugriff auf alle Funktionen.

Wenn Sie den Umfang der Aktionen einschränken möchten, die ein IAM-Benutzer ausführen kann AWS CloudShell, können Sie eine benutzerdefinierte Richtlinie erstellen, die die `AWSCloudShellFullAccess` verwaltete Richtlinie als Vorlage verwendet. Weitere Informationen zur Einschränkung der Aktionen, die Benutzern zur Verfügung stehen CloudShell, finden Sie im AWS CloudShell Benutzerhandbuch unter Verwaltung von AWS CloudShell Zugriff und Nutzung mit IAM-Richtlinien.

> **Note**
>
> Für Ihre IAM-Identität ist außerdem eine Richtlinie erforderlich, die Ihnen die Erlaubnis erteilt, Amazon Keyspaces aufzurufen.

Sie können eine AWS verwaltete Richtlinie verwenden, um Ihrer IAM-Identität Zugriff auf Amazon Keyspaces zu gewähren, oder Sie können mit der verwalteten Richtlinie als Vorlage beginnen und die Berechtigungen entfernen, die Sie nicht benötigen. Sie können den Zugriff auch auf bestimmte Schlüsselbereiche und Tabellen einschränken, um eine benutzerdefinierte Richtlinie zu erstellen. Die folgende verwaltete Richtlinie für Amazon Keyspaces kann an IAM-Identitäten angehängt werden:

- AmazonKeyspacesFullAccess— Diese Richtlinie gewährt die Erlaubnis zur Nutzung von Amazon Keyspaces mit vollem Zugriff auf alle Funktionen.

Eine ausführliche Erläuterung der in der verwalteten Richtlinie definierten Aktionen finden Sie unterthe section called “AWS verwaltete Richtlinien”.

Weitere Informationen darüber, wie Sie Aktionen einschränken oder den Zugriff auf bestimmte Ressourcen in Amazon Keyspaces einschränken können, finden Sie unterthe section called “So funktioniert Amazon Keyspaces mit IAM”.

Den Cqlsh-Client installieren

AWS CloudShell wird nicht vorinstalliert geliefert`cqlsh`. Sie müssen es installieren, um mit Amazon Keyspaces zu interagieren. Sie können die Keyspaces-specific Amazon-Version von installieren`cqlsh`. `cqlsh-expansion`Installationsanweisungen finden Sie unterthe section called “Verwendung der `cqlsh-expansion`”.

Konfiguration von TLS-Zertifikaten

Um über eine TLS-Verbindung eine Verbindung zu Amazon Keyspaces herzustellen, müssen Sie die erforderlichen digitalen Amazon-Zertifikate herunterladen. Detaillierte Anweisungen zum Herunterladen und Konfigurieren des SSL-Zertifikats finden Sie unterthe section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”.

Verbindung zu Amazon Keyspaces herstellen

Sie können entweder mit IAM-Anmeldeinformationen (empfohlen) oder dienstspezifischen Anmeldeinformationen eine Verbindung zu Amazon Keyspaces herstellen.

Verbindung mit IAM-Anmeldeinformationen herstellen (empfohlen)

AWS CloudShell ist bereits mit Ihren IAM-Anmeldeinformationen authentifiziert. Um mit diesen Anmeldeinformationen eine Verbindung herzustellen, können Sie das SigV4-Authentifizierungs-Plugin verwenden. Ausführliche Anweisungen zur Verwendung von IAM-Anmeldeinformationen mit finden Sie `cqlsh` unter. the section called “Verwendung der `cqlsh-expansion`”

Verbindung mit dienstspezifischen Anmeldeinformationen herstellen

Wenn Sie dienstspezifische Anmeldeinformationen für Amazon Keyspaces erstellt haben, können Sie diese für die Verbindung verwenden. Informationen zum Erstellen und Verwenden

von dienstspezifischen Anmeldeinformationen finden Sie unter. the section called “Erstellen Sie dienstspezifische Anmeldeinformationen”

Verwenden von CQLSH mit Amazon Keyspaces

Wenn eine Verbindung hergestellt ist, können Sie standardmäßige CQL-Befehle verwenden, um mit Ihren Keyspaces und Tabellen zu interagieren. Ausführliche Informationen zu unterstützten CQL-Befehlen und -Vorgängen finden Sie unter. the section called “Verwendung der `cqlsh-expansion`”

Dauerhafte Daten in AWS CloudShell

AWS CloudShell stellt 1 GB persistenten Speicher in Ihrem Home-Verzeichnis bereit. Das bedeutet, dass Skripts, Zertifikate und Konfigurationsdateien, die Sie erstellen, sitzungsübergreifend verfügbar bleiben. Sie können eine Konfigurationsdatei erstellen`cqlsh`, um Verbindungen zu vereinfachen.

Mit einer Konfigurationsdatei können Sie ausführen, `cqlsh` ohne zusätzliche Parameter angeben zu müssen. Weitere Hinweise zum Erstellen und Verwenden von `cqlsh` Konfigurationsdateien finden Sie unterthe section called “Verwendung der `cqlsh-expansion`”.

Best Practices

Bei der Verwendung von Amazon Keyspaces mit AWS CloudShell empfehlen wir die folgenden bewährten Methoden:

- Erstellen Sie Skripts für sich wiederholende Aufgaben und speichern Sie sie in Ihrem persistenten Home-Verzeichnis.
- Verwenden Sie Umgebungsvariablen für vertrauliche Informationen, anstatt sie fest zu codieren.
- Denken Sie daran, dass AWS CloudShell Sitzungen nach Inaktivität zu einem Timeout führen können.
- Beachten Sie die AWS CloudShell Servicequotas und -beschränkungen.
- Erwägen Sie die Verwendung von AWS CLI für automatisierte Operationen mit Amazon Keyspaces.

Fehlerbehebung

Wenn Sie Probleme haben, eine Verbindung zu Amazon Keyspaces herzustellen von AWS CloudShell:

- Stellen Sie sicher, dass Ihre IAM-Berechtigungen die erforderlichen Amazon Keyspaces-Aktionen beinhalten.
- Stellen Sie sicher, dass Sie den richtigen Endpunkt für Ihre Region verwenden.
- Vergewissern Sie sich, dass die SSL-Zertifikate korrekt heruntergeladen und referenziert wurden.
- Stellen Sie sicher, dass Ihre dienstspezifischen Anmeldeinformationen korrekt und nicht abgelaufen sind.
- Wenn Sie eine cqlshrc-Datei verwenden, überprüfen Sie deren Formatierung und Berechtigungen.

Beispiel: So interagieren Sie mit Amazon Keyspaces mithilfe von AWS CloudShell

Nachdem Sie AWS CloudShell von der aus gestartet haben AWS-Managementkonsole, können Sie sofort mit der Interaktion mit Amazon Keyspaces beginnen, indem Sie `cqlsh` oder die Befehlszeilenschnittstelle verwenden. Falls Sie das noch nicht installiert haben`cqlsh-expansion`, finden Sie detaillierte the section called “Verwendung der `cqlsh-expansion`” Anweisungen unter.

> **Note**
>
> Wenn Sie den `cqlsh-expansion` in verwenden AWS CloudShell, müssen Sie keine Anmeldeinformationen konfigurieren, bevor Sie Anrufe tätigen, da Sie bereits in der Shell authentifiziert sind.

Connect zu Amazon Keyspaces her und erstellen Sie einen neuen Keyspace. Lesen Sie dann aus einer Systemtabelle, um zu bestätigen, dass der Schlüsselraum erstellt wurde mit AWS CloudShell

1. Von der aus können Sie starten AWS-Managementkonsole, CloudShell indem Sie die folgenden Optionen auswählen, die in der Navigationsleiste verfügbar sind:
   - Wählen Sie das CloudShell Symbol.
   - Beginnen Sie mit der Eingabe `cloudshell` in das Suchfeld und wählen Sie dann die CloudShell Option.
2. Mit dem folgenden Befehl können Sie eine Verbindung zu Amazon Keyspaces herstellen. Stellen Sie sicher, dass Sie `cassandra.`*`us-east-1`*`.amazonaws.com` durch den richtigen Endpunkt für Ihre Region ersetzen.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

Wenn die Verbindung erfolgreich ist, sollten Sie eine Ausgabe sehen, die dem folgenden Beispiel ähnelt.

```
Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
[cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh current consistency level is ONE.
cqlsh>
```

3. Erstellen Sie einen neuen Schlüsselraum mit dem Namen`mykeyspace`. Sie können dazu den folgenden Befehl verwenden.

```
CREATE KEYSPACE mykeyspace WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

4. Um zu bestätigen, dass der Schlüsselraum erstellt wurde, können Sie mit dem folgenden Befehl aus einer Systemtabelle lesen.

```
SELECT * FROM system_schema_mcs.keyspaces WHERE keyspace_name = 'mykeyspace';
```

Wenn der Aufruf erfolgreich ist, zeigt die Befehlszeile eine Antwort des Services an, die der folgenden Ausgabe ähnelt:

```
 keyspace_name  | durable_writes | replication
----------------+----------------
+-------------------------------------------------------------------------------------
 mykeyspace     |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(1 rows)
```

Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen

Um Benutzern und Anwendungen Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces-Ressourcen zur Verfügung zu stellen, können Sie einen der folgenden Schritte ausführen:

- Erstellen Sie dienstspezifische Anmeldeinformationen, die dem herkömmlichen Benutzernamen und Passwort ähneln, die Cassandra für die Authentifizierung und Zugriffsverwaltung verwendet. AWS dienstspezifische Anmeldeinformationen sind einem bestimmten AWS Identity and Access Management (IAM-) Benutzer zugeordnet und können nur für den Dienst verwendet werden, für den sie erstellt wurden. Weitere Informationen finden Sie unter Using IAM with Amazon Keyspaces (für Apache Cassandra) im IAM-Benutzerhandbuch.

  > **Warning**
  >
  > IAM-Benutzer verfügen über langfristige Anmeldeinformationen, was ein Sicherheitsrisiko darstellt. Um dieses Risiko zu minimieren, empfehlen wir, diesen Benutzern nur die Berechtigungen zu gewähren, die sie für die Ausführung der Aufgabe benötigen, und diese Benutzer zu entfernen, wenn sie nicht mehr benötigt werden.

- Aus Sicherheitsgründen empfehlen wir, IAM-Identitäten zu erstellen, die für alle AWS Dienste verwendet werden, und temporäre Anmeldeinformationen zu verwenden. Das Amazon Keyspaces SigV4-Authentifizierungs-Plugin für Cassandra-Client-Treiber ermöglicht es Ihnen, Anrufe an Amazon Keyspaces mithilfe von IAM-Zugriffsschlüsseln anstelle von Benutzername und Passwort zu authentifizieren. Weitere Informationen darüber, wie das Amazon Keyspaces SigV4-Plugin es IAM-Benutzern, -Rollen und föderierten Identitäten ermöglicht, sich in Amazon Keyspaces-API-Anfragen zu authentifizieren, finden Sie unter AWS Signature Version 4-Prozess (Sigv4).

  Sie können die SigV4-Plugins von den folgenden Orten herunterladen.

  - Java:https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin.
  - Node.js: https://github.com/aws/aws-sigv4-auth-cassandra-nodejs-driver-plugin.
  - Python: https://github.com/aws/aws-sigv4-auth-cassandra-python-driver-plugin.
  - Geh:https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Codebeispiele, die zeigen, wie Verbindungen mithilfe des SigV4-Authentifizierungs-Plug-ins hergestellt werden, finden Sie unterthe section called “Verwenden eines Cassandra-Client-Treibers”.

Themen

- Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen
- AWS Anmeldeinformationen für Amazon Keyspaces erstellen und konfigurieren

Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen

Service-specific Die Anmeldeinformationen ähneln dem herkömmlichen Benutzernamen und Passwort, die Cassandra für die Authentifizierung und Zugriffsverwaltung verwendet. Service-specific Anmeldeinformationen ermöglichen es IAM-Benutzern, auf einen bestimmten AWS Dienst zuzugreifen. Diese langfristigen Anmeldeinformationen können nicht für den Zugriff auf andere AWS Dienste verwendet werden. Sie sind einem bestimmten IAM-Benutzer zugeordnet und können nicht von anderen IAM-Benutzern verwendet werden.

> **Important**
>
> Service-specific Anmeldeinformationen sind langfristige Anmeldeinformationen, die einem bestimmten IAM-Benutzer zugeordnet sind und nur für den Dienst verwendet werden können, für den sie erstellt wurden. Um IAM-Rollen oder föderierten Identitäten Berechtigungen für den Zugriff auf all Ihre AWS Ressourcen mit temporären Anmeldeinformationen zu erteilen, sollten Sie die AWS Authentifizierung mit dem SigV4-Authentifizierungs-Plugin für Amazon Keyspaces verwenden.

Verwenden Sie eines der folgenden Verfahren, um dienstspezifische Anmeldeinformationen zu generieren.

Console

Erstellen Sie dienstspezifische Anmeldeinformationen mithilfe der Konsole

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die AWS Identity and Access Management Konsole unterhttps://console.aws.amazon.com/iam/home.
2. Wählen Sie im Navigationsbereich Benutzer und dann den Benutzer aus, den Sie zuvor erstellt haben und der über Amazon Keyspaces-Berechtigungen verfügt (Richtlinie beigefügt).
3. Wählen Sie Security Credentials (Sicherheitsanmeldeinformationen) aus. Wählen Sie unter Anmeldeinformationen für Amazon Keyspaces die Option Anmeldeinformationen generieren aus, um die dienstspezifischen Anmeldeinformationen zu generieren.

   Ihre servicespezifischen Anmeldeinformationen sind nun verfügbar. Dies ist das einzige Mal, dass Sie das Passwort herunterladen oder anzeigen können. Später kann er nicht mehr wiederhergestellt werden. Sie können Ihr Passwort jedoch jederzeit zurücksetzen. Speichern Sie den Benutzer und das Passwort an einem sicheren Ort, da Sie sie später benötigen.

CLI

Erstellen Sie dienstspezifische Anmeldeinformationen mit dem AWS CLI

Bevor Sie dienstspezifische Anmeldeinformationen generieren, müssen Sie Folgendes herunterladen, installieren und konfigurieren AWS Command Line Interface :AWS CLI

1. Laden Sie das AWS CLI unter http://aws.amazon.com/cliherunter.

   > **Note**
   >
   > Das AWS CLI läuft unter Windows, MacOS oder Linux.

2. Folgen Sie den Anweisungen zur Installation der AWS CLI und zur Konfiguration der AWS CLI im AWS Command Line Interface Benutzerhandbuch.
3. Führen Sie mit dem folgenden Befehl den folgenden Befehl aus AWS CLI, um dienstspezifische Anmeldeinformationen für die Benutzerin zu generieren`alice`, damit sie auf Amazon Keyspaces zugreifen kann.

```
aws iam create-service-specific-credential \
    --user-name alice \
```

```
    --service-name cassandra.amazonaws.com
```

Die Ausgabe sieht wie folgt aus.

```
{
    "ServiceSpecificCredential": {
        "CreateDate": "2019-10-09T16:12:04Z",
        "ServiceName": "cassandra.amazonaws.com",
        "ServiceUserName": "alice-at-111122223333",
        "ServicePassword": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "ServiceSpecificCredentialId": "ACCAYFI33SINPGJEBYESF",
        "UserName": "alice",
        "Status": "Active"
    }
}
```

Notieren Sie sich in der Ausgabe die Werte für `ServiceUserName` und. `ServicePassword` Speichern Sie diese Werte an einem sicheren Ort, da Sie sie später benötigen werden.

> **Important**
>
> Dies ist das einzige Mal, dass `ServicePassword` Ihnen das zur Verfügung steht.

AWS Anmeldeinformationen für Amazon Keyspaces erstellen und konfigurieren

Um programmgesteuert mit dem AWS SDK oder mit Cassandra-Client-Treibern und dem SigV4-Plugin auf Amazon Keyspaces zuzugreifen, benötigen Sie einen IAM-Benutzer mit Zugriffsschlüsseln. AWS CLI Wenn Sie Amazon Keyspaces programmgesteuert verwenden, geben Sie Ihre AWS Zugriffsschlüssel an, damit Sie Ihre Identität bei programmatischen Aufrufen überprüfen AWS können. Ihre Zugangsschlüssel bestehen aus einer Zugriffsschlüssel-ID (zum Beispiel AKIAIOSFODNN7EXAMPLE) und einem geheimen Zugriffsschlüssel (zum Beispiel /b CYEXAMPLEKEY). wJalrXUtnFEMI/K7MDENG PxRfi In diesem Thema werden Sie durch die erforderlichen Schritte in diesem Prozess geführt.

Bewährte Sicherheitsmethoden empfehlen, dass Sie IAM-Benutzer mit eingeschränkten Berechtigungen erstellen und stattdessen IAM-Rollen den Berechtigungen zuordnen, die für die

Ausführung bestimmter Aufgaben erforderlich sind. IAM-Benutzer können dann vorübergehend IAM-Rollen übernehmen, um die erforderlichen Aufgaben auszuführen. Beispielsweise können IAM-Benutzer in Ihrem Konto, die die Amazon Keyspaces-Konsole verwenden, zu einer Rolle wechseln, um vorübergehend die Berechtigungen der Rolle in der Konsole zu nutzen. Die Benutzer verlieren ihre ursprünglichen Berechtigungen und übernehmen die Berechtigungen der zugewiesenen Rolle. Wenn der Benutzer die Rolle verlässt, werden die ursprünglichen Berechtigungen wiederhergestellt. Die Anmeldeinformationen, die die Benutzer verwenden, um die Rolle anzunehmen, sind temporär. Im Gegenteil, IAM-Benutzer verfügen über langfristige Anmeldeinformationen, was ein Sicherheitsrisiko darstellt, wenn ihnen Rechte direkt zugewiesen werden, anstatt Rollen anzunehmen. Um dieses Risiko zu minimieren, empfehlen wir, diesen Benutzern nur die Berechtigungen zu gewähren, die sie für die Ausführung der Aufgabe benötigen, und diese Benutzer zu entfernen, wenn sie nicht mehr benötigt werden. Weitere Informationen zu Rollen finden Sie unter Allgemeine Szenarien für Rollen: Benutzer, Anwendungen und Dienste im IAM-Benutzerhandbuch.

Themen

- Anmeldeinformationen AWS CLI, die für das AWS SDK oder das Amazon Keyspaces SigV4-Plugin für Cassandra-Client-Treiber erforderlich sind
- Erstellen Sie temporäre Anmeldeinformationen, um mithilfe einer IAM-Rolle und des SigV4-Plug-ins eine Verbindung zu Amazon Keyspaces herzustellen
- Erstellen Sie einen IAM-Benutzer für den programmatischen Zugriff auf Amazon Keyspaces in Ihrem Konto AWS
- Erstellen Sie neue Zugriffsschlüssel für einen IAM-Benutzer
- Speichern Sie die Zugriffstasten für den programmatischen Zugriff

Anmeldeinformationen AWS CLI, die für das AWS SDK oder das Amazon Keyspaces SigV4-Plugin für Cassandra-Client-Treiber erforderlich sind

Die folgenden Anmeldeinformationen sind erforderlich, um den IAM-Benutzer oder die IAM-Rolle zu authentifizieren:

`AWS_ACCESS_KEY_ID`

Gibt einen AWS Zugriffsschlüssel an, der einem IAM-Benutzer oder einer IAM-Rolle zugeordnet ist.

Der Zugriffsschlüssel `aws_access_key_id` ist erforderlich, um programmgesteuert eine Verbindung zu Amazon Keyspaces herzustellen.

`AWS_SECRET_ACCESS_KEY`

Gibt den geheimen Schlüssel an, der mit dem Zugriffsschlüssel verknüpft ist. Dies ist im Wesentlichen das "Passwort" für den Zugriffsschlüssel.

Das `aws_secret_access_key` ist erforderlich, um programmgesteuert eine Verbindung zu Amazon Keyspaces herzustellen.

`AWS_SESSION_TOKEN`— Fakultativ

Gibt den Sitzungstokenwert an, der erforderlich ist, wenn Sie temporäre Sicherheitsanmeldeinformationen verwenden, die Sie direkt aus AWS -Security-Token-Service -Operationen abgerufen haben. Weitere Informationen finden Sie unter the section called “Temporäre Anmeldeinformationen für die Verbindung mit Amazon Keyspaces erstellen”.

Wenn Sie eine Verbindung mit einem IAM-Benutzer herstellen, `aws_session_token` ist das nicht erforderlich.

Erstellen Sie temporäre Anmeldeinformationen, um mithilfe einer IAM-Rolle und des SigV4-Plug-ins eine Verbindung zu Amazon Keyspaces herzustellen

Die empfohlene Methode für den programmgesteuerten Zugriff auf Amazon Keyspaces ist die Verwendung temporärer Anmeldeinformationen zur Authentifizierung mit dem SigV4-Plugin. In vielen Fällen benötigen Sie keine langfristigen Zugriffsschlüssel, die nie ablaufen (wie es bei IAM-Benutzern der Fall ist). Stattdessen können Sie eine IAM-Rolle erstellen und temporäre Sicherheitsanmeldedaten generieren. Temporäre Sicherheitsanmeldeinformationen bestehen aus einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel, enthalten aber auch ein Sicherheits-Token, das angibt, wann die Anmeldeinformationen ablaufen. Weitere Informationen zur Verwendung von IAM-Rollen anstelle von langfristigen Zugriffsschlüsseln finden Sie unter Zu einer IAM-Rolle (AWS API) wechseln.

Um mit temporären Anmeldeinformationen zu beginnen, müssen Sie zunächst eine IAM-Rolle erstellen.

Erstellen Sie eine IAM-Rolle, die nur Lesezugriff auf Amazon Keyspaces gewährt

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im Navigationsbereich Rollen und anschließend Rolle erstellen aus.

3. Wählen Sie auf der Seite Rolle erstellen unter Typ der vertrauenswürdigen Entität auswählen die Option AWS Dienst aus. Wählen Sie unter Anwendungsfall auswählen die Option Amazon EC2 und dann Weiter aus.
4. Wählen Sie auf der Seite Berechtigungen hinzufügen unter Berechtigungsrichtlinien die Option Amazon Keyspaces Read Only Access aus der Richtlinienliste aus und klicken Sie dann auf Weiter.
5. Geben Sie auf der Seite Name, Überprüfung und Erstellung einen Namen für die Rolle ein und überprüfen Sie die Abschnitte Vertrauenswürdige Entitäten auswählen und Berechtigungen hinzufügen. Sie können auf dieser Seite auch optionale Tags für die Rolle hinzufügen. Wenn Sie fertig sind, wählen Sie Rolle erstellen aus. Merken Sie sich diesen Namen, da Sie ihn benötigen, wenn Sie Ihre Amazon EC2 EC2-Instance starten.

Um temporäre Sicherheitsanmeldedaten im Code zu verwenden, rufen Sie programmgesteuert eine AWS -Security-Token-Service API auf `AssumeRole` und extrahieren die resultierenden Anmeldeinformationen und das Sitzungstoken aus Ihrer IAM-Rolle, die Sie im vorherigen Schritt erstellt haben. Anschließend verwenden Sie diese Werte als Anmeldeinformationen für nachfolgende Aufrufe von. AWS Das folgende Beispiel zeigt Pseudocode für die Verwendung temporärer Sicherheitsanmeldeinformationen:

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
cassandraRequest = CreateAmazoncassandraClient(tempCredentials);
```

Ein Beispiel, das temporäre Anmeldeinformationen mithilfe des Python-Treibers für den Zugriff auf Amazon Keyspaces implementiert, finden Sie unter???.

Einzelheiten über den Aufruf von `AssumeRole`, `GetFederationToken` und anderen API-Vorgängen finden Sie in der AWS -Security-Token-Service -API-Referenz. Informationen dazu, wie die temporären Sicherheitsanmeldeinformationen und das Sitzungstoken aus dem Ergebnis abgerufen werden, finden Sie in der Dokumentation für das SDK, mit dem Sie arbeiten. Sie finden die Dokumentation für alle AWS SDKs auf der AWS Hauptdokumentationsseite im Abschnitt SDKs und Toolkits.

Erstellen Sie einen IAM-Benutzer für den programmatischen Zugriff auf Amazon Keyspaces in Ihrem Konto AWS

Um Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces mit dem AWS CLI, dem AWS SDK oder dem SigV4-Plugin zu erhalten, müssen Sie zunächst einen IAM-Benutzer oder eine IAM-Rolle erstellen. Der Prozess der Erstellung eines IAM-Benutzers und der Konfiguration dieses IAM-Benutzers für den programmatischen Zugriff auf Amazon Keyspaces wird in den folgenden Schritten gezeigt:

1. Erstellen Sie den Benutzer in den AWS-Managementkonsole Tools für Windows oder mithilfe einer PowerShell API-Operation. AWS CLI AWS Wenn Sie den Benutzer in der erstellen AWS-Managementkonsole, werden die Anmeldeinformationen automatisch erstellt.
2. Wenn Sie den Benutzer programmgesteuert erstellen, müssen Sie in einem zusätzlichen Schritt einen Zugriffsschlüssel (Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel) für diesen Benutzer erstellen.
3. Erteilen Sie dem Benutzer Berechtigungen für den Zugriff auf Amazon Keyspaces.

Informationen zu den Berechtigungen, die Sie benötigen, um einen IAM-Benutzer zu erstellen, finden Sie unter Erforderliche Berechtigungen für den Zugriff auf IAM-Ressourcen.

Console

Erstellen Sie einen IAM-Benutzer mit programmatischem Zugriff (Konsole)

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im Navigationsbereich Users und dann Add User.
3. Geben Sie den Benutzernamen für den neuen Benutzer an. Dies ist der Anmeldename für. AWS

> **Note**
>
> Benutzernamen können eine Kombination aus bis zu 64 Buchstaben, Ziffern und den folgenden Zeichen enthalten: Plus (+), Gleichheitszeichen (=), Komma (,), Punkt (.), at-Zeichen (@), Unterstrich (_) und Bindestrich (-). Namen müssen innerhalb eines Kontos eindeutig sein. Sie werden nicht nach Groß- und Kleinschreibung

unterschieden. So können Sie beispielsweise keine zwei Gruppen mit Namen TESTBENUTZER und testbenutzer erstellen.

4. Wählen Sie Zugriffsschlüssel — Programmgesteuerter Zugriff, um einen Zugriffsschlüssel für den neuen Benutzer zu erstellen. Sie können den Zugriffsschlüssel ansehen oder herunterladen, wenn Sie zur letzten Seite gelangen.

   Wählen Sie Weiter: Berechtigungen aus.

5. Wählen Sie auf der Seite Berechtigungen festlegen die Option Bestehende Richtlinien direkt anhängen aus, um dem neuen Benutzer Berechtigungen zuzuweisen.

   Diese Option zeigt die Liste der AWS verwalteten und vom Kunden verwalteten Richtlinien an, die in Ihrem Konto verfügbar sind. Sie können `keyspaces` in das Suchfeld eingeben, um nur die Richtlinien anzuzeigen, die sich auf Amazon Keyspaces beziehen.

   Für Amazon Keyspaces sind die verfügbaren verwalteten Richtlinien `AmazonKeyspacesFullAccess` und`AmazonKeyspacesReadOnlyAccess`. Weitere Informationen zu den einzelnen Richtlinien finden Sie unterthe section called “AWS verwaltete Richtlinien”.

   Wählen Sie zu Testzwecken und um den Verbindungs-Tutorials zu folgen, die `AmazonKeyspacesReadOnlyAccess` Richtlinie für den neuen IAM-Benutzer aus. Hinweis: Als bewährte Methode empfehlen wir, das Prinzip der geringsten Rechte zu befolgen und benutzerdefinierte Richtlinien zu erstellen, die den Zugriff auf bestimmte Ressourcen einschränken und nur die erforderlichen Aktionen zulassen. Weitere Informationen zu IAM-Richtlinien und Beispielrichtlinien für Amazon Keyspaces finden Sie unter. the section called “Identitätsbasierte Richtlinien von Amazon Keyspaces” Nachdem Sie benutzerdefinierte Berechtigungsrichtlinien erstellt haben, fügen Sie Ihre Richtlinien den Rollen hinzu und lassen Sie die Benutzer dann vorübergehend die entsprechenden Rollen übernehmen.

   Wählen Sie Weiter: Tags aus.

6. Auf der Seite Stichwörter hinzufügen (optional) können Sie Stichwörter für den Benutzer hinzufügen oder Weiter: Überprüfen wählen.

7. Auf der Seite „Überprüfen“ können Sie alle Entscheidungen sehen, die Sie bis zu diesem Zeitpunkt getroffen haben. Wenn Sie bereit sind, fortzufahren, wählen Sie Benutzer erstellen aus.

8. Um die Zugriffsschlüssel des Benutzers (Zugriffsschlüssel-IDs und geheime Zugriffsschlüssel) anzuzeigen, wählen Sie neben dem Passwort und dem Zugriffsschlüssel die Option Show (Anzeigen) aus. Zum Speichern der Zugriffsschlüssel wählen Sie Download .csv. Speichern Sie die Datei an einem sicheren Speicherort.

> **Important**
>
> Dies ist Ihre einzige Möglichkeit, die geheimen Zugriffsschlüssel anzusehen oder herunterzuladen, und Sie benötigen diese Informationen, bevor sie das SigV4-Plugin verwenden können. Speichern Sie die neue Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel des Benutzers an einem sicheren Speicherort. Sie haben nach diesem Schritt keinen Zugriff mehr auf die geheimen Zugriffsschlüssel.

CLI

Erstellen Sie einen IAM-Benutzer mit programmatischem Zugriff ()AWS CLI

1. Erstellen Sie einen Benutzer mit dem folgenden AWS CLI Code.

   - aws iam create-user

2. Erteilen Sie dem Benutzer programmatischen Zugriff. Dazu sind Zugriffsschlüssel erforderlich, die auf folgende Weise generiert werden können.

   - AWS CLI: aws iam create-access-key
   - Tools für Windows PowerShell: New-IAMAccessKey
   - IAM-API: CreateAccessKey

> **Important**
>
> Dies ist Ihre einzige Möglichkeit, die geheimen Zugriffsschlüssel anzusehen oder herunterzuladen, und Sie benötigen diese Informationen, bevor sie das SigV4-Plugin verwenden können. Speichern Sie die neue Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel des Benutzers an einem sicheren Speicherort. Sie haben nach diesem Schritt keinen Zugriff mehr auf die geheimen Zugriffsschlüssel.

3. Hängen Sie die `AmazonKeyspacesReadOnlyAccess` Richtlinie, die die Benutzerberechtigungen definiert, an den Benutzer an. Hinweis: Als bewährte Methode

empfehlen wir, Benutzerberechtigungen zu verwalten, indem Sie den Benutzer zu einer Gruppe hinzufügen und der Gruppe eine Richtlinie anhängen, anstatt sie direkt an einen Benutzer anzuhängen.

- AWS CLI: aws iam attach-user-policy

Erstellen Sie neue Zugriffsschlüssel für einen IAM-Benutzer

Wenn Sie bereits einen IAM-Benutzer haben, können Sie jederzeit neue Zugriffsschlüssel erstellen. Weitere Informationen zur Schlüsselverwaltung, z. B. zur Aktualisierung von Zugriffsschlüsseln, finden Sie unter Zugriffsschlüssel für IAM-Benutzer verwalten.

So erstellen Sie Zugriffsschlüssel für einen IAM-Benutzer (Konsole)

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Klicken Sie im Navigationsbereich auf Users (Benutzer).
3. Wählen Sie den Namen des Benutzers, dessen Zugriffsschlüssel Sie erstellen möchten.
4. Wählen Sie auf der Übersichtsseite des Benutzers die Registerkarte Sicherheitsanmeldeinformationen aus.
5. Wählen Sie im Abschnitt Zugriffstasten unter Bewährte Methoden und Alternativen für Zugriffsschlüssel den Anwendungsfall Andere aus. Klicken Sie auf Weiter, geben Sie nach Bedarf optionale Informationen ein und wählen Sie Zugriffsschlüssel erstellen aus.

   Wählen Sie zum Anzeigen des neuen Zugriffsschlüsselpaars Show (Anzeigen) aus. Ihre Anmeldeinformationen sehen etwa folgendermaßen aus:

   - Zugriffsschlüssel-ID: AKIAIOSFODNN7EXAMPLE
   - Geheimer Zugriffsschlüssel: wJalrXUtnFEMI/K7MDENG /b PxRfi CYEXAMPLEKEY

   > **Note**
   >
   > Sie haben keinen Zugriff auf den geheimen Zugriffsschlüssel mehr, nachdem das Dialogfeld geschlossen wird.

   Beachten Sie die folgenden bewährten Methoden für das von Ihnen erstellte key pair.

- Speichern Sie Ihren Zugriffsschlüssel niemals im Klartext, in einem Code-Repository oder in Code.
- Deaktivieren oder löschen Sie Zugriffsschlüssel, wenn sie nicht mehr benötigt werden.
- Aktivieren Sie Berechtigungen mit den geringsten Rechten.
- Wechseln Sie die Zugriffsschlüssel regelmäßig.

6. Wählen Sie zum Herunterladen des Schlüsselpaares Download .csv file aus. Speichern Sie die Schlüssel an einem sicheren Ort.
7. Nachdem Sie die CSV-Datei heruntergeladen haben, klicken Sie auf Close (Schließen).

Wenn Sie einen Zugriffsschlüssel erstellen, ist das Schlüsselpaar standardmäßig aktiv, und Sie können es sofort verwenden.

Speichern Sie die Zugriffstasten für den programmatischen Zugriff

Als bewährte Methode empfehlen wir, Zugriffsschlüssel nicht direkt in Code einzubetten. Mit den AWS SDKs und den AWS Befehlszeilentools können Sie Zugriffsschlüssel an bekannten Orten platzieren, sodass Sie sie nicht im Code speichern müssen. Legen Sie Zugriffsschlüssel an einem der folgenden Orte ab:

- Umgebungsvariablen — Wählen Sie auf einem Mehrmandantensystem Benutzerumgebungsvariablen und nicht Systemumgebungsvariablen.
- CLI-Anmeldeinformationsdatei – Die `credentials`- und `config`-Dateien werden aktualisiert, wenn Sie den Befehl `aws configure` ausführen. Die `credentials` Datei befindet sich `~/.aws/credentials` unter Linux, macOS oder Unix oder unter Windows `C:\Users\`*`USERNAME`*`\.aws\credentials` unter. Diese Datei kann die Anmeldeinformationsdetails für das `default`-Profil und alle benannten Profile enthalten.
- CLI-Konfigurationsdatei – Die `credentials`- und `config`-Dateien werden aktualisiert, wenn Sie den Befehl `aws configure` ausführen. Die `config` Datei befindet sich `~/.aws/config` unter Linux, macOS oder Unix oder unter Windows `C:\Users\`*`USERNAME`*`\.aws\config` unter. Diese Datei enthält die Konfigurationseinstellungen für das Standardprofil sowie alle benannten Profile.

Das Speichern von Zugriffsschlüsseln als Umgebungsvariablen ist eine Voraussetzung für diethe section called “Authentifizierungs-Plugin für Java 4.x”. Beachten Sie, dass dies die Standardeinstellung beinhaltet AWS-Region. Der Client sucht mithilfe der standardmäßigen Anbieterkette für Anmeldeinformationen nach Anmeldeinformationen, und Zugriffsschlüssel, die

als Umgebungsvariablen gespeichert sind, haben Vorrang vor allen anderen Speicherorten, z. B. Konfigurationsdateien. Weitere Informationen finden Sie unter Konfigurationseinstellungen und Rangfolge.

Die folgenden Beispiele zeigen, wie Sie Umgebungsvariablen für den Standardbenutzer konfigurieren können.

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of security token>
$ export AWS_DEFAULT_REGION=us-east-1
```

Durch die Festlegung der Umgebungsvariablen wird der verwendete Wert bis zum Ende der Shell-Sitzung oder bis zur Festlegung eines anderen Wertes für die Variable geändert. Sie können Variablen für zukünftige Sitzungen persistent machen, indem Sie sie im Startup-Skript Ihrer Shell festlegen.

Windows Command Prompt

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> setx AWS_SESSION_TOKEN AQoDYXdzEJr...<remainder of security token>
C:\> setx AWS_DEFAULT_REGION us-east-1
```

Bei Verwendung von `set` zur Festlegung einer Umgebungsvariablen wird der verwendete Wert bis zum Ende der aktuellen Eingabeaufforderungssitzung oder bis zur Festlegung eines anderen Wertes für die Variable geändert. Bei Verwendung von `setx` zur Festlegung einer Umgebungsvariablen wird der verwendete Wert in der aktuellen Eingabeaufforderungssitzung und allen nach Ausführung des Befehls erstellten Eingabeaufforderungssitzungen geändert. Andere Befehls-Shells, die zum Zeitpunkt der Befehlsausführung bereits ausgeführt werden, sind hiervon nicht betroffen.

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
PS C:\> $Env:AWS_SESSION_TOKEN="AQoDYXdzEJr...<remainder of security token>"
PS C:\> $Env:AWS_DEFAULT_REGION="us-east-1"
```

Wenn Sie an der PowerShell Eingabeaufforderung eine Umgebungsvariable festlegen, wie in den vorherigen Beispielen gezeigt, wird der Wert nur für die Dauer der aktuellen Sitzung gespeichert. Um die Einstellung der Umgebungsvariablen für alle Sitzungen PowerShell und Befehlszeilensitzungen beizubehalten, speichern Sie sie mithilfe der Systemanwendung in der Systemsteuerung. Alternativ können Sie die Variable für alle future PowerShell Sitzungen festlegen, indem Sie sie zu Ihrem PowerShell Profil hinzufügen. Weitere Informationen zum Speichern von Umgebungsvariablen oder deren Beibehaltung über mehrere Sitzungen hinweg finden Sie in der PowerShell Dokumentation.

Service-Endpunkte für Amazon Keyspaces

Themen

- Ports und Protokolle
- Globale Endpunkte
- AWS GovCloud (US) Region FIPS-Endpunkte
- Endpunkte der China Regionen
- Streaming-Endpunkte
- Verbindung zu Dual-Stack-Endpunkten herstellen

Ports und Protokolle

Sie können programmgesteuert auf Amazon Keyspaces zugreifen, indem Sie einen `cqlsh` Client ausführen, einen für Apache 2.0 lizenzierten Cassandra-Treiber verwenden oder das SDK und das AWS CLI SDK verwenden. AWS

Die folgende Tabelle zeigt die Ports und Protokolle für die verschiedenen Zugriffsmechanismen.

Programmatischer Zugriff	Port	Protocol (Protokoll)
CQLSH	9142	TLS
Cassandra-Treiber	9142	TLS
AWS CLI	443	HTTPS
AWS SDK	443	HTTPS

Für TLS-Verbindungen verwendet Amazon Keyspaces Zertifikate, die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden, um sich gegenüber dem Server zu authentifizieren. Weitere Informationen finden Sie unter the section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell” oder im Abschnitt Bevor Sie beginnen zu Ihrem Treiber im Kapitel. the section called “Verwenden eines Cassandra-Client-Treibers”

Globale Endpunkte

Amazon Keyspaces unterstützt sowohl öffentliche IPv4- als auch IPv6-Endpunkte. Sie können zwischen IPv4-Endpunkten und Dual-Stack-Endpunkten wählen. Die Endpunkte verwenden die folgende Namenskonvention, bei der Sie sie durch eine andere aus der Tabelle *`us-east-1`* ersetzen können. AWS-Region

- IPv4-Endpunkte — `cassandra.`*`us-east-1`*`.amazonaws.com`
- Dual-stack Endpunkte — `cassandra.`*`us-east-1`*`.api.aws`

Weitere Informationen zu Dual-Stack-Endpunkten und zur Konfiguration von Verbindungen finden Sie unter. the section called “Verbindung zu Dual-Stack-Endpunkten herstellen”

Amazon Keyspaces ist in den folgenden Regionen verfügbar.

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
USA Ost (Ohio)	us-east-2	cassandra.us-east-2.amazonaws.com cassandra.us-east-2.api.aws	HTTPS und TLS HTTPS und TLS	
USA Ost (Nord-Virginia)	us-east-1	cassandra.us-east-1.amazonaws.com cassandra-fips.us-east-1.api.aws cassandra-fips.us-east-1.amazonaws.com cassandra.us-east-1.api.aws	HTTPS und TLS HTTPS und TLS HTTPS und TLS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)
			HTTPS und TLS
USA West (Nordkalifornien)	us-west-1	cassandra.us-west-1.amazonaws.com cassandra.us-west-1.api.aws	HTTPS und TLS HTTPS und TLS
USA West (Oregon)	us-west-2	cassandra.us-west-2.amazonaws.com cassandra-fips.us-west-2.api.aws cassandra-fips.us-west-2.amazonaws.com cassandra.us-west-2.api.aws	HTTPS und TLS HTTPS und TLS HTTPS und TLS HTTPS und TLS
Afrika (Kapstadt)	af-south-1	cassandra.af-south-1.amazonaws.com cassandra.af-south-1.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Hongkong)	ap-east-1	cassandra.ap-east-1.amazonaws.com cassandra.ap-east-1.api.aws	HTTPS und TLS HTTPS und TLS

Name der Region	Region	Endpunkt	Protocol (Protokoll)
Asien-Pazifik (Hyderabad)	ap-south-2	cassandra.ap-south-2.api.aws	HTTPS und TLS
Asien-Pazifik (Jakarta)	ap-southeast-3	cassandra.ap-southeast-3.api.aws	HTTPS und TLS
Asien-Pazifik (Malaysia)	ap-southeast-5	cassandra.ap-southeast-5.amazonaws.com cassandra.ap-southeast-5.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Melbourne)	ap-southeast-4	cassandra.ap-southeast-4.api.aws	HTTPS und TLS
Asien-Pazifik (Mumbai)	ap-south-1	cassandra.ap-south-1.amazonaws.com cassandra.ap-south-1.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Neuseeland)	ap-southeast-6	cassandra.ap-southeast-6.api.aws	HTTPS und TLS
Asien-Pazifik (Osaka)	ap-northeast-3	cassandra.ap-northeast-3.api.aws	HTTPS und TLS

Name der Region	Region	Endpunkt	Protocol (Protokoll)
Asien-Pazifik (Seoul)	ap-northeast-2	cassandra.ap-northeast-2.amazonaws.com cassandra.ap-northeast-2.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Singapur)	ap-southeast-1	cassandra.ap-southeast-1.amazonaws.com cassandra.ap-southeast-1.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Sydney)	ap-southeast-2	cassandra.ap-southeast-2.amazonaws.com cassandra.ap-southeast-2.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Taipeh)	ap-east-2	cassandra.ap-east-2.api.aws	HTTPS und TLS
Asien-Pazifik (Thailand)	ap-southeast-7	cassandra.ap-southeast-7.amazonaws.com cassandra.ap-southeast-7.api.aws	HTTPS und TLS HTTPS und TLS
Asien-Pazifik (Tokio)	ap-northeast-1	cassandra.ap-northeast-1.amazonaws.com cassandra.ap-northeast-1.api.aws	HTTPS und TLS HTTPS und TLS

Name der Region	Region	Endpunkt	Protocol (Protokoll)
Kanada (Zentral)	ca-central-1	cassandra.ca-central-1.amazonaws.com cassandra.ca-central-1.api.aws	HTTPS und TLS HTTPS und TLS
Kanada West (Calgary)	ca-west-1	cassandra.ca-west-1.amazonaws.com cassandra.ca-west-1.api.aws	HTTPS und TLS HTTPS und TLS
Europa (Frankfurt)	eu-central-1	cassandra.eu-central-1.amazonaws.com cassandra.eu-central-1.api.aws	HTTPS und TLS HTTPS und TLS
Europa (Irland)	eu-west-1	cassandra.eu-west-1.amazonaws.com cassandra.eu-west-1.api.aws	HTTPS und TLS HTTPS und TLS
Europa (London)	eu-west-2	cassandra.eu-west-2.amazonaws.com cassandra.eu-west-2.api.aws	HTTPS und TLS HTTPS und TLS
Europa (Mailand)	eu-south-1	cassandra.eu-south-1.api.aws	HTTPS und TLS

Name der Region	Region	Endpunkt	Protocol (Protokoll)
Europa (Paris)	eu-west-3	cassandra.eu-west-3.amazonaws.com cassandra.eu-west-3.api.aws	HTTPS und TLS HTTPS und TLS
Europa (Spanien)	eu-south-2	cassandra.eu-south-2.api.aws	HTTPS und TLS
Europa (Stockholm)	eu-north-1	cassandra.eu-north-1.amazonaws.com cassandra.eu-north-1.api.aws	HTTPS und TLS HTTPS und TLS
Europa (Zürich)	eu-central-2	cassandra.eu-central-2.api.aws	HTTPS und TLS
Israel (Tel Aviv)	il-central-1	cassandra.il-central-1.api.aws	HTTPS und TLS
Mexiko (Zentral)	mx-central-1	cassandra.mx-central-1.api.aws	HTTPS und TLS
Naher Osten (Bahrain)	me-south-1	cassandra.me-south-1.amazonaws.com cassandra.me-south-1.api.aws	HTTPS und TLS HTTPS und TLS
Naher Osten (VAE)	me-central-1	cassandra.me-central-1.amazonaws.com cassandra.me-central-1.api.aws	HTTPS und TLS HTTPS und TLS

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
Südamerika (São Paulo)	sa-east-1	cassandra.sa-east-1.amazonaws.com cassandra.sa-east-1.api.aws	HTTPS und TLS HTTPS und TLS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com cassandra.us-gov-east-1.api.aws	HTTPS und TLS HTTPS und TLS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com cassandra.us-gov-west-1.api.aws	HTTPS und TLS HTTPS und TLS	

AWS GovCloud (US) Region FIPS-Endpunkte

Verfügbare FIPS-Endpunkte in der. AWS GovCloud (US) Region Amazon Keyspaces unterstützt sowohl IPv4- als auch IPv6-FIPS-Endpunkte. Sie können zwischen IPv4-Endpunkten und Dual-Stack-Endpunkten wählen. Weitere Informationen finden Sie unter Amazon Keyspaces im AWS GovCloud (US) Benutzerhandbuch.

Name der Region	Region	FIPS-Endpunkt IPv4	Dual-Stack am FIPS-Endpunkt	Protocol (Protokoll)
AWS GovClo (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com	cassandra.us-gov-east-1.api.aws	HTTPS und TLS

Name der Region	Region	FIPS-Endpunkt IPv4	Dual-Stack am FIPS-Endpunkt	Protocol (Protokoll)
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com	cassandra.us-gov-west-1.api.aws	HTTPS und TLS

Endpunkte der China Regionen

Amazon Keyspaces unterstützt IPv4-Endpunkte in den AWS Regionen China.

Um auf diese Endpunkte zugreifen zu können, müssen Sie sich für einen separaten Satz von Kontoanmeldeinformationen anmelden, die nur für die Regionen China gelten. Weitere Informationen finden Sie unter Registrierung, Konten und Anmeldeinformationen in China.

Name der Region	Region	Endpunkt	Protocol (Protokoll)
China (Beijing)	cn-north-1	cassandra.cn-north-1.amazonaws.com.cn	HTTPS und TLS
China (Ningxia)	cn-northwest-1	cassandra.cn-northwest-1.amazonaws.com.cn	HTTPS und TLS

Streaming-Endpunkte

Amazon Keyspaces CDC streams ist im Folgenden AWS-Regionen verfügbar. Diese Tabelle zeigt den verfügbaren Dual-Stack-Serviceendpunkt für jede Region. Weitere Informationen zu finden Sie Amazon Keyspaces CDC streams unterthe section called “Greifen Sie auf CDC-Stream-Endpunkte zu”.

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
USA Ost (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS	
USA Ost (Nord-Virginia)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
USA West (Nordkalifornien)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	
USA West (Oregon)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
Afrika (Kapstadt)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	
Asien-Pazifik (Hongkong)	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	
Asien-Pazifik (Hyderabad)	ap-south-2	cassandra-streams.ap-south-2.api.aws	HTTPS	
Asien-Pazifik (Jakarta)	ap-southeast-3	cassandra-streams.ap-southeast-3.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
Asien-Pazifik (Malaysia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Asien-Pazifik (Melbourne)	ap-southeast-4	cassandra-streams.ap-southeast-4.api.aws	HTTPS	
Asien-Pazifik (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	
Asien-Pazifik (Neuseeland)	ap-southeast-6	cassandra-streams.ap-southeast-6.api.aws	HTTPS	
Asien-Pazifik (Osaka)	ap-northeast-3	cassandra-streams.ap-northeast-3.api.aws	HTTPS	
Asien-Pazifik (Seoul)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	
Asien-Pazifik (Singapur)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)
Asien-Pazifik (Sydney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS
Asien-Pazifik (Taipeh)	ap-east-2	cassandra-streams.ap-east-2.api.aws	HTTPS
Asien-Pazifik (Thailand)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS
Asien-Pazifik (Tokio)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS
Kanada (Zentral)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS
Kanada West (Calgary)	ca-west-1	cassandra-streams.ca-west-1.api.aws	HTTPS
Europa (Frankfurt)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS
Europa (Irland)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS
Europa (London)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
Europa (Mailand)	eu-south-1	cassandra-streams.eu-south-1.api.aws	HTTPS	
Europa (Paris)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	
Europa (Spanien)	eu-south-2	cassandra-streams.eu-south-2.api.aws	HTTPS	
Europa (Stockholm)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Europa (Zürich)	eu-central-2	cassandra-streams.eu-central-2.api.aws	HTTPS	
Israel (Tel Aviv)	il-central-1	cassandra-streams.il-central-1.api.aws	HTTPS	
Mexiko (Zentral)	mx-central-1	cassandra-streams.mx-central-1.api.aws	HTTPS	
Naher Osten (Bahrain)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	
Naher Osten (VAE)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
Südamerika (São Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
AWS GovCloud (US-East)	us-gov-east-1	cassandra-streams.us-gov-east-1.api.aws	HTTPS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra-streams.us-gov-west-1.api.aws	HTTPS	

Verbindung zu Dual-Stack-Endpunkten herstellen

Die globalen Endpunkte von Amazon Keyspaces sind Dual-Stack-Endpunkte, die IPv4- und IPv6-Anfragen akzeptieren.

Wenn Sie über IPv6 eine Verbindung zu Amazon Keyspaces herstellen, passt der Service die Antworten der Systemtabelle automatisch an Ihr Verbindungsprotokoll an. Dadurch wird sichergestellt, dass Ihre Anwendungen konsistente Netzwerkadressinformationen erhalten, die ihrem Verbindungstyp entsprechen. Dadurch erhält der Client genaue Informationen zur Netzwerktopologie und gewährleistet gleichzeitig die Abwärtskompatibilität vorhandener CQL-Anwendungen.

Amazon Keyspaces erkennt das von Ihrer Client-Verbindung verwendete Netzwerkprotokoll (IPv4 oder IPv6) automatisch und passt die Systemtabellenantworten entsprechend an. Diese Erkennung erfolgt transparent während des ersten Verbindungs-Handshakes, sodass keine zusätzliche Konfiguration durch die Client-Anwendung erforderlich ist.

Amazon Keyspaces gibt IP-Adressen auf der Grundlage Ihres Verbindungsprotokolls zurück. Beispielsweise gibt eine Anfrage von einem IPv4-Netzwerk die folgende Antwort zurück.

```
SELECT * FROM system.peers;
-- Returns IPv4 addresses in peer column
-- Example: 172.31.1.1, 172.31.1.2, etc.
```

Eine Verbindung von einem IPv6-Netzwerk zu einem Dual-Stack-Endpunkt gibt `cassandra.us-east-1.api.aws` beispielsweise die folgende Antwort zurück.

```
SELECT * FROM system.peers;
-- Returns IPv6 addresses in peer column
-- Example: 2001:db8::1, 2001:db8::2, etc.
```

Weitere Informationen zur IPv6-Unterstützung in Amazon Keyspaces finden Sie unter. the section called “IPv6-Unterstützung in Amazon Keyspaces”

IPv6-Unterstützung in Amazon Keyspaces

Die IPv6-Unterstützung in Amazon Keyspaces ermöglicht es Anwendungen, Verbindungen mithilfe von Internet Protocol Version 6 herzustellen, dem Internetprotokoll der nächsten Generation, das im Vergleich zu IPv4 einen erheblich erweiterten Adressraum bietet. Die Implementierung verwendet Dual-Stack-Endpunkte, die sowohl IPv4 als auch IPv6 gleichzeitig unterstützen, wodurch Abwärtskompatibilität gewährleistet und gleichzeitig eine zukunftsfähige Konnektivität ermöglicht wird. Eine Liste der Endpunkte finden Sie unter the section called “Globale Endpunkte”.

Amazon Keyspaces implementiert IPv6-Unterstützung über eine Dual-Stack-Architektur, die vollständige Abwärtskompatibilität gewährleistet und gleichzeitig IPv6-Konnektivität ermöglicht.

DNS-Auflösung in Amazon Keyspaces

Wenn Anwendungen eine Verbindung zu Dual-Stack-Endpunkten herstellen, gibt der DNS-Auflösungsprozess beide Adresstypen zurück:

A-Datensätze (IPv4)

: Herkömmliche IPv4-Adressen für Abwärtskompatibilität

AAAA-Einträge (IPv6)

: Neue IPv6-Adressen für moderne Konnektivität

Das Betriebssystem und der Netzwerk-Stack des Clients wählen automatisch das am besten geeignete Protokoll auf der Grundlage der lokalen Konfiguration, der Netzwerkverfügbarkeit und der Systemeinstellungen aus.

Das Cassandra Query Language (CQL) -Protokoll unterstützt nahtlos IPv6-Konnektivität, ohne dass Änderungen am Anwendungscode erforderlich sind.

Automatische Protokollauswahl

- Anwendungen spezifizieren den Dual-Stack-Endpunkt
- Der Netzwerk-Stack wählt je nach Verfügbarkeit IPv4 oder IPv6
- Für bestehende CQL-Anwendungen sind keine Codeänderungen erforderlich

Treiberkompatibilität

- Alle wichtigen CQL-Treiber unterstützen IPv6 transparent
- DataStax Treiber verarbeiten IPv6-Adressen nativ
- Open-source Treiber funktionieren ohne Änderung

Konsistenz der Verbindung

- Die Systemtabellen geben das verwendete Verbindungsprotokoll wieder
- IPv6-Verbindungen zeigen IPv6-Adressen in `system.peers`
- IPv4-Verbindungen zeigen weiterhin IPv4-Adressen

Verwenden`cqlsh`, um eine Verbindung zu Amazon Keyspaces herzustellen

Um eine Verbindung zu Amazon Keyspaces herzustellen`cqlsh`, können Sie den `cqlsh-expansion` verwenden. Dies ist ein Toolkit, das gängige Apache Cassandra-Tools wie `cqlsh` und Hilfsprogramme enthält, die für Amazon Keyspaces vorkonfiguriert sind und gleichzeitig die volle Kompatibilität mit Apache Cassandra gewährleisten. Es `cqlsh-expansion` integriert das SigV4-Authentifizierungs-Plugin und ermöglicht es Ihnen, eine Verbindung mithilfe von IAM-Zugriffsschlüsseln anstelle von Benutzername und Passwort herzustellen. Sie müssen nur die `cqlsh` Skripts installieren, um eine Verbindung herzustellen, und nicht die vollständige Apache Cassandra-Distribution, da Amazon Keyspaces serverlos ist. Dieses einfache Installationspaket enthält die `cqlsh-expansion` und die klassischen `cqlsh` Skripte, die Sie auf jeder Plattform installieren können, die Python unterstützt.

> **Note**
>
> `Murmur3Partitioner`ist der empfohlene Partitionierer für Amazon Keyspaces und die. `cqlsh-expansion` Das unterstützt die Amazon Keyspaces `cqlsh-expansion`

`DefaultPartitioner` nicht. Weitere Informationen finden Sie unter the section called “Mit Partitionierern arbeiten”.

Allgemeine Informationen zu `cqlsh` finden Sie unter `cqlsh`: Die CQL-Shell.

Themen

- Verwenden von`cqlsh-expansion`, um eine Verbindung zu Amazon Keyspaces herzustellen
- Wie konfiguriert man cqlsh Verbindungen für TLS manuell

Verwenden von**`cqlsh-expansion`**, um eine Verbindung zu Amazon Keyspaces herzustellen

Installation und Konfiguration des **`cqlsh-expansion`**

1. Um das `cqlsh-expansion` Python-Paket zu installieren, können Sie einen `pip` Befehl ausführen. Dadurch werden die `cqlsh-expansion` Skripts mithilfe einer Pip-Installation zusammen mit einer Datei, die eine Liste von Abhängigkeiten enthält, auf Ihrem Computer installiert. Das `--user flag` weist `pip` an, das Python-Benutzerinstallationsverzeichnis für Ihre Plattform zu verwenden. Auf einem Unix-basierten System sollte das das `~/.local/` Verzeichnis sein.

   Sie benötigen Python 3, um das zu installieren`cqlsh-expansion`, um Ihre Python-Version herauszufinden, verwenden`Python --version`. Zur Installation können Sie den folgenden Befehl ausführen.

   ```
   python3 -m pip install --user cqlsh-expansion
   ```

   Die Ausgabe sollte in etwa so aussehen.

   ```
   Collecting cqlsh-expansion
     Downloading cqlsh_expansion-0.9.6-py3-none-any.whl (153 kB)
        ######################################## 153.7/153.7 KB 3.3 MB/s eta 0:00:00
   Collecting cassandra-driver
     Downloading cassandra_driver-3.28.0-cp310-cp310-
   manylinux_2_17_x86_64.manylinux2014_x86_64.whl (19.1 MB)
        ######################################## 19.1/19.1 MB 44.5 MB/s eta 0:00:00
   ```

```
Requirement already satisfied: six>=1.12.0 in /usr/lib/python3/dist-packages (from
 cqlsh-expansion) (1.16.0)
Collecting boto3
  Downloading boto3-1.29.2-py3-none-any.whl (135 kB)
     ######################################## 135.8/135.8 KB 17.2 MB/s eta 0:00:00
Collecting cassandra-sigv4>=4.0.2
  Downloading cassandra_sigv4-4.0.2-py2.py3-none-any.whl (9.8 kB)
Collecting botocore<1.33.0,>=1.32.2
  Downloading botocore-1.32.2-py3-none-any.whl (11.4 MB)
     ######################################## 11.4/11.4 MB 60.9 MB/s eta 0:00:00
Collecting s3transfer<0.8.0,>=0.7.0
  Downloading s3transfer-0.7.0-py3-none-any.whl (79 kB)
     ######################################## 79.8/79.8 KB 13.1 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting geomet<0.3,>=0.1
  Downloading geomet-0.2.1.post1-py3-none-any.whl (18 kB)
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ######################################## 247.7/247.7 KB 33.1 MB/s eta 0:00:00
Requirement already satisfied: urllib3<2.1,>=1.25.4 in /usr/lib/python3/dist-
packages (from botocore<1.33.0,>=1.32.2->boto3->cqlsh-expansion) (1.26.5)
Requirement already satisfied: click in /usr/lib/python3/dist-packages (from
 geomet<0.3,>=0.1->cassandra-driver->cqlsh-expansion) (8.0.3)
Installing collected packages: python-dateutil, jmespath, geomet, cassandra-driver,
 botocore, s3transfer, boto3, cassandra-sigv4, cqlsh-expansion
  WARNING: The script geomet is installed in '/home/ubuntu/.local/bin' which is not
 on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this
 warning, use --no-warn-script-location.
  WARNING: The scripts cqlsh, cqlsh-expansion and cqlsh-expansion.init are
 installed in '/home/ubuntu/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this
 warning, use --no-warn-script-location.
Successfully installed boto3-1.29.2 botocore-1.32.2 cassandra-driver-3.28.0
 cassandra-sigv4-4.0.2 cqlsh-expansion-0.9.6 geomet-0.2.1.post1 jmespath-1.0.1
 python-dateutil-2.8.2 s3transfer-0.7.0
```

Wenn sich das Installationsverzeichnis nicht im befindet`PATH`, müssen Sie es gemäß den Anweisungen Ihres Betriebssystems hinzufügen. Im Folgenden finden Sie ein Beispiel für Ubuntu Linux.

```
export PATH="$PATH:/home/ubuntu/.local/bin"
```

Um zu bestätigen, dass das Paket installiert ist, können Sie den folgenden Befehl ausführen.

```
cqlsh-expansion --version
```

Die Ausgabe sollte so aussehen.

```
cqlsh 6.1.0
```

2. Um das zu konfigurieren`cqlsh-expansion`, können Sie nach der Installation ein Skript ausführen, um die folgenden Schritte automatisch auszuführen:

   1. Erstellen Sie das `.cassandra` Verzeichnis im Home-Verzeichnis des Benutzers, falls es noch nicht vorhanden ist.
   2. Kopieren Sie eine vorkonfigurierte `cqlshrc` Konfigurationsdatei in das `.cassandra` Verzeichnis.
   3. Kopieren Sie die kombinierte Zertifikatsdatei in das `.cassandra` Verzeichnis. Amazon Keyspaces verwendet dieses Zertifikat, um die sichere Verbindung mit Transport Layer Security (TLS) zu konfigurieren. Die Verschlüsselung bei der Übertragung bietet eine zusätzliche Datenschutzebene, indem Ihre Daten auf dem Weg zu und von Amazon Keyspaces verschlüsselt werden. Weitere Informationen zu Zertifikaten finden Sie unter. the section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”

   Um das Skript zunächst zu überprüfen, können Sie es im Github-Repo unter `post_install.py`aufrufen.

   Um das Skript zu verwenden, können Sie den folgenden Befehl ausführen.

   ```
   cqlsh-expansion.init
   ```

> **Note**
>
> Das Verzeichnis und die Datei, die durch das Post-Installationsskript erstellt wurden`pip uninstall`, werden bei der Deinstallation `cqlsh-expansion` von using nicht entfernt und müssen manuell gelöscht werden.

Herstellen einer Verbindung zu Amazon Keyspaces mithilfe der **`cqlsh-expansion`**

1. Konfigurieren Sie Ihre AWS-Region und fügen Sie sie als Benutzerumgebungsvariable hinzu.

   Um Ihre Standardregion als Umgebungsvariable auf einem Unix-basierten System hinzuzufügen, können Sie den folgenden Befehl ausführen. Für dieses Beispiel verwenden wir`us-east-1`.

   ```
   export AWS_DEFAULT_REGION=us-east-1
   ```

   Weitere Informationen zum Festlegen von Umgebungsvariablen, auch für andere Plattformen, finden Sie unter So legen Sie Umgebungsvariablen fest.

2. Finden Sie Ihren Service-Endpunkt.

   Wählen Sie den entsprechenden Service-Endpunkt für Ihre Region aus. Informationen zu den verfügbaren Endpunkten für Amazon Keyspaces finden Sie unter. the section called “Service-Endpunkte” In diesem Beispiel verwenden wir den Endpunkt. `cassandra.us-east-1.amazonaws.com`

3. Konfigurieren Sie die Authentifizierungsmethode.

   Die empfohlene Methode zur Erhöhung der Sicherheit ist die Verbindung mit IAM-Zugriffsschlüsseln (IAM-Benutzer, Rollen und föderierte Identitäten).

   Bevor Sie eine Verbindung mit IAM-Zugriffsschlüsseln herstellen können, müssen Sie die folgenden Schritte ausführen:

   a. Erstellen Sie einen IAM-Benutzer oder folgen Sie den bewährten Methoden und erstellen Sie eine IAM-Rolle, die IAM-Benutzer übernehmen können. Weitere Informationen zum Erstellen von IAM-Zugriffsschlüsseln finden Sie unter. the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”

b. Erstellen Sie eine IAM-Richtlinie, die der Rolle (oder dem IAM-Benutzer) mindestens schreibgeschützten Zugriff auf Amazon Keyspaces gewährt. Weitere Informationen zu den Berechtigungen, die der IAM-Benutzer oder die IAM-Rolle benötigt, um eine Verbindung zu Amazon Keyspaces herzustellen, finden Sie unter. the section called “Zugreifen auf Amazon Keyspaces-Tabellen”

c. Fügen Sie die Zugriffsschlüssel des IAM-Benutzers zu den Umgebungsvariablen des Benutzers hinzu, wie im folgenden Beispiel gezeigt.

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Weitere Informationen zum Festlegen von Umgebungsvariablen, auch für andere Plattformen, finden Sie unter So legen Sie Umgebungsvariablen fest.

> **Note**
>
> Wenn Sie von einer Amazon EC2 EC2-Instance aus eine Verbindung herstellen, müssen Sie auch eine ausgehende Regel in der Sicherheitsgruppe konfigurieren, die den Datenverkehr von der Instance zu Amazon Keyspaces zulässt. Weitere Informationen zum Anzeigen und Bearbeiten von EC2-Regeln für ausgehenden Datenverkehr finden Sie unter Regeln zu einer Sicherheitsgruppe hinzufügen im Amazon EC2 EC2-Benutzerhandbuch.

4. Stellen Sie mithilfe der `cqlsh-expansion` und SigV4-Authentifizierung eine Connect zu Amazon Keyspaces her.

Um mit dem eine Verbindung zu Amazon Keyspaces herzustellen`cqlsh-expansion`, können Sie den folgenden Befehl verwenden. Stellen Sie sicher, dass Sie den Service-Endpunkt durch den richtigen Endpunkt für Ihre Region ersetzen.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

Wenn die Verbindung erfolgreich ist, sollten Sie eine Ausgabe sehen, die dem folgenden Beispiel ähnelt.

```
Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
[cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
```

```
cqlsh current consistency level is ONE.
cqlsh>
```

Wenn Sie auf einen Verbindungsfehler stoßen, finden Sie Informationen the section called “Cqlsh-Verbindungsfehler” zur Problembehandlung unter.

- Stellen Sie mit dienstspezifischen Anmeldeinformationen eine Connect zu Amazon Keyspaces her.

  Um eine Verbindung mit der herkömmlichen Kombination aus Benutzername und Passwort herzustellen, die Cassandra für die Authentifizierung verwendet, müssen Sie zunächst dienstspezifische Anmeldeinformationen für Amazon Keyspaces erstellen, wie unter beschrieben. the section called “Erstellen Sie dienstspezifische Anmeldeinformationen” Sie müssen diesem Benutzer auch Berechtigungen für den Zugriff auf Amazon Keyspaces erteilen. Weitere Informationen finden Sie unterthe section called “Zugreifen auf Amazon Keyspaces-Tabellen”.

  Nachdem Sie dienstspezifische Anmeldeinformationen und Berechtigungen für den Benutzer erstellt haben, müssen Sie die `cqlshrc` Datei aktualisieren, die sich normalerweise im Benutzerverzeichnispfad befindet. `~/.cassandra/` Gehen Sie in der `cqlshrc` Datei zum `[authentication]` Abschnitt Cassandra und kommentieren Sie das SigV4-Modul und die Klasse unter `[auth_provider]` mithilfe des Zeichens „;“ aus, wie im folgenden Beispiel gezeigt.

  ```
  [auth_provider]

  ; module = cassandra_sigv4.auth

  ; classname = SigV4AuthProvider
  ```

  Nachdem Sie die `cqlshrc` Datei aktualisiert haben, können Sie mit dem folgenden Befehl eine Verbindung zu Amazon Keyspaces mit dienstspezifischen Anmeldeinformationen herstellen.

  ```
  cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 -u myUserName -
  p myPassword --ssl
  ```

Bereinigen

- Um das `cqlsh-expansion` Paket zu entfernen, können Sie den `pip uninstall` Befehl verwenden.

  ```
  pip3 uninstall cqlsh-expansion
  ```

  Der `pip3 uninstall` Befehl entfernt nicht das Verzeichnis und die zugehörigen Dateien, die durch das Post-Installationsskript erstellt wurden. Um den Ordner und die Dateien zu entfernen, die durch das Post-Installationsskript erstellt wurden, können Sie das `.cassandra` Verzeichnis löschen.

Wie konfiguriert man **`cqlsh`** Verbindungen für TLS manuell

Amazon Keyspaces akzeptiert nur sichere Verbindungen mit Transport Layer Security (TLS). Sie können das `cqlsh-expansion` Hilfsprogramm verwenden, das die Zertifikate automatisch für Sie herunterlädt und eine vorkonfigurierte `cqlshrc` Konfigurationsdatei installiert. Weitere Informationen finden Sie the section called “Verwendung der `cqlsh-expansion`” auf dieser Seite.

Wenn Sie die Zertifikate herunterladen und die Verbindung manuell konfigurieren möchten, können Sie dies mit den folgenden Schritten tun.

1. Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.
   a. AmazonRootCA1
   b. AmazonRootCA 2
   c. AmazonRootCA 3
   d. AmazonRootCA 4
   e. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

   Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

   ```
   curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
   curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
   curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
   curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
   curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
   ```

> **Note**
>
> Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2 CA verankert waren. AWS migriert alle AWS-Regionen auf Zertifikate, die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden. Während dieses Übergangs konfigurieren Sie die Clients so, dass sie sowohl den Amazon Root-CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

2. Kombinieren Sie alle heruntergeladenen Zertifikate in einer einzigen `pem` Datei mit dem Namen *`keyspaces-bundle.pem`* in unseren Beispielen. Sie erreichen dies mit dem -Befehl- Notieren Sie sich den Pfad zur Datei, den Sie später benötigen.

```
cat AmazonRootCA1.pem \
 AmazonRootCA2.pem \
 AmazonRootCA3.pem \
 AmazonRootCA4.pem \
 sf-class2-root.crt \
 > keyspaces-bundle.pem
```

3. Öffnen Sie beispielsweise die `cqlshrc` Konfigurationsdatei im Cassandra-Home-Verzeichnis `${HOME}/.cassandra/cqlshrc` und fügen Sie die folgenden Zeilen hinzu.

```
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory

[ssl]
validate = true
certfile =  path_to_file/keyspaces-bundle.pem
```

Verwenden von AWS CLI , um eine Verbindung zu Amazon Keyspaces herzustellen

Sie können die AWS Command Line Interface (AWS CLI) verwenden, um mehrere AWS Dienste von der Befehlszeile aus zu steuern und sie mithilfe von Skripten zu automatisieren. Mit Amazon Keyspaces können Sie DDL-Operationen (Data Definition Language) verwenden, z. B. das Erstellen

einer Tabelle. AWS CLI Darüber hinaus können Sie Dienste und Tools wie Terraform (Infrastructure as Code, IaC) verwenden. AWS CloudFormation

Bevor Sie das AWS CLI mit Amazon Keyspaces verwenden können, benötigen Sie eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Eine vollständige Liste aller Befehle, die für Amazon Keyspaces in verfügbar sind AWS CLI, finden Sie in der AWS CLI Befehlsreferenz.

Themen

- Herunterladen und Konfigurieren des AWS CLI
- Verwenden von AWS CLI mit Amazon Keyspaces

Herunterladen und Konfigurieren des AWS CLI

Das AWS CLI ist verfügbar unterhttps://aws.amazon.com/cli. Sie kann auf Windows, macOS oder Linux ausgeführt werden. Gehen Sie nach dem Herunterladen wie folgt vor AWS CLI, um es zu installieren und zu konfigurieren:

1. Gehen Sie AWS Command Line Interface zum Benutzerhandbuch
2. Folgen Sie den Anweisungen zur Installation AWS CLI und Konfiguration von AWS CLI

Verwenden von AWS CLI mit Amazon Keyspaces

Das Befehlszeilenformat besteht aus einem Amazon Keyspaces-Operationsnamen, gefolgt von den Parametern für diesen Vorgang. Das AWS CLI unterstützt eine Kurzsyntax für die Parameterwerte sowie JSON. Die folgenden Amazon Keyspaces-Beispiele verwenden AWS CLI Kurzsyntax. Weitere Informationen finden Sie unter Verwenden der Kurzsyntax mit der AWS CLI.

Der folgende Befehl erstellt einen Schlüsselraum mit dem Namen catalog.

```
aws keyspaces create-keyspace --keyspace-name 'catalog'
```

Der Befehl gibt die Ressource Amazon Resource Name (ARN) in der Ausgabe zurück.

```
{
```

```
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/"
}
```

Um zu bestätigen, dass der Keyspace-Katalog existiert, können Sie den folgenden Befehl verwenden.

```
aws keyspaces get-keyspace --keyspace-name 'catalog'
```

Die Ausgabe des Befehls gibt die folgenden Werte zurück.

```
{
    "keyspaceName": "catalog",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/"
}
```

Der folgende Befehl erstellt eine Tabelle mit dem Namen book_awards. Der Partitionsschlüssel der Tabelle besteht aus den Spalten `year` `award` und der Gruppierungsschlüssel besteht aus den Spalten`rank`, `category` und beide Clusterspalten verwenden die aufsteigende Sortierreihenfolge. (Für eine bessere Lesbarkeit werden lange Befehle in diesem Abschnitt über mehrere Zeilen verteilt.)

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'book_awards'
            --schema-definition 'allColumns=[{name=year,type=int},
{name=award,type=text},{name=rank,type=int},
            {name=category,type=text}, {name=author,type=text},
{name=book_title,type=text},{name=publisher,type=text}],
            partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},{name=rank,orderBy=ASC}]'
```

Dieser Befehl führt zu der folgenden Ausgabe.

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/table/
book_awards"
}
```

Um die Metadaten und Eigenschaften der Tabelle zu bestätigen, können Sie den folgenden Befehl verwenden.

```
aws keyspaces get-table --keyspace-name 'catalog' --table-name 'book_awards'
```

Dieser Befehl gibt die folgende Ausgabe zurück.

```
{
    "keyspaceName": "catalog",
    "tableName": "book_awards",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/table/
book_awards",
    "creationTimestamp": 1645564368.628,
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "year",
                "type": "int"
            },
            {
                "name": "award",
                "type": "text"
            },
            {
                "name": "category",
                "type": "text"
            },
            {
                "name": "rank",
                "type": "int"
            },
            {
                "name": "author",
                "type": "text"
            },
            {
                "name": "book_title",
                "type": "text"
            },
            {
                "name": "publisher",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "year"
            },
```

```
            {
                "name": "award"
            }
        ],
        "clusteringKeys": [
            {
                "name": "category",
                "orderBy": "ASC"
            },
            {
                "name": "rank",
                "orderBy": "ASC"
            }
        ],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": 1645564368.628
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Beim Erstellen von Tabellen mit komplexen Schemas kann es hilfreich sein, die Schemadefinition der Tabelle aus einer JSON-Datei zu laden. Das Folgende ist ein Beispiel dafür. Laden Sie die JSON-Beispieldatei für die Schemadefinition von schema_definition.zip herunter und extrahieren Sie sie`schema_definition.json`, wobei Sie sich den Pfad zur Datei notieren. In diesem Beispiel befindet sich die JSON-Datei mit der Schemadefinition im aktuellen Verzeichnis. Informationen zu verschiedenen Dateipfadoptionen finden Sie unter So laden Sie Parameter aus einer Datei.

```
aws keyspaces create-table --keyspace-name 'catalog'
            --table-name 'book_awards' --schema-definition 'file://
schema_definition.json'
```

Die folgenden Beispiele zeigen, wie Sie eine einfache Tabelle mit dem Namen MyTable mit zusätzlichen Optionen erstellen. Beachten Sie, dass die Befehle zur besseren Lesbarkeit in separate Zeilen aufgeteilt sind. Dieser Befehl zeigt, wie eine Tabelle erstellt wird und:

- stellt den Kapazitätsmodus der Tabelle ein
- aktivieren Sie die Point-in-time Wiederherstellung für die Tabelle
- setzen Sie den Standardwert Time to Live (TTL) für die Tabelle auf ein Jahr
- fügen Sie zwei Tags für die Tabelle hinzu

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --capacity-specification
 'throughputMode=PROVISIONED,readCapacityUnits=5,writeCapacityUnits=5'
            --point-in-time-recovery 'status=ENABLED'
            --default-time-to-live '31536000'
            --tags 'key=env,value=test' 'key=dpt,value=sec'
```

Dieses Beispiel zeigt, wie Sie eine neue Tabelle erstellen, die einen vom Kunden verwalteten Schlüssel für die Verschlüsselung verwendet und TTL aktiviert hat, sodass Sie Ablaufdaten für Spalten und Zeilen festlegen können. Um dieses Beispiel auszuführen, müssen Sie den Ressourcen-ARN für den vom Kunden verwalteten AWS KMS Schlüssel durch Ihren eigenen Schlüssel ersetzen und sicherstellen, dass Amazon Keyspaces Zugriff darauf hat.

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --encryption-specification
 'type=CUSTOMER_MANAGED_KMS_KEY,kmsKeyIdentifier=arn:aws:kms:us-
east-1:111122223333:key/11111111-2222-3333-4444-555555555555'
            --ttl 'status=ENABLED'
```

Mithilfe der API eine Verbindung zu Amazon Keyspaces herstellen

Sie können das AWS SDK und das AWS Command Line Interface (AWS CLI) verwenden, um interaktiv mit Amazon Keyspaces zu arbeiten. Sie können die API für DDL-Operationen (Data Language Definition) verwenden, z. B. für die Erstellung eines Schlüsselraums oder einer Tabelle. Darüber hinaus können Sie Dienste und Tools wie Terraform (Infrastructure as Code, IaC) verwenden. AWS CloudFormation

Bevor Sie das AWS CLI mit Amazon Keyspaces verwenden können, benötigen Sie eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Eine vollständige Liste aller Operationen, die für Amazon Keyspaces in der API verfügbar sind, finden Sie unter Amazon Keyspaces API-Referenz.

Verwenden eines Cassandra-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

Sie können viele Open-Source-Cassandra-Treiber von Drittanbietern verwenden, um eine Verbindung zu Amazon Keyspaces herzustellen. Amazon Keyspaces ist mit Cassandra-Treibern kompatibel, die Apache Cassandra Version 3.11.2 unterstützen. Dies sind die Treiber und neuesten Versionen, die wir getestet haben und die wir zur Verwendung mit Amazon Keyspaces empfehlen:

- `Java v3.3`
- `Java v4.17`
- `Python Cassandra-driver 3.29.1`
- `Node.js cassandra driver -v 4.7.2`
- `GO using GOCQL v1.6`
- `.NET CassandraCSharpDriver -v 3.20.1`

Weitere Informationen zu Cassandra-Treibern finden Sie unter Apache Cassandra-Client-Treiber.

> **Note**
>
> Um Ihnen den Einstieg zu erleichtern, können Sie sich Ende-zu-Ende-Codebeispiele ansehen und herunterladen, die Verbindungen zu Amazon Keyspaces mit gängigen Treibern herstellen. Beispiele für Amazon Keyspaces finden Sie unter. GitHub

Die Tutorials in diesem Kapitel enthalten eine einfache CQL-Abfrage, um zu bestätigen, dass die Verbindung zu Amazon Keyspaces erfolgreich hergestellt wurde. Informationen zum Arbeiten mit Schlüsselräumen und Tabellen, nachdem Sie eine Verbindung zu einem Amazon Keyspaces-Endpunkt hergestellt haben, finden Sie unter. CQL-Sprachreferenz Eine schrittweise Anleitung, die zeigt, wie Sie von einem Amazon VPC-Endpunkt aus eine Verbindung zu Amazon Keyspaces herstellen, finden Sie unter. the section called “Verbindung mit VPC-Endpunkten herstellen”

Themen

- Verwenden eines Cassandra-Java-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces
- Verwenden eines Cassandra-Python-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces
- Verwenden eines Node.js Cassandra-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces
- Verwenden eines Cassandra.NET Core-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces
- Verwenden eines Cassandra Go-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces
- Verwenden eines Cassandra Perl-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

Verwenden eines Cassandra-Java-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

In diesem Abschnitt erfahren Sie, wie Sie mithilfe eines Java-Client-Treibers eine Verbindung zu Amazon Keyspaces herstellen.

> **Note**
>
> Java 17 und der DataStax Java-Treiber 4.17 werden derzeit nur in der Betaversion unterstützt. Weitere Informationen finden Sie unter https://docs.datastax.com/en/developer/java-driver/4.17/upgrade_guide/.

Um Benutzern und Anwendungen Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces-Ressourcen zur Verfügung zu stellen, können Sie einen der folgenden Schritte ausführen:

- Erstellen Sie dienstspezifische Anmeldeinformationen, die einem bestimmten Benutzer AWS Identity and Access Management (IAM) zugeordnet sind.
- Zur Erhöhung der Sicherheit empfehlen wir, IAM-Zugriffsschlüssel für IAM-Identitäten zu erstellen, die für alle Dienste verwendet werden. AWS Das Amazon Keyspaces SigV4-Authentifizierungs-Plugin für Cassandra-Client-Treiber ermöglicht es Ihnen, Anrufe an Amazon Keyspaces mithilfe von IAM-Zugriffsschlüsseln anstelle von Benutzername und Passwort zu authentifizieren. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

> **Note**
>
> Ein Beispiel zur Verwendung von Amazon Keyspaces mit Spring Boot finden Sie unter. https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring

Themen

- Bevor Sie beginnen
- Step-by-step Tutorial zum Herstellen einer Verbindung zu Amazon Keyspaces mithilfe des DataStax Java-Treibers für Apache Cassandra unter Verwendung dienstspezifischer Anmeldeinformationen
- Step-by-step Tutorial zum Herstellen einer Verbindung zu Amazon Keyspaces mithilfe des DataStax 4.x-Java-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins
- Stellen Sie mithilfe des DataStax 3.x-Java-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Bevor Sie beginnen

Um eine Verbindung zu Amazon Keyspaces herzustellen, müssen Sie die folgenden Aufgaben ausführen, bevor Sie beginnen können.

1. Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern.

   a. Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.

      1. AmazonRootCA1
      2. AmazonRootCA2
      3. AmazonRootCA3
      4. AmazonRootCA4
      5. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

      Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

      ```
      curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
      curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
      curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
      curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
      curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
      ```

      > **Note**
      >
      > Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2-CA verankert waren. AWS migriert alle Zertifikate AWS-Regionen , die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden. Konfigurieren Sie während dieser Umstellung die Clients so, dass sie sowohl Amazon Root CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

   b. Konvertieren Sie die digitalen Zertifikate in TrustStore-Dateien und fügen Sie sie dem Keystore hinzu.

      ```
      openssl x509 -outform der -in AmazonRootCA1.pem -out temp_file.der
      ```

```
keytool -import -alias amazon-root-ca-1 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA2.pem -out temp_file.der
keytool -import -alias amazon-root-ca-2 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA3.pem -out temp_file.der
keytool -import -alias amazon-root-ca-3 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in AmazonRootCA4.pem -out temp_file.der
keytool -import -alias amazon-root-ca-4 -keystore cassandra_truststore.jks -
file temp_file.der

openssl x509 -outform der -in sf-class2-root.crt -out temp_file.der
keytool -import -alias cassandra -keystore cassandra_truststore.jks -file
 temp_file.der
```

Im letzten Schritt müssen Sie ein Passwort für den Keystore erstellen und jedem Zertifikat vertrauen. Der interaktive Befehl sieht so aus.

```
Enter keystore password:
Re-enter new password:
Owner: CN=Amazon Root CA 1, O=Amazon, C=US
Issuer: CN=Amazon Root CA 1, O=Amazon, C=US
Serial number: 66c9fcf99bf8c0a39e2f0788a43e696365bca
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sun Jan 17 00:00:00 UTC 2038
Certificate fingerprints:
     SHA1: 8D:A7:F9:65:EC:5E:FC:37:91:0F:1C:6E:59:FD:C1:CC:6A:6E:DE:16
     SHA256:
 8E:CD:E6:88:4F:3D:87:B1:12:5B:A3:1A:C3:FC:B1:3D:70:16:DE:7F:57:CC:90:4F:E1:CB:97:C6:AE
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]
```

```
#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 84 18 CC 85 34 EC BC 0C   94 94 2E 08 59 9C C7 B2  ....4.......Y...
0010: 10 4E 0A 08                                        .N..
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 2, O=Amazon, C=US
Issuer: CN=Amazon Root CA 2, O=Amazon, C=US
Serial number: 66c9fd29635869f0a0fe58678f85b26bb8a37
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
     SHA1: 5A:8C:EF:45:D7:A6:98:59:76:7A:8C:8B:44:96:B5:78:CF:47:4B:1A
     SHA256:
 1B:A5:B2:AA:8C:65:40:1A:82:96:01:18:F8:0B:EC:4F:62:30:4D:83:CE:C4:71:3A:19:C3:9C:01:1E
Signature algorithm name: SHA384withRSA
Subject Public Key Algorithm: 4096-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
```

```
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: B0 0C F0 4C 30 F4 05 58   02 48 FD 33 E5 52 AF 4B  ...L0..X.H.3.R.K
0010: 84 E3 66 52                                        ..fR
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 3, O=Amazon, C=US
Issuer: CN=Amazon Root CA 3, O=Amazon, C=US
Serial number: 66c9fd5749736663f3b0b9ad9e89e7603f24a
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
     SHA1: 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E
     SHA256:
 18:CE:6C:FE:7B:F1:4E:60:B2:E3:47:B8:DF:E8:68:CB:31:D0:2E:BB:3A:DA:27:15:69:F5:03:43:B4
Signature algorithm name: SHA256withECDSA
Subject Public Key Algorithm: 256-bit EC (secp256r1) key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: AB B6 DB D7 06 9E 37 AC   30 86 07 91 70 C7 9C C4  ......7.0...p...
```

```
0010: 19 B1 78 C0                                        ..x.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 4, O=Amazon, C=US
Issuer: CN=Amazon Root CA 4, O=Amazon, C=US
Serial number: 66c9fd7c1bb104c2943e5717b7b2cc81ac10e
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
     SHA1: F6:10:84:07:D6:F8:BB:67:98:0C:C2:E2:44:C2:EB:AE:1C:EF:63:BE
     SHA256:
 E3:5D:28:41:9E:D0:20:25:CF:A6:90:38:CD:62:39:62:45:8D:A5:C6:95:FB:DE:A3:C2:2B:0B:FB:25
Signature algorithm name: SHA384withECDSA
Subject Public Key Algorithm: 384-bit EC (secp384r1) key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: D3 EC C7 3A 65 6E CC E1   DA 76 9A 56 FB 9C F3 86  ...:en...v.V....
0010: 6D 57 E5 81                                        mW..
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

```
Enter keystore password:
Owner: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US
Issuer: OU=Starfield Class 2 Certification Authority, O="Starfield
 Technologies, Inc.", C=US
Serial number: 0
Valid from: Tue Jun 29 17:39:16 UTC 2004 until: Thu Jun 29 17:39:16 UTC 2034
Certificate fingerprints:
     SHA1: AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A
     SHA256:
 14:65:FA:20:53:97:B8:76:FA:A6:F0:A9:95:8E:55:90:E4:0F:CC:7F:AA:4F:B7:C2:C8:67:75:21:FB
Signature algorithm name: SHA1withRSA (weak)
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
[OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
 Inc.", C=US]
SerialNumber: [    00]
]

#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: BF 5F B7 D1 CE DD 1F 86   F4 5B 55 AC DC D7 10 C2  ._.......[U.....
0010: 0E A9 88 E7                                        ....
]
]


Warning:
```

```
The input uses the SHA1withRSA signature algorithm which is considered a
 security risk. This algorithm will be disabled in a future update.

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

2. Hängen Sie die TrustStore-Datei an die JVM-Argumente an:

```
-Djavax.net.ssl.trustStore=path_to_file/cassandra_truststore.jks
-Djavax.net.ssl.trustStorePassword=my_password
```

Step-by-step Tutorial zum Herstellen einer Verbindung zu Amazon Keyspaces mithilfe des DataStax Java-Treibers für Apache Cassandra unter Verwendung dienstspezifischer Anmeldeinformationen

Das folgende Tutorial führt Sie Schritt für Schritt durch die Herstellung einer Verbindung zu Amazon Keyspaces mithilfe eines Java-Treibers für Cassandra unter Verwendung dienstspezifischer Anmeldeinformationen. Insbesondere verwenden Sie die 4.0-Version des DataStax Java-Treibers für Apache Cassandra.

Themen

- Schritt 1: Voraussetzungen
- Schritt 2: Konfigurieren Sie den Treiber
- Schritt 3: Führen Sie die Beispielanwendung aus

Schritt 1: Voraussetzungen

Um diesem Tutorial zu folgen, müssen Sie dienstspezifische Anmeldeinformationen generieren und den DataStax Java-Treiber für Apache Cassandra zu Ihrem Java-Projekt hinzufügen.

- Generieren Sie servicespezifische Anmeldeinformationen für Ihren Amazon Keyspaces-IAM-Benutzer, indem Sie die Schritte unter ausführen. the section called “Erstellen Sie dienstspezifische Anmeldeinformationen” Wenn Sie es vorziehen, IAM-Zugriffsschlüssel für die Authentifizierung zu verwenden, finden Sie weitere Informationen unter. the section called “Authentifizierungs-Plugin für Java 4.x”
- Fügen Sie den DataStax Java-Treiber für Apache Cassandra zu Ihrem Java-Projekt hinzu. Stellen Sie sicher, dass Sie eine Version des Treibers verwenden, die Apache Cassandra 3.11.2

unterstützt. Weitere Informationen finden Sie in der Dokumentation zum DataStax Java-Treiber für Apache Cassandra.

Schritt 2: Konfigurieren Sie den Treiber

Sie können Einstellungen für den DataStax Java Cassandra-Treiber angeben, indem Sie eine Konfigurationsdatei für Ihre Anwendung erstellen. Diese Konfigurationsdatei überschreibt die Standardeinstellungen und weist den Treiber an, über Port 9142 eine Verbindung zum Amazon Keyspaces-Serviceendpunkt herzustellen. Eine Liste der verfügbaren Service-Endpunkte finden Sie unter. the section called “Service-Endpunkte”

Erstellen Sie eine Konfigurationsdatei und speichern Sie die Datei im Ressourcenordner der Anwendung, z. B. `src/main/resources/application.conf` Öffnen Sie die folgenden `application.conf` Konfigurationseinstellungen und fügen Sie sie hinzu.

1. Authentifizierungsanbieter — Erstellen Sie den Authentifizierungsanbieter mit der `PlainTextAuthProvider` Klasse. *`ServiceUserName`*und *`ServicePassword`* sollte mit dem Benutzernamen und dem Passwort übereinstimmen, das Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben, indem Sie die Schritte unter ausgeführt haben. Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen

   > **Note**
   >
   > Sie können kurzfristige Anmeldeinformationen verwenden, indem Sie das Authentifizierungs-Plugin für den DataStax Java-Treiber für Apache Cassandra verwenden, anstatt die Anmeldeinformationen in Ihrer Treiberkonfigurationsdatei fest zu codieren. Um mehr zu erfahren, folgen Sie den Anweisungen für. the section called “Authentifizierungs-Plugin für Java 4.x”

2. Lokales Rechenzentrum — Stellen Sie den Wert für `local-datacenter` die Region ein, mit der Sie eine Verbindung herstellen. Wenn die Anwendung beispielsweise eine Verbindung herstellt`cassandra.us-east-2.amazonaws.com`, stellen Sie das lokale Rechenzentrum auf ein`us-east-2`. Alle verfügbaren AWS-Regionen Informationen finden Sie unter???. Auf `slow-replica-avoidance = false` Lastenausgleich für weniger Knoten eingestellt.
3. SSL/TLS— Initialisieren Sie das, SSLEngineFactory indem Sie der Konfigurationsdatei einen Abschnitt mit einer einzigen Zeile hinzufügen, der die Klasse mit `class = DefaultSslEngineFactory` angibt. Geben Sie den Pfad zur TrustStore-Datei und das

Passwort an, das Sie zuvor erstellt haben. Amazon Keyspaces unterstützt keine `hostname-validation` Peers. Setzen Sie diese Option daher auf false.

```
datastax-java-driver {

    basic.contact-points = [ "cassandra.us-east-2.amazonaws.com:9142"]
    advanced.auth-provider{
        class = PlainTextAuthProvider
        username = "ServiceUserName"
        password = "ServicePassword"
    }
    basic.load-balancing-policy {
        local-datacenter = "us-east-2"
        slow-replica-avoidance = false
    }

    advanced.ssl-engine-factory {
        class = DefaultSslEngineFactory
        truststore-path = "./src/main/resources/cassandra_truststore.jks"
        truststore-password = "my_password"
        hostname-validation = false
      }
}
```

> **Note**
>
> Anstatt den Pfad zum TrustStore in der Konfigurationsdatei hinzuzufügen, können Sie den TrustStore-Pfad auch direkt im Anwendungscode hinzufügen oder Sie können den Pfad zum TrustStore zu Ihren JVM-Argumenten hinzufügen.

Schritt 3: Führen Sie die Beispielanwendung aus

Dieses Codebeispiel zeigt eine einfache Befehlszeilenanwendung, die mithilfe der zuvor erstellten Konfigurationsdatei einen Verbindungspool zu Amazon Keyspaces erstellt. Es bestätigt, dass die Verbindung durch Ausführen einer einfachen Abfrage hergestellt wird.

```
package <your package>;
// add the following imports to your project
import com.datastax.oss.driver.api.core.CqlSession;
```

```
import com.datastax.oss.driver.api.core.config.DriverConfigLoader;
import com.datastax.oss.driver.api.core.cql.ResultSet;
import com.datastax.oss.driver.api.core.cql.Row;

public class App
{

    public static void main( String[] args )
    {
        //Use DriverConfigLoader to load your configuration file
        DriverConfigLoader loader =
 DriverConfigLoader.fromClasspath("application.conf");
        try (CqlSession session = CqlSession.builder()
                .withConfigLoader(loader)
                .build()) {

            ResultSet rs = session.execute("select * from system_schema.keyspaces");
            Row row = rs.one();
            System.out.println(row.getString("keyspace_name"));
        }
    }
}
```

> **Note**
>
> Verwenden Sie einen `try` Block, um die Verbindung herzustellen, um sicherzustellen, dass sie immer geschlossen ist. Wenn Sie keinen `try` Block verwenden, denken Sie daran, Ihre Verbindung zu schließen, um zu vermeiden, dass Ressourcen verloren gehen.

Step-by-step Tutorial zum Herstellen einer Verbindung zu Amazon Keyspaces mithilfe des DataStax 4.x-Java-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins

Im folgenden Abschnitt wird beschrieben, wie Sie das SigV4-Authentifizierungs-Plugin für den DataStax Open-Source-4.x-Java-Treiber für Apache Cassandra verwenden, um auf Amazon Keyspaces (für Apache Cassandra) zuzugreifen. Das Plugin ist im Repository verfügbar. GitHub

Das SigV4-Authentifizierungs-Plugin ermöglicht es Ihnen, IAM-Anmeldeinformationen für Benutzer oder Rollen zu verwenden, wenn Sie eine Verbindung zu Amazon Keyspaces herstellen. Anstatt einen Benutzernamen und ein Passwort zu benötigen, signiert dieses Plugin API-Anfragen mithilfe

von Zugriffsschlüsseln. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Schritt 1: Voraussetzungen

Um diesem Tutorial zu folgen, müssen Sie die folgenden Aufgaben ausführen.

- Wenn Sie dies noch nicht getan haben, erstellen Sie Anmeldeinformationen für Ihren IAM-Benutzer oder Ihre IAM-Rolle, indem Sie den Schritten unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ” folgen. In diesem Tutorial wird davon ausgegangen, dass die Zugriffsschlüssel als Umgebungsvariablen gespeichert sind. Weitere Informationen finden Sie unter the section called “Verwalten von Zugriffsschlüsseln”.
- Fügen Sie den DataStax Java-Treiber für Apache Cassandra zu Ihrem Java-Projekt hinzu. Stellen Sie sicher, dass Sie eine Version des Treibers verwenden, die Apache Cassandra 3.11.2 unterstützt. Weitere Informationen finden Sie in der Dokumentation zum DataStax Java-Treiber für Apache Cassandra.
- Fügen Sie das Authentifizierungs-Plugin zu Ihrer Anwendung hinzu. Das Authentifizierungs-Plugin unterstützt Version 4.x des DataStax Java-Treibers für Apache Cassandra. Wenn Sie Apache Maven oder ein Build-System verwenden, das Maven-Abhängigkeiten verwenden kann, fügen Sie Ihrer Datei die folgenden Abhängigkeiten hinzu. `pom.xml`

> **Important**
>
> Ersetzen Sie die Version des Plugins durch die neueste Version, wie im GitHub Repository angezeigt.

```
<dependency>
        <groupId>software.aws.mcs</groupId>
        <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin</artifactId>
        <version>4.0.9</version>
</dependency>
```

Schritt 2: Konfigurieren Sie den Treiber

Sie können Einstellungen für den DataStax Java Cassandra-Treiber angeben, indem Sie eine Konfigurationsdatei für Ihre Anwendung erstellen. Diese Konfigurationsdatei überschreibt die Standardeinstellungen und weist den Treiber an, über Port 9142 eine Verbindung zum Amazon

Keyspaces-Serviceendpunkt herzustellen. Eine Liste der verfügbaren Service-Endpunkte finden Sie unter. the section called “Service-Endpunkte”

Erstellen Sie eine Konfigurationsdatei und speichern Sie die Datei im Ressourcenordner der Anwendung, z. B. `src/main/resources/application.conf` Öffnen Sie die folgenden `application.conf` Konfigurationseinstellungen und fügen Sie sie hinzu.

1. Authentifizierungsanbieter — Stellen Sie den `advanced.auth-provider.class` auf eine neue Instanz von ein`software.aws.mcs.auth.SigV4AuthProvider`. Das SigV4AuthProvider ist der Authentifizierungshandler, der vom Plugin für die Durchführung der SigV4-Authentifizierung bereitgestellt wird.
2. Lokales Rechenzentrum — Stellen Sie den Wert für `local-datacenter` die Region ein, mit der Sie eine Verbindung herstellen. Wenn die Anwendung beispielsweise eine Verbindung herstellt`cassandra.us-east-2.amazonaws.com`, stellen Sie das lokale Rechenzentrum auf ein`us-east-2`. Alle verfügbaren AWS-Regionen Informationen finden Sie unter???. Auf `slow-replica-avoidance = false` Lastenausgleich für alle verfügbaren Knoten eingestellt.
3. Idempotenz — Legt die Standardeinstellung `idempotence` für die Anwendung auf fest, um den Treiber so `true` zu konfigurieren, dass fehlgeschlagene /-Anfragen immer wiederholt werden. read/write prepare/execute Dies ist eine bewährte Methode für verteilte Anwendungen, die bei vorübergehenden Ausfällen hilft, indem fehlgeschlagene Anfragen wiederholt werden.
4. SSL/TLS— Initialisieren Sie das, SSLEngineFactory indem Sie der Konfigurationsdatei einen Abschnitt mit einer einzigen Zeile hinzufügen, in dem die Klasse mit angegeben ist. `class = DefaultSslEngineFactory` Geben Sie den Pfad zur TrustStore-Datei und das Passwort an, das Sie zuvor erstellt haben. Amazon Keyspaces unterstützt keine `hostname-validation` Peers. Setzen Sie diese Option daher auf false.
5. Verbindungen — Erstellen Sie mindestens 3 lokale Verbindungen pro Endpunkt, indem Sie Folgendes festlegen`local.size = 3`. Dies ist eine bewährte Methode, die Ihrer Anwendung hilft, Overhead und Datenverkehrsspitzen zu bewältigen. Weitere Informationen zur Berechnung der Anzahl der lokalen Verbindungen pro Endpunkt, die Ihre Anwendung auf der Grundlage der erwarteten Datenverkehrsmuster benötigt, finden Sie unterthe section called “Wie konfiguriert man Verbindungen”.
6. Wiederholungsrichtlinie — Implementieren Sie die Amazon Keyspaces-Wiederholungsrichtlinie `AmazonKeyspacesExponentialRetryPolicy` anstelle der im `DefaultRetryPolicy` Cassandra-Treiber enthaltenen Richtlinie. Auf diese Weise können Sie die Anzahl der Wiederholungsversuche so konfigurieren, dass sie Ihren Anforderungen entspricht. `AmazonKeyspacesExponentialRetryPolicy` Standardmäßig ist die Anzahl der

Wiederholungsversuche für den auf `AmazonKeyspacesExponentialRetryPolicy` 3 festgelegt. Weitere Informationen finden Sie unter the section called “Wie konfiguriert man Richtlinien für Wiederholungsversuche”.

7. Vorbereitete Anweisungen — Auf „Falsch“ setzen`prepare-on-all-nodes`, um die Netzwerknutzung zu optimieren.

```
datastax-java-driver {
    basic {
        contact-points = [ "cassandra.us-east-2.amazonaws.com:9142"]
        request {
            timeout = 2 seconds
            consistency = LOCAL_QUORUM
            page-size = 1024
            default-idempotence = true
        }
        load-balancing-policy {
            local-datacenter = "us-east-2"
            class = DefaultLoadBalancingPolicy
            slow-replica-avoidance = false
        }
    }
    advanced {
        auth-provider {
            class = software.aws.mcs.auth.SigV4AuthProvider
            aws-region = us-east-2
        }
        ssl-engine-factory {
            class = DefaultSslEngineFactory
            truststore-path = "./src/main/resources/cassandra_truststore.jks"
            truststore-password = "my_password"
            hostname-validation = false
        }
        connection {
      connect-timeout = 5 seconds
      max-requests-per-connection = 512
      pool {
                local.size = 3
      }
        }
       retry-policy {
           class =  com.aws.ssa.keyspaces.retry.AmazonKeyspacesExponentialRetryPolicy
```

```
     max-attempts = 3
     min-wait = 10 mills
     max-wait = 100 mills
       }
       prepared-statements {
     prepare-on-all-nodes = false
       }
    }
}
```

> **Note**
>
> Anstatt den Pfad zum TrustStore in der Konfigurationsdatei hinzuzufügen, können Sie den TrustStore-Pfad auch direkt im Anwendungscode hinzufügen oder Sie können den Pfad zum TrustStore zu Ihren JVM-Argumenten hinzufügen.

Schritt 3: Führen Sie die Anwendung aus

Dieses Codebeispiel zeigt eine einfache Befehlszeilenanwendung, die mithilfe der zuvor erstellten Konfigurationsdatei einen Verbindungspool zu Amazon Keyspaces erstellt. Es bestätigt, dass die Verbindung durch Ausführen einer einfachen Abfrage hergestellt wird.

```
package <your package>;
// add the following imports to your project
import com.datastax.oss.driver.api.core.CqlSession;
import com.datastax.oss.driver.api.core.config.DriverConfigLoader;
import com.datastax.oss.driver.api.core.cql.ResultSet;
import com.datastax.oss.driver.api.core.cql.Row;

public class App
{

    public static void main( String[] args )
    {
        //Use DriverConfigLoader to load your configuration file
        DriverConfigLoader loader =
 DriverConfigLoader.fromClasspath("application.conf");
        try (CqlSession session = CqlSession.builder()
                .withConfigLoader(loader)
                .build()) {
```

```
            ResultSet rs = session.execute("select * from system_schema.keyspaces");
            Row row = rs.one();
            System.out.println(row.getString("keyspace_name"));
        }
    }
}
```

> **Note**
>
> Verwenden Sie einen `try` Block, um die Verbindung herzustellen, um sicherzustellen, dass sie immer geschlossen ist. Wenn Sie keinen `try` Block verwenden, denken Sie daran, Ihre Verbindung zu schließen, um zu vermeiden, dass Ressourcen verloren gehen.

Stellen Sie mithilfe des DataStax 3.x-Java-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Im folgenden Abschnitt wird beschrieben, wie Sie das SigV4-Authentifizierungs-Plugin für den DataStax 3.x-Open-Source-Java-Treiber für Apache Cassandra verwenden, um auf Amazon Keyspaces zuzugreifen. Das Plugin ist im Repository verfügbar. GitHub

Das SigV4-Authentifizierungs-Plugin ermöglicht es Ihnen, IAM-Anmeldeinformationen für Benutzer und Rollen zu verwenden, wenn Sie eine Verbindung zu Amazon Keyspaces herstellen. Anstatt einen Benutzernamen und ein Passwort zu benötigen, signiert dieses Plugin API-Anfragen mithilfe von Zugriffsschlüsseln. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Schritt 1: Voraussetzungen

Um dieses Codebeispiel auszuführen, müssen Sie zunächst die folgenden Aufgaben ausführen.

- Erstellen Sie Anmeldeinformationen für Ihren IAM-Benutzer oder Ihre IAM-Rolle, indem Sie den Schritten unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ” folgen. In diesem Tutorial wird davon ausgegangen, dass die Zugriffsschlüssel als Umgebungsvariablen gespeichert sind. Weitere Informationen finden Sie unter the section called “Verwalten von Zugriffsschlüsseln”.
- Folgen Sie den Schritten unter, the section called “Bevor Sie beginnen” um die digitalen Zertifikate herunterzuladen, sie in TrustStore-Dateien zu konvertieren und den Keystore in den JVM-Argumenten an Ihre Anwendung anzuhängen.

- Fügen Sie den DataStax Java-Treiber für Apache Cassandra zu Ihrem Java-Projekt hinzu. Stellen Sie sicher, dass Sie eine Version des Treibers verwenden, die Apache Cassandra 3.11.2 unterstützt. Weitere Informationen finden Sie in der Dokumentation zum DataStax Java-Treiber für Apache Cassandra.
- Fügen Sie das Authentifizierungs-Plugin zu Ihrer Anwendung hinzu. Das Authentifizierungs-Plugin unterstützt Version 3.x des DataStax Java-Treibers für Apache Cassandra. Wenn Sie Apache Maven oder ein Build-System verwenden, das Maven-Abhängigkeiten verwenden kann, fügen Sie Ihrer Datei die folgenden Abhängigkeiten hinzu. `pom.xml` Ersetzen Sie die Version des Plugins durch die neueste Version, wie im GitHub Repository angezeigt.

```
<dependency>
        <groupId>software.aws.mcs</groupId>
        <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin_3</artifactId>
        <version>3.0.3</version>
</dependency>
```

Schritt 2: Führen Sie die Anwendung aus

Dieses Codebeispiel zeigt eine einfache Befehlszeilenanwendung, die einen Verbindungspool zu Amazon Keyspaces erstellt. Es bestätigt, dass die Verbindung hergestellt wird, indem eine einfache Abfrage ausgeführt wird.

```
package <your package>;
// add the following imports to your project

import software.aws.mcs.auth.SigV4AuthProvider;
import com.datastax.driver.core.Cluster;
import com.datastax.driver.core.ResultSet;
import com.datastax.driver.core.Row;
import com.datastax.driver.core.Session;

public class App
{

    public static void main( String[] args )
    {
        String endPoint = "cassandra.us-east-2.amazonaws.com";
        int portNumber = 9142;
        Session session = Cluster.builder()
                                 .addContactPoint(endPoint)
```

```
                                    .withPort(portNumber)
                                    .withAuthProvider(new SigV4AuthProvider("us-east-2"))

                                    .withSSL()
                                    .build()
                                    .connect();

        ResultSet rs = session.execute("select * from system_schema.keyspaces");
        Row row = rs.one();
        System.out.println(row.getString("keyspace_name"));
    }
}
```

Hinweise zur Verwendung:

Eine Liste der verfügbaren Endpunkte finden Sie unterthe section called “Service-Endpunkte”.

Im folgenden Repository finden Sie hilfreiche Java-Treiberrichtlinien, Beispiele und bewährte Methoden für die Verwendung des Java-Treibers mit Amazon Keyspaces:. https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers

Verwenden eines Cassandra-Python-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

In diesem Abschnitt zeigen wir Ihnen, wie Sie mithilfe eines Python-Client-Treibers eine Verbindung zu Amazon Keyspaces herstellen. Um Benutzern und Anwendungen Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces-Ressourcen zur Verfügung zu stellen, können Sie einen der folgenden Schritte ausführen:

- Erstellen Sie dienstspezifische Anmeldeinformationen, die einem bestimmten Benutzer AWS Identity and Access Management (IAM) zugeordnet sind.
- Zur Erhöhung der Sicherheit empfehlen wir, IAM-Zugriffsschlüssel für IAM-Benutzer oder -Rollen zu erstellen, die für alle Dienste verwendet werden. AWS Das Amazon Keyspaces SigV4-Authentifizierungs-Plugin für Cassandra-Client-Treiber ermöglicht es Ihnen, Anrufe an Amazon Keyspaces mithilfe von IAM-Zugriffsschlüsseln anstelle von Benutzername und Passwort zu authentifizieren. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Themen

- Bevor Sie beginnen
- Stellen Sie mithilfe des Python-Treibers für Apache Cassandra und dienstspezifische Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her
- Stellen Sie mithilfe des DataStax Python-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Bevor Sie beginnen

Sie müssen die folgende Aufgabe abschließen, bevor Sie beginnen können.

Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern. Um mithilfe von TLS eine Verbindung zu Amazon Keyspaces herzustellen, müssen Sie digitale Amazon-Zertifikate herunterladen und den Python-Treiber für die Verwendung von TLS konfigurieren.

Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

> **Note**
>
> Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2-CA verankert waren. AWS migriert alle Zertifikate AWS-Regionen , die unter Amazon Trust

Services (Amazon Root CAs 1—4) ausgestellt wurden. Konfigurieren Sie während dieser Umstellung die Clients so, dass sie sowohl Amazon Root CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

Kombinieren Sie alle heruntergeladenen Zertifikate in einer einzigen `pem` Datei mit dem Namen *`keyspaces-bundle.pem`* in unseren Beispielen. Sie erreichen dies mit dem -Befehl- Notieren Sie sich den Pfad zur Datei, diesen benötigen Sie später.

```
cat AmazonRootCA1.pem \
 AmazonRootCA2.pem \
 AmazonRootCA3.pem \
 AmazonRootCA4.pem \
 sf-class2-root.crt \
 > keyspaces-bundle.pem
```

Stellen Sie mithilfe des Python-Treibers für Apache Cassandra und dienstspezifische Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her

Das folgende Codebeispiel zeigt Ihnen, wie Sie mit einem Python-Client-Treiber und dienstspezifischen Anmeldeinformationen eine Verbindung zu Amazon Keyspaces herstellen.

```
from cassandra.cluster import Cluster
from ssl import SSLContext, PROTOCOL_TLSv1_2 , CERT_REQUIRED
from cassandra.auth import PlainTextAuthProvider

ssl_context = SSLContext(PROTOCOL_TLSv1_2 )

ssl_context.load_verify_locations('path_to_file/keyspaces-bundle.pem')

ssl_context.verify_mode = CERT_REQUIRED
auth_provider = PlainTextAuthProvider(username='ServiceUserName',
 password='ServicePassword')
cluster = Cluster(['cassandra.us-east-2.amazonaws.com'], ssl_context=ssl_context,
 auth_provider=auth_provider, port=9142)
session = cluster.connect()
r = session.execute('select * from system_schema.keyspaces')
print(r.current_rows)
```

Hinweise zur Verwendung:

1. "*path_to_file/keyspaces-bundle.pem*"Ersetzen Sie durch den Pfad zur kombinierten Zertifikatsdatei, die im ersten Schritt gespeichert wurde.
2. Stellen Sie sicher, dass *ServiceUserName* und *ServicePassword* mit dem Benutzernamen und dem Passwort übereinstimmen, das Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben, indem Sie die Schritte unter ausführen. Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen
3. Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Service-Endpunkte”

Stellen Sie mithilfe des DataStax Python-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Der folgende Abschnitt zeigt, wie Sie das SigV4-Authentifizierungs-Plugin für den DataStax Open-Source-Python-Treiber für Apache Cassandra verwenden, um auf Amazon Keyspaces (für Apache Cassandra) zuzugreifen.

Falls Sie dies noch nicht getan haben, beginnen Sie mit der Erstellung von Anmeldeinformationen für Ihre IAM-Rolle, indem Sie den Schritten unter folgen. the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ” In diesem Tutorial werden temporäre Anmeldeinformationen verwendet, wofür eine IAM-Rolle erforderlich ist. Weitere Informationen zu temporären Anmeldeinformationen finden Sie unterthe section called “Temporäre Anmeldeinformationen für die Verbindung mit Amazon Keyspaces erstellen”.

Fügen Sie dann das Python SigV4-Authentifizierungs-Plugin aus dem GitHub Repository zu Ihrer Umgebung hinzu.

```
pip install cassandra-sigv4
```

Das folgende Codebeispiel zeigt, wie Sie mithilfe des DataStax Open-Source-Python-Treibers für Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces herstellen. Das Plugin hängt vom AWS SDK für Python (Boto3) ab. Es wird verwendet`boto3.session`, um temporäre Anmeldeinformationen zu erhalten.

```
from cassandra.cluster import Cluster
from ssl import SSLContext, PROTOCOL_TLSv1_2 , CERT_REQUIRED
from cassandra.auth import PlainTextAuthProvider
import boto3
from cassandra_sigv4.auth import SigV4AuthProvider
```

```
ssl_context = SSLContext(PROTOCOL_TLSv1_2)
ssl_context.load_verify_locations('path_to_file/keyspaces-bundle.pem')
ssl_context.verify_mode = CERT_REQUIRED

# use this if you want to use Boto to set the session parameters.
boto_session = boto3.Session(aws_access_key_id="AKIAIOSFODNN7EXAMPLE",
                             aws_secret_access_key="wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
                             aws_session_token="AQoDYXdzEJr...<remainder of token>",
                             region_name="us-east-2")
auth_provider = SigV4AuthProvider(boto_session)

# Use this instead of the above line if you want to use the Default Credentials and not
 bother with a session.
# auth_provider = SigV4AuthProvider()

cluster = Cluster(['cassandra.us-east-2.amazonaws.com'], ssl_context=ssl_context,
 auth_provider=auth_provider,
                  port=9142)
session = cluster.connect()
r = session.execute('select * from system_schema.keyspaces')
print(r.current_rows)
```

Hinweise zur Verwendung:

1. "*path_to_file/keyspaces-bundle.pem*"Ersetzen Sie durch den Pfad zum Zertifikat, das im ersten Schritt gespeichert wurde.
2. Stellen Sie sicher*aws_access_key_id*, dass die*aws_secret_access_key*, und die *aws_session_token* mit`Access Key`, und übereinstimmen`Secret Access Key`, die `Session Token` Sie mit erhalten haben`boto3.session`. Weitere Informationen finden Sie unter Anmeldeinformationen im AWS SDK für Python (Boto3).
3. Eine Liste der verfügbaren Endpunkte finden Sie unterthe section called “Service-Endpunkte”.

Verwenden eines Node.js Cassandra-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

In diesem Abschnitt erfahren Sie, wie Sie mithilfe eines Client-Treibers eine Node.js Verbindung zu Amazon Keyspaces herstellen. Um Benutzern und Anwendungen Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces-Ressourcen zur Verfügung zu stellen, können Sie einen der folgenden Schritte ausführen:

- Erstellen Sie dienstspezifische Anmeldeinformationen, die einem bestimmten Benutzer AWS Identity and Access Management (IAM) zugeordnet sind.
- Zur Erhöhung der Sicherheit empfehlen wir, IAM-Zugriffsschlüssel für IAM-Benutzer oder -Rollen zu erstellen, die für alle Dienste verwendet werden. AWS Das Amazon Keyspaces SigV4-Authentifizierungs-Plugin für Cassandra-Client-Treiber ermöglicht es Ihnen, Anrufe an Amazon Keyspaces mithilfe von IAM-Zugriffsschlüsseln anstelle von Benutzername und Passwort zu authentifizieren. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Themen

- Bevor Sie beginnen
- Stellen Sie mithilfe des Apache Node.js DataStax Cassandra-Treibers und der dienstspezifischen Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her
- Stellen Sie mithilfe des DataStax Node.js Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Bevor Sie beginnen

Sie müssen die folgende Aufgabe abschließen, bevor Sie beginnen können.

Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern. Um mithilfe von TLS eine Verbindung zu Amazon Keyspaces herzustellen, müssen Sie ein digitales Amazon-Zertifikat herunterladen und den Python-Treiber für die Verwendung von TLS konfigurieren.

Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

> **Note**
>
> Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2-CA verankert waren. AWS migriert alle Zertifikate AWS-Regionen , die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden. Konfigurieren Sie während dieser Umstellung die Clients so, dass sie sowohl Amazon Root CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

Kombinieren Sie alle heruntergeladenen Zertifikate in einer einzigen `pem` Datei mit dem Namen *`keyspaces-bundle.pem`* in unseren Beispielen. Sie erreichen dies mit dem -Befehl- Notieren Sie sich den Pfad zur Datei, diesen benötigen Sie später.

```
cat AmazonRootCA1.pem \
 AmazonRootCA2.pem \
 AmazonRootCA3.pem \
 AmazonRootCA4.pem \
 sf-class2-root.crt \
 > keyspaces-bundle.pem
```

Stellen Sie mithilfe des Apache Node.js DataStax Cassandra-Treibers und der dienstspezifischen Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her

Konfigurieren Sie Ihren Treiber so, dass er die kombinierte Zertifikatsdatei `keyspaces-bundle.pem` für TLS verwendet, und authentifizieren Sie sich mit dienstspezifischen Anmeldeinformationen. Beispiel:

```
const cassandra = require('cassandra-driver');
const fs = require('fs');
const auth = new cassandra.auth.PlainTextAuthProvider('ServiceUserName',
 'ServicePassword');
const sslOptions1 = {
         ca: [
```

```
                    fs.readFileSync('path_to_file/keyspaces-bundle.pem', 'utf-8')],

                    host: 'cassandra.us-west-2.amazonaws.com',
                    rejectUnauthorized: true
        };
const client = new cassandra.Client({
                   contactPoints: ['cassandra.us-west-2.amazonaws.com'],
                   localDataCenter: 'us-west-2',
                   authProvider: auth,
                   sslOptions: sslOptions1,
                   protocolOptions: { port: 9142 }
        });
const query = 'SELECT * FROM system_schema.keyspaces';

client.execute(query)
                    .then( result => console.log('Row from Keyspaces %s',
 result.rows[0]))
                    .catch( e=> console.log(`${e}`));
```

Hinweise zur Verwendung:

1. "*path_to_file/keyspaces-bundle.pem*"Ersetzen Sie durch den Pfad zur kombinierten Zertifikatsdatei, die im ersten Schritt gespeichert wurde.
2. Stellen Sie sicher, dass *ServiceUserName* und *ServicePassword* mit dem Benutzernamen und dem Passwort übereinstimmen, das Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben, indem Sie die Schritte unter ausführen. Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen
3. Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Service-Endpunkte”

Stellen Sie mithilfe des DataStax Node.js Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Der folgende Abschnitt zeigt, wie Sie das SigV4-Authentifizierungs-Plugin für den DataStax Node.js Open-Source-Treiber für Apache Cassandra verwenden, um auf Amazon Keyspaces (für Apache Cassandra) zuzugreifen.

Falls Sie dies noch nicht getan haben, erstellen Sie Anmeldeinformationen für Ihren IAM-Benutzer oder Ihre IAM-Rolle, indem Sie den Schritten unter folgen. the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”

Fügen Sie das Node.js SigV4-Authentifizierungs-Plugin aus dem Repository zu Ihrer Anwendung hinzu. GitHub Das Plugin unterstützt Version 4.x des DataStax Node.js Treibers für Cassandra und hängt vom AWS SDK für ab. Node.js Es wird verwendet, `AWSCredentialsProvider` um Anmeldeinformationen zu erhalten.

```
$ npm install aws-sigv4-auth-cassandra-plugin --save
```

Dieses Codebeispiel zeigt, wie eine Region-specific Instanz von `SigV4AuthProvider` als Authentifizierungsanbieter festgelegt wird.

```
const cassandra = require('cassandra-driver');
const fs = require('fs');
const sigV4 = require('aws-sigv4-auth-cassandra-plugin');

const auth = new sigV4.SigV4AuthProvider({
    region: 'us-west-2',
    accessKeyId:'AKIAIOSFODNN7EXAMPLE',
    secretAccessKey: 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY'});

const sslOptions1 = {
  ca: [
      fs.readFileSync('path_to_file/keyspaces-bundle.pem', 'utf-8')],
  host: 'cassandra.us-west-2.amazonaws.com',
  rejectUnauthorized: true
};


const client = new cassandra.Client({
  contactPoints: ['cassandra.us-west-2.amazonaws.com'],
  localDataCenter: 'us-west-2',
  authProvider: auth,
  sslOptions: sslOptions1,
  protocolOptions: { port: 9142 }
});


const query = 'SELECT * FROM system_schema.keyspaces';

client.execute(query).then(
    result => console.log('Row from Keyspaces %s', result.rows[0]))
    .catch( e=> console.log(`${e}`));
```

Hinweise zur Verwendung:

1. "*path_to_file/keyspaces-bundle.pem*"Ersetzen Sie durch den Pfad zum Zertifikat, das im ersten Schritt gespeichert wurde.
2. Stellen Sie sicher, dass *accessKeyId* und *secretAccessKey* mit dem Zugriffsschlüssel und dem geheimen Zugriffsschlüssel übereinstimmen, den Sie mit diesem Schlüssel erhalten habenAWSCredentialsProvider. Weitere Informationen finden Sie unter Einstellen von Anmeldeinformationen Node.js im AWS SDK für JavaScript in Node.js.
3. Informationen zum Speichern von Zugriffsschlüsseln außerhalb des Codes finden Sie unter Best Practicesthe section called “Verwalten von Zugriffsschlüsseln”.
4. Eine Liste der verfügbaren Endpunkte finden Sie unterthe section called “Service-Endpunkte”.

Verwenden eines Cassandra.NET Core-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

In diesem Abschnitt erfahren Sie, wie Sie mithilfe eines .NET Core-Clienttreibers eine Verbindung zu Amazon Keyspaces herstellen. Die Einrichtungsschritte variieren je nach Ihrer Umgebung und Ihrem Betriebssystem. Möglicherweise müssen Sie sie entsprechend ändern. Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern. Um mithilfe von TLS eine Verbindung zu Amazon Keyspaces herzustellen, konfigurieren Sie Ihren Treiber so, dass er den System Trust Store verwendet, der die Amazon Root CAs 1-4 umfasst.

1. Installieren Sie das CassandraCSharpDriver Through-Nuget mithilfe der Nuget-Konsole.

```
PM> Install-Package CassandraCSharpDriver
```

2. Das folgende Beispiel verwendet ein.NET Core C#-Konsolenprojekt, um eine Verbindung zu Amazon Keyspaces herzustellen und eine Abfrage auszuführen.

```
using Cassandra;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.Security;
using System.Runtime.ConstrainedExecution;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using System.Threading.Tasks;
```

```
namespace CSharpKeyspacesExample
{
    class Program
    {
        public Program(){}

        static void Main(string[] args)
        {
            var userName = "ServiceUserName";
            var pwd = "ServicePassword";
            certCollection.Add(amazoncert);

            var awsEndpoint =  "cassandra.us-east-2.amazonaws.com" ;

            var cluster = Cluster.Builder()
                     .AddContactPoints(awsEndpoint)
                     .WithPort(9142)
                     .WithAuthProvider(new PlainTextAuthProvider(userName, pwd))
                     .WithSSL(new
 SSLOptions().SetCertificateCollection(certCollection))
                     .Build();

            var session = cluster.Connect();
            var rs = session.Execute("SELECT * FROM system_schema.tables;");
            foreach (var row in rs)
            {
                var name = row.GetValue<String>("keyspace_name");
                Console.WriteLine(name);
            }
        }
    }
}
```

Hinweise zur Verwendung:

1. Stellen Sie sicher, dass Sie den Standard-System-Trust Store verwenden, der die Amazon Root-CAs 1—4 umfasst.
2. Stellen Sie sicher, dass *ServiceUserName* und *ServicePassword* mit dem Benutzernamen und dem Passwort übereinstimmen, das Sie bei der Generierung der dienstspezifischen

Anmeldeinformationen erhalten haben, indem Sie die Schritte unter ausführen. Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen

3. Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Service-Endpunkte”

Verwenden eines Cassandra Go-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

In diesem Abschnitt erfahren Sie, wie Sie mithilfe eines Go Cassandra-Client-Treibers eine Verbindung zu Amazon Keyspaces herstellen. Um Benutzern und Anwendungen Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces-Ressourcen zur Verfügung zu stellen, können Sie einen der folgenden Schritte ausführen:

- Erstellen Sie dienstspezifische Anmeldeinformationen, die einem bestimmten Benutzer AWS Identity and Access Management (IAM) zugeordnet sind.
- Zur Erhöhung der Sicherheit empfehlen wir, IAM-Zugriffsschlüssel für IAM-Prinzipale zu erstellen, die für alle Dienste verwendet werden. AWS Das Amazon Keyspaces SigV4-Authentifizierungs-Plugin für Cassandra-Client-Treiber ermöglicht es Ihnen, Anrufe an Amazon Keyspaces mithilfe von IAM-Zugriffsschlüsseln anstelle von Benutzername und Passwort zu authentifizieren. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Themen

- Bevor Sie beginnen
- Stellen Sie mithilfe des Gocql-Treibers für Apache Cassandra und dienstspezifische Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her
- Stellen Sie mithilfe des Go-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Bevor Sie beginnen

Sie müssen die folgende Aufgabe abschließen, bevor Sie beginnen können.

Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern. Um mithilfe von TLS eine Verbindung zu Amazon Keyspaces

herzustellen, müssen Sie ein digitales Amazon-Zertifikat herunterladen und den Go-Treiber für die Verwendung von TLS konfigurieren.

Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

> **Note**
>
> Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2-CA verankert waren. AWS migriert alle Zertifikate AWS-Regionen , die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden. Konfigurieren Sie während dieser Umstellung die Clients so, dass sie sowohl Amazon Root CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

Kombinieren Sie alle heruntergeladenen Zertifikate in einer einzigen `pem` Datei mit dem Namen *`keyspaces-bundle.pem`* in unseren Beispielen. Sie erreichen dies mit dem -Befehl- Notieren Sie sich den Pfad zur Datei, diesen benötigen Sie später.

```
cat AmazonRootCA1.pem \
 AmazonRootCA2.pem \
 AmazonRootCA3.pem \
 AmazonRootCA4.pem \
 sf-class2-root.crt \
```

```
> keyspaces-bundle.pem
```

Stellen Sie mithilfe des Gocql-Treibers für Apache Cassandra und dienstspezifische Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her

1. Erstellen Sie ein Verzeichnis für die Anwendung.

```
mkdir ./gocqlexample
```

2. Navigieren Sie zum neuen Verzeichnis.

```
cd gocqlexample
```

3. Erstellen Sie eine Datei für Ihre Anwendung.

```
touch cqlapp.go
```

4. Laden Sie den Go-Treiber herunter.

```
go get github.com/gocql/gocql
```

5. Fügen Sie der Datei cqlapp.go den folgenden Beispielcode hinzu.

```
package main

import (
     "fmt"
     "github.com/gocql/gocql"
     "log"
)

func main() {

    // add the Amazon Keyspaces service endpoint
    cluster := gocql.NewCluster("cassandra.us-east-2.amazonaws.com")
    cluster.Port=9142
    // add your service specific credentials
    cluster.Authenticator = gocql.PasswordAuthenticator{
            Username: "ServiceUserName",
            Password: "ServicePassword"}

    // provide the path to the keyspaces-bundle.pem
```

```
    cluster.SslOpts = &gocql.SslOptions{
            CaPath: "path_to_file/keyspaces-bundle.pem",
            EnableHostVerification: false,
     }

     // Override default Consistency to LocalQuorum
     cluster.Consistency = gocql.LocalQuorum
     cluster.DisableInitialHostLookup = false

     session, err := cluster.CreateSession()
     if err != nil {
            fmt.Println("err>", err)
     }
     defer session.Close()

     // run a sample query from the system keyspace
     var text string
     iter := session.Query("SELECT keyspace_name FROM system_schema.tables;").Iter()
     for iter.Scan(&text) {
            fmt.Println("keyspace_name:", text)
     }
     if err := iter.Close(); err != nil {
            log.Fatal(err)
     }
     session.Close()
}
```

Hinweise zur Verwendung:

a. "*path_to_file/keyspaces-bundle.pem*"Ersetzen Sie durch den Pfad zur kombinierten Zertifikatsdatei, die im ersten Schritt gespeichert wurde.

b. Stellen Sie sicher, dass *ServiceUserName* und *ServicePassword* mit dem Benutzernamen und dem Passwort übereinstimmen, das Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben, indem Sie die Schritte unter ausführen. Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen

c. Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Service-Endpunkte”

6. Erstellen Sie das Programm.

```
go build cqlapp.go
```

7. Führen Sie das Programm aus.

```
./cqlapp
```

Stellen Sie mithilfe des Go-Treibers für Apache Cassandra und des SigV4-Authentifizierungs-Plug-ins eine Verbindung zu Amazon Keyspaces her

Das folgende Codebeispiel zeigt, wie Sie das SigV4-Authentifizierungs-Plugin für den Open-Source-Go-Treiber verwenden, um auf Amazon Keyspaces (für Apache Cassandra) zuzugreifen.

Falls Sie dies noch nicht getan haben, erstellen Sie Anmeldeinformationen für Ihren IAM-Principal, indem Sie den Schritten unter folgen. the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ” Wenn eine Anwendung auf Lambda oder einer Amazon EC2-Instance ausgeführt wird, verwendet Ihre Anwendung automatisch die Anmeldeinformationen der Instance. Um dieses Tutorial lokal auszuführen, können Sie die Anmeldeinformationen als lokale Umgebungsvariablen speichern.

Fügen Sie Ihrer Anwendung das Go SigV4-Authentifizierungs-Plugin aus dem GitHub Repository hinzu. Das Plugin unterstützt Version 1.2.x des Open-Source-Go-Treibers für Cassandra und hängt vom SDK für Go ab. AWS

```
$ go mod init
$ go get github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin
```

In diesem Codebeispiel wird der Amazon Keyspaces-Endpunkt durch die Klasse dargestellt. `Cluster` Es verwendet die Eigenschaft `AwsAuthenticator` for the authenticator des Clusters, um Anmeldeinformationen abzurufen.

```
package main

import (
        "fmt"
        "github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin/sigv4"
        "github.com/gocql/gocql"
        "log"
)

func main() {
    // configuring the cluster options
```

```
    cluster := gocql.NewCluster("cassandra.us-west-2.amazonaws.com")
    cluster.Port=9142

    // the authenticator uses the default credential chain to find AWS credentials
    cluster.Authenticator = sigv4.NewAwsAuthenticator()

    cluster.SslOpts = &gocql.SslOptions{

            CaPath: "path_to_file/keyspaces-bundle.pem",
            EnableHostVerification: false,
    }
    cluster.Consistency = gocql.LocalQuorum
    cluster.DisableInitialHostLookup = false

    session, err := cluster.CreateSession()
    if err != nil {
     fmt.Println("err>", err)
     return
    }
    defer session.Close()

    // doing the query
    var text string
    iter := session.Query("SELECT keyspace_name FROM system_schema.tables;").Iter()
    for iter.Scan(&text) {
     fmt.Println("keyspace_name:", text)
    }
    if err := iter.Close(); err != nil {
     log.Fatal(err)
    }
}
```

Hinweise zur Verwendung:

1. "*path_to_file/keyspaces-bundle.pem*"Ersetzen Sie durch den Pfad zur Zertifikatsdatei, die im ersten Schritt gespeichert wurde.
2. Damit dieses Beispiel lokal ausgeführt werden kann, müssen Sie die folgenden Variablen als Umgebungsvariablen definieren:
   - `AWS_ACCESS_KEY_ID`
   - `AWS_SECRET_ACCESS_KEY`
   - `AWS_DEFAULT_REGION`

3. Informationen zum Speichern von Zugriffsschlüsseln außerhalb des Codes finden Sie unter bewährte Methodenthe section called “Verwalten von Zugriffsschlüsseln”.
4. Eine Liste der verfügbaren Endpunkte finden Sie unterthe section called “Service-Endpunkte”.

Verwenden eines Cassandra Perl-Client-Treibers für den programmgesteuerten Zugriff auf Amazon Keyspaces

In diesem Abschnitt erfahren Sie, wie Sie mithilfe eines Perl-Client-Treibers eine Verbindung zu Amazon Keyspaces herstellen. Für dieses Codebeispiel haben wir Perl 5 verwendet. Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern.

> **Important**
>
> Amazon Keyspaces-Zertifikate werden in die Amazon Trust Services (ATS) -Hierarchie überführt. Stellen Sie sicher, dass Ihre Umgebung den Amazon Root-CAs 1—4 vertraut, um Verbindungsfehler während dieser Rotation zu vermeiden. Der Perl-Treiber validiert das Amazon-SSL-Zertifikat des Servers nicht, was bedeutet, dass Sie nicht bestätigen können, dass Sie eine Verbindung zu Amazon Keyspaces herstellen. Der zweite Schritt, die Konfiguration des Treibers für die Verwendung von TLS bei der Verbindung zu Amazon Keyspaces, ist weiterhin erforderlich und stellt sicher, dass die zwischen dem Client und dem Server übertragenen Daten verschlüsselt werden.

1. Laden Sie den Cassandra-DBI-Treiber von herunter https://metacpan.org/pod/DBD::Cassandra und installieren Sie den Treiber in Ihrer Perl-Umgebung. Die genauen Schritte hängen von der Umgebung ab. Das Folgende ist ein gängiges Beispiel.

```
cpanm DBD::Cassandra
```

2. Erstellen Sie eine Datei für Ihre Anwendung.

```
touch cqlapp.pl
```

3. Fügen Sie der Datei cqlapp.pl den folgenden Beispielcode hinzu.

```
use DBI;
my $user = "ServiceUserName";
```

```
my $password = "ServicePassword";
my $db = DBI->connect("dbi:Cassandra:host=cassandra.us-
east-2.amazonaws.com;port=9142;tls=1;",
$user, $password);

my $rows = $db->selectall_arrayref("select * from system_schema.keyspaces");
print "Found the following Keyspaces...\n";
for my $row (@$rows) {
      print join(" ",@$row['keyspace_name']),"\n";
}

$db->disconnect;
```

> **Important**
>
> Stellen Sie sicher, dass *ServiceUserName* und *ServicePassword* mit dem Benutzernamen und dem Passwort übereinstimmen, das Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben, indem Sie die Schritte unter ausführen. Dienstspezifische Anmeldeinformationen für den programmatischen Zugriff auf Amazon Keyspaces erstellen

> **Note**
>
> Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Service-Endpunkte”

4. Führen Sie die Anwendung aus.

```
perl cqlapp.pl
```

Kontenübergreifenden Zugriff auf Amazon Keyspaces mit VPC-Endpunkten konfigurieren

Sie können separate Ressourcen erstellen und verwenden, um Ressourcen AWS-Konten zu isolieren und sie in verschiedenen Umgebungen zu verwenden, beispielsweise in der Entwicklungs- und Produktionsumgebung. Dieses Thema führt Sie durch den kontoübergreifenden Zugriff auf Amazon

Keyspaces mithilfe von Schnittstellen-VPC-Endpunkten in einem. Amazon Virtual Private Cloud Weitere Informationen zur Konfiguration des kontoübergreifenden IAM-Zugriffs finden Sie im IAM-Benutzerhandbuch unter Beispielszenario mit separaten Entwicklungs- und Produktionskonten.

Weitere Informationen zu Amazon Keyspaces und privaten VPC-Endpunkten finden Sie unter. the section called “Verwendung von -Schnittstellen-VPC-Endpunkten”

Themen

- Konfigurieren Sie den kontoübergreifenden Zugriff auf Amazon Keyspaces mithilfe von VPC-Endpunkten in einer gemeinsam genutzten VPC
- Konfiguration des kontoübergreifenden Zugriffs auf Amazon Keyspaces ohne gemeinsame VPC

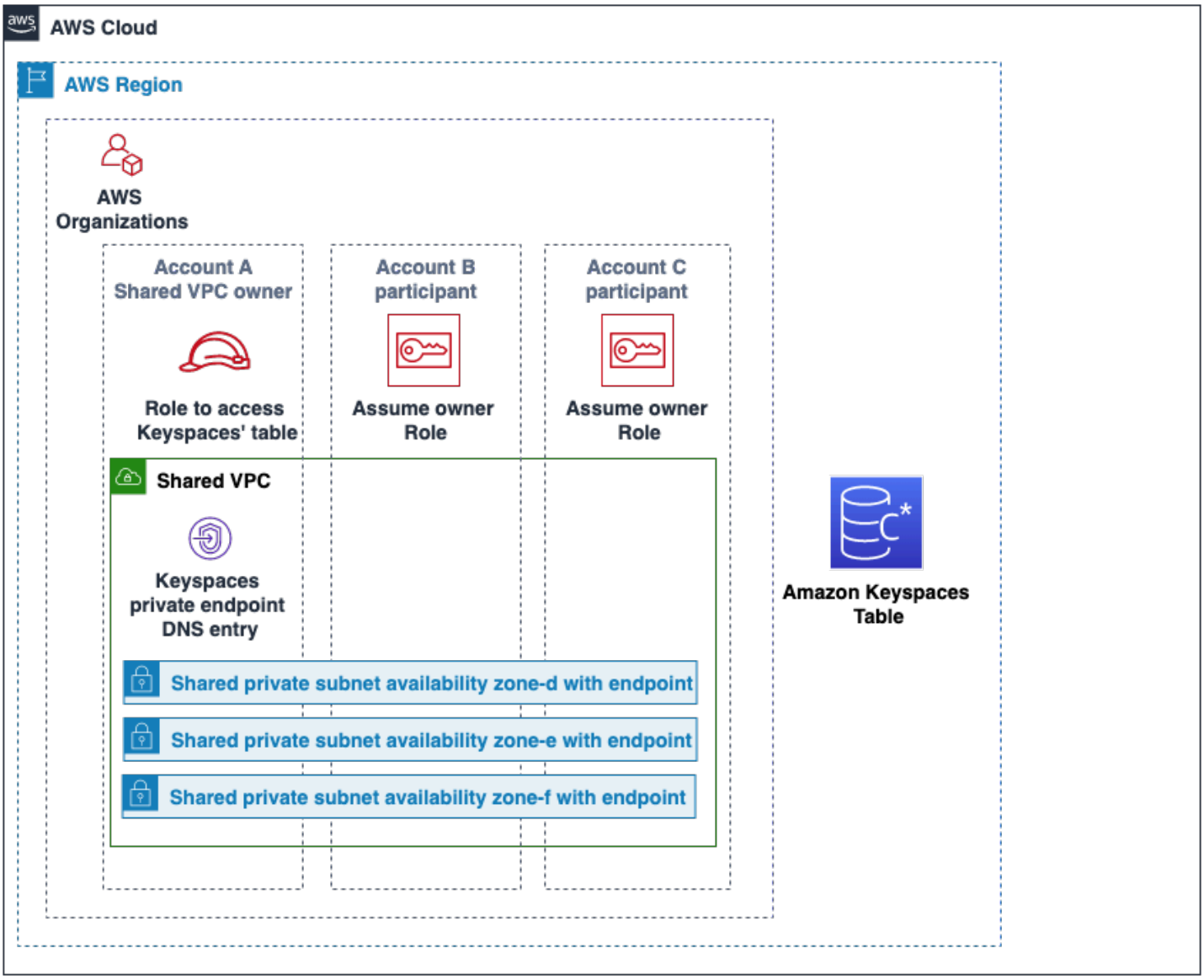
Konfigurieren Sie den kontoübergreifenden Zugriff auf Amazon Keyspaces mithilfe von VPC-Endpunkten in einer gemeinsam genutzten VPC

Sie können verschiedene Ressourcen erstellen AWS-Konten , um sie von Anwendungen zu trennen. Sie können beispielsweise ein Konto für Ihre Amazon Keyspaces-Tabellen, ein anderes Konto für Anwendungen in einer Entwicklungsumgebung und ein weiteres Konto für Anwendungen in einer Produktionsumgebung erstellen. Dieses Thema führt Sie durch die Konfigurationsschritte, die erforderlich sind, um den kontoübergreifenden Zugriff für Amazon Keyspaces mithilfe von VPC-Schnittstellen-Endpunkten in einer gemeinsam genutzten VPC einzurichten.

Ausführliche Schritte zur Konfiguration eines VPC-Endpunkts für Amazon Keyspaces finden Sie unter. the section called “Schritt 3: Erstellen Sie einen VPC-Endpunkt für Amazon Keyspaces”

In diesem Beispiel verwenden wir die folgenden drei Konten in einer gemeinsam genutzten VPC:

- `Account A:111111111111`— Dieses Konto enthält die Infrastruktur, einschließlich der VPC-Endpunkte, der VPC-Subnetze und der Amazon Keyspaces-Tabellen.
- `Account B:222222222222`— Dieses Konto enthält eine Anwendung in einer Entwicklungsumgebung, die eine Verbindung zur Amazon Keyspaces-Tabelle in `Account A:111111111111` herstellen muss.
- `Account C:333333333333`— Dieses Konto enthält eine Anwendung in einer Produktionsumgebung, die eine Verbindung zur Amazon Keyspaces-Tabelle in `Account A:111111111111` herstellen muss.



`Account A:111111111111`ist das Konto, das die Ressourcen (eine Amazon Keyspaces-Tabelle) enthält, auf die zugegriffen werden `Account C:333333333333` muss `Account B:222222222222` und auf die zugegriffen werden muss, `Account A:111111111111` ebenso das vertrauenswürdige Konto. `Account B:222222222222`und `Account C:333333333333` sind die Konten bei den Principals, die Zugriff auf die Ressourcen (eine Amazon Keyspaces-Tabelle) benötigen`Account A:111111111111`, also `Account B:222222222222` `Account C:333333333333` die vertrauenswürdigen Konten. Das vertrauenswürdige Konto erteilt die Berechtigungen für die vertrauenswürdigen Konten, indem es sich eine IAM-Rolle teilt. Das folgende Verfahren beschreibt die Konfigurationsschritte, die in erforderlich sind. `Account A:111111111111`

Konfiguration für **`Konto A:111111111111`**

1. Wird verwendet AWS Resource Access Manager , um eine Ressourcenfreigabe für das Subnetz zu erstellen und das private Subnetz mit `Account B:222222222222` und zu teilen. `Account C:333333333333`

   `Account B:222222222222`und `Account C:333333333333` kann jetzt Ressourcen in dem Subnetz sehen und erstellen, das mit ihnen geteilt wurde.
2. Erstellen Sie einen privaten VPC-Endpunkt von Amazon Keyspaces, der von betrieben wird. AWS PrivateLink Dadurch werden mehrere Endpunkte in gemeinsam genutzten Subnetzen und DNS-Einträgen für den Amazon Keyspaces-Serviceendpunkt erstellt.
3. Erstellen Sie einen Amazon Keyspaces-Schlüsselraum und eine Tabelle.
4. Erstellen Sie eine IAM-Rolle in`Account A:111111111111`, die vollen Zugriff auf die Amazon Keyspaces-Tabelle und Lesezugriff auf die Amazon Keyspaces-Systemtabellen hat und die Amazon EC2 EC2-VPC-Ressourcen beschreiben kann, wie im folgenden Richtlinienbeispiel gezeigt.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "CrossAccountAccess",
               "Effect": "Allow",
               "Action": [
                   "ec2:DescribeNetworkInterfaces",
                   "ec2:DescribeVpcEndpoints",
                   "cassandra:*"
               ],
               "Resource": "*"
           }
       ]
   }
   ```
5. Konfigurieren Sie eine Vertrauensrichtlinie für die IAM-Rolle in, `Account A:111111111111` sodass Sie die Rolle als vertrauenswürdige `Account B:222222222222` Konten übernehmen `Account C:333333333333` können. Dies wird im folgenden Beispiel veranschaulicht.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
   ```

```
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
          "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Weitere Informationen zu kontoübergreifenden IAM-Richtlinien finden Sie unter Cross-account Richtlinien im IAM-Benutzerhandbuch.

Konfiguration in **`Konto B:222222222222`** und **`Konto C:333333333333`**

1. Erstellen Sie in `Account B:222222222222` und `Account C:333333333333` neue Rollen und fügen Sie die folgende Richtlinie hinzu, die es dem Prinzipal ermöglicht, die in erstellte gemeinsame Rolle zu übernehmen`Account A:111111111111`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

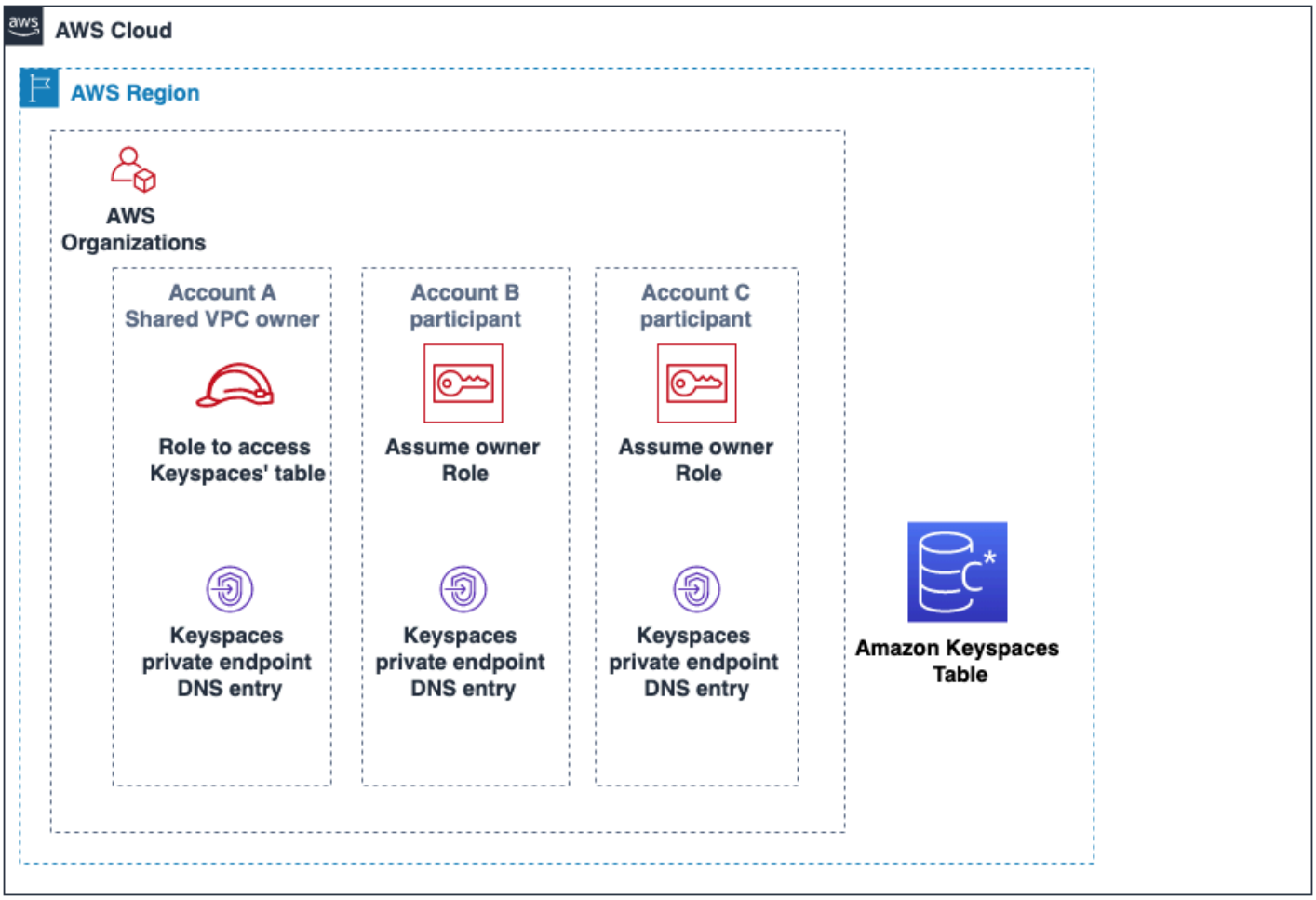
Dem Prinzipal die Übernahme der gemeinsamen Rolle zu ermöglichen, wird mithilfe der `AssumeRole` API von AWS -Security-Token-Service (AWS STS) implementiert. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

2. In `Account B:222222222222` und können Sie Anwendungen erstellen`Account C:333333333333`, die das SIGV4-Authentifizierungs-Plugin verwenden, das es einer Anwendung ermöglicht, die gemeinsame Rolle zu übernehmen, um eine Verbindung zur Amazon Keyspaces-Tabelle herzustellen, die sich `Account A:111111111111` über den VPC-Endpunkt in der gemeinsam genutzten VPC befindet. Weitere Informationen zum SIGV4-Authentifizierungs-Plugin finden Sie unter. the section called “Erstellen Sie programmatische Zugangsdaten” Weitere Informationen darüber, wie Sie eine Anwendung so konfigurieren, dass sie eine Rolle in einem anderen AWS Konto übernimmt, finden Sie unter Authentifizierung und Zugriff im Referenzhandbuch für AWS SDKs und Tools.

Konfiguration des kontoübergreifenden Zugriffs auf Amazon Keyspaces ohne gemeinsame VPC

Wenn die Amazon Keyspaces-Tabelle und der private VPC-Endpunkt unterschiedlichen Konten gehören, sich aber keine VPC teilen, können Anwendungen trotzdem kontoübergreifende Verbindungen über VPC-Endpunkte herstellen. Weil die Konten die VPC-Endpunkte,, `Account A:111111111111`Account `B:222222222222`, nicht gemeinsam nutzen und ihre eigenen VPC-Endpunkte `Account C:333333333333` benötigen. Für den Cassandra-Client-Treiber erscheint Amazon Keyspaces wie ein einzelner Knoten statt wie ein Cluster mit mehreren Knoten. Nach der Verbindung erreicht der Client-Treiber den DNS-Server, der einen der verfügbaren Endpunkte in der VPC des Kontos zurückgibt.

Sie können auch über verschiedene Konten hinweg auf Amazon Keyspaces-Tabellen zugreifen, ohne einen gemeinsamen VPC-Endpunkt zu haben, indem Sie die öffentlichen Endpunkte verwenden oder in jedem Konto einen privaten VPC-Endpunkt bereitstellen. Wenn Sie keine gemeinsam genutzte VPC verwenden, benötigt jedes Konto einen eigenen VPC-Endpunkt. In diesem `Account A:111111111111` Beispiel `Account C:333333333333` benötigen sie ihre eigenen VPC-Endpunkte`Account B:222222222222`, um auf die Tabelle in zuzugreifen. `Account A:111111111111` Bei Verwendung von VPC-Endpunkten in dieser Konfiguration erscheint Amazon Keyspaces für den Cassandra-Client-Treiber als Einzelknoten-Cluster und nicht als Cluster mit mehreren Knoten. Nach der Verbindung erreicht der Client-Treiber den DNS-Server, der einen der verfügbaren Endpunkte in der VPC des Kontos zurückgibt. Der Client-Treiber kann jedoch nicht auf die `system.peers` Tabelle zugreifen, um weitere Endpunkte zu ermitteln. Da weniger Hosts verfügbar sind, stellt der Treiber weniger Verbindungen her. Um dies anzupassen, erhöhen Sie die Verbindungspool-Einstellung des Treibers um den Faktor drei.



`Account A:111111111111`ist das Konto, das die Ressourcen (eine Amazon Keyspaces-Tabelle) enthält, auf die zugegriffen werden `Account C:333333333333` muss `Account B:222222222222` und auf die zugegriffen werden muss, `Account A:111111111111` ebenso das vertrauenswürdige Konto. `Account B:222222222222`und `Account C:333333333333` sind die Konten bei den Principals, die Zugriff auf die Ressourcen (eine Amazon Keyspaces-Tabelle) benötigen`Account A:111111111111`, also `Account B:222222222222` `Account C:333333333333` die vertrauenswürdigen Konten. Das vertrauenswürdige Konto erteilt die Berechtigungen für die vertrauenswürdigen Konten, indem es sich eine IAM-Rolle teilt. Das folgende Verfahren beschreibt die Konfigurationsschritte, die in erforderlich sind. `Account A:111111111111`

Konfiguration für **`Konto A:111111111111`**

1. Erstellen Sie einen Amazon Keyspaces-Schlüsselraum und eine Tabelle in. `Account A:111111111111`

2. Erstellen Sie eine IAM-Rolle in`Account A:111111111111`, die vollen Zugriff auf die Amazon Keyspaces-Tabelle und Lesezugriff auf die Amazon Keyspaces-Systemtabellen hat.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111111111111:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111111111111:/keyspace/system*"
         ]
      }
   ]
}
```

3. Konfigurieren Sie eine Vertrauensrichtlinie für die IAM-Rolle in, `Account A:111111111111` sodass Prinzipale die Rolle als `Account B:222222222222` vertrauenswürdige `Account C:333333333333` Konten übernehmen können. Dies wird im folgenden Beispiel veranschaulicht.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
          "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Weitere Informationen zu kontoübergreifenden IAM-Richtlinien finden Sie unter Cross-account Richtlinien im IAM-Benutzerhandbuch.

4. Konfigurieren Sie den VPC-Endpunkt in `Account A:111111111111` und fügen Sie dem Endpunkt Berechtigungen hinzu, die es den Rollen von `Account B:222222222222` und `Account C:333333333333` die Übernahme der Rolle bei der `Account A` Verwendung des VPC-Endpunkts ermöglichen. Diese Berechtigungen gelten für den VPC-Endpunkt, an den sie angehängt sind. Weitere Informationen zu VPC-Endpunktrichtlinien finden Sie unterthe section called “Steuerung des Zugriffs auf VPC-Schnittstellen-Endpunkte für Amazon Keyspaces”.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessfromSpecificIAMroles",
      "Effect": "Allow",
      "Action": "cassandra:*",
      "Resource": "*",
      "Principal": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
            "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
          ]
        }
      }
    }
  ]
}
```

Konfiguration in **`Konto B:222222222222`** und **`Konto C:333333333333`**

1. Erstellen Sie in `Account B:222222222222` und `Account C:333333333333` neue Rollen und fügen Sie die folgende Richtlinie hinzu, die es dem Prinzipal ermöglicht, die in erstellte gemeinsame Rolle zu übernehmen`Account A:111111111111`.

```
{
    "Version": "2012-10-17",
    "Statement": {
```

```
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::111111111111:role/keyspaces_access"
        }
}
```

Dem Prinzipal die Übernahme der gemeinsamen Rolle zu ermöglichen, wird mithilfe der `AssumeRole` API von AWS -Security-Token-Service (AWS STS) implementiert. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

2. In `Account B:222222222222` und können Sie Anwendungen erstellen`Account C:333333333333`, die das SIGV4-Authentifizierungs-Plugin verwenden, das es einer Anwendung ermöglicht, die gemeinsame Rolle für die Verbindung mit der Amazon Keyspaces-Tabelle zu übernehmen, die sich in befindet. `Account A:111111111111` Weitere Informationen zum SIGV4-Authentifizierungs-Plugin finden Sie unter. the section called “Erstellen Sie programmatische Zugangsdaten” Weitere Informationen darüber, wie Sie eine Anwendung so konfigurieren, dass sie eine Rolle in einem anderen AWS Konto übernimmt, finden Sie unter Authentifizierung und Zugriff im Referenzhandbuch für AWS SDKs und Tools.

Erste Schritte mit Amazon Keyspaces (für Apache Cassandra)

Wenn Sie Apache Cassandra und Amazon Keyspaces noch nicht kennen, führt Sie dieses Tutorial durch die Installation der erforderlichen Programme und Tools, um Amazon Keyspaces erfolgreich zu verwenden. Sie lernen, wie Sie mithilfe der Cassandra Query Language (CQL), der oder der () einen Schlüsselraum und eine Tabelle erstellen. AWS-Managementkonsole AWS Command Line Interface AWS CLI Anschließend verwenden Sie Cassandra Query Language (CQL), um CRUD-Operationen (Create, Lese, Update, Delete) für Daten in Ihrer Amazon Keyspaces-Tabelle durchzuführen.

Dieses Tutorial behandelt die folgenden Schritte.

- Voraussetzungen — Bevor Sie mit dem Tutorial beginnen, folgen Sie den Anweisungen zur AWS Einrichtung, um sich für einen IAM-Benutzer mit Zugriff auf Amazon Keyspaces zu registrieren AWS und diesen zu erstellen. Dann richten Sie das und ein. `cqhsh-expansion` AWS CloudShell Alternativ können Sie das verwenden AWS CLI , um Ressourcen in Amazon Keyspaces zu erstellen.
- Schritt 1: Erstellen Sie einen Schlüsselraum und eine Tabelle — In diesem Abschnitt erstellen Sie einen Schlüsselraum mit dem Namen „catalog“ und eine Tabelle mit dem Namen „book_awards“ darin. Sie geben die Spalten, Datentypen, den Partitionsschlüssel und die Clustering-Spalte der Tabelle an, indem Sie die AWS-Managementkonsole, CQL oder die verwenden. AWS CLI
- Schritt 2: CRUD-Operationen ausführen — Hier verwenden Sie die PIN, CloudShell um Daten `cqlsh-expansion` in der Tabelle „book_awards“ einzufügen, zu lesen, zu aktualisieren und zu löschen. Sie lernen, verschiedene CQL-Anweisungen wie SELECT, INSERT, UPDATE und DELETE zu verwenden, und üben das Filtern und Ändern von Daten.
- Schritt 3: Ressourcen bereinigen — Um zu vermeiden, dass Gebühren für ungenutzte Ressourcen anfallen, führt Sie dieser Abschnitt durch das Löschen der Tabelle „book_awards“ und des Schlüsselraums „catalog“ mithilfe der Konsole, CQL oder der. AWS CLI

Tutorials zur programmgesteuerten Verbindung zu Amazon Keyspaces mithilfe verschiedener Apache Cassandra-Client-Treiber finden Sie unter. the section called “Verwenden eines Cassandra-Client-Treibers” Codebeispiele mit unterschiedlichen AWS SDKs finden Sie unter Codebeispiele für Amazon Keyspaces mit SDKs. AWS

Themen

- Voraussetzungen und Überlegungen zum Tutorial
- Erstellen Sie einen Schlüsselraum in Amazon Keyspaces
- Überprüfen Sie den Status der Keyspace-Erstellung in Amazon Keyspaces
- Erstellen Sie eine Tabelle in Amazon Keyspaces
- Überprüfen Sie den Status der Tabellenerstellung in Amazon Keyspaces
- Erstellen, Lesen, Aktualisieren und Löschen von Daten (CRUD) mithilfe von CQL in Amazon Keyspaces
- Löschen Sie eine Tabelle in Amazon Keyspaces
- Löschen Sie einen Schlüsselraum in Amazon Keyspaces

Voraussetzungen und Überlegungen zum Tutorial

Bevor Sie mit Amazon Keyspaces beginnen können, folgen Sie den Anweisungen zur AWS Einrichtung unter. Zugreifen auf Amazon Keyspaces (für Apache Cassandra) Zu diesen Schritten gehören die Registrierung AWS und Erstellung eines AWS Identity and Access Management (IAM-) Benutzers mit Zugriff auf Amazon Keyspaces.

Um alle Schritte des Tutorials abzuschließen, müssen Sie es installieren. `cqlsh` Sie können den Anweisungen zur Einrichtung unter folgenVerwenden`cqlsh`, um eine Verbindung zu Amazon Keyspaces herzustellen.

Um mit `cqlsh` oder auf Amazon Keyspaces zuzugreifen AWS CLI, empfehlen wir die Verwendung von AWS CloudShell. CloudShellist eine browserbasierte, vorauthentifizierte Shell, die Sie direkt von der aus starten können. AWS-Managementkonsole Sie können AWS Command Line Interface (AWS CLI) -Befehle für Amazon Keyspaces ausführen, indem Sie Ihre bevorzugte Shell (Bash oder Z-Shell) verwenden. PowerShell Zur Verwendung `cqlsh` müssen Sie das installieren. `cqlsh-expansion` Anweisungen zur `cqlsh-expansion` Installation finden Sie unterthe section called “Verwendung der `cqlsh-expansion`”. Weitere Informationen zu CloudShell siehethe section called “Benutzen AWS CloudShell”.

Um Ressourcen in Amazon Keyspaces AWS CLI zu erstellen, anzuzeigen und zu löschen, folgen Sie den Anweisungen zur Einrichtung unterthe section called “Herunterladen und Konfigurieren des AWS CLI”.

Nachdem Sie die erforderlichen Schritte abgeschlossen haben, fahren Sie mit fortErstellen Sie einen Schlüsselraum in Amazon Keyspaces.

Erstellen Sie einen Schlüsselraum in Amazon Keyspaces

In diesem Abschnitt erstellen Sie einen Schlüsselraum mithilfe der Konsole oder der. `cqlsh` AWS CLI

> **Note**
>
> Bevor Sie beginnen, stellen Sie sicher, dass Sie alle Voraussetzungen für das Tutorial konfiguriert haben.

Ein Schlüsselraum gruppiert verwandte Tabellen, die für eine oder mehrere Anwendungen relevant sind. Ein Schlüsselraum enthält eine oder mehrere Tabellen und definiert die Replikationsstrategie für alle darin enthaltenen Tabellen. Weitere Informationen zu Schlüsselräumen finden Sie in den folgenden Themen:

- DDL-Anweisungen (Data Definition Language) in der CQL-Sprachreferenz: Keyspaces
- Kontingente für Amazon Keyspaces (für Apache Cassandra)

In diesem Tutorial erstellen wir einen Schlüsselraum mit einer einzelnen Region, und die Replikationsstrategie für den Schlüsselraum lautet: `SingleRegionStrategy` Mithilfe von `SingleRegionStrategy` Amazon Keyspaces repliziert Amazon Keyspaces Daten über drei Availability Zones in einer. AWS-Region Informationen zum Erstellen von Schlüsselräumen für mehrere Regionen finden Sie unter. the section called “Erstellen Sie einen Schlüsselraum für mehrere Regionen”

Verwenden der Konsole

So erstellen Sie einen Schlüsselraum mit der Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter. https://console.aws.amazon.com/keyspaces/home
2. Wählen Sie im Navigationsbereich Keyspaces aus.
3. Wählen Sie Schlüsselraum erstellen.
4. Geben Sie **`catalog`** im Feld Schlüsselraumname den Namen für Ihren Schlüsselraum ein.

   Einschränkungen bei der Benennung:

- Der Name darf nicht leer sein.
- Zulässige Zeichen: alphanumerische Zeichen und Unterstrich ()_.
- Die maximale Länge beträgt 48 Zeichen.

5. Bestätigen Sie unter AWS-Regionen, dass Single-Region Replikation die Replikationsstrategie für den Schlüsselraum ist.
6. Um den Schlüsselraum zu erstellen, wählen Sie Schlüsselraum erstellen.
7. Stellen Sie sicher, dass der Schlüsselraum erstellt `catalog` wurde, indem Sie wie folgt vorgehen:

   a. Wählen Sie im Navigationsbereich Keyspaces aus.
   b. Suchen Sie Ihren Schlüsselraum `catalog` in der Liste der Schlüsselräume.

Verwenden von CQL

Das folgende Verfahren erstellt einen Schlüsselraum mit CQL.

Um einen Schlüsselraum mit CQL zu erstellen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *`us-east-1`* mit Ihrer eigenen Region aktualisieren.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

   Die Ausgabe dieses Befehls sollte so aussehen.

   ```
   Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
   [cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
   Use HELP for help.
   cqlsh current consistency level is ONE.
   ```

2. Erstellen Sie Ihren Schlüsselraum mit dem folgenden CQL-Befehl.

   ```
   CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
   ```

   `SingleRegionStrategy`verwendet einen Replikationsfaktor von drei und repliziert Daten über drei AWS Availability Zones in seiner Region.

> **Note**
>
> Amazon Keyspaces verwendet standardmäßig alle Eingaben in Kleinbuchstaben, sofern Sie sie nicht in Anführungszeichen setzen.

3. Vergewissern Sie sich, dass Ihr Keyspace erstellt wurde.

```
SELECT * from system_schema.keyspaces;
```

Die Ausgabe dieses Befehls sollte in etwa so aussehen.

```
cqlsh> SELECT * from system_schema.keyspaces;

 keyspace_name           | durable_writes | replication
-------------------------+----------------
+-------------------------------------------------------------------------------------
           system_schema |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
      system_schema_mcs |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
                 system |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
 system_multiregion_info |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
                catalog |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(5 rows)
```

Verwendung der AWS CLI

Das folgende Verfahren erstellt einen Schlüsselraum mit dem AWS CLI.

Um einen Schlüsselraum zu erstellen, verwenden Sie den AWS CLI

1. Um zu bestätigen, dass Ihre Umgebung eingerichtet ist, können Sie den folgenden Befehl in CloudShell ausführen.

```
aws keyspaces help
```

2. Erstellen Sie Ihren Schlüsselraum mit der folgenden AWS CLI Anweisung.

```
aws keyspaces create-keyspace --keyspace-name 'catalog'
```

3. Vergewissern Sie sich, dass Ihr Schlüsselraum mit der folgenden Anweisung erstellt wurde AWS CLI

```
aws keyspaces get-keyspace --keyspace-name 'catalog'
```

Die Ausgabe dieses Befehls sollte ähnlich wie in diesem Beispiel aussehen.

```
{
    "keyspaceName": "catalog",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/",
    "replicationStrategy": "SINGLE_REGION"
}
```

Überprüfen Sie den Status der Keyspace-Erstellung in Amazon Keyspaces

Amazon Keyspaces führt DDL-Operationen (Data Definition Language), wie das Erstellen und Löschen von Schlüsselräumen, asynchron durch.

Sie können den Erstellungsstatus neuer Schlüsselräume in der überwachen. Dies gibt an AWS-Managementkonsole, wann ein Schlüsselraum aussteht oder aktiv ist. Sie können den Erstellungsstatus eines neuen Schlüsselraums auch programmgesteuert überwachen, indem Sie den Schlüsselraum verwenden. `system_schema_mcs` Ein Schlüsselraum wird in der `system_schema_mcs` `keyspaces` Tabelle sichtbar, wenn er einsatzbereit ist.

Das empfohlene Entwurfsmuster, um zu überprüfen, wann ein neuer Schlüsselraum einsatzbereit ist, besteht darin, die Amazon `system_schema_mcs` `keyspaces` Keyspaces-Tabelle (system_schema_mcs.*) abzufragen. Eine Liste der DDL-Anweisungen für Schlüsselräume finden Sie im Abschnitt in der CQL-Sprachreferenz. the section called “Keyspaces”

Die folgende Abfrage zeigt, ob ein Schlüsselraum erfolgreich erstellt wurde.

```
SELECT * FROM system_schema_mcs.keyspaces WHERE keyspace_name = 'mykeyspace';
```

Für einen Schlüsselraum, der erfolgreich erstellt wurde, sieht die Ausgabe der Abfrage wie folgt aus.

```
keyspace_name | durable_writes  | replication
--------------+-----------------+--------------
   mykeyspace | true            |{...} 1 item
```

Erstellen Sie eine Tabelle in Amazon Keyspaces

In diesem Abschnitt erstellen Sie eine Tabelle mithilfe der Konsole oder der. `cqlsh` AWS CLI

In einer Tabelle werden Ihre Daten organisiert und gespeichert. Der Primärschlüssel Ihrer Tabelle bestimmt, wie Daten in Ihrer Tabelle partitioniert werden. Der Primärschlüssel besteht aus einem erforderlichen Partitionsschlüssel und einer oder mehreren optionalen Clusterspalten. Die kombinierten Werte, aus denen der Primärschlüssel besteht, müssen für alle Daten der Tabelle eindeutig sein. Weitere Informationen zu Tabellen finden Sie in den folgenden Themen:

- Entwurf des Partitionsschlüssels: the section called “Entwerfen von Partitionsschlüsseln”
- Arbeiten mit Tabellen: the section called “Überprüfen Sie den Status der Tabellenerstellung”
- DDL-Anweisungen in der CQL-Sprachreferenz: Tabellen
- Verwaltung der Tabellen-Ressourcen: Verwaltung serverloser Ressourcen
- Überwachung der Auslastung der Tabellenressourcen: the section called “Überwachung mit CloudWatch”
- Kontingente für Amazon Keyspaces (für Apache Cassandra)

Wenn Sie eine Tabelle erstellen, geben Sie Folgendes an:

- Der Name der Tabelle.
- Der Name und der Datentyp jeder Spalte in der Tabelle.
- Der Primärschlüssel für die Tabelle.
  - Partitionsschlüssel — Erforderlich
  - Gruppierung von Spalten — optional

Gehen Sie wie folgt vor, um eine Tabelle mit den angegebenen Spalten, Datentypen, Partitionsschlüsseln und Clusterspalten zu erstellen.

Verwenden der Konsole

Das folgende Verfahren erstellt die Tabelle `book_awards` mit diesen Spalten und Datentypen.

```
year           int
award          text
rank           int
category       text
book_title     text
author         text
publisher      text
```

Um eine Tabelle mit der Konsole zu erstellen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces aus.
3. Wählen Sie `catalog` als Schlüsselraum, in dem Sie diese Tabelle erstellen möchten.
4. Wählen Sie Create table (Tabelle erstellen) aus.
5. Geben Sie **`book_awards`** im Feld Tabellenname einen Namen für Ihre Tabelle ein.

   Einschränkungen bei der Benennung:

   - Der Name darf nicht leer sein.
   - Zulässige Zeichen: alphanumerische Zeichen und Unterstrich ()_.
   - Die maximale Länge beträgt 48 Zeichen.
6. Wiederholen Sie im Abschnitt Spalten die folgenden Schritte für jede Spalte, die Sie dieser Tabelle hinzufügen möchten.

   Fügen Sie die folgenden Spalten und Datentypen hinzu.

   ```
   year           int
   award          text
   rank           int
   category       text
   book_title     text
   ```

```
author         text
publisher      text
```

a. Name — Geben Sie einen Namen für die Spalte ein.

   Einschränkungen bei der Benennung:

   - Der Name darf nicht leer sein.
   - Zulässige Zeichen: alphanumerische Zeichen und Unterstrich ()_.
   - Die maximale Länge beträgt 48 Zeichen.

b. Typ — Wählen Sie in der Liste der Datentypen den Datentyp für diese Spalte aus.

c. Um eine weitere Spalte hinzuzufügen, wählen Sie Spalte hinzufügen.

7. Wählen Sie `award` und `year` als Partitionsschlüssel unter Partitionsschlüssel. Für jede Tabelle ist ein Partitionsschlüssel erforderlich. Ein Partitionsschlüssel kann aus einer oder mehreren Spalten bestehen.
8. Fügen Sie `category` und `rank` als Clustering-Spalten hinzu. Cluster-Spalten sind optional und bestimmen die Sortierreihenfolge innerhalb jeder Partition.

   a. Um eine Clustering-Spalte hinzuzufügen, wählen Sie Cluster-Spalte hinzufügen.

   b. Wählen Sie in der Spaltenliste eine Kategorie aus. Wählen Sie in der Liste Reihenfolge die Option ASC aus, um die Werte in dieser Spalte in aufsteigender Reihenfolge zu sortieren. (Wählen Sie DESC für absteigende Reihenfolge.)

   c. Wählen Sie dann Clusterspalte hinzufügen und dann Rang aus.

9. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Standardeinstellungen aus.
10. Wählen Sie Create table (Tabelle erstellen) aus.
11. Vergewissern Sie sich, dass Ihre Tabelle erstellt wurde.

    a. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.

    b. Vergewissern Sie sich, dass Ihre Tabelle in der Tabellenliste enthalten ist.

    c. Wählen Sie den Namen Ihrer Tabelle.

    d. Vergewissern Sie sich, dass alle Ihre Spalten und Datentypen korrekt sind.

> **Note**
>
> Die Spalten sind möglicherweise nicht in der Reihenfolge aufgeführt, in der Sie sie der Tabelle hinzugefügt haben.

Verwenden von CQL

Dieses Verfahren erstellt mithilfe von CQL eine Tabelle mit den folgenden Spalten und Datentypen. Die `award` Spalten `year` und sind Partitionsschlüssel mit `category` und `rank` als Cluster-Spalten. Zusammen bilden sie den Primärschlüssel der Tabelle.

```
year           int
award          text
rank           int
category       text
book_title     text
author         text
publisher      text
```

Um eine Tabelle mit CQL zu erstellen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *`us-east-1`* mit Ihrer eigenen Region aktualisieren.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

   Die Ausgabe dieses Befehls sollte so aussehen.

   ```
   Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
   [cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
   Use HELP for help.
   cqlsh current consistency level is ONE.
   ```

2. Erstellen Sie an der Keyspace-Eingabeaufforderung (`cqlsh:keyspace_name>`) Ihre Tabelle, indem Sie den folgenden Code in Ihr Befehlsfenster eingeben.

   ```
   CREATE TABLE catalog.book_awards (
   ```

```
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
   PRIMARY KEY ((year, award), category, rank)
   );
```

> **Note**
>
> ASCist die standardmäßige Clusterreihenfolge. Sie können auch eine DESC absteigende Reihenfolge angeben.

Beachten Sie, dass es sich bei den Spalten `year` und um Partitionsschlüsselspalten `award` handelt. Dann `category` und `rank` sind die Cluster-Spalten in aufsteigender Reihenfolge () `ASC` sortiert. Zusammen bilden diese Spalten den Primärschlüssel der Tabelle.

3. Stellen Sie sicher, dass Ihre Tabelle erstellt wurde.

```
SELECT * from system_schema.tables WHERE keyspace_name='catalog.book_awards' ;
```

Die Ausgabe sollte in etwa so aussehen.

```
 keyspace_name | table_name | bloom_filter_fp_chance | caching | cdc | comment
 | compaction | compression | crc_check_chance | dclocal_read_repair_chance
 | default_time_to_live | extensions | flags | gc_grace_seconds | id |
 max_index_interval | memtable_flush_period_in_ms | min_index_interval |
 read_repair_chance | speculative_retry
---------------+------------+-------------------------+---------+-----+---------
+------------+-------------+------------------+--------------------------
+----------------------+------------+-------+------------------+----
+--------------------+-----------------------------+-------------------
+--------------------+------------------

(0 rows)
```

4. Überprüfen Sie die Struktur Ihrer Tabelle.

```
SELECT * FROM system_schema.columns WHERE keyspace_name = 'catalog' AND table_name
 = 'book_awards';
```

Die Ausgabe dieser Anweisung sollte ähnlich wie in diesem Beispiel aussehen.

```
 keyspace_name | table_name  | column_name | clustering_order | column_name_bytes
    | kind          | position | type
---------------+-------------+-------------+------------------
+-----------------------+---------------+----------+------
       catalog | book_awards |        year |             none |
 0x79656172 | partition_key |        0 |  int
       catalog | book_awards |       award |             none |
 0x6177617264 | partition_key |        1 | text
       catalog | book_awards |    category |              asc |
 0x63617465676f7279 |    clustering |        0 | text
       catalog | book_awards |        rank |              asc |
 0x72616e6b |    clustering |        1 |  int
       catalog | book_awards |      author |             none |
 0x617574686f72 |       regular |       -1 | text
       catalog | book_awards |  book_title |             none |
 0x626f6f6b5f7469746c65 |       regular |       -1 | text
       catalog | book_awards |   publisher |             none |
 0x7075626c6973686572 |       regular |       -1 | text

(7 rows)
```

Vergewissern Sie sich, dass alle Spalten und Datentypen Ihren Erwartungen entsprechen. Die Reihenfolge der Spalten kann sich von der Reihenfolge in der `CREATE` Anweisung unterscheiden.

Verwendung der AWS CLI

Dieses Verfahren erstellt eine Tabelle mit den folgenden Spalten und Datentypen unter Verwendung von AWS CLI. Die `award` Spalten `year` und bilden den Partitionsschlüssel mit `category` und `rank` als Clusterspalten.

```
year           int
award          text
rank           int
category       text
```

```
book_title     text
author         text
publisher      text
```

Um eine Tabelle zu erstellen, verwenden Sie den AWS CLI

Der folgende Befehl erstellt eine Tabelle mit dem Namen book_awards. Der Partitionsschlüssel der Tabelle besteht aus den Spalten `year` `award` und der Gruppierungsschlüssel besteht aus den Spalten. Beide Cluster-Spalten `category` verwenden die aufsteigende Sortierreihenfolge. `rank` (Der besseren Lesbarkeit halber ist `schema-definition` der Befehl table create in diesem Abschnitt in separate Zeilen unterteilt.)

1. Sie können die Tabelle mit der folgenden Anweisung erstellen.

   ```
   aws keyspaces create-table --keyspace-name 'catalog' \
                        --table-name 'book_awards' \
                        --schema-definition 'allColumns=[{name=year,type=int},
   {name=award,type=text},{name=rank,type=int},
               {name=category,type=text}, {name=author,type=text},
   {name=book_title,type=text},{name=publisher,type=text}],
               partitionKeys=[{name=year},
   {name=award}],clusteringKeys=[{name=category,orderBy=ASC},{name=rank,orderBy=ASC}]'
   ```

   Dieser Befehl führt zu der folgenden Ausgabe.

   ```
   {
       "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/
   table/book_awards"
   }
   ```

2. Um die Metadaten und Eigenschaften der Tabelle zu bestätigen, können Sie den folgenden Befehl verwenden.

   ```
   aws keyspaces get-table --keyspace-name 'catalog' --table-name 'book_awards'
   ```

   Dieser Befehl gibt die folgende Ausgabe zurück.

   ```
   {
       "keyspaceName": "catalog",
       "tableName": "book_awards",
   ```

```
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards",
    "creationTimestamp": "2024-07-11T15:12:55.571000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "year",
                "type": "int"
            },
            {
                "name": "award",
                "type": "text"
            },
            {
                "name": "category",
                "type": "text"
            },
            {
                "name": "rank",
                "type": "int"
            },
            {
                "name": "author",
                "type": "text"
            },
            {
                "name": "book_title",
                "type": "text"
            },
            {
                "name": "publisher",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "year"
            },
            {
                "name": "award"
            }
        ],
        "clusteringKeys": [
```

```
            {
                "name": "category",
                "orderBy": "ASC"
            },
            {
                "name": "rank",
                "orderBy": "ASC"
            }
        ],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-07-11T15:12:55.571000+00:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    },
    "replicaSpecifications": []
}
```

Um CRUD-Operationen (Erstellen, Lesen, Aktualisieren und Löschen) für die Daten in Ihrer Tabelle auszuführen, fahren Sie mit fortthe section called “CRUD-Operationen”.

Überprüfen Sie den Status der Tabellenerstellung in Amazon Keyspaces

Amazon Keyspaces führt DDL-Operationen (Data Definition Language), wie das Erstellen und Löschen von Tabellen, asynchron durch. Sie können den Erstellungsstatus neuer Tabellen in der überwachen. Dies gibt an AWS-Managementkonsole, wann eine Tabelle ausstehend oder aktiv ist. Sie können den Erstellungsstatus einer neuen Tabelle auch programmgesteuert überwachen, indem Sie die Systemschematabelle verwenden.

Eine Tabelle wird im Systemschema als aktiv angezeigt, wenn sie einsatzbereit ist. Das empfohlene Entwurfsmuster, um zu überprüfen, wann eine neue Tabelle einsatzbereit ist, besteht darin, die Amazon Keyspaces-Systemschematabellen abzufragen (`system_schema_mcs.*`). Eine Liste der DDL-Anweisungen für Tabellen finden Sie im the section called “Tabellen” Abschnitt in der CQL-Sprachreferenz.

Die folgende Abfrage zeigt den Status einer Tabelle.

```
SELECT keyspace_name, table_name, status FROM system_schema_mcs.tables WHERE
 keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Für eine Tabelle, die noch erstellt wird und noch aussteht, sieht die Ausgabe der Abfrage wie folgt aus.

```

keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | CREATING
```

Für eine Tabelle, die erfolgreich erstellt wurde und aktiv ist, sieht die Ausgabe der Abfrage wie folgt aus.

```

keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | ACTIVE
```

Erstellen, Lesen, Aktualisieren und Löschen von Daten (CRUD) mithilfe von CQL in Amazon Keyspaces

In diesem Schritt des Tutorials erfahren Sie, wie Sie mithilfe von CQL-Anweisungen (DML) Daten in eine Amazon Keyspaces-Tabelle einfügen, lesen, aktualisieren und löschen. In Amazon Keyspaces können Sie nur DML-Anweisungen in der CQL-Sprache erstellen. In diesem Tutorial üben Sie die Ausführung von DML-Anweisungen mithilfe des mit im`cqlsh-expansion`. AWS CloudShell AWS-Managementkonsole

- Daten einfügen — In diesem Abschnitt wird das Einfügen einzelner und mehrerer Datensätze in eine Tabelle mithilfe der Anweisung behandelt. `INSERT` Sie erfahren, wie Sie Daten aus einer CSV-Datei hochladen und erfolgreiche Einfügungen mithilfe `SELECT` von Abfragen überprüfen.
- Daten lesen — Hier lernen Sie verschiedene Varianten der `SELECT` Anweisung kennen, um Daten aus einer Tabelle abzurufen. Zu den Themen gehören das Auswählen aller Daten, das Auswählen bestimmter Spalten, das Filtern von Zeilen anhand von Bedingungen mithilfe der `WHERE` Klausel und das Verständnis einfacher und zusammengesetzter Bedingungen.
- Daten aktualisieren — In diesem Abschnitt erfahren Sie, wie Sie vorhandene Daten in einer Tabelle mithilfe der `UPDATE` Anweisung ändern. Sie üben das Aktualisieren einzelner und mehrerer Spalten und lernen dabei die Einschränkungen beim Aktualisieren von Primärschlüsselspalten kennen.
- Löschen von Daten — Der letzte Abschnitt behandelt das Löschen von Daten aus einer Tabelle mithilfe der `DELETE` Anweisung. Sie erfahren, wie Sie bestimmte Zellen und ganze Zeilen löschen und welche Auswirkungen das Löschen von Daten im Vergleich zum Löschen der gesamten Tabelle oder des Schlüsselraums hat.

Im Laufe des Tutorials finden Sie Beispiele, Tipps und Möglichkeiten, das Schreiben eigener CQL-Abfragen für verschiedene Szenarien zu üben.

Themen

- Daten in eine Amazon Keyspaces-Tabelle einfügen und laden
- Lesen Sie Daten aus einer Tabelle mithilfe der CQL SELECT-Anweisung in Amazon Keyspaces
- Aktualisieren Sie Daten in einer Amazon Keyspaces-Tabelle mithilfe von CQL
- Löschen Sie Daten aus einer Tabelle mit der CQL DELETE-Anweisung

Daten in eine Amazon Keyspaces-Tabelle einfügen und laden

Um Daten in Ihrer `book_awards` Tabelle zu erstellen, verwenden Sie die `INSERT` Anweisung, um eine einzelne Zeile hinzuzufügen.

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *`us-east-1`* mit Ihrer eigenen Region aktualisieren.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

Die Ausgabe dieses Befehls sollte so aussehen.

```
Connected to Amazon Keyspaces at cassandra.us-east-1.amazonaws.com:9142
[cqlsh 6.1.0 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh current consistency level is ONE.
```

2. Bevor Sie mit cqlsh Daten in Ihre Amazon Keyspaces-Tabelle schreiben können, müssen Sie die Schreibkonsistenz für die aktuelle cqlsh-Sitzung auf festlegen. `LOCAL_QUORUM` Weitere Informationen zu den unterstützten Konsistenzstufen finden Sie unter. the section called “Konsistenzstufen für Schreibvorgänge” Beachten Sie, dass dieser Schritt nicht erforderlich ist, wenn Sie den CQL-Editor in der AWS-Managementkonsole verwenden.

```
CONSISTENCY LOCAL_QUORUM;
```

3. Um einen einzelnen Datensatz einzufügen, führen Sie den folgenden Befehl im CQL-Editor aus.

```
INSERT INTO catalog.book_awards (award, year, category, rank, author, book_title,
 publisher)
VALUES ('Wolf', 2023, 'Fiction',3,'Shirley Rodriguez','Mountain', 'AnyPublisher') ;
```

4. Stellen Sie sicher, dass die Daten korrekt zu Ihrer Tabelle hinzugefügt wurden, indem Sie den folgenden Befehl ausführen.

```
SELECT * FROM catalog.book_awards;
```

Die Ausgabe der Anweisung sollte wie folgt aussehen.

```
 year | award | category | rank | author            | book_title | publisher
------+-------+----------+------+-------------------+------------+-------------
 2023 |  Wolf |  Fiction |    3 | Shirley Rodriguez |   Mountain | AnyPublisher

(1 rows)
```

Um mehrere Datensätze aus einer Datei mit cqlsh einzufügen

1. Laden Sie die CSV-Beispieldatei (`keyspaces_sample_table.csv`) herunter, die in der Archivdatei samplemigration.zip enthalten ist. Entpacken Sie das Archiv und notieren Sie sich den Pfad zu`keyspaces_sample_table.csv`.

award	year	category	rank	author	book_title	publisher
Kwesi Manu Prize	2020	Fiction	1	Akua Mansa	Where did you go?	SomePublisher
Kwesi Manu Prize	2020	Fiction	2	John Stiles	Yesterday	Example Books
Kwesi Manu Prize	2020	Fiction	3	Nikki Wolf	Moving to the Chateau	AnyPublisher
Wolf	2020	Non-Fiction	1	Wang Xiulan	History of Ideas	Example Books
Wolf	2020	Non-Fiction	2	Ana Carolina Silva	Science Today	SomePublisher
Wolf	2020	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice	AnyPublisher
Richard Roe	2020	Fiction	1	Alejandro Rosalez	Long Summer	SomePublisher
Richard Roe	2020	Fiction	2	Arnav Desai	The Key	Example Books
Richard Roe	2020	Fiction	3	Mateo Jackson	Inside the Whale	AnyPublisher

2. Öffnen Sie AWS CloudShell das AWS-Managementkonsole und stellen Sie mit dem folgenden Befehl eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *`us-east-1`* mit Ihrer eigenen Region aktualisieren.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

3. Geben Sie an der `cqlsh` Eingabeaufforderung (`cqlsh>`) einen Schlüsselraum an.

```
USE catalog ;
```

4. Stellen Sie die Schreibkonsistenz auf ein. `LOCAL_QUORUM` Weitere Hinweise zu den unterstützten Konsistenzstufen finden Sie unterthe section called “Konsistenzstufen für Schreibvorgänge”.

```
CONSISTENCY LOCAL_QUORUM;
```

5. AWS CloudShell Wählen Sie oben rechts auf dem Bildschirm Aktionen aus und wählen Sie dann Datei hochladen, um die zuvor heruntergeladene CSV-Datei hochzuladen. Notieren Sie sich den Pfad zur Datei.

6. Führen Sie an der Keyspace-Eingabeaufforderung (`cqlsh:catalog>`) die folgende Anweisung aus.

```
COPY book_awards (award, year, category, rank, author, book_title, publisher) FROM
 '/home/cloudshell-user/keyspaces_sample_table.csv' WITH header=TRUE ;
```

Die Ausgabe der Anweisung sollte in etwa so aussehen.

```
cqlsh:catalog> COPY book_awards (award, year, category, rank, author,
 book_title, publisher)                     FROM '/home/cloudshell-user/
keyspaces_sample_table.csv' WITH delimiter=',' AND header=TRUE ;
cqlsh current consistency level is LOCAL_QUORUM.
Reading options from /home/cloudshell-user/.cassandra/cqlshrc:[copy]:
 {'numprocesses': '16', 'maxattempts': '1000'}
Reading options from /home/cloudshell-user/.cassandra/cqlshrc:[copy-from]:
 {'ingestrate': '1500', 'maxparseerrors': '1000', 'maxinserterrors': '-1',
 'maxbatchsize': '10', 'minbatchsize': '1', 'chunksize': '30'}
Reading options from the command line: {'delimiter': ',', 'header': 'TRUE'}
Using 16 child processes

Starting copy of catalog.book_awards with columns [award, year, category, rank,
 author, book_title, publisher].
OSError: handle is closed      0 rows/s; Avg. rate:       0 rows/s
Processed: 9 rows; Rate:       0 rows/s; Avg. rate:       0 rows/s
9 rows imported from 1 files in 0 day, 0 hour, 0 minute, and 26.706 seconds (0
 skipped).
```

7. Stellen Sie sicher, dass die Daten korrekt zu Ihrer Tabelle hinzugefügt wurden, indem Sie die folgende Abfrage ausführen.

```
SELECT * FROM book_awards ;
```

Die Ausgabe sollte folgendermaßen aussehen.

```
 year | award            | category    | rank | author             | book_title
       | publisher
------+------------------+-------------+------+--------------------
+----------------------+--------------
 2020 |             Wolf | Non-Fiction |    1 |        Wang Xiulan |      History
 of Ideas | Example Books
 2020 |             Wolf | Non-Fiction |    2 | Ana Carolina Silva |
 Science Today | SomePublisher
 2020 |             Wolf | Non-Fiction |    3 |  Shirley Rodriguez | The Future of
 Sea Ice |  AnyPublisher
 2020 | Kwesi Manu Prize |     Fiction |    1 |         Akua Mansa |     Where did
 you go? | SomePublisher
```

```
 2020 | Kwesi Manu Prize |     Fiction |    2 |        John Stiles |
 Yesterday | Example Books
 2020 | Kwesi Manu Prize |     Fiction |    3 |         Nikki Wolf | Moving to the
 Chateau |  AnyPublisher
 2020 |      Richard Roe |     Fiction |    1 |  Alejandro Rosalez |           Long
 Summer | SomePublisher
 2020 |      Richard Roe |     Fiction |    2 |        Arnav Desai |
 The Key | Example Books
 2020 |      Richard Roe |     Fiction |    3 |      Mateo Jackson |      Inside
 the Whale |  AnyPublisher

(9 rows)
```

Weitere Informationen zur Verwendung `cqlsh COPY` zum Hochladen von Daten aus CSV-Dateien in eine Amazon Keyspaces-Tabelle finden Sie unterthe section called “Daten mit cqlsh laden”.

Lesen Sie Daten aus einer Tabelle mithilfe der CQL SELECT-Anweisung in Amazon Keyspaces

In Daten in eine Amazon Keyspaces-Tabelle einfügen und laden diesem Abschnitt haben Sie die `SELECT` Anweisung verwendet, um zu überprüfen, ob Sie Ihrer Tabelle erfolgreich Daten hinzugefügt haben. In diesem Abschnitt verfeinern Sie Ihre Verwendung von, `SELECT` um bestimmte Spalten und nur Zeilen anzuzeigen, die bestimmte Kriterien erfüllen.

Die allgemeine Form der `SELECT` Aussage lautet wie folgt.

```
SELECT column_list FROM table_name [WHERE condition [ALLOW FILTERING]] ;
```

Themen

- Wählen Sie alle Daten in Ihrer Tabelle aus
- Wählen Sie eine Teilmenge von Spalten aus
- Wählen Sie eine Teilmenge von Zeilen aus

Wählen Sie alle Daten in Ihrer Tabelle aus

Die einfachste Form der `SELECT` Anweisung gibt alle Daten in Ihrer Tabelle zurück.

> **Important**
>
> In einer Produktionsumgebung empfiehlt es sich in der Regel nicht, diesen Befehl auszuführen, da er alle Daten in Ihrer Tabelle zurückgibt.

Um alle Daten Ihrer Tabelle auszuwählen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *us-east-1* mit Ihrer eigenen Region aktualisieren.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

2. Führen Sie die folgende Abfrage aus.

```
SELECT * FROM catalog.book_awards ;
```

   Wenn Sie das Platzhalterzeichen (*) für das verwenden, werden alle `column_list` Spalten ausgewählt. Die Ausgabe der Anweisung sieht wie im folgenden Beispiel aus.

```
 year | award            | category    | rank | author             | book_title
       | publisher
------+------------------+-------------+------+--------------------
+----------------------+--------------
 2020 |             Wolf | Non-Fiction |    1 |        Wang Xiulan |      History
 of Ideas |  AnyPublisher
 2020 |             Wolf | Non-Fiction |    2 | Ana Carolina Silva |
 Science Today | SomePublisher
 2020 |             Wolf | Non-Fiction |    3 |  Shirley Rodriguez | The Future of
 Sea Ice |  AnyPublisher
 2020 | Kwesi Manu Prize |     Fiction |    1 |         Akua Mansa |     Where did
 you go? | SomePublisher
 2020 | Kwesi Manu Prize |     Fiction |    2 |        John Stiles |
 Yesterday | Example Books
 2020 | Kwesi Manu Prize |     Fiction |    3 |         Nikki Wolf | Moving to the
 Chateau |  AnyPublisher
 2020 |      Richard Roe |     Fiction |    1 |  Alejandro Rosalez |           Long
 Summer | SomePublisher
 2020 |      Richard Roe |     Fiction |    2 |        Arnav Desai |
 The Key | Example Books
```

```
 2020 |      Richard Roe |     Fiction |    3 |      Mateo Jackson |      Inside
the Whale |  AnyPublisher
```

Wählen Sie eine Teilmenge von Spalten aus

Um eine Teilmenge von Spalten abzufragen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *us-east-1* mit Ihrer eigenen Region aktualisieren.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

2. Führen Sie die folgende Abfrage aus `awardcategory`, um nur die `year` Spalten, und abzurufen.

   ```
   SELECT award, category, year FROM catalog.book_awards ;
   ```

   Die Ausgabe enthält nur die angegebenen Spalten in der Reihenfolge, die in der `SELECT` Anweisung aufgeführt ist.

   ```
    award            | category    | year
   ------------------+-------------+------
                Wolf | Non-Fiction | 2020
                Wolf | Non-Fiction | 2020
                Wolf | Non-Fiction | 2020
    Kwesi Manu Prize |     Fiction | 2020
    Kwesi Manu Prize |     Fiction | 2020
    Kwesi Manu Prize |     Fiction | 2020
         Richard Roe |     Fiction | 2020
         Richard Roe |     Fiction | 2020
         Richard Roe |     Fiction | 2020
   ```

Wählen Sie eine Teilmenge von Zeilen aus

Wenn Sie einen großen Datensatz abfragen, benötigen Sie möglicherweise nur Datensätze, die bestimmte Kriterien erfüllen. Dazu können Sie eine `WHERE` Klausel an das Ende unserer `SELECT` Anweisung anhängen.

Um eine Teilmenge von Zeilen abzufragen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *us-east-1* mit Ihrer eigenen Region aktualisieren.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

2. Um nur die Datensätze für die Auszeichnungen eines bestimmten Jahres abzurufen, führen Sie die folgende Abfrage aus.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Wolf' ;
```

   Die vorherige SELECT Anweisung gibt die folgende Ausgabe zurück.

```
 year | award | category    | rank | author             | book_title            |
 publisher
------+-------+-------------+------+--------------------+-----------------------
+---------------
 2020 |  Wolf | Non-Fiction |    1 |        Wang Xiulan |      History of Ideas |
 AnyPublisher
 2020 |  Wolf | Non-Fiction |    2 | Ana Carolina Silva |         Science Today |
 SomePublisher
 2020 |  Wolf | Non-Fiction |    3 |  Shirley Rodriguez | The Future of Sea Ice |
 AnyPublisher
```

Die WHERE-Klausel verstehen

Die WHERE Klausel wird verwendet, um die Daten zu filtern und nur die Daten zurückzugeben, die die angegebenen Kriterien erfüllen. Bei den angegebenen Kriterien kann es sich um eine einfache oder eine zusammengesetzte Bedingung handeln.

Wie verwendet man Bedingungen in einer WHERE-Klausel

- Eine einfache Bedingung — Eine einzelne Spalte.

```
WHERE column_name=value
```

Sie können eine einfache Bedingung in einer `WHERE` Klausel verwenden, wenn eine der folgenden Bedingungen erfüllt ist:

- Die Spalte ist die einzige Partitionsschlüsselspalte der Tabelle.
- Sie fügen `ALLOW FILTERING` nach der Bedingung in der `WHERE` Klausel hinzu.

  Beachten Sie, dass die Verwendung zu einer inkonsistenten Leistung führen `ALLOW FILTERING` kann, insbesondere bei großen und mehrfach partitionierten Tabellen.

- Eine zusammengesetzte Bedingung — Mehrere einfache Bedingungen, die miteinander verbunden sind. `AND`

```
WHERE column_name1=value1 AND column_name2=value2 AND column_name3=value3...
```

Sie können zusammengesetzte Bedingungen in einer `WHERE` Klausel verwenden, wenn eine der folgenden Bedingungen erfüllt ist:

- Die Spalten, die Sie in der `WHERE` Klausel verwenden können, müssen entweder alle oder eine Teilmenge der Spalten im Partitionsschlüssel der Tabelle enthalten. Wenn Sie nur eine Teilmenge der Spalten in der `WHERE` Klausel verwenden möchten, müssen Sie eine zusammenhängende Gruppe von Partitionsschlüsselspalten von links nach rechts einbeziehen, wobei Sie mit der führenden Spalte des Partitionsschlüssels beginnen. Wenn die Partitionsschlüsselspalten beispielsweise, lauten `yearmonth`, und `award` dann können Sie die folgenden Spalten in der Klausel verwenden: `WHERE`
  - `year`
  - `year`UND `month`
  - `year`UND `month` UND `award`
- Sie fügen `ALLOW FILTERING` nach der zusammengesetzten Bedingung in der `WHERE` Klausel hinzu, wie im folgenden Beispiel.

```
SELECT * FROM my_table WHERE col1=5 AND col2='Bob' ALLOW FILTERING ;
```

Beachten Sie, dass die Verwendung zu einer inkonsistenten Leistung führen `ALLOW FILTERING` kann, insbesondere bei großen und mehrfach partitionierten Tabellen.

Probieren Sie es aus

Erstellen Sie Ihre eigenen CQL-Abfragen, um Folgendes aus Ihrer `book_awards` Tabelle zu finden:

- Finden Sie die Gewinner der Wolf Awards 2020 und zeigen Sie die Buchtitel und Autoren, sortiert nach Rang, an.
- Zeigt die ersten Preisträger aller Auszeichnungen im Jahr 2020 an und zeigt die Buchtitel und Preisnamen an.

Aktualisieren Sie Daten in einer Amazon Keyspaces-Tabelle mithilfe von CQL

Verwenden Sie die UPDATE Anweisung, um die Daten in Ihrer book_awards Tabelle zu aktualisieren.

Die allgemeine Form der UPDATE Aussage lautet wie folgt.

```
UPDATE table_name SET column_name=new_value WHERE primary_key=value ;
```

Tip

- Sie können mehrere Spalten aktualisieren, indem Sie eine durch Kommas getrennte Liste von column_names UND-Werten verwenden, wie im folgenden Beispiel dargestellt.

  ```
  UPDATE my_table SET col1='new_value_1', col2='new_value2' WHERE col3='1' ;
  ```

- Wenn der Primärschlüssel aus mehreren Spalten besteht, müssen alle Primärschlüsselspalten und ihre Werte in der WHERE Klausel enthalten sein.
- Sie können keine Spalte im Primärschlüssel aktualisieren, da sich dadurch der Primärschlüssel für den Datensatz ändern würde.

Um eine einzelne Zelle zu aktualisieren

Ändern Sie anhand Ihrer book_awards Tabelle den Namen eines Verlags, der 2020 die Wolf Awards für Sachbücher gewonnen hat.

```
UPDATE book_awards SET publisher='new Books' WHERE year = 2020 AND award='Wolf' AND
 category='Non-Fiction' AND rank=1;
```

Vergewissern Sie sich, dass es sich jetzt new Books um einen Verlag handelt.

```
SELECT * FROM book_awards WHERE year = 2020 AND award='Wolf' AND category='Non-Fiction'
 AND rank=1;
```

Die Anweisung sollte die folgende Ausgabe zurückgeben.

```
 year | award | category    | rank | author      | book_title       | publisher
------+-------+-------------+------+-------------+------------------+-----------
 2020 |  Wolf | Non-Fiction |    1 | Wang Xiulan | History of Ideas | new Books
```

Versuch es

Fortgeschritten: Der Gewinner des „Kwezi Manu Prize“ 2020 für Belletristik hat seinen Namen geändert. Aktualisiere diesen Datensatz, um den Namen in zu ändern. `'Akua Mansa-House'`

Löschen Sie Daten aus einer Tabelle mit der CQL **`DELETE-Anweisung`**

Verwenden Sie die Anweisung, um Daten in Ihrer `book_awards` Tabelle zu löschen. `DELETE`

Sie können Daten aus einer Zeile oder aus einer Partition löschen. Seien Sie vorsichtig beim Löschen von Daten, da Löschungen irreversibel sind.

Durch das Löschen einer oder aller Zeilen aus einer Tabelle wird die Tabelle nicht gelöscht. Sie können sie also erneut mit Daten füllen. Durch das Löschen einer Tabelle werden die Tabelle und alle darin enthaltenen Daten gelöscht. Um die Tabelle wieder verwenden zu können, müssen Sie sie neu erstellen und ihr Daten hinzufügen. Durch das Löschen eines Schlüsselraums werden der Schlüsselraum und alle darin enthaltenen Tabellen gelöscht. Um den Schlüsselraum und die Tabellen verwenden zu können, müssen Sie sie neu erstellen und sie dann mit Daten füllen. Sie können Amazon Keyspaces Point-in-time (PITR) Recovery verwenden, um gelöschte Tabellen wiederherzustellen. Weitere Informationen finden Sie unter. the section called “Backup und Wiederherstellung mit Point-in-Time-Recovery” Informationen zum Wiederherstellen einer gelöschten Tabelle mit aktiviertem PITR finden Sie unter. the section called “Eine gelöschte Tabelle wiederherstellen”

Zellen löschen

Beim Löschen einer Spalte aus einer Zeile werden die Daten aus der angegebenen Zelle entfernt. Wenn Sie diese Spalte mithilfe einer `SELECT` Anweisung anzeigen, werden die Daten als angezeigt*`null`*, obwohl an dieser Stelle kein Nullwert gespeichert ist.

Die allgemeine Syntax zum Löschen einer oder mehrerer bestimmter Spalten lautet wie folgt.

```
DELETE column_name1[, column_name2...] FROM table_name WHERE condition ;
```

In Ihrer `book_awards` Tabelle können Sie sehen, dass der Titel des Buches, das den ersten Preis des „Richard Roe“ -Preises 2020 gewonnen hat, „Long Summer“ lautet. Stellen Sie sich vor, dieser Titel wurde zurückgerufen und Sie müssen die Daten aus dieser Zelle löschen.

Um eine bestimmte Zelle zu löschen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *us-east-1* mit Ihrer eigenen Region aktualisieren.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

2. Führen Sie die folgende `DELETE` Abfrage aus.

   ```
   DELETE book_title FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe'
    AND category='Fiction' AND rank=1;
   ```

3. Stellen Sie sicher, dass die Löschanforderung wie erwartet gestellt wurde.

   ```
   SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
    category='Fiction' AND rank=1;
   ```

   Die Ausgabe dieser Anweisung sieht wie folgt aus.

   ```
    year | award       | category | rank | author            | book_title | publisher
   ------+-------------+----------+------+-------------------+------------
   +---------------
    2020 | Richard Roe |  Fiction |    1 | Alejandro Rosalez |       null |
    SomePublisher
   ```

Zeilen löschen

Es kann vorkommen, dass Sie eine ganze Zeile löschen müssen, z. B. um einer Datenlöschungsanforderung nachzukommen. Die allgemeine Syntax für das Löschen einer Zeile lautet wie folgt.

```
DELETE FROM table_name WHERE condition ;
```

Um eine Zeile zu löschen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *us-east-1* mit Ihrer eigenen Region aktualisieren.

   ```
   cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
   ```

2. Führen Sie die folgende DELETE Abfrage aus.

   ```
   DELETE FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
    category='Fiction' AND rank=1;
   ```

3. Stellen Sie sicher, dass der Löschvorgang wie erwartet durchgeführt wurde.

   ```
   SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
    category='Fiction' AND rank=1;
   ```

   Die Ausgabe dieser Anweisung sieht nach dem Löschen der Zeile wie folgt aus.

   ```
    year | award | category | rank | author | book_title | publisher
   ------+-------+----------+------+--------+------------+-----------

   (0 rows)
   ```

Mit Amazon Keyspaces Time to Live können Sie abgelaufene Daten automatisch aus Ihrer Tabelle löschen. Weitere Informationen finden Sie unterthe section called “Daten mit Time to Live ablaufen lassen”.

Löschen Sie eine Tabelle in Amazon Keyspaces

Löschen Sie alle Tabellen, die Sie nicht verwenden, um zu vermeiden, dass Ihnen für Tabellen und Daten, die Sie nicht benötigen, Gebühren berechnet werden. Wenn Sie eine Tabelle löschen, werden die Tabelle und ihre Daten gelöscht, und es fallen keine Gebühren mehr für sie an.
Der Schlüsselraum bleibt jedoch erhalten. Wenn Sie einen Schlüsselraum löschen, werden der Schlüsselraum und alle zugehörigen Tabellen gelöscht und es fallen keine Gebühren mehr für sie an.

Sie können eine Tabelle mit der Konsole, CQL oder der löschen. AWS CLI Wenn Sie eine Tabelle löschen, werden die Tabelle und alle ihre Daten gelöscht.

Verwenden der Konsole

Das folgende Verfahren löscht eine Tabelle und alle zugehörigen Daten mithilfe von. AWS-Managementkonsole

Um eine Tabelle mit der Konsole zu löschen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.
3. Wählen Sie das Feld links neben dem Namen jeder Tabelle aus, die Sie löschen möchten.
4. Wählen Sie Löschen aus.
5. Geben Sie auf dem Bildschirm „Tabelle löschen“ **`Delete`** in das Feld ein. Wählen Sie dann Tabelle löschen.
6. Um zu überprüfen, ob die Tabelle gelöscht wurde, wählen Sie im Navigationsbereich Tabellen aus und vergewissern Sie sich, dass die `book_awards` Tabelle nicht mehr aufgeführt wird.

Verwenden von CQL

Das folgende Verfahren löscht eine Tabelle und alle ihre Daten mithilfe von CQL.

Um eine Tabelle mit CQL zu löschen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *`us-east-1`* mit Ihrer eigenen Region aktualisieren.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

2. Löschen Sie Ihre Tabelle, indem Sie die folgende Anweisung eingeben.

```
DROP TABLE IF EXISTS catalog.book_awards ;
```

3. Vergewissern Sie sich, dass Ihre Tabelle gelöscht wurde.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = 'catalog' ;
```

Die Ausgabe sollte wie folgt aussehen. Beachten Sie, dass dies einige Zeit in Anspruch nehmen kann. Führen Sie die Anweisung daher nach einer Minute erneut aus, wenn Sie dieses Ergebnis nicht sehen.

```
keyspace_name | table_name | bloom_filter_fp_chance | caching | cdc | comment
 | compaction | compression | crc_check_chance | dclocal_read_repair_chance
 | default_time_to_live | extensions | flags | gc_grace_seconds | id |
 max_index_interval | memtable_flush_period_in_ms | min_index_interval |
 read_repair_chance | speculative_retry
---------------+------------+------------------------+---------+-----+---------
+------------+-------------+------------------+---------------------------
+----------------------+------------+-------+------------------+----
+--------------------+-----------------------------+--------------------
+--------------------+------------------

(0 rows)
```

Verwendung der AWS CLI

Das folgende Verfahren löscht eine Tabelle und alle zugehörigen Daten mithilfe von. AWS CLI

Um eine Tabelle zu löschen, verwenden Sie den AWS CLI

1. Öffnen CloudShell
2. Löschen Sie Ihre Tabelle mit der folgenden Anweisung.

```
aws keyspaces delete-table --keyspace-name 'catalog' --table-name 'book_awards'
```

3. Um zu überprüfen, ob Ihre Tabelle gelöscht wurde, können Sie alle Tabellen in einem Schlüsselraum auflisten.

```
aws keyspaces list-tables --keyspace-name 'catalog'
```

Die Ausgabe sollte folgendermaßen aussehen. Beachten Sie, dass dieser asynchrone Vorgang einige Zeit in Anspruch nehmen kann. Re-run Führen Sie den Befehl nach kurzer Zeit erneut aus, um zu bestätigen, dass die Tabelle gelöscht wurde.

```
{
    "tables": []
}
```

Löschen Sie einen Schlüsselraum in Amazon Keyspaces

Löschen Sie alle Schlüsselräume, die Sie nicht verwenden, um zu vermeiden, dass Ihnen Schlüsselräume in Rechnung gestellt werden. Wenn Sie einen Schlüsselraum löschen, werden der Schlüsselraum und alle zugehörigen Tabellen gelöscht und es fallen keine Gebühren mehr dafür an.

Sie können einen Schlüsselraum entweder mit der Konsole, CQL oder dem löschen. AWS CLI

Verwenden der Konsole

Das folgende Verfahren löscht einen Schlüsselraum und alle zugehörigen Tabellen und Daten mithilfe der Konsole.

Um einen Schlüsselraum mithilfe der Konsole zu löschen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter. https://console.aws.amazon.com/keyspaces/home
2. Wählen Sie im Navigationsbereich Keyspaces aus.
3. Wählen Sie das Feld links neben dem Namen jedes Schlüsselraums aus, den Sie löschen möchten.
4. Wählen Sie Löschen aus.
5. Geben Sie auf dem Bildschirm „Schlüsselraum löschen“ den Text **`Delete`** in das Feld ein. Wählen Sie dann „Schlüsselraum löschen“.
6. Um zu überprüfen, ob der Schlüsselraum gelöscht `catalog` wurde, wählen Sie im Navigationsbereich die Option Schlüsselräume aus und vergewissern Sie sich, dass er nicht mehr aufgeführt wird. Da Sie den Schlüsselraum gelöscht haben, sollte die **`book_awards`** Tabelle unter Tabellen ebenfalls nicht aufgeführt werden.

Verwenden von CQL

Das folgende Verfahren löscht einen Schlüsselraum und alle zugehörigen Tabellen und Daten mithilfe von CQL.

Um einen Schlüsselraum mit CQL zu löschen

1. Öffnen Sie Amazon Keyspaces AWS CloudShell und stellen Sie mithilfe des folgenden Befehls eine Verbindung zu Amazon Keyspaces her. Stellen Sie sicher, dass Sie *us-east-1* mit Ihrer eigenen Region aktualisieren.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

2. Löschen Sie Ihren Schlüsselraum, indem Sie die folgende Anweisung eingeben.

```
DROP KEYSPACE IF EXISTS catalog ;
```

3. Vergewissern Sie sich, dass Ihr Schlüsselraum gelöscht wurde.

```
SELECT * from system_schema.keyspaces ;
```

   Ihr Schlüsselraum sollte nicht aufgeführt werden. Beachten Sie, dass es sich um einen asynchronen Vorgang handelt, zu einer Verzögerung kommen kann, bis der Schlüsselraum gelöscht wird. Nachdem der Schlüsselraum gelöscht wurde, sollte die Ausgabe der Anweisung wie folgt aussehen.

```
keyspace_name           | durable_writes | replication
-------------------------+----------------
+-------------------------------------------------------------------------------------
           system_schema |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
      system_schema_mcs |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
                 system |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
 system_multiregion_info |           True | {'class':
 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(4 rows)
```

Verwendung der AWS CLI

Das folgende Verfahren löscht einen Schlüsselraum und alle zugehörigen Tabellen und Daten mithilfe von. AWS CLI

Um einen Schlüsselraum zu löschen, verwenden Sie den AWS CLI

1. Öffnen AWS CloudShell
2. Löschen Sie Ihren Schlüsselraum, indem Sie die folgende Anweisung eingeben.

```
aws keyspaces delete-keyspace --keyspace-name 'catalog'
```

3. Vergewissern Sie sich, dass Ihr Schlüsselraum gelöscht wurde.

```
aws keyspaces list-keyspaces
```

Die Ausgabe dieser Anweisung sollte ähnlich wie die folgende aussehen und nur die Systemschlüsselräume auflisten. Beachten Sie, dass es sich um einen asynchronen Vorgang handelt, zu einer Verzögerung kommen kann, bis Ihr Schlüsselraum gelöscht wird.

```
{
    "keyspaces": [
        {
            "keyspaceName": "system_schema",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system_schema/",
            "replicationStrategy": "SINGLE_REGION"
        },
        {
            "keyspaceName": "system_schema_mcs",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system_schema_mcs/",
            "replicationStrategy": "SINGLE_REGION"
        },
        {
            "keyspaceName": "system",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system/",
            "replicationStrategy": "SINGLE_REGION"
        },
        {
            "keyspaceName": "system_multiregion_info",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
system_multiregion_info/",
            "replicationStrategy": "SINGLE_REGION"
        }
    ]
```

```
}
```

Tutorials und Lösungen für Amazon Keyspaces

Dieses Thema umfasst verschiedene Tutorials, in denen die detaillierten Schritte beschrieben werden, die erforderlich sind, um eine programmgesteuerte Verbindung zu Amazon Keyspaces herzustellen oder Amazon Keyspaces mit anderen AWS Diensten zur Erstellung von Lösungen zu verwenden. Die verfügbaren Tutorials behandeln verschiedene beliebte serviceübergreifende Zugriffsszenarien und zeigen, wie Amazon Keyspaces mit Amazon Virtual Private Cloud, Amazon Elastic Kubernetes Service, Amazon Simple Storage Service und/oder mit Open-Source-Technologien wie AWS Glue Apache Spark integriert werden kann.

Schrittweise Anleitungen, die zeigen, wie Sie mithilfe von Apache Cassandra-Treibern eine Verbindung zu Amazon Keyspaces herstellen, finden Sie unter. the section called “Verwenden eines Cassandra-Client-Treibers”

Themen

- Tutorial: Stellen Sie über einen VPC-Endpunkt mit einer Schnittstelle eine Verbindung zu Amazon Keyspaces her
- Tutorial: Integrieren Sie Apache Spark, um Daten zu importieren oder zu exportieren
- Tutorial: Connect über eine containerisierte Anwendung her, die auf Amazon Elastic Kubernetes Service gehostet wird
- Tutorial: Exportieren Sie eine Amazon Keyspaces-Tabelle nach Amazon S3 mit AWS Glue

Tutorial: Stellen Sie über einen VPC-Endpunkt mit einer Schnittstelle eine Verbindung zu Amazon Keyspaces her

Dieses Tutorial führt Sie durch die Einrichtung und Verwendung eines Schnittstellen-VPC-Endpunkts für Amazon Keyspaces.

Schnittstellen-VPC-Endpunkte ermöglichen die private Kommunikation zwischen Ihrer Virtual Private Cloud (VPC), die in Amazon VPC läuft, und Amazon Keyspaces. Schnittstelle, über die VPC-Endpunkte betrieben werden. Dabei handelt es sich um einen AWS Dienst AWS PrivateLink, der die private Kommunikation zwischen VPCs und Diensten ermöglicht. AWS Weitere Informationen finden Sie unter the section called “Verwendung von -Schnittstellen-VPC-Endpunkten”.

Themen

- Voraussetzungen und Überlegungen zum Tutorial

- Schritt 1: Starten einer Amazon-EC2-Instance
- Schritt 2: Konfigurieren Ihrer Amazon-EC2-Instance
- Schritt 3: Erstellen Sie einen VPC-Endpunkt für Amazon Keyspaces
- Schritt 4: Berechtigungen für die VPC-Endpunktverbindung konfigurieren
- Schritt 5: Konfigurieren Sie die Überwachung mit CloudWatch
- Schritt 6: (Optional) Bewährte Methoden zur Konfiguration der Größe des Verbindungspools für Ihre Anwendung
- Schritt 7: (Optional) Aufräumen

Voraussetzungen und Überlegungen zum Tutorial

Bevor Sie mit diesem Tutorial beginnen, folgen Sie den Anweisungen zur AWS Einrichtung unter. Zugreifen auf Amazon Keyspaces (für Apache Cassandra) Zu diesen Schritten gehören die Registrierung AWS und Erstellung eines AWS Identity and Access Management (IAM-) Principals mit Zugriff auf Amazon Keyspaces. Notieren Sie sich den Namen des IAM-Benutzers und die Zugriffsschlüssel, da Sie sie später in diesem Tutorial benötigen.

Erstellen Sie einen Schlüsselraum mit dem Namen `myKeyspace` und mindestens einer Tabelle, um die Verbindung mit dem VPC-Endpunkt weiter unten in diesem Tutorial zu testen. Eine ausführliche Anleitung finden Sie unter. Erste Schritte

Nachdem Sie die erforderlichen Schritte abgeschlossen haben, fahren Sie mit fortSchritt 1: Starten einer Amazon-EC2-Instance.

Schritt 1: Starten einer Amazon-EC2-Instance

In diesem Schritt starten Sie eine Amazon-EC2-Instance in Ihrer Standard-Amazon-VPC. Anschließend können Sie einen VPC-Endpunkt für Amazon Keyspaces erstellen und verwenden.

So starten Sie eine Amazon EC2 Instance

1. Öffnen Sie die Amazon-EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie Launch Instance aus und gehen Sie folgendermaßen vor:

   Wählen Sie im Dashboard der EC2-Konsole im Feld Launch Instance die Option Launch Instance und dann aus den angezeigten Optionen Launch Instance aus.

   Geben Sie unter Name und Tags für Name einen aussagekräftigen Namen für Ihre Instance ein.

Unter Anwendungs- und Betriebssystemimages (Amazon Machine Image):

- Wählen Sie Schnellstart und dann Ubuntu. Dies ist das Betriebssystem für Ihre Instance.
- Unter Amazon Machine Image (AMI) können Sie das Standard-Image verwenden, für das das kostenlose Kontingent gilt. Ein Amazon Machine Image (AMI) dient als grundlegende Konfigurationsvorlage für Ihre Instance.

Unter Instanztyp:

- Wählen Sie aus der Liste Instance-Typ den Instance-Typ t2.micro aus, der standardmäßig ausgewählt ist.

Wählen Sie unter Schlüsselpaar (Anmeldung) für Name des Schlüsselpaars eine der folgenden Optionen für dieses Tutorial aus:

- Wenn Sie nicht über ein Amazon-EC2-Schlüsselpaar verfügen, wählen Sie Neues Schlüsselpaar erstellen aus und befolgen Sie die Anweisungen. Sie werden aufgefordert, eine private Schlüsseldatei (.pem-Datei) herunterzuladen. Sie benötigen diese Datei später, wenn Sie sich bei Ihrer Amazon EC2 EC2-Instance anmelden. Notieren Sie sich also den Dateipfad.
- Wenn Sie bereits über ein bestehendes Amazon-EC2-Schlüsselpaar verfügen, navigieren Sie zu Schlüsselpaar wählen und wählen Sie Ihr Schlüsselpaar aus der Liste aus. Beachten Sie, dass Sie bereits über die private Schlüsseldatei (.pem-Datei) verfügen müssen, um sich bei Ihrer Amazon-EC2-Instance anmelden zu können.

Unter Netzwerkeinstellungen:

- Wählen Sie Bearbeiten aus.
- Wählen Sie Bestehende Sicherheitsgruppe auswählen aus.
- Wählen Sie in der Liste der Sicherheitsgruppen default aus. Dies ist die Standard-Sicherheitsgruppe für Ihre VPC.

Weiter zur Zusammenfassung.

- Sehen Sie sich im Bereich Zusammenfassung eine Zusammenfassung Ihrer Instanzkonfiguration an. Sobald Sie bereit sind, wählen Sie Instance starten aus.

3. Wählen Sie auf dem Abschlussbildschirm für die neue Amazon EC2 EC2-Instance die Kachel Connect to Instance aus. Der nächste Bildschirm zeigt die erforderlichen Informationen und die erforderlichen Schritte, um eine Verbindung zu Ihrer neuen Instance herzustellen. Notieren Sie sich die folgenden Informationen:

   - Der Beispielbefehl zum Schutz der Schlüsseldatei
   - Die Verbindungszeichenfolge
   - Der öffentliche IPv4-DNS-Name

   Nachdem Sie die Informationen auf dieser Seite zur Kenntnis genommen haben, können Sie mit dem nächsten Schritt in diesem Tutorial fortfahren (Schritt 2: Konfigurieren Ihrer Amazon-EC2-Instance).

> **Note**
>
> Es dauert einige Minuten, bis Ihre Amazon-EC2-Instance verfügbar ist. Überprüfen Sie, ob Instance-Zustand `running` wird und alle Status-Kontrollen bestanden hat, bevor Sie mit dem nächsten Schritt fortfahren.

Schritt 2: Konfigurieren Ihrer Amazon-EC2-Instance

Wenn Ihre Amazon EC2 EC2-Instance verfügbar ist, können Sie sich bei ihr anmelden und sie für die erste Verwendung vorbereiten.

> **Note**
>
> Bei den folgenden Schritten wird davon ausgegangen, dass Sie von einem Computer aus, auf dem Linux ausgeführt wird, eine Verbindung zu Ihrer Amazon EC2 EC2-Instance herstellen. Weitere Verbindungsmöglichkeiten finden Sie unter Connect to your Linux Instance im Amazon EC2 EC2-Benutzerhandbuch.

So konfigurieren Sie Ihre Amazon EC2 EC2-Instance

1. Sie müssen eingehenden SSH-Verkehr zu Ihrer Amazon EC2 EC2-Instance autorisieren. Erstellen Sie dazu eine neue EC2-Sicherheitsgruppe und weisen Sie die Sicherheitsgruppe dann Ihrer EC2-Instance zu.

   a. Wählen Sie im Navigationsbereich Sicherheitsgruppen aus.

   b. Wählen Sie Sicherheitsgruppen erstellen aus. Führen Sie im Fenster Sicherheitsgruppen erstellen Folgendes aus:

      - Name der Sicherheitsgruppe — Geben Sie einen Namen für Ihre Sicherheitsgruppe ein. Beispiel: `my-ssh-access`
      - Beschreibung — Geben Sie eine kurze Beschreibung für die Sicherheitsgruppe ein.
      - VPC — Wählen Sie Ihre Standard-VPC.
      - Wählen Sie im Abschnitt Eingehende Regeln die Option Regel hinzufügen aus und gehen Sie wie folgt vor:
        - Typ — Wählen Sie SSH.
        - Quelle — Wähle meine IP.
        - Wählen Sie Regel hinzufügen aus.

      Bestätigen Sie unten auf der Seite die Konfigurationseinstellungen und wählen Sie Create Security Group aus.

   c. Wählen Sie im Navigationsbereich Instances aus.

   d. Wählen Sie die Amazon-EC2-Instance aus, die Sie in Schritt 1: Starten einer Amazon-EC2-Instance gestartet haben.

   e. Wählen Sie Aktionen, Sicherheit und dann Sicherheitsgruppen ändern aus.

   f. Gehen Sie unter Sicherheitsgruppen ändern zu Zugeordnete Sicherheitsgruppen und geben Sie die Sicherheitsgruppe ein, die Sie zuvor in diesem Verfahren erstellt haben (z. B.`my-ssh-access`). Die vorhandene `default`-Sicherheitsgruppe sollte ebenfalls ausgewählt werden. Bestätigen Sie die Konfigurationseinstellungen und wählen Sie Speichern.

2. Verwenden Sie den folgenden Befehl, um Ihre private Schlüsseldatei vor Zugriff zu schützen. Wenn Sie diesen Schritt überspringen, schlägt die Verbindung fehl.

```
chmod 400 path_to_file/my-keypair.pem
```

3. Verwenden Sie den Befehl `ssh`, wie im folgenden Beispiel gezeigt, um sich bei der Amazon-EC2-Instance anzumelden.

```
ssh -i path_to_file/my-keypair.pem ubuntu@public-dns-name
```

Sie müssen Ihre private Schlüsseldatei (.pem-Datei) und den öffentlichen DNS-Namen Ihrer Instance angeben. (Siehe Schritt 1: Starten einer Amazon-EC2-Instance).

Die Anmelde-ID lautet `ubuntu`. Es ist kein Passwort erforderlich.

Weitere Informationen zum Zulassen von Verbindungen zu Ihrer Amazon EC2 EC2-Instance und AWS CLI Anweisungen finden Sie unter Autorisieren von eingehendem Datenverkehr für Ihre Linux-Instances im Amazon EC2 EC2-Benutzerhandbuch.

4. Laden Sie die neueste Version von herunter und installieren Sie sie. AWS Command Line Interface

   a. Installieren `unzip`.

```
sudo apt install unzip
```

   b. Laden Sie die `zip` Datei mit dem herunter AWS CLI.

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
```

   c. Entpacken Sie die Datei.

```
unzip awscliv2.zip
```

   d. Installieren Sie das AWS CLI.

```
sudo ./aws/install
```

   e. Bestätigen Sie die Version der AWS CLI Installation.

```
aws --version
```

Die Ausgabe sollte in etwa wie folgt aussehen:

```
aws-cli/2.9.19 Python/3.9.11 Linux/5.15.0-1028-aws exe/x86_64.ubuntu.22 prompt/
off
```

5. Konfigurieren Sie Ihre AWS Anmeldeinformationen, wie im folgenden Beispiel gezeigt. Geben Sie Ihre AWS Zugriffsschlüssel-ID, Ihren geheimen Schlüssel und den Namen der Standardregion ein, wenn Sie dazu aufgefordert werden.

```
aws configure

AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-east-1
Default output format [None]:
```

6. Fügen Sie Ihrer VPCs-Sicherheitsgruppe Regeln hinzu, die eingehenden HTTP-, HTTPS- und SSH-Zugriff von IPv6-Adressen aus ermöglichen.
7. Um zu bestätigen, dass Ihr VPC-Endpunkt korrekt konfiguriert wurde, müssen Sie eine `cqlsh` Verbindung zu Amazon Keyspaces verwenden. Wenn Sie Ihre lokale Umgebung oder den Amazon Keyspaces CQL-Editor in der verwenden AWS-Managementkonsole, geht die Verbindung automatisch über den öffentlichen Endpunkt statt über Ihren VPC-Endpunkt. Um Ihre VPC-Endpunktverbindung in diesem Tutorial `cqlsh` zu testen, folgen Sie den Anweisungen zur Einrichtung unter. Verwenden`cqlsh`, um eine Verbindung zu Amazon Keyspaces herzustellen

Sie sind jetzt bereit, einen Dual-Stack-VPC-Endpunkt für Amazon Keyspaces zu erstellen.

Schritt 3: Erstellen Sie einen VPC-Endpunkt für Amazon Keyspaces

In diesem Schritt erstellen Sie einen Dual-Stack-VPC-Endpunkt für Amazon Keyspaces mithilfe von. AWS CLIUm den VPC-Endpunkt mithilfe der VPC-Konsole zu erstellen, können Sie den Anweisungen zum Erstellen eines VPC-Endpunkts im Handbuch folgen.AWS PrivateLink Geben Sie ein, wenn Sie nach dem Dienstnamen filtern. **`Cassandra`**

So erstellen Sie einen VPC-Endpunkt mit dem AWS CLI

1. Bevor Sie beginnen, stellen Sie sicher, dass Sie über den öffentlichen Endpunkt mit Amazon Keyspaces kommunizieren können.

```
aws keyspaces list-tables --keyspace-name 'myKeyspace'
```

Die Ausgabe zeigt eine Liste von Amazon Keyspaces-Tabellen, die im angegebenen Schlüsselraum enthalten sind. Wenn Sie keine Tabellen haben, ist die Liste leer.

```
{
    "tables": [
        {
            "keyspaceName": "myKeyspace",
            "tableName": "myTable1",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
catalog/table/myTable1"
        },
        {
            "keyspaceName": "myKeyspace",
            "tableName": "myTable2",
            "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
catalog/table/myTable2"
        }
    ]
}
```

2. Stellen Sie sicher, dass Amazon Keyspaces ein verfügbarer Service für die Erstellung von VPC-Endpunkten in der aktuellen Region ist. AWS (Der Befehl wird in Fettschrift angezeigt, gefolgt von einer Beispielausgabe.)

```
aws ec2 describe-vpc-endpoint-services

{
    "ServiceNames": [
        "com.amazonaws.us-east-1.cassandra",
        "com.amazonaws.us-east-1.cassandra-fips"
        "api.aws.us-east-1.cassandra-streams"
    ]
}
```

Wenn Amazon Keyspaces einer der verfügbaren Dienste in der Ausgabe des Befehls ist, können Sie mit der Erstellung eines VPC-Endpunkts fortfahren.

3. Um über IPv6-fähige Dual-Stack-Endpunkte eine Verbindung zu Amazon Keyspaces herzustellen, stellen Sie sicher, dass Ihre VPC IPv6 unterstützt, und konfigurieren Sie Subnetze mit IPv6-Unterstützung. Informationen zum Hinzufügen von IPv6-Unterstützung zu einer

bestehenden VPC, die derzeit nur IPv4 unterstützt, finden Sie unter IPv6-Unterstützung für Ihre VPC im Amazon VPC-Benutzerhandbuch;.

4. Ermitteln Sie den VPC-Bezeichner.

```
aws ec2 describe-vpcs

{
    "Vpcs": [
        {
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "Ipv6CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-0000aaa0a00a00aa0",
                    "Ipv6CidrBlock": "2600:1f18:e19:7d00::/56",
                    "Ipv6CidrBlockState": {
                        "State": "associated"
                    },
                    "NetworkBorderGroup": "us-east-1",
                    "Ipv6Pool": "Amazon",
                    "Ipv6AddressAttribute": "public",
                    "IpSource": "amazon"
                }
            ],
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00a0000a",
                    "CidrBlock": "111.11.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": true,
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "VpcId": "vpc-a1234bcd",
            "State": "available",
            "CidrBlock": "111.11.0.0/16",
            "DhcpOptionsId": "dopt-a00aaaaa"
        }
    ]
```

```
}
```

In der Beispielausgabe ist die VPC-ID `vpc-a1234bcd`.

5. Verwenden Sie einen Filter, um Details zu den Subnetzen der VPC zu sammeln.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-a1234bcd"

{
    "Subnets": [
        {
            "AvailabilityZoneId": "use1-az1",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "111122223333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [
                {
                    "AssociationId": "subnet-cidr-assoc-05d75732736740283",
                    "Ipv6CidrBlock": "**********************",
                    "Ipv6CidrBlockState": {
                        "State": "associated"
                    },
                    "Ipv6AddressAttribute": "public",
                    "IpSource": "amazon"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-70b24b16",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-70b24b16",
            "State": "available",
            "VpcId": "vpc-a1234bcd",
            "CidrBlock": "**********/20",
            "AvailableIpAddressCount": 4089,
            "AvailabilityZone": "us-east-1a",
```

```
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true
        },
        {
            "AvailabilityZoneId": "use1-az2",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "111122223333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [
                {
                    "AssociationId": "subnet-cidr-assoc-0ec6fb253e05b17eb",
                    "Ipv6CidrBlock": "**********************",
                    "Ipv6CidrBlockState": {
                        "State": "associated"
                    },
                    "Ipv6AddressAttribute": "public",
                    "IpSource": "amazon"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/subnet-
c63ffbe7",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-c63ffbe7",
            "State": "available",
            "VpcId": "vpc-a1234bcd",
            "CidrBlock": "***********/20",
            "AvailableIpAddressCount": 4087,
            "AvailabilityZone": "us-east-1b",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true
        }
    ]
}
```

In der Beispielausgabe sind zwei Subnetz-IDs verfügbar: und. `subnet-70b24b16` `subnet-c63ffbe7`

6. Erstellen des VPC-Endpunkts Geben Sie für den Parameter `--vpc-id` die VPC-ID aus dem vorherigen Schritt an. Geben Sie für den `--subnet-ids` Parameter die Subnetz-IDs aus dem vorherigen Schritt an. Verwenden Sie den `--vpc-endpoint-type` Parameter, um den Endpunkt als Schnittstelle zu definieren. Um einen Dual-Stack-Endpunkt zu erstellen, verwenden Sie`--ip-address-type dualstack`. Weitere Informationen zu dem Befehl finden Sie `create-vpc-endpoint`in der AWS CLI Befehlsreferenz.

```
aws ec2 create-vpc-endpoint \
 --vpc-endpoint-type Interface \
 --vpc-id vpc-a1234bcd \
 --ip-address-type dualstack \
 --service-name com.amazonaws.us-east-1.cassandra \
 --subnet-ids subnet-70b24b16 subnet-c63ffbe7

{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-000000abc111d2ef3",
        "VpcEndpointType": "Interface",
        "VpcId": "vpc-a1234bcd",
        "ServiceName": "com.amazonaws.us-east-1.cassandra",
        "State": "pending",
        "RouteTableIds": [],
        "SubnetIds": [
            "subnet-70b24b16",
            "subnet-c63ffbe7"
        ],
        "Groups": [
            {
                "GroupId": "sg-0123456789",
                "GroupName": "default"
            }
        ],
        "IpAddressType": "dualstack",
        "DnsOptions": {
            "DnsRecordIpType": "dualstack"
        },
        "PrivateDnsEnabled": true,
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
```

```
            "eni-08cd525f72ea6f1fa",
            "eni-07b1f6c895169d8fb"
        ],
        "DnsEntries": [
            {
                "DnsName": "vpce-0000000000-1234567.cassandra.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-0000000000-1234567-us-east-1a.cassandra.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "cassandra.us-east-1.amazonaws.com",
                "HostedZoneId": "ZONEIDPENDING"
            },
            {
                "DnsName": "cassandra.us-east-1.api.aws",
                "HostedZoneId": "ZONEIDPENDING"
            }
        ],
        "CreationTimestamp": "2025-09-19T15:19:19.266000+00:00",
        "OwnerId": "111122223333",
        "ServiceRegion": "us-east-1"
    }
}
```

Schritt 4: Berechtigungen für die VPC-Endpunktverbindung konfigurieren

Die Verfahren in diesem Schritt zeigen, wie Regeln und Berechtigungen für die Verwendung des VPC-Endpunkts mit Amazon Keyspaces konfiguriert werden.

Um eine eingehende Regel für den neuen Endpunkt zu konfigurieren, um eingehenden TCP-Verkehr zuzulassen

1. Wählen Sie in der Amazon VPC-Konsole im linken Bereich Endpoints und dann den Endpoint aus, den Sie im vorherigen Schritt erstellt haben.
2. Wählen Sie Sicherheitsgruppen verwalten und wählen Sie dann die Sicherheitsgruppe aus, die diesem Endpunkt zugeordnet ist.

3. Wählen Sie Eingehende Regeln und dann Regeln für eingehenden Datenverkehr bearbeiten aus.
4. Fügen Sie eine Regel für eingehenden Datenverkehr mit dem Typ CQLSH/CASSANDRA hinzu. Dadurch wird der Portbereich automatisch auf 9142 festgelegt.
5. Um die neue Regel für eingehenden Datenverkehr zu speichern, wählen Sie Regeln speichern.

Um IAM-Benutzerberechtigungen zu konfigurieren

1. Vergewissern Sie sich, dass der IAM-Benutzer, der für die Verbindung zu Amazon Keyspaces verwendet wurde, über die entsprechenden Berechtigungen verfügt. In AWS Identity and Access Management (IAM) können Sie die AWS verwaltete Richtlinie verwenden, `AmazonKeyspacesReadOnlyAccess` um dem IAM-Benutzer Lesezugriff auf Amazon Keyspaces zu gewähren.
   a. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
   b. Wählen Sie auf dem IAM-Konsolen-Dashboard Users (Benutzer) und anschließend Ihren IAM-Benutzer aus der Liste aus.
   c. Wählen Sie auf der Seite Summary (Übersicht) die Option Add permissions (Berechtigungen hinzufügen) aus.
   d. Wählen Sie Attach existing policies directly (Vorhandene Richtlinien direkt zuordnen).
   e. Wählen Sie AmazonKeyspacesReadOnlyAccessin der Liste der Richtlinien die Option und anschließend Weiter: Überprüfen aus.
   f. Wählen Sie Add permissions (Berechtigungen hinzufügen) aus.
2. Stellen Sie sicher, dass Sie über den VPC-Endpunkt auf Amazon Keyspaces zugreifen können.

```
aws keyspaces list-tables --keyspace-name 'my_Keyspace'
```

Wenn Sie möchten, können Sie einige andere AWS CLI Befehle für Amazon Keyspaces ausprobieren. Weitere Informationen finden Sie in der AWS CLI -Befehlsreferenz.

> **Note**
>
> Die Mindestberechtigungen, die ein IAM-Benutzer oder eine IAM-Rolle für den Zugriff auf Amazon Keyspaces benötigt, sind Leseberechtigungen für die Systemtabelle, wie in der folgenden Richtlinie dargestellt. Weitere Informationen zu richtlinienbasierten

Berechtigungen finden Sie unter. the section called “Identity-based Beispiele für Richtlinien”

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

3. Gewähren Sie dem IAM-Benutzer Lesezugriff auf die Amazon EC2 EC2-Instance mit der VPC.

   Wenn Sie Amazon Keyspaces mit VPC-Endpunkten verwenden, müssen Sie dem IAM-Benutzer oder der IAM-Rolle, der auf Amazon Keyspaces zugreift, nur Leseberechtigungen für Ihre Amazon EC2 EC2-Instance und die VPC gewähren, um Endpunkt- und Netzwerkschnittstellendaten zu sammeln. Amazon Keyspaces speichert diese Informationen in der `system.peers` Tabelle und verwendet sie zur Verwaltung von Verbindungen.

   > **Note**
   >
   > Die verwalteten Richtlinien `AmazonKeyspacesReadOnlyAccess_v2` `AmazonKeyspacesFullAccess` enthalten die erforderlichen Berechtigungen, damit Amazon Keyspaces auf die Amazon EC2 EC2-Instance zugreifen kann, um Informationen über verfügbare Schnittstellen-VPC-Endpunkte zu lesen.

   a. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/

   b. Wählen Sie im Dashboard der IAM-Konsole die Option Richtlinien aus.

   c. Wählen Sie Richtlinie erstellen und dann die Registerkarte JSON aus.

d. Kopieren Sie die folgende Richtlinie und wählen Sie Weiter: Tags.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource": "*"
      }
   ]
}
```

e. Wählen Sie Weiter: Überprüfen, geben Sie den Namen `keyspacesVPCendpoint` für die Richtlinie ein und wählen Sie Richtlinie erstellen aus.

f. Wählen Sie auf dem IAM-Konsolen-Dashboard Users (Benutzer) und anschließend Ihren IAM-Benutzer aus der Liste aus.

g. Wählen Sie auf der Seite Summary (Übersicht) die Option Add permissions (Berechtigungen hinzufügen) aus.

h. Wählen Sie Attach existing policies directly (Vorhandene Richtlinien direkt zuordnen).

i. Wählen Sie in der Liste der Richtlinien KeySpaceSvpcEndpoint und dann Weiter: Überprüfen aus.

j. Wählen Sie Add permissions (Berechtigungen hinzufügen) aus.

4. Um zu überprüfen, ob die Amazon `system.peers` Keyspaces-Tabelle mit VPC-Informationen aktualisiert wird, führen Sie die folgende Abfrage von Ihrer Amazon EC2 EC2-Instance aus mit. `cqlsh` Wenn Sie in Schritt 2 noch nicht `cqlsh` auf Ihrer Amazon EC2 EC2-Instance installiert haben, folgen Sie den Anweisungen unterthe section called “Verwendung der `cqlsh-expansion`”.

```
SELECT * FROM system.peers;
```

Die Ausgabe gibt Knoten mit privaten IPv6-IP-Adressen zurück, abhängig von Ihrer VPC- und Subnetzkonfiguration in Ihrer Region. AWS

```
 peer                                    | data_center | host_id
           | preferred_ip                                  | rack      | release_version
 | rpc_address                             | schema_version                       |
 tokens
-----------------------------------------+-------------
+--------------------------------------+----------------------------------------
+-----------+-----------------+--------------------------------------
+-------------------------------------
+-------------------------------------------
  2600:1111:2222:3333:283b:8e6:d04f      |   us-east-1 | ddddddd-7a22-3582-
a73d-49338a686a53 |  2600:1111:2222:3333:283b:8e6:d04f      | us-east-1 |
    3.11.2 |  2600:1111:2222:3333:283b:8e6:d04f      | 05deae2d-6405-494d-a965-
c0e5836bcb3c |  {'850705917302346158658436518579420528 63'}
 2600:1111:2222:4444:7d26:5a09:1b44      |   us-east-1 | 66666666-035d-37ef-
a247-19a6a867ab09 | 2600:1111:2222:4444:7d26:5a09:1b44      | us-east-1 |
    3.11.2 | 2600:1111:2222:4444:7d26:5a09:1b44      | 05deae2d-6405-494d-a965-
c0e5836bcb3c | {'170141183460469231731687303715884105726'}
```

> **Note**
>
> Sie müssen eine `cqlsh` Verbindung zu Amazon Keyspaces verwenden, um zu bestätigen, dass Ihr VPC-Endpunkt korrekt konfiguriert wurde. Wenn Sie Ihre lokale Umgebung oder den Amazon Keyspaces CQL-Editor in der verwenden AWS-Managementkonsole, geht die Verbindung automatisch über den öffentlichen Endpunkt statt über Ihren VPC-Endpunkt. Wenn Sie neun IP-Adressen sehen, sind dies die Einträge, die Amazon Keyspaces automatisch in die `system.peers` Tabelle für öffentliche Endpunktverbindungen schreibt.

Schritt 5: Konfigurieren Sie die Überwachung mit CloudWatch

In diesem Schritt erfahren Sie, wie Sie Amazon verwenden CloudWatch , um die VPC-Endpunktverbindung zu Amazon Keyspaces zu überwachen.

AWS PrivateLink veröffentlicht Datenpunkte zu CloudWatch etwa Ihren Schnittstellen-Endpunkten. Mit den Metriken können Sie überprüfen, ob Ihr System die erwartete Leistung zeigt. Der `AWS/PrivateLinkEndpoints` Namespace in CloudWatch enthält die Metriken für

Schnittstellenendpunkte. Weitere Informationen finden Sie im Handbuch unter CloudWatch AWS PrivateLink Metriken für.AWS PrivateLink

So erstellen Sie ein CloudWatch Dashboard mit VPC-Endpunktmetriken

1. Öffnen Sie die CloudWatch Konsole unter. https://console.aws.amazon.com/cloudwatch/
2. Wählen Sie im Navigationsbereich Dashboards aus. Wählen Sie dann Dashboard erstellen. Geben Sie einen Namen für das Dashboard ein und wählen Sie Erstellen.
3. Wählen Sie unter Widget hinzufügen die Option Zahl aus.
4. Wählen Sie unter Metriken die Option aus AWS/PrivateLinkEndpoints.
5. Wählen Sie Endpunkttyp, Dienstname, VPC-Endpunkt-ID, VPC-ID aus.
6. Wählen Sie die Metriken `ActiveConnections` und `NewConnections` dann Widget erstellen aus.
7. Speichern Sie das Dashboard.

Die `ActiveConnections` Metrik ist definiert als die Anzahl der gleichzeitigen aktiven Verbindungen, die der Endpunkt in der letzten Minute empfangen hat. Die `NewConnections` Metrik ist definiert als die Anzahl der neuen Verbindungen, die in der letzten Minute über den Endpunkt hergestellt wurden.

Weitere Informationen zum Erstellen von Dashboards finden Sie unter Dashboard erstellen im CloudWatch Benutzerhandbuch.

Schritt 6: (Optional) Bewährte Methoden zur Konfiguration der Größe des Verbindungspools für Ihre Anwendung

In diesem Abschnitt wird beschrieben, wie Sie die ideale Größe des Verbindungspools auf der Grundlage der Anforderungen an den Abfragedurchsatz Ihrer Anwendung ermitteln können.

Amazon Keyspaces erlaubt maximal 3.000 CQL-Abfragen pro Sekunde pro TCP-Verbindung. Die Anzahl der Verbindungen, die ein Treiber mit Amazon Keyspaces herstellen kann, ist also praktisch unbegrenzt. Wir empfehlen Ihnen jedoch, die Größe des Verbindungspools an die Anforderungen Ihrer Anwendung anzupassen und die verfügbaren Endpunkte zu berücksichtigen, wenn Sie Amazon Keyspaces mit VPC-Endpunktverbindungen verwenden.

Sie konfigurieren die Größe des Verbindungspools im Client-Treiber. Basierend auf einer lokalen Poolgröße von 2 und einem VPC-Schnittstellenendpunkt, der über 3 Availability Zones erstellt wurde,

richtet der Treiber beispielsweise 6 Verbindungen für Abfragen ein (insgesamt 7, einschließlich einer Kontrollverbindung). Mit diesen 6 Verbindungen können Sie maximal 18.000 CQL-Abfragen pro Sekunde unterstützen.

Wenn Ihre Anwendung 40.000 CQL-Abfragen pro Sekunde unterstützen muss, gehen Sie von der Anzahl der Abfragen aus, die erforderlich sind, um die erforderliche Größe des Verbindungspools zu ermitteln. Um 40.000 CQL-Abfragen pro Sekunde zu unterstützen, müssen Sie die Größe des lokalen Pools auf mindestens 5 konfigurieren, was mindestens 45.000 CQL-Abfragen pro Sekunde unterstützt.

Mithilfe der `PerConnectionRequestRateExceeded` CloudWatch Metrik im Namespace können Sie überwachen, ob Sie das Kontingent für die maximale Anzahl von Vorgängen pro Sekunde und Verbindung überschreiten. `AWS/Cassandra` Die `PerConnectionRequestRateExceeded` Metrik zeigt die Anzahl der Anfragen an Amazon Keyspaces, die das Kontingent für die Anforderungsrate pro Verbindung überschreiten.

Die Codebeispiele in diesem Schritt zeigen, wie Sie das Verbindungspooling schätzen und konfigurieren, wenn Sie VPC-Endpunkte mit Schnittstellen verwenden.

Java

Sie können die Anzahl der Verbindungen pro Pool im Java-Treiber konfigurieren. Ein vollständiges Beispiel für eine Verbindung mit einem Java-Clienttreiber finden Sie unterthe section called “Verwenden eines Cassandra-Java-Client-Treibers”.

Wenn der Client-Treiber gestartet wird, wird zunächst die Steuerverbindung für Verwaltungsaufgaben hergestellt, z. B. für Schema- und Topologieänderungen. Dann werden die zusätzlichen Verbindungen erstellt.

Im folgenden Beispiel ist die Treiberkonfiguration für die Größe des lokalen Pools mit 2 angegeben. Wenn der VPC-Endpunkt über 3 Subnetze innerhalb der VPC erstellt wird, ergibt dies 7 `NewConnections` in CloudWatch für den Schnittstellenendpunkt, wie in der folgenden Formel dargestellt.

```
NewConnections = 3 (VPC subnet endpoints created across) * 2 (pool size) + 1
 ( control connection)
```

```
datastax-java-driver {
```

```
    basic.contact-points = [ "cassandra.us-east-1.amazonaws.com:9142"]
    advanced.auth-provider{
        class = PlainTextAuthProvider
           username = "ServiceUserName"
           password = "ServicePassword"
    }
    basic.load-balancing-policy {
        local-datacenter = "us-east-1"
        slow-replica-avoidance = false
    }

    advanced.ssl-engine-factory {
        class = DefaultSslEngineFactory
        truststore-path = "./src/main/resources/cassandra_truststore.jks"
        truststore-password = "my_password"
        hostname-validation = false
      }
    advanced.connection {
        pool.local.size = 2
          }
}
```

Wenn die Anzahl der aktiven Verbindungen nicht Ihrer konfigurierten Poolgröße (Aggregation über Subnetze) +1 Kontrollverbindung entspricht, verhindert etwas, dass die Verbindungen erstellt werden.

Node.js

Sie können die Anzahl der Verbindungen pro Pool im Node.js Treiber konfigurieren. Ein vollständiges Beispiel für eine Node.js Client-Treiberverbindung finden Sie unterthe section called “Verwenden eines Cassandra-Client-Treibers Node.js ”.

Im folgenden Codebeispiel wird die *keyspaces-bundle.pem* Datei für das digitale Zertifikat verwendet. Weitere Informationen finden Sie unter the section called “Bevor Sie beginnen”.

Für das folgende Codebeispiel ist die Treiberkonfiguration für die Größe des lokalen Pools als 1 angegeben. Wenn der VPC-Endpunkt über 4 Subnetze innerhalb der VPC erstellt wird, ergibt dies 5 `NewConnections` in CloudWatch für den Schnittstellenendpunkt, wie in der folgenden Formel dargestellt.

```
NewConnections = 4 (VPC subnet endpoints created across) * 1 (pool size) + 1
 ( control connection)
```

```
const cassandra = require('cassandra-driver');
const fs = require('fs');
const types = cassandra.types;
const auth = new cassandra.auth.PlainTextAuthProvider('ServiceUserName',
 'ServicePassword');
const sslOptions1 = {
         ca: [
                    fs.readFileSync('/home/ec2-user/keyspaces-bundle.pem',
 'utf-8')],
                    host: 'cassandra.us-east-1.amazonaws.com',
                    rejectUnauthorized: true
        };
const client = new cassandra.Client({
                   contactPoints: ['cassandra.us-east-1.amazonaws.com'],
                   localDataCenter: 'us-east-1',
                   pooling: { coreConnectionsPerHost: { [types.distance.local]:
 1 } },
                   consistency: types.consistencies.localQuorum,
                   queryOptions: { isIdempotent: true },
                   authProvider: auth,
                   sslOptions: sslOptions1,
                   protocolOptions: { port: 9142 }
        });
```

Schritt 7: (Optional) Aufräumen

Gehen Sie wie folgt vor, wenn Sie die Ressourcen löschen möchten, die Sie in diesem Tutorial erstellt haben.

So entfernen Sie Ihren VPC-Endpunkt für Amazon Keyspaces

1. Melden Sie sich bei Ihrer Amazon-EC2-Instance an.
2. Ermitteln Sie die VPC-Endpunkt-ID, die für Amazon Keyspaces verwendet wird. Wenn Sie die `grep` Parameter weglassen, werden VPC-Endpunktinformationen für alle Dienste angezeigt.

```
aws ec2 describe-vpc-endpoint-services | grep ServiceName | grep cassandra

{
    "VpcEndpoint": {
        "PolicyDocument": "{\"Version\":\"2000-00-00",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*\"}]}",
```

```
        "VpcId": "vpc-a1234bcd",
        "State": "available",
        "ServiceName": "com.amazonaws.us-east-1.cassandra",
        "RouteTableIds": [],
        "VpcEndpointId": "vpce-1a23b4c5",
        "CreationTimestamp": "2025-07-26T22:00:14Z"
    }
}
```

In der Beispielausgabe ist die ID des VPC-Endpunkts *vpce-1a23b4c5*. Achten Sie darauf, diesen Wert durch Ihren eigenen zu ersetzen.

3. Löschen Sie den VPC-Endpunkt.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-1a23b4c5

{
    "Unsuccessful": []
}
```

Das leere Array `[]` zeigt den Erfolg an (keine fehlgeschlagenen Anforderungen).

So beenden Sie Ihre Amazon-EC2-Instance

1. Öffnen Sie die Amazon-EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie im Navigationsbereich Instances aus.
3. Wählen Sie Ihre Amazon EC2-Instance aus.
4. Wählen Sie „Aktionen“, „Instanzstatus“ und anschließend „Beenden“.
5. Wählen Sie im Bestätigungsfenster Yes, Terminate aus.

Tutorial: Integrieren Sie Apache Spark, um Daten zu importieren oder zu exportieren

Apache Spark ist eine Open-Source-Engine für umfangreiche Datenanalysen. Mit Apache Spark können Sie Analysen von Daten, die in Amazon Keyspaces gespeichert sind, effizienter durchführen. Sie können Amazon Keyspaces auch verwenden, um Anwendungen einen konsistenten, einstelligen Millisekunden-Lesezugriff auf Analysedaten von Spark zu ermöglichen. Der Open-Source-Spark

Cassandra Connector vereinfacht das Lesen und Schreiben von Daten zwischen Amazon Keyspaces und Spark.

Die Unterstützung von Amazon Keyspaces für den Spark Cassandra Connector optimiert die Ausführung von Cassandra-Workloads in Spark-based Analyse-Pipelines mithilfe eines vollständig verwalteten und serverlosen Datenbankdienstes. Mit Amazon Keyspaces müssen Sie sich keine Sorgen machen, dass Spark um dieselben zugrunde liegenden Infrastrukturressourcen wie Ihre Tabellen konkurriert. Amazon Keyspaces-Tabellen werden basierend auf Ihrem Anwendungsdatenverkehr automatisch nach oben und unten skaliert.

Das folgende Tutorial führt Sie durch die Schritte und bewährten Methoden, die zum Lesen und Schreiben von Daten in Amazon Keyspaces mit dem Spark Cassandra Connector erforderlich sind. Das Tutorial zeigt, wie Daten zu Amazon Keyspaces migriert werden, indem Daten aus einer Datei mit dem Spark Cassandra Connector geladen und in eine Amazon Keyspaces-Tabelle geschrieben werden. Anschließend zeigt das Tutorial, wie Sie die Daten mit dem Spark Cassandra Connector aus Amazon Keyspaces zurücklesen. Sie würden dies tun, um Cassandra-Workloads in Analytics-Pipelines auszuführen. Spark-based

Topics

- Voraussetzungen für den Aufbau von Verbindungen zu Amazon Keyspaces mit dem Spark Cassandra Connector
- Schritt 1: Amazon Keyspaces für die Integration mit dem Apache Cassandra Spark Connector konfigurieren
- Schritt 2: Konfigurieren Sie den Apache Cassandra Spark Connector
- Schritt 3: Erstellen Sie die Anwendungskonfigurationsdatei
- Schritt 4: Bereiten Sie die Quelldaten und die Zieltabelle in Amazon Keyspaces vor
- Schritt 5: Amazon Keyspaces-Daten mit dem Apache Cassandra Spark Connector schreiben und lesen
- Behebung häufiger Fehler bei der Verwendung des Spark Cassandra Connectors mit Amazon Keyspaces

Voraussetzungen für den Aufbau von Verbindungen zu Amazon Keyspaces mit dem Spark Cassandra Connector

Bevor Sie mit dem Spark Cassandra Connector eine Verbindung zu Amazon Keyspaces herstellen, müssen Sie sicherstellen, dass Sie Folgendes installiert haben. Die Kompatibilität von Amazon

Keyspaces mit dem Spark Cassandra Connector wurde mit den folgenden empfohlenen Versionen getestet:

- Java-Version 8
- Maßstab 2.12
- Spark 3.4
- Cassandra Connector 2.5 und höher
- Cassandra-Treiber 4.12

1. Um Scala zu installieren, folgen Sie den Anweisungen unter. https://www.scala-lang.org/download/scala2.html
2. Folgen Sie diesem Beispiel, um Spark 3.4.1 zu installieren.

```
curl -o spark-3.4.1-bin-hadoop3.tgz -k https://dlcdn.apache.org/spark/spark-3.4.1/
spark-3.4.1-bin-hadoop3.tgz

# now to untar
tar -zxvf spark-3.4.1-bin-hadoop3.tgz

# set this variable.
export SPARK_HOME=$PWD/spark-3.4.1-bin-hadoop3
```

Schritt 1: Amazon Keyspaces für die Integration mit dem Apache Cassandra Spark Connector konfigurieren

In diesem Schritt bestätigen Sie, dass der Partitionierer für Ihr Konto mit dem Apache Spark Connector kompatibel ist, und richten die erforderlichen IAM-Berechtigungen ein. Die folgenden bewährten Methoden helfen Ihnen dabei, ausreichend read/write Kapazität für die Tabelle bereitzustellen.

1. Vergewissern Sie sich, dass der `Murmur3Partitioner` Partitionierer der Standardpartitionierer für Ihr Konto ist. Dieser Partitionierer ist mit dem Spark Cassandra Connector kompatibel. Weitere Informationen zu Partitionierern und deren Änderung finden Sie unter. the section called “Mit Partitionierern arbeiten”

2. Richten Sie Ihre IAM-Berechtigungen für Amazon Keyspaces mithilfe von Schnittstellen-VPC-Endpunkten mit Apache Spark ein.

   - Weisen Sie read/write Zugriff auf die Benutzertabelle und Lesezugriff auf die Systemtabellen zu, wie im unten aufgeführten Beispiel für eine IAM-Richtlinie dargestellt.
   - Das Auffüllen der system.peers-Tabelle mit Ihren verfügbaren Schnittstellen-VPC-Endpunkten ist für Clients erforderlich, die mit Spark über VPC-Endpunkte auf Amazon Keyspaces zugreifen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      },
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

3. Beachten Sie die folgenden bewährten Methoden, um eine ausreichende read/write Durchsatzkapazität für Ihre Amazon Keyspaces-Tabelle zu konfigurieren, um den Datenverkehr vom Spark Cassandra Connector zu unterstützen.

   - Beginnen Sie mit der Nutzung von On-Demand-Kapazität, um das Szenario zu testen.

- Um die Kosten des Tabellendurchsatzes für Produktionsumgebungen zu optimieren, verwenden Sie einen Ratenbegrenzer für den Datenverkehr vom Connector und konfigurieren Sie Ihre Tabelle so, dass die bereitgestellte Kapazität mit automatischer Skalierung genutzt wird. Weitere Informationen finden Sie unter the section called “Verwalten Sie die Durchsatzkapazität mit Auto Scaling”.
- Sie können einen festen Ratenbegrenzer verwenden, der im Lieferumfang des Cassandra-Treibers enthalten ist. Im AWS Samples-Repo gibt es einige auf Amazon Keyspaces zugeschnittene Ratenbegrenzer.
- Weitere Informationen zur Kapazitätsverwaltung finden Sie unter. the section called “ read/write Kapazitätsmodi konfigurieren”

Schritt 2: Konfigurieren Sie den Apache Cassandra Spark Connector

Apache Spark ist eine Allzweck-Rechenplattform, die Sie auf verschiedene Arten konfigurieren können. Um Spark und den Spark Cassandra Connector für die Integration mit Amazon Keyspaces zu konfigurieren, empfehlen wir, mit den im folgenden Abschnitt beschriebenen Mindestkonfigurationseinstellungen zu beginnen und diese später entsprechend Ihrer Arbeitslast zu erhöhen.

- Erstellen Sie Spark-Partitionen mit einer Größe von weniger als 8 MB.

  In Spark stellen Partitionen einen atomaren Datenblock dar, der parallel ausgeführt werden kann. Wenn Sie mit dem Spark Cassandra Connector Daten in Amazon Keyspaces schreiben, gilt: Je kleiner die Spark-Partition, desto geringer ist die Anzahl der Datensätze, die die Aufgabe schreiben wird. Wenn eine Spark-Aufgabe auf mehrere Fehler stößt, schlägt sie fehl, nachdem die festgelegte Anzahl von Wiederholungsversuchen ausgeschöpft ist. Um zu vermeiden, dass große Aufgaben wiederholt und viele Daten erneut verarbeitet werden, sollten Sie die Größe der Spark-Partition klein halten.
- Verwenden Sie eine geringe Anzahl gleichzeitiger Schreibvorgänge pro Executor mit einer großen Anzahl von Wiederholungsversuchen.

  Amazon Keyspaces gibt Fehler mit unzureichender Kapazität als Betriebs-Timeouts an Cassandra-Treibern zurück. Sie können Timeouts, die durch unzureichende Kapazität verursacht wurden, nicht beheben, indem Sie die konfigurierte Timeout-Dauer ändern, da der Spark Cassandra Connector versucht, Anfragen transparent mit dem zu wiederholen. `MultipleRetryPolicy` Um sicherzustellen, dass Wiederholungsversuche den Verbindungspool des Treibers nicht überlasten,

sollten Sie eine geringe Anzahl gleichzeitiger Schreibvorgänge pro Executor mit einer großen Anzahl von Wiederholungen verwenden. Der folgende Codeausschnitt ist ein Beispiel dafür.

```
spark.cassandra.query.retry.count = 500
spark.cassandra.output.concurrent.writes = 3
```

- Teilen Sie den Gesamtdurchsatz auf und verteilen Sie ihn auf mehrere Cassandra-Sitzungen.
  - Der Cassandra Spark Connector erstellt eine Sitzung für jeden Spark-Executor. Stellen Sie sich diese Sitzung als Maßstabseinheit zur Bestimmung des erforderlichen Durchsatzes und der Anzahl der erforderlichen Verbindungen vor.
  - Wenn Sie die Anzahl der Kerne pro Executor und die Anzahl der Kerne pro Task definieren, beginnen Sie niedrig und erhöhen Sie sie nach Bedarf.
  - Stellen Sie Spark-Aufgabenfehler so ein, dass die Verarbeitung bei vorübergehenden Fehlern möglich ist. Nachdem Sie sich mit den Verkehrseigenschaften und Anforderungen Ihrer Anwendung vertraut gemacht haben, empfehlen wir, einen begrenzten Wert festzulegen`spark.task.maxFailures`.
  - Die folgende Konfiguration kann beispielsweise zwei gleichzeitige Aufgaben pro Executor und Sitzung verarbeiten:

    ```
    spark.executor.instances = configurable -> number of executors for the session.
    spark.executor.cores = 2 -> Number of cores per executor.
    spark.task.cpus = 1 -> Number of cores per task.
    spark.task.maxFailures = -1
    ```

- Schalten Sie die Batchverarbeitung aus.
  - Wir empfehlen, die Batchverarbeitung zu deaktivieren, um die Direktzugriffsmuster zu verbessern. Der folgende Codeausschnitt ist ein Beispiel dafür.

    ```
    spark.cassandra.output.batch.size.rows = 1 (Default = None)
    spark.cassandra.output.batch.grouping.key = none (Default = Partition)
    spark.cassandra.output.batch.grouping.buffer.size = 100 (Default = 1000)
    ```

- Stellen **`SPARK_LOCAL_DIRS`** Sie eine schnelle, lokale Festplatte mit ausreichend Speicherplatz ein.
  - Standardmäßig speichert Spark Kartenausgabedateien und Resilient Distributed Datasets (RDDs) in einem `/tmp` Ordner. Abhängig von der Konfiguration Ihres Spark-Hosts kann dies dazu führen, dass auf dem Gerät kein Platz mehr übrig ist.

- Um die `SPARK_LOCAL_DIRS` Umgebungsvariable auf ein Verzeichnis mit dem Namen festzulegen`/example/spark-dir`, können Sie den folgenden Befehl verwenden.

```
export SPARK_LOCAL_DIRS=/example/spark-dir
```

Schritt 3: Erstellen Sie die Anwendungskonfigurationsdatei

Um den Open-Source-Spark Cassandra Connector mit Amazon Keyspaces zu verwenden, müssen Sie eine Anwendungskonfigurationsdatei bereitstellen, die die Einstellungen enthält, die für die Verbindung mit dem DataStax Java-Treiber erforderlich sind. Sie können entweder dienstspezifische Anmeldeinformationen oder das SigV4-Plugin verwenden, um eine Verbindung herzustellen.

Falls Sie dies noch nicht getan haben, müssen Sie das digitale Zertifikat, mit dem die TLS-Verbindung hergestellt wurde, in eine TrustStore-Datei konvertieren. Die detaillierten Schritte finden Sie im the section called “Bevor Sie beginnen” Tutorial zur Java-Treiberverbindung. Notieren Sie sich den TrustStore-Dateipfad und das Passwort, da Sie diese Informationen benötigen, wenn Sie die Anwendungskonfigurationsdatei erstellen.

Connect mit SigV4-Authentifizierung her

Dieser Abschnitt zeigt Ihnen eine `application.conf` Beispieldatei, die Sie verwenden können, wenn Sie eine Verbindung mit AWS Anmeldeinformationen und dem SigV4-Plugin herstellen. Falls Sie dies noch nicht getan haben, müssen Sie Ihre IAM-Zugriffsschlüssel (eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel) generieren und sie in Ihrer AWS Konfigurationsdatei oder als Umgebungsvariablen speichern. Detaillierte Anweisungen finden Sie unter the section called “Erforderliche Anmeldeinformationen für die Authentifizierung AWS ”.

Ersetzen Sie im folgenden Beispiel den Dateipfad zu Ihrer TrustStore-Datei und ersetzen Sie das Passwort.

```
datastax-java-driver {
        basic.contact-points = ["cassandra.us-east-1.amazonaws.com:9142"]
        basic.load-balancing-policy {
            class = DefaultLoadBalancingPolicy
            local-datacenter = us-east-1
            slow-replica-avoidance = false
        }
        basic.request {
              consistency = LOCAL_QUORUM
```

```
        }
        advanced {
                auth-provider = {
                   class = software.aws.mcs.auth.SigV4AuthProvider
                   aws-region = us-east-1
                 }
            ssl-engine-factory {
                class = DefaultSslEngineFactory
                truststore-path = "path_to_file/cassandra_truststore.jks"
                truststore-password = "password"
        hostname-validation=false
            }
   }
        advanced.connection.pool.local.size = 3
}
```

Aktualisieren und speichern Sie diese Konfigurationsdatei unter. `/home/user1/application.conf` Die folgenden Beispiele verwenden diesen Pfad.

Connect mit dienstspezifischen Anmeldeinformationen her

In diesem Abschnitt finden Sie eine `application.conf` Beispieldatei, die Sie verwenden können, wenn Sie eine Verbindung mit dienstspezifischen Anmeldeinformationen herstellen. Falls Sie dies noch nicht getan haben, müssen Sie dienstspezifische Anmeldeinformationen für Amazon Keyspaces generieren. Detaillierte Anweisungen finden Sie unter the section called “Erstellen Sie dienstspezifische Anmeldeinformationen”.

Ersetzen Sie im folgenden Beispiel `username` und `password` durch Ihre eigenen Anmeldeinformationen. Ersetzen Sie außerdem den Dateipfad zu Ihrer TrustStore-Datei und das Passwort.

```
datastax-java-driver {
        basic.contact-points = ["cassandra.us-east-1.amazonaws.com:9142"]
        basic.load-balancing-policy {
            class = DefaultLoadBalancingPolicy
            local-datacenter = us-east-1
        }
        basic.request {
              consistency = LOCAL_QUORUM
        }
        advanced {
            auth-provider = {
```

```
            class = PlainTextAuthProvider
                    username = "username"
                    password = "password"
                    aws-region = "us-east-1"
            }
            ssl-engine-factory {
                class = DefaultSslEngineFactory
                truststore-path = "path_to_file/cassandra_truststore.jks"
                truststore-password = "password"
                hostname-validation=false
            }
            metadata = {
                schema {
                     token-map.enabled = true
                }
            }
        }
}
```

Aktualisieren und speichern Sie diese Konfigurationsdatei`/home/user1/application.conf`, um sie mit dem Codebeispiel zu verwenden.

Connect dich mit einem festen Tarif

Um eine feste Rate pro Spark-Executor zu erzwingen, können Sie einen Request-Throttler definieren. Dieser Anforderungs-Throttler begrenzt die Rate der Anfragen pro Sekunde. Der Spark Cassandra Connector stellt pro Executor eine Cassandra-Sitzung bereit. Mithilfe der folgenden Formel können Sie einen konsistenten Durchsatz für eine Tabelle erzielen.

```
max-request-per-second * numberOfExecutors = total throughput against a table
```

Sie können dieses Beispiel der Anwendungskonfigurationsdatei hinzufügen, die Sie zuvor erstellt haben.

```
datastax-java-driver {
  advanced.throttler {
    class = RateLimitingRequestThrottler

    max-requests-per-second = 3000
    max-queue-size = 30000
    drain-interval = 1 millisecond
```

```
  }
}
```

Schritt 4: Bereiten Sie die Quelldaten und die Zieltabelle in Amazon Keyspaces vor

In diesem Schritt erstellen Sie eine Quelldatei mit Beispieldaten und einer Amazon Keyspaces-Tabelle.

1. Erstellen Sie die Quelldatei. Sie können eine der folgenden Optionen wählen:
   - Für dieses Tutorial verwenden Sie eine Datei mit kommagetrennten Werten (CSV) mit dem Namen `keyspaces_sample_table.csv` als Quelldatei für die Datenmigration. Die mitgelieferte Beispieldatei enthält einige Datenzeilen für eine Tabelle mit dem Namen. `book_awards`
     - Laden Sie die CSV-Beispieldatei (`keyspaces_sample_table.csv`) herunter, die in der folgenden Archivdatei samplemigration.zip enthalten ist. Entpacken Sie das Archiv und notieren Sie sich den Pfad zu`keyspaces_sample_table.csv`.
   - Wenn Sie Ihre eigene CSV-Datei verwenden möchten, um Daten in Amazon Keyspaces zu schreiben, stellen Sie sicher, dass die Daten randomisiert sind. Daten, die direkt aus einer Datenbank gelesen oder in Flatfiles exportiert werden, werden in der Regel nach Partition und Primärschlüssel sortiert. Der Import bestellter Daten in Amazon Keyspaces kann dazu führen, dass sie in kleinere Segmente von Amazon Keyspaces-Partitionen geschrieben werden, was zu einer ungleichmäßigen Verteilung des Datenverkehrs führt. Dies kann zu einer langsameren Leistung und höheren Fehlerraten führen.

     Im Gegensatz dazu trägt die Randomisierung von Daten dazu bei, die Vorteile der integrierten Lastenausgleichsfunktionen von Amazon Keyspaces zu nutzen, indem der Datenverkehr gleichmäßiger auf die Partitionen verteilt wird. Es gibt verschiedene Tools, die Sie für die Randomisierung von Daten verwenden können. Ein Beispiel, das das Open-Source-Tool Shuf verwendet, finden Sie the section called “Schritt 2: Bereiten Sie die Daten vor” im Tutorial zur Datenmigration. Im Folgenden finden Sie ein Beispiel, das zeigt, wie Daten gemischt werden. `DataFrame`

     ```
     import org.apache.spark.sql.functions.randval
     shuffledDF = dataframe.orderBy(rand())
     ```

2. Erstellen Sie den Zielschlüsselraum und die Zieltabelle in Amazon Keyspaces.

a. Stellen Sie mit dem eine Verbindung zu Amazon Keyspaces her. `cqlsh-expansion` `cqlsh-expansion`Installationsanweisungen finden Sie unterthe section called “Verwendung der `cqlsh-expansion`”.

   Ersetzen Sie den Dienstendpunkt im folgenden Beispiel durch Ihren eigenen Wert.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --ssl
```

b. Erstellen Sie einen neuen Schlüsselraum mit dem Namen, `catalog` wie im folgenden Beispiel gezeigt.

```
CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

c. Wenn der neue Schlüsselraum den Status verfügbar hat, verwenden Sie den folgenden Code, um die Zieltabelle zu erstellen. `book_awards` Weitere Informationen zur asynchronen Ressourcenerstellung und zur Überprüfung, ob eine Ressource verfügbar ist, finden Sie unter. the section called “Überprüfen Sie den Status der Keyspace-Erstellung”

```
CREATE TABLE catalog.book_awards (
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
   PRIMARY KEY ((year, award), category, rank)
   );
```

Schritt 5: Amazon Keyspaces-Daten mit dem Apache Cassandra Spark Connector schreiben und lesen

In diesem Schritt laden Sie zunächst die Daten aus der Beispieldatei `DataFrame` mit dem Spark Cassandra Connector in eine. Als Nächstes schreiben Sie die Daten aus der `DataFrame` in Ihre Amazon Keyspaces-Tabelle. Sie können diesen Teil auch unabhängig verwenden, um beispielsweise Daten in eine Amazon Keyspaces-Tabelle zu migrieren. Schließlich lesen Sie `DataFrame` mit dem Spark Cassandra Connector die Daten aus Ihrer Tabelle in eine. Sie können diesen Teil auch

unabhängig verwenden, um beispielsweise Daten aus einer Amazon Keyspaces-Tabelle zu lesen, um Datenanalysen mit Apache Spark durchzuführen.

1. Starten Sie die Spark-Shell wie im folgenden Beispiel gezeigt. Beachten Sie, dass in diesem Beispiel die SigV4-Authentifizierung verwendet wird.

```
./spark-shell --files application.conf --conf
 spark.cassandra.connection.config.profile.path=application.conf
 --packages software.aws.mcs:aws-sigv4-auth-cassandra-java-driver-
plugin:4.0.5,com.datastax.spark:spark-cassandra-connector_2.12:3.1.0 --conf
 spark.sql.extensions=com.datastax.spark.connector.CassandraSparkExtensions
```

2. Importieren Sie den Spark Cassandra Connector mit dem folgenden Code.

```
import org.apache.spark.sql.cassandra._
```

3. Um Daten aus der CSV-Datei zu lesen und sie in einer zu speichern`DataFrame`, können Sie das folgende Codebeispiel verwenden.

```
var df =
 spark.read.option("header","true").option("inferSchema","true").csv("keyspaces_sample_tabl
```

Sie können das Ergebnis mit dem folgenden Befehl anzeigen.

```
scala> df.show();
```

Die Ausgabe sollte in etwa so aussehen.

```
+----------------+----+-----------+----+-----------------+--------------------
+-------------+
|           award|year|   category|rank|           author|          book_title|
 publisher|
+----------------+----+-----------+----+-----------------+--------------------
+-------------+
|Kwesi Manu Prize|2020|    Fiction|   1|        Akua Mansa|   Where did you go?|
SomePublisher|
|Kwesi Manu Prize|2020|    Fiction|   2|       John Stiles|           Yesterday|
Example Books|
|Kwesi Manu Prize|2020|    Fiction|   3|        Nikki Wolf|Moving to the Cha...|
 AnyPublisher|
```

```
|            Wolf|2020|Non-Fiction|   1|       Wang Xiulan|    History of Ideas|
Example Books|
|            Wolf|2020|Non-Fiction|   2|Ana Carolina Silva|       Science Today|
SomePublisher|
|            Wolf|2020|Non-Fiction|   3| Shirley Rodriguez|The Future of Sea...|
 AnyPublisher|
|     Richard Roe|2020|    Fiction|   1| Alejandro Rosalez|         Long Summer|
SomePublisher|
|     Richard Roe|2020|    Fiction|   2|       Arnav Desai|             The Key|
Example Books|
|     Richard Roe|2020|    Fiction|   3|     Mateo Jackson|    Inside the Whale|
 AnyPublisher|
+----------------+----+-----------+----+------------------+--------------------
+-------------+
```

Sie können das Schema der Daten in der überprüfen, `DataFrame` wie im folgenden Beispiel gezeigt.

```
scala> df.printSchema
```

Die Ausgabe sollte so aussehen.

```
root
|-- award: string (nullable = true)
|-- year: integer (nullable = true)
|-- category: string (nullable = true)
|-- rank: integer (nullable = true)
|-- author: string (nullable = true)
|-- book_title: string (nullable = true)
|-- publisher: string (nullable = true)
```

4. Verwenden Sie den folgenden Befehl, um die Daten in die Amazon Keyspaces-Tabelle `DataFrame` zu schreiben.

```
df.write.cassandraFormat("book_awards", "catalog").mode("APPEND").save()
```

5. Um zu bestätigen, dass die Daten gespeichert wurden, können Sie sie in einen Datenrahmen zurücklesen, wie im folgenden Beispiel gezeigt.

```
var newDf = spark.read.cassandraFormat("book_awards", "catalog").load()
```

Anschließend können Sie die Daten anzeigen, die jetzt im Datenrahmen enthalten sind.

```
scala> newDf.show()
```

Die Ausgabe dieses Befehls sollte so aussehen.

```
+--------------------+-----------------+---------------+----------+-------------
+----+----+
|          book_title|           author|          award|  category|
 publisher|rank|year|
+--------------------+-----------------+---------------+----------+-------------
+----+----+
|         Long Summer| Alejandro Rosalez|     Richard Roe|    Fiction|
SomePublisher|   1|2020|
|    History of Ideas|       Wang Xiulan|           Wolf|Non-Fiction|Example
 Books|   1|2020|
|   Where did you go?|        Akua Mansa|Kwesi Manu Prize|    Fiction|
SomePublisher|   1|2020|
|    Inside the Whale|     Mateo Jackson|     Richard Roe|    Fiction|
 AnyPublisher|   3|2020|
|           Yesterday|       John Stiles|Kwesi Manu Prize|    Fiction|Example
 Books|   2|2020|
|Moving to the Cha...|        Nikki Wolf|Kwesi Manu Prize|    Fiction|
 AnyPublisher|   3|2020|
|The Future of Sea...| Shirley Rodriguez|           Wolf|Non-Fiction|
 AnyPublisher|   3|2020|
|       Science Today|Ana Carolina Silva|           Wolf|Non-Fiction|
SomePublisher|   2|2020|
|             The Key|       Arnav Desai|     Richard Roe|    Fiction|Example
 Books|   2|2020|
+--------------------+-----------------+---------------+----------+-------------
+----+----+
```

Behebung häufiger Fehler bei der Verwendung des Spark Cassandra Connectors mit Amazon Keyspaces

Wenn Sie Amazon Virtual Private Cloud verwenden und eine Verbindung zu Amazon Keyspaces herstellen, werden die häufigsten Fehler bei der Verwendung des Spark-Connectors durch die folgenden Konfigurationsprobleme verursacht.

- Dem in der VPC verwendeten IAM-Benutzer oder der Rolle fehlen die erforderlichen Berechtigungen für den Zugriff auf die `system.peers` Tabelle in Amazon Keyspaces. Weitere Informationen finden Sie unter the section called “Auffüllen von `system.peers` Tabelleneinträgen mit VPC-Endpunktinformationen der Schnittstelle”.
- Dem IAM-Benutzer oder der IAM-Rolle fehlen die erforderlichen read/write Berechtigungen für die Benutzertabelle und der Lesezugriff auf die Systemtabellen in Amazon Keyspaces. Weitere Informationen finden Sie unter the section called “Schritt 1: Amazon Keyspaces konfigurieren”.
- Die Java-Treiberkonfiguration deaktiviert die Überprüfung des Hostnamens beim Herstellen der Verbindung nicht. SSL/TLS Beispiele finden Sie unter the section called “Schritt 2: Konfigurieren Sie den Treiber”.

Ausführliche Schritte zur Fehlerbehebung bei Verbindungen finden Sie unterthe section called “Verbindungsfehler am VPC-Endpunkt”.

Darüber hinaus können Sie CloudWatch Amazon-Metriken verwenden, um Probleme mit Ihrer Spark Cassandra Connector-Konfiguration in Amazon Keyspaces zu beheben. Weitere Informationen zur Verwendung von Amazon Keyspaces mit finden Sie CloudWatch unterthe section called “Überwachung mit CloudWatch”.

Im folgenden Abschnitt werden die nützlichsten Messwerte beschrieben, die Sie bei der Verwendung des Spark Cassandra Connectors beobachten sollten.

PerConnectionRequestRateExceeded

Amazon Keyspaces hat ein Kontingent von 3.000 Anfragen pro Sekunde und Verbindung. Jeder Spark-Executor stellt eine Verbindung mit Amazon Keyspaces her. Wenn Sie mehrere Wiederholungen durchführen, kann Ihr Kontingent für die Anforderungsrate pro Verbindung ausgeschöpft sein. Wenn Sie dieses Kontingent überschreiten, gibt Amazon Keyspaces eine `PerConnectionRequestRateExceeded` Metrik in aus. CloudWatch

Wenn Sie feststellen, dass PerConnectionRequestRateExceeded Ereignisse zusammen mit anderen System- oder Benutzerfehlern auftreten, führt Spark wahrscheinlich mehrere Wiederholungen durch, die über die zugewiesene Anzahl von Anfragen pro Verbindung hinausgehen.

Wenn Sie `PerConnectionRequestRateExceeded` Ereignisse ohne andere Fehler sehen, müssen Sie möglicherweise die Anzahl der Verbindungen in Ihren Treibereinstellungen erhöhen,

um einen höheren Durchsatz zu erzielen, oder Sie müssen möglicherweise die Anzahl der Executoren in Ihrem Spark-Job erhöhen.

StoragePartitionThroughputCapacityExceeded

Amazon Keyspaces hat ein Kontingent von 1.000 WCUs oder WRUs pro second/3 .000 RCUs oder RRUs pro Sekunde und Partition. Wenn Sie `StoragePartitionThroughputCapacityExceeded` CloudWatch Ereignisse sehen, könnte dies darauf hindeuten, dass Daten beim Laden nicht randomisiert werden. Beispiele für das Mischen von Daten finden Sie unter. the section called “Schritt 4: Bereiten Sie die Quelldaten und die Zieltabelle vor”

Häufige Fehler und Warnungen

Wenn Sie Amazon Virtual Private Cloud verwenden und eine Verbindung zu Amazon Keyspaces herstellen, gibt der Cassandra-Treiber möglicherweise eine Warnmeldung über den Kontrollknoten selbst in der `system.peers` Tabelle aus. Weitere Informationen finden Sie unter the section called “Häufige Fehler und Warnungen”. Sie können diese Warnung getrost ignorieren.

Tutorial: Connect über eine containerisierte Anwendung her, die auf Amazon Elastic Kubernetes Service gehostet wird

Dieses Tutorial führt Sie durch die Schritte, die erforderlich sind, um einen Amazon Elastic Kubernetes Service (Amazon EKS) -Cluster einzurichten, der eine containerisierte Anwendung hostet, die über SigV4-Authentifizierung eine Verbindung zu Amazon Keyspaces herstellt.

Amazon EKS ist ein verwalteter Service, der die Installation, den Betrieb und die Wartung Ihrer eigenen Kubernetes-Steuerebene überflüssig macht. Kubernetes ist ein Open-Source-System zur Automatisierung der Bereitstellung, Skalierung und Verwaltung von Anwendungen in Containern.

Das Tutorial bietet eine schrittweise Anleitung zur Konfiguration, Erstellung und Bereitstellung einer containerisierten Java-Anwendung in Amazon EKS. Im letzten Schritt führen Sie die Anwendung aus, um Daten in eine Amazon Keyspaces-Tabelle zu schreiben.

Topics

- Voraussetzungen für die Verbindung von Amazon EKS zu Amazon Keyspaces
- Schritt 1: Den Amazon EKS-Cluster konfigurieren und IAM-Berechtigungen einrichten

- Schritt 2: Konfigurieren Sie die Anwendung
- Schritt 3: Erstellen Sie das Anwendungs-Image und laden Sie die Docker-Datei in Ihr Amazon ECR-Repository hoch
- Schritt 4: Stellen Sie die Anwendung in Amazon EKS bereit und schreiben Sie Daten in Ihre Tabelle
- Schritt 5: (Optional) Aufräumen

Voraussetzungen für die Verbindung von Amazon EKS zu Amazon Keyspaces

Erstellen Sie die folgenden AWS Ressourcen, bevor Sie mit dem Tutorial beginnen können

1. Bevor Sie mit diesem Tutorial beginnen, folgen Sie den Anweisungen zur AWS Einrichtung unterZugreifen auf Amazon Keyspaces (für Apache Cassandra). Zu diesen Schritten gehören die Registrierung AWS und Erstellung eines AWS Identity and Access Management (IAM-) Principals mit Zugriff auf Amazon Keyspaces.
2. Erstellen Sie einen Amazon Keyspaces-Schlüsselraum mit dem Namen `aws` und eine Tabelle mit dem Namen`user`, in die Sie später in diesem Tutorial aus der containerisierten Anwendung, die in Amazon EKS ausgeführt wird, schreiben können. Sie können dies entweder mit oder mithilfe von tun. AWS CLI `cqlsh`

   AWS CLI

   ```
   aws keyspaces create-keyspace --keyspace-name 'aws'
   ```

   Um zu bestätigen, dass der Keyspace erstellt wurde, können Sie den folgenden Befehl verwenden.

   ```
   aws keyspaces list-keyspaces
   ```

   Um die Tabelle zu erstellen, können Sie den folgenden Befehl verwenden.

   ```
   aws keyspaces create-table --keyspace-name 'aws' --table-name 'user' --schema-
   definition 'allColumns=[
               {name=username,type=text}, {name=fname,type=text},
   {name=last_update_date,type=timestamp},{name=lname,type=text}],
   ```

```
            partitionKeys=[{name=username}]'
```

Um zu bestätigen, dass Ihre Tabelle erstellt wurde, können Sie den folgenden Befehl verwenden.

```
aws keyspaces list-tables --keyspace-name 'aws'
```

Weitere Informationen finden Sie unter Create Keyspace und Create Table in der AWS CLI Befehlsreferenz.

cqlsh

```
CREATE KEYSPACE aws WITH replication = {'class': 'SimpleStrategy',
 'replication_factor': '3'}  AND durable_writes = true;
CREATE TABLE aws.user (
    username text PRIMARY KEY,
    fname text,
    last_update_date timestamp,
    lname text
);
```

Um zu überprüfen, ob Ihre Tabelle erstellt wurde, können Sie die folgende Anweisung verwenden.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "aws";
```

Ihre Tabelle sollte in der Ausgabe dieser Anweisung aufgeführt sein. Beachten Sie, dass es zu Verzögerungen kommen kann, bis die Tabelle erstellt wird. Weitere Informationen finden Sie unter the section called “CREATE TABLE”.

3. Erstellen Sie einen Amazon EKS-Cluster mit einem Fargate-Linux-Knotentyp. Fargate ist eine serverlose Compute-Engine, mit der Sie Kubernetes-Pods bereitstellen können, ohne Amazon Amazon EC2 EC2-Instances verwalten zu müssen. Um diesem Tutorial zu folgen, ohne den Clusternamen in allen Beispielbefehlen aktualisieren zu müssen, erstellen Sie einen Cluster mit dem Namen gemäß den `my-eks-cluster` Anweisungen unter Erste Schritte mit Amazon EKS — `eksctl` im Amazon EKS-Benutzerhandbuch. Stellen Sie bei der Erstellung Ihres Clusters sicher, dass Ihre Knoten und die beiden Standard-Pods laufen und fehlerfrei sind. Sie können dies mit dem folgenden Befehl tun.

```
kubectl get pods -A -o wide
```

Sie sollten etwas Ähnliches wie diese Ausgabe sehen.

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE   IP
     NODE                                                NOMINATED NODE   READINESS
 GATES
kube-system   coredns-1234567890-abcde   1/1     Running   0          18m
 192.0.2.0   fargate-ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          18m
 192.0.2.1   fargate-ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
```

4. Docker-Installation. Anweisungen zur Installation von Docker auf einer Amazon EC2 EC2-Instance finden Sie unter Docker installieren im Amazon Elastic Container Registry-Benutzerhandbuch.

   Docker ist auf vielen verschiedenen Betriebssystemen verfügbar, darunter die meisten modernen Linux-Verteilungen wie Ubuntu und sogar macOS und Windows. Weitere Informationen zur Installation von Docker unter einem bestimmten Betriebssystem finden Sie im Docker-Installationshandbuch.

5. Erstellen Sie ein Amazon-ECR-Repository. Amazon ECR ist ein AWS verwalteter Container-Image-Registry-Service, den Sie mit Ihrer bevorzugten CLI verwenden können, um Docker-Images zu übertragen, abzurufen und zu verwalten. Weitere Informationen zu Amazon ECR-Repositorys finden Sie im Amazon Elastic Container Registry User Guide. Sie können den folgenden Befehl verwenden, um ein Repository mit dem Namen zu erstellen. `my-ecr-repository`

   ```
   aws ecr create-repository --repository-name my-ecr-repository
   ```

Nachdem Sie die erforderlichen Schritte abgeschlossen haben, fahren Sie mit fortthe section called “Schritt 1: Den Amazon EKS-Cluster konfigurieren”.

Schritt 1: Den Amazon EKS-Cluster konfigurieren und IAM-Berechtigungen einrichten

Konfigurieren Sie den Amazon EKS-Cluster und erstellen Sie die IAM-Ressourcen, die erforderlich sind, damit ein Amazon EKS-Servicekonto eine Verbindung zu Ihrer Amazon Keyspaces-Tabelle herstellen kann

1. Erstellen Sie einen Open ID Connect (OIDC) -Anbieter für den Amazon EKS-Cluster. Dies ist erforderlich, um IAM-Rollen für Dienstkonten zu verwenden. Weitere Informationen zu OIDC-Anbietern und deren Erstellung finden Sie unter Erstellen eines IAM-OIDC-Anbieters für Ihren Cluster im Amazon EKS-Benutzerhandbuch.

   a. Erstellen Sie einen IAM-OIDC-Identitätsanbieter für Ihren Cluster mit dem folgenden Befehl. In diesem Beispiel wird davon ausgegangen, dass Ihr Clustername lautet. `my-eks-cluster` Wenn Sie einen Cluster mit einem anderen Namen haben, denken Sie daran, den Namen in allen future Befehlen zu aktualisieren.

   ```
   eksctl utils associate-iam-oidc-provider --cluster my-eks-cluster --approve
   ```

   b. Bestätigen Sie mit dem folgenden Befehl, dass der OIDC-Identitätsanbieter bei IAM registriert wurde.

   ```
   aws iam list-open-id-connect-providers --region us-east-1
   ```

   Die Ausgabe sollte in etwa so aussehen. Notieren Sie sich den Amazon Resource Name (ARN) des OIDC. Sie benötigen ihn im nächsten Schritt, wenn Sie eine Vertrauensrichtlinie für das Servicekonto erstellen.

   ```
   {
       "OpenIDConnectProviderList": [
           ..
           {
               "Arn": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
   east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
           }
       ]
   }
   ```

2. Erstellen Sie ein Dienstkonto für den Amazon EKS-Cluster. Dienstkonten bieten eine Identität für Prozesse, die in einem Pod ausgeführt werden. Ein Pod ist das kleinste und einfachste Kubernetes-Objekt, mit dem Sie eine containerisierte Anwendung bereitstellen können. Erstellen Sie als Nächstes eine IAM-Rolle, die das Dienstkonto übernehmen kann, um Berechtigungen für Ressourcen zu erhalten. Sie können auf jeden AWS Dienst von einem Pod aus zugreifen, der für die Verwendung eines Dienstkontos konfiguriert wurde, das eine IAM-Rolle mit Zugriffsberechtigungen für diesen Dienst übernehmen kann.

   a. Erstellen Sie einen neuen Namespace für das Dienstkonto. Ein Namespace hilft dabei, Clusterressourcen zu isolieren, die für dieses Tutorial erstellt wurden. Mit dem folgenden Befehl können Sie einen neuen Namespace erstellen.

   ```
   kubectl create namespace my-eks-namespace
   ```

   b. Um einen benutzerdefinierten Namespace zu verwenden, müssen Sie ihn mit einem Fargate-Profil verknüpfen. Der folgende Code ist ein Beispiel dafür.

   ```
   eksctl create fargateprofile \
       --cluster my-eks-cluster \
       --name my-fargate-profile \
       --namespace my-eks-namespace \
       --labels *=*
   ```

   c. Erstellen Sie mithilfe des folgenden Befehls ein Dienstkonto mit dem Namen `my-eks-serviceaccount` im Namespace `my-eks-namespace` für Ihren Amazon EKS-Cluster.

   ```
   cat >my-serviceaccount.yaml <<EOF
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     name: my-eks-serviceaccount
     namespace: my-eks-namespace
   EOF
   kubectl apply -f my-serviceaccount.yaml
   ```

   d. Führen Sie den folgenden Befehl aus, um eine Vertrauensrichtliniendatei zu erstellen, die die IAM-Rolle anweist, Ihrem Servicekonto zu vertrauen. Diese Vertrauensstellung ist erforderlich, bevor ein Principal eine Rolle übernehmen kann. Sie müssen die folgenden Änderungen an der Datei vornehmen:

- Geben Sie für den `Principal` den ARN ein, den IAM an den `list-open-id-connect-providers` Befehl zurückgegeben hat. Die ARN enthält Ihre Kontonummer und Region.
- Ersetzen Sie in der `condition` Erklärung die AWS-Region und die OIDC-ID.
- Vergewissern Sie sich, dass der Name und der Namespace des Dienstkontos korrekt sind.

Sie müssen die Vertrauensrichtliniendatei im nächsten Schritt anhängen, wenn Sie die IAM-Rolle erstellen.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount",
                    "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
EOF
```

Optional: Sie können den `StringLike` Bedingungen `StringEquals` oder auch mehrere Einträge hinzufügen, damit mehrere Dienstkonten oder Namespaces die Rolle übernehmen können. Informationen dazu, dass Ihr Servicekonto eine IAM-Rolle in einem anderen AWS

Konto übernehmen kann, finden Sie unter Cross-account IAM-Berechtigungen im Amazon EKS-Benutzerhandbuch.

3. Erstellen Sie eine IAM-Rolle mit dem Namen`my-iam-role`, den das Amazon EKS-Servicekonto annehmen soll. Hängen Sie die im letzten Schritt erstellte Vertrauensrichtlinien-Datei an die Rolle an. Die Vertrauensrichtlinie gibt das Dienstkonto und den OIDC-Anbieter an, denen die IAM-Rolle vertrauen kann.

```
aws iam create-role --role-name my-iam-role --assume-role-policy-document file://
trust-relationship.json --description "EKS service account role"
```

4. Weisen Sie Amazon Keyspaces die IAM-Rollenberechtigungen zu, indem Sie eine Zugriffsrichtlinie anhängen.

   a. Fügen Sie eine Zugriffsrichtlinie hinzu, um die Aktionen zu definieren, die die IAM-Rolle für bestimmte Amazon Keyspaces-Ressourcen ausführen kann. Für dieses Tutorial verwenden wir die AWS verwaltete Richtlinie`AmazonKeyspacesFullAccess`, da unsere Anwendung Daten in Ihre Amazon Keyspaces-Tabelle schreibt. Als bewährte Methode wird jedoch empfohlen, benutzerdefinierte Zugriffsrichtlinien zu erstellen, die das Prinzip der geringsten Rechte umsetzen. Weitere Informationen finden Sie unter the section called “So funktioniert Amazon Keyspaces mit IAM”.

```
aws iam attach-role-policy --role-name my-iam-role --policy-
arn=arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess
```

   Bestätigen Sie mit der folgenden Erklärung, dass die Richtlinie erfolgreich an die IAM-Rolle angehängt wurde.

```
aws iam list-attached-role-policies --role-name my-iam-role
```

   Die Ausgabe sollte wie folgt aussehen.

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonKeyspacesFullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess"
        }
    ]
```

```
}
```

b. Versehen Sie das Dienstkonto mit dem Amazon-Ressourcennamen (ARN) der IAM-Rolle, die es annehmen kann. Achten Sie darauf, den Rollen-ARN mit Ihrer Konto-ID zu aktualisieren.

```
kubectl annotate serviceaccount -n my-eks-namespace my-eks-serviceaccount
 eks.amazonaws.com/role-arn=arn:aws:iam::111122223333:role/my-iam-role
```

5. Vergewissern Sie sich, dass die IAM-Rolle und das Dienstkonto korrekt konfiguriert sind.

a. Stellen Sie mit der folgenden Anweisung sicher, dass die Vertrauensrichtlinie der IAM-Rolle korrekt konfiguriert ist.

```
aws iam get-role --role-name my-iam-role --query Role.AssumeRolePolicyDocument
```

Die Ausgabe sollte in etwa so aussehen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount"
                }
            }
        }
    ]
}
```

b. Vergewissern Sie sich, dass das Amazon EKS-Servicekonto mit der IAM-Rolle versehen ist.

```
kubectl describe serviceaccount my-eks-serviceaccount -n my-eks-namespace
```

Die Ausgabe sollte in etwa so aussehen.

```
Name: my-eks-serviceaccount
Namespace:my-eks-namespace
Labels: <none>
Annotations: eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-iam-
role
Image pull secrets: <none>
Mountable secrets: <none>
Tokens: <none>
[...]
```

Nachdem Sie das Amazon EKS-Servicekonto und die IAM-Rolle erstellt und die erforderlichen Beziehungen und Berechtigungen konfiguriert haben, fahren Sie mit fortthe section called “Schritt 2: Konfigurieren Sie die Anwendung”.

Schritt 2: Konfigurieren Sie die Anwendung

In diesem Schritt erstellen Sie Ihre Anwendung, die mithilfe des SigV4-Plug-ins eine Verbindung zu Amazon Keyspaces herstellt. Sie können die Java-Beispielanwendung aus dem Amazon Keyspaces-Beispielcode-Repository auf Github ansehen und herunterladen. Oder Sie können mit Ihrer eigenen Anwendung weitermachen und dabei sicherstellen, dass alle Konfigurationsschritte abgeschlossen sind.

Konfigurieren Sie Ihre Anwendung und fügen Sie die erforderlichen Abhängigkeiten hinzu.

1. Sie können die Java-Beispielanwendung herunterladen, indem Sie das Github-Repository mit dem folgenden Befehl klonen.

   ```
   git clone https://github.com/aws-samples/amazon-keyspaces-examples.git
   ```

2. Nachdem Sie das Github-Repo heruntergeladen haben, entpacken Sie die heruntergeladene Datei und navigieren Sie zum `resources` Verzeichnis der Datei. `application.conf`

   a. Konfiguration der Anwendung

In diesem Schritt konfigurieren Sie das SigV4-Authentifizierungs-Plugin. Sie können das folgende Beispiel in Ihrer Anwendung verwenden. Falls Sie dies noch nicht getan haben, müssen Sie Ihre IAM-Zugriffsschlüssel (eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel) generieren und sie in Ihrer AWS Konfigurationsdatei oder als Umgebungsvariablen speichern. Detaillierte Anweisungen finden Sie unter the section called “Erforderliche Anmeldeinformationen für die Authentifizierung AWS ”. Aktualisieren Sie die AWS Region und den Service-Endpunkt für Amazon Keyspaces nach Bedarf. Weitere Service-Endpunkte finden Sie unter. the section called “Service-Endpunkte” Ersetzen Sie den Truststore-Standort, den Truststore-Namen und das Truststore-Passwort durch Ihr eigenes.

```
datastax-java-driver {
  basic.contact-points = ["cassandra.us-east-1.amazonaws.com:9142"]
  basic.load-balancing-policy.local-datacenter = "us-east-1"
  advanced.auth-provider {
    class = software.aws.mcs.auth.SigV4AuthProvider
    aws-region = "us-east-1"
  }
  advanced.ssl-engine-factory {
    class = DefaultSslEngineFactory
    truststore-path = "truststore_locationtruststore_name.jks"
    truststore-password = "truststore_password;"
  }
}
```

b. Fügen Sie die Abhängigkeit vom STS-Modul hinzu.

Dadurch wird die Möglichkeit hinzugefügt`WebIdentityTokenCredentialsProvider`, eine zu verwenden, die die AWS Anmeldeinformationen zurückgibt, die die Anwendung bereitstellen muss, damit das Dienstkonto die IAM-Rolle übernehmen kann. Sie können dies anhand des folgenden Beispiels tun.

```
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>aws-java-sdk-sts</artifactId>
            <version>1.11.717</version>
        </dependency>
```

c. Fügen Sie die SigV4-Abhängigkeit hinzu.

Dieses Paket implementiert das SigV4-Authentifizierungs-Plugin, das für die Authentifizierung bei Amazon Keyspaces benötigt wird.

```
        <dependency>
            <groupId>software.aws.mcs</groupId>
            <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin</
artifactId>
            <version>4.0.3</version>
        </dependency>
```

3. Fügen Sie eine Protokollierungsabhängigkeit hinzu.

Ohne Protokolle ist die Behebung von Verbindungsproblemen nicht möglich. In diesem Tutorial verwenden `slf4j` wir das Framework für die Protokollierung und speichern `logback.xml` die Protokollausgabe. Wir setzen die Protokollierungsebene auf`debug`, um die Verbindung herzustellen. Sie können das folgende Beispiel verwenden, um die Abhängigkeit hinzuzufügen.

```
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>2.0.5</version>
        </dependency>
```

Sie können den folgenden Codeausschnitt verwenden, um die Protokollierung zu konfigurieren.

```
<configuration>
    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">

        <encoder>
            <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</
pattern>
        </encoder>
    </appender>

    <root level="debug">
        <appender-ref ref="STDOUT" />
    </rootv
</configuration>
```

> **Note**
>
> Die `debug` Ebene wird benötigt, um Verbindungsfehler zu untersuchen. Nachdem Sie von Ihrer Anwendung aus erfolgreich eine Verbindung zu Amazon Keyspaces hergestellt haben, können Sie die Protokollierungsebene nach `warning` Bedarf auf `info` oder ändern.

Schritt 3: Erstellen Sie das Anwendungs-Image und laden Sie die Docker-Datei in Ihr Amazon ECR-Repository hoch

In diesem Schritt kompilieren Sie die Beispielanwendung, erstellen ein Docker-Image und übertragen das Image in Ihr Amazon ECR-Repository.

Erstellen Sie Ihre Anwendung, erstellen Sie ein Docker-Image und reichen Sie es bei Amazon Elastic Container Registry ein

1. Legen Sie Umgebungsvariablen für den Build fest, die Ihre AWS-Region definieren. Ersetzen Sie die Regionen in den Beispielen durch Ihre eigenen.

   ```
   export CASSANDRA_HOST=cassandra.us-east-1.amazonaws.com:9142
   export CASSANDRA_DC=us-east-1
   ```

2. Kompilieren Sie Ihre Anwendung mit Apache Maven Version 3.6.3 oder höher mit dem folgenden Befehl.

   ```
   mvn clean install
   ```

   Dadurch wird eine `JAR` Datei erstellt, in der alle Abhängigkeiten im Verzeichnis enthalten sind. `target`

3. Rufen Sie Ihre ECR-Repository-URI, die für den nächsten Schritt benötigt wird, mit dem folgenden Befehl ab. Stellen Sie sicher, dass Sie die Region auf die Region aktualisieren, die Sie verwendet haben.

   ```
   aws ecr describe-repositories --region us-east-1
   ```

   Die Ausgabe sollte wie im folgenden Beispiel aussehen.

```
"repositories": [
 {
 "repositoryArn": "arn:aws:ecr:us-east-1:111122223333:repository/my-ecr-
repository",
 "registryId": "111122223333",
 "repositoryName": "my-ecr-repository",
 "repositoryUri": "111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository",
 "createdAt": "2023-11-02T03:46:34+00:00",
 "imageTagMutability": "MUTABLE",
 "imageScanningConfiguration": {
 "scanOnPush": false
 },
 "encryptionConfiguration": {
 "encryptionType": "AES256"
 }
 },
```

4. Erstellen Sie das Docker-Image aus dem Stammverzeichnis der Anwendung mit der Repository-URI aus dem letzten Schritt. Ändern Sie die Docker-Datei nach Bedarf. Stellen Sie sicher, dass Sie im Build-Befehl Ihre Konto-ID ersetzen und die Region AWS-Region angeben, in der sich das Amazon ECR-Repository `my-ecr-repository` befindet.

```
docker build -t 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-
repository:latest .
```

5. Rufen Sie ein Authentifizierungstoken ab, um das Docker-Image an Amazon ECR zu übertragen. Sie können dies mit dem folgenden Befehl tun.

```
aws ecr get-login-password --region us-east-1 | docker login --username AWS --
password-stdin 111122223333.dkr.ecr.us-east-1.amazonaws.com
```

6. Suchen Sie zunächst in Ihrem Amazon ECR-Repository nach vorhandenen Bildern. Sie können folgenden Befehl verwenden.

```
aws ecr describe-images --repository-name my-ecr-repository --region us-east-1
```

Dann übertragen Sie das Docker-Image in das Repo. Sie können folgenden Befehl verwenden.

```
docker push 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository:latest
```

Schritt 4: Stellen Sie die Anwendung in Amazon EKS bereit und schreiben Sie Daten in Ihre Tabelle

In diesem Schritt des Tutorials konfigurieren Sie die Amazon EKS-Bereitstellung für Ihre Anwendung und stellen sicher, dass die Anwendung läuft und eine Verbindung zu Amazon Keyspaces herstellen kann.

Um eine Anwendung in Amazon EKS bereitzustellen, müssen Sie alle relevanten Einstellungen in einer Datei mit dem Namen konfigurieren`deployment.yaml`. Diese Datei wird dann von Amazon EKS zur Bereitstellung der Anwendung verwendet. Die Metadaten in der Datei sollten die folgenden Informationen enthalten:

- Anwendungsname Der Name der Anwendung. Für dieses Tutorial verwenden wir`my-keyspaces-app`.
- Kubernetes-Namespace — der Namespace des Amazon EKS-Clusters. Für dieses Tutorial verwenden wir. `my-eks-namespace`
- Name des Amazon EKS-Servicekontos Der Name des Amazon EKS-Servicekontos. Für dieses Tutorial verwenden wir`my-eks-serviceaccount`.
- Bildname der Name des Anwendungsabbilds. Für dieses Tutorial verwenden wir`my-keyspaces-app`.
- Image-URI Die Docker-Image-URI von Amazon ECR.
- AWS Konto-ID Ihre AWS Konto-ID.
- IAM-Rolle ARN Der ARN der IAM-Rolle, die für das Dienstkonto erstellt wurde. Für dieses Tutorial verwenden wir. `my-iam-role`
- AWS-Region des Amazon EKS-Clusters, in dem AWS-Region Sie Ihren Amazon EKS-Cluster erstellt haben.

In diesem Schritt stellen Sie die Anwendung bereit und führen sie aus, die eine Verbindung zu Amazon Keyspaces herstellt und Daten in die Tabelle schreibt.

1. Konfigurieren Sie die `deployment.yaml`-Datei. Sie müssen die folgenden Werte ersetzen:
    - `name`
    - `namespace`
    - `serviceAccountName`

- `image`
- `AWS_ROLE_ARN value`
- Der AWS-Region Eingang `CASSANDRA_HOST`
- `AWS_REGION`

Sie können die folgende Datei als Beispiel verwenden.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-keyspaces-app
  namespace: my-eks-namespace
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-keyspaces-app
  template:
    metadata:
      labels:
        app: my-keyspaces-app
    spec:
      serviceAccountName: my-eks-serviceaccount
      containers:
      - name: my-keyspaces-app
        image: 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-
repository:latest
        ports:
        - containerPort: 8080
        env:
        - name: CASSANDRA_HOST
          value: "cassandra.us-east-1.amazonaws.com:9142"
        - name: CASSANDRA_DC
          value: "us-east-1"
        - name: AWS_WEB_IDENTITY_TOKEN_FILE
          value: /var/run/secrets/eks.amazonaws.com/serviceaccount/token
        - name: AWS_ROLE_ARN
          value: "arn:aws:iam::111122223333:role/my-iam-role"
        - name: AWS_REGION
          value: "us-east-1"
```

2. Stellen Sie `deployment.yaml` bereit.

```
kubectl apply -f deployment.yaml
```

Die Ausgabe sollte so aussehen.

```
deployment.apps/my-keyspaces-app created
```

3. Überprüfen Sie den Status des Pods in Ihrem Namespace des Amazon EKS-Clusters.

```
kubectl get pods -n my-eks-namespace
```

Die Ausgabe sollte diesem Beispiel ähneln.

```
NAME                    READY STATUS RESTARTS AGE
my-keyspaces-app-123abcde4f-g5hij 1/1 Running 0 75s
```

Für weitere Informationen können Sie den folgenden Befehl verwenden.

```
kubectl describe pod my-keyspaces-app-123abcde4f-g5hij -n my-eks-namespace
```

```
Name:                 my-keyspaces-app-123abcde4f-g5hij
Namespace:            my-eks-namespace
Priority:             2000001000
Priority Class Name:  system-node-critical
Service Account:      my-eks-serviceaccount
Node:                 fargate-ip-192-168-102-209.ec2.internal/192.168.102.209
Start Time:           Thu, 23 Nov 2023 12:15:43 +0000
Labels:               app=my-keyspaces-app
                      eks.amazonaws.com/fargate-profile=my-fargate-profile
                      pod-template-hash=6c56fccc56
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
Status:               Running
IP:                   192.168.102.209
IPs:
  IP:           192.168.102.209
Controlled By:  ReplicaSet/my-keyspaces-app-6c56fccc56
```

```
Containers:
  my-keyspaces-app:
    Container ID:
 containerd://41ff7811d33ae4bc398755800abcdc132335d51d74f218ba81da0700a6f8c67b
    Image:          111122223333.dkr.ecr.us-east-1.amazonaws.com/
my_eks_repository:latest
    Image ID:       111122223333.dkr.ecr.us-east-1.amazonaws.com/
my_eks_repository@sha256:fd3c6430fc5251661efce99741c72c1b4b03061474940200d0524b84a951439c
    Port:           8080/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Thu, 23 Nov 2023 12:15:19 +0000
      Finished:     Thu, 23 Nov 2023 12:16:17 +0000
    Ready:          True
    Restart Count:  1
    Environment:
      CASSANDRA_HOST:               cassandra.us-east-1.amazonaws.com:9142
      CASSANDRA_DC:                 us-east-1
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
      AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-iam-role
      AWS_REGION:                   us-east-1
      AWS_STS_REGIONAL_ENDPOINTS:   regional
    Mounts:
      /var/run/secrets/eks.amazonaws.com/serviceaccount from aws-iam-token (ro)
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-fssbf (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  aws-iam-token:
    Type:                     Projected (a volume that contains injected data from
 multiple sources)
    TokenExpirationSeconds:  86400
  kube-api-access-fssbf:
    Type:                     Projected (a volume that contains injected data from
 multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
```

```
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for
 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for
 300s
Events:
  Type     Reason           Age                From               Message
  ----     ------           ----               ----               -------
  Warning  LoggingDisabled  2m13s              fargate-scheduler  Disabled logging
 because aws-logging configmap was not found. configmap "aws-logging" not found
  Normal   Scheduled        89s                fargate-scheduler  Successfully
 assigned my-eks-namespace/my-keyspaces-app-6c56fccc56-mgs2m to fargate-
ip-192-168-102-209.ec2.internal
  Normal   Pulled           75s                kubelet
 Successfully pulled image "111122223333.dkr.ecr.us-east-1.amazonaws.com/
my_eks_repository:latest" in 13.027s (13.027s including waiting)
  Normal   Pulling          54s (x2 over 88s)  kubelet            Pulling image
 "111122223333.dkr.ecr.us-east-1.amazonaws.com/my_eks_repository:latest"
  Normal   Created          54s (x2 over 75s)  kubelet            Created container
 my-keyspaces-app
  Normal   Pulled           54s                kubelet
 Successfully pulled image "111122223333.dkr.ecr.us-east-1.amazonaws.com/
my_eks_repository:latest" in 222ms (222ms including waiting)
  Normal   Started          53s (x2 over 75s)  kubelet            Started container
 my-keyspaces-app
```

4. Überprüfen Sie die Protokolle des Pods, um sicherzustellen, dass Ihre Anwendung läuft und eine Verbindung zu Ihrer Amazon Keyspaces-Tabelle herstellen kann. Sie können dies mit dem folgenden Befehl tun. Achten Sie darauf, den Namen Ihrer Bereitstellung zu ersetzen.

```
kubectl logs -f my-keyspaces-app-123abcde4f-g5hij -n my-eks-namespace
```

Sie sollten in der Lage sein, Anwendungsprotokolleinträge zu sehen, die die Verbindung zu Amazon Keyspaces bestätigen, wie im Beispiel unten.

```
2:47:20.553 [s0-admin-0] DEBUG c.d.o.d.i.c.metadata.MetadataManager
 - [s0] Adding initial contact points [Node(endPoint=cassandra.us-
east-1.amazonaws.com/1.222.333.44:9142, hostId=null, hashCode=e750d92)]
22:47:20.562 [s0-admin-1] DEBUG c.d.o.d.i.c.c.ControlConnection - [s0] Initializing
 with event types [SCHEMA_CHANGE, STATUS_CHANGE, TOPOLOGY_CHANGE]
```

```
22:47:20.564 [s0-admin-1] DEBUG c.d.o.d.i.core.context.EventBus - [s0] Registering
 com.datastax.oss.driver.internal.core.metadata.LoadBalancingPolicyWrapper$$Lambda
$812/0x0000000801105e88@769afb95 for class
 com.datastax.oss.driver.internal.core.metadata.NodeStateEvent
22:47:20.566 [s0-admin-1] DEBUG c.d.o.d.i.c.c.ControlConnection -
 [s0] Trying to establish a connection to Node(endPoint=cassandra.us-
east-1.amazonaws.com/1.222.333.44:9142, hostId=null, hashCode=e750d92)
```

5. Führen Sie die folgende CQL-Abfrage in Ihrer Amazon Keyspaces-Tabelle aus, um zu bestätigen, dass eine Datenzeile in Ihre Tabelle geschrieben wurde:

   ```
   SELECT * from aws.user;
   ```

   Die Ausgabe sollte folgendermaßen aussehen:

   ```
   fname    | lname | username | last_update_date
   ----------+-------+----------+---------------------------
   random    | k     | test     | 2023-12-07 13:58:31.57+0000
   ```

Schritt 5: (Optional) Aufräumen

Gehen Sie wie folgt vor, um alle in diesem Tutorial erstellten Ressourcen zu entfernen.

Entfernen Sie die in diesem Tutorial erstellten Ressourcen

1. Löschen Sie Ihre Bereitstellung. Sie können dazu den folgenden Befehl verwenden.

   ```
   kubectl delete deployment my-keyspaces-app -n my-eks-namespace
   ```

2. Löschen Sie den Amazon EKS-Cluster und alle darin enthaltenen Pods. Dadurch werden auch zugehörige Ressourcen wie das Dienstkonto und der OIDC-Identitätsanbieter gelöscht. Sie können dazu den folgenden Befehl verwenden.

   ```
   eksctl delete cluster --name my-eks-cluster --region us-east-1
   ```

3. Löschen Sie die IAM-Rolle, die für das Amazon EKS-Servicekonto mit Zugriffsberechtigungen für Amazon Keyspaces verwendet wurde. Zunächst müssen Sie die verwaltete Richtlinie entfernen, die der Rolle zugeordnet ist.

```
aws iam detach-role-policy --role-name my-iam-role --policy-arn
 arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess
```

Anschließend können Sie die Rolle mit dem folgenden Befehl löschen.

```
aws iam delete-role --role-name my-iam-role
```

Weitere Informationen finden Sie unter Löschen einer IAM-Rolle (AWS CLI) im IAM-Benutzerhandbuch.

4. Löschen Sie das Amazon ECR-Repository einschließlich aller darin gespeicherten Bilder. Sie können dies mit dem folgenden Befehl tun.

```
aws ecr delete-repository \
      --repository-name my-ecr-repository \
      --force \
      --region us-east-1
```

Beachten Sie, dass das `force` Flag erforderlich ist, um ein Repository zu löschen, das Bilder enthält. Um Ihr Bild zuerst zu löschen, können Sie dies mit dem folgenden Befehl tun.

```
aws ecr batch-delete-image \
      --repository-name my-ecr-repository \
      --image-ids imageTag=latest \
      --region us-east-1
```

Weitere Informationen finden Sie unter Löschen eines Images im Amazon Elastic Container Registry-Benutzerhandbuch.

5. Löschen Sie den Amazon Keyspaces-Schlüsselraum und die Tabelle. Durch das Löschen des Schlüsselraums werden automatisch alle Tabellen in diesem Schlüsselraum gelöscht. Sie können dazu eine der folgenden Optionen verwenden.

   AWS CLI

   ```
   aws keyspaces delete-keyspace --keyspace-name 'aws'
   ```

   Um zu bestätigen, dass der Keyspace gelöscht wurde, können Sie den folgenden Befehl verwenden.

```
aws keyspaces list-keyspaces
```

Um die Tabelle zuerst zu löschen, können Sie den folgenden Befehl verwenden.

```
aws keyspaces delete-table --keyspace-name 'aws' --table-name 'user'
```

Um zu bestätigen, dass Ihre Tabelle gelöscht wurde, können Sie den folgenden Befehl verwenden.

```
aws keyspaces list-tables --keyspace-name 'aws'
```

Weitere Informationen finden Sie unter Delete Keyspace und Delete Table in der AWS CLI Befehlsreferenz.

cqlsh

```
DROP KEYSPACE IF EXISTS "aws";
```

Um zu überprüfen, ob Ihre Schlüsselräume gelöscht wurden, können Sie die folgende Anweisung verwenden.

```
SELECT * FROM system_schema.keyspaces ;
```

Ihr Schlüsselraum sollte in der Ausgabe dieser Anweisung nicht aufgeführt werden. Beachten Sie, dass es zu einer Verzögerung kommen kann, bis die Schlüsselräume gelöscht werden. Weitere Informationen finden Sie unter the section called “SCHLÜSSELRAUM LÖSCHEN”.

Um die Tabelle zuerst zu löschen, können Sie den folgenden Befehl verwenden.

```
DROP TABLE "aws.user"
```

Um zu bestätigen, dass Ihre Tabelle gelöscht wurde, können Sie den folgenden Befehl verwenden.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "aws";
```

Ihre Tabelle sollte in der Ausgabe dieser Anweisung nicht aufgeführt werden. Beachten Sie, dass es zu Verzögerungen kommen kann, bis die Tabelle gelöscht wird. Weitere Informationen finden Sie unter the section called “DROP TABLE”.

Tutorial: Exportieren Sie eine Amazon Keyspaces-Tabelle nach Amazon S3 mit AWS Glue

Dieses Tutorial zeigt Ihnen, wie Sie eine Amazon Keyspaces-Tabelle mithilfe von Amazon S3-Bucket exportieren. AWS Glue Das Tutorial verwendet das `keyspaces-bulk-cli` CLI-Tool, das im Amazon GitHub Keyspaces-Repo verfügbar ist. Mit diesem Tool können Sie Amazon Keyspaces-Daten nach Amazon S3 exportieren, ohne einen Spark-Cluster einrichten zu müssen.

Topics

- Voraussetzungen für den Export von Daten aus Amazon Keyspaces nach Amazon S3
- Schritt 1: Bootstrap der Infrastruktur und AWS Glue jobs
- Schritt 2: Führen Sie den Exportauftrag aus
- Schritt 3: (Optional) Erstellen Sie einen Trigger, um den Exportauftrag zu planen
- Schritt 4: (Optional) Bereinigung

Voraussetzungen für den Export von Daten aus Amazon Keyspaces nach Amazon S3

Bestätigen Sie die folgenden Voraussetzungen und erstellen Sie die Amazon Keyspaces-Ressourcen, bevor Sie mit dem Tutorial beginnen

1. Bevor Sie mit diesem Tutorial beginnen, folgen Sie den Anweisungen zur AWS Einrichtung unterZugreifen auf Amazon Keyspaces (für Apache Cassandra). Zu diesen Schritten gehören die Registrierung AWS und Erstellung eines AWS Identity and Access Management (IAM-) Principal mit Zugriff auf Amazon Keyspaces.
2. Das CLI-Tool in diesem Tutorial verwendet Ihre Anmeldeinformationen und die Standarddaten, die an einem bekannten Ort AWS-Region gespeichert sind. Weitere Informationen finden Sie unter the section called “Verwalten von Zugriffsschlüsseln”.

Das folgende Beispiel zeigt, wie die erforderlichen Werte als Umgebungsvariablen für den Standardbenutzer gespeichert werden.

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-east-1
```

3. Um die `keyspaces-bulk-cli` CLI in diesem Tutorial ausführen zu können, müssen die folgende Software und Tools auf Ihrem Computer installiert sein:

   - Python 3.8 oder höher
   - Java (JDK 11 oder höher)
   - Apache Maven
   - Git
   - curl
   - AWS CLI

   Java, Git und Maven sind für den Bootstrap-Schritt erforderlich, um den Retry Policy Helper herunterzuladen und zu erstellen. Dieses Tutorial wurde mit AWS CLI 2, Python 3.11, Java 17.0.13 und Apache Maven 3.8.7 getestet.

4. Sie benötigen eine Amazon Keyspaces-Tabelle mit Beispieldaten, die Sie später in diesem Tutorial exportieren können. Sie können Ihre eigene Amazon Keyspaces-Tabelle verwenden oder eine Beispieltabelle erstellen, indem Sie den Schritten im Tutorial folgen. Erste Schritte

   a. Um das zu installieren`cqlsh-expansion`, folgen Sie den Schritten unterthe section called “Verwendung der `cqlsh-expansion`”.

   b. Vergewissern Sie sich, dass der `Murmur3Partitioner` Partitionierer der Standardpartitionierer für Ihr Konto ist. Dieser Partitionierer ist kompatibel mit dem Apache Spark Cassandra Connector und mit. AWS Glue Weitere Informationen zu Partitionierern finden Sie unter. the section called “Mit Partitionierern arbeiten”

      Um den Partitionierer Ihres Kontos zu ändern, können Sie die folgende Anweisung verwenden.

      ```
      SELECT partitioner FROM system.local;
      ```

```
UPDATE system.local set
 partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

c. Um einen Amazon Keyspaces-Schlüsselraum zu erstellen, folgen Sie den Schritten unter. the section called “Erstellen Sie einen Schlüsselraum”

d. Um die Amazon Keyspaces-Tabelle zu erstellen, folgen Sie den Schritten unter. the section called “Erstellen einer Tabelle”

e. Um Beispieldaten für den Export nach Amazon S3 in die Tabelle zu laden, folgen Sie den Schritten unterthe section called “Create”.

Nachdem Sie die erforderlichen Schritte abgeschlossen haben, fahren Sie mit fortthe section called “Schritt 1: Bootstrap-Infrastruktur und Jobs AWS Glue ”.

Schritt 1: Bootstrap der Infrastruktur und AWS Glue jobs

In diesem Schritt erstellen und konfigurieren Sie mithilfe der `keyspaces-bulk-cli` CLI alle AWS Ressourcen, die für den automatisierten Datenexport einer Amazon Keyspaces-Tabelle in einen Amazon S3-Bucket erforderlich sind. AWS Glue Der `bootstrap` Befehl führt alle Einrichtungsaufgaben in einem einzigen Schritt aus.

Der `bootstrap` Befehl automatisiert die folgenden Aufgaben.

1. Erstellt einen Amazon S3-Bucket und eine IAM-Servicerolle mithilfe von. CloudFormation
2. Lädt den Apache Spark Cassandra Connector, das SigV4-Authentifizierungs-Plugin und die Apache Spark-Erweiterungen herunter.
3. Lädt den Keyspaces Retry Policy Helper herunter und erstellt ihn mithilfe von Maven.
4. Lädt alle JAR-Dateien und die `keyspaces-application.conf` Konfigurationsdatei in den Amazon S3-Bucket hoch.
5. Stellt drei AWS Glue Jobs bereit: Exportieren (Exportieren einer Tabelle nach Amazon S3), Importieren (Daten aus Amazon S3 in eine Tabelle importieren) und Zählen (Zählen von Zeilen in einer Tabelle).
6. Speichert die Stack-Konfiguration in einer lokalen `.keyspaces-bulk-cli.json` Datei zur Verwendung durch nachfolgende Befehle.

Um die Infrastruktur zu booten und AWS Glue jobs

1. Klonen Sie die Dateien aus dem https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue aws-glue-Repository GitHub auf Ihren lokalen Computer.

```
$ git clone https://github.com/aws-samples/amazon-keyspaces-examples.git
$ cd amazon-keyspaces-examples/scala/datastax-v4/aws-glue
```

2. Führen Sie den Befehl `bootstrap` aus. Das folgende Beispiel verwendet den Standard-Stacknamen `aksglue` und gibt den Schlüsselraum und die Tabelle als Standardwerte für die bereitgestellten Jobs an. Sie können diese Werte überschreiben, wenn Sie einzelne Befehle wie `exportimport`, oder ausführen. `count`

```
$ ./keyspaces-bulk-cli bootstrap --stack aksglue --keyspace catalog --
table book_awards
```

In der folgenden Tabelle werden die optionalen Parameter beschrieben, die Sie zur Anpassung des Bootstraps übergeben können.

Option	Standard	Description
`--stack`	`aksglue`	CloudFormation Präfix für den Stacknamen
`--bucket`	Auto-generated aus Stapel und Konto	Amazon S3-Bucket für Artefakte
`--role-name`	Auto-generated aus dem Stapel	Name der IAM-Dienstrolle
`--keyspace`	`mykeyspace`	Standard-Schlüsselraum für bereitgestellte Jobs
`--table`	`mytable`	Standardtabelle für bereitgestellte Jobs
`--s3-uri`	`s3://{bucket}/export`	Standardmäßiger Amazon S3-Pfad für exportierte Daten
`--format`	`parquet`	Standard-Datenformat

Option	Standard	Description
`--region`	Aus der AWS Konfiguration	AWS-Region
`--profile`	Aus der AWS Konfiguration	AWS benanntes Profil

Um zu bestätigen, dass der von Bootstrap erstellte Amazon S3-Bucket existiert, können Sie den folgenden AWS CLI Befehl verwenden. Ersetzen Sie es *YOURACCOUNTID* durch Ihre AWS Konto-ID.

```
$ aws s3 ls s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID
```

Die Ausgabe des Befehls sieht ähnlich wie folgt aus:

```
                           PRE conf/
                           PRE jars/
                           PRE scripts/
```

Um zu bestätigen, dass der AWS Glue Exportauftrag bereitgestellt wurde, können Sie den folgenden Befehl verwenden.

```
$ aws glue list-jobs
```

Die Ausgabe listet die bereitgestellten Jobs auf:

```
{
    "JobNames": [
        "AmazonKeyspacesExportToS3-aksglue",
        "AmazonKeyspacesImportFromS3-aksglue",
        "AmazonKeyspacesCount-aksglue"
    ]
}
```

Wenn der CloudFormation Stack-Prozess fehlschlägt, können Sie die detaillierten Fehlerinformationen zum ausgefallenen Stack in der CloudFormation Konsole überprüfen. Um es

erneut zu versuchen, löschen Sie den fehlgeschlagenen Stack mit`aws cloudformation delete-stack --stack-name aksglue`, beheben Sie das zugrunde liegende Problem und führen Sie den Bootstrap-Befehl erneut aus. Der Bootstrap-Befehl ist idempotent und überspringt Ressourcen, die bereits vorhanden sind.

Nachdem der Bootstrap abgeschlossen ist und alle Ressourcen erstellt wurden, fahren Sie mit fort. the section called “Schritt 2: Führen Sie den Exportauftrag aus”

Schritt 2: Führen Sie den Exportauftrag aus

In diesem Schritt führen Sie den im vorherigen Schritt erstellten AWS Glue Exportauftrag aus. Nachdem Sie den Export gestartet haben, können Sie den Auftragsfortschritt mit den `logs` Befehlen `status` und überwachen.

Um den Exportauftrag auszuführen und zu überwachen

1. Führen Sie den Exportbefehl aus. Geben Sie den Schlüsselraum, die Tabelle und den Amazon S3-Bucket an, in den exportiert werden soll. Verwenden Sie den Bucket, der im vorherigen Schritt mit dem Bootstrap-Befehl erstellt wurde. Ersetzen Sie es *YOURACCOUNTID* durch Ihre AWS Konto-ID.

   ```
   $ ./keyspaces-bulk-cli export --keyspace catalog --table book_awards \
       --s3-uri s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID
   ```

   - Sie können zusätzliche Parameter überschreiben, wenn Sie den Export ausführen. Der folgende Befehl erhöht die Anzahl der AWS Glue Arbeiter. Beginnen Sie mit der Standardeinstellung von 2 Workern und erhöhen Sie diese bei Bedarf je nach Tabellengröße und Exportdauer. Überwachen Sie den Job mit dem `status` Befehl und skalieren Sie ihn, falls der Export länger als erwartet dauert.

     ```
     $ ./keyspaces-bulk-cli export --keyspace catalog --table book_awards \
         --s3-uri s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID --workers 8
     ```

2. Überwachen Sie den Status des Exportauftrags. Sie können den aktuellen Ausführungsstatus überprüfen und Protokolle einsehen.

   ```
   $ ./keyspaces-bulk-cli status export
   ```

Die Ausgabe zeigt die Details der Auftragsausführung. Warten Sie, bis das `State` Feld Folgendes anzeigt`SUCCEEDED`:

```
        Job Run: jr_6f8de24d7131da54...
################################################################
# Field          # Value                                       #
################################################################
# Job Name       # AmazonKeyspacesExportToS3-aksglue           #
# State          # SUCCEEDED                                   #
# Started        # 2025-01-30 14:54:22                         #
# Completed      # 2025-01-30 14:56:45                         #
# Execution Time # 143s                                        #
# Workers        # 2                                           #
# Worker Type    # G.2X                                        #
################################################################
```

Verwenden Sie den folgenden Befehl, um die Protokolle für den laufenden oder abgeschlossenen Job anzuzeigen.

```
$ ./keyspaces-bulk-cli logs export
```

Verwenden Sie den folgenden Befehl, um nur Fehlerprotokolle anzuzeigen.

```
$ ./keyspaces-bulk-cli logs export --log-type error
```

3. Bestätigen Sie, dass der Export abgeschlossen ist, indem Sie den Inhalt des Amazon S3-Buckets auflisten. Je nach Größe der Tabelle kann der Export einige Zeit in Anspruch nehmen. Wenn der Exportauftrag abgeschlossen ist, können Sie die folgenden Ordner im Bucket sehen. *YOURACCOUNTID*Ersetzen Sie es durch Ihre AWS Konto-ID.

```
$ aws s3 ls s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID
```

Die Ausgabe zeigt die folgende Struktur in Ihrem Bucket:

```
                           PRE conf/
                           PRE export/
                           PRE jars/
                           PRE scripts/
                           PRE spark-logs/
```

Ihre exportierten Datendateien befinden sich in der folgenden Ordnerstruktur (Datums- und Uhrzeitwerte entsprechen Ihrem eigenen Exportlauf):

```
\------- export
            \----- catalog
                \----- book_awards
                   \----- snapshot
                       \----- year=2025
                           \----- month=01
                              \----- day=30
                                  \----- hour=14
                                      \----- minute=54
                                          \--- part-00000-*.snappy.parquet
```

Um den AWS Glue Job zu planen, den Sie gerade manuell ausgeführt haben, fahren Sie mit fortthe section called “Schritt 3: (Optional) Planen Sie den Exportauftrag”.

Schritt 3: (Optional) Erstellen Sie einen Trigger, um den Exportauftrag zu planen

Um den Exportauftrag regelmäßig auszuführen, können Sie mit dem einen geplanten Trigger erstellen. AWS CLI Weitere Informationen finden Sie unter AWS Glue Trigger im AWS Glue Entwicklerhandbuch.

Um den zu planen AWS Glue Job exportieren

1. Um den Auftragsnamen und die Parameter einer früheren Ausführung zu ermitteln, verwenden Sie das `--json` Flag, um die Ausführungsdetails einschließlich der verwendeten Argumente auszugeben.

   ```
   $ ./keyspaces-bulk-cli runs export --json
   ```

   Die Ausgabe umfasst den Jobnamen, die Worker-Konfiguration und alle Argumente aus jeder Ausführung:

   ```
   [
     {
       "JobName": "AmazonKeyspacesExportToS3-aksglue",
   ```

```
    "RunId": "jr_2d827b0637a36b25f32f03b83e107cf8...",
    "State": "SUCCEEDED",
    "Started": "2025-01-30T14:54:22.480000-04:00",
    "Duration": 127,
    "Workers": 2,
    "WorkerType": "G.2X",
    "Arguments": {
      "--KEYSPACE_NAME": "catalog",
      "--DRIVER_CONF": "keyspaces-application.conf",
      "--TABLE_NAME": "book_awards",
      "--S3_URI": "s3://amazon-keyspaces-bulk-cli-aksglue-111122223333",
      "--FORMAT": "parquet"
    }
  }
]
```

Verwenden Sie im nächsten Schritt die `Arguments` Werte `JobName` `WorkersWorkerType`,, und aus der Ausgabe, um den Trigger zu erstellen.

2. Der folgende AWS CLI Befehl erstellt einen Trigger mit dem Namen`KeyspacesExportWeeklyTrigger`, der den AWS Glue Exportauftrag einmal pro Woche am Montag um 12:00 UTC ausführt. Verwenden Sie die Werte aus der JSON-Ausgabe des vorherigen Schritts für den Jobnamen, die Worker-Konfiguration und die Argumente.

```
aws glue create-trigger \
  --name KeyspacesExportWeeklyTrigger \
  --type SCHEDULED \
  --schedule "cron(0 12 ? * MON *)" \
  --start-on-creation \
  --actions '[{
     "JobName": "AmazonKeyspacesExportToS3-aksglue",
     "NumberOfWorkers": 2,
     "WorkerType": "G.2X",
     "Arguments": {
       "--KEYSPACE_NAME": "catalog",
       "--DRIVER_CONF": "keyspaces-application.conf",
       "--TABLE_NAME": "book_awards",
       "--S3_URI": "s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID",
       "--FORMAT": "parquet"
     }
  }]'
```

- Um bestimmte Parameter für den geplanten Job zu überschreiben, ändern Sie die Werte im `Arguments` Block. Im folgenden Beispiel wird ein Export einer anderen Tabelle mit mehr Workern geplant.

```
aws glue create-trigger \
  --name KeyspacesExportWeeklyTrigger \
  --type SCHEDULED \
  --schedule "cron(0 12 ? * MON *)" \
  --start-on-creation \
  --actions '[{
     "JobName": "AmazonKeyspacesExportToS3-aksglue",
     "NumberOfWorkers": 8,
     "WorkerType": "G.2X",
     "Arguments": {
       "--KEYSPACE_NAME": "my_keyspace",
       "--DRIVER_CONF": "keyspaces-application.conf",
       "--TABLE_NAME": "my_table",
       "--S3_URI": "s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID",
       "--FORMAT": "parquet"
     }
  }]'
```

3. Verwenden Sie den folgenden Befehl, um zu bestätigen, dass der Trigger erstellt wurde.

```
$ aws glue list-triggers
```

Die Ausgabe des Befehls sieht ähnlich wie folgt aus:

```
{
    "TriggerNames": [
        "KeyspacesExportWeeklyTrigger"
    ]
}
```

Um die in diesem Tutorial erstellten AWS Ressourcen zu bereinigen, fahren Sie mit fortthe section called “Schritt 4: (Optional) Bereinigung”.

Schritt 4: (Optional) Bereinigung

Folgen Sie diesen Schritten, um alle in diesem Tutorial erstellten AWS Ressourcen zu entfernen. Die AWS Glue Job-Stacks müssen vor dem übergeordneten Infrastruktur-Stack gelöscht werden, da sie auf die exportierten Werte verweisen.

Um die in diesem Tutorial erstellten Ressourcen zu entfernen

1. Löschen Sie den mit Bootstrap erstellten Amazon S3-Bucket-Inhalt. Ersetzen Sie es *YOURACCOUNTID* durch Ihre AWS Konto-ID.

```
$ aws s3 rm s3://amazon-keyspaces-bulk-cli-aksglue-YOURACCOUNTID --recursive
```

2. Löschen Sie die mit dem Bootstrap-Befehl erstellten AWS Glue Job-Stacks. Diese untergeordneten Stapel müssen vor dem übergeordneten Stapel gelöscht werden. Sie können die folgenden Befehle verwenden.

```
$ aws cloudformation delete-stack --stack-name aksglue-export
$ aws cloudformation delete-stack --stack-name aksglue-import
$ aws cloudformation delete-stack --stack-name aksglue-count
```

   Warten Sie, bis das Löschen der untergeordneten Stapel abgeschlossen ist, bevor Sie fortfahren. Sie können den Status mit dem folgenden Befehl überprüfen.

```
$ aws cloudformation wait stack-delete-complete --stack-name aksglue-export
$ aws cloudformation wait stack-delete-complete --stack-name aksglue-import
$ aws cloudformation wait stack-delete-complete --stack-name aksglue-count
```

3. Löschen Sie den übergeordneten CloudFormation Stapel. Dadurch werden die Amazon S3-Bucket-Ressource und die in diesem Tutorial erstellte IAM-Rolle entfernt.

```
$ aws cloudformation delete-stack --stack-name aksglue
```

4. Wenn Sie im vorherigen Schritt einen Trigger erstellt haben, löschen Sie ihn mit dem folgenden Befehl.

```
$ aws glue delete-trigger --name KeyspacesExportWeeklyTrigger
```

5. Löschen Sie den Amazon Keyspaces-Schlüsselraum und die Tabelle. Durch das Löschen des Schlüsselraums werden automatisch alle Tabellen in diesem Schlüsselraum gelöscht. Verwenden Sie eine der folgenden Optionen.

   AWS CLI

   ```
   $ aws keyspaces delete-keyspace --keyspace-name 'catalog'
   ```

   Um zu bestätigen, dass der Schlüsselraum gelöscht wurde, können Sie den folgenden Befehl verwenden.

   ```
   $ aws keyspaces list-keyspaces
   ```

   Um die Tabelle zuerst zu löschen, können Sie den folgenden Befehl verwenden.

   ```
   $ aws keyspaces delete-table --keyspace-name 'catalog' --table-name
    'book_awards'
   ```

   Um zu bestätigen, dass Ihre Tabelle gelöscht wurde, können Sie den folgenden Befehl verwenden.

   ```
   $ aws keyspaces list-tables --keyspace-name 'catalog'
   ```

   Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz unter Keyspace löschen und Tabelle löschen.

   cqlsh

   ```
   DROP KEYSPACE IF EXISTS "catalog";
   ```

   Um zu überprüfen, ob Ihr Schlüsselraum gelöscht wurde, können Sie die folgende Anweisung verwenden.

   ```
   SELECT * FROM system_schema.keyspaces ;
   ```

   Ihr Schlüsselraum sollte in der Ausgabe dieser Anweisung nicht aufgeführt sein. Beachten Sie, dass es zu einer Verzögerung kommen kann, bis der Schlüsselraum gelöscht wird. Weitere Informationen finden Sie unter the section called “SCHLÜSSELRAUM LÖSCHEN”.

Um die Tabelle zuerst zu löschen, können Sie den folgenden Befehl verwenden.

```
DROP TABLE "catalog"."book_awards";
```

Um zu bestätigen, dass Ihre Tabelle gelöscht wurde, können Sie den folgenden Befehl verwenden.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "catalog";
```

Ihre Tabelle sollte in der Ausgabe dieser Anweisung nicht aufgeführt werden. Beachten Sie, dass es zu einer Verzögerung kommen kann, bis die Tabelle gelöscht wird. Weitere Informationen finden Sie unter the section called “DROP TABLE”.

Verwaltung serverloser Ressourcen in Amazon Keyspaces (für Apache Cassandra)

Amazon Keyspaces (für Apache Cassandra) ist eine Serverless-Anwendung. Anstatt Speicher- und Rechenressourcen für Ihre Arbeitslast über Knoten in einem Cluster bereitzustellen, zu verwalten und zu warten, weist Amazon Keyspaces Speicher- und read/write Durchsatzressourcen direkt Tabellen zu.

Amazon Keyspaces stellt Speicherplatz automatisch auf der Grundlage der in Ihren Tabellen gespeicherten Daten bereit. Es skaliert den Speicherplatz beim Schreiben, Aktualisieren und Löschen von Daten nach oben und unten, und Sie zahlen nur für den Speicherplatz, den Sie tatsächlich nutzen. Daten werden für eine hohe Verfügbarkeit über mehrere Availability Zones hinweg repliziert. Amazon Keyspaces überwacht kontinuierlich die Größe Ihrer Tabellen, um Ihre Speichergebühren zu ermitteln. Weitere Informationen darüber, wie Amazon Keyspaces die fakturierbare Größe der Daten berechnet, finden Sie unter. the section called “Schätzen Sie die Zeilengröße”

Dieses Kapitel behandelt die wichtigsten Aspekte des Ressourcenmanagements in Amazon Keyspaces.

- Zeilengröße schätzen — Um die kodierte Größe von Zeilen in Amazon Keyspaces zu schätzen, berücksichtigen Sie Faktoren wie Partitionsschlüssel-Metadaten, Cluster-Spaltenmetadaten, Spaltenbezeichner, Datentypen und Zeilenmetadaten. Diese kodierte Zeilengröße wird für die Abrechnung, die Kontingentverwaltung und die Kapazitätsplanung für den bereitgestellten Durchsatz verwendet.
- Schätzung des Kapazitätsverbrauchs — In diesem Abschnitt finden Sie Beispiele für die Schätzung des Lese- und Schreibkapazitätsverbrauchs für gängige Szenarien wie Bereichsabfragen, Grenzwertabfragen, Tabellenscans, einfache Transaktionen, statische Spalten und Tabellen mit mehreren Regionen. Sie können Amazon verwenden CloudWatch , um die tatsächliche Kapazitätsauslastung zu überwachen. Weitere Informationen zur Überwachung mit CloudWatch finden Sie unterthe section called “Überwachung mit CloudWatch”.
- read/write Kapazitätsmodi konfigurieren — Sie können zwischen zwei Kapazitätsmodi für die Verarbeitung von Lese- und Schreibvorgängen in Ihren Tabellen wählen:
  - On-demand Modus (Standard) — Bezahlung pro Anfrage für den Lese- und Schreibdurchsatz. Amazon Keyspaces kann die Kapazität sofort auf jedes zuvor erreichte Verkehrsniveau skalieren.

  - Bereitgestellter Modus — Geben Sie im Voraus die erforderliche Anzahl an Lese- und Schreibkapazitätseinheiten an. Dieser Modus trägt zur Aufrechterhaltung einer vorhersehbaren Durchsatzleistung bei.
- Verwaltung der Durchsatzkapazität mit automatischer Skalierung — Für bereitgestellte Tabellen können Sie die automatische Skalierung aktivieren, um die Durchsatzkapazität automatisch an den tatsächlichen Anwendungsdatenverkehr anzupassen. Amazon Keyspaces verwendet Target Tracking, um die bereitgestellte Kapazität zu erhöhen oder zu verringern und die Auslastung auf Ihrem angegebenen Ziel zu halten.
- Effektive Nutzung von Burst-Kapazität — Amazon Keyspaces bietet Burst-Kapazität, indem ein Teil des ungenutzten Durchsatzes für die Bewältigung von Verkehrsspitzen reserviert wird. Diese Flexibilität ermöglicht gelegentliche Aktivitätsschübe, die Ihren bereitgestellten Durchsatz übersteigen.

Informationen zur Behebung von Kapazitätsfehlern finden Sie unter. the section called “Kapazitätsfehler bei serverlosen Servern”

Themen

- Schätzen Sie die Zeilengröße in Amazon Keyspaces
- Schätzen Sie den Kapazitätsverbrauch des Lese- und Schreibdurchsatzes in Amazon Keyspaces
- read/write Kapazitätsmodi in Amazon Keyspaces konfigurieren
- Automatische Verwaltung der Durchsatzkapazität mit Amazon Keyspaces Auto Scaling
- Effektive Nutzung von Burst-Kapazitäten in Amazon Keyspaces

Schätzen Sie die Zeilengröße in Amazon Keyspaces

Amazon Keyspaces bietet vollständig verwalteten Speicher, der eine Lese- und Schreibleistung im einstelligen Millisekundenbereich bietet und Daten dauerhaft in mehreren Availability Zones speichert. AWS Amazon Keyspaces fügt Metadaten an alle Zeilen und Primärschlüsselspalten an, um einen effizienten Datenzugriff und hohe Verfügbarkeit zu unterstützen.

Dieses Thema enthält Einzelheiten zur Schätzung der codierten Zeilengröße in Amazon Keyspaces. Die kodierte Zeilengröße wird bei der Berechnung Ihrer Rechnung und der Kontingentnutzung verwendet. Sie können die kodierte Zeilengröße auch verwenden, um die Anforderungen an die bereitgestellte Durchsatzkapazität für Tabellen zu schätzen.

Um die kodierte Größe von Zeilen in Amazon Keyspaces zu berechnen, können Sie die folgenden Richtlinien verwenden.

Themen

- Schätzen Sie die kodierte Größe von Spalten
- Schätzen Sie die kodierte Größe von Datenwerten auf der Grundlage des Datentyps
- Berücksichtigen Sie die Auswirkungen der Funktionen von Amazon Keyspaces auf die Zeilengröße
- Wählen Sie die richtige Formel, um die kodierte Größe einer Zeile zu berechnen
- Beispiel für die Berechnung der Zeilengröße

Schätzen Sie die kodierte Größe von Spalten

In diesem Abschnitt wird gezeigt, wie die kodierte Größe von Spalten in Amazon Keyspaces geschätzt wird.

- Reguläre Spalten — Verwenden Sie für reguläre Spalten, bei denen es sich nicht um Primärschlüssel, Clusterspalten oder `STATIC` Spalten handelt, die Rohgröße der Zellendaten basierend auf dem Datentyp und fügen Sie die erforderlichen Metadaten hinzu. Die Datentypen und einige wichtige Unterschiede in der Art und Weise, wie Amazon Keyspaces Datentypwerte und Metadaten speichert, sind im nächsten Abschnitt aufgeführt.
- Spalten mit Partitionsschlüsseln — Partitionsschlüssel können bis zu 2048 Byte an Daten enthalten. Jede Schlüsselspalte im Partitionsschlüssel benötigt bis zu 3 Byte an Metadaten. Bei der Berechnung der Größe Ihrer Zeile sollten Sie davon ausgehen, dass jede Partitionsschlüsselspalte die vollen 3 Byte an Metadaten verwendet.
- Cluster-Spalten — In Cluster-Spalten können bis zu 850 Byte an Daten gespeichert werden. Zusätzlich zur Größe des Datenwerts benötigt jede Clusterspalte bis zu 20% der Datenwertgröße für Metadaten. Bei der Berechnung der Größe Ihrer Zeile sollten Sie 1 Byte Metadaten für jeweils 5 Byte des Datenwerts der Clusterspalte hinzufügen.

> **Note**
>
> Um effiziente Abfragen und integrierte Indizierung zu unterstützen, speichert Amazon Keyspaces den Datenwert jeder Partitionsschlüssel- und Clusterschlüsselspalte zweimal.

- Spaltennamen — Der für jeden Spaltennamen benötigte Speicherplatz wird unter Verwendung einer Spalten-ID gespeichert und zu jedem in der Spalte gespeicherten Datenwert hinzugefügt. Der Speicherwert der Spalten-ID hängt von der Gesamtzahl der Spalten in Ihrer Tabelle ab:
  - 1—62 Spalten: 1 Byte
  - 63—124 Spalten: 2 Byte
  - 125—186 Spalten: 3 Byte

  Fügen Sie für jede weitere 62 Spalten 1 Byte hinzu. Beachten Sie, dass in Amazon Keyspaces bis zu 225 reguläre Spalten mit einer einzigen `INSERT` `UPDATE` OR-Anweisung geändert werden können. Weitere Informationen finden Sie unter the section called “Amazon Keyspaces-Servicekontingente”.

Schätzen Sie die kodierte Größe von Datenwerten auf der Grundlage des Datentyps

In diesem Abschnitt wird gezeigt, wie die kodierte Größe verschiedener Datentypen in Amazon Keyspaces geschätzt wird.

- String-Typen — Cassandra- `ASCII`, `TEXT` und `VARCHAR` String-Datentypen werden alle in Amazon Keyspaces unter Verwendung von Unicode mit UTF-8 binärer Kodierung gespeichert. Die Größe einer Zeichenfolge in Amazon Keyspaces entspricht der Anzahl der UTF-8 codierten Byte.
- Numerische Typen — Cassandra `INT`, `BIGINT`, `SMALLINT`, `TINYINT` und `VARINT` Datentypen werden in Amazon Keyspaces als Datenwerte mit variabler Länge mit bis zu 38 signifikanten Ziffern gespeichert. Nullen am Anfang und am Ende werden abgeschnitten. Die Größe jedes dieser Datentypen beträgt ungefähr 1 Byte pro zwei signifikanten Ziffern + 1 Byte.
- Blob-Typ — A `BLOB` in Amazon Keyspaces wird mit der Roh-Bytelänge des Werts gespeichert.
- Boolescher Typ — Die Größe eines `Boolean` Werts oder Werts beträgt 1 Byte`Null`.
- Sammlungstypen — Eine Spalte, in der Sammlungsdatentypen wie Metadaten von 3 Byte gespeichert werden `LIST` oder diese `MAP` erfordern, unabhängig von ihrem Inhalt. Die Größe eines `LIST` oder `MAP` ist (Spalten-ID) + Summe (Größe der verschachtelten Elemente) + (3 Byte). Die Größe eines leeren `LIST` Or `MAP` ist (Spalten-ID) + (3 Byte). Jedes Individuum `LIST` oder `MAP` Element benötigt außerdem 1 Byte an Metadaten.
- User-defined Typen — Ein benutzerdefinierter Typ (UDT) benötigt unabhängig von seinem Inhalt 3 Byte für Metadaten. Für jedes UDT-Element benötigt Amazon Keyspaces zusätzlich 1 Byte an Metadaten.

Um die kodierte Größe einer UDT zu berechnen, beginnen Sie mit dem `field name` und dem `field value` für die Felder einer UDT:

- Feldname — Jeder Feldname der UDT der obersten Ebene wird unter Verwendung eines Bezeichners gespeichert. Der Speicherwert des Bezeichners hängt von der Gesamtzahl der Felder in Ihrer UDT der obersten Ebene ab und kann zwischen 1 und 3 Byte variieren:
  - 1—62 Felder: 1 Byte
  - 63—124 Felder: 2 Byte
  - 125 — maximale Anzahl an Feldern: 3 Byte
- Feldwert — Die Byte, die zum Speichern der Feldwerte der UDT der obersten Ebene erforderlich sind, hängen vom gespeicherten Datentyp ab:
  - Skalarer Datentyp — Die für die Speicherung erforderlichen Byte sind dieselben wie für denselben Datentyp, der in einer regulären Spalte gespeichert ist.
  - Eingefrorenes UDT — Für jedes eingefrorene verschachtelte UDT hat das verschachtelte UDT dieselbe Größe wie im CQL-Binärprotokoll. Bei einer verschachtelten UDT werden 4 Byte für jedes Feld (einschließlich leerer Felder) gespeichert, und der Wert des gespeicherten Felds entspricht dem Serialisierungsformat des Feldwerts im CQL-Binärprotokoll.
  - Gefrorene Sammlungen:
    - LIST und SET — Bei einem verschachtelten eingefrorenen `LIST` oder `SET` werden 4 Byte für jedes Element der Sammlung sowie das Serialisierungsformat des Werts der Sammlung im CQL-Binärprotokoll gespeichert.
    - MAP — Für ein verschachteltes eingefrorenes Objekt gelten `MAP` für jedes Schlüssel-Wert-Paar die folgenden Speicheranforderungen:
      - Ordnen Sie jedem Schlüssel 4 Byte zu und fügen Sie dann das Serialisierungsformat des Schlüssels im CQL-Binärprotokoll hinzu.
      - Ordnen Sie jedem Wert 4 Byte zu und fügen Sie dann das Serialisierungsformat des Werts im CQL-Binärprotokoll hinzu.
- Schlüsselwort FROZEN — Für eingefrorene Sammlungen, die in eingefrorenen Sammlungen verschachtelt sind, benötigt Amazon Keyspaces keine zusätzlichen Byte für Metadaten.
- STATISCHES Schlüsselwort — `STATIC` Spaltendaten werden nicht auf die maximale Zeilengröße von 1 MB angerechnet. Informationen zur Berechnung der Datengröße statischer Spalten finden Sie unterthe section called “Berechnen Sie die statische Spaltengröße pro logischer Partition”.

Berücksichtigen Sie die Auswirkungen der Funktionen von Amazon Keyspaces auf die Zeilengröße

In diesem Abschnitt wird gezeigt, wie sich Client-side Zeitstempel und Time to Live (TTL) -Funktionen in Amazon Keyspaces auf die kodierte Größe einer Zeile auswirken.

Wenn clientseitige Zeitstempel aktiviert oder TTL verwendet wird (unter Verwendung der `USING TTL` Klausel), werden zusätzliche Metadaten für jede Spalte jeder Zeile in Ihrer Tabelle gespeichert. Darüber hinaus können bei bestimmten Erfassungsdatentypen auch Metadaten für jedes Element gespeichert werden. Die folgende Tabelle zeigt den Metadaten-Overhead für jeden Datentyp.

Column/Data Typ	Client-side Zeitstempel aktiviert	TTL ohne clientseitige Zeitstempel	Client-side Zeitstempel sind mit TTL aktiviert
Skalare Typen	18 bytes/column	16 bytes/column	27 bytes/column
Kollektion Die Eiskönigin//Die Eiskönigin UDT//Tuple	18 bytes/column	16 bytes/column	27 bytes/column
Non-frozen Liste	30 bytes/element + 19 bytes/column	15 bytes/element	39 bytes/element + 19 bytes/column
Non-frozen Satz	14 bytes/element + 26 bytes/column	12 bytes/element + 7 bytes/column	23 bytes/element + 26 bytes/column
Non-frozen Landkarte/ Non-frozen UDT	17 bytes/element + 26 bytes/column	15 bytes/element + 7 bytes/column	26 bytes/element + 26 bytes/column
Zähler (einzelne Region)	51 bytes/column	N/A (keine TTL-Unterstützung)	N/A (keine TTL-Unterstützung)
Zähler (mehrere Regionen)	25 + (N_Regionen × 26) bytes/column	N/A (keine TTL-Unterstützung)	N/A (keine TTL-Unterstützung)

Diese Metadaten-Bytes werden auf Ihr Kontingent für die Zeilengröße von 1 MB angerechnet. In der Zählerformel steht N_Regions für die Anzahl der AWS Regionen, in denen die Zählerspalte

mindestens einen Schreibvorgang erhalten hat. Die Werte für den Zähleraufwand sind Näherungswerte und können leicht variieren. Weitere Hinweise zu clientseitigen Zeitstempeln finden Sie unter. the section called “Client-side Zeitstempel” Weitere Informationen zu TTL finden Sie unter the section called “Daten mit Time to Live ablaufen lassen”.

Wählen Sie die richtige Formel, um die kodierte Größe einer Zeile zu berechnen

Dieser Abschnitt zeigt die verschiedenen Formeln, mit denen Sie entweder den Speicher- oder den Kapazitätsdurchsatzbedarf für eine Datenzeile in Amazon Keyspaces schätzen können.

Die kodierte Gesamtgröße einer Datenzeile kann auf der Grundlage einer der folgenden Formeln geschätzt werden, basierend auf Ihrem Ziel:

- Durchsatzkapazität — Um die kodierte Größe einer Zeile zu schätzen, um die erforderlichen read/write Anforderungseinheiten (RRUs/WRUs) oder read/write Kapazitätseinheiten (RCUs/WCUs) zu ermitteln:

```
total encoded size of row = partition key columns + clustering columns + regular
 columns
```

- Speichergröße — Um die kodierte Größe einer Zeile zu schätzen und das vorherzusagen`BillableTableSizeInBytes`, fügen Sie die erforderlichen Metadaten für die Speicherung der Zeile hinzu:

```
total encoded size of row = partition key columns + clustering columns + regular
 columns + row metadata (100 bytes)
```

> **Important**
>
> Alle Spaltenmetadaten, z. B. Spalten-IDs, Partitionsschlüssel-Metadaten, Cluster-Spaltenmetadaten sowie clientseitige Zeitstempel, TTL und Zeilenmetadaten werden auf die maximale Zeilengröße von 1 MB angerechnet.

Beispiel für die Berechnung der Zeilengröße

Betrachten Sie das folgende Beispiel für eine Tabelle, in der alle Spalten vom Typ Integer sind. Die Tabelle hat zwei Partitionsschlüsselspalten, zwei Clusterspalten und eine reguläre Spalte. Da diese Tabelle fünf Spalten hat, beträgt der für den Spaltennamen benötigte Speicherplatz 1 Byte.

```
CREATE TABLE mykeyspace.mytable(pk_col1 int, pk_col2 int, ck_col1 int, ck_col2 int,
 reg_col1 int, primary key((pk_col1, pk_col2),ck_col1, ck_col2));
```

In diesem Beispiel berechnen wir die Datengröße, wenn wir eine Zeile in die Tabelle schreiben, wie in der folgenden Anweisung dargestellt:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1)
 values(1,2,3,4,5);
```

Um die Gesamtanzahl der für diesen Schreibvorgang benötigten Byte zu schätzen, können Sie die folgenden Schritte verwenden.

1. Berechnen Sie die Größe einer Partitionsschlüsselspalte, indem Sie die Byte für den in der Spalte gespeicherten Datentyp und die Metadaten-Bytes addieren. Wiederholen Sie diesen Vorgang für alle Partitionsschlüsselspalten.

   a. Berechnet die Größe der ersten Spalte des Partitionsschlüssels (pk_col1):

   ```
   (2 bytes for the integer data type) x 2 + 1 byte for the column id + 3 bytes
    for partition key metadata = 8 bytes
   ```

   b. Berechnet die Größe der zweiten Spalte des Partitionsschlüssels (pk_col2):

   ```
   (2 bytes for the integer data type) x 2 + 1 byte for the column id + 3 bytes
    for partition key metadata = 8 bytes
   ```

   c. Fügen Sie beide Spalten hinzu, um die geschätzte Gesamtgröße der Partitionsschlüsselspalten zu erhalten:

   ```
   8 bytes + 8 bytes = 16 bytes for the partition key columns
   ```

2. Berechnen Sie die Größe der Clusterspalte, indem Sie die Byte für den in der Spalte gespeicherten Datentyp und die Metadaten-Bytes hinzufügen. Wiederholen Sie diesen Vorgang für alle Clusterspalten.

a. Berechnet die Größe der ersten Spalte der Cluster-Spalte (ck_col1):

```
(2 bytes for the integer data type) x 2 + 20% of the data value (2 bytes) for
 clustering column metadata + 1 byte for the column id  = 6 bytes
```

b. Berechnet die Größe der zweiten Spalte der Cluster-Spalte (ck_col2):

```
(2 bytes for the integer data type) x 2 + 20% of the data value (2 bytes) for
 clustering column metadata + 1 byte for the column id = 6 bytes
```

c. Fügen Sie beide Spalten hinzu, um die geschätzte Gesamtgröße der Cluster-Spalten zu erhalten:

```
6 bytes + 6 bytes = 12 bytes for the clustering columns
```

3. Fügen Sie die Größe der regulären Spalten hinzu. In diesem Beispiel haben wir nur eine Spalte, die eine einstellige Ganzzahl speichert, was 2 Byte mit 1 Byte für die Spalten-ID erfordert.
4. Um schließlich die gesamte codierte Zeilengröße zu erhalten, addieren Sie die Byte für alle Spalten. Um die abrechnungsfähige Speichergröße zu schätzen, fügen Sie die zusätzlichen 100 Byte für Zeilenmetadaten hinzu:

```
16 bytes for the partition key columns + 12 bytes for clustering columns + 3 bytes
 for the regular column + 100 bytes for row metadata = 131 bytes.
```

Informationen zur Überwachung serverloser Ressourcen mit Amazon finden Sie CloudWatch unterthe section called “Überwachung mit CloudWatch”.

Schätzen Sie den Kapazitätsverbrauch des Lese- und Schreibdurchsatzes in Amazon Keyspaces

Wenn Sie Daten in Amazon Keyspaces lesen oder schreiben, hängt die Anzahl der read/write Anforderungseinheiten (RRUs/WRUs) oder read/write Kapazitätseinheiten (RCUs/WCUs), die Ihre Abfrage verbraucht, von der Gesamtmenge an Daten ab, die Amazon Keyspaces verarbeiten muss, um die Abfrage auszuführen. In einigen Fällen können die an den Client zurückgegebenen Daten eine Teilmenge der Daten sein, die Amazon Keyspaces lesen musste, um die Abfrage zu verarbeiten. Bei bedingten Schreibvorgängen verbraucht Amazon Keyspaces Schreibkapazität, auch wenn die bedingte Prüfung fehlschlägt.

Um die Gesamtmenge der für eine Anfrage verarbeiteten Daten abzuschätzen, müssen Sie die kodierte Größe einer Zeile und die Gesamtzahl der Zeilen berücksichtigen. In diesem Thema werden einige Beispiele für gängige Szenarien und Zugriffsmuster behandelt, um zu zeigen, wie Amazon Keyspaces Abfragen verarbeitet und wie sich dies auf den Kapazitätsverbrauch auswirkt. Sie können den Beispielen folgen, um den Kapazitätsbedarf Ihrer Tabellen abzuschätzen, und Amazon verwenden, CloudWatch um den Lese- und Schreibkapazitätsverbrauch für diese Anwendungsfälle zu beobachten.

Informationen zur Berechnung der codierten Zeilengröße in Amazon Keyspaces finden Sie unter. the section called “Schätzen Sie die Zeilengröße”

Themen

- Schätzen Sie den Kapazitätsverbrauch von Bereichsabfragen in Amazon Keyspaces
- Schätzen Sie den Lesekapazitätsverbrauch von Grenzabfragen
- Schätzen Sie den Lesekapazitätsverbrauch von Tabellenscans ab
- Schätzung des Kapazitätsverbrauchs leichter Transaktionen in Amazon Keyspaces
- Schätzen Sie den Kapazitätsverbrauch für statische Spalten in Amazon Keyspaces
- Schätzung und Bereitstellung der Kapazität für eine Tabelle mit mehreren Regionen in Amazon Keyspaces
- Schätzen Sie den Lese- und Schreibkapazitätsverbrauch mit Amazon CloudWatch in Amazon Keyspaces

Schätzen Sie den Kapazitätsverbrauch von Bereichsabfragen in Amazon Keyspaces

Um den Lesekapazitätsverbrauch einer Bereichsabfrage zu untersuchen, verwenden wir die folgende Beispieltabelle, die den On-Demand-Kapazitätsmodus verwendet.

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-------
a | b | 1 | a | b | 50 | <any value that results in a row size larger than 4KB>
a | b | 1 | a | b | 60 | value_1
a | b | 1 | a | b | 70 | <any value that results in a row size larger than 4KB>
```

Führen Sie nun die folgende Abfrage für diese Tabelle aus.

```
SELECT * FROM amazon_keyspaces.example_table_1 WHERE pk1='a' AND pk2='b' AND pk3=1 AND
 ck1='a' AND ck2='b' AND ck3 > 50 AND ck3 < 70;
```

Sie erhalten die folgende Ergebnismenge aus der Abfrage und der von Amazon Keyspaces ausgeführte Lesevorgang verbraucht 2 RRUs im `LOCAL_QUORUM` Konsistenzmodus.

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-------
a | b | 1 | a | b | 60 | value_1
```

Amazon Keyspaces verwendet 2 RRUs, um die Zeilen mit den Werten auszuwerten `ck3=60` und die Abfrage `ck3=70` zu verarbeiten. Amazon Keyspaces gibt jedoch nur die Zeile zurück, in der die in der Abfrage angegebene `WHERE` Bedingung wahr ist, also die Zeile mit dem Wert`ck3=60`. Um den in der Abfrage angegebenen Bereich auszuwerten, liest Amazon Keyspaces in diesem Fall die Zeile, die der Obergrenze des Bereichs entspricht`ck3 = 70`, gibt diese Zeile jedoch nicht im Ergebnis zurück. Der Lesekapazitätsverbrauch basiert auf den bei der Verarbeitung der Abfrage gelesenen Daten, nicht auf den zurückgegebenen Daten.

Schätzen Sie den Lesekapazitätsverbrauch von Grenzabfragen

Bei der Verarbeitung einer Abfrage, die die `LIMIT` Klausel verwendet, liest Amazon Keyspaces Zeilen bis zur maximalen Seitengröße, wenn versucht wird, die in der Abfrage angegebene Bedingung zu erfüllen. Wenn Amazon Keyspaces nicht genügend übereinstimmende Daten finden kann, die dem `LIMIT` Wert auf der ersten Seite entsprechen, sind möglicherweise ein oder mehrere paginierte Aufrufe erforderlich. Um mit dem Lesen auf der nächsten Seite fortzufahren, können Sie ein Paginierungstoken verwenden. Die Standardseitengröße ist 1 MB. Um bei der Verwendung von `LIMIT` Klauseln weniger Lesekapazität zu verbrauchen, können Sie die Seitengröße reduzieren. Weitere Hinweise zur Seitennummerierung finden Sie unter. the section called “Ergebnisse paginieren”

Schauen wir uns als Beispiel die folgende Abfrage an.

```
SELECT * FROM my_table WHERE partition_key=1234 LIMIT 1;
```

Wenn Sie die Seitengröße nicht festlegen, liest Amazon Keyspaces 1 MB an Daten, obwohl es Ihnen nur 1 Zeile zurückgibt. Damit Amazon Keyspaces nur eine Zeile liest, können Sie die Seitengröße für diese Abfrage auf 1 setzen. In diesem Fall würde Amazon Keyspaces nur eine Zeile lesen,

vorausgesetzt, Sie haben keine abgelaufenen Zeilen, die auf Time-to-live Einstellungen oder clientseitigen Zeitstempeln basieren.

Der `PAGE SIZE` Parameter bestimmt, wie viele Zeilen Amazon Keyspaces für jede Anfrage von der Festplatte scannt, nicht, wie viele Zeilen Amazon Keyspaces an den Client zurückgibt. Amazon Keyspaces wendet die von Ihnen angegebenen Filter an, z. B. Ungleichheit bei Nichtschlüsselspalten oder `LIMIT` nach dem Scannen der Daten auf der Festplatte. Wenn Sie das nicht explizit festlegen`PAGE SIZE`, liest Amazon Keyspaces bis zu 1 MB an Daten, bevor Filter angewendet werden. Wenn Sie beispielsweise `LIMIT 1` ohne Angabe von verwenden`PAGE SIZE`, könnte Amazon Keyspaces Tausende von Zeilen von der Festplatte lesen, bevor die Limit-Klausel angewendet und nur eine einzige Zeile zurückgegeben wird.

Um zu viel Lesen zu vermeiden, reduzieren Sie `PAGE SIZE` die Anzahl der Zeilen, die Amazon Keyspaces bei jedem Abruf scannt. Wenn Sie beispielsweise `LIMIT 5` in Ihrer Abfrage definieren, legen Sie für einen Wert zwischen 5 und 10 fest, sodass Amazon Keyspaces bei jedem paginierten Aufruf nur 5 bis 10 Zeilen scannt. `PAGE SIZE` Sie können diese Zahl ändern, um die Anzahl der Abrufe zu reduzieren. Für Grenzwerte, die größer als die Seitengröße sind, behält Amazon Keyspaces die Gesamtzahl der Ergebnisse mit Paginierungsstatus bei. Bei 10.000 Zeilen kann Amazon Keyspaces diese Ergebnisse auf zwei Seiten mit jeweils 5.000 Zeilen abrufen. `LIMIT` Die Grenze von 1 MB ist die Obergrenze für jede festgelegte Seitengröße.

Schätzen Sie den Lesekapazitätsverbrauch von Tabellenscans ab

Abfragen, die zu vollständigen Tabellenscans führen, z. B. Abfragen, die die `ALLOW FILTERING` Option verwenden, sind ein weiteres Beispiel für Abfragen, die mehr Lesevorgänge verarbeiten, als sie als Ergebnisse zurückgeben. Und der Verbrauch an Lesekapazität basiert auf den gelesenen Daten, nicht auf den zurückgegebenen Daten.

Für das Beispiel für den Tabellenscan verwenden wir die folgende Beispieltabelle im On-Demand-Kapazitätsmodus.

```
pk | ck | value
---+----+---------
pk | 10 | <any value that results in a row size larger than 4KB>
pk | 20 | value_1
pk | 30 | <any value that results in a row size larger than 4KB>
```

Amazon Keyspaces erstellt standardmäßig eine Tabelle im On-Demand-Kapazitätsmodus mit vier Partitionen. In dieser Beispieltabelle werden alle Daten in einer Partition gespeichert und die restlichen drei Partitionen sind leer.

Führen Sie nun die folgende Abfrage für die Tabelle aus.

```
SELECT * from amazon_keyspaces.example_table_2;
```

Diese Abfrage führt zu einem Tabellenscanvorgang, bei dem Amazon Keyspaces alle vier Partitionen der Tabelle scannt und im `LOCAL_QUORUM` Konsistenzmodus 6 RRUs verbraucht. Erstens verbraucht Amazon Keyspaces 3 RRUs zum Lesen der drei Zeilen mit. `pk='pk'` Dann verbraucht Amazon Keyspaces die zusätzlichen 3 RRUs zum Scannen der drei leeren Partitionen der Tabelle. Da diese Abfrage zu einem Tabellenscan führt, scannt Amazon Keyspaces alle Partitionen in der Tabelle, einschließlich Partitionen ohne Daten.

Schätzung des Kapazitätsverbrauchs leichter Transaktionen in Amazon Keyspaces

Lightweight Transactions (LWT) ermöglichen es Ihnen, bedingte Schreiboperationen für Ihre Tabellendaten durchzuführen. Bedingte Aktualisierungsoperationen sind nützlich, wenn Datensätze auf der Grundlage von Bedingungen eingefügt, aktualisiert und gelöscht werden, die den aktuellen Status bewerten.

In Amazon Keyspaces erfordern alle Schreibvorgänge die Konsistenz LOCAL_QUORUM, und es fallen keine zusätzlichen Gebühren für die Verwendung von LWTs an. Der Unterschied bei LWTs besteht darin, dass Amazon Keyspaces Schreibkapazitätseinheiten (WCUs) oder Schreibanforderungseinheiten (WRUs) verbrauchtFALSE, wenn eine LWT-Zustandsprüfung ergibt. Die Anzahl der WCUs/WRUs verbrauchten Daten hängt von der Größe der Zeile ab.

Wenn die Zeilengröße beispielsweise 2 KB beträgt, verbraucht der fehlgeschlagene bedingte Schreibvorgang zwei WCUs/WRUs. Wenn die Zeile derzeit nicht in der Tabelle vorhanden ist, verbraucht der Vorgang eine Zeile. WCUs/WRUs

Um die Anzahl der Anfragen zu ermitteln, die zu Fehlschlägen bei der Zustandsprüfung geführt haben, können Sie die `ConditionalCheckFailed` Metrik in CloudWatch überwachen.

Schätzen Sie die LWT-Kosten für Tabellen mit Time to Live (TTL)

LWTs können zusätzliche Lesekapazitätseinheiten (RCUs) oder Read Request Units (RRUs) für Tabellen benötigen, die mit TTL konfiguriert sind und keine clientseitigen Zeitstempel verwenden.

Bei Verwendung von `IF EXISTS` oder `IF NOT EXISTS` Schlüsselwörtern mit Ergebnissen der Zustandsprüfung werden die folgenden Kapazitätseinheiten `FALSE` verbraucht:

- RCUs/RRUs — Wenn die Zeile existiert, basieren die RCUs/RRUs verbrauchten Zeilen auf der Größe der vorhandenen Zeile.
- RCUs/RRUs — Wenn die Zeile nicht existiert, RCU/RRU wird eine einzelne Zeile verbraucht.

Wenn die ausgewertete Bedingung zu einem erfolgreichen Schreibvorgang führt, WCUs/WRUs werden sie auf der Grundlage der Größe der neuen Zeile verbraucht.

Schätzen Sie den Kapazitätsverbrauch für statische Spalten in Amazon Keyspaces

In einer Amazon Keyspaces-Tabelle mit Clusterspalten können Sie das `STATIC` Schlüsselwort verwenden, um eine statische Spalte zu erstellen. Der in einer statischen Spalte gespeicherte Wert wird von allen Zeilen in einer logischen Partition gemeinsam genutzt. Wenn Sie den Wert dieser Spalte aktualisieren, wendet Amazon Keyspaces die Änderung automatisch auf alle Zeilen in der Partition an.

In diesem Abschnitt wird beschrieben, wie Sie die kodierte Datengröße berechnen, wenn Sie in statische Spalten schreiben. Dieser Vorgang wird getrennt von dem Prozess durchgeführt, bei dem Daten in die nichtstatischen Spalten einer Zeile geschrieben werden. Neben den Größenkontingenten für statische Daten wirken sich Lese- und Schreibvorgänge an statischen Spalten auch unabhängig voneinander auf die Messung und die Durchsatzkapazität von Tabellen aus. Informationen zu den funktionellen Unterschieden zu Apache Cassandra bei der Verwendung statischer Spalten und paginierter Leseergebnisse finden Sie unter. the section called “Paginierung”

Themen

- Berechnen Sie die statische Spaltengröße pro logischer Partition in Amazon Keyspaces
- Schätzen Sie die Kapazitätsdurchsatzanforderungen für read/write Operationen mit statischen Daten in Amazon Keyspaces

Berechnen Sie die statische Spaltengröße pro logischer Partition in Amazon Keyspaces

Dieser Abschnitt enthält Einzelheiten zur Schätzung der codierten Größe statischer Spalten in Amazon Keyspaces. Die kodierte Größe wird bei der Berechnung Ihrer Rechnung und der

Kontingentnutzung verwendet. Sie sollten die kodierte Größe auch verwenden, wenn Sie die Anforderungen an die bereitgestellte Durchsatzkapazität für Tabellen berechnen. Um die kodierte Größe statischer Spalten in Amazon Keyspaces zu berechnen, können Sie die folgenden Richtlinien verwenden.

- Partitionsschlüssel können bis zu 2048 Byte an Daten enthalten. Jede Schlüsselspalte im Partitionsschlüssel benötigt bis zu 3 Byte an Metadaten. Diese Metadaten-Bytes werden auf Ihr statisches Datengrößenkontingent von 1 MB pro Partition angerechnet. Bei der Berechnung der Größe Ihrer statischen Daten sollten Sie davon ausgehen, dass jede Partitionsschlüsselspalte die vollen 3 Byte an Metadaten verwendet.
- Verwenden Sie die Rohgröße der statischen Spaltendatenwerte basierend auf dem Datentyp. Weitere Informationen zu Datentypen finden Sie unter the section called “Datentypen”.
- Fügen Sie der Größe der statischen Daten für Metadaten 104 Byte hinzu.
- Clusterspalten und normale Spalten, die keine Primärschlüsselspalten sind, werden nicht auf die Größe statischer Daten angerechnet. Informationen zum Schätzen der Größe nichtstatischer Daten innerhalb von Zeilen finden Sie unter. the section called “Schätzen Sie die Zeilengröße”

Die kodierte Gesamtgröße einer statischen Spalte basiert auf der folgenden Formel:

```
partition key columns + static columns + metadata = total encoded size of static data
```

Betrachten Sie das folgende Beispiel für eine Tabelle, in der alle Spalten vom Typ Integer sind. Die Tabelle hat zwei Partitionsschlüsselspalten, zwei Clusterspalten, eine reguläre Spalte und eine statische Spalte.

```
CREATE TABLE mykeyspace.mytable(pk_col1 int, pk_col2 int, ck_col1 int, ck_col2
 int, reg_col1 int, static_col1 int static, primary key((pk_col1, pk_col2),ck_col1,
 ck_col2));
```

In diesem Beispiel berechnen wir die Größe der statischen Daten der folgenden Anweisung:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, static_col1) values(1,2,6);
```

Um die Gesamtanzahl der für diesen Schreibvorgang benötigten Byte abzuschätzen, können Sie die folgenden Schritte verwenden.

1. Berechnen Sie die Größe einer Partitionsschlüsselspalte, indem Sie die Byte für den in der Spalte gespeicherten Datentyp und die Metadaten-Bytes addieren. Wiederholen Sie diesen Vorgang für alle Partitionsschlüsselspalten.

   a. Berechnet die Größe der ersten Spalte des Partitionsschlüssels (pk_col1):

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   b. Berechnet die Größe der zweiten Spalte des Partitionsschlüssels (pk_col2):

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   c. Fügen Sie beide Spalten hinzu, um die geschätzte Gesamtgröße der Partitionsschlüsselspalten zu erhalten:

   ```
   7 bytes + 7 bytes = 14 bytes for the partition key columns
   ```

2. Fügen Sie die Größe der statischen Spalten hinzu. In diesem Beispiel haben wir nur eine statische Spalte, die eine Ganzzahl speichert (was 4 Byte erfordert).
3. Um schließlich die gesamte kodierte Größe der statischen Spaltendaten zu ermitteln, addieren Sie die Byte für die Primärschlüsselspalten und statischen Spalten und fügen Sie die zusätzlichen 104 Byte für Metadaten hinzu:

   ```
   14 bytes for the partition key columns + 4 bytes for the static column + 104 bytes
    for metadata = 122 bytes.
   ```

Sie können statische und nicht statische Daten auch mit derselben Anweisung aktualisieren. Um die Gesamtgröße des Schreibvorgangs abzuschätzen, müssen Sie zunächst den Umfang der nichtstatischen Datenaktualisierung berechnen. Berechnen Sie dann die Größe der Zeilenaktualisierung, wie im Beispiel unter gezeigtthe section called “Schätzen Sie die Zeilengröße”, und fügen Sie die Ergebnisse hinzu.

In diesem Fall können Sie insgesamt 2 MB schreiben — 1 MB ist das maximale Zeilengrößenkontingent und 1 MB ist das Kontingent für die maximale statische Datengröße pro logischer Partition.

Um die Gesamtgröße einer Aktualisierung statischer und nicht statischer Daten in derselben Anweisung zu berechnen, können Sie die folgende Formel verwenden:

```
(partition key columns + static columns + metadata = total encoded size of static data)
 + (partition key columns + clustering columns + regular columns + row metadata = total
 encoded size of row)
= total encoded size of data written
```

Betrachten Sie das folgende Beispiel für eine Tabelle, in der alle Spalten vom Typ Integer sind. Die Tabelle hat zwei Partitionsschlüsselspalten, zwei Clusterspalten, eine reguläre Spalte und eine statische Spalte.

```
CREATE TABLE mykeyspace.mytable(pk_col1 int, pk_col2 int, ck_col1 int, ck_col2
 int, reg_col1 int, static_col1 int static, primary key((pk_col1, pk_col2),ck_col1,
 ck_col2));
```

In diesem Beispiel berechnen wir die Datengröße, wenn wir eine Zeile in die Tabelle schreiben, wie in der folgenden Anweisung dargestellt:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1,
 static_col1) values(2,3,4,5,6,7);
```

Um die Gesamtanzahl der für diesen Schreibvorgang benötigten Byte zu schätzen, können Sie die folgenden Schritte verwenden.

1. Berechnen Sie die gesamte kodierte Größe der statischen Daten wie zuvor gezeigt. In diesem Beispiel sind es 122 Byte.
2. Fügen Sie die Größe der codierten Gesamtgröße der Zeile hinzu, die auf der Aktualisierung nicht statischer Daten basiert. Gehen Sie dabei wie unter beschrieben vor. the section called “Schätzen Sie die Zeilengröße” In diesem Beispiel beträgt die Gesamtgröße der Zeilenaktualisierung 134 Byte.

   ```
   122 bytes for static data + 134 bytes for nonstatic data = 256 bytes.
   ```

Schätzen Sie die Kapazitätsdurchsatzanforderungen für read/write Operationen mit statischen Daten in Amazon Keyspaces

Statische Daten werden logischen Partitionen in Cassandra zugeordnet, nicht einzelnen Zeilen. Logische Partitionen in Amazon Keyspaces können in ihrer Größe praktisch unbegrenzt sein, da sie sich über mehrere physische Speicherpartitionen erstrecken. Daher schreiben Amazon Keyspaces-Messgeräte Operationen für statische und nichtstatische Daten getrennt. Darüber hinaus erfordern Schreibvorgänge, die sowohl statische als auch nicht statische Daten enthalten, zusätzliche zugrunde liegende Operationen, um die Datenkonsistenz zu gewährleisten.

Wenn Sie einen gemischten Schreibvorgang für statische und nicht statische Daten ausführen, führt dies zu zwei separaten Schreibvorgängen — einer für nicht statische und einer für statische Daten. Dies gilt sowohl für den bedarfsgesteuerten als auch für den Modus mit bereitgestellter Kapazität. read/write

Das folgende Beispiel enthält Details zur Schätzung der erforderlichen Lesekapazitätseinheiten (RCUs) und Schreibkapazitätseinheiten (WCUs), wenn Sie die Anforderungen an die bereitgestellte Durchsatzkapazität für Tabellen in Amazon Keyspaces mit statischen Spalten berechnen. Mithilfe der folgenden Formel können Sie abschätzen, wie viel Kapazität Ihre Tabelle für die Verarbeitung von Schreibvorgängen benötigt, die sowohl statische als auch nichtstatische Daten enthalten:

```
2 x WCUs required for nonstatic data + 2 x WCUs required for static data
```

Wenn Ihre Anwendung beispielsweise 27 KB an Daten pro Sekunde schreibt und jeder Schreibvorgang 25,5 KB nichtstatische Daten und 1,5 KB statische Daten umfasst, benötigt Ihre Tabelle 56 WCUs (2 x 26 WCUs + 2 x 2 WCUs).

Amazon Keyspaces misst die Lesevorgänge statischer und nichtstatischer Daten genauso wie die Lesevorgänge mehrerer Zeilen. Daher basiert der Preis für das Lesen statischer und nichtstatischer Daten in derselben Operation auf der Gesamtgröße der Daten, die für den Lesevorgang verarbeitet wurden.

Informationen zur Überwachung serverloser Ressourcen mit Amazon finden Sie CloudWatch unterthe section called “Überwachung mit CloudWatch”.

Schätzung und Bereitstellung der Kapazität für eine Tabelle mit mehreren Regionen in Amazon Keyspaces

Sie können die Durchsatzkapazität einer Tabelle mit mehreren Regionen auf zwei Arten konfigurieren:

- On-demand Kapazitätsmodus, gemessen in Schreibanforderungseinheiten (WRUs)
- Bereitgestellter Kapazitätsmodus mit auto Skalierung, gemessen in Schreibkapazitätseinheiten (WCUs)

Sie können den Modus für bereitgestellte Kapazität mit Auto Scaling oder den On-Demand-Kapazitätsmodus verwenden, um sicherzustellen, dass eine Tabelle mit mehreren Regionen über ausreichend Kapazität verfügt, um replizierte Schreibvorgänge für alle durchzuführen. AWS-Regionen

> **Note**
>
> Wenn Sie den Kapazitätsmodus der Tabelle in einer der Regionen ändern, wird der Kapazitätsmodus für alle Replikate geändert.

Standardmäßig verwendet Amazon Keyspaces den On-Demand-Modus für Tabellen mit mehreren Regionen. Im On-Demand-Modus müssen Sie nicht angeben, wie viel Lese- und Schreibdurchsatz Sie von Ihrer Anwendung erwarten. Amazon Keyspaces passt sich sofort Ihren Workloads an, wenn sie auf ein zuvor erreichtes Datenverkehrsniveau hoch- oder herunterfahren. Wenn der Traffic eines Workloads einen neuen Höhepunkt erreicht, passt sich Amazon Keyspaces schnell an den Workload an.

Wenn Sie den Modus Bereitgestellte Kapazität für eine Tabelle wählen, müssen Sie die Anzahl der Lesekapazitätseinheiten (RCUs) und Schreibkapazitätseinheiten (WCUs) pro Sekunde konfigurieren, die Ihre Anwendung benötigt.

Um den Durchsatzkapazitätsbedarf einer Tabelle mit mehreren Regionen zu planen, sollten Sie zunächst schätzen, wie viele WCUs pro Sekunde für jede Region benötigt werden. Dann fügen Sie die Schreibvorgänge aus allen Regionen hinzu, in denen Ihre Tabelle repliziert wird, und verwenden die Summe, um Kapazität für jede Region bereitzustellen. Dies ist erforderlich, da jeder Schreibvorgang, der in einer Region ausgeführt wird, auch in jeder Replikatregion wiederholt werden muss.

Wenn die Tabelle nicht über genügend Kapazität verfügt, um die Schreibvorgänge aus allen Regionen zu verarbeiten, treten Kapazitätsausnahmen auf. Darüber hinaus werden die Wartezeiten bei der interregionalen Replikation zunehmen.

Wenn Sie beispielsweise über eine Tabelle mit mehreren Regionen verfügen, in der Sie 5 Schreibvorgänge pro Sekunde in USA Ost (Nord-Virginia), 10 Schreibvorgänge pro Sekunde in USA Ost (Ohio) und 5 Schreibvorgänge pro Sekunde in Europa (Irland) erwarten, sollten Sie damit rechnen, dass die Tabelle 20 WCUs in jeder Region verbraucht: USA Ost (Nord-Virginia), USA Ost (Ohio) und Europa (Irland). Das bedeutet, dass Sie in diesem Beispiel 20 WCUs für jedes Replikat der Tabelle bereitstellen müssen. Sie können den Kapazitätsverbrauch Ihrer Tabelle mit Amazon überwachen CloudWatch. Weitere Informationen finden Sie unter the section called “Überwachung mit CloudWatch”.

Jeder Schreibvorgang wird mit 1 WCU abgerechnet. In diesem Beispiel würden also insgesamt 60 WCUs abgerechnet. Weitere Informationen zu den Preisen finden Sie unter Amazon Keyspaces (für Apache Cassandra) — Preise.

Weitere Informationen zur bereitgestellten Kapazität mit Amazon Keyspaces Auto Scaling finden Sie unter. the section called “Verwalten Sie die Durchsatzkapazität mit Auto Scaling”

> **Note**
>
> Wenn eine Tabelle im Modus für bereitgestellte Kapazität mit Auto Scaling ausgeführt wird, darf die bereitgestellte Schreibkapazität innerhalb dieser Auto Scaling-Einstellungen für jede Region schwanken.

Schätzen Sie den Lese- und Schreibkapazitätsverbrauch mit Amazon CloudWatch in Amazon Keyspaces

Um den Lese- und Schreibkapazitätsverbrauch abzuschätzen und zu überwachen, können Sie ein CloudWatch Dashboard verwenden. Weitere Informationen zu verfügbaren Metriken für Amazon Keyspaces finden Sie unterthe section called “Metriken und Dimensionen”.

Gehen Sie wie folgt vor, um die Lese- und Schreibkapazitätseinheiten zu überwachen CloudWatch, die von einer bestimmten Anweisung mit verbraucht werden.

1. Erstellen Sie eine neue Tabelle mit Beispieldaten

2. Konfigurieren Sie ein Amazon CloudWatch Keyspaces-Dashboard für die Tabelle. Zu Beginn können Sie eine Dashboard-Vorlage verwenden, die auf Github verfügbar ist.
3. Führen Sie die CQL-Anweisung aus, beispielsweise mit der `ALLOW FILTERING` Option, und überprüfen Sie im Dashboard, welche Lesekapazitätseinheiten für den vollständigen Tabellenscan verbraucht wurden.

read/write Kapazitätsmodi in Amazon Keyspaces konfigurieren

Amazon Keyspaces bietet zwei read/write Kapazitätsmodi für die Verarbeitung von Lese- und Schreibvorgängen in Ihren Tabellen:

- On-demand (Standard)
- Bereitgestellt

Der von Ihnen gewählte read/write Kapazitätsmodus steuert, wie Ihnen der Lese- und Schreibdurchsatz in Rechnung gestellt wird und wie die Tabellendurchsatzkapazität verwaltet wird.

Themen

- Konfigurieren Sie den Kapazitätsmodus auf Anforderung
- Konfigurieren Sie den Modus für bereitgestellte Kapazität
- Den Kapazitätsmodus einer Tabelle in Amazon Keyspaces anzeigen
- Ändern Sie den Kapazitätsmodus einer Tabelle in Amazon Keyspaces
- Vorwärmung für Tabellen in Amazon Keyspaces konfigurieren

Konfigurieren Sie den Kapazitätsmodus auf Anforderung

Der On-Demand-Kapazitätsmodus von Amazon Keyspaces (für Apache Cassandra) ist eine flexible Abrechnungsoption, mit der Tausende von Anfragen pro Sekunde ohne Kapazitätsplanung bearbeitet werden können. Diese Option bietet Pay-per-Request-Preise für Lese- und Schreibanfragen, sodass Sie nur für das bezahlen, was Sie tatsächlich nutzen.

Wenn Sie den On-Demand-Modus wählen, kann Amazon Keyspaces die Durchsatzkapazität für Ihre Tabelle sofort auf jedes zuvor erreichte Datenverkehrsniveau hochskalieren und dann wieder herunterfahren, wenn der Anwendungsverkehr abnimmt. Wenn der Traffic eines Workloads einen neuen Höhepunkt erreicht, passt sich der Service schnell an, um die Durchsatzkapazität für Ihre

Tabelle zu erhöhen. Sie können den On-Demand-Kapazitätsmodus sowohl für neue als auch für bestehende Tabellen aktivieren.

On-demand Der Modus ist eine gute Option, wenn eine der folgenden Bedingungen zutrifft:

- Sie erstellen neue Tabellen mit unbekannten Workloads.
- Der Datenverkehr Ihrer Anwendung ist nicht berechenbar.
- Sie ziehen es vor, nur für Ihre tatsächliche Nutzung zu zahlen.

Um mit dem On-Demand-Modus zu beginnen, können Sie mithilfe der Konsole oder mit einigen Zeilen Cassandra Query Language (CQL) -Code eine neue Tabelle erstellen oder eine vorhandene Tabelle aktualisieren, um den On-Demand-Kapazitätsmodus zu verwenden. Weitere Informationen finden Sie unter the section called “Tabellen”.

Themen

- Leseanforderungseinheiten und Schreibanforderungseinheiten
- Spitzenaufkommen an Datenverkehr und Skalierungseigenschaften
- Anfänglicher Durchsatz für den On-Demand-Kapazitätsmodus

Leseanforderungseinheiten und Schreibanforderungseinheiten

Bei Tabellen im On-Demand-Kapazitätsmodus müssen Sie nicht im Voraus angeben, wie viel Lese- und Schreibdurchsatz Ihre Anwendung voraussichtlich verwenden soll. Amazon Keyspaces berechnet Ihnen die Lese- und Schreibvorgänge, die Sie an Ihren Tabellen durchführen, in Form von Read Request Units (RRUs) und WRITE Request Units (WRUs).

- Eine RRU steht für eine `LOCAL_QUORUM` Leseanforderung oder zwei `LOCAL_ONE` Leseanforderungen für eine Zeile mit einer Größe von bis zu 4 KB. Wenn Sie eine Zeile lesen müssen, die größer als 4 KB ist, verwendet der Lesevorgang zusätzliche RRUs. Die Gesamtzahl der erforderlichen RRUs hängt von der Zeilengröße und davon ab, ob Sie Konsistenz verwenden `LOCAL_QUORUM` oder `LOCAL_ONE` lesen möchten. Zum Lesen einer 8-KB-Zeile sind beispielsweise 2 RRUs mit `LOCAL_QUORUM` Lesekonsistenz und 1 RRU erforderlich, wenn Sie Lesekonsistenz wählen`LOCAL_ONE`.
- Eine WRU steht für einen Schreibvorgang für eine Zeile mit einer Größe von bis zu 1 KB. Bei allen Schreibvorgängen wird `LOCAL_QUORUM` Konsistenz verwendet, und es fallen keine zusätzlichen Gebühren für die Verwendung von Lightweight Transactions (LWTs) an. Wenn Sie eine Zeile

schreiben müssen, die größer als 1 KB ist, verwendet der Schreibvorgang zusätzliche WRUs. Die Gesamtzahl der benötigten WRUs hängt von der Zeilengröße ab. Wenn Ihre Zeilengröße beispielsweise 2 KB beträgt, benötigen Sie 2 WRUs, um eine Schreibanforderung auszuführen.

Informationen zu den unterstützten Konsistenzstufen finden Sie unterthe section called “Unterstützte Cassandra-Konsistenzstufen”.

Spitzenaufkommen an Datenverkehr und Skalierungseigenschaften

Amazon Keyspaces-Tabellen, die den On-Demand-Kapazitätsmodus verwenden, passen sich automatisch an das Datenverkehrsvolumen Ihrer Anwendung an. On-demand Der Kapazitätsmodus nimmt sofort das Doppelte des vorherigen Spitzenverkehrs auf einer Tabelle auf. Beispielsweise kann das Datenverkehrsmuster Ihrer Anwendung zwischen 5.000 und 10.000 `LOCAL_QUORUM` Lesevorgängen pro Sekunde variieren, wobei 10.000 Lesevorgänge pro Sekunde die vorherige Verkehrsspitze sind.

Bei diesem Muster ermöglicht der On-Demand-Kapazitätsmodus sofort einen anhaltenden Datenverkehr von bis zu 20.000 Lesevorgängen pro Sekunde. Wenn Ihre Anwendung einen Datenverkehr von 20.000 Lesevorgängen pro Sekunde aushält, wird dieser Spitzenwert zu Ihrem neuen vorherigen Spitzenwert, sodass der nachfolgende Datenverkehr bis zu 40.000 Lesevorgänge pro Sekunde erreichen kann.

Wenn Sie mehr als das Doppelte Ihres vorherigen Spitzenwerts in einer Tabelle benötigen, weist Amazon Keyspaces automatisch mehr Kapazität zu, wenn Ihr Verkehrsaufkommen steigt. Dadurch wird sichergestellt, dass Ihre Tabelle über genügend Durchsatzkapazität verfügt, um die zusätzlichen Anfragen zu verarbeiten. Sie können jedoch Fehler mit unzureichender Durchsatzkapazität feststellen, wenn Sie innerhalb von 30 Minuten das Doppelte Ihres vorherigen Spitzenwerts überschreiten.

Nehmen wir beispielsweise an, dass das Datenverkehrsmuster Ihrer Anwendung zwischen 5.000 und 10.000 stark konsistenten Lesevorgängen pro Sekunde variiert, wobei 20.000 Lesevorgänge pro Sekunde die zuvor erreichte Verkehrsspitze sind. In diesem Fall empfiehlt der Service, dass Sie Ihr Traffic-Wachstum über mindestens 30 Minuten verteilen, bevor Sie bis zu 40.000 Lesevorgänge pro Sekunde erreichen.

Informationen zur Schätzung des Lese- und Schreibkapazitätsverbrauchs einer Tabelle finden Sie unterthe section called “Schätzen Sie den Kapazitätsverbrauch”.

Weitere Informationen zu Standardkontingenten für Ihr Konto und deren Erhöhung finden Sie unterKontingente.

Anfänglicher Durchsatz für den On-Demand-Kapazitätsmodus

Wenn Sie eine neue Tabelle mit aktiviertem On-Demand-Kapazitätsmodus erstellen oder eine bestehende Tabelle zum ersten Mal in den On-Demand-Kapazitätsmodus wechseln, weist die Tabelle die folgenden vorherigen Spitzeneinstellungen auf, obwohl sie zuvor keinen Datenverkehr im On-Demand-Kapazitätsmodus bedient hat:

- Neu erstellte Tabelle mit On-Demand-Kapazitätsmodus: Der bisherige Spitzenwert lag bei 2.000 WRUs und 6.000 RRUs. Sie können den vorherigen Spitzenwert sofort auf das Doppelte erhöhen. Auf diese Weise können neu erstellte On-Demand-Tabellen bis zu 4.000 WRUs und 12.000 RRUs verarbeiten.
- Bestehende Tabelle wurde in den On-Demand-Kapazitätsmodus umgeschaltet: Der vorherige Spitzenwert entspricht der Hälfte der zuvor für die Tabelle bereitgestellten WCUs und RCUs oder den Einstellungen für eine neu erstellte Tabelle mit On-Demand-Kapazitätsmodus, je nachdem, welcher Wert höher ist.

Konfigurieren Sie den Modus für bereitgestellte Kapazität

Wenn Sie den Modus „Bereitgestellte Durchsatzkapazität“ wählen, geben Sie die Anzahl der Lese- und Schreibvorgänge pro Sekunde an, die für Ihre Anwendung erforderlich sind. Auf diese Weise können Sie Ihre Amazon Keyspaces-Nutzung so verwalten, dass sie bei oder unter einer definierten Anforderungsrate bleibt, um die Vorhersehbarkeit zu gewährleisten. Weitere Informationen zur automatischen Skalierung für den bereitgestellten Durchsatz finden Sie unter. the section called “Verwalten Sie die Durchsatzkapazität mit Auto Scaling”

Der Kapazitätsmodus für bereitgestellten Durchsatz ist eine gute Option, wenn eine der folgenden Bedingungen zutrifft:

- Der Datenverkehr Ihrer Anwendung ist berechenbar.
- Sie führen Anwendungen aus, deren Datenverkehr konsistent ist oder allmählich ansteigt.
- Sie können den Kapazitätsbedarf prognostizieren.

Lesekapazitätseinheiten und Schreibkapazitätseinheiten

Für bereitgestellte Durchsatzkapazitätsmodus-Tabellen geben Sie die Durchsatzkapazität in Form von Lesekapazitätseinheiten (RCUs) und Schreibkapazitätseinheiten (WCUs) an:

- Eine RCU entspricht einem `LOCAL_QUORUM` Lesevorgang pro Sekunde oder zwei `LOCAL_ONE` Lesevorgängen pro Sekunde für eine Zeile mit einer Größe von bis zu 4 KB. Wenn Sie eine Zeile lesen müssen, die größer als 4 KB ist, verwendet der Lesevorgang zusätzliche RCUs.

  Die Gesamtzahl der erforderlichen RCUs hängt von der Zeilengröße und davon ab, ob Sie lesen `LOCAL_QUORUM` oder `LOCAL_ONE` lesen möchten. Wenn Ihre Zeilengröße beispielsweise 8 KB beträgt, benötigen Sie 2 RCUs, um einen `LOCAL_QUORUM` Lesevorgang pro Sekunde durchzuführen, und 1 RCU, wenn Sie Lesevorgänge wählen. `LOCAL_ONE`
- Eine WCU entspricht einem Schreibvorgang pro Sekunde für eine Zeile mit einer Größe von bis zu 1 KB. Bei allen Schreibvorgängen wird `LOCAL_QUORUM` Konsistenz verwendet, und es fallen keine zusätzlichen Gebühren für die Verwendung von Lightweight Transactions (LWTs) an. Wenn Sie eine Zeile schreiben müssen, die größer als 1 KB ist, verwendet der Schreibvorgang zusätzliche WCUs.

  Die Gesamtzahl der benötigten WCUs hängt von der Zeilengröße ab. Wenn Ihre Zeilengröße beispielsweise 2 KB beträgt, benötigen Sie 2 WCUs, um eine Schreibanforderung pro Sekunde aufrechtzuerhalten. Weitere Hinweise zur Schätzung des Lese- und Schreibkapazitätsverbrauchs einer Tabelle finden Sie unterthe section called “Schätzen Sie den Kapazitätsverbrauch”.

Wenn Ihre Anwendung größere Zeilen liest oder schreibt (bis zur maximalen Zeilengröße von Amazon Keyspaces von 1 MB), verbraucht sie mehr Kapazitätseinheiten. Weitere Informationen zur Schätzung der Zeilengröße finden Sie unter. the section called “Schätzen Sie die Zeilengröße” Nehmen wir beispielsweise an, Sie erstellen eine bereitgestellte Tabelle mit 6 RCUs und 6 WCUs. Mit diesen Einstellungen hat Ihre Anwendung folgende Möglichkeiten:

- Führen Sie `LOCAL_QUORUM` Lesevorgänge mit bis zu 24 KB pro Sekunde (4 KB × 6 RCUs) durch.
- Führen Sie `LOCAL_ONE` Lesevorgänge mit bis zu 48 KB pro Sekunde durch (doppelt so viel Lesedurchsatz).
- Schreiben Sie bis zu 6 KB pro Sekunde (1 KB × 6 WCUs).

Der bereitgestellte Durchsatz ist die maximale Menge an Durchsatzkapazität, die eine Anwendung aus einer Tabelle verbrauchen kann. Wenn Ihre Anwendung die bereitgestellte Durchsatzkapazität überschreitet, treten möglicherweise Fehler bei unzureichender Kapazität auf.

Beispielsweise schlägt eine Leseanforderung, die nicht über genügend Durchsatzkapazität verfügt, mit einer `Read_Timeout` Ausnahme fehl und wird der `ReadThrottleEvents` Metrik zugewiesen. Eine Schreibanforderung, die nicht über genügend Durchsatzkapazität verfügt, schlägt mit einer `Write_Timeout` Ausnahme fehl und wird der `WriteThrottleEvents` Metrik zugewiesen.

Sie können Amazon verwenden CloudWatch , um Ihre bereitgestellten und tatsächlichen Durchsatzmetriken sowie Ereignisse mit unzureichender Kapazität zu überwachen. Weitere Informationen zu diesen Metriken finden Sie unter the section called “Metriken und Dimensionen”.

> **Note**
>
> Wiederholte Fehler aufgrund unzureichender Kapazität können zu clientseitigen treiberspezifischen Ausnahmen führen, z. B. schlägt der DataStax Java-Treiber mit einem fehl. `NoHostAvailableException`

Um die Durchsatzkapazitätseinstellungen für Tabellen zu ändern, können Sie die Anweisung AWS-Managementkonsole oder die `ALTER TABLE` Anweisung mithilfe von CQL verwenden. Weitere Informationen finden Sie unter. the section called “ALTER TABLE”

Weitere Informationen zu Standardkontingenten für Ihr Konto und deren Erhöhung finden Sie unterKontingente.

Den Kapazitätsmodus einer Tabelle in Amazon Keyspaces anzeigen

Sie können die Systemtabelle im Amazon Keyspaces-Systemschlüsselraum abfragen, um Informationen zum Kapazitätsmodus einer Tabelle zu überprüfen. Sie können auch sehen, ob eine Tabelle den On-Demand-Durchsatzkapazitätsmodus oder den bereitgestellten Durchsatzkapazitätsmodus verwendet. Wenn die Tabelle mit dem Modus „Bereitgestellte Durchsatzkapazität“ konfiguriert ist, können Sie die für die Tabelle bereitgestellte Durchsatzkapazität sehen.

Sie können den auch verwenden AWS CLI , um den Kapazitätsmodus einer Tabelle anzuzeigen.

Informationen zum Ändern des bereitgestellten Durchsatzes einer Tabelle finden Sie unterthe section called “Ändern Sie den Kapazitätsmodus”.

Cassandra Query Language (CQL)

Beispiel

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'mykeyspace' and
 table_name = 'mytable';
```

Eine Tabelle, die mit dem On-Demand-Kapazitätsmodus konfiguriert ist, gibt Folgendes zurück.

```
{
   "capacity_mode":{
      "last_update_to_pay_per_request_timestamp":"1579551547603",
      "throughput_mode":"PAY_PER_REQUEST"
   }
}
```

Eine mit dem Modus „Bereitgestellte Durchsatzkapazität“ konfigurierte Tabelle gibt Folgendes zurück.

```
{
   "capacity_mode":{
      "last_update_to_pay_per_request_timestamp":"1579048006000",
      "read_capacity_units":"5000",
      "throughput_mode":"PROVISIONED",
      "write_capacity_units":"6000"
   }
}
```

Der `last_update_to_pay_per_request_timestamp` Wert wird in Millisekunden gemessen.

CLI

Zeigen Sie den Durchsatzkapazitätsmodus einer Tabelle an, indem Sie AWS CLI

```
aws keyspaces get-table --keyspace-name myKeyspace --table-name myTable
```

Die Ausgabe des Befehls kann für eine Tabelle im Modus „Bereitgestellte Kapazität“ ähnlich aussehen.

```
"capacitySpecification": {
        "throughputMode": "PROVISIONED",
```

```
        "readCapacityUnits": 4000,
        "writeCapacityUnits": 2000
    }
```

Die Ausgabe für eine Tabelle im On-Demand-Modus sieht wie folgt aus.

```
"capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
    }
```

Ändern Sie den Kapazitätsmodus einer Tabelle in Amazon Keyspaces

Wenn Sie eine Tabelle vom Bereitstellungskapazitätsmodus in den On-Demand-Kapazitätsmodus wechseln, nimmt Amazon Keyspaces mehrere Änderungen an der Struktur Ihrer Tabelle und Partitionen vor. Dieser Vorgang kann einige Minuten dauern. Während der Umstellungsphase liefert Ihre Tabelle einen Durchsatz, der den zuvor bereitgestellten WCU- und RCU-Mengen entspricht.

Wenn Sie vom On-Demand-Kapazitätsmodus zurück in den Modus mit bereitgestellter Kapazität wechseln, liefert Ihre Tabelle einen Durchsatz, der dem vorherigen Spitzenwert entspricht, der erreicht wurde, als die Tabelle in den On-Demand-Kapazitätsmodus versetzt wurde.

Beim Umschalten zwischen den Kapazitätsmodi gelten die folgenden Wartezeiten:

- Sie können eine neu erstellte Tabelle im On-Demand-Modus jederzeit in den Modus für bereitgestellte Kapazität umschalten. Sie können sie jedoch erst 24 Stunden nach dem Erstellungszeitstempel der Tabelle wieder in den On-Demand-Modus zurückschalten.
- Sie können eine bestehende Tabelle im On-Demand-Modus jederzeit in den Modus für bereitgestellte Kapazität umschalten. Sie können den Kapazitätsmodus jedoch nur einmal innerhalb von 24 Stunden vom Bereitstellungs- zum On-Demand-Modus wechseln.

Cassandra Query Language (CQL)

Ändern Sie den Durchsatzkapazitätsmodus einer Tabelle mithilfe von CQL

1. Um den Kapazitätsmodus einer Tabelle in zu ändern, müssen `PROVISIONED` Sie die Lese- und Schreibkapazitätseinheiten auf der Grundlage der erwarteten Spitzenwerte Ihrer Workloads konfigurieren. Die folgende Aussage ist ein Beispiel dafür. Sie können diese

Anweisung auch ausführen, um die Lese- oder Schreibkapazitätseinheiten der Tabelle anzupassen.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 6000,
 'write_capacity_units': 3000}};
```

Informationen zur Konfiguration des Modus für bereitgestellte Kapazität mit auto-scaling finden Sie unter. the section called “Konfigurieren Sie die automatische Skalierung für eine bestehende Tabelle”

2. Um den Kapazitätsmodus einer Tabelle in den On-Demand-Modus zu ändern, stellen Sie den Durchsatzmodus auf ein. `PAY_PER_REQUEST` Die folgende Aussage ist ein Beispiel dafür.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PAY_PER_REQUEST'}};
```

3. Sie können die folgende Anweisung verwenden, um den Kapazitätsmodus der Tabelle zu bestätigen.

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'catalog' and
 table_name = 'book_awards';
```

Eine Tabelle, die mit dem On-Demand-Kapazitätsmodus konfiguriert ist, gibt Folgendes zurück.

```
{
   "capacity_mode":{
      "last_update_to_pay_per_request_timestamp":"1727952499092",
      "throughput_mode":"PAY_PER_REQUEST"
   }
}
```

Der `last_update_to_pay_per_request_timestamp` Wert wird in Millisekunden gemessen.

CLI

Ändern Sie den Durchsatzkapazitätsmodus einer Tabelle mithilfe des AWS CLI

1. Um den Kapazitätsmodus der Tabelle auf zu ändern, müssen `PROVIOSIONED` Sie die Lese- und Schreibkapazitätseinheiten auf der Grundlage der erwarteten Spitzenwerte Ihrer Arbeitslast konfigurieren. Der folgende Befehl ist ein Beispiel dafür. Sie können diesen Befehl auch ausführen, um die Lese- oder Schreibkapazitätseinheiten der Tabelle anzupassen.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards
                                    \--capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=6000,writeCapacityUnits=3000
```

   Informationen zur Konfiguration des Modus für bereitgestellte Kapazität mit auto-scaling finden Sie unter. the section called “Konfigurieren Sie die automatische Skalierung für eine bestehende Tabelle”

2. Um den Kapazitätsmodus einer Tabelle in den On-Demand-Modus zu ändern, stellen Sie den Durchsatzmodus auf ein. `PAY_PER_REQUEST` Die folgende Aussage ist ein Beispiel dafür.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards
                                    \--capacity-specification
 throughputMode=PAY_PER_REQUEST
```

3. Sie können den folgenden Befehl verwenden, um den Kapazitätsmodus zu überprüfen, der für eine Tabelle konfiguriert ist.

```
aws keyspaces get-table --keyspace-name catalog --table-name book_awards
```

   Die Ausgabe für eine Tabelle im On-Demand-Modus sieht wie folgt aus.

```
"capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
    }
```

Vorwärmung für Tabellen in Amazon Keyspaces konfigurieren

Amazon Keyspaces skaliert Speicherpartitionen automatisch auf der Grundlage des On-Demand-Durchsatzes oder des bereitgestellten Durchsatzes. Bei neuen Tabellen oder plötzlichen Durchsatzspitzen kann es jedoch länger dauern, bis die erforderlichen Speicherpartitionen zugewiesen sind. Um sicherzustellen, dass eine neue oder bestehende Tabelle über genügend Kapazität verfügt, um den erwarteten Spitzendurchsatz zu unterstützen, können Sie manuell bestimmte Warmdurchsatzwerte festlegen, um Ihre Tabelle vorzuwärmen.

Der warme Durchsatz bezieht sich auf die Anzahl der Lese- und Schreibvorgänge, die Ihre Amazon Keyspaces-Tabelle sofort unterstützen kann. Diese Werte sind standardmäßig für alle neuen und vorhandenen Tabellen verfügbar. Wenn Sie den On-Demand-Modus verwenden oder Ihren bereitgestellten Durchsatz aktualisieren, stellt Amazon Keyspaces sicher, dass Ihre Anwendung Anfragen bis zu diesen Werten sofort ausgeben kann.

Amazon Keyspaces passt die Warmdurchsatzwerte automatisch an, wenn Ihre Nutzung zunimmt. Um die Durchsatzkapazität an bevorstehende Spitzenereignisse anzupassen, z. B. wenn Sie Daten aus einer anderen Datenbank migrieren, was das Laden von Terabyte an Daten in kurzer Zeit erfordern kann, können Sie die Warmdurchsatzwerte Ihrer Tabellen manuell erhöhen. Dies ist nützlich für geplante Spitzenereignisse, bei denen die Anforderungsraten um das Zehnfache, das Hundertfache oder mehr steigen könnten. Prüfen Sie zunächst, ob der aktuelle Durchsatz im Warmbetrieb ausreicht, um den zu erwartenden Verkehr zu bewältigen. Wenn Sie dann die Tabelle für die geplante Spitzenauslastung vorwärmen müssen, können Sie den Wert für den Warmdurchsatz manuell erhöhen, ohne Ihre Durchsatzeinstellungen oder den Kapazitätsmodus zu ändern.

Sie können Tabellen für Lesevorgänge, Schreibvorgänge oder beides vorwärmen. Sie können diesen Wert für neue und bestehende Tabellen mit einer Region und Tabellen mit mehreren Regionen erhöhen. Die von Ihnen festgelegten Einstellungen für den Warmdurchsatz gelten dann automatisch für alle Replikate der Tabellen mit mehreren Regionen. Die Anzahl der Amazon Keyspaces-Tabellen, die Sie jederzeit vorwärmen können, ist unbegrenzt. Die Zeit bis zum Abschluss des Vorwärmens hängt von den von Ihnen eingestellten Werten und der Größe des Tisches ab. Sie können gleichzeitig Anfragen zum Vorwärmen einreichen, ohne dass diese Anfragen den Betrieb der Tabelle beeinträchtigen. Sie können Ihren Tisch vorwärmen, bis das Tischkontingentlimit für Ihr Konto in dieser Region erreicht ist. Verwenden Sie die Service Quotas Quotas-Konsole, um Ihre aktuellen Kontingente zu überprüfen und sie bei Bedarf zu erhöhen.

Die Warmdurchsatzwerte, die Amazon Keyspaces auf der Grundlage Ihrer On-Demand-Nutzung oder der bereitgestellten Kapazität anpasst, sind standardmäßig für alle Tabellen ohne zusätzliche

Gebühren verfügbar. Wenn Sie jedoch die Standardwerte für den Warmdurchsatz manuell auf Tabellen für die Vorwärmphase für Ereignisse mit hohem Verkehrsaufkommen erhöhen, fallen zusätzliche Gebühren an. Weitere Informationen finden Sie unter Amazon Keyspaces-Preise.

Im Folgenden finden Sie einige verschiedene Szenarien und bewährte Methoden, die Sie beim Vorwärmen von Amazon Keyspaces-Tabellen berücksichtigen könnten.

Warmdurchsatz und ungleiche Zugriffsmuster

Eine Tabelle kann einen warmen Durchsatz von 30.000 Leseeinheiten pro Sekunde und 10.000 Schreibeinheiten pro Sekunde haben, aber es kann trotzdem zu Kapazitätsüberschreitungen bei Lese- oder Schreibvorgängen kommen, bevor diese Werte erreicht werden. Das liegt wahrscheinlich an einer Hot-Partition. Amazon Keyspaces kann zwar weiter skaliert werden, um praktisch unbegrenzten Durchsatz zu unterstützen, aber jede einzelne Partition ist auf 1.000 Schreibeinheiten pro Sekunde und 3.000 Leseeinheiten pro Sekunde begrenzt. Wenn Ihre Anwendung zu viel Traffic auf einen kleinen Teil der Tabellenpartitionen lenkt, kann es zu Kapazitätsüberschreitungen kommen, noch bevor Sie die warmen Durchsatzwerte der Tabelle erreichen. Wir empfehlen, die Best Practices von Amazon Keyspaces zu befolgen, um eine nahtlose Skalierbarkeit zu gewährleisten und heiße Partitionen zu vermeiden.

Warmdurchsatz für eine bereitgestellte Tabelle

Stellen Sie sich eine bereitgestellte Tabelle vor, die einen Warmdurchsatz von 30.000 Leseeinheiten pro Sekunde und 10.000 Schreibeinheiten pro Sekunde hat, derzeit jedoch einen bereitgestellten Durchsatz von 4.000 RCUs und 8.000 WCUs hat. Sie können den bereitgestellten Durchsatz der Tabelle sofort auf bis zu 30.000 RCUs oder 10.000 WCUs skalieren, indem Sie Ihre bereitgestellten Durchsatzeinstellungen aktualisieren. Wenn Sie den bereitgestellten Durchsatz über diese Werte hinaus erhöhen, passt sich der Warmdurchsatz automatisch an die neuen höheren Werte an, da Sie einen neuen Spitzendurchsatz festgelegt haben. Wenn Sie beispielsweise den bereitgestellten Durchsatz auf 50.000 RCU festlegen, steigt der Durchsatz im Warmbetrieb auf 50.000 Leseeinheiten pro Sekunde.

```
"ProvisionedThroughput":
    {
        "ReadCapacityUnits": 4000,
        "WriteCapacityUnits": 8000
    }
"WarmThroughput":
    {
```

```
        "ReadUnitsPerSecond": 30000,
        "WriteUnitsPerSecond": 10000
    }
```

Warmdurchsatz für eine On-Demand-Tabelle

Eine neue On-Demand-Tabelle beginnt mit einem Warmdurchsatz von 12 000 Leseeinheiten pro Sekunde und 4 000 Schreibeinheiten pro Sekunde. Ihre Tabelle kann anhaltenden Datenverkehr bis zu diesen Werten sofort verarbeiten. Wenn Ihre Anfragen 12.000 Leseeinheiten pro Sekunde oder 4.000 Schreibeinheiten pro Sekunde überschreiten, passt sich der Warmdurchsatz automatisch an die höheren Werte an.

```
"WarmThroughput":
    {
        "ReadUnitsPerSecond": 12000,
        "WriteUnitsPerSecond": 4000
    }
```

Bewährte Methoden für das Vorwärmen von Amazon Keyspaces-Tabellen

Folgen Sie diesen bewährten Methoden, wenn Sie Pre-Warming für Ihre Amazon Keyspaces-Tabellen implementieren:

Schätzen Sie die benötigte Kapazität genau ab

Da das Vorwärmen mit einmaligen Kosten verbunden ist, sollten Sie den benötigten Durchsatz anhand der zu erwartenden Arbeitslast sorgfältig berechnen, um eine übermäßige Bereitstellung zu vermeiden.

Betrachten Sie das Schema der Tabelle

Tabellen mit größeren Zeilen benötigen möglicherweise mehr Partitionen für denselben Durchsatz. Berücksichtigen Sie bei der Schätzung des Bedarfs vor der Erwärmung Ihre durchschnittliche Zeilengröße.

Überwachen Sie die Tabellenleistung

Überprüfen Sie nach dem Vorwärmen anhand von CloudWatch Messwerten, ob Ihre Tabelle die Last erwartungsgemäß bewältigt. Weitere Informationen finden Sie unter the section called “Überwachen Sie die Vorwärmung mit CloudWatch”.

Kontingente verwalten

Wenn Ihre Anwendung einen höheren Durchsatz benötigt, als es die Standardkontingente zulassen (40.000 RCUs/WCUs oder 2.000 Partitionen), wird das Anforderungskontingent rechtzeitig vor dem Ereignis mit hohem Datenaufkommen erhöht. Zum Anfordern einer Erhöhung für ein Kontingent können Sie die Service-Quotas-Konsole verwenden.

Optimieren Sie die Kosten

Bei temporären Ereignissen mit hohem Datenaufkommen sollten Sie die Verwendung von Pre-Warming in Betracht ziehen, anstatt in den Bereitstellungsmodus mit hoher Kapazität zu wechseln, da dies für Ereignisse mit kurzer Dauer kostengünstiger sein kann. Weitere Informationen zur Preisgestaltung finden Sie unter Amazon Keyspaces-Preise.

> **Note**
>
> Überwachen Sie die Leistungskennzahlen Ihrer Anwendung während der Testphase, um sicherzustellen, dass Ihre Pre-Warming-Konfiguration Ihre Workload-Anforderungen angemessen unterstützt.

Themen

- Erstellen einer neuen Amazon-Keyspaces-Tabelle mit einem höheren Warmdurchsatz
- Erhöhen des Warmdurchsatzes für die vorhandene Amazon-Keyspaces-Tabelle
- Warmdurchsatz einer Amazon Keyspaces-Tabelle anzeigen
- Überwachen Sie die Leistung eines vorgewärmten Tisches mit Amazon CloudWatch

Erstellen einer neuen Amazon-Keyspaces-Tabelle mit einem höheren Warmdurchsatz

Sie können die Warmdurchsatzwerte anpassen, wenn Sie Ihre Amazon Keyspaces-Tabelle mithilfe der Konsole, CQL oder der erstellen. AWS CLI

Console

So erstellen Sie eine neue Tabelle mit Warmdurchsatzeinstellungen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.

2. Wählen Sie im Navigationsbereich Tables (Tabellen) und anschließend Create table (Tabelle erstellen) aus.
3. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die neue Tabelle ein.
4. Erstellen Sie im Abschnitt Spalten das Schema für Ihre Tabelle.
5. Definieren Sie im Abschnitt Primärschlüssel den Primärschlüssel der Tabelle und wählen Sie optionale Clusterspalten aus.
6. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.
7. Fahren Sie mit den Read/write Kapazitätseinstellungen fort.
8. Für den Kapazitätsmodus können Sie entweder On-demandoder Bereitgestellt wählen.
9. Im Abschnitt Pre-warming für Tabellen können Sie die Werte für Leseeinheiten pro Sekunde und Schreibeinheiten pro Sekunde nach Bedarf erhöhen, um Ihre Tabelle auf geplante Spitzenereignisse vorzubereiten.

   Die Warmdurchsatzwerte, die Amazon Keyspaces auf der Grundlage Ihrer On-Demand-Nutzung oder der bereitgestellten Kapazität anpasst, sind standardmäßig für alle Tabellen ohne zusätzliche Gebühren verfügbar. Beachten Sie, dass zusätzliche Gebühren anfallen, wenn Sie die Standardwerte für den Warmdurchsatz manuell erhöhen, um die Tabelle für Ereignisse mit hohem Verkehrsaufkommen vorzuwärmen.
10. Konfigurieren Sie nach Bedarf weitere optionale Tabellenfunktionen. Wählen Sie dann Tabelle erstellen aus.

Cassandra Query Language (CQL)

- Erstellen Sie mit einer der folgenden Methoden eine Tabelle mit warmem Durchsatz:
  - Erstellen Sie für den Bereitstellungsmodus eine Tabelle und geben Sie die erwartete Spitzenkapazität für Lese- und Schreibvorgänge mithilfe der folgenden CQL-Syntax an:

```
CREATE TABLE catalog.book_awards (
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
```

```
   PRIMARY KEY ((year, award), category, rank))
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
       'throughput_mode': 'PROVISIONED',
       'read_capacity_units': 20000,
       'write_capacity_units': 10000
     },
    'warm_throughput': {
        'read_units_per_second': 40000,
        'write_units_per_second': 20000
     }
};
```

- Erstellen Sie für den On-Demand-Modus eine Tabelle und geben Sie die erwartete Spitzenkapazität für Lese- und Schreibvorgänge mit der folgenden CQL-Syntax an:

```
CREATE TABLE catalog.book_awards (
   year int,
   award text,
   rank int,
   category text,
   book_title text,
   author text,
   publisher text,
   PRIMARY KEY ((year, award), category, rank))
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
       'throughput_mode': 'PAY_PER_REQUEST'
     },
    'warm_throughput': {
        'read_units_per_second': 40000,
        'write_units_per_second': 20000
     }
};
```

Informationen zur Bestätigung der Kapazitätseinstellungen der Tabelle finden Sie unterthe section called “Sehen Sie sich den Warmdurchsatz Ihrer Tabelle an”.

CLI

1. Erstellen Sie eine Tabelle mit warmem Durchsatz mit einer der folgenden Methoden unter Verwendung von AWS CLI

- Erstellen Sie eine neue Tabelle im Bereitstellungsmodus und geben Sie die erwarteten Spitzenkapazitätswerte für Lese- und Schreibvorgänge für die neue Tabelle an. Die folgende Aussage ist ein Beispiel dafür.

```
aws keyspaces create-table \
--keyspace-name 'catalog' \
--table-name 'book_awards' \
--schema-definition 'allColumns=[{name=year,type=int},
{name=award,type=text},{name=rank,type=int},{name=category,type=text},
{name=book_title,type=text},{name=author,type=text},
{name=publisher,type=text}],partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},
{name=rank,orderBy=ASC}]' \
--capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=20000,writeCapacityUnits=10000
 \
--warm-throughput-specification
 readUnitsPerSecond=40000,writeUnitsPerSecond=20000
```

- Erstellen Sie eine neue Tabelle im On-Demand-Modus und geben Sie die erwarteten Spitzenkapazitätswerte für Lese- und Schreibvorgänge für die neue Tabelle an. Die folgende Aussage ist ein Beispiel dafür.

```
aws keyspaces create-table \
--keyspace-name 'catalog' \
--table-name 'book_awards' \
--schema-definition 'allColumns=[{name=year,type=int},
{name=award,type=text},{name=rank,type=int},{name=category,type=text},
{name=book_title,type=text},{name=author,type=text},
{name=publisher,type=text}],partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},
{name=rank,orderBy=ASC}]' \
--warmThroughputSpecification
 readUnitsPerSecond=40000,writeUnitsPerSecond=20000
```

2. Die Ausgabe des Befehls gibt den ARN der Tabelle zurück, wie im folgenden Beispiel gezeigt.

```
{
    "resourceArn": "arn:aws::cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards>"
```

```
}
```

Informationen zur Bestätigung der Kapazitätseinstellungen der Tabelle finden Sie unterthe section called “Sehen Sie sich den Warmdurchsatz Ihrer Tabelle an”.

Java

Erstellen Sie eine neue Tabelle mit dem SDK for Java.

- Erstellen Sie eine neue Tabelle im Bereitstellungsmodus und geben Sie die erwarteten Spitzenkapazitätswerte für Lese- und Schreibvorgänge für die neue Tabelle an. Das folgende Codebeispiel ist ein Beispiel dafür.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.*;

public class PreWarmingExample {
    public static void main(String[] args) {
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();

        // Define schema
        List<ColumnDefinition> columns = Arrays.asList(
            ColumnDefinition.builder().name("year").type("int").build(),
            ColumnDefinition.builder().name("award").type("text").build(),
            ColumnDefinition.builder().name("rank").type("int").build(),
            ColumnDefinition.builder().name("category").type("text").build(),
            ColumnDefinition.builder().name("book_title").type("text").build(),
            ColumnDefinition.builder().name("author").type("text").build(),
            ColumnDefinition.builder().name("publisher").type("text").build()
        );

        List<PartitionKey> partitionKeys = Arrays.asList(
            PartitionKey.builder().name("year").build(),
            PartitionKey.builder().name("award").build()
        );

        List<ClusteringKey> clusteringKeys = Arrays.asList(
            ClusteringKey.builder().name("category").orderBy("ASC").build(),
            ClusteringKey.builder().name("rank").orderBy("ASC").build()
        );

        SchemaDefinition schema = SchemaDefinition.builder()
```

```
            .allColumns(columns)
            .partitionKeys(partitionKeys)
            .clusteringKeys(clusteringKeys)
            .build();

        // Define capacity specification
        CapacitySpecification capacitySpec = CapacitySpecification.builder()
            .throughputMode(ThroughputMode.PROVISIONED)
            .readCapacityUnits(20000)
            .writeCapacityUnits(10000)
            .build();

        // Define warm throughput specification
        WarmThroughputSpecification warmThroughput =
 WarmThroughputSpecification.builder()
            .readUnitsPerSecond(40000L)
            .writeUnitsPerSecond(20000L)
            .build();

        // Create table with PreWarming
        CreateTableRequest request = CreateTableRequest.builder()
            .keyspaceName("catalog")
            .tableName("book_awards")
            .schemaDefinition(schema)
            .capacitySpecification(capacitySpec)
            .warmThroughputSpecification(warmThroughput)
            .build();

        CreateTableResponse response = keyspacesClient.createTable(request);
        System.out.println("Table created with ARN: " + response.resourceArn());
    }
}
```

Erhöhen des Warmdurchsatzes für die vorhandene Amazon-Keyspaces-Tabelle

Sie können die aktuellen Warmdurchsatzwerte Ihrer Amazon Keyspaces-Tabelle mithilfe der Konsole, CQL oder der erhöhen. AWS CLI

Console

So erhöhen Sie die Vorwärmeinstellungen einer Tabelle mithilfe der Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tabellen und dann die Tabelle aus, die Sie aktualisieren möchten.
3. Fahren Sie auf der Registerkarte Kapazität der Tabelle mit Pre-warming Für Tabellen fort.
4. Wählen Sie im Abschnitt Pre-warming für Tabellen die Option Bearbeiten aus.
5. Auf der Seite Vorwärmen für Tabellen bearbeiten können Sie die Werte für Leseeinheiten pro Sekunde und für Schreibeinheiten pro Sekunde aktualisieren.
6. Wählen Sie Änderungen speichern aus. Ihre Tabelle wird mit den angegebenen Vorwärmeinstellungen aktualisiert.

Cassandra Query Language (CQL)

Erhöhen Sie die Warmdurchsatzeinstellungen einer Tabelle mithilfe von CQL

- Verwenden Sie die `ALTER TABLE` Anweisung, um den Warmdurchsatz einer Tabelle zu erhöhen. Die folgende Aussage ist ein Beispiel dafür.

```
ALTER TABLE catalog.book_awards
WITH CUSTOM_PROPERTIES = {
    'warm_throughput': {
        'read_units_per_second': 60000,
        'write_units_per_second': 30000
    }
};
```

Informationen zur Bestätigung der aktualisierten Kapazitätseinstellungen der Tabelle finden Sie unterthe section called “Sehen Sie sich den Warmdurchsatz Ihrer Tabelle an”.

CLI

Erhöhen Sie die Vorwärmeinstellungen eines Tisches mithilfe der AWS CLI

- Um den Warmdurchsatz einer Tabelle zu erhöhen, können Sie den Befehl verwenden. `update-table` Die folgende Aussage ist ein Beispiel dafür.

```
aws keyspaces update-table \
--keyspace-name 'catalog' \
--table-name 'book_awards' \
--warmThroughputSpecification readUnitsPerSecond=60000,writeUnitsPerSecond=30000
```

Informationen zur Bestätigung der aktualisierten Kapazitätseinstellungen der Tabelle finden Sie unterthe section called “Sehen Sie sich den Warmdurchsatz Ihrer Tabelle an”.

Java

Aktualisieren Sie die Vorwärmeinstellungen einer Tabelle mithilfe des SDK for Java.

- Aktualisieren Sie die Warmdurchsatzeinstellungen für eine Tabelle. Das folgende Codebeispiel ist ein Beispiel dafür.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.*;

public class UpdatePreWarmingExample {
    public static void main(String[] args) {
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();

        // Define new warm throughput specification
        WarmThroughputSpecification warmThroughput =
 WarmThroughputSpecification.builder()
            .readUnitsPerSecond(60000L)
            .writeUnitsPerSecond(30000L)
            .build();

        // Update table with new PreWarming settings
        UpdateTableRequest request = UpdateTableRequest.builder()
            .keyspaceName("catalog")
            .tableName("book_awards")
            .warmThroughputSpecification(warmThroughput)
```

```
            .build();

        UpdateTableResponse response = keyspacesClient.updateTable(request);
        System.out.println("Table update requested: " + response.resourceArn());
    }
}
```

Warmdurchsatz einer Amazon Keyspaces-Tabelle anzeigen

Sie können die aktuellen Warmdurchsatzwerte Ihrer Amazon Keyspaces-Tabelle mit der Konsole, CQL oder dem anzeigen. AWS CLI

Console

So zeigen Sie die Vorwärmeinstellungen Ihres Tisches mithilfe der Konsole an.

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tabellen und dann die Tabelle aus, die Sie überprüfen möchten.
3. Fahren Sie auf der Registerkarte Kapazität der Tabelle mit Pre-warming Für Tabellen fort.

Cassandra Query Language (CQL)

Zeigen Sie die Warmdurchsatz-Einstellungen einer Tabelle mithilfe von CQL an

- Um die Warmdurchsatzeinstellungen einer Tabelle anzuzeigen, können Sie die folgende CQL-Anweisung verwenden.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name='catalog' and table_name='book_awards';

// Output:
...
custom_properties
--------------------------------------------------------------------------------
{
    'warm_throughput':
    {
```

```
        'read_units_per_second': '40000',
        'write_units_per_second': '20000',
        'status': 'AVAILABLE'
    }
}
...
```

CLI

Zeigen Sie die Warmdurchsatz-Einstellungen einer Tabelle mit dem AWS CLI

- Sie können die Warmdurchsatz-Einstellungen einer Tabelle mithilfe des `get-table` Befehls anzeigen, wie im folgenden Beispiel gezeigt.

```
aws keyspaces get-table \
--keyspace-name 'catalog' \
--table-name 'book_awards'
```

Im Folgenden wird die Beispielausgabe des `get-table` Befehls für eine Tabelle mit nur einer Region im Bereitstellungsmodus gezeigt.

```
{
    "keyspaceName": "catalog",
    "tableName": "book_awards",
    ... Existing Fields ...,
    "capacitySpecificationSummary": {
        "throughputMode": "PROVISIONED",
        "readCapacityUnits": 20000,
        "writeCapacityUnits": 10000
    },
    "warmThroughputSpecificationSummary": {
        "readUnitsPerSecond": 40000,
        "writeUnitsPerSecond": 20000,
        "status": "AVAILABLE"
    }
}
```

Im Folgenden wird die Beispielausgabe für eine Tabelle mit einer einzigen Region im On-Demand-Modus gezeigt.

```
{
    "keyspaceName": "catalog",
    "tableName": "book_awards_ondemand",
    ... Existing Fields ...,
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST"
    },
    "warmThroughputSpecificationSummary": {
        "readUnitsPerSecond": 40000,
        "writeUnitsPerSecond": 20000,
        "status": "AVAILABLE"
    }
}
```

Java

Lesen Sie die Vorwärmeinstellungen einer Tabelle mit dem SDK for Java.

- Lesen Sie die Warmdurchsatzwerte einer Tabelle mit. `get-table` Das folgende Codebeispiel ist ein Beispiel dafür.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.*;

public class GetTableWithPreWarmingExample {
    public static void main(String[] args) {
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();

        // Get table details including PreWarming specification
        GetTableRequest request = GetTableRequest.builder()
            .keyspaceName("catalog")
            .tableName("book_awards")
            .build();

        GetTableResponse response = keyspacesClient.getTable(request);

        // Access PreWarming details
        if (response.warmThroughputSpecification() != null) {
            WarmThroughputSpecificationSummary warmThroughputSummary =
 response.warmThroughputSpecification();
```

```
            System.out.println("PreWarming Status: " +
 warmThroughputSummary.status());
            System.out.println("Read Units: " +
 warmThroughputSummary.readUnitsPerSecond());
            System.out.println("Write Units: " +
 warmThroughputSummary.writeUnitsPerSecond());

            // Check if PreWarming is active
            if (warmThroughputSummary.status().equals("AVAILABLE")) {
                System.out.println("Table is fully pre-warmed and ready for high
 throughput");
            } else if (warmThroughputSummary.status().equals("UPDATING")) {
                System.out.println("Table PreWarming is currently being
 updated");
            }
        } else {
            System.out.println("Table does not have PreWarming enabled");
        }
    }
}
```

Überwachen Sie die Leistung eines vorgewärmten Tisches mit Amazon CloudWatch

Amazon Keyspaces Pre-Warming führt keine neuen CloudWatch Metriken ein, aber Sie können die Leistung vorgewärmter Tabellen anhand vorhandener Amazon Keyspaces-Metriken überwachen:

SuccessfulRequestLatency

Überwachen Sie diese Metrik, um sicherzustellen, dass die vorgewärmte Tabelle Anfragen mit erwarteter Latenz verarbeitet.

WriteThrottleEvents und ReadThrottleEvents

Diese Messwerte sollten für eine ordnungsgemäß vorgewärmte Tabelle niedrig bleiben. Wenn Sie trotz Vorwärmen Fehler mit unzureichender Kapazität feststellen, müssen Sie möglicherweise Ihre Werte für den Warmdurchsatz anpassen.

ConsumedReadCapacityUnits und ConsumedWriteCapacityUnits

Diese Messwerte zeigen den tatsächlichen Kapazitätsverbrauch, sodass Sie überprüfen können, ob Ihre Vorwärmkonfiguration angemessen ist.

ProvisionedReadCapacityUnits und ProvisionedWriteCapacityUnits

Für bereitgestellte Tabellen zeigen diese Metriken die aktuell zugewiesene Kapazität.

Diese Metriken können in der CloudWatch Konsole angezeigt oder über die API abgefragt werden. CloudWatch Weitere Informationen finden Sie unter the section called “Überwachung mit CloudWatch”.

Automatische Verwaltung der Durchsatzkapazität mit Amazon Keyspaces Auto Scaling

Viele Datenbank-Workloads sind zyklischer Natur oder lassen sich nur schwer vorhersagen. Nehmen wir als Beispiel eine Social Networking-App, bei der die meisten Benutzer tagsüber aktiv sind. Die Datenbank muss in der Lage sein, die Aktivitäten am Tag zu verarbeiten, während in der Nacht nicht so viel Durchsatz erforderlich ist.

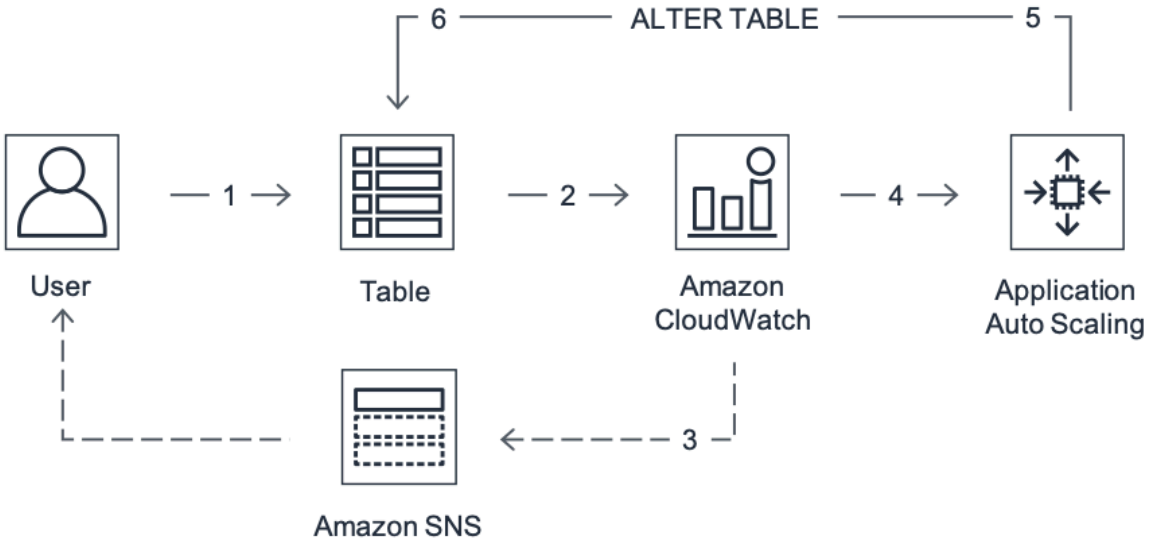
Ein weiteres Beispiel wäre eine neue Mobile Gaming-App, die sich schnell großer Beliebtheit erfreut. Wenn das Spiel sehr beliebt wird, könnte es die verfügbaren Datenbankressourcen überschreiten, was zu einer langsamen Leistung und unzufriedenen Kunden führen würde. Diese Arten von Workloads erfordern häufig manuelle Eingriffe, um die Datenbankressourcen nach oben oder unten zu skalieren, um sie an die jeweiligen Nutzungsschwankungen anzupassen.

Amazon Keyspaces (für Apache Cassandra) hilft Ihnen dabei, Durchsatzkapazität für variable Workloads effizient bereitzustellen, indem die Durchsatzkapazität automatisch an den tatsächlichen Anwendungsverkehr angepasst wird. Amazon Keyspaces verwendet den Application Auto Scaling-Service, um die Lese- und Schreibkapazität einer Tabelle in Ihrem Namen zu erhöhen und zu verringern. Weitere Informationen zu Application Auto Scaling finden Sie im Application Auto Scaling-Benutzerhandbuch.

> **Note**
>
> Informationen zum schnellen Einstieg in die automatische Skalierung von Amazon Keyspaces finden Sie unterthe section called “Auto Scaling-Richtlinien konfigurieren und aktualisieren”.

So funktioniert die automatische Skalierung von Amazon Keyspaces

Das folgende Diagramm bietet einen allgemeinen Überblick darüber, wie die automatische Skalierung von Amazon Keyspaces die Durchsatzkapazität für eine Tabelle verwaltet.



Um die automatische Skalierung für eine Tabelle zu aktivieren, erstellen Sie eine Skalierungsrichtlinie. Die Skalierungsrichtlinie gibt an, ob die Lese- oder Schreibkapazität (oder beide) skaliert werden sollen, und enthält die Einstellungen für die minimalen und maximalen bereitgestellten Kapazitätseinheiten für die Tabelle.

Die Skalierungsrichtlinie definiert auch eine Zielauslastung. Die Zielauslastung bezeichnet das Verhältnis zwischen den verbrauchten Kapazitätseinheiten und den bereitgestellten Kapazitätseinheiten zu einem bestimmten Zeitpunkt. Sie wird als Prozentsatz angegeben. Bei der automatischen Skalierung wird ein Algorithmus zur Zielverfolgung verwendet, um den bereitgestellten Durchsatz der Tabelle entsprechend der tatsächlichen Arbeitslast nach oben oder unten anzupassen. Dadurch wird sichergestellt, dass die tatsächliche Kapazitätsauslastung bei oder in der Nähe Ihrer Zielauslastung bleibt.

Sie können die Zielauslastungswerte für die automatische Skalierung zwischen 20 und 90 Prozent für Ihre Lese- und Schreibkapazität festlegen. Die standardmäßige Zielauslastungsrate beträgt 70 Prozent. Sie können die Zielauslastung auf einen niedrigeren Prozentsatz festlegen, wenn sich Ihr Traffic schnell ändert und Sie möchten, dass die Kapazität früher erhöht wird. Sie können die Zielauslastungsrate auch auf einen höheren Wert festlegen, wenn sich Ihr Anwendungsdatenverkehr langsamer ändert und Sie die Durchsatzkosten senken möchten.

Weitere Informationen zu Skalierungsrichtlinien finden Sie unter Ziel-Tracking-Skalierungsrichtlinien für Application Auto Scaling im Application Auto Scaling-Benutzerhandbuch.

Wenn Sie eine Skalierungsrichtlinie erstellen, erstellt Amazon Keyspaces in Ihrem Namen zwei Paare von CloudWatch Amazon-Alarmen. Jedes Paar steht für Ihre Ober- und Untergrenzen für die Einstellungen für den bereitgestellten und verbrauchten Durchsatz. Diese CloudWatch Alarme werden ausgelöst, wenn die tatsächliche Auslastung der Tabelle über einen längeren Zeitraum von Ihrer Zielauslastung abweicht. Weitere Informationen zu Amazon CloudWatch finden Sie im CloudWatch Amazon-Benutzerhandbuch.

Wenn einer der CloudWatch Alarme ausgelöst wird, sendet Ihnen Amazon Simple Notification Service (Amazon SNS) eine Benachrichtigung (sofern Sie diese aktiviert haben). Der CloudWatch Alarm ruft dann Application Auto Scaling auf, um Ihre Skalierungsrichtlinie zu bewerten. Dadurch wird wiederum eine Alter-Table-Anfrage an Amazon Keyspaces gesendet, um die bereitgestellte Kapazität der Tabelle je nach Bedarf nach oben oder unten anzupassen. Weitere Informationen zu Amazon SNS-Benachrichtigungen finden Sie unter Amazon SNS-Benachrichtigungen einrichten.

Amazon Keyspaces verarbeitet die Alter Table-Anfrage, indem die bereitgestellte Durchsatzkapazität der Tabelle erhöht (oder verringert) wird, sodass sie sich Ihrer Zielauslastung nähert.

> **Note**
>
> Amazon Keyspaces Auto Scaling ändert die bereitgestellten Durchsatzeinstellungen nur, wenn die tatsächliche Arbeitslast über einen längeren Zeitraum von mehreren Minuten erhöht (oder niedrig) bleibt. Der Ziel-Tracking-Algorithmus von versucht, dafür zu sorgen, dass die Zielauslastung langfristig Ihrem gewählten Wert (nahezu) entspricht. Plötzliche Aktivitätsspitzen werden von der integrierten Burst-Kapazität der Tabelle bewältigt.

So funktioniert Auto Scaling für Tabellen mit mehreren Regionen

Um sicherzustellen, dass in einer Tabelle mit mehreren Regionen im Modus für bereitgestellte Kapazität immer genügend Lese- und Schreibkapazität für alle AWS-Regionen Tabellenrepliken vorhanden ist, empfehlen wir, Amazon Keyspaces Auto Scaling zu konfigurieren.

Wenn Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus mit Auto Scaling verwenden, können Sie Auto Scaling nicht für ein einzelnes Tabellenreplikat deaktivieren. Sie können jedoch die Autoscaling-Einstellungen der Tabelle für Lesevorgänge für verschiedene Regionen anpassen.

Beispielsweise können Sie für jede Region, in der die Tabelle repliziert wird, unterschiedliche Lesekapazität und Autoscaling-Einstellungen für Lesen angeben.

Die Autoscaling-Einstellungen für Lesen, die Sie für ein Tabellenreplikat in einer angegebenen Region konfigurieren, überschreiben die allgemeinen Auto Scaling-Einstellungen der Tabelle. Die Schreibkapazität muss jedoch für alle Tabellenreplikate synchronisiert bleiben, um sicherzustellen, dass genügend Kapazität vorhanden ist, um Schreibvorgänge in allen Regionen zu replizieren.

Amazon Keyspaces Auto Scaling aktualisiert die bereitgestellte Kapazität der Tabellen in jeder AWS-Region Tabelle unabhängig von der Nutzung in dieser Region. Daher kann die bereitgestellte Kapazität in jeder Region für eine Tabelle mit mehreren Regionen unterschiedlich sein, wenn Auto Scaling aktiv ist.

Sie können die Autoscaling-Einstellungen einer Tabelle mit mehreren Regionen und ihrer Replikate mithilfe der Amazon Keyspaces-Konsole, API oder CQL konfigurieren. AWS CLI Weitere Informationen zum Erstellen und Aktualisieren von Auto Scaling-Einstellungen für Tabellen mit mehreren Regionen finden Sie unter. the section called “Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling für eine Tabelle mit mehreren Regionen”

> **Note**
>
> Wenn Sie Auto Scaling für Tabellen mit mehreren Regionen verwenden, müssen Sie immer Amazon Keyspaces-API-Operationen verwenden, um die Auto Scaling-Einstellungen zu konfigurieren. Wenn Sie die Application Auto Scaling-API-Operationen direkt verwenden, um Auto Scaling-Einstellungen zu konfigurieren, haben Sie nicht die Möglichkeit, die AWS-Regionen der Tabelle mit mehreren Regionen anzugeben. Dies kann dazu führen, dass Konfigurationen nicht unterstützt werden.

Nutzungshinweise

Bevor Sie mit der automatischen Skalierung von Amazon Keyspaces beginnen, sollten Sie Folgendes beachten:

- Die automatische Skalierung von Amazon Keyspaces ist in der Region Naher Osten (VAE) nicht verfügbar.
- Die automatische Skalierung von Amazon Keyspaces kann die Lese- oder Schreibkapazität gemäß Ihrer Skalierungsrichtlinie so oft wie nötig erhöhen. Alle Amazon Keyspaces-Kontingente bleiben in Kraft, wie unter beschrieben. Kontingente

- Die automatische Skalierung von Amazon Keyspaces hindert Sie nicht daran, die bereitgestellten Durchsatzeinstellungen manuell zu ändern. Diese manuellen Anpassungen wirken sich nicht auf bestehende CloudWatch Alarme aus, die der Skalierungsrichtlinie zugeordnet sind.
- Wenn Sie die Konsole verwenden, um eine Tabelle mit bereitgestellter Durchsatzkapazität zu erstellen, ist die automatische Skalierung von Amazon Keyspaces standardmäßig aktiviert. Sie können Ihre Einstellungen für die automatische Skalierung jederzeit ändern. Weitere Informationen finden Sie unter the section called “auto Skalierung von Amazon Keyspaces für eine Tabelle ausschalten”.
- Wenn Sie CloudFormation zum Erstellen von Skalierungsrichtlinien verwenden, sollten Sie die Skalierungsrichtlinien CloudFormation so verwalten, dass der Stack mit der Stack-Vorlage synchronisiert ist. Wenn Sie die Skalierungsrichtlinien von Amazon Keyspaces ändern, werden diese beim Zurücksetzen des Stacks mit den Originalwerten aus der CloudFormation Stack-Vorlage überschrieben.
- Wenn Sie die automatische Skalierung von Amazon Keyspaces CloudTrail zur Überwachung verwenden, erhalten Sie möglicherweise Warnmeldungen für Aufrufe, die von Application Auto Scaling im Rahmen des Konfigurationsvalidierungsprozesses getätigt werden. Sie können diese Warnungen herausfiltern, indem Sie das `invokedBy` Feld verwenden, das die Informationen `application-autoscaling.amazonaws.com` für diese Validierungsprüfungen enthält.

Automatische Skalierungsrichtlinien von Amazon Keyspace konfigurieren und aktualisieren

Sie können die Konsole, CQL oder die AWS Command Line Interface (AWS CLI) verwenden, um die automatische Skalierung von Amazon Keyspaces für neue und bestehende Tabellen zu konfigurieren. Sie können auch die Einstellungen für die automatische Skalierung ändern oder die automatische Skalierung deaktivieren.

Für erweiterte Funktionen wie das Einstellen von Scale-In- und Scale-Out-Abklingzeiten empfehlen wir, dass Sie CQL oder die verwenden, um die Skalierungsrichtlinien von Amazon Keyspaces AWS CLI zu verwalten.

Themen

- Berechtigungen für die automatische Skalierung von Amazon Keyspaces konfigurieren
- Erstellen Sie eine neue Tabelle mit automatischer Skalierung
- Konfigurieren Sie die automatische Skalierung für eine bestehende Tabelle

- Sehen Sie sich die Amazon Keyspaces-Auto-Scaling-Konfiguration Ihrer Tabelle an
- auto Skalierung von Amazon Keyspaces für eine Tabelle ausschalten
- Auto-Scaling-Aktivitäten für eine Amazon Keyspaces-Tabelle in Amazon anzeigen CloudWatch

Berechtigungen für die automatische Skalierung von Amazon Keyspaces konfigurieren

Stellen Sie zunächst sicher, dass der Principal über die entsprechenden Berechtigungen zum Erstellen und Verwalten von Einstellungen für die automatische Skalierung verfügt. In AWS Identity and Access Management (IAM) `AmazonKeyspacesFullAccess` ist die AWS verwaltete Richtlinie erforderlich, um die Skalierungsrichtlinien von Amazon Keyspaces zu verwalten.

> **Important**
>
> `application-autoscaling:*`Berechtigungen sind erforderlich, um die automatische Skalierung für eine Tabelle zu deaktivieren. Sie müssen Auto Scaling für eine Tabelle deaktivieren, bevor Sie sie löschen können.

Um einen IAM-Benutzer oder eine IAM-Rolle für den Zugriff auf die Amazon Keyspaces-Konsole und die automatische Skalierung von Amazon Keyspaces einzurichten, fügen Sie die folgende Richtlinie hinzu.

Um die Richtlinie anzuhängen **`AmazonKeyspacesFullAccess`**

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Dashboard der IAM-Konsole Benutzer und dann Ihren IAM-Benutzer oder Ihre IAM-Rolle aus der Liste aus.
3. Wählen Sie auf der Seite Summary (Übersicht) die Option Add permissions (Berechtigungen hinzufügen) aus.
4. Wählen Sie Attach existing policies directly (Vorhandene Richtlinien direkt zuordnen).
5. Wählen Sie AmazonKeyspacesFullAccessin der Liste der Richtlinien die Option und anschließend Weiter: Überprüfen aus.
6. Wählen Sie Add permissions (Berechtigungen hinzufügen) aus.

Erstellen Sie eine neue Tabelle mit automatischer Skalierung

Wenn Sie eine neue Amazon Keyspaces-Tabelle erstellen, können Sie automatisch Auto Scaling für die Schreib- oder Lesekapazität der Tabelle aktivieren. Dadurch kann Amazon Keyspaces in Ihrem Namen Application Auto Scaling kontaktieren, um die Tabelle als skalierbares Ziel zu registrieren und die bereitgestellte Schreib- oder Lesekapazität anzupassen.

Weitere Informationen zum Erstellen einer Tabelle mit mehreren Regionen und zum Konfigurieren verschiedener Auto Scaling-Einstellungen für Tabellenreplikate finden Sie unter. the section called “Erstellen Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus”

> **Note**
>
> Die automatische Skalierung von Amazon Keyspace erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`), die automatische Skalierungsaktionen in Ihrem Namen ausführt. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “Verwenden von servicegebundenen Rollen”.

Console

Erstellen Sie mithilfe der Konsole eine neue Tabelle mit aktivierter automatischer Skalierung

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) und anschließend Create table (Tabelle erstellen) aus.
3. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die neue Tabelle ein.
4. Erstellen Sie im Abschnitt Spalten das Schema für Ihre Tabelle.
5. Definieren Sie im Abschnitt Primärschlüssel den Primärschlüssel der Tabelle und wählen Sie optionale Clusterspalten aus.
6. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.
7. Fahren Sie mit den Read/write Kapazitätseinstellungen fort.
8. Wählen Sie für den Kapazitätsmodus die Option Bereitgestellt aus.

9. Vergewissern Sie sich, dass im Abschnitt Lesekapazität die Option Automatisch skalieren ausgewählt ist.

   In diesem Schritt wählen Sie die minimalen und maximalen Lesekapazitätseinheiten für die Tabelle sowie die Zielauslastung aus.

   - Mindestkapazitätseinheiten — Geben Sie den Wert für den Mindestdurchsatz ein, den die Tabelle immer unterstützen soll. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
   - Einheiten mit maximaler Kapazität — Geben Sie den maximalen Durchsatz ein, den Sie für die Tabelle bereitstellen möchten. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
   - Zielauslastung — Geben Sie eine Zielauslastungsrate zwischen 20 und 90% ein. Wenn der Verkehr die definierte Zielauslastungsrate überschreitet, wird die Kapazität automatisch hochskaliert. Wenn der Verkehr unter das definierte Ziel fällt, wird er automatisch wieder herunterskaliert.

   > **Note**
   >
   > Weitere Informationen zu Standardkontingenten für Ihr Konto und deren Erhöhung finden Sie unterKontingente.

10. Wählen Sie im Abschnitt Schreibkapazität dieselben Einstellungen wie im vorherigen Schritt für die Lesekapazität aus, oder konfigurieren Sie die Kapazitätswerte manuell.
11. Wählen Sie Create table (Tabelle erstellen) aus. Ihre Tabelle wird mit den angegebenen Parametern für die automatische Skalierung erstellt.

Cassandra Query Language (CQL)

Erstellen Sie eine neue Tabelle mit der automatischen Skalierung von Amazon Keyspaces mithilfe von CQL

Um Auto Scaling-Einstellungen für eine Tabelle programmgesteuert zu konfigurieren, verwenden Sie die `AUTOSCALING_SETTINGS` Anweisung, die die Parameter für Amazon Keyspaces Auto Scaling enthält. Die Parameter definieren die Bedingungen, unter denen Amazon Keyspaces angewiesen wird, den bereitgestellten Durchsatz Ihrer Tabelle anzupassen, und welche

zusätzlichen optionalen Aktionen zu ergreifen sind. In diesem Beispiel definieren Sie die Auto Scaling-Einstellungen für mytable.

Die Richtlinie enthält die folgenden Elemente:

- `AUTOSCALING_SETTINGS`— Gibt an, ob Amazon Keyspaces die Durchsatzkapazität in Ihrem Namen anpassen darf. Die folgenden Werte sind erforderlich:
  - `provisioned_write_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `provisioned_read_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `scaling_policy`— Amazon Keyspaces unterstützt die Richtlinie zur Zielverfolgung. Um die Ziel-Tracking-Richtlinie zu definieren, konfigurieren Sie die folgenden Parameter.
    - `target_value`— Die auto Skalierung von Amazon Keyspaces stellt sicher, dass das Verhältnis von verbrauchter Kapazität zu bereitgestellter Kapazität auf oder nahe diesem Wert bleibt. Sie definieren `target_value` als Prozentsatz.
    - `disableScaleIn`: (Optional) Ein Wert, der `boolean` angibt, ob für die Tabelle deaktiviert oder aktiviert `scale-in` ist. Dieser Parameter ist standardmäßig deaktiviert. Stellen Sie zum `scale-in` Einschalten den `boolean` Wert auf ein`FALSE`. Das bedeutet, dass die Kapazität für eine Tabelle in Ihrem Namen automatisch herunterskaliert wird.
    - `scale_out_cooldown`— Eine Scale-Out-Aktivität erhöht den bereitgestellten Durchsatz Ihrer Tabelle. Um eine Abklingzeit für Scale-Out-Aktivitäten hinzuzufügen, geben Sie einen Wert in Sekunden für an. `scale_out_cooldown` Wenn Sie keinen Wert angeben, ist der Standardwert 0. Weitere Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.
    - `scale_in_cooldown`— Eine Scale-In-Aktivität verringert den bereitgestellten Durchsatz Ihrer Tabelle. Um eine Abklingzeit für Scale-In-Aktivitäten hinzuzufügen, geben Sie einen Wert in Sekunden für an. `scale_in_cooldown` Wenn Sie keinen Wert angeben, ist der Standardwert 0. Weitere Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.

> **Note**
>
> Um besser zu verstehen, wie der `target_value` funktioniert, nehmen Sie an, dass Sie über eine Tabelle mit einer Einstellung des bereitgestellten Durchsatzes von 200 Schreibkapazitätseinheiten verfügen. Sie entscheiden sich dafür, eine Skalierungsrichtlinie für diese Tabelle mit einem `target_value` von 70 % zu erstellen.
> Angenommen, Sie beginnen den Schreibverkehr zu der Tabelle zu leiten, damit der tatsächliche Schreibdurchsatz bei 150 Kapazitätseinheiten liegt. Das Verhältnis von verbraucht zu bereitgestellt ist jetzt (150/200) oder 75 %. Dieses Verhältnis übersteigt Ihr Ziel, sodass Auto Scaling die bereitgestellte Schreibkapazität auf 215 erhöht, sodass das Verhältnis (150/ 215) oder 69,77 Prozent so nahe `target_value` wie möglich an Ihrem liegt, aber nicht überschritten wird.

Für mytable haben Sie sowohl die Lese- als auch `TargetValue` die Schreibkapazität auf 50 Prozent festgelegt. Amazon Keyspaces Auto Scaling passt den bereitgestellten Durchsatz der Tabelle im Bereich von 5 bis 10 Kapazitätseinheiten an, sodass das Verhältnis von verbrauchten zu bereitgestellten Kapazitäten bei oder nahe 50 Prozent bleibt. Für die Lesekapazität legen Sie die Werte für und auf 60 Sekunden fest. `ScaleOutCooldown` `ScaleInCooldown`

Sie können die folgende Anweisung verwenden, um eine neue Amazon Keyspaces-Tabelle mit aktivierter auto Skalierung zu erstellen.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
    },
```

```
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    }
};
```

CLI

Erstellen Sie eine neue Tabelle mit der automatischen Skalierung von Amazon Keyspaces mithilfe der AWS CLI

Um Auto Scaling-Einstellungen für eine Tabelle programmgesteuert zu konfigurieren, verwenden Sie die `autoScalingSpecification` Aktion, die die Parameter für Amazon Keyspaces Auto Scaling definiert. Die Parameter definieren die Bedingungen, unter denen Amazon Keyspaces angewiesen wird, den bereitgestellten Durchsatz Ihrer Tabelle anzupassen, und welche zusätzlichen optionalen Aktionen zu ergreifen sind. In diesem Beispiel definieren Sie die Auto Scaling-Einstellungen für mytable.

Die Richtlinie enthält die folgenden Elemente:

- `autoScalingSpecification`— Gibt an, ob Amazon Keyspaces den Kapazitätsdurchsatz in Ihrem Namen anpassen darf. Sie können Auto Scaling für Lese- und Schreibkapazität separat aktivieren. Dann müssen Sie die folgenden Parameter angeben für`autoScalingSpecification`:
  - `writeCapacityAutoScaling`— Die Einheiten mit der maximalen und minimalen Schreibkapazität.
  - `readCapacityAutoScaling`— Die maximalen und minimalen Lesekapazitätseinheiten.
  - `scalingPolicy`— Amazon Keyspaces unterstützt die Richtlinie zur Zielverfolgung. Um die Ziel-Tracking-Richtlinie zu definieren, konfigurieren Sie die folgenden Parameter.
    - `targetValue`— Die auto Skalierung von Amazon Keyspaces stellt sicher, dass das Verhältnis von verbrauchter Kapazität zu bereitgestellter Kapazität auf oder nahe diesem Wert bleibt. Sie definieren `targetValue` als Prozentsatz.

- `disableScaleIn`: (Optional) Ein Wert, der `boolean` angibt, ob für die Tabelle deaktiviert oder aktiviert `scale-in` ist. Dieser Parameter ist standardmäßig deaktiviert. Stellen Sie zum `scale-in` Einschalten den `boolean` Wert auf ein`FALSE`. Das bedeutet, dass die Kapazität für eine Tabelle in Ihrem Namen automatisch herunterskaliert wird.
- `scaleOutCooldown`— Eine Scale-Out-Aktivität erhöht den bereitgestellten Durchsatz Ihrer Tabelle. Um eine Abklingzeit für Scale-Out-Aktivitäten hinzuzufügen, geben Sie einen Wert in Sekunden für an. `ScaleOutCooldown` Der Standardwert lautet 0. Weitere Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.
- `scaleInCooldown`— Eine Scale-In-Aktivität verringert den bereitgestellten Durchsatz Ihrer Tabelle. Um eine Abklingzeit für Scale-In-Aktivitäten hinzuzufügen, geben Sie einen Wert in Sekunden für an. `ScaleInCooldown` Der Standardwert lautet 0. Weitere Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.

> **Note**
>
> Um besser zu verstehen, wie der `TargetValue` funktioniert, nehmen Sie an, dass Sie über eine Tabelle mit einer Einstellung des bereitgestellten Durchsatzes von 200 Schreibkapazitätseinheiten verfügen. Sie entscheiden sich dafür, eine Skalierungsrichtlinie für diese Tabelle mit einem `TargetValue` von 70 % zu erstellen.
> Angenommen, Sie beginnen den Schreibverkehr zu der Tabelle zu leiten, damit der tatsächliche Schreibdurchsatz bei 150 Kapazitätseinheiten liegt. Das Verhältnis von verbraucht zu bereitgestellt ist jetzt (150/200) oder 75 %. Dieses Verhältnis übersteigt Ihr Ziel, sodass Auto Scaling die bereitgestellte Schreibkapazität auf 215 erhöht, sodass das Verhältnis (150/ 215) oder 69,77 Prozent so nahe `TargetValue` wie möglich an Ihrem liegt, aber nicht überschritten wird.

Für mytable haben Sie sowohl die Lese- als auch `TargetValue` die Schreibkapazität auf 50 Prozent festgelegt. Amazon Keyspaces Auto Scaling passt den bereitgestellten Durchsatz der Tabelle im Bereich von 5 bis 10 Kapazitätseinheiten an, sodass das Verhältnis von verbrauchten zu bereitgestellten Kapazitäten bei oder nahe 50 Prozent bleibt. Für die Lesekapazität legen Sie die Werte für und auf 60 Sekunden fest. `ScaleOutCooldown` `ScaleInCooldown`

Beim Erstellen von Tabellen mit komplexen Auto-Scaling-Einstellungen ist es hilfreich, die Auto-Scaling-Einstellungen aus einer JSON-Datei zu laden. Für das folgende Beispiel können Sie die JSON-Beispieldatei von auto-scaling.zip herunterladen und extrahieren`auto-scaling.json`, wobei Sie sich den Pfad zur Datei notieren. In diesem Beispiel befindet sich die JSON-Datei im aktuellen Verzeichnis. Informationen zu verschiedenen Dateipfadoptionen finden Sie unter So laden Sie Parameter aus einer Datei.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable
            \ --schema-definition 'allColumns=[{name=pk,type=int},
{name=ck,type=int}],partitionKeys=[{name=pk},{name=ck}]'
            \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
            \ --auto-scaling-specification file://auto-scaling.json
```

Konfigurieren Sie die automatische Skalierung für eine bestehende Tabelle

Sie können eine bestehende Amazon Keyspaces-Tabelle aktualisieren, um die auto Skalierung für die Schreib- oder Lesekapazität der Tabelle zu aktivieren. Wenn Sie eine Tabelle aktualisieren, die sich derzeit im On-Demand-Kapazitätsmodus befindet, müssen Sie zunächst den Kapazitätsmodus der Tabelle in den Modus Bereitgestellte Kapazität ändern.

Weitere Informationen zum Aktualisieren der Auto Scaling-Einstellungen für eine Tabelle mit mehreren Regionen finden Sie unterthe section called “Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling für eine Tabelle mit mehreren Regionen”.

Die automatische Skalierung von Amazon Keyspace erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`), die automatische Skalierungsaktionen in Ihrem Namen ausführt. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “Verwenden von servicegebundenen Rollen”.

Console

Automatische Skalierung von Amazon Keyspaces für eine bestehende Tabelle konfigurieren

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie die Tabelle aus, mit der Sie arbeiten möchten, und wechseln Sie zur Registerkarte Kapazität.

3. Wählen Sie im Abschnitt Kapazitätseinstellungen die Option Bearbeiten aus.
4. Stellen Sie unter Kapazitätsmodus sicher, dass für die Tabelle der Modus Bereitgestellte Kapazität verwendet wird.
5. Wählen Sie Automatisch skalieren aus und lesen Sie Schritt 6 unterthe section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”, um die Lese- und Schreibkapazität zu bearbeiten.
6. Wenn die Einstellungen für die automatische Skalierung definiert sind, wählen Sie Speichern.

Cassandra Query Language (CQL)

Konfiguration einer vorhandenen Tabelle mit der automatischen Skalierung von Amazon Keyspaces mithilfe von CQL

Sie können die `ALTER TABLE` Anweisung für eine bestehende Amazon Keyspaces-Tabelle verwenden, um Auto Scaling für die Schreib- oder Lesekapazität der Tabelle zu konfigurieren. Wenn Sie eine Tabelle aktualisieren, die sich derzeit im On-Demand-Kapazitätsmodus befindet, müssen Sie sie `capacity_mode` auf bereitgestellt setzen. Wenn sich Ihre Tabelle bereits im Modus für bereitgestellte Kapazität befindet, kann dieses Feld weggelassen werden.

Im folgenden Beispiel aktualisiert die Anweisung die Tabelle mytable, die sich im On-Demand-Kapazitätsmodus befindet. Die Anweisung ändert den Kapazitätsmodus der Tabelle in den Bereitstellungsmodus mit aktivierter auto Skalierung.

Die Schreibkapazität wird im Bereich von 5—10 Kapazitätseinheiten mit einem Zielwert von 50% konfiguriert. Die Lesekapazität wird ebenfalls im Bereich von 5—10 Kapazitätseinheiten mit einem Zielwert von 50% konfiguriert. Für die Lesekapazität legen Sie die Werte für `scale_out_cooldown` und `scale_in_cooldown` auf 60 Sekunden fest.

```
ALTER TABLE mykeyspace.mytable
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
```

```
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    }
};
```

CLI

Konfigurieren Sie eine bestehende Tabelle mit der automatischen Skalierung von Amazon Keyspaces mithilfe der AWS CLI

Für eine bestehende Amazon Keyspaces-Tabelle können Sie mithilfe des `UpdateTable` Vorgangs die auto Skalierung für die Schreib- oder Lesekapazität der Tabelle aktivieren.

Sie können den folgenden Befehl verwenden, um Amazon Keyspaces Auto Scaling für eine bestehende Tabelle zu aktivieren. Die Auto-Scaling-Einstellungen für die Tabelle werden aus einer JSON-Datei geladen. Für das folgende Beispiel können Sie die JSON-Beispieldatei von auto-scaling.zip herunterladen und extrahieren`auto-scaling.json`, wobei Sie sich den Pfad zur Datei notieren. In diesem Beispiel befindet sich die JSON-Datei im aktuellen Verzeichnis. Informationen zu verschiedenen Dateipfadoptionen finden Sie unter So laden Sie Parameter aus einer Datei.

Weitere Informationen zu den im folgenden Beispiel verwendeten Auto Scaling-Einstellungen finden Sie unterthe section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
            \ --capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1
            \ --auto-scaling-specification file://auto-scaling.json
```

Sehen Sie sich die Amazon Keyspaces-Auto-Scaling-Konfiguration Ihrer Tabelle an

Sie können die Konsole, CQL oder die verwenden, AWS CLI um die automatischen Skalierungseinstellungen von Amazon Keyspaces einer Tabelle anzuzeigen und zu aktualisieren.

Console

Die Einstellungen für die automatische Skalierung mithilfe der Konsole anzeigen

1. Wählen Sie die Tabelle aus, die Sie anzeigen möchten, und wechseln Sie zur Registerkarte Kapazität.
2. Wählen Sie im Abschnitt Kapazitätseinstellungen die Option Bearbeiten aus. Sie können jetzt die Einstellungen in den Abschnitten Lesekapazität oder Schreibkapazität ändern. Weitere Informationen zu diesen Einstellungen finden Sie unter the section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”.

Cassandra Query Language (CQL)

Die automatische Skalierungsrichtlinie von Amazon Keyspaces für Ihre Tabelle mithilfe von CQL anzeigen

Verwenden Sie den folgenden Befehl, um Details der Auto Scaling-Konfiguration einer Tabelle anzuzeigen.

```
SELECT * FROM system_schema_mcs.autoscaling WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

Die Ausgabe für diesen Befehl sieht wie folgt aus.

```
 keyspace_name | table_name | provisioned_read_capacity_autoscaling_update

                                                                                |
 provisioned_write_capacity_autoscaling_update
---------------+------------
+-------------------------------------------------------------------------------------------
+-------------------------------------------------------------------------------------------
 mykeyspace    | mytable    | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 60, 'disable_scale_in': false, 'target_value':
```

```
 50, 'scale_in_cooldown': 60}}} | {'minimum_units': 5, 'maximum_units':
 10, 'scaling_policy': {'target_tracking_scaling_policy_configuration':
 {'scale_out_cooldown': 0, 'disable_scale_in': false, 'target_value': 50,
 'scale_in_cooldown': 0}}}
```

CLI

Sehen Sie sich die automatische Skalierungsrichtlinie von Amazon Keyspaces für Ihre Tabelle an, indem Sie AWS CLI

Um die Auto Scaling-Konfiguration einer Tabelle anzuzeigen, können Sie den `get-table-auto-scaling-settings` Vorgang verwenden. Der folgende CLI-Befehl ist ein Beispiel dafür.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-
name mytable
```

Die Ausgabe für diesen Befehl sieht so aus.

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable",
    "autoScalingSpecification": {
        "writeCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 10,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 0,
                    "scaleOutCooldown": 0,
                    "targetValue": 50.0
                }
            }
        },
        "readCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 10,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
```

```
                    "disableScaleIn": false,
                    "scaleInCooldown": 60,
                    "scaleOutCooldown": 60,
                    "targetValue": 50.0
                }
            }
        }
    }
}
```

auto Skalierung von Amazon Keyspaces für eine Tabelle ausschalten

Sie können die auto Skalierung von Amazon Keyspaces für Ihre Tabelle jederzeit deaktivieren. Wenn Sie die Lese- oder Schreibkapazität Ihrer Tabelle nicht mehr skalieren müssen, sollten Sie erwägen, Auto Scaling zu deaktivieren, damit Amazon Keyspaces die Lese- oder Schreibkapazitätseinstellungen Ihrer Tabelle nicht weiter ändert. Sie können die Tabelle mit der Konsole, CQL oder dem aktualisieren. AWS CLI

Wenn Sie Auto Scaling ausschalten, werden auch die CloudWatch Alarme gelöscht, die in Ihrem Namen erstellt wurden.

Um die serviceverknüpfte Rolle zu löschen, die von Application Auto Scaling für den Zugriff auf Ihre Amazon Keyspaces-Tabelle verwendet wird, folgen Sie den Schritten unter. the section called “Löschen einer serviceverknüpften Rolle für Amazon Keyspaces”

> **Note**
>
> Um die dienstverknüpfte Rolle zu löschen, die Application Auto Scaling verwendet, müssen Sie die automatische Skalierung für alle Tabellen im Konto deaktivieren. AWS-Regionen

Console

Schalten Sie die automatische Skalierung von Amazon Keyspace für Ihre Tabelle mithilfe der Konsole aus

Verwenden der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.

2. Wählen Sie die Tabelle aus, die Sie aktualisieren möchten, und wechseln Sie zur Registerkarte Kapazität.
3. Wählen Sie im Abschnitt Kapazitätseinstellungen die Option Bearbeiten aus.
4. Um die automatische Skalierung von Amazon Keyspaces zu deaktivieren, deaktivieren Sie das Kontrollkästchen Automatisch skalieren. Wenn Sie die automatische Skalierung deaktivieren, wird die Tabelle mit Application Auto Scaling als skalierbares Ziel abgemeldet.

Cassandra Query Language (CQL)

Deaktivieren Sie die automatische Skalierung von Amazon Keyspace für Ihre Tabelle mithilfe von CQL

Die folgende Anweisung deaktiviert die auto Skalierung für die Schreibkapazität der Tabelle mytable.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

CLI

Deaktivieren Sie die automatische Skalierung von Amazon Keyspace für Ihre Tabelle mithilfe der AWS CLI

Der folgende Befehl deaktiviert die auto Skalierung für die Lesekapazität der Tabelle. Außerdem werden die CloudWatch Alarme gelöscht, die in Ihrem Namen erstellt wurden.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
            \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

Auto-Scaling-Aktivitäten für eine Amazon Keyspaces-Tabelle in Amazon anzeigen CloudWatch

Sie können mithilfe von Amazon überwachen, wie die automatische Skalierung von Amazon Keyspaces Ressourcen verwendet. Amazon CloudWatch generiert Messwerte zu Ihrer Nutzung

und Leistung. Folgen Sie den Schritten im Application Auto Scaling Benutzerhandbuch, um ein CloudWatch Dashboard zu erstellen.

Effektive Nutzung von Burst-Kapazitäten in Amazon Keyspaces

Amazon Keyspaces bietet eine gewisse Flexibilität bei der Durchsatzbereitstellung pro Partition, indem es Burst-Kapazität bereitstellt. Immer wenn Sie den Durchsatz einer Partition nicht vollständig nutzen, reserviert Amazon Keyspaces einen Teil dieser ungenutzten Kapazität für spätere Durchsatzspitzen, um Nutzungsspitzen zu bewältigen.

Amazon Keyspaces behält derzeit bis zu 5 Minuten (300 Sekunden) ungenutzte Lese- und Schreibkapazität. Bei gelegentlichen Lese- oder Schreibaktivitäten können diese zusätzlichen Kapazitätseinheiten schnell verbraucht werden — sogar schneller als die pro Sekunde bereitgestellte Durchsatzkapazität, die Sie für Ihre Tabelle definiert haben.

Amazon Keyspaces kann auch ohne vorherige Ankündigung Burst-Kapazität für die Hintergrundwartung und andere Aufgaben verbrauchen.

Beachten Sie, dass die Burst-Kapazität in Zukunft geändert werden kann.

Arbeiten mit Funktionen von Amazon Keyspaces (für Apache Cassandra)

Dieses Kapitel enthält Einzelheiten zur Arbeit mit Amazon Keyspaces und verschiedenen Datenbankfunktionen, z. B. Sicherung und Wiederherstellung, Time to Live und Replikation in mehreren Regionen.

- Time to Live — Amazon Keyspaces lässt Daten aus Tabellen automatisch auf der Grundlage des von Ihnen festgelegten Time-to-Live-Werts ablaufen. Erfahren Sie, wie Sie TTL konfigurieren und wie Sie es in Ihren Tabellen verwenden können.
- PITR — Schützen Sie Ihre Amazon Keyspaces-Tabellen vor versehentlichen Schreib- oder Löschvorgängen, indem Sie kontinuierliche Backups Ihrer Tabellendaten erstellen. Erfahren Sie, wie Sie PITR für Ihre Tabellen konfigurieren und wie Sie eine Tabelle zu einem bestimmten Zeitpunkt wiederherstellen oder wie Sie eine Tabelle wiederherstellen, die versehentlich gelöscht wurde.
- Arbeiten mit Tabellen mit mehreren Regionen — Bei Multi-Region Tabellen in Amazon Keyspaces muss die Schreibdurchsatzkapazität entweder im On-Demand-Modus oder im Bereitstellungsmodus mit auto Skalierung konfiguriert sein. Planen Sie den Durchsatzkapazitätsbedarf, indem Sie die erforderlichen Schreibkapazitätseinheiten (WCUs) für jede Region schätzen und die Summe der Schreibvorgänge aus allen Regionen bereitstellen, um eine ausreichende Kapazität für replizierte Schreibvorgänge sicherzustellen.
- Amazon Keyspaces Change Data Capture (CDC) — Amazon Keyspaces CDC-Streams zeichnen Änderungsereignisse auf Zeilenebene aus Ihrer Tabelle nahezu in Echtzeit auf. Erfahren Sie, wie Sie die Kinesis Client Library (KCL) verwenden, um Daten aus Amazon Keyspaces CDC-Streams zu nutzen und zu verarbeiten.
- Abfragen und Paginierung — Amazon Keyspaces unterstützt erweiterte Abfragefunktionen wie die Verwendung des `IN` Operators mit `SELECT` Anweisungen, das Sortieren von Ergebnissen mit `ORDER BY` und die automatische Paginierung großer Ergebnismengen. In diesem Abschnitt wird erklärt, wie Amazon Keyspaces diese Abfragen verarbeitet, und es werden Beispiele bereitgestellt.
- Partitionierer — Amazon Keyspaces bietet drei Partitionierer: `Murmur3Partitioner` (Standard), und. `RandomPartitioner` `DefaultPartitioner` Sie können die Partitionierung pro Region auf Kontoebene mithilfe der AWS-Managementkonsole oder Cassandra Query Language (CQL) ändern.

- Client-side Zeitstempel — Client-side Zeitstempel sind Cassandra-compatible Zeitstempel, die Amazon Keyspaces für jede Zelle in Ihrer Tabelle beibehält. Verwenden Sie clientseitige Zeitstempel zur Konfliktlösung und damit Ihre Client-Anwendung die Reihenfolge der Schreibvorgänge bestimmen kann.
- User-defined Typen (UDTs) — Mit UDTs können Sie Datenstrukturen in Ihren Anwendungen definieren, die reale Datenhierarchien repräsentieren.
- Ressourcen kennzeichnen — Sie können Amazon Keyspaces-Ressourcen wie Keyspaces und Tabellen mithilfe von Tags kennzeichnen. Tags helfen dabei, Ressourcen zu kategorisieren, ermöglichen die Kostenverfolgung und ermöglichen es Ihnen, die Zugriffskontrolle auf der Grundlage von Stichwörtern zu konfigurieren. In diesem Abschnitt werden Einschränkungen, Abläufe und bewährte Methoden für das Tagging für Amazon Keyspaces behandelt.
- CloudFormation Vorlagen — CloudFormation hilft Ihnen dabei, Ihre Amazon Keyspaces und -Tabellen zu modellieren und einzurichten, sodass Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen müssen.

Themen

- Systemschlüsselräume in Amazon Keyspaces
- User-defined Typen (UDTs) in Amazon Keyspaces
- Arbeiten mit CQL-Abfragen in Amazon Keyspaces
- Arbeiten mit Change Data Capture (CDC) -Streams in Amazon Keyspaces
- Arbeiten mit Partitionierern in Amazon Keyspaces
- Client-side Zeitstempel in Amazon Keyspaces
- Multi-Region Replikation für Amazon Keyspaces (für Apache Cassandra)
- Daten mit Point-in-Time-Recovery für Amazon Keyspaces Backup und wiederherstellen
- Daten mit Time to Live (TTL) für Amazon Keyspaces (für Apache Cassandra) ablaufen lassen
- Verwenden Sie diesen Service mit einem AWS SDK
- Arbeiten mit Tags und Labels für Amazon Keyspaces-Ressourcen
- Erstellen Sie Amazon Keyspaces-Ressourcen mit AWS CloudFormation
- Verwenden von NoSQL Workbench mit Amazon Keyspaces (für Apache Cassandra)

Systemschlüsselräume in Amazon Keyspaces

Systemschlüsselräume und Systemtabellen in Amazon Keyspaces (für Apache Cassandra) sind schreibgeschützte Ressourcen, die Metadaten zu Ihren Amazon Keyspaces-Ressourcen speichern. Systemschlüsselräume sind in allen vorhanden AWS-Konto, unabhängig davon, ob Sie irgendwelche Schlüsselräume oder Tabellen erstellt haben. Sie sind ein Kompatibilitätsmerkmal mit Apache Cassandra und werden ohne zusätzliche Kosten zur Verfügung gestellt.

Sie können Systemschlüsselräume nicht ändern oder löschen. In der Amazon Keyspaces-Konsole werden nur von Benutzern erstellte Schlüsselbereiche angezeigt. Systemschlüsselräume sind programmgesteuert über CQL zugänglich und erscheinen in Diensten wie und. AWS CloudFormation AWS Config

Amazon Keyspaces verwendet vier Systemschlüsselräume:

- `system`
- `system_schema`
- `system_schema_mcs`
- `system_multiregion_info`

Die folgenden Abschnitte enthalten Einzelheiten zu den Systemschlüsselräumen und den Systemtabellen, die in Amazon Keyspaces unterstützt werden.

`system`

Dies ist ein Cassandra-Schlüsselraum. Amazon Keyspaces verwendet die folgenden Tabellen.

Tabellennamen	Spaltennamen	Kommentare
`local`	`key, bootstrapped, broadcast_address, cluster_name, cql_version, data_center, gossip_generation, host_id, listen_address,`	Informationen über den lokalen Schlüsselraum.

Tabellennamen	Spaltennamen	Kommentare
	`native_protocol_version, partitioner, rack, release_version, rpc_address, schema_version, thrift_version, tokens, truncated_at`	
`peers`	`peer, data_center, host_id, preferred_ip, rack, release_version, rpc_address, schema_version, tokens`	Fragen Sie diese Tabelle ab, um die verfügbaren Endpunkte zu sehen. Wenn Sie beispielsweise eine Verbindung über einen öffentlichen Endpunkt herstellen, wird eine Liste mit neun verfügbaren IP-Adressen angezeigt. Wenn Sie eine Verbindung über einen FIPS-Endpunkt herstellen, wird eine Liste mit drei IP-Adressen angezeigt. Wenn Sie eine Verbindung über einen AWS PrivateLink VPC-Endpunkt herstellen, wird die Liste der IP-Adressen angezeigt, die Sie konfiguriert haben. Weitere Informationen finden Sie unter the section called “Auffüllen von `system.peers` Tabelleneinträgen mit VPC-Endpunktinformationen der Schnittstelle”.

Tabellennamen	Spaltennamen	Kommentare
`size_estimates`	`keyspace_name, table_name, range_start, range_end, mean_partition_size, partitions_count`	Diese Tabelle definiert die Gesamtgröße und Anzahl der Partitionen für jeden Token-Bereich für jede Tabelle. Dies wird für den Apache Cassandra Spark Connector benötigt, der die geschätzte Partitionsgröße verwendet, um die Arbeit zu verteilen.
`prepared_statements`	`prepared_id, logged_keyspace, query_string`	Diese Tabelle enthält Informationen zu gespeicherten Abfragen.

`system_schema`

Dies ist ein Cassandra-Keyspace. Amazon Keyspaces verwendet die folgenden Tabellen.

Tabellennamen	Spaltennamen	Kommentare
`keyspaces`	`keyspace_name, durable_writes, replication`	Informationen zu einem bestimmten Schlüsselraum.
`tables`	`keyspace_name, table_name, bloom_filter_fp_chance, caching, comment, compaction, compression, crc_check_chance, dclocal_read_repair_chance, default_time_to_live, extensions, flags, gc_grace_`	Informationen zu einer bestimmten Tabelle.

Tabellennamen	Spaltennamen	Kommentare
	`seconds, id, max_index_interval , memtable_flush_per iod_in_ms, min_index _interval, read_repa ir_chance, speculati ve_retry`	
`types`	`keyspace_name, type_name, field_nam es, field_types`	Informationen zu einem bestimmten benutzerdefinierte n Typ (UDT).
`columns`	`keyspace_name, table_name, column_na me, clusterin g_order, column_na me_bytes, kind, position, type`	Informationen zu einer bestimmten Spalte.

`system_schema_mcs`

Dies ist ein Amazon Keyspaces-Schlüsselraum, der Informationen über AWS oder Amazon Keyspaces-spezifische Einstellungen speichert.

Tabellennamen	Spaltennamen	Kommentare
`keyspaces`	`keyspace_name, durable_writes, replication`	Fragen Sie diese Tabelle ab, um programmgesteuert herauszufinden, ob ein Schlüsselraum erstellt wurde. Weitere Informationen finden Sie unter the section called “Überprüfen Sie den Status der Keyspace-Erstellung”.

Tabellennamen	Spaltennamen	Kommentare
`tables`	`keyspace_name, creation_time, speculative_retry, cdc, gc_grace_seconds, crc_check_chance, min_index_interval, bloom_filter_fp_chance, flags, custom_properties, dclocal_read_repair_chance, table_name, caching, default_time_to_live, read_repair_chance, max_index_interval, extensions, compaction, comment, id, compression, memtable_flush_period_in_ms, cdc_specification, latest_stream_arn status`	Fragen Sie diese Tabelle ab, um den Status einer bestimmten Tabelle herauszufinden. Weitere Informationen finden Sie unter the section called “Überprüfen Sie den Status der Tabellenerstellung”. Sie können diese Tabelle auch abfragen, um Einstellungen aufzulisten, die spezifisch für Amazon Keyspaces sind und unter gespeichert sind. `custom_properties` Beispiel: • `capacity_mode` • `cdc_specification` • `client_side_timestamps` • `encryption_specification` • `point_in_time_recovery` • `ttl`
`tables_history`	`keyspace_name, table_name, event_time, creation_time, custom_properties, event`	Fragen Sie diese Tabelle ab, um mehr über Schemaänderungen für eine bestimmte Tabelle zu erfahren.

Tabellennamen	Spaltennamen	Kommentare
`columns`	`keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type`	Diese Tabelle ist identisch mit der Cassandra-Tabelle im `system_schema` Schlüsselraum.
`tags`	`resource_id, keyspace_name, resource_name, resource_type, tags`	Fragen Sie diese Tabelle ab, um herauszufinden, ob ein Schlüsselraum Tags enthält. Weitere Informationen finden Sie unter the section called “Tabellen-Tags anzeigen”.
`types`	`keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types`	Fragen Sie diese Tabelle ab, um Informationen über benutzerdefinierte Typen (UDTs) zu erhalten. Sie können diese Tabelle beispielsweise abfragen, um alle UDTs für einen bestimmten Schlüsselraum aufzulisten. Weitere Informationen finden Sie unter the section called “User-defined Typen (UDTs)”.

Tabellennamen	Spaltennamen	Kommentare
`autoscaling`	`keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update`	Fragen Sie diese Tabelle ab, um die Auto Scaling-Einstellungen einer bereitgestellten Tabelle abzurufen. Beachten Sie, dass diese Einstellungen erst verfügbar sind, wenn die Tabelle aktiv ist. Um diese Tabelle abzufragen, müssen Sie `keyspace_name` und `table_name` in der `WHERE` Klausel angeben. Weitere Informationen finden Sie unter the section called “Sehen Sie sich die Amazon Keyspaces -Auto-Scaling-Konfiguration Ihrer Tabelle an”.

`system_multiregion_info`

Dies ist ein Amazon Keyspaces-Schlüsselraum, der Informationen zur Replikation in mehreren Regionen speichert.

Tabellennamen	Spaltennamen	Kommentare
`tables`	`keyspace_name, table_name, region, status`	Diese Tabelle enthält Informationen über Tabellen mit mehreren Regionen, z. B. die Daten, in die AWS-Regionen die Tabelle repliziert wird, und den Status der Tabelle. Sie können diese Tabelle auch abfragen, um Einstellungen aufzulisten, die spezifisch für Amazon Keyspaces

Tabellennamen	Spaltennamen	Kommentare
		sind und unter gespeichert sind. `custom_properties` Beispiel: • `capacity_mode` Um diese Tabelle abzufragen, müssen Sie `keyspace_name` und `table_name` in der `WHERE` Klausel angeben. Weitere Informationen finden Sie unter the section called “Erstellen Sie einen Schlüsselraum für mehrere Regionen”.
`keyspaces`	`keyspace_name, region, status, tables_replication_progress`	Diese Tabelle enthält Informationen über den Fortschritt eines `ALTER KEYSPACE` Vorgangs, bei dem ein Replikat zu einem Schlüsselraum hinzugefügt wird, z. B. wie viele Tabellen in der neuen Region bereits erstellt wurden und wie viele Tabellen noch in Bearbeitung sind. Beispiele finden Sie unter. the section called “Überprüfen Sie den Fortschritt der Replikation”

Tabellennamen	Spaltennamen	Kommentare
`autoscaling`	`keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update, region`	Fragen Sie diese Tabelle ab, um die Einstellungen für die auto Skalierung einer in mehreren Regionen bereitgestellten Tabelle abzurufen. Beachten Sie, dass diese Einstellungen erst verfügbar sind, wenn die Tabelle aktiv ist. Um diese Tabelle abzufragen, müssen Sie `keyspace_name` und `table_name` in der `WHERE` Klausel angeben. Weitere Informationen finden Sie unter the section called “Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling für eine Tabelle mit mehreren Regionen”.

Tabellennamen	Spaltennamen	Kommentare
`types`	`keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types, region`	Fragen Sie diese Tabelle ab, um Informationen über benutzerdefinierte Typen (UDTs) in Schlüsselräumen mit mehreren Regionen zu erhalten. Sie können diese Tabelle beispielsweise abfragen, um alle Tabellenreplikate und ihre jeweiligen AWS Regionen aufzulisten, die UDTs für einen bestimmten Schlüsselraum verwenden. Weitere Informationen finden Sie unter the section called “User-defined Typen (UDTs)”.

User-defined Typen (UDTs) in Amazon Keyspaces

Ein benutzerdefinierter Typ (UDT) ist eine Gruppierung von Feldern und Datentypen, die Sie verwenden können, um eine einzelne Spalte in Amazon Keyspaces zu definieren. Gültige Datentypen für UDTs sind alle unterstützten Cassandra-Datentypen, einschließlich Sammlungen und anderer UDTs, die Sie bereits im selben Schlüsselraum erstellt haben. Weitere Informationen zu den unterstützten Cassandra-Datentypen finden Sie unter. the section called “Unterstützung des Cassandra-Datentyps”

Sie können benutzerdefinierte Typen (UDTs) in Amazon Keyspaces verwenden, um Daten effizienter zu organisieren. Sie können beispielsweise UDTs mit verschachtelten Sammlungen erstellen, wodurch Sie eine komplexere Datenmodellierung in Ihren Anwendungen implementieren können. Sie können das Schlüsselwort frozen auch zur Definition von UDTs verwenden.

UDTs sind an einen Schlüsselraum gebunden und für alle Tabellen und UDTs im selben Schlüsselraum verfügbar. Sie können UDTs in Schlüsselräumen mit einer oder mehreren Regionen erstellen.

Sie können neue Tabellen erstellen oder bestehende Tabellen ändern und neue Spalten hinzufügen, die eine UDT verwenden. Um eine UDT mit einer verschachtelten UDT zu erstellen, muss die verschachtelte UDT eingefroren werden.

Informationen darüber, wie viele UDTs pro Schlüsselraum unterstützt werden, welche Verschachtelungsebenen unterstützt werden und welche anderen Standardwerte und Kontingente für UDTs gelten, finden Sie unter. the section called “Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces”

Informationen zur Berechnung der codierten Größe von UDTs finden Sie unter. the section called “Schätzen Sie die kodierte Größe von Datenwerten auf der Grundlage des Datentyps”

Weitere Hinweise zur CQL-Syntax finden Sie unter. the section called “Typen”

Weitere Informationen zu UDTs und zeitpunktbezogener Wiederherstellung finden Sie unter. the section called “PITR und UDTs”

Themen

- Berechtigungen für die Arbeit mit benutzerdefinierten Typen (UDTs) in Amazon Keyspaces konfigurieren
- Erstellen Sie einen benutzerdefinierten Typ (UDT) in Amazon Keyspaces
- Benutzerdefinierte Typen (UDTs) in Amazon Keyspaces anzeigen
- Löschen Sie einen benutzerdefinierten Typ (UDT) in Amazon Keyspaces

Berechtigungen für die Arbeit mit benutzerdefinierten Typen (UDTs) in Amazon Keyspaces konfigurieren

Wie Tabellen sind UDTs an einen bestimmten Schlüsselraum gebunden. Im Gegensatz zu Tabellen können Sie Berechtigungen für UDTs jedoch nicht direkt definieren. UDTs gelten nicht als Ressourcen in AWS und haben keine eindeutigen Bezeichner im Format eines Amazon-Ressourcennamens (ARN). Um einem IAM-Prinzipal Berechtigungen zur Ausführung bestimmter Aktionen an einer UDT zu erteilen, müssen Sie stattdessen Berechtigungen für den Schlüsselraum definieren, an den die UDT gebunden ist. Um mit UDTs in Schlüsselräumen mit mehreren Regionen zu arbeiten, sind zusätzliche Berechtigungen erforderlich.

Um UDTs erstellen, anzeigen oder löschen zu können, benötigt der Principal, z. B. ein IAM-Benutzer oder eine IAM-Rolle, dieselben Berechtigungen, die erforderlich sind, um dieselbe Aktion für den Schlüsselraum auszuführen, an den die UDT gebunden ist.

Weitere Informationen dazu finden Sie unter. AWS Identity and Access Managementthe section called “AWS Identity and Access Management”

Berechtigungen zum Erstellen einer UDT

Um eine UDT in einem Schlüsselraum mit nur einer Region zu erstellen, benötigt der Principal `Create` Berechtigungen für den Schlüsselraum.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cassandra:Create",
            "Resource": [
                "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
            ]
        }
    ]
}
```

Um eine UDT in einem Schlüsselraum mit mehreren Regionen zu erstellen, benötigt der Principal zusätzlich zu den `Create` Berechtigungen auch Berechtigungen für die Aktion `CreateMultiRegionResource` für den angegebenen Schlüsselraum.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action":  [ "cassandra:Create", "cassandra:CreateMultiRegionResource" ],
            "Resource": [
                "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
            ]
        }
    ]
}
```

Berechtigungen zum Anzeigen einer UDT

Um UDTs in einem Schlüsselraum mit nur einer Region anzuzeigen oder aufzulisten, benötigt der Principal Leseberechtigungen für den Systemschlüsselraum. Weitere Informationen finden Sie unter the section called “system_schema_mcs”.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"cassandra:Select",
         "Resource":[
             "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Um UDTs für einen Schlüsselraum mit mehreren Regionen anzuzeigen oder aufzulisten, benötigt der Principal Berechtigungen für die Aktionen `SELECT` und `SelectMultiRegionResource` für den Systemschlüsselraum. Weitere Informationen finden Sie unter the section called “system_multiregion_info”.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": ["cassandra:Select", "cassandra:SelectMultiRegionResource"],
         "Resource":[
             "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Berechtigungen zum Löschen einer UDT

Um eine UDT aus einem Schlüsselraum mit einer einzigen Region zu löschen, benötigt der Principal Berechtigungen für die `Drop` Aktion für den angegebenen Schlüsselraum.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cassandra:Drop",
            "Resource": [
                "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
            ]
        }
    ]
}
```

Um eine UDT aus einem Schlüsselraum mit mehreren Regionen zu löschen, benötigt der Principal Berechtigungen für die `Drop` Aktion und für die Aktion für den angegebenen `DropMultiRegionResource` Schlüsselraum.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action":  [ "cassandra:Drop", "cassandra:DropMultiRegionResource" ],
            "Resource": [
                "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
            ]
        }
    ]
}
```

Erstellen Sie einen benutzerdefinierten Typ (UDT) in Amazon Keyspaces

Um eine UDT in einem Single-Region-Schlüsselraum zu erstellen, können Sie die `CREATE TYPE` Anweisung in CQL, den `create-type` Befehl mit der oder die Konsole verwenden. AWS CLI

UDT-Namen müssen 48 Zeichen oder weniger enthalten, müssen mit einem alphabetischen Zeichen beginnen und dürfen nur alphanumerische Zeichen und Unterstriche enthalten. Amazon Keyspaces wandelt Großbuchstaben automatisch in Kleinbuchstaben um.

Alternativ können Sie einen UDT-Namen in doppelten Anführungszeichen deklarieren. Bei der Deklaration eines UDT-Namens in doppelten Anführungszeichen behält Amazon Keyspaces Großbuchstaben bei und erlaubt Sonderzeichen.

Sie können bei der Erstellung der UDT auch doppelte Anführungszeichen als Teil des Namens verwenden, aber Sie müssen jedes doppelte Anführungszeichen durch ein zusätzliches doppeltes Anführungszeichen ersetzen.

Die folgende Tabelle zeigt Beispiele für zulässige UDT-Namen. Die erste Spalte zeigt, wie Sie den Namen eingeben, wenn Sie den Typ erstellen, die zweite Spalte zeigt, wie Amazon Keyspaces den Namen intern formatiert. Amazon Keyspaces erwartet den formatierten Namen für Operationen wie. `GetType`

Eingegebener Name	Formatierter Name	Hinweis
`MY_UDT`	`my_udt`	Ohne doppelte Anführungszeichen konvertiert Amazon Keyspaces alle Großbuchstaben in Kleinbuchstaben.
`"MY_UDT"`	`MY_UDT`	Bei doppelten Anführungszeichen respektiert Amazon Keyspaces die Großbuchstaben und entfernt die doppelten Anführungszeichen aus dem formatierten Namen.
`"1234"`	`1234`	Bei doppelten Anführungszeichen kann der Name mit einer Zahl beginnen, und Amazon Keyspaces entfernt die doppelten Anführungszeichen aus dem formatierten Namen.

Eingegebener Name	Formatierter Name	Hinweis
`"Special_Ch@r@cters<>!!"`	`Special_Ch@r@cters<>!!`	Bei doppelten Anführungszeichen kann der Name Sonderzeichen enthalten, und Amazon Keyspaces entfernt die doppelten Anführungszeichen aus dem formatierten Namen.
`"nested""""""quotes"`	`nested"""quotes`	Amazon Keyspaces entfernt die äußeren doppelten Anführungszeichen und die doppelten Escape-Anführungszeichen aus dem formatierten Namen.

Console

Erstellen Sie einen benutzerdefinierten Typ (UDT) mit der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces und dann einen Keyspace aus der Liste aus.
3. Wählen Sie die Registerkarte „UDTs“.
4. Wählen Sie Create UDT
5. Geben Sie unter UDT-Details den Namen für das UDT ein. Unter UDT-Felder definieren Sie das Schema der UDT.
6. Um den Vorgang abzuschließen, wählen Sie Create UDT.

Cassandra Query Language (CQL)

Erstellen Sie einen benutzerdefinierten Typ (UDT) mit CQL

In diesem Beispiel erstellen wir eine neue Version der Buchpreistabelle, die in verwendet wird. the section called “Erstellen einer Tabelle” In dieser Tabelle speichern wir alle Auszeichnungen, die ein Autor für ein bestimmtes Buch erhält. Wir erstellen zwei UDTs, die verschachtelt sind und Informationen über das Buch enthalten, das eine Auszeichnung erhalten hat.

1. Erstellen Sie einen Schlüsselraum mit dem Namen. `catalog`

```
CREATE KEYSPACE catalog WITH REPLICATION = {'class': 'SingleRegionStrategy'};
```

2. Erstellen Sie den ersten Typ. Dieser Typ speichert BISAC-Codes, die zur Definition des Genres von Büchern verwendet werden. Ein BISAC-Code besteht aus einem alphanumerischen Code und bis zu vier Themenbereichen.

```
CREATE TYPE catalog.bisac (
    bisac_code text,
    subject1 text,
    subject2 text,
    subject3 text,
    subject4 text
);
```

3. Erstellen Sie einen zweiten Typ für Buchpreise, der die erste UDT verwendet. Das verschachtelte UDT muss eingefroren werden.

```
CREATE TYPE catalog.book (
    award_title text,
    book_title text,
    publication_date date,
    page_count int,
    ISBN text,
    genre FROZEN <bisac>
);
```

4. Erstellt eine Tabelle mit einer Spalte für den Namen des Autors und verwendet einen Listentyp für die Buchpreise. Beachten Sie, dass das in der Liste verwendete UDT eingefroren werden muss.

```
CREATE TABLE catalog.authors (
    author_name text PRIMARY KEY,
    awards list <FROZEN <book>>
);
```

5. In diesem Schritt fügen wir eine Datenzeile in die neue Tabelle ein.

```
CONSISTENCY LOCAL_QUORUM;
```

```
INSERT INTO catalog.authors (author_name, awards) VALUES (
```

```
'John Stiles' ,
[{
      award_title: 'Wolf',
      book_title: 'Yesterday',
      publication_date: '2020-10-10',
      page_count: 345,
      ISBN: '026204630X',
      genre: { bisac_code:'FIC014090', subject1: 'FICTION', subject2:
 'Historical', subject3: '20th Century', subject4: 'Post-World War II'}
      },
      {award_title: 'Richard Roe',
      book_title: 'Who ate the cake?',
      publication_date: '2019-05-13',
      page_count: 193,
      ISBN: '9780262046305',
      genre: { bisac_code:'FIC022130', subject1: 'FICTION', subject2: 'Mystery &
 Detective', subject3: 'Cozy', subject4: 'Culinary'}
      }]
);
```

6. Im letzten Schritt lesen wir die Daten aus der Tabelle.

```
SELECT * FROM catalog.authors;
```

Die Ausgabe des Befehls sollte so aussehen.

```
 author_name | awards
-------------
+-------------------------------------------------------------------------------------
 John Stiles | [{award_title: 'Wolf', book_title: 'Yesterday', publication_date:
 2020-10-10, page_count: 345, isbn: '026204630X', genre: {bisac_code:
 'FIC014090', subject1: 'FICTION', subject2: 'Historical', subject3: '20th
 Century', subject4: 'Post-World War II'}}, {award_title: 'Richard Roe',
 book_title: 'Who ate the cake?', publication_date: 2019-05-13, page_count: 193,
 isbn: '9780262046305', genre: {bisac_code: 'FIC022130', subject1: 'FICTION',
 subject2: 'Mystery & Detective', subject3: 'Cozy', subject4: 'Culinary'}}]

(1 rows)
```

Weitere Hinweise zur CQL-Syntax finden Sie unterthe section called “CREATE TYPE”.

CLI

Erstellen Sie einen benutzerdefinierten Typ (UDT) mit dem AWS CLI

1. Um einen Typ zu erstellen, können Sie die folgende Syntax verwenden.

```
aws keyspaces create-type
--keyspace-name 'my_keyspace'
--type-name 'my_udt'
--field-definitions
    '[
        {"name" : "field1", "type" : "int"},
        {"name" : "field2", "type" : "text"}
    ]'
```

2. Die Ausgabe dieses Befehls sieht ähnlich aus wie in diesem Beispiel. Beachten Sie, dass der formatierte Name der UDT `typeName` zurückgegeben wird.

```
{
    "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
    "typeName": "my_udt"
}
```

Benutzerdefinierte Typen (UDTs) in Amazon Keyspaces anzeigen

Um alle UDTs in einem Schlüsselraum mit nur einer Region anzuzeigen oder aufzulisten, können Sie die Tabelle `system_schema_mcs.types` im Systemschlüsselraum mit einer Anweisung in CQL abfragen oder die `list-type` Befehle `get-type` und mit der oder der Konsole verwenden. AWS CLI

Für beide Optionen benötigt der IAM-Principal Leseberechtigungen für den Systemschlüsselraum. Weitere Informationen finden Sie unter the section called “Konfigurieren von -Berechtigungen”.

Console

Benutzerdefinierte Typen (UDT) mit der Amazon Keyspaces-Konsole anzeigen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.

2. Wählen Sie im Navigationsbereich Keyspaces und dann einen Keyspace aus der Liste aus.
3. Wählen Sie die Registerkarte UDTs, um die Liste aller UDTs im Schlüsselraum zu überprüfen.
4. Um eine UDT im Detail zu überprüfen, wählen Sie eine UDT aus der Liste aus.
5. Auf der Registerkarte Schema können Sie das Schema überprüfen. Auf der Registerkarte Verwendet in können Sie sehen, ob diese UDT in Tabellen oder anderen UDTs verwendet wird. Beachten Sie, dass Sie nur UDTs löschen können, die weder von Tabellen noch von anderen UDTs verwendet werden.

Cassandra Query Language (CQL)

Zeigen Sie mit CQL die benutzerdefinierten Typen (UDTs) eines Schlüsselraums mit einer einzelnen Region an

1. Um die Typen zu sehen, die in einem bestimmten Schlüsselraum verfügbar sind, können Sie die folgende Anweisung verwenden.

```
SELECT type_name
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace';
```

2. Um die Details zu einem bestimmten Typ anzuzeigen, können Sie die folgende Anweisung verwenden.

```
SELECT
    keyspace_name,
    type_name,
    field_names,
    field_types,
    max_nesting_depth,
    last_modified_timestamp,
    status,
    direct_referring_tables,
    direct_parent_types
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace' AND type_name = 'my_udt';
```

3. Sie können alle UDTs auflisten, die im Konto vorhanden sind, verwenden Sie`DESC TYPE`.

```
DESC TYPES;
```

```
 Keyspace my_keyspace
 ---------------------------
 my_udt1  my_udt2

 Keyspace my_keyspace2
 ---------------------------
 my_udt1
```

4. Sie können alle UDTs im aktuell ausgewählten Schlüsselraum mit auflisten. `DESC TYPE`

```
USE my_keyspace;
my_keyspace DESC TYPES;

my_udt1  my_udt2
```

5. Um alle UDTs in einem Schlüsselraum mit mehreren Regionen aufzulisten, können Sie die Systemtabelle im Schlüsselraum abfragen. `types system_multiregion_info` Die folgende Abfrage ist ein Beispiel dafür.

```
SELECT keyspace_name, type_name, region, status FROM
 system_multiregion_info.types WHERE keyspace_name = 'mykeyspace' AND table_name
 = 'mytable';
```

Die Ausgabe dieses Befehls sieht in etwa so aus.

```
keyspace_name     | table_name         | region                | status
mykeyspace        | mytable            | us-east-1             | ACTIVE
mykeyspace        | mytable            | ap-southeast-1        | ACTIVE
mykeyspace        | mytable            | eu-west-1             | ACTIVE
```

CLI

Zeigen Sie benutzerdefinierte Typen (UDTs) mit dem AWS CLI

1. Um die in einem Schlüsselraum verfügbaren Typen aufzulisten, können Sie den Befehl verwenden. `list-types`

```
aws keyspaces list-types
--keyspace-name 'my_keyspace'
```

Die Ausgabe dieses Befehls sieht ähnlich aus wie in diesem Beispiel.

```
{
    "types": [
        "my_udt",
        "parent_udt"
    ]
}
```

2. Um die Details zu einem bestimmten Typ anzuzeigen, können Sie den `get-type` Befehl verwenden.

```
aws keyspaces get-type
--type-name 'my_udt'
--keyspace-name 'my_keyspace'
```

Die Ausgabe dieses Befehls sieht ähnlich aus wie in diesem Beispiel.

```
{
    "keyspaceName": "my_keyspace",
    "typeName": "my_udt",
    "fieldDefinitions": [
        {
            "name": "a",
            "type": "int"
        },
        {
            "name": "b",
            "type": "text"
        }
    ],
    "lastModifiedTimestamp": 1721328225776,
    "maxNestingDepth": 3
    "status": "ACTIVE",
    "directReferringTables": [],
    "directParentTypes": [
        "parent_udt"
    ],
    "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/"
}
```

Löschen Sie einen benutzerdefinierten Typ (UDT) in Amazon Keyspaces

Um eine UDT in einem Schlüsselraum zu löschen, können Sie die `DROP TYPE` Anweisung in CQL, den `delete-type` Befehl mit der oder die AWS CLI Konsole verwenden.

Console

Löschen Sie einen benutzerdefinierten Typ (UDT) mit der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces und dann einen Keyspace aus der Liste aus.
3. Wählen Sie die Registerkarte „UDTs“.
4. Wählen Sie das UDT aus, das Sie löschen möchten. Unter Benutzt in können Sie überprüfen, ob der Typ, den Sie löschen möchten, derzeit nicht von einer Tabelle oder einem anderen UDT verwendet wird.
5. Wählen Sie oberhalb der Zusammenfassung die Option Löschen aus.
6. Geben Sie den Text `Delete` in das daraufhin angezeigte Dialogfeld ein und wählen Sie „UDT löschen“.

Cassandra Query Language (CQL)

Löschen Sie einen benutzerdefinierten Typ (UDT) mit CQL

- Um einen Typ zu löschen, können Sie die folgende Anweisung verwenden.

```
DROP TYPE my_keyspace.my_udt;
```

Weitere Hinweise zur CQL-Syntax finden Sie unterthe section called “DROP TYPE”.

CLI

Löschen Sie einen benutzerdefinierten Typ (UDT) mit dem AWS CLI

1. Um einen Typ zu löschen, können Sie den folgenden Befehl verwenden.

```
aws keyspaces delete-type
--keyspace-name 'my_keyspace'
```

```
--type-name 'my_udt'
```

2. Die Ausgabe des Befehls sieht ähnlich aus wie in diesem Beispiel.

```
{
    "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
    "typeName": "my_udt"
}
```

Arbeiten mit CQL-Abfragen in Amazon Keyspaces

Dieser Abschnitt enthält eine Einführung in die Arbeit mit Abfragen in Amazon Keyspaces (für Apache Cassandra). Die verfügbaren CQL-Anweisungen zum Abfragen, Transformieren und Verwalten von Daten sind`SELECT`,`INSERT`, `UPDATE` und. `DELETE` In den folgenden Themen werden einige der komplexeren Optionen beschrieben, die beim Arbeiten mit Abfragen zur Verfügung stehen. Die vollständige Sprachsyntax mit Beispielen finden Sie unterthe section called “DML-Anweisungen.”.

Themen

- Verwenden Sie den IN-Operator mit der SELECT-Anweisung in einer Abfrage in Amazon Keyspaces
- Verwenden Sie Batch-Anweisungen in Amazon Keyspaces
- Ergebnisse mit ORDER BY in Amazon Keyspaces bestellen
- Ergebnisse in Amazon-Schlüsselräumen paginieren

Verwenden Sie **den** IN-Operator mit der SELECT-Anweisung in einer Abfrage in Amazon Keyspaces

WÄHLEN SIE IN

Sie können Daten aus Tabellen mithilfe der `SELECT` Anweisung abfragen, die eine oder mehrere Spalten für eine oder mehrere Zeilen in einer Tabelle liest und eine Ergebnismenge zurückgibt, die die Zeilen enthält, die der Anforderung entsprechen. Eine `SELECT` Anweisung enthält eine`select_clause`, die bestimmt, welche Spalten gelesen und in der Ergebnismenge zurückgegeben werden sollen. Die Klausel kann Anweisungen zum Transformieren der Daten vor der Rückgabe enthalten. Die optionale `WHERE` Klausel gibt an, welche Zeilen abgefragt werden

müssen, und besteht aus Beziehungen zu den Spalten, die Teil des Primärschlüssels sind. Amazon Keyspaces unterstützt das `IN` Schlüsselwort in der Klausel. `WHERE` In diesem Abschnitt wird anhand von Beispielen veranschaulicht, wie Amazon Keyspaces `SELECT` Anweisungen mit dem `IN` Schlüsselwort verarbeitet.

Dieses Beispiel zeigt, wie Amazon Keyspaces die `SELECT` Anweisung mit dem `IN` Schlüsselwort in Unterabfragen aufteilt. In diesem Beispiel verwenden wir eine Tabelle mit dem Namen. `my_keyspace.customers` Die Tabelle hat eine Primärschlüsselspalte`department_id`, zwei Clusterspalten `sales_region_id` und eine Spalte`sales_representative_id`, die den Namen des Kunden in der `customer_name` Spalte enthält.

```
SELECT * FROM my_keyspace.customers;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        0        |            0            |    a
  0            |        0        |            1            |    b
  0            |        1        |            0            |    c
  0            |        1        |            1            |    d
  1            |        0        |            0            |    e
  1            |        0        |            1            |    f
  1            |        1        |            0            |    g
  1            |        1        |            1            |    h
```

Mithilfe dieser Tabelle können Sie die folgende `SELECT` Anweisung ausführen, um die Kunden in den Abteilungen und Vertriebsregionen zu finden, für die Sie das `IN` Schlüsselwort in der `WHERE` Klausel verwenden. Die folgende Anweisung ist ein Beispiel dafür.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1) AND sales_region_id
 IN (0, 1);
```

Amazon Keyspaces unterteilt diese Anweisung in vier Unterabfragen, wie in der folgenden Ausgabe dargestellt.

```
SELECT * FROM my_keyspace.customers WHERE department_id = 0 AND sales_region_id = 0;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        0        |            0            |    a
  0            |        0        |            1            |    b
```

```
SELECT * FROM my_keyspace.customers WHERE department_id = 0 AND sales_region_id = 1;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        1        |          0              |    c
  0            |        1        |          1              |    d

SELECT * FROM my_keyspace.customers WHERE department_id = 1 AND sales_region_id = 0;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  1            |        0        |          0              |    e
  1            |        0        |          1              |    f

SELECT * FROM my_keyspace.customers WHERE department_id = 1 AND sales_region_id = 1;

 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  1            |        1        |           0             |    g
  1            |        1        |           1             |    h
```

Wenn das `IN` Schlüsselwort verwendet wird, paginiert Amazon Keyspaces die Ergebnisse in jedem der folgenden Fälle automatisch:

- Nach jeder 10. Unterabfrage wird verarbeitet.
- Nach der Verarbeitung von 1 MB logischer I/O.
- Wenn Sie eine konfiguriert habenPAGE SIZE, paginiert Amazon Keyspaces nach dem Lesen der Anzahl der Abfragen, die auf der Grundlage des Satzes verarbeitet werden sollen. `PAGE SIZE`
- Wenn Sie das `LIMIT` Schlüsselwort verwenden, um die Anzahl der zurückgegebenen Zeilen zu reduzieren, paginiert Amazon Keyspaces nach dem Lesen der Anzahl der Abfragen, die auf der Grundlage des Satzes verarbeitet werden sollen. `LIMIT`

Die folgende Tabelle wird verwendet, um dies anhand eines Beispiels zu veranschaulichen.

Weitere Hinweise zur Paginierung finden Sie unterthe section called “Ergebnisse paginieren”.

```
SELECT * FROM my_keyspace.customers;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
          2            |        0        |          0              |    g
```

```
          2             |        1        |          1              |    h
          2             |        2        |          2              |    i
          0             |        0        |          0              |    a
          0             |        1        |          1              |    b
          0             |        2        |          2              |    c
          1             |        0        |          0              |    d
          1             |        1        |          1              |    e
          1             |        2        |          2              |    f
          3             |        0        |          0              |    j
          3             |        1        |          1              |    k
          3             |        2        |          2              |    l
```

Sie können die folgende Anweisung für diese Tabelle ausführen, um zu sehen, wie die Paginierung funktioniert.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1, 2, 3) AND
 sales_region_id IN (0, 1, 2) AND sales_representative_id IN (0, 1);
```

Amazon Keyspaces verarbeitet diese Anweisung als 24 Unterabfragen, da die Kardinalität des kartesischen Produkts aller in dieser Abfrage enthaltenen `IN` Begriffe 24 ist.

```
 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  0            |        0        |          0              |    a
  0            |        1        |          1              |    b
  1            |        0        |          0              |    d
  1            |        1        |          1              |    e

---MORE---
 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  2            |        0        |          0              |    g
  2            |        1        |          1              |    h
  3            |        0        |          0              |    j

---MORE---
 department_id | sales_region_id | sales_representative_id | customer_name
---------------+-----------------+-------------------------+--------------
  3            |        1        |          1              |    k
```

Dieses Beispiel zeigt, wie Sie die `ORDER BY` Klausel in einer Anweisung mit dem Schlüsselwort verwenden können. `SELECT IN`

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (3, 2, 1) ORDER BY
 sales_region_id DESC;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
          3            |        2        |          2              |    l
          3            |        1        |          1              |    k
          3            |        0        |          0              |    j
          2            |        2        |          2              |    i
          2            |        1        |          1              |    h
          2            |        0        |          0              |    g
          1            |        2        |          2              |    f
          1            |        1        |          1              |    e
          1            |        0        |          0              |    d
```

Unterabfragen werden in der Reihenfolge verarbeitet, in der die Spalten Partitionsschlüssel und Clusterschlüssel in der Abfrage dargestellt werden. Im folgenden Beispiel werden zuerst Unterabfragen für die Partitionsschlüsselwerte“ 2 „verarbeitet, gefolgt von Unterabfragen für die Partitionsschlüsselwerte“ 3 „und“ 1“. Die Ergebnisse einer bestimmten Unterabfrage werden entsprechend der Ordnungsklausel der Abfrage, falls vorhanden, oder der Gruppierungsreihenfolge der Tabelle, die bei der Tabellenerstellung definiert wurde, sortiert.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (2, 3, 1) ORDER BY
 sales_region_id DESC;

         department_id | sales_region_id | sales_representative_id | customer_name
        ---------------+-----------------+-------------------------+--------------
          2            |        2        |          2              |    i
          2            |        1        |          1              |    h
          2            |        0        |          0              |    g
          3            |        2        |          2              |    l
          3            |        1        |          1              |    k
          3            |        0        |          0              |    j
          1            |        2        |          2              |    f
          1            |        1        |          1              |    e
          1            |        0        |          0              |    d
```

Verwenden Sie Batch-Anweisungen in Amazon Keyspaces

Sie können mehrere `INSERT`, `UPDATE`, und `DELETE` -Operationen in einer `BATCH` Anweisung kombinieren. `LOGGED`Batches sind die Standardeinstellung.

```
batch_statement ::=     BEGIN [ UNLOGGED ] BATCH
                        [ USING update_parameter( AND update_parameter)* ]
                        modification_statement ( ';' modification_statement )*
                        APPLY BATCH
modification_statement ::= insert_statement | update_statement | delete_statement
```

Wenn Sie eine Batch-Anweisung ausführen, fasst der Treiber alle Anweisungen im Batch zu einer einzigen Batch-Operation zusammen.

Bei der Entscheidung, welche Art von Batchoperation verwendet werden soll, können Sie die folgenden Richtlinien berücksichtigen.

Verwenden Sie protokollierte Batches, wenn:

- Sie benötigen atomare Transaktionsgarantien.
- Etwas höhere Latenzen sind ein akzeptabler Kompromiss.

Verwenden Sie nicht protokollierte Stapel, wenn:

- Sie müssen den Betrieb mit einer Partition optimieren.
- Sie möchten den Netzwerk-Overhead reduzieren.
- Sie haben hohe Durchsatzanforderungen.

Hinweise zu Kontingenten für Batch-Kontoauszüge finden Sie unterKontingente.

Nicht protokollierte Stapel

Bei nicht protokollierten Batches verarbeitet Amazon Keyspaces mehrere Vorgänge als eine einzige Anfrage, ohne ein Batch-Protokoll zu führen. Bei einem nicht protokollierten Batch-Vorgang ist es möglich, dass einige der Aktionen erfolgreich sind, während andere fehlschlagen. Stapel ohne Protokollierung sind nützlich, wenn Sie:

- Optimieren Sie den Betrieb innerhalb einer einzelnen Partition.
- Reduzieren Sie den Netzwerkverkehr, indem Sie verwandte Anfragen gruppieren.

Die Syntax für einen nicht protokollierten Stapel ähnelt der eines protokollierten Batches, allerdings mit dem Zusatz eines SchlüsselwortsUNLOGGED.

```
BEGIN UNLOGGED BATCH
```

```
    INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
    INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Geloggte Stapel

Ein protokollierter Batch kombiniert mehrere Schreibaktionen zu einer einzigen atomaren Operation. Wenn Sie einen protokollierten Batch ausführen:

- Alle Aktionen sind entweder zusammen erfolgreich oder schlagen zusammen fehl.
- Die Operation ist synchron und idempotent.
- Sie können in mehrere Amazon Keyspaces-Tabellen schreiben, solange sie sich im selben Konto befinden und. AWS AWS-Region

Protokollierte Batches können etwas höhere Latenzen aufweisen. Für Anwendungen mit hohem Durchsatz sollten Sie erwägen, nicht protokollierte Batches zu verwenden.

Für die Verwendung protokollierter Batches in Amazon Keyspaces fallen keine zusätzlichen Kosten an. Sie zahlen nur für die Schreibvorgänge, die Teil Ihrer Batch-Operationen sind. Amazon Keyspaces führt zwei grundlegende Schreibvorgänge für jede Zeile im Batch durch: einen, um die Zeile für den Batch vorzubereiten, und einen, um den Batch zu übertragen. Denken Sie bei der Kapazitätsplanung für Tabellen, die protokollierte Batches verwenden, daran, dass jede Zeile in einem Batch die doppelte Kapazität eines Standard-Schreibvorgangs benötigt. Wenn Ihre Anwendung beispielsweise einen protokollierten Batch pro Sekunde mit drei 1-KB-Zeilen ausführt, müssen Sie sechs Schreibkapazitätseinheiten (WCUs) bereitstellen, anstatt nur drei WCUs für einzelne Schreibvorgänge oder nicht protokollierte Batches.

Informationen zur Preisgestaltung finden Sie unter Preise für Amazon Keyspaces (für Apache Cassandra).

Verwenden Sie vorbereitete Kontoauszüge in protokollierten Batches

Protokollierte Batches unterstützen vorbereitete (gebundene) Anweisungen mit vollständiger atomarer Ausführung (Alles-oder-Nichts). Sie können die Leistung vorbereiteter Anweisungen mit den Atomizitätsgarantien eines protokollierten Batches kombinieren.

Sie können einen protokollierten Stapel vorbereiteter Anweisungen auf zwei Arten erstellen. Die folgenden Beispiele verwenden einen Java-Treiber, der mit Apache Cassandra kompatibel ist.

Bei der ersten Methode bereiten Sie jede Anweisung einzeln vor und fügen die gebundenen Anweisungen zu a `BatchStatement` hinzu. Dieser Ansatz unterstützt Batches, die sich über mehrere Tabellen in demselben AWS Konto und AWS-Region erstrecken. Das folgende Java-Codebeispiel zeigt diesen Ansatz:

```
import com.datastax.driver.core.BatchStatement;
import com.datastax.driver.core.PreparedStatement;
import com.datastax.driver.core.Session;

PreparedStatement psInsert = session.prepare(
    "INSERT INTO my_keyspace.orders (order_id, status) VALUES (?, ?)");
PreparedStatement psUpdate = session.prepare(
    "UPDATE my_keyspace.inventory SET qty = ? WHERE sku = ?");

BatchStatement batch = new BatchStatement(BatchStatement.Type.LOGGED);
batch.add(psInsert.bind(orderId, "CONFIRMED"));
batch.add(psUpdate.bind(newQty, sku));

try {
    session.execute(batch);
} catch (Exception e) {
    // Handle the batch execution error.
}
```

Beim zweiten Ansatz bereiten Sie einen ganzen `BEGIN BATCH ... APPLY BATCH` Block als einzelne Anweisung vor, binden ihn dann und führen ihn aus. Dieser Ansatz eignet sich am besten für einen Batch, dessen Anweisungen auf eine einzelne Tabelle abzielen. Verwenden Sie den ersten Ansatz, um einen Batch vorzubereiten, der sich über mehrere Tabellen erstreckt. Das folgende Java-Codebeispiel zeigt, wie ein ganzer Batchblock als einzelne Anweisung vorbereitet wird:

```
import com.datastax.driver.core.PreparedStatement;
import com.datastax.driver.core.Session;

PreparedStatement psBatch = session.prepare(
    "BEGIN BATCH " +
    "  INSERT INTO my_keyspace.orders (order_id, status) VALUES (?, ?); " +
    "  INSERT INTO my_keyspace.orders (order_id, status) VALUES (?, ?); " +
    "APPLY BATCH");

try {
    session.execute(psBatch.bind(orderId1, "CONFIRMED", orderId2, "PENDING"));
} catch (Exception e) {
```

```
    // Handle the batch execution error.
}
```

Sie können vorbereitete Anweisungen auch in einem protokollierten Batch wie folgt verwenden:

- Bedingte Anweisungen (`IF NOT EXISTS`oder`IF`)
- Benannte Bindungsmarkierungen
- Eine Mischung aus vorbereiteten und unvorbereiteten Anweisungen in derselben Charge
- Wiederverwendung einer einzelnen vorbereiteten Abrechnung mit unterschiedlichen Grenzwerten für den gesamten Stapel

Bewährte Verfahren für Batch-Operationen

Beachten Sie bei der Verwendung von Amazon Keyspaces-Batchoperationen die folgenden empfohlenen Verfahren.

- Aktivieren Sie die automatische Skalierung, damit Ihre Tabellen über eine ausreichende Durchsatzkapazität verfügen, um Batch-Operationen und die zusätzlichen Durchsatzanforderungen protokollierter Batches abzuwickeln.
- Verwenden Sie einzelne Operationen oder nicht protokollierte Batches, wenn Vorgänge unabhängig voneinander ausgeführt werden können, ohne die Richtigkeit der Anwendung zu beeinträchtigen.
- Entwerfen Sie Ihre Anwendung so, dass die Anzahl gleichzeitiger Aktualisierungen derselben Zeilen minimiert wird, da gleichzeitige Batchvorgänge zu Konflikten führen und fehlschlagen können.
- Verwenden Sie einzelne Schreibvorgänge oder nicht protokollierte Batches, um Massendaten mit hohem Durchsatz ohne Atomarität aufzunehmen.

Konsistenz und Parallelität

Amazon Keyspaces erzwingt die folgenden Konsistenz- und Parallelitätsregeln für protokollierte Batches:

- Bei allen Batch-Vorgängen wird das Konsistenzniveau verwendet. `LOCAL_QUORUM`
- Gleichzeitige Batches, die sich auf verschiedene Zeilen auswirken, können gleichzeitig ausgeführt werden.

- Gleichzeitige oder DELETE Operationen mit Zeilen INSERTUPDATE, die an einem laufenden Batch beteiligt sind, schlagen aufgrund eines Konflikts fehl.

Unterstützte Operatoren und Bedingungen

Unterstützte WHERE Klauseloperatoren:

- Gleichheit (=)

Nicht unterstützte Operatoren:

- Bereichsoperatoren (>, <, >=, < =)
- IN-Operator
- LIKE-Operator
- BETWEEN-Operator

Wird in protokollierten Batches nicht unterstützt:

- Mehrere Anweisungen, die dieselbe Zeile betreffen
- Gegenoperationen
- Löscht den Bereich

Eine DELETE Anweisung in einem protokollierten Batch muss auf eine einzelne Zeile abzielen und dabei den vollständigen Primärschlüssel (alle Partitionsschlüssel- und Clusterspalten) mit Gleichheitsbedingungen angeben. Das Löschen eines Bereichs, z. B. ein Löschen nach Partitionsschlüssel allein oder über einen Bereich von Clusterspalten, wird in einem protokollierten Batch nicht unterstützt. Um einen Bereich zu löschen, verwenden Sie eine eigenständige DELETE Anweisung oder einen nicht protokollierten Batch.

Fehlerbedingungen protokollierter Batch-Anweisungen

Ein protokollierter Batchvorgang kann in einem der folgenden Fälle fehlschlagen:

- Bedingungsausdrücke (wie IF NOT EXISTS oderIF) ergeben falsch.
- Eine oder mehrere Operationen enthalten ungültige Parameter.
- Die Anforderung steht in Konflikt mit einer anderen Batchoperation, die in denselben Zeilen ausgeführt wird.
- Der Tabelle fehlt die ausreichend bereitgestellte Kapazität.

- Eine Zeile überschreitet die maximale Größenbeschränkung.
- Das Eingabedatenformat ist ungültig.

Batch-Anweisungen und Replikation in mehreren Regionen

Bei Bereitstellungen mit mehreren Regionen:

- Die Operationen in der Quellregion sind synchron und atomar.
- Die Operationen in der Zielregion sind asynchron.
- Alle Batchvorgänge werden in die Zielregionen repliziert, die Isolierung während der Anwendung wird jedoch möglicherweise nicht aufrechterhalten.

Überwachen Sie Batch-Operationen

Sie können Batch-Operationen mithilfe von CloudWatch Amazon-Metriken überwachen, um Leistung, Fehler und Nutzungsmuster zu verfolgen. Amazon Keyspaces bietet die folgenden CloudWatch Metriken für die Überwachung von Batch-Vorgängen pro Tabelle:

- `SuccessfulRequestCount`— Verfolgen Sie erfolgreiche Batch-Operationen.
- `Latency`— Messen Sie die Leistung des Chargenbetriebs.
- `ConsumedWriteCapacityUnits`— Überwachen Sie den Kapazitätsverbrauch des Chargenbetriebs.

Weitere Informationen finden Sie unter the section called “Amazon-Keyspaces-Metriken”.

Zusätzlich zu den CloudWatch Metriken können Sie AWS CloudTrail damit alle Amazon Keyspaces-API-Aktionen protokollieren. Jede API-Aktion im Batch wird protokolliert, CloudTrail sodass Sie Batch-Operationen in Ihren Amazon Keyspaces-Tabellen leichter verfolgen und überprüfen können.

Beispiele für Batch-Operationen

Das Folgende ist ein Beispiel für eine einfache protokollierte Batchanweisung.

```
BEGIN BATCH
    INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
    INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Dies ist ein Beispiel für einen Batch, der `INSERT` `UPDATE` ,- und `DELETE` -Anweisungen enthält.

```
BEGIN BATCH
    INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
    UPDATE users SET firstname = 'Johnny' WHERE id = 2;
    DELETE FROM users WHERE id = 3;
APPLY BATCH;
```

Dies ist ein Beispiel für einen Batch, der clientseitige Zeitstempel verwendet.

```
BEGIN BATCH
    INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Stiles') USING
 TIMESTAMP 1669069624;
    INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Doe') USING
 TIMESTAMP 1669069624;
APPLY BATCH;

BEGIN BATCH
    UPDATE users USING TIMESTAMP 1669069624 SET firstname = 'Carlos' WHERE id = 1;
    UPDATE users USING TIMESTAMP 1669069624 SET firstname = 'Diego' WHERE id = 2;
APPLY BATCH;
```

Dies ist ein Beispiel für einen bedingten Batch.

```
BEGIN BATCH
    INSERT INTO users (id, firstname, lastname) VALUES (1, 'Jane', 'Doe') IF NOT
 EXISTS;
    INSERT INTO users (id, firstname, lastname) VALUES (2, 'John', 'Doe') IF NOT
 EXISTS;
APPLY BATCH;


BEGIN BATCH
    UPDATE users SET lastname = 'Stiles' WHERE id = 1 IF lastname = 'Doe';
    UPDATE users SET lastname = 'Stiles' WHERE id = 2 IF lastname = 'Doe';
APPLY BATCH;
```

Dies ist ein Beispiel für einen Stapel, der Time to Live (TTL) verwendet.

```
BEGIN BATCH
    INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe') USING TTL
 3600;
```

```
    INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith') USING TTL
 7200;
APPLY BATCH;
```

Dies ist ein Beispiel für eine Batch-Anweisung, die mehrere Tabellen aktualisiert.

```
BEGIN BATCH
    INSERT INTO users (id, firstname) VALUES (1, 'John');
    INSERT INTO user_emails (user_id, email) VALUES (1, 'john@example.com');
APPLY BATCH;
```

Dies ist ein Beispiel für eine Batchoperation, bei der benutzerdefinierte Typen (UDTs) verwendet werden. Das Beispiel geht davon aus, dass das UDT `address` existiert.

```
BEGIN BATCH
    INSERT INTO users (id, firstname, address)
    VALUES (1, 'John', {street: '123 Main St', city: 'NYC', zip: '10001'});
    INSERT INTO users (id, firstname, address)
    VALUES (2, 'Jane', {street: '456 Oak Ave', city: 'LA', zip: '90210'});
APPLY BATCH;

BEGIN BATCH
    UPDATE users SET address.zip = '10002' WHERE id = 1;
    UPDATE users SET address.city = 'Boston' WHERE id = 2;
APPLY BATCH;
```

Ergebnisse mit ORDER BY **in Amazon Keyspaces** bestellen

Die `ORDER BY` Klausel gibt die Sortierreihenfolge der in einer `SELECT` Anweisung zurückgegebenen Ergebnisse an. Die Anweisung verwendet eine Liste von Spaltennamen als Argumente, und für jede Spalte können Sie die Sortierreihenfolge für die Daten angeben. Sie können gruppierende Spalten nur in Ordnungsklauseln angeben. Spalten ohne Cluster-Bildung sind nicht zulässig.

Die beiden verfügbaren Sortierreihenfolge-Optionen für die zurückgegebenen Ergebnisse sind für die aufsteigende und `ASC` `DESC` für die absteigende Sortierreihenfolge.

```
SELECT * FROM my_keyspace.my_table ORDER BY (col1 ASC, col2 DESC, col3 ASC);

         col1 | col2 | col3
        ------+------+------
```

```
 0   |  6   |  a
 1   |  5   |  b
 2   |  4   |  c
 3   |  3   |  d
 4   |  2   |  e
 5   |  1   |  f
 6   |  0   |  g
```

```
SELECT * FROM my_keyspace.my_table ORDER BY (col1 DESC, col2 ASC, col3 DESC);

 col1 | col2 | col3
------+------+------
 6   |  0   |  g
 5   |  1   |  f
 4   |  2   |  e
 3   |  3   |  d
 2   |  4   |  c
 1   |  5   |  b
 0   |  6   |  a
```

Wenn Sie die Sortierreihenfolge in der Abfrageanweisung nicht angeben, wird die Standardreihenfolge der Clusterspalte verwendet.

Die möglichen Sortierreihenfolgen, die Sie in einer Sortierklausel verwenden können, hängen von der Sortierreihenfolge ab, die jeder Cluster-Spalte bei der Tabellenerstellung zugewiesen wurde. Abfrageergebnisse können nur in der Reihenfolge sortiert werden, die für alle Cluster-Spalten bei der Tabellenerstellung definiert wurde, oder in umgekehrter Reihenfolge der definierten Sortierreihenfolge. Andere mögliche Kombinationen sind nicht zulässig.

Wenn die Tabelle beispielsweise (col1 ASC, col2 DESC, col3 ASC) `CLUSTERING ORDER` lautet, dann `ORDER BY` sind die gültigen Parameter für entweder (col1 ASC, col2 DESC, col3 ASC) oder (col1 DESC, col2 ASC, col3 DESC) oder (col1 DESC, col2 ASC, col3 DESC). Weitere Informationen dazu finden Sie unter. `CLUSTERING ORDER` `table_options` the section called “CREATE TABLE”

Ergebnisse in Amazon-Schlüsselräumen paginieren

Amazon Keyspaces paginiert automatisch die Ergebnisse von `SELECT` Anweisungen, wenn die zur Verarbeitung der Anweisung gelesenen Daten 1 MB überschreiten. `SELECT` Bei der Paginierung werden die Ergebnisse der `SELECT` Anweisung in „Seiten“ mit Daten unterteilt, die 1 MB (oder weniger) groß sind. Eine Anwendung kann die erste Ergebnisseite verarbeiten, dann die zweite

Seite und so weiter. Kunden sollten bei der Verarbeitung von `SELECT` Abfragen, die mehrere Zeilen zurückgeben, immer nach Paginierungstoken suchen.

Wenn ein Kunde eine ausgibt, für `PAGE SIZE` die mehr als 1 MB an Daten gelesen werden müssen, teilt Amazon Keyspaces die Ergebnisse automatisch in mehrere Seiten auf, basierend auf den Inkrementen der Lesevorgänge von 1 MB.

Wenn beispielsweise die durchschnittliche Größe einer Zeile 100 KB beträgt und Sie einen Wert `PAGE SIZE` von 20 angeben, paginiert Amazon Keyspaces die Daten automatisch, nachdem es 10 Zeilen gelesen hat (1000 KB gelesene Daten).

Da Amazon Keyspaces Ergebnisse auf der Grundlage der Anzahl der Zeilen paginiert, die es zur Verarbeitung einer Anforderung liest, und nicht auf der Anzahl der im Resultset zurückgegebenen Zeilen, enthalten einige Seiten möglicherweise keine Zeilen, wenn Sie gefilterte Abfragen ausführen.

Wenn Sie beispielsweise den Wert `PAGE SIZE` auf 10 setzen und Keyspaces 30 Zeilen auswertet, um Ihre `SELECT` Abfrage zu verarbeiten, gibt Amazon Keyspaces drei Seiten zurück. Wenn nur eine Teilmenge der Zeilen mit Ihrer Abfrage übereinstimmt, haben einige Seiten möglicherweise weniger als 10 Zeilen. Ein Beispiel, wie sich die Anzahl `PAGE SIZE` der `LIMIT` Abfragen auf die Lesekapazität auswirken kann, finden Sie unterthe section called “Schätzen Sie den Lesekapazitätsverbrauch von Grenzabfragen”.

Einen Vergleich mit der Apache Cassandra-Paginierung finden Sie unter. the section called “Paginierung”

Arbeiten mit Change Data Capture (CDC) -Streams in Amazon Keyspaces

Amazon Keyspaces Change Data Capture (CDC) zeichnet Änderungsereignisse auf Zeilenebene aus einer Amazon Keyspaces-Tabelle nahezu in Echtzeit auf.

Amazon Keyspaces CDC ermöglicht ereignisgesteuerte Anwendungsfälle wie industrielles IoT und Betrugserkennung sowie Anwendungsfälle für die Datenverarbeitung wie Volltextsuche und Datenarchivierung. Die Änderungsereignisse, die Amazon Keyspaces CDC in Streams erfasst, können von nachgelagerten Anwendungen verarbeitet werden, die geschäftskritische Funktionen wie Datenanalyse, Textsuche training/inference, ML und kontinuierliche Datensicherungen für die Archivierung ausführen. Sie können beispielsweise Stream-Daten zur weiteren Verarbeitung an AWS Analyse- und Speicherdienste wie Amazon OpenSearch Service, Amazon Redshift und Amazon S3 übertragen.

Amazon Keyspaces CDC bietet zeitlich geordnete und deduplizierte Änderungsdatensätze für Tabellen mit automatischer Skalierung des Datendurchsatzes und einer Aufbewahrungszeit von bis zu 24 Stunden.

Amazon Keyspaces CDC-Streams sind vollständig serverlos, und Sie müssen die Dateninfrastruktur für die Erfassung von Änderungsereignissen nicht verwalten. Darüber hinaus verbraucht Amazon Keyspaces CDC keine Tabellenkapazität, weder für die Datenverarbeitung noch für die Speicherung. Weitere Informationen finden Sie unter the section called “Wie funktionieren CDC-Streams”.

Sie können die Amazon Keyspaces Streams-API verwenden, um Anwendungen zu erstellen, die Amazon Keyspaces CDC-Streams verwenden, und auf der Grundlage der Inhalte Maßnahmen ergreifen. Informationen zu verfügbaren Endpunkten finden Sie unter. the section called “Greifen Sie auf CDC-Stream-Endpunkte zu”

Eine vollständige Liste aller Operationen, die für Amazon Keyspaces in der Streams-API verfügbar sind, finden Sie unter Amazon Keyspaces Streams API-Referenz.

Themen

- So funktionieren Change Data Capture (CDC) -Streams in Amazon Keyspaces
- So verwenden Sie Change Data Capture (CDC) -Streams in Amazon Keyspaces

So funktionieren Change Data Capture (CDC) -Streams in Amazon Keyspaces

Dieser Abschnitt bietet einen Überblick darüber, wie Change Data Capture (CDC) -Streams in Amazon Keyspaces funktionieren.

Amazon Keyspaces Change Data Capture (CDC) zeichnet eine geordnete Abfolge von Änderungen auf Zeilenebene in Amazon Keyspaces-Tabellen auf und speichert diese Informationen bis zu 24 Stunden lang in einem Protokoll namens Stream. Jede Änderung auf Zeilenebene generiert einen neuen CDC-Datensatz, der die Primärschlüsselspalteninformationen sowie die Vorher- und Nachher-Zustände der Zeile einschließlich aller Spalten enthält. Anwendungen können auf den Stream zugreifen und die Mutationen nahezu in Echtzeit anzeigen.

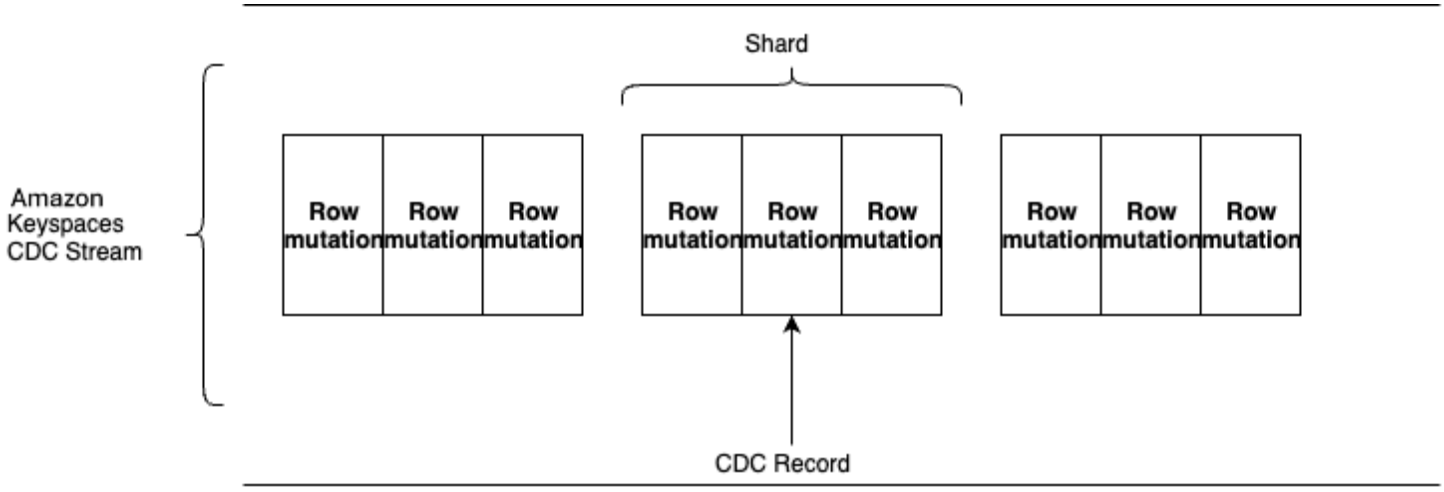
Wenn Sie CDC für Ihre Tabelle aktivieren, erstellt Amazon Keyspaces einen neuen CDC-Stream und beginnt, Informationen über jede Änderung in der Tabelle zu erfassen. Der CDC-Stream hat einen Amazon-Ressourcennamen (ARN) mit dem folgenden Format:

```
arn:${Partition}:cassandra:{Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

Sie können den Informationstyp oder den Ansichtstyp auswählen, den der CDC-Stream für jeden Datensatz erfasst, wenn Sie den CDC-Stream zum ersten Mal aktivieren. Sie können den Ansichtstyp des Streams danach nicht mehr ändern. Amazon Keyspaces unterstützt die folgenden Ansichtstypen:

- `NEW_AND_OLD_IMAGES`— Erfasst die Versionen der Zeile vor und nach der Mutation. Das ist die Standardeinstellung.
- `NEW_IMAGE`— Erfasst die Version der Zeile nach der Mutation.
- `OLD_IMAGE`— Erfasst die Version der Zeile vor der Mutation.
- `KEYS_ONLY`— Erfasst die Partitions- und Clusterschlüssel der Zeile, die mutiert wurde.

Jeder CDC-Stream besteht aus Datensätzen. Jeder Datensatz steht für eine einzelne Zeilenänderung in einer Amazon Keyspaces-Tabelle. Datensätze sind logisch in Gruppen organisiert, die als Shards bezeichnet werden. Diese Gruppen sind logisch nach Bereichen des Primärschlüssels (Kombination aus Partitionsschlüssel und Cluster-Schlüsselbereichen) organisiert und sind ein internes Konstrukt von Amazon Keyspaces. Jeder Shard fungiert als Container für mehrere Datensätze und enthält Informationen, die für den Zugriff auf und die Iteration durch diese Datensätze erforderlich sind.



Jedem CDC-Datensatz wird eine Sequenznummer zugewiesen, die die Reihenfolge widerspiegelt, in der der Datensatz innerhalb des Shards veröffentlicht wurde. Die Sequenznummer nimmt garantiert zu und ist innerhalb jedes Shards eindeutig.

Amazon Keyspaces erstellt und löscht Shards automatisch. Basierend auf den Datenverkehrslasten kann Amazon Keyspaces auch Shards im Laufe der Zeit teilen oder zusammenführen. Amazon

Keyspaces können beispielsweise einen Shard in mehrere neue Shards aufteilen oder Shards zu einem neuen einzelnen Shard zusammenführen. Die Amazon Keyspaces-APIs veröffentlichen die Shard- und CDC-Stream-Informationen, damit nutzende Anwendungen Datensätze in der richtigen Reihenfolge verarbeiten können, indem sie auf das gesamte Abstammungsdiagramm eines Shards zugreifen.

Amazon Keyspaces CDC basiert auf den folgenden Prinzipien, auf die Sie sich bei der Erstellung Ihrer Anwendung verlassen können:

- Jeder Mutationsdatensatz auf Zeilenebene erscheint genau einmal im CDC-Stream.
- Wenn Sie Shards in der Reihenfolge ihrer Abstammung konsumieren, erscheint jeder Mutationsdatensatz auf Zeilenebene in derselben Reihenfolge wie die tatsächliche Mutationsreihenfolge auf dem Primärschlüssel.

Themen

- So funktioniert die Datenspeicherung für CDC-Streams in Amazon Keyspaces
- So funktioniert der Ablauf von Time to Live (TTL) -Daten mit CDC-Streams in Amazon Keyspaces
- So funktionieren Batch-Operationen für CDC-Streams in Amazon Keyspaces
- So funktionieren statische Spalten in CDC-Streams in Amazon Keyspaces
- So funktioniert die Verschlüsselung im Ruhezustand für CDC-Streams in Amazon Keyspaces
- So funktioniert die multiregionale Replikation für CDC-Streams in Amazon Keyspaces
- CDC-Streams und Integration mit AWS service

So funktioniert die Datenspeicherung für CDC-Streams in Amazon Keyspaces

Amazon Keyspaces speichert die Aufzeichnungen im CDC-Stream für einen Zeitraum von 24 Stunden. Sie können die Aufbewahrungsfrist nicht ändern. Wenn Sie CDC für eine Tabelle deaktivieren, sind die Daten im Stream weiterhin 24 Stunden lang lesbar. Nach dieser Zeit laufen die Daten ab und die Datensätze werden automatisch gelöscht.

So funktioniert der Ablauf von Time to Live (TTL) -Daten mit CDC-Streams in Amazon Keyspaces

Amazon Keyspaces zeigt die Ablaufzeit sowohl auf column/cell Ebene als auch auf Zeilenebene in einem Metadatenfeld an, das `expirationTime` in den CDC-Änderungsdatensätzen aufgerufen wird. Wenn Amazon Keyspaces TTL den Ablauf einer Zelle feststellt, erstellt CDC einen neuen

Änderungsdatensatz, in dem TTL als Ursprung der Änderung angegeben ist. Weitere Informationen zu TTL finden Sie unter the section called “Daten mit Time to Live ablaufen lassen”.

So funktionieren Batch-Operationen für CDC-Streams in Amazon Keyspaces

Batch-Operationen sind intern in einzelne Änderungen auf Zeilenebene unterteilt. Amazon Keyspaces speichert alle Datensätze in CDC-Streams auf Zeilenebene, auch wenn die Änderung in einem Batch-Vorgang erfolgt ist. Amazon Keyspaces verwaltet die Reihenfolge der Datensätze innerhalb des CDC-Streams in derselben Reihenfolge wie die Mutationsreihenfolge, die auf Zeilenebene oder auf dem Primärschlüssel aufgetreten ist.

So funktionieren statische Spalten in CDC-Streams in Amazon Keyspaces

Statische Spaltenwerte werden von allen Zeilen in einer Partition in Cassandra gemeinsam genutzt. Aufgrund dieses Verhaltens erfasst Amazon Keyspaces alle Aktualisierungen einer statischen Spalte als separaten Datensatz im CDC-Stream. Die folgenden Beispiele fassen das Verhalten statischer Spaltenmutationen zusammen:

- Wenn nur die statische Spalte aktualisiert wird, enthält der CDC-Stream eine Zeilenänderung für die statische Spalte als einzige Spalte in der Zeile.
- Wenn eine Zeile ohne Änderung der statischen Spalte aktualisiert wird, enthält der CDC-Stream eine Zeilenänderung, die alle Spalten außer der statischen Spalte enthält.
- Wenn eine Zeile zusammen mit der statischen Spalte aktualisiert wird, enthält der CDC-Stream zwei separate Zeilenänderungen, eine für die statische Spalte und die andere für den Rest der Zeile.

So funktioniert die Verschlüsselung im Ruhezustand für CDC-Streams in Amazon Keyspaces

Um die ruhenden Daten im von CDC bestellten Protokoll zu verschlüsseln, verwendet Amazon Keyspaces denselben Verschlüsselungsschlüssel, der bereits für die Tabelle verwendet wird. Weitere Informationen zur Verschlüsselung im Ruhezustand finden Sie unter the section called “Verschlüsselung im Ruhezustand”.

So funktioniert die multiregionale Replikation für CDC-Streams in Amazon Keyspaces

Sie können CDC-Streams für einzelne Replikate einer Tabelle mit mehreren Regionen aktivieren und deaktivieren, indem Sie entweder die `update-table` API oder den CQL-Befehl verwenden. `ALTER`

TABLE Aufgrund der asynchronen Replikation und Konfliktlösung sind CDC-Streams für Tabellen mit mehreren Regionen nicht überall konsistent. AWS-Regionen Daher können die Datensätze, die Amazon Keyspaces im Stream erfasst, in verschiedenen Regionen in einer anderen Reihenfolge erscheinen.

Weitere Informationen zur multiregionalen Replikation finden Sie unter. the section called “Multi-Region Replikation”

CDC-Streams und Integration mit AWS service

Wie arbeitet man mit VPC-Endpunkten für CDC-Streams in Amazon Keyspaces

Sie können VPC-Endpunkte verwenden, um auf CDC-Streams von Amazon Keyspaces zuzugreifen. Informationen zum Erstellen und Zugreifen auf VPC-Endpunkte für Streams finden Sie unter. the section called “Verwenden von Schnittstellen-VPC-Endpunkten mit Streams”

So CloudWatch funktioniert die Überwachung für CDC-Streams in Amazon Keyspaces

Sie können Amazon verwenden, um API-Aufrufe CloudWatch zu überwachen, die an den CDC-Endpunkt von Amazon Keyspaces gesendet werden. Weitere Informationen zu den verfügbaren Metriken finden Sie unter. the section called “Metriken für Amazon Keyspaces Change Data Capture (CDC)”

So CloudTrail funktioniert die Protokollierung mit für CDC-Streams in Amazon Keyspaces

Amazon Keyspaces CDC ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS Dienstes in Amazon Keyspaces bereitstellt. CloudTrail erfasst DDL-API-Aufrufe (Data Definition Language) und DML-API-Aufrufe (Data Manipulation Language) für Amazon Keyspaces als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Amazon Keyspaces-Konsole und programmatische Aufrufe der Amazon Keyspaces-API-Operationen.

Weitere Informationen zu den CDC-Ereignissen, die von erfasst wurden, finden Sie unter. CloudTrail the section called “Protokollierung mit CloudTrail”

So funktioniert das Tagging für CDC-Streams in Amazon Keyspaces

Die CDC-Streams von Amazon Keyspaces sind eine markierbare Ressource. Sie können einen Stream taggen, wenn Sie eine Tabelle programmgesteuert mit CQL, dem SDK oder dem erstellen. AWS AWS CLI Sie können auch vorhandene Streams taggen, Tags löschen oder Tags

eines Streams anzeigen. Weitere Informationen finden Sie unter the section called “Taggen Sie Schlüsselräume, Tabellen und Streams”.

So verwenden Sie Change Data Capture (CDC) -Streams in Amazon Keyspaces

Themen

- Konfigurieren Sie die Berechtigungen für die Arbeit mit CDC-Streams in Amazon Keyspaces
- Wie greife ich in Amazon Keyspaces auf CDC-Stream-Endpunkte zu
- Aktivieren Sie einen CDC-Stream, wenn Sie eine neue Tabelle in Amazon Keyspaces erstellen
- Aktivieren Sie einen CDC-Stream für eine vorhandene Tabelle in Amazon Keyspaces
- Deaktivieren Sie einen CDC-Stream in Amazon Keyspaces
- CDC-Streams in Amazon Keyspaces anzeigen
- Greifen Sie in Amazon Keyspaces auf Datensätze in CDC-Streams zu
- Verwenden Sie die Kinesis Client Library (KCL), um Amazon Keyspaces-Streams zu verarbeiten

Konfigurieren Sie die Berechtigungen für die Arbeit mit CDC-Streams in Amazon Keyspaces

Um CDC-Streams zu aktivieren, benötigt der Principal, beispielsweise ein IAM-Benutzer oder eine IAM-Rolle, die folgenden Berechtigungen.

Weitere Informationen dazu finden Sie AWS Identity and Access Management unter. the section called “AWS Identity and Access Management”

Berechtigungen zum Aktivieren eines CDC-Streams für eine Tabelle

Um einen CDC-Stream für eine Amazon Keyspaces-Tabelle zu aktivieren, benötigt der Principal zunächst die Berechtigungen, um eine Tabelle zu erstellen oder zu ändern, und zweitens die Berechtigungen, um die mit dem Service verknüpfte Rolle zu erstellen. AWSServiceRoleForAmazonKeyspacesCDC Amazon Keyspaces verwendet die serviceverknüpfte Rolle, um in Ihrem Namen CloudWatch Kennzahlen in Ihrem Konto zu veröffentlichen

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
    "Version":"2012-10-17",
```

```
    "Statement":[
        {
            "Effect":"Allow",
            "Action":[
                "cassandra:Create",
                "cassandra:CreateMultiRegionResource",
                "cassandra:Alter",
                "cassandra:AlterMultiRegionResource"
            ],
            "Resource":[
                "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/*",
                "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
            ]
        },
        {
            "Sid": "KeyspacesCDCServiceLinkedRole",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/cassandra-
streams.amazonaws.com/AWSServiceRoleForAmazonKeyspacesCDC",
            "Condition": {
              "StringLike": {
                "iam:AWSServiceName": "cassandra-streams.amazonaws.com"
              }
            }
        }
    ]
}
```

Um einen Stream zu deaktivieren, sind nur `ALTER TABLE` Berechtigungen erforderlich.

Berechtigungen zum Anzeigen eines CDC-Streams

Um CDC-Streams anzuzeigen oder aufzulisten, benötigt der Prinzipal Leseberechtigungen für den Systemschlüsselraum. Weitere Informationen finden Sie unter the section called “system_schema_mcs”.

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
         "Effect":"Allow",
         "Action":"cassandra:Select",
         "Resource":[
             "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Um CDC-Streams mit der oder der Amazon Keyspaces-API anzuzeigen AWS CLI oder aufzulisten, benötigt der Prinzipal zusätzliche Berechtigungen für die Aktionen und. `cassandra:ListStreams` `cassandra:GetStream`

Die folgende IAM-Richtlinie ist ein Beispiel dafür.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select",
        "cassandra:ListStreams",
        "cassandra:GetStream"
      ],
      "Resource": "*"
    }
  ]
}
```

Berechtigungen zum Lesen eines CDC-Streams

Um CDC-Streams lesen zu können, benötigt der Prinzipal die folgenden Berechtigungen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:GetStream",
            "cassandra:GetShardIterator",
            "cassandra:GetRecords"
```

```
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/stream_label"
         ]
      }
   ]
}
```

Berechtigungen zur Verarbeitung von CDC-Streams von Amazon Keyspaces mit der Kinesis Client Library (KCL)

Um Amazon Keyspaces CDC-Streams mit KCL zu verarbeiten, benötigt der IAM-Principal die folgenden Berechtigungen.

- `Amazon Keyspaces`— Read-only Zugriff auf einen bestimmten Amazon Keyspaces CDC-Stream.
- `DynamoDB`— Berechtigungen zum Erstellen von `shard lease` Tabellen, Lese- und Schreibzugriff auf die Tabellen und Lesezugriff auf den Index, wie für die KCL-Stream-Verarbeitung erforderlich.
- `CloudWatch`— Berechtigungen zur Veröffentlichung von Metrikdaten aus Amazon Keyspaces CDC-Streams, die mit KCL verarbeitet werden, im Namespace Ihrer KCL-Client-Anwendung in Ihrem Konto. CloudWatch Weitere Informationen zur Überwachung finden Sie unter Überwachen der Kinesis-Clientbibliothek mit Amazon. CloudWatch

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:GetStream",
            "cassandra:GetShardIterator",
            "cassandra:GetRecords"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/stream_label"
         ]
      },
      {
         "Effect":"Allow",
```

```
         "Action":[
            "dynamodb:CreateTable",
            "dynamodb:DescribeTable",
            "dynamodb:UpdateTable",
            "dynamodb:GetItem",
            "dynamodb:UpdateItem",
            "dynamodb:PutItem",
            "dynamodb:DeleteItem",
            "dynamodb:Scan"
         ],
         "Resource":[
            "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "dynamodb:CreateTable",
            "dynamodb:DescribeTable",
            "dynamodb:GetItem",
            "dynamodb:UpdateItem",
            "dynamodb:PutItem",
            "dynamodb:DeleteItem",
            "dynamodb:Scan"
         ],
         "Resource":[
            "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME-
WorkerMetricStats",
            "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME-
CoordinatorState"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "dynamodb:Query"
         ],
         "Resource":[
            "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME/index/
*"
         ]
      },
      {
         "Effect":"Allow",
```

```
         "Action":[
            "cloudwatch:PutMetricData"
         ],
         "Resource":"*"
      }
   ]
}
```

Wie greife ich in Amazon Keyspaces auf CDC-Stream-Endpunkte zu

Amazon Keyspaces verwaltet in jedem Bereich, in AWS-Region dem Amazon Keyspaces verfügbar ist, separate Endpunkte für keyspaces/tables und für CDC-Streams. Um auf einen CDC-Stream zuzugreifen, wählen Sie die Region der Tabelle aus und ersetzen Sie das `cassandra` Präfix durch den Namen des Endpunkts, wie `cassandra-streams` im folgenden Beispiel gezeigt:

Name der Region	Region	Endpunkt für Schlüsselräume und Tabellen	Endpunkt für Streams
USA Ost (Nord-Virginia)	us-east-1	*cassandra* .us-east-1.api.aws	*cassandra-streams* .us-east-1.api.aws

Die folgende Tabelle enthält eine vollständige Liste der verfügbaren öffentlichen Endpunkte für. Amazon Keyspaces change data capture streams Amazon Keyspaces CDC streams unterstützt sowohl IPv4 als auch IPv6. Alle öffentlichen Endpunkte sind beispielsweise Dual-Stack-Endpunkte`cassandra-streams.us-east-1.api.aws`, die für IPv4 und IPv6 konfiguriert werden können.

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
USA Ost (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
USA Ost (Nord-Virginia)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
USA West (Nordkalifornien)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	
USA West (Oregon)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
Afrika (Kapstadt)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	
Asien-Pazifik (Hongkong)	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	
Asien-Pazifik (Hyderabad)	ap-south-2	cassandra-streams.ap-south-2.api.aws	HTTPS	
Asien-Pazifik (Jakarta)	ap-southeast-3	cassandra-streams.ap-southeast-3.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
Asien-Pazifik (Malaysia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Asien-Pazifik (Melbourne)	ap-southeast-4	cassandra-streams.ap-southeast-4.api.aws	HTTPS	
Asien-Pazifik (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	
Asien-Pazifik (Neuseeland)	ap-southeast-6	cassandra-streams.ap-southeast-6.api.aws	HTTPS	
Asien-Pazifik (Osaka)	ap-northeast-3	cassandra-streams.ap-northeast-3.api.aws	HTTPS	
Asien-Pazifik (Seoul)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	
Asien-Pazifik (Singapur)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)
Asien-Pazifik (Sydney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS
Asien-Pazifik (Taipeh)	ap-east-2	cassandra-streams.ap-east-2.api.aws	HTTPS
Asien-Pazifik (Thailand)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS
Asien-Pazifik (Tokio)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS
Kanada (Zentral)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS
Kanada West (Calgary)	ca-west-1	cassandra-streams.ca-west-1.api.aws	HTTPS
Europa (Frankfurt)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS
Europa (Irland)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS
Europa (London)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
Europa (Mailand)	eu-south-1	cassandra-streams.eu-south-1.api.aws	HTTPS	
Europa (Paris)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	
Europa (Spanien)	eu-south-2	cassandra-streams.eu-south-2.api.aws	HTTPS	
Europa (Stockholm)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Europa (Zürich)	eu-central-2	cassandra-streams.eu-central-2.api.aws	HTTPS	
Israel (Tel Aviv)	il-central-1	cassandra-streams.il-central-1.api.aws	HTTPS	
Mexiko (Zentral)	mx-central-1	cassandra-streams.mx-central-1.api.aws	HTTPS	
Naher Osten (Bahrain)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	
Naher Osten (VAE)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
Südamerika (São Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	

Name der Region	Region	Endpunkt	Protocol (Protokoll)	
AWS GovCloud (US-East)	us-gov-east-1	cassandra-streams.us-gov-east-1.api.aws	HTTPS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra-streams.us-gov-west-1.api.aws	HTTPS	

Aktivieren Sie einen CDC-Stream, wenn Sie eine neue Tabelle in Amazon Keyspaces erstellen

Um einen CDC-Stream beim Erstellen einer Tabelle zu aktivieren, können Sie die `CREATE TABLE` Anweisung in CQL oder den Befehl mit dem `create-table` verwenden. AWS CLI

Für jede geänderte Zeile in der Tabelle kann Amazon Keyspaces auf der Grundlage der `view_type` von Ihnen ausgewählten Änderungen die folgenden Änderungen erfassen: `cdc_specification`

- `NEW_AND_OLD_IMAGES`— beide Versionen der Zeile, vor und nach der Änderung. Das ist die Standardeinstellung.
- `NEW_IMAGE`— die Version der Zeile nach der Änderung.
- `OLD_IMAGE`— die Version der Zeile vor der Änderung.
- `KEYS_ONLY`— die Partitions- und Clusterschlüssel der Zeile, die geändert wurde.

Hinweise zum Taggen eines Streams finden Sie unterthe section called “Erstellen Sie einen Stream mit Tags mit neuer Tabelle”.

> **Note**
>
> Amazon Keyspaces CDC erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForAmazonKeyspacesCDC`), die in Ihrem Namen Metrikdaten aus

Amazon Keyspaces CDC-Streams `"cloudwatch:namespace": "AWS/Cassandra"` in Ihrem CloudWatch Konto veröffentlicht. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “CDC-Streams”.

Cassandra Query Language (CQL)

Aktivieren Sie einen CDC-Stream, wenn Sie eine Tabelle mit CQL erstellen

1.
```
CREATE TABLE mykeyspace.mytable (a text, b text, PRIMARY KEY(a))
WITH CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'NEW_IMAGE'}} AND CDC
 = TRUE;
```

2. Um die Stream-Einstellungen zu bestätigen, können Sie die folgende Anweisung verwenden.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
 system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
 'mytable';
```

Die Ausgabe dieser Anweisung sollte in etwa so aussehen.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
 system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
 'mytable';keyspace_name | table_name | cdc  | custom_properties
---------------+------------+------
+-------------------------------------------------------------------------------------------
    mykeyspace |   mytable  | True | {'capacity_mode':
 {'last_update_to_pay_per_request_timestamp': '1741383893782',
 'throughput_mode': 'PAY_PER_REQUEST'}, 'cdc_specification':
 {'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable/stream/2025-03-07T21:44:53.783', 'status': 'ENABLED',
 'view_type': 'NEW_IMAGE'}, 'encryption_specification': {'encryption_type':
 'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status': 'disabled'}}>
```

CLI

Aktivieren Sie einen CDC-Stream, wenn Sie eine Tabelle mit dem erstellen AWS CLI

1. Um einen Stream zu erstellen, können Sie die folgende Syntax verwenden.

```
aws keyspaces create-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--schema-definition 'allColumns=[{name=a,type=text},
{name=b,type=text}],partitionKeys=[{name=a}]' \
--cdc-specification status=ENABLED,viewType=NEW_IMAGE
```

2. Die Ausgabe dieses Befehls zeigt die `create-table` Standardantwort und sieht ähnlich aus wie in diesem Beispiel.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable" }
```

Aktivieren Sie einen CDC-Stream für eine vorhandene Tabelle in Amazon Keyspaces

Um einen CDC-Stream für eine bestehende Tabelle zu aktivieren, können Sie die `ALTER TABLE` Anweisung in CQL, den `update-table` Befehl mit dem AWS CLI, oder Sie können die Konsole verwenden.

Für jede geänderte Zeile in der Tabelle kann Amazon Keyspaces auf der Grundlage der `view_type` von Ihnen ausgewählten Änderungen die folgenden Änderungen erfassen: `cdc_specification`

- `NEW_AND_OLD_IMAGES`— beide Versionen der Zeile, vor und nach der Änderung. Das ist die Standardeinstellung.
- `NEW_IMAGE`— die Version der Zeile nach der Änderung.
- `OLD_IMAGE`— die Version der Zeile vor der Änderung.
- `KEYS_ONLY`— die Partitions- und Clusterschlüssel der Zeile, die geändert wurde.

Hinweise zum Taggen eines Streams finden Sie unterthe section called “Füge neue Tags zu einem Stream hinzu”.

> **Note**
>
> Amazon Keyspaces CDC erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForAmazonKeyspacesCDC`), die in Ihrem Namen Metrikdaten aus Amazon Keyspaces CDC-Streams `"cloudwatch:namespace": "AWS/Cassandra"` in

Ihrem CloudWatch Konto veröffentlicht. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “CDC-Streams”.

Cassandra Query Language (CQL)

Aktivieren Sie einen Stream (CDC-Stream) mit CQL

Sie können es verwendenALTER TABLE, um einen Stream für eine vorhandene Tabelle zu aktivieren.

1. Im folgenden Beispiel wird ein Stream erstellt, der nur Änderungen an den Partitions- und Clusterschlüsseln einer geänderten Zeile erfasst.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = TRUE
AND CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'KEYS_ONLY'}};
```

2. Um die Stream-Einstellungen zu überprüfen, können Sie die folgende Anweisung verwenden.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
 system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
 'mytable';
```

Die Ausgabe der Anweisung sieht ähnlich aus wie folgt.

```
 keyspace_name | table_name | cdc  | custom_properties
---------------+------------+------
+-------------------------------------------------------------------------------------------
    mykeyspace |    mytable | True | {'capacity_mode':
 {'last_update_to_pay_per_request_timestamp': '1741385897045',
 'throughput_mode': 'PAY_PER_REQUEST'}, 'cdc_specification':
 {'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable/stream/2025-03-07T22:20:10.454', 'status': 'ENABLED',
 'view_type': 'KEYS_ONLY'}, 'encryption_specification': {'encryption_type':
 'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status': 'disabled'}}
```

CLI

Erstellen Sie einen CDC-Stream mit dem AWS CLI

1. Um einen Stream für eine vorhandene Tabelle zu erstellen, können Sie die folgende Syntax verwenden.

```
aws keyspaces update-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--cdc-specification status=ENABLED,viewType=NEW_AND_OLD_IMAGES
```

2. Die Ausgabe dieses Befehls zeigt die `create-table` Standardantwort und sieht ähnlich aus wie in diesem Beispiel.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable" }
```

Console

Aktivieren Sie einen CDC-Stream mit der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter. https://console.aws.amazon.com/keyspaces/home
2. Wählen Sie im Navigationsbereich Tabellen aus und wählen Sie dann eine Tabelle aus der Liste aus.
3. Wählen Sie den Tab Streams.
4. Wähle Bearbeiten, um einen Stream zu aktivieren.
5. Wähle Streams einschalten aus.
6. Wählen Sie den Ansichtstyp des Streams. Verfügbar sind die nachfolgend aufgeführten Optionen. Beachten Sie, dass Sie den Ansichtstyp eines Streams nicht mehr ändern können, nachdem er erstellt wurde.

   - Neue und alte Bilder — Amazon Keyspaces erfasst beide Versionen der Zeile, vor und nach der Änderung. Das ist die Standardeinstellung.
   - Neues Bild — Amazon Keyspaces erfasst nur die Version der Zeile nach der Änderung.
   - Altes Bild — Amazon Keyspaces erfasst nur die Version der Zeile vor der Änderung.

- Nur Primärschlüssel — Amazon Keyspaces erfasst nur die Partitions- und Clusterschlüsselspalten der Zeile, die geändert wurde.

7. Um den Vorgang abzuschließen, wählen Sie Änderungen speichern.

Deaktivieren Sie einen CDC-Stream in Amazon Keyspaces

Um einen CDC-Stream in einem Keyspace zu deaktivieren, können Sie die `ALTER TABLE` Anweisung in CQL, den `update-table` Befehl mit dem oder die Konsole verwenden. AWS CLI

Cassandra Query Language (CQL)

Deaktivieren Sie einen Stream (CDC-Stream) mit CQL

1. Um einen Stream zu deaktivieren, können Sie die folgende Anweisung verwenden.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = FALSE;
```

2. Um zu bestätigen, dass der Stream deaktiviert ist, können Sie die folgende Anweisung verwenden.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
 system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
 'mytable';
```

Die Ausgabe dieser Anweisung sieht ähnlich aus wie folgt.

```
 keyspace_name | table_name | cdc   | custom_properties
---------------+------------+-------
+-------------------------------------------------------------------------------------
   mykeyspace  |   mytable  | False | {'capacity_mode':
 {'last_update_to_pay_per_request_timestamp': '1741385668642',
 'throughput_mode': 'PAY_PER_REQUEST'}, 'encryption_specification':
 {'encryption_type': 'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status':
 'disabled'}}
```

CLI

Deaktivieren Sie einen Stream (CDC-Stream) mit dem AWS CLI

1. Um einen Stream zu deaktivieren, können Sie den folgenden Befehl verwenden.

```
aws keyspaces update-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--cdc-specification status=DISABLED
```

2. Die Ausgabe des Befehls sieht ähnlich aus wie in diesem Beispiel.

```
{
    "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
    "streamName": "my_stream"
}
```

Console

Deaktivieren Sie einen Stream (CDC-Stream) mit der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter. https://console.aws.amazon.com/keyspaces/home
2. Wählen Sie im Navigationsbereich Tabellen aus und wählen Sie dann eine Tabelle aus der Liste aus.
3. Wählen Sie den Tab Streams.
4. Wählen Sie Bearbeiten aus.
5. Entferne das Häkchen bei Streams einschalten.
6. Wählen Sie Änderungen speichern, um den Stream zu deaktivieren.

CDC-Streams in Amazon Keyspaces anzeigen

Um alle Streams im Schlüsselraum anzuzeigen oder aufzulisten, können Sie die Tabelle `system_schema_mcs.streams` im Systemschlüsselraum mithilfe einer Anweisung in CQL abfragen oder die `list-stream` Befehle `get-stream` und mit der, oder der AWS CLI Konsole verwenden.

Die erforderlichen Berechtigungen finden Sie unter the section called “Konfigurieren von - Berechtigungen”.

Cassandra Query Language (CQL)

CDC-Streams mit CQL anzeigen

- Um den CDC-Status Ihrer Tabelle zu überwachen, können Sie die folgende Anweisung verwenden.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name='my_keyspace' and table_name='my_table';
```

Die Ausgabe des Befehls sieht ähnlich aus wie folgt.

```
...
custom_properties
--------------------------------------------------------------------------------
{'cdc_specification':{'status': 'Enabled', 'view_type': 'NEW_IMAGE',
 'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table/stream/stream_label''}}
...
```

CLI

Sehen Sie sich CDC-Streams an mit dem AWS CLI

1. Dieses Beispiel zeigt, wie die Stream-Informationen für eine Tabelle angezeigt werden.

```
aws keyspaces get-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table'
```

Die Ausgabe des Befehls sieht so aus.

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    ... Other fields ...,
```

```
    "latestStreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table/stream/stream_label",
    "cdcSpecification": {
        "status": "ENABLED",
        "viewType": "NEW_AND_OLD_IMAGES"
    }
}
```

2. Sie können alle Streams in Ihrem Konto in einer bestimmten Liste auflisten AWS-Region. Der folgende Befehl ist ein Beispiel dafür.

```
aws keyspacesstreams list-streams --region us-east-1
```

Die Ausgabe des Befehls könnte ähnlich wie folgt aussehen.

```
{
    "Streams": [
        {
            "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t1/stream/2023-05-11T21:21:33.291",
            "StreamLabel": "2023-05-11T21:21:33.291",
            "KeyspaceName": "ks_1"
            "TableName": "t1",
        },
        {
            "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
            "StreamLabel": "2023-05-11T21:21:33.291",
            "KeyspaceName": "ks_1"Create a keyspace with the name catalog. Note
                                that streams are not supported in multi-Region
 keyspaces.
            "TableName": "t2",
        },
        {
            "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_2/table/t1/stream/2023-05-11T21:21:33.291",
            "StreamLabel": "2023-05-11T21:21:33.291",
            "KeyspaceName": "ks_3"
            "TableName": "t1",
        }
    ]
}
```

3. Sie können die CDC-Streams für einen bestimmten Schlüsselraum auch mithilfe der folgenden Parameter auflisten.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --region us-east-1
```

Die Ausgabe des Befehls sieht ähnlich aus wie folgt.

```
{
    "Streams": [
        {
            "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t1/stream/2023-05-11T21:21:33.291",
            "StreamLabel": "2023-05-11T21:21:33.291",
            "KeyspaceName": "ks_1"
            "TableName": "t1",
        },
        {
            "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
            "StreamLabel": "2023-05-11T21:21:33.291",
            "KeyspaceName": "ks_1"
            "TableName": "t2",
        }
    ]
}
```

4. Sie können die CDC-Streams für eine bestimmte Tabelle auch mithilfe der folgenden Parameter auflisten.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --table-name t2 --
region us-east-1
```

Die Ausgabe des Befehls sieht ähnlich aus wie folgt.

```
{
    "Streams": [
        {
            "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
            "StreamLabel": "2023-05-11T21:21:33.291",
            "KeyspaceName": "ks_1"
```

```
            "TableName": "t2",
        }
    ]
}
```

Console

Sehen Sie sich CDC-Streams in der Amazon Keyspaces-Konsole an

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter. https://console.aws.amazon.com/keyspaces/home
2. Wählen Sie im Navigationsbereich Tabellen aus und wählen Sie dann eine Tabelle aus der Liste aus.
3. Wählen Sie den Tab Streams, um die Stream-Details zu überprüfen.

Greifen Sie in Amazon Keyspaces auf Datensätze in CDC-Streams zu

Um auf die Datensätze in einem Stream zuzugreifen, verwenden Sie die Amazon Keyspaces Streams-API. Der folgende Abschnitt enthält Beispiele für den Zugriff auf Datensätze mithilfe von. AWS CLI

Die erforderlichen Berechtigungen finden Sie unter the section called “Konfigurieren von - Berechtigungen”.

Greifen Sie mit dem auf Datensätze in einem Stream zu AWS CLI

1. Sie können die Amazon Keyspaces Streams-API verwenden, um auf die Änderungsdatensätze des Streams zuzugreifen. Weitere Informationen finden Sie in der Amazon Keyspaces Streams-API-Referenz. Um die Shards innerhalb des Streams abzurufen, können Sie die `get-stream` API verwenden, wie im folgenden Beispiel gezeigt.

   ```
   aws keyspacesstreams get-stream \
   --stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
   mytable/stream/STREAM_LABEL'
   ```

   Es folgt ein Beispiel für die Ausgabe.

   ```
   {
   ```

```
   "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable/stream/2023-05-11T21:21:33.291",
   "StreamStatus": "ENABLED",
   "StreamViewType": "NEW_AND_OLD_IMAGES",
   "CreationRequestDateTime": "<CREATION_TIME>",
   "KeyspaceName": "mykeyspace",
   "TableName": "mytable",
   "StreamLabel": "2023-05-11T21:21:33.291",
    "Shards": [
        {
            "SequenceNumberRange": {
                "EndingSequenceNumber": "<END_SEQUENCE_NUMBER>",
                "StartingSequenceNumber": "<START_SEQUENCE_NUMBER>"
            },
            "ShardId": "<SHARD_ID>"
        },
    ]
}
```

2. Um Datensätze aus dem Stream abzurufen, benötigen Sie zunächst einen Iterator, der Ihnen den Ausgangspunkt für den Zugriff auf Datensätze bietet. Dazu können Sie die Shards innerhalb des CDC-Streams verwenden, die im vorherigen Schritt von der API zurückgegeben wurden. Um den Iterator zu sammeln, können Sie die API verwenden. `get-shard-iterator` Für dieses Beispiel verwenden Sie einen Iterator vom Typ`TRIM_HORIZON`, der vom letzten getrimmten Punkt (oder Anfang) des Shards abruft.

```
aws keyspacesstreams get-shard-iterator \
--stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable/stream/STREAM_LABEL' \
--shard-id 'SHARD_ID' \
--shard-iterator-type 'TRIM_HORIZON'
```

Die Ausgabe des Befehls sieht wie im folgenden Beispiel aus.

```
{
    "ShardIterator": "<SHARD_ITERATOR>"
}
```

3. Um die CDC-Datensätze mithilfe der `get-records` API abzurufen, können Sie den im letzten Schritt zurückgegebenen Iterator verwenden. Der folgende Befehl ist ein Beispiel dafür.

```
aws keyspacesstreams get-records \
--shard-iterator 'SHARD_ITERATOR' \
--limit 100
```

Es folgt ein Beispiel für die Ausgabe.

```
{
    "changeRecords": [...],
    "nextShardIterator": "<NEXT_SHARD_ITERATOR>",
    "iteratorDescription": {
        "iteratorPosition": "BEHIND_TIP"
    }
}
```

Optimieren Sie die Abfragefrequenz anhand der Iteratorposition

Die `GetRecords` Antwort enthält ein `iteratorDescription` Feld, das die aktuelle Position Ihres Verbrauchers innerhalb des Shards angibt:

- `AT_TIP`— Derzeit sind keine weiteren Datensätze verfügbar. Erwägen Sie, Ihre Abfragefrequenz zu reduzieren, um die Kosten zu optimieren.
- `BEHIND_TIP`— Der Stream-Tip bewegt sich kontinuierlich weiter. Möglicherweise sind weitere Datensätze verfügbar. Fahren Sie mit Ihrer normalen Frequenz fort.

`BEHIND_TIP`Eine leere `changeRecords` Liste zeigt an, dass der Stream läuft, aber an dieser Position sind keine Kundendatensätze verfügbar. Fahren Sie normal mit der Umfrage fort.

Verwenden Sie die Kinesis Client Library (KCL), um Amazon Keyspaces-Streams zu verarbeiten

In diesem Thema wird beschrieben, wie Sie die Kinesis Client Library (KCL) verwenden, um Daten aus Amazon Keyspaces Change Data Capture (CDC) -Streams zu verarbeiten und zu verarbeiten.

Anstatt direkt mit der Amazon Keyspaces Streams API zu arbeiten, bietet die Arbeit mit der Kinesis Client Library (KCL) viele Vorteile, zum Beispiel:

- Integriertes Shard-Lineage-Tracking und Iterator-Handling.
- Automatischer Lastenausgleich zwischen Mitarbeitern.

- Fehlertoleranz und Wiederherstellung nach Ausfällen von Mitarbeitern.
- Checkpointing zur Verfolgung des Verarbeitungsfortschritts.
- Anpassung an Änderungen der Stream-Kapazität.
- Vereinfachtes verteiltes Rechnen für die Verarbeitung von CDC-Datensätzen.

Der folgende Abschnitt beschreibt, warum und wie die Kinesis Client Library (KCL) zur Verarbeitung von Streams verwendet wird, und enthält ein Beispiel für die Verarbeitung eines Amazon Keyspaces CDC-Streams mit der KCL.

Informationen zur Preisgestaltung finden Sie unter Preise für Amazon Keyspaces (für Apache Cassandra).

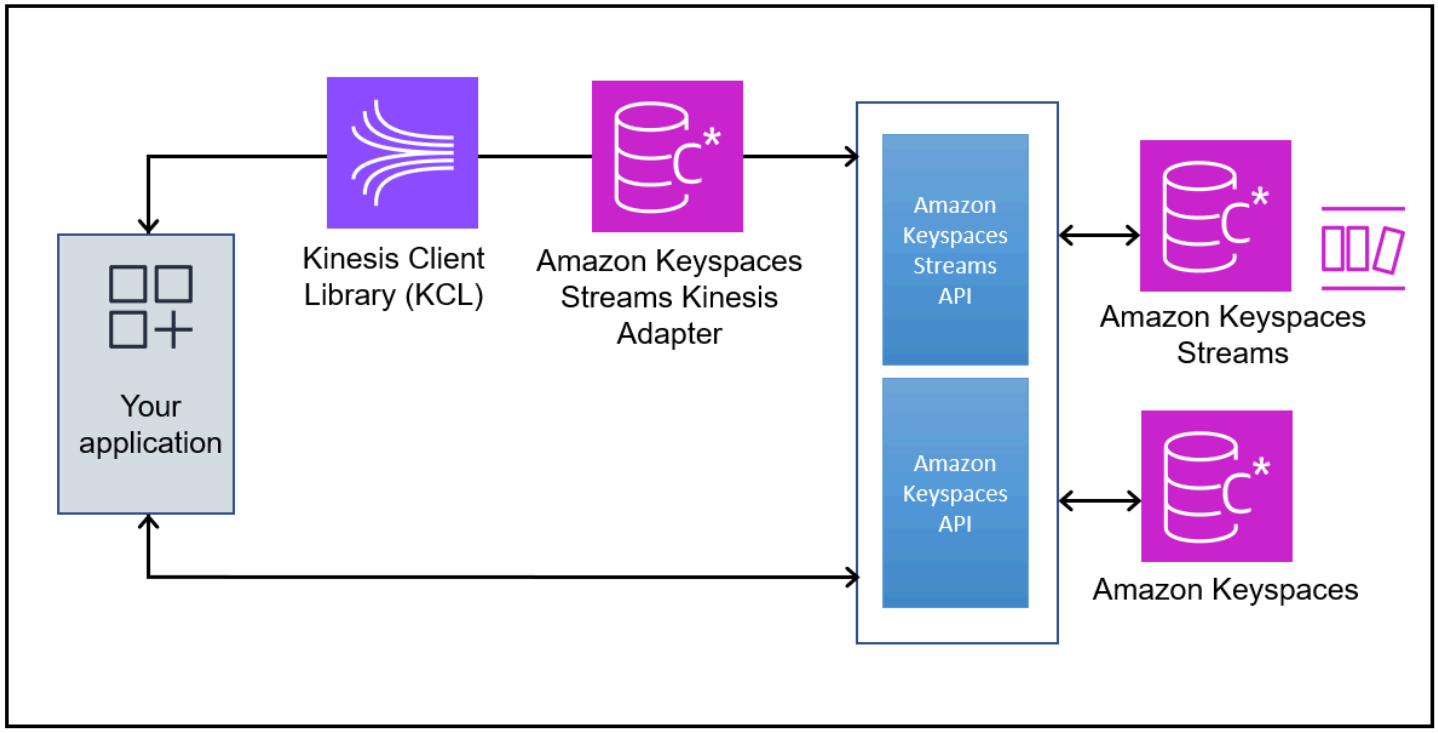
Was ist die Kinesis Client Library?

Die Kinesis Client Library (KCL) ist eine eigenständige Java-Softwarebibliothek, die entwickelt wurde, um den Prozess der Nutzung und Verarbeitung von Daten aus Streams zu vereinfachen. KCL erledigt viele der komplexen Aufgaben, die mit verteilter Datenverarbeitung verbunden sind, sodass Sie sich bei der Verarbeitung von Stream-Daten auf die Implementierung Ihrer Geschäftslogik konzentrieren können. KCL verwaltet Aktivitäten wie den Lastausgleich zwischen mehreren Workern, das Reagieren auf Worker-Ausfälle, Checkpoints verarbeiteter Datensätze und das Reagieren auf Änderungen der Anzahl der Shards im Stream.

Um CDC-Streams von Amazon Keyspaces zu verarbeiten, können Sie die in der KCL enthaltenen Entwurfsmuster für die Arbeit mit Stream-Shards und Stream-Datensätzen verwenden. Die KCL vereinfacht die Codierung durch Bereitstellen nützlicher Abstraktionen oberhalb der Low-Level-Kinesis-Data-Streams-API. Weitere Informationen zur KCL finden Sie unter Develop Consumer with KCL im Amazon Kinesis Data Streams Developer Guide.

Um Anwendungen mit der KCL zu schreiben, verwenden Sie den Amazon Keyspaces Streams Kinesis Adapter. Der Kinesis Adapter implementiert die Kinesis Data Streams-Schnittstelle, sodass Sie die KCL für die Nutzung und Verarbeitung von Datensätzen aus Amazon Keyspaces-Streams verwenden können. Anweisungen zur Einrichtung und Installation des Amazon Keyspaces Streams Kinesis-Adapters finden Sie im Repository. GitHub

Das folgende Diagramm zeigt, wie diese Bibliotheken miteinander interagieren.



KCL wird regelmäßig aktualisiert, um neuere Versionen der zugrunde liegenden Bibliotheken, Sicherheitsverbesserungen und Bugfixes zu integrieren. Wir empfehlen Ihnen, die neueste Version von KCL zu verwenden, um bekannte Probleme zu vermeiden und von den neuesten Verbesserungen zu profitieren. Die neueste KCL-Version finden Sie im KCL-Repository. GitHub

KCL-Konzepte

Bevor Sie eine Verbraucheranwendung mit KCL implementieren, sollten Sie die folgenden Konzepte verstehen:

KCL-Verbraucheranwendung

Eine KCL-Verbraucheranwendung ist ein Programm, das Daten aus einem Amazon Keyspaces-CDC-Stream verarbeitet. Das KCL fungiert als Vermittler zwischen Ihrem Anwendungscode für Verbraucher und dem Amazon Keyspaces CDC-Stream.

Arbeiter

Ein Worker ist eine Ausführungseinheit Ihrer KCL-Verbraucheranwendung, die Daten aus dem Amazon Keyspaces CDC-Stream verarbeitet. In Ihrer Anwendung können mehrere Worker ausgeführt werden, die auf mehrere Instances verteilt sind.

Prozessor aufnehmen

Ein Datensatzprozessor ist die Logik in Ihrer Anwendung, die Daten von einem Shard im Amazon Keyspaces CDC-Stream verarbeitet. Ein Datensatzprozessor wird von einem Worker für jeden Shard, den er verwaltet, instanziiert.

Leasing

Ein Leasing steht für die Verarbeitungsverantwortung für einen Shard. Arbeiter nutzen Leasingverträge, um zu koordinieren, welcher Arbeiter welchen Shard bearbeitet. KCL speichert Leasingdaten in einer Tabelle in Amazon DynamoDB.

Checkpoint

Ein Checkpoint ist eine Aufzeichnung der Position im Shard, bis zu der der Datensatzprozessor Datensätze erfolgreich verarbeitet hat. Checkpointing ermöglicht es Ihrer Anwendung, die Verarbeitung an der Stelle fortzusetzen, an der sie aufgehört hat, wenn ein Worker ausfällt.

Wenn der Amazon Keyspaces Kinesis-Adapter installiert ist, können Sie mit der Entwicklung über die KCL-Schnittstelle beginnen, wobei die API-Aufrufe nahtlos an den Amazon Keyspaces-Stream-Endpunkt weitergeleitet werden. Eine Liste der verfügbaren Endpunkte finden Sie unter. the section called “Greifen Sie auf CDC-Stream-Endpunkte zu”

Beim Start der Anwendung wird die KCL aufgerufen, einen Worker zu instanziieren. Sie müssen dem Worker Konfigurationsinformationen für die Anwendung zur Verfügung stellen, z. B. den Stream-Deskriptor und die AWS Anmeldeinformationen sowie den Namen einer Record-Processor-Klasse, die Sie angeben. Da der Code im Datensatzprozessor ausgeführt wird, erledigt der Worker die folgenden Aufgaben:

- Stellt eine Verbindung mit dem Stream her
- Listet die Shards innerhalb des Streams auf
- Koordiniert Shard-Zuordnungen mit anderen Auftragnehmern (wenn vorhanden)
- Instanziiert einen Datensatzverarbeiter für jeden Shard, der verwaltet wird
- Ruft Datensätze aus dem Stream per Pull ab
- Überträgt per Push Datensätze an den entsprechenden Datensatzverarbeiter
- Verwendet Checkpoints für verarbeitete Datensätze
- Gleicht Shard-Auftragnehmer-Zuordnungen aus, wenn die Auftragnehmer-Instance Änderungen zählt

- Gleicht Shard-Worker-Zuordnungen aus, wenn Shards aufgeteilt werden

Implementieren Sie eine KCL-Verbraucheranwendung für Amazon Keyspaces CDC-Streams

Dieses Thema enthält eine schrittweise Anleitung zur Implementierung einer KCL-Verbraucheranwendung zur Verarbeitung von Amazon Keyspaces-CDC-Streams.

1. Voraussetzungen: Bevor Sie beginnen, stellen Sie sicher, dass Sie über Folgendes verfügen:
   - Eine Amazon Keyspaces-Tabelle mit einem CDC-Stream
   - Erforderliche IAM-Berechtigungen für den IAM-Prinzipal, um auf den Amazon Keyspaces-CDC-Stream zuzugreifen, DynamoDB-Tabellen für die KCL-Stream-Verarbeitung zu erstellen und darauf zuzugreifen, sowie Berechtigungen zum Veröffentlichen von Metriken. CloudWatch Weitere Informationen und ein Beispiel für eine Richtlinie finden Sie unter. the section called “Berechtigungen für KCL”
   - Stellen Sie sicher, dass in Ihrer lokalen Konfiguration gültige AWS Anmeldeinformationen eingerichtet sind. Weitere Informationen finden Sie unter the section called “Verwalten von Zugriffsschlüsseln”.
   - Java Development Kit (JDK) 8 oder höher
   - Die Anforderungen sind in der Readme-Datei auf Github aufgeführt.
2. In diesem Schritt fügen Sie Ihrem Projekt die KCL-Abhängigkeit hinzu. Fügen Sie für Maven Folgendes zu Ihrer pom.xml hinzu:

```
<dependencies>
        <dependency>
            <groupId>software.amazon.kinesis</groupId>
            <artifactId>amazon-kinesis-client</artifactId>
            <version>3.1.0</version>
        </dependency>
        <dependency>
            <groupId>software.amazon.keyspaces</groupId>
            <artifactId>keyspaces-streams-kinesis-adapter</artifactId>
            <version>1.0.0</version>
        </dependency>
    </dependencies>
```

> **Note**
>
> Suchen Sie im KCL-Repository immer nach der neuesten Version von GitHub KCL.

3. Erstellen Sie eine Factory-Klasse, die Record-Processor-Instanzen erzeugt:

```
import software.amazon.awssdk.services.keyspacesstreams.model.Record;
import
 software.amazon.keyspaces.streamsadapter.adapter.KeyspacesStreamsClientRecord;
import
 software.amazon.keyspaces.streamsadapter.model.KeyspacesStreamsProcessRecordsInput;
import
 software.amazon.keyspaces.streamsadapter.processor.KeyspacesStreamsShardRecordProcessor;
import software.amazon.kinesis.lifecycle.events.InitializationInput;
import software.amazon.kinesis.lifecycle.events.LeaseLostInput;
import software.amazon.kinesis.lifecycle.events.ShardEndedInput;
import software.amazon.kinesis.lifecycle.events.ShutdownRequestedInput;
import software.amazon.kinesis.processor.RecordProcessorCheckpointer;

public class RecordProcessor implements KeyspacesStreamsShardRecordProcessor {
    private String shardId;

    @Override
    public void initialize(InitializationInput initializationInput) {
        this.shardId = initializationInput.shardId();
        System.out.println("Initializing record processor for shard: " + shardId);
    }

    @Override
    public void processRecords(KeyspacesStreamsProcessRecordsInput
 processRecordsInput) {
        try {
            for (KeyspacesStreamsClientRecord record :
 processRecordsInput.records()) {
                Record keyspacesRecord = record.getRecord();
                System.out.println("Received record: " + keyspacesRecord);
            }

            if (!processRecordsInput.records().isEmpty()) {
                RecordProcessorCheckpointer checkpointer =
 processRecordsInput.checkpointer();
                try {
```

```
                    checkpointer.checkpoint();
                    System.out.println("Checkpoint successful for shard: " +
shardId);
                } catch (Exception e) {
                    System.out.println("Error while checkpointing for shard: " +
shardId + " " + e);
                }
            }
        } catch (Exception e) {
            System.out.println("Error processing records for shard: " + shardId + "
" + e);
        }
    }

    @Override
    public void leaseLost(LeaseLostInput leaseLostInput) {
        System.out.println("Lease lost for shard: " + shardId);
    }

    @Override
    public void shardEnded(ShardEndedInput shardEndedInput) {
        System.out.println("Shard ended: " + shardId);
        try {
            // This is required. Checkpoint at the end of the shard
            shardEndedInput.checkpointer().checkpoint();
            System.out.println("Final checkpoint successful for shard: " +
shardId);
        } catch (Exception e) {
            System.out.println("Error while final checkpointing for shard: " +
shardId + " " + e);
            throw new RuntimeException("Error while final checkpointing", e);
        }
    }

    @Override
    public void shutdownRequested(ShutdownRequestedInput shutdownRequestedInput) {
        System.out.println("Shutdown requested for shard " + shardId);
        try {
            shutdownRequestedInput.checkpointer().checkpoint();
        } catch (Exception e) {
            System.out.println("Error while checkpointing on shutdown for shard: "
+ shardId + " " + e);
        }
    }
```

```
}
```

4. Erstellen Sie eine Record Factory, wie im folgenden Beispiel gezeigt.

```
import software.amazon.kinesis.processor.ShardRecordProcessor;
import software.amazon.kinesis.processor.ShardRecordProcessorFactory;

import java.util.Queue;
import java.util.concurrent.ConcurrentLinkedQueue;

public class RecordProcessorFactory implements ShardRecordProcessorFactory {
    private final Queue<RecordProcessor> processors = new
 ConcurrentLinkedQueue<>();

    @Override
    public ShardRecordProcessor shardRecordProcessor() {
        System.out.println("Creating new RecordProcessor");
        RecordProcessor processor = new RecordProcessor();
        processors.add(processor);
        return processor;
    }
}
```

5. In diesem Schritt erstellen Sie die Basisklasse zur Konfiguration von KCLv3 und des Amazon Keyspaces-Adapters.

```
import com.example.KCLExample.utils.RecordProcessorFactory;
import
 software.amazon.keyspaces.streamsadapter.AmazonKeyspacesStreamsAdapterClient;
import software.amazon.keyspaces.streamsadapter.StreamsSchedulerFactory;
import java.util.Arrays;
import java.util.List;
import java.util.concurrent.ExecutionException;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatch.CloudWatchAsyncClient;
import software.amazon.awssdk.services.dynamodb.DynamoDbAsyncClient;
import software.amazon.awssdk.services.dynamodb.model.DeleteTableRequest;
import software.amazon.awssdk.services.dynamodb.model.DeleteTableResponse;
import software.amazon.awssdk.services.keyspacesstreams.KeyspacesStreamsClient;
import software.amazon.awssdk.services.kinesis.KinesisAsyncClient;
import software.amazon.kinesis.common.ConfigsBuilder;
import software.amazon.kinesis.common.InitialPositionInStream;
```

```
import software.amazon.kinesis.common.InitialPositionInStreamExtended;
import software.amazon.kinesis.coordinator.CoordinatorConfig;
import software.amazon.kinesis.coordinator.Scheduler;
import software.amazon.kinesis.leases.LeaseManagementConfig;
import software.amazon.kinesis.processor.ProcessorConfig;
import software.amazon.kinesis.processor.StreamTracker;
import software.amazon.kinesis.retrieval.polling.PollingConfig;

public class KCLTestBase {

    protected KeyspacesStreamsClient streamsClient;
    protected KinesisAsyncClient adapterClient;
    protected DynamoDbAsyncClient dynamoDbAsyncClient;
    protected CloudWatchAsyncClient cloudWatchClient;
    protected Region region;
    protected RecordProcessorFactory recordProcessorFactory;
    protected Scheduler scheduler;
    protected Thread schedulerThread;

    public void baseSetUp() {
        recordProcessorFactory = new RecordProcessorFactory();
        setupKCLBase();
    }

    protected void setupKCLBase() {
        region = Region.US_EAST_1;

        streamsClient = KeyspacesStreamsClient.builder()
                .region(region)
                .build();
        adapterClient = new AmazonKeyspacesStreamsAdapterClient(
                streamsClient,
                region);
        dynamoDbAsyncClient = DynamoDbAsyncClient.builder()
                .region(region)
                .build();
        cloudWatchClient = CloudWatchAsyncClient.builder()
                .region(region)
                .build();
    }

    protected void startScheduler(Scheduler scheduler) {
        this.scheduler = scheduler;
        schedulerThread = new Thread(() -> scheduler.run());
```

```
        schedulerThread.start();
    }

    protected void shutdownScheduler() {
        if (scheduler != null) {
            scheduler.shutdown();
            try {
                schedulerThread.join(30000);
            } catch (InterruptedException e) {
                System.out.println("Error while shutting down scheduler " + e);
            }
        }
    }

    protected Scheduler createScheduler(String streamArn, String leaseTableName) {
        String workerId = "worker-" + System.currentTimeMillis();

        // Create ConfigsBuilder
        ConfigsBuilder configsBuilder = createConfigsBuilder(streamArn, workerId,
leaseTableName);

        // Configure retrieval config for polling
        PollingConfig pollingConfig = new PollingConfig(streamArn, adapterClient);

        // Create the Scheduler
        return StreamsSchedulerFactory.createScheduler(
                configsBuilder.checkpointConfig(),
                configsBuilder.coordinatorConfig(),
                configsBuilder.leaseManagementConfig(),
                configsBuilder.lifecycleConfig(),
                configsBuilder.metricsConfig(),
                configsBuilder.processorConfig(),

configsBuilder.retrievalConfig().retrievalSpecificConfig(pollingConfig),
                streamsClient,
                region
        );
    }

    private ConfigsBuilder createConfigsBuilder(String streamArn, String workerId,
String leaseTableName) {
        ConfigsBuilder configsBuilder = new ConfigsBuilder(
                streamArn,
                leaseTableName,
```

```
                adapterClient,
                dynamoDbAsyncClient,
                cloudWatchClient,
                workerId,
                recordProcessorFactory);

        configureCoordinator(configsBuilder.coordinatorConfig());
        configureLeaseManagement(configsBuilder.leaseManagementConfig());
        configureProcessor(configsBuilder.processorConfig());
        configureStreamTracker(configsBuilder, streamArn);

        return configsBuilder;
    }

    private void configureCoordinator(CoordinatorConfig config) {
        config.skipShardSyncAtWorkerInitializationIfLeasesExist(true)
                .parentShardPollIntervalMillis(1000)
                .shardConsumerDispatchPollIntervalMillis(500);
    }

    private void configureLeaseManagement(LeaseManagementConfig config) {
        config.shardSyncIntervalMillis(0)
                .leasesRecoveryAuditorInconsistencyConfidenceThreshold(0)
                .leasesRecoveryAuditorExecutionFrequencyMillis(5000)
                .leaseAssignmentIntervalMillis(1000L);
    }

    private void configureProcessor(ProcessorConfig config) {
        config.callProcessRecordsEvenForEmptyRecordList(true);
    }

    private void configureStreamTracker(ConfigsBuilder configsBuilder, String
streamArn) {
        StreamTracker streamTracker =
StreamsSchedulerFactory.createSingleStreamTracker(
                streamArn,

InitialPositionInStreamExtended.newInitialPosition(InitialPositionInStream.TRIM_HORIZON)
        );
        configsBuilder.streamTracker(streamTracker);
    }

    public void deleteAllDdbTables(String baseTableName) {
        List<String> tablesToDelete = Arrays.asList(
```

```
                baseTableName,
                baseTableName + "-CoordinatorState",
                baseTableName + "-WorkerMetricStats"
        );

        for (String tableName : tablesToDelete) {
            deleteTable(tableName);
        }
    }

    private void deleteTable(String tableName) {
        DeleteTableRequest deleteTableRequest = DeleteTableRequest.builder()
                .tableName(tableName)
                .build();

        try {
            DeleteTableResponse response =
 dynamoDbAsyncClient.deleteTable(deleteTableRequest).get();
            System.out.println("Table deletion response " + response);
        } catch (InterruptedException | ExecutionException e) {
            System.out.println("Error deleting table: " + tableName + " " + e);
        }
    }
}
```

6. In diesem Schritt implementieren Sie die Record Processor-Klasse für Ihre Anwendung, um mit der Verarbeitung von Änderungsereignissen zu beginnen.

```
 import software.amazon.kinesis.coordinator.Scheduler;

public class KCLTest {

    private static final int APP_RUNTIME_SECONDS = 1800;
    private static final int SLEEP_INTERNAL_MS = 60*1000;

    public static void main(String[] args) {
        KCLTestBase kclTestBase;

        kclTestBase = new KCLTestBase();
        kclTestBase.baseSetUp();

        // Create and start scheduler
        String leaseTableName = generateUniqueApplicationName();
```

```
        // Update below to your Stream ARN
        String streamArn = "arn:aws:cassandra:us-east-1:759151643516:/keyspace/
cdc_sample_test/table/test_kcl_bool/stream/2025-07-01T15:52:57.529";
        Scheduler scheduler = kclTestBase.createScheduler(streamArn,
 leaseTableName);
        kclTestBase.startScheduler(scheduler);

        // Wait for specified time before shutting down - KCL applications are
 designed to run forever, however in this
        // example we will shut it down after APP_RUNTIME_SECONDS
        long startTime = System.currentTimeMillis();
        long endTime = startTime + (APP_RUNTIME_SECONDS * 1000);
        while (System.currentTimeMillis() < endTime) {
            try {
                // Print and sleep every minute
                Thread.sleep(SLEEP_INTERNAL_MS);
                System.out.println("Application is running");
            } catch (InterruptedException e) {
                System.out.println("Interrupted while waiting for records");
                Thread.currentThread().interrupt();
                break;
            }
        }

        // Stop the scheduler
        kclTestBase.shutdownScheduler();
        kclTestBase.deleteAllDdbTables(leaseTableName);
    }

    public static String generateUniqueApplicationName() {
        String timestamp = String.valueOf(System.currentTimeMillis());
        String randomString = java.util.UUID.randomUUID().toString().substring(0,
 8);
        return String.format("KCL-App-%s-%s", timestamp, randomString);
    }
}
```

Bewährte Methoden

Folgen Sie diesen bewährten Methoden, wenn Sie KCL mit Amazon Keyspaces CDC-Streams verwenden:

Fehlerbehandlung

Implementieren Sie eine robuste Fehlerbehandlung in Ihrem Record Processor, um Ausnahmen ordnungsgemäß behandeln zu können. Erwägen Sie die Implementierung einer Wiederholungslogik für vorübergehende Fehler.

Häufigkeit der Checkpointing

Gleichen Sie die Häufigkeit der Checkpoints aus, um doppelte Bearbeitungen zu minimieren und gleichzeitig eine angemessene Fortschrittsverfolgung zu gewährleisten. Zu häufiges Checkpointing kann sich auf die Leistung auswirken, während zu seltenes Checkpointing zu mehr Nachbearbeitung führen kann, wenn ein Mitarbeiter ausfällt.

Skalierung von Workern

Skalieren Sie die Anzahl der Worker basierend auf der Anzahl der Shards in Ihrem CDC-Stream. Ein guter Ausgangspunkt ist es, einen Worker pro Shard zu haben, aber möglicherweise müssen Sie dies an Ihre Verarbeitungsanforderungen anpassen.

Überwachung

Verwenden Sie die von KCL bereitgestellten CloudWatch Kennzahlen, um den Zustand und die Leistung Ihrer Verbraucheranwendung zu überwachen. Zu den wichtigsten Kennzahlen gehören die Verarbeitungslatenz, das Checkpoint-Alter und die Anzahl der Leasings.

Testen

Testen Sie Ihre Consumer-Anwendung gründlich, einschließlich Szenarien wie Mitarbeiterausfälle, Stream-Resharding und unterschiedliche Lastbedingungen.

Verwenden von KCL mit Sprachen, die nicht Java sind

Obwohl KCL in erster Linie eine Java-Bibliothek ist, können Sie sie mit anderen Programmiersprachen über die verwenden. MultiLangDaemon Das MultiLangDaemon ist ein Java-based Daemon, der die Interaktion zwischen Ihrem Nicht-Java-Recordprozessor und der KCL verwaltet.

KCL bietet Unterstützung für die folgenden Sprachen:

- Python
- Ruby
- Node.js

- .NET

Weitere Informationen zur Verwendung von KCL mit Nicht-Java-Sprachen finden Sie in der KCL-Dokumentation. MultiLangDaemon

Fehlerbehebung

Dieser Abschnitt enthält Lösungen für häufig auftretende Probleme, die bei der Verwendung von KCL mit Amazon Keyspaces CDC-Streams auftreten können.

Langsame Verarbeitung

Wenn Ihre Verbraucheranwendung Datensätze langsam verarbeitet, sollten Sie Folgendes berücksichtigen:

- Erhöhung der Anzahl der Worker-Instanzen
- Optimieren Sie Ihre Logik zur Datensatzverarbeitung
- Überprüfung auf Engpässe in nachgelagerten Systemen

Doppelte Verarbeitung

Wenn Sie feststellen, dass Datensätze doppelt verarbeitet werden, überprüfen Sie Ihre Checkpointing-Logik. Stellen Sie sicher, dass Sie nach der erfolgreichen Verarbeitung von Datensätzen Checkpoints durchführen.

Ausfälle von Mitarbeitern

Wenn Mitarbeiter häufig versagen, überprüfen Sie Folgendes:

- Ressourcenbeschränkungen (CPU, Speicher)
- Probleme mit der Netzwerkkonnektivität
- Probleme mit den Berechtigungen

Probleme mit der Leasing-Tabelle

Wenn Sie Probleme mit der KCL-Leasetabelle haben:

- Vergewissern Sie sich, dass Ihre Anwendung über die entsprechenden Berechtigungen für den Zugriff auf die Amazon Keyspaces-Tabelle verfügt
- Stellen Sie sicher, dass die Tabelle über einen ausreichenden bereitgestellten Durchsatz verfügt

Arbeiten mit Partitionierern in Amazon Keyspaces

In Apache Cassandra steuern Partitionierer, auf welchen Knoten Daten im Cluster gespeichert werden. Partitionierer erstellen ein numerisches Token unter Verwendung eines Hashwerts des Partitionsschlüssels. Cassandra verwendet dieses Token, um Daten auf Knoten zu verteilen. Kunden können diese Token auch in `SELECT` Operationen und `WHERE` Klauseln verwenden, um Lese- und Schreibvorgänge zu optimieren. Beispielsweise können Clients effizient parallel Abfragen an großen Tabellen durchführen, indem sie in jedem parallelen Job unterschiedliche Tokenbereiche angeben, die abgefragt werden sollen.

Amazon Keyspaces bietet drei verschiedene Partitionierer.

Murmur3 Partitioner (Standard)

Apache Cassandra-compatible `Murmur3Partitioner`. Das `Murmur3Partitioner` ist die Standard-Cassandra-Partitionierung in Amazon Keyspaces und in Cassandra 1.2 und späteren Versionen.

RandomPartitioner

Cassandra-compatible `RandomPartitioner`Apache. Das `RandomPartitioner` ist die Standard-Cassandra-Partitionierung für Versionen vor Cassandra 1.2.

Keyspaces Standardpartitionierung

Der `DefaultPartitioner` gibt dieselben `token` Funktionsergebnisse zurück wie der. `RandomPartitioner`

Die Partitionierungseinstellung wird pro Region auf Kontoebene angewendet. Wenn Sie beispielsweise den Partitionierer in USA Ost (Nord-Virginia) ändern, wird die Änderung auf alle Tabellen desselben Kontos in dieser Region angewendet. Sie können Ihre Partitionierung jederzeit sicher ändern. Beachten Sie, dass es ungefähr 10 Minuten dauert, bis die Konfigurationsänderung abgeschlossen ist. Sie müssen Ihre Amazon Keyspaces-Daten nicht neu laden, wenn Sie die Partitionierungseinstellung ändern. Clients verwenden automatisch die neue Partitionierungseinstellung, wenn sie das nächste Mal eine Verbindung herstellen.

So ändern Sie den Partitionierer in Amazon Keyspaces

Sie können den Partitionierer mithilfe der AWS-Managementkonsole oder Cassandra Query Language (CQL) ändern.

AWS-Managementkonsole

So ändern Sie den Partitionierer mithilfe der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Configuration (Konfiguration) aus.
3. Gehen Sie auf der Konfigurationsseite zu Partitionierung bearbeiten.
4. Wählen Sie den Partitionierer aus, der mit Ihrer Version von Cassandra kompatibel ist. Es dauert ungefähr 10 Minuten, bis die Änderung des Partitionierers angewendet wird.

> **Note**
>
> Nachdem die Konfigurationsänderung abgeschlossen ist, müssen Sie die Verbindung zu Amazon Keyspaces trennen und erneut verbinden, um Anfragen zur Verwendung der neuen Partitionierung zu stellen.

Cassandra Query Language (CQL)

1. Um zu sehen, welcher Partitionierer für das Konto konfiguriert ist, können Sie die folgende Abfrage verwenden.

```
SELECT partitioner from system.local;
```

Wenn der Partitionierer nicht geändert wurde, hat die Abfrage die folgende Ausgabe.

```
partitioner
-------------------------------------------
com.amazonaws.cassandra.DefaultPartitioner
```

2. Um die Partitionierung auf die Partitionierung zu aktualisieren, können Sie die `Murmur3` folgende Anweisung verwenden.

```
UPDATE system.local set
 partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

3. Beachten Sie, dass es ungefähr 10 Minuten dauert, bis diese Konfigurationsänderung abgeschlossen ist. Um zu bestätigen, dass der Partitionierer eingerichtet wurde, können

Sie die `SELECT` Abfrage erneut ausführen. Beachten Sie, dass die Antwort aufgrund der eventuellen Lesekonsistenz möglicherweise noch nicht die Ergebnisse der kürzlich abgeschlossenen Änderung der Partitionierung widerspiegelt. Wenn Sie den `SELECT` Vorgang nach kurzer Zeit erneut wiederholen, sollte die Antwort die neuesten Daten zurückgeben.

```
SELECT partitioner from system.local;
```

> **Note**
>
> Sie müssen die Verbindung zu Amazon Keyspaces trennen und erneut verbinden, damit Anfragen den neuen Partitionierer verwenden.

Client-side Zeitstempel in Amazon Keyspaces

In Amazon Keyspaces sind clientseitige Zeitstempel Zeitstempel, die für Cassandra-compatible jede Zelle in Ihrer Tabelle beibehalten werden. Sie können clientseitige Zeitstempel zur Konfliktlösung verwenden, indem Sie Ihre Client-Anwendungen die Reihenfolge der Schreibvorgänge bestimmen lassen. Wenn beispielsweise Clients einer global verteilten Anwendung dieselben Daten aktualisieren, wird durch die clientseitigen Zeitstempel die Reihenfolge beibehalten, in der die Aktualisierungen auf den Clients vorgenommen wurden. Amazon Keyspaces verwendet diese Zeitstempel, um die Schreibvorgänge zu verarbeiten.

Die clientseitigen Zeitstempel von Amazon Keyspace werden vollständig verwaltet. Sie müssen sich nicht um grundlegende Systemeinstellungen wie Säuberungs- und Verdichtungsstrategien kümmern.

Wenn Sie Daten löschen, werden die Zeilen mit einem Grabstein zum Löschen markiert. Amazon Keyspaces entfernt veraltete Daten automatisch (in der Regel innerhalb von 10 Tagen), ohne die Leistung oder Verfügbarkeit Ihrer Anwendung zu beeinträchtigen. Als veraltet eingestufte Daten sind nicht für DML-Anweisungen (Data Manipulation Language) verfügbar. Wenn Sie weiterhin Lese- und Schreibvorgänge in Zeilen ausführen, die als veraltet eingestufte Daten enthalten, werden diese Daten weiterhin auf Speicher, Lesekapazitätseinheiten (RCUs) und Schreibkapazitätseinheiten (WCUs) angerechnet, bis sie aus dem Speicher gelöscht werden.

Nachdem clientseitige Zeitstempel für eine Tabelle aktiviert wurden, können Sie in Ihrer DML-CQL-Abfrage (Data Manipulation Language) einen Zeitstempel mit der `USING TIMESTAMP` Klausel angeben. Weitere Informationen finden Sie unter the section called “Verwenden Sie clientseitige

Zeitstempel in Abfragen". Wenn Sie in Ihrer CQL-Abfrage keinen Zeitstempel angeben, verwendet Amazon Keyspaces den von Ihrem Client-Treiber übergebenen Zeitstempel. Wenn der Client-Treiber keine Zeitstempel bereitstellt, weist Amazon Keyspaces automatisch einen Zeitstempel auf Zellenebene zu, da Zeitstempel dies nicht sein können. `NULL` Um Zeitstempel abzufragen, können Sie die Funktion in Ihrer DML-Anweisung verwenden. `WRITETIME`

Amazon Keyspaces berechnet keine zusätzlichen Gebühren für die Aktivierung von clientseitigen Zeitstempeln. Bei clientseitigen Zeitstempeln speichern und schreiben Sie jedoch zusätzliche Daten für jeden Wert in Ihrer Zeile. Dies kann zu zusätzlicher Speichernutzung und in einigen Fällen zu zusätzlicher Durchsatznutzung führen. Weitere Informationen zu den Preisen von Amazon Keyspaces finden Sie unter Preise für Amazon Keyspaces (für Apache Cassandra).

Wenn clientseitige Zeitstempel in Amazon Keyspaces aktiviert sind, werden neben Ihren Zeilendaten zusätzliche Metadaten gespeichert. Der Mehraufwand pro Zeile hängt von Ihren Spaltentypen, davon ab, ob die Zeile TTL verwendet, und (bei Tabellen mit mehreren Regionen) von der Anzahl der replizierenden Regionen. Der Overhead kann von einigen Byte für Zeilen mit einfachen skalaren Spalten bis hin zu mehreren zehn Byte oder mehr für Zeilen mit nicht eingefrorenen Sammlungen oder Zählern für mehrere Regionen reichen. Diese Metadaten werden sowohl auf Ihre Speicherkosten als auch auf Ihr Kontingent für die Zeilengröße von 1 MB angerechnet.

Um die Gesamtauswirkung auf Speicher und Durchsatz zu ermitteln, sollten Sie die Anzahl der Spalten in Ihrer Tabelle, die verwendeten Datentypen und die Anzahl der Sammlungselemente in jeder Zeile berücksichtigen. Beispielsweise haben Zeilen mit vielen nicht eingefrorenen Sammlungsspalten, die eine große Anzahl von Elementen enthalten, einen höheren Overhead als Zeilen mit nur skalaren Spalten. Die zusätzlichen Metadaten wirken sich auch auf die Anzahl der pro Schreibvorgang verbrauchten Schreibkapazitätseinheiten (WCUs) aus. Weitere Informationen zur Schätzung des Overheads für Ihr spezielles Schema finden Sie unterthe section called "Schätzen Sie die Zeilengröße". Weitere Informationen zur Berechnung der Lese- und Schreibkapazität finden Sie unterthe section called " read/write Kapazitätsmodi konfigurieren".

Nachdem die clientseitigen Zeitstempel für eine Tabelle aktiviert wurden, können Sie sie nicht mehr deaktivieren.

Weitere Informationen zur Verwendung von clientseitigen Zeitstempeln in Abfragen finden Sie unter. the section called "Verwenden Sie clientseitige Zeitstempel in Abfragen"

Themen

- So lassen sich clientseitige Zeitstempel von Amazon Keyspace integrieren AWS service

- Erstellen Sie eine neue Tabelle mit clientseitigen Zeitstempeln in Amazon Keyspaces
- Client-seitige Zeitstempel für eine Tabelle in Amazon Keyspaces konfigurieren
- Verwenden Sie clientseitige Zeitstempel in Abfragen in Amazon Keyspaces

So lassen sich clientseitige Zeitstempel von Amazon Keyspace integrieren AWS service

Die folgende clientseitige Zeitstempelmetrik ist in Amazon verfügbar, um eine kontinuierliche Überwachung CloudWatch zu ermöglichen.

- `SystemReconciliationDeletes`— Die Anzahl der Löschvorgänge, die erforderlich sind, um veraltete Daten zu entfernen.

Weitere Informationen zur Überwachung von CloudWatch Metriken finden Sie unter. the section called “Überwachung mit CloudWatch”

Wenn Sie verwenden CloudFormation, können Sie beim Erstellen einer Amazon Keyspaces-Tabelle clientseitige Zeitstempel aktivieren. Weitere Informationen finden Sie im AWS CloudFormation - Benutzerhandbuch.

Erstellen Sie eine neue Tabelle mit clientseitigen Zeitstempeln in Amazon Keyspaces

Folgen Sie diesen Beispielen, um eine neue Amazon Keyspaces-Tabelle mit aktivierten clientseitigen Zeitstempeln zu erstellen. Verwenden Sie dazu Amazon Keyspaces AWS-Managementkonsole, Cassandra Query Language (CQL) oder AWS Command Line Interface

Console

Erstellen Sie eine neue Tabelle mit clientseitigen Zeitstempeln (Konsole)

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) und anschließend Create table (Tabelle erstellen) aus.
3. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die neue Tabelle ein.

4. Erstellen Sie im Abschnitt Schema das Schema für Ihre Tabelle.
5. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.
6. Fahren Sie mit den Client-side Zeitstempeln fort.

   Wählen Sie Clientseitige Zeitstempel aktivieren, um clientseitige Zeitstempel für die Tabelle zu aktivieren.
7. Wählen Sie Create table (Tabelle erstellen) aus. Ihre Tabelle wird mit aktivierten clientseitigen Zeitstempeln erstellt.

Cassandra Query Language (CQL)

Erstellen Sie eine neue Tabelle mit CQL

1. Um eine neue Tabelle mit aktivierten clientseitigen Zeitstempeln mithilfe von CQL zu erstellen, können Sie das folgende Beispiel verwenden.

```
CREATE TABLE my_keyspace.my_table (
   userid uuid,
   time timeuuid,
   subject text,
   body text,
   user inet,
   PRIMARY KEY (userid, time)
) WITH CUSTOM_PROPERTIES = {'client_side_timestamps': {'status': 'enabled'}};
```

2. Um die clientseitigen Zeitstempeleinstellungen für die neue Tabelle zu bestätigen, verwenden Sie eine SELECT Anweisung, um sie zu überprüfen, wie im `custom_properties` folgenden Beispiel gezeigt.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

   Die Ausgabe dieser Anweisung zeigt den Status der clientseitigen Zeitstempel.

```
'client_side_timestamps': {'status': 'enabled'}
```

AWS CLI

Erstellen Sie eine neue Tabelle mit AWS CLI

1. Um eine neue Tabelle mit aktivierten clientseitigen Zeitstempeln zu erstellen, können Sie das folgende Beispiel verwenden.

```
./aws keyspaces create-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED' \
--schema-definition 'allColumns=[{name=id,type=int},{name=date,type=timestamp},
{name=name,type=text}],partitionKeys=[{name=id}]'
```

2. Führen Sie den folgenden Code aus, um zu überprüfen, ob die clientseitigen Zeitstempel für die neue Tabelle aktiviert sind.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

Die Ausgabe sollte in etwa wie in diesem Beispiel aussehen.

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table",
    "creationTimestamp": 1662681206.032,
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
```

```
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": 1662681206.032
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "clientSideTimestamps": {
        "status": "ENABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Client-seitige Zeitstempel für eine Tabelle in Amazon Keyspaces konfigurieren

Folgen Sie diesen Beispielen, um clientseitige Zeitstempel für bestehende Tabellen mithilfe von Amazon Keyspaces AWS-Managementkonsole, Cassandra Query Language (CQL) oder dem zu aktivieren. AWS Command Line Interface

Console

So aktivieren Sie clientseitige Zeitstempel für eine bestehende Tabelle (Konsole)

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie die Tabelle aus, die Sie aktualisieren möchten, und wählen Sie dann den Tab Zusätzliche Einstellungen.
3. Gehen Sie auf der Registerkarte Zusätzliche Einstellungen zu Clientseitige Zeitstempel ändern und wählen Sie Clientseitige Zeitstempel aktivieren
4. Wählen Sie Änderungen speichern, um die Einstellungen der Tabelle zu ändern.

Cassandra Query Language (CQL)

Verwenden Sie eine CQL-Anweisung

1. Aktivieren Sie clientseitige Zeitstempel für eine bestehende Tabelle mit der CQL-Anweisung. `ALTER TABLE`

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps':
 {'status': 'enabled'}};;
```

2. Um die clientseitigen Zeitstempeleinstellungen für die neue Tabelle zu überprüfen, verwenden Sie eine `SELECT` Anweisung, um die Einstellungen zu überprüfen, wie im `custom_properties` folgenden Beispiel gezeigt.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

Die Ausgabe dieser Anweisung zeigt den Status der clientseitigen Zeitstempel.

```
'client_side_timestamps': {'status': 'enabled'}
```

AWS CLI

Verwendung der AWS CLI

1. Mithilfe des folgenden Beispiels können Sie clientseitige Zeitstempel für eine vorhandene Tabelle aktivieren. AWS CLI

```
./aws keyspaces update-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED'
```

2. Führen Sie den folgenden Code aus, um zu überprüfen, ob die clientseitigen Zeitstempel für die Tabelle aktiviert sind.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

Die Ausgabe sollte diesem Beispiel ähneln und den Status für clientseitige Zeitstempel als angeben. ENABLED

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table",
    "creationTimestamp": 1662681312.906,
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
```

```
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": 1662681312.906
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "clientSideTimestamps": {
        "status": "ENABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

Verwenden Sie clientseitige Zeitstempel in Abfragen in Amazon Keyspaces

Nachdem Sie die clientseitigen Zeitstempel aktiviert haben, können Sie den Zeitstempel in Ihren `INSERT`, `UPDATE`, und `DELETE` -Anweisungen mit der Klausel übergeben. `USING TIMESTAMP`

Der Zeitstempelwert `bigint` entspricht einer Anzahl von Mikrosekunden seit der Standardbasiszeit, bekannt alsepoch: 1. Januar 1970 um 00:00:00 GMT. Ein vom Kunden bereitgestellter Zeitstempel muss zwischen 2 Tagen in der Vergangenheit und 5 Minuten in der future von der aktuellen Wanduhrzeit liegen.

Amazon Keyspaces speichert Zeitstempel-Metadaten für die gesamte Lebensdauer der Daten. Sie können die `WRITETIME` Funktion verwenden, um nach Zeitstempeln zu suchen, die Jahre in der Vergangenheit aufgetreten sind. Weitere Hinweise zur CQL-Syntax finden Sie unter. the section called “DML-Anweisungen.”

Die folgende CQL-Anweisung ist ein Beispiel für die Verwendung eines Zeitstempels als. `update_parameter`

```
INSERT INTO catalog.book_awards (year, award, rank, category, book_title, author,
 publisher)
   VALUES (2022, 'Wolf', 4, 'Non-Fiction', 'Science Update', 'Ana Carolina Silva',
 'SomePublisher')
   USING TIMESTAMP 1669069624;
```

Wenn Sie in Ihrer CQL-Abfrage keinen Zeitstempel angeben, verwendet Amazon Keyspaces den von Ihrem Client-Treiber übergebenen Zeitstempel. Wenn der Client-Treiber keinen Zeitstempel bereitstellt, weist Amazon Keyspaces Ihrem Schreibvorgang einen serverseitigen Zeitstempel zu.

Um den Zeitstempelwert zu sehen, der für eine bestimmte Spalte gespeichert ist, können Sie die `WRITETIME` Funktion in einer `SELECT` Anweisung verwenden, wie im folgenden Beispiel gezeigt.

```
SELECT year, award, rank, category, book_title, author, publisher, WRITETIME(year),
 WRITETIME(award), WRITETIME(rank),
  WRITETIME(category), WRITETIME(book_title), WRITETIME(author), WRITETIME(publisher)
 from catalog.book_awards;
```

Multi-Region Replikation für Amazon Keyspaces (für Apache Cassandra)

Sie können Amazon Keyspaces Multi-Region-Replikation verwenden, um Ihre Daten mit automatisierter, vollständig verwalteter, aktiv-aktiver Replikation an einem beliebigen Ort Ihrer Wahl zu replizieren. AWS-Regionen Bei der aktiv-aktiven Replikation ist jede Region in der Lage, Lese- und Schreibvorgänge isoliert durchzuführen. Sie können sowohl die Verfügbarkeit als auch die Resilienz aufgrund der regionalen Verschlechterung verbessern und gleichzeitig von lokalen Lese- und Schreibvorgängen mit niedriger Latenz für globale Anwendungen profitieren.

Bei der Replikation mit mehreren Regionen repliziert Amazon Keyspaces asynchron Daten zwischen Regionen, und Daten werden in der Regel innerhalb einer Sekunde über Regionen verteilt.

Außerdem müssen Sie sich bei der Replikation über mehrere Regionen nicht mehr die schwierige Arbeit machen, Konflikte zu lösen und Datendivergenzen zu korrigieren, sodass Sie sich auf Ihre Anwendung konzentrieren können.

Standardmäßig repliziert Amazon Keyspaces Daten über drei Availability Zones innerhalb derselben, um Haltbarkeit und hohe Verfügbarkeit AWS-Region zu gewährleisten. Mit der Replikation in mehreren Regionen können Sie Schlüsselbereiche für mehrere Regionen erstellen, die Ihre Tabellen in verschiedenen Regionen Ihrer Wahl replizieren. AWS-Regionen

Themen

- Vorteile der Verwendung der Replikation in mehreren Regionen
- Kapazitätsmodi und Preise
- So funktioniert die multiregionale Replikation in Amazon Keyspaces
- Nutzungshinweise zur Replikation in mehreren Regionen von Amazon Keyspaces
- Konfiguration der Replikation mehrerer Regionen für Amazon Keyspaces (für Apache Cassandra)

Vorteile der Verwendung der Replikation in mehreren Regionen

Multi-Region Die Replikation bietet die folgenden Vorteile.

- Globale Lese- und Schreibvorgänge mit Latenz im einstelligen Millisekundenbereich — In Amazon Keyspaces ist die Replikation aktiv-aktiv. Sie können sowohl Lese- als auch Schreibvorgänge lokal aus den Regionen bereitstellen, die Ihren Kunden am nächsten sind, mit einer Latenz im einstelligen Millisekundenbereich in jeder Größenordnung. Sie können Amazon Keyspaces-Tabellen mit mehreren Regionen für globale Anwendungen verwenden, die überall auf der Welt eine schnelle Reaktionszeit benötigen.
- Verbesserte Geschäftskontinuität und Schutz vor Beeinträchtigung einzelner Regionen — Mit der Replikation für mehrere Regionen können Sie sich nach einer Verschlechterung in einer einzigen Region erholen, AWS-Region indem Sie Ihre Anwendung in eine andere Region in Ihrem Schlüsselraum für mehrere Regionen umleiten. Da Amazon Keyspaces eine aktiv-aktive Replikation bietet, hat dies keine Auswirkungen auf Ihre Lese- und Schreibvorgänge.

  Amazon Keyspaces verfolgt alle Schreibvorgänge, die in Ihrem Multi-Region-Keyspace ausgeführt wurden, aber nicht an alle Replikatregionen weitergegeben wurden. Sobald die Region wieder online ist, synchronisiert Amazon Keyspaces automatisch alle fehlenden Änderungen, sodass Sie sie ohne Auswirkungen auf die Anwendung wiederherstellen können.

- High-speed regionsübergreifende Multi-Region Replikation — Bei der Replikation wird eine schnelle, speicherbasierte physische Replikation von Daten zwischen Regionen verwendet, wobei die Replikationsverzögerung in der Regel weniger als 1 Sekunde beträgt.

  Die Replikation in Amazon Keyspaces hat kaum bis gar keine Auswirkungen auf Ihre Datenbankabfragen, da sie keine Rechenressourcen mit Ihrer Anwendung teilt. Das bedeutet, dass Sie Anwendungsfälle mit hohem Schreibdurchsatz oder Anwendungsfälle mit plötzlichen Durchsatzspitzen oder Durchsatzspitzen bewältigen können, ohne dass sich dies auf die Anwendung auswirkt.
- Konsistenz und Konfliktlösung — Alle Änderungen, die an Daten in einer Region vorgenommen werden, werden in einem Schlüsselraum für mehrere Regionen auf die anderen Regionen repliziert. Wenn Anwendungen dieselben Daten in verschiedenen Regionen gleichzeitig aktualisieren, kann es zu Konflikten kommen.

  Um letztendlich für Konsistenz zu sorgen, verwendet Amazon Keyspaces Zeitstempel auf Zellenebene, und ein letzter Writer gewinnt den Abgleich zwischen gleichzeitigen Aktualisierungen. Die Konfliktlösung wird vollständig verwaltet und erfolgt im Hintergrund, ohne dass sich dies auf die Anwendung auswirkt.

Weitere Informationen zu unterstützten Konfigurationen und Funktionen finden Sie unterthe section called “Nutzungshinweise”.

Kapazitätsmodi und Preise

Für einen Schlüsselraum mit mehreren Regionen können Sie entweder den On-Demand-Kapazitätsmodus oder den Bereitstellungsmodus verwenden. Weitere Informationen finden Sie unter the section called “ read/write Kapazitätsmodi konfigurieren”.

Für den On-Demand-Modus wird Ihnen eine WRITE (WRU) in Rechnung gestellt, um bis zu 1 KB Daten pro Zeile auf dieselbe Weise zu schreiben wie für Tabellen mit nur einer Region. Ihnen werden jedoch Schreibvorgänge in jeder Region Ihres Multi-Region-Schlüsselraums in Rechnung gestellt. Zum Schreiben einer Datenzeile mit 3 KB in einen Multiregions-Schlüsselraum mit zwei Regionen sind beispielsweise 6 WRUs erforderlich: 3 * 2 = 6 WRUs. Darüber hinaus erfordern Schreibvorgänge, die sowohl statische als auch nicht statische Daten enthalten, zusätzliche Schreiboperationen.

Im Bereitstellungsmodus wird Ihnen eine Schreibkapazitätseinheit (WCU) für das Schreiben von bis zu 1 KB Daten pro Zeile in Rechnung gestellt, genau wie bei Tabellen mit einer Region. Ihnen werden

jedoch Schreibvorgänge in jeder Region Ihres Multi-Region-Schlüsselraums in Rechnung gestellt. Wenn Sie beispielsweise eine Zeile mit 3 KB Daten pro Sekunde in einen Multiregion-Schlüsselraum mit zwei Regionen schreiben möchten, sind 6 WCUs erforderlich: 3 * 2 = 6 WCUs. Darüber hinaus erfordern Schreibvorgänge, die sowohl statische als auch nicht statische Daten enthalten, zusätzliche Schreibvorgänge.

Weitere Informationen zu den Preisen finden Sie unter Amazon Keyspaces (für Apache Cassandra) — Preise.

So funktioniert die multiregionale Replikation in Amazon Keyspaces

Dieser Abschnitt bietet einen Überblick darüber, wie die multiregionale Replikation von Amazon Keyspaces funktioniert. Weitere Informationen zur Preisgestaltung finden Sie unter Preise für Amazon Keyspaces (für Apache Cassandra).

Themen

- So funktioniert die multiregionale Replikation in Amazon Keyspaces
- Multi-Region Lösung von Replikationskonflikten
- Multi-Region Replikation, Notfallwiederherstellung
- Multi-Region Replikation in AWS-Regionen standardmäßig deaktiviert
- Multi-Region Replikation und Integration mit Point-in-Time Recovery (PITR)
- Multi-Region Replikation und Integration mit AWS service

So funktioniert die multiregionale Replikation in Amazon Keyspaces

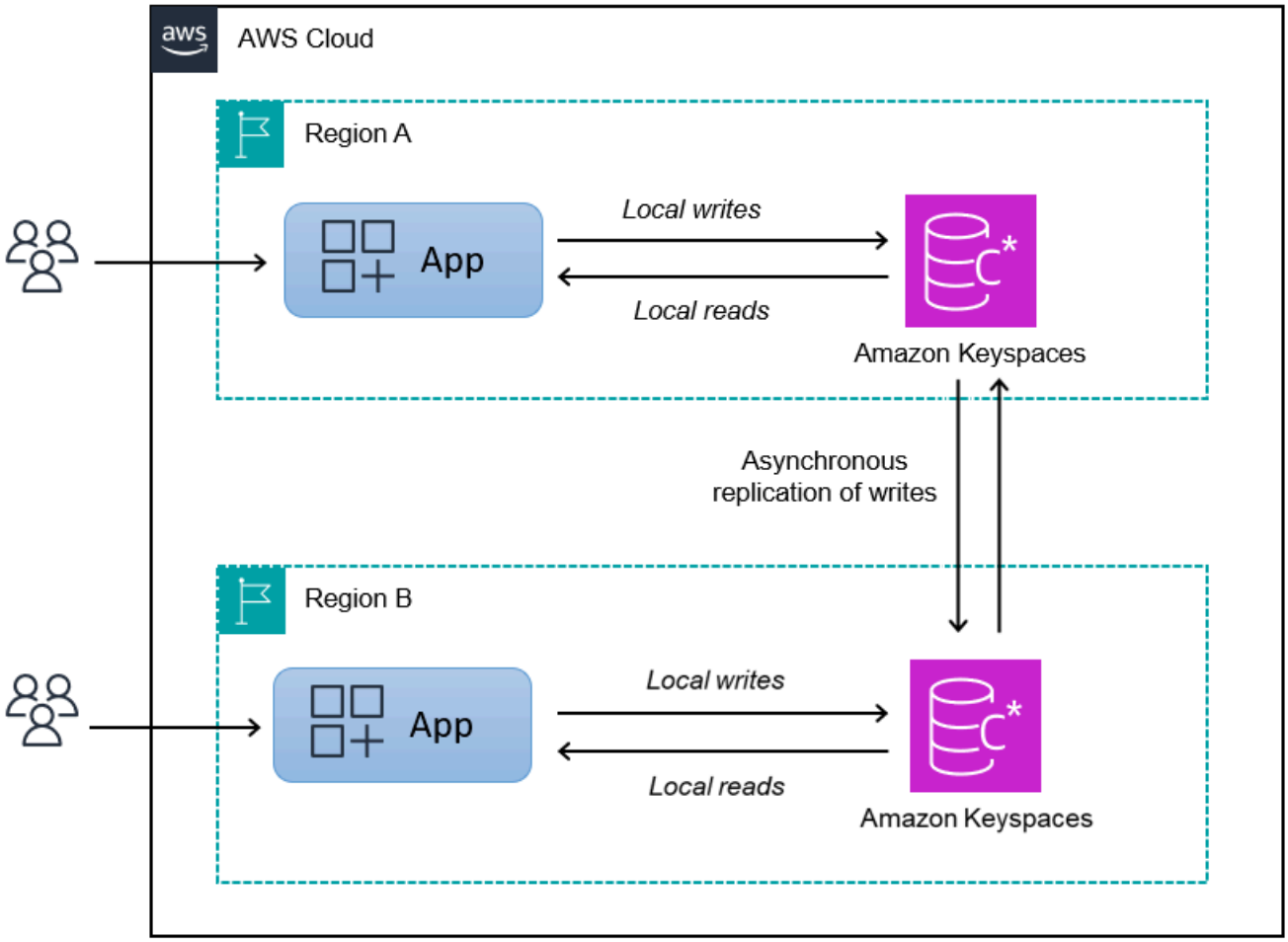
Die multiregionale Replikation von Amazon Keyspaces implementiert eine Architektur für Datenstabilität, die Ihre Daten unabhängig und geografisch verteilt verteilt. AWS-Regionen Es verwendet eine aktiv-aktive Replikation, die eine lokale niedrige Latenz bietet und jede Region in der Lage ist, Lese- und Schreibvorgänge isoliert durchzuführen.

Wenn Sie einen Amazon Keyspaces-Schlüsselraum für mehrere Regionen erstellen, können Sie zusätzliche Regionen auswählen, in die die Daten repliziert werden sollen. Jede Tabelle, die Sie in einem Schlüsselraum mit mehreren Regionen erstellen, besteht aus mehreren Replikattabellen (eine pro Region), die Amazon Keyspaces als eine Einheit betrachtet.

Jedes Replikat hat den gleichen Tabellennamen und das gleiche Primärschlüsselschema. Wenn eine Anwendung Daten in eine lokale Tabelle in einer Region schreibt, werden die Daten dauerhaft unter Verwendung der Konsistenzstufe geschrieben. `LOCAL_QUORUM` Amazon Keyspaces repliziert

die Daten automatisch asynchron in die anderen Replikationsregionen. Die Replikationsverzögerung zwischen den Regionen beträgt in der Regel weniger als eine Sekunde und hat keinen Einfluss auf die Leistung oder den Durchsatz Ihrer Anwendung.

Nachdem die Daten geschrieben wurden, können Sie sie aus der Tabelle mit mehreren Regionen in einer anderen Replikationsregion mit den `LOCAL_ONE/LOCAL_QUORUM` Konsistenzstufen lesen. Weitere Informationen zu unterstützten Konfigurationen und Funktionen finden Sie unterthe section called “Nutzungshinweise”.



Multi-Region Lösung von Replikationskonflikten

Die multiregionale Replikation von Amazon Keyspaces wird vollständig verwaltet. Das bedeutet, dass Sie keine Replikationsaufgaben wie regelmäßige Reparaturvorgänge zur Behebung von Problemen bei der Datensynchronisierung durchführen müssen. Amazon Keyspaces überwacht die Datenkonsistenz zwischen Tabellen in verschiedenen Tabellen, AWS-Regionen indem Konflikte erkannt und behoben werden, und die Replikate werden automatisch synchronisiert.

Amazon Keyspaces verwendet die Methode des Datenabgleichs, bei der der letzte Writer gewinnt. Mit diesem Konfliktlösungsmechanismus einigen sich alle Regionen in einem Schlüsselraum mit mehreren Regionen auf das neueste Update und nähern sich einem Staat an, in dem alle über identische Daten verfügen. Der Abgleichprozess hat keine Auswirkungen auf die Anwendungsleistung. Zur Unterstützung der Konfliktlösung werden clientseitige Zeitstempel für Tabellen mit mehreren Regionen automatisch aktiviert und können nicht deaktiviert werden. Weitere Informationen finden Sie unter the section called “Client-side Zeitstempel”.

Multi-Region Replikation, Notfallwiederherstellung

Bei der multiregionalen Replikation von Amazon Keyspaces werden Schreibvorgänge asynchron in jeder Region repliziert. Im seltenen Fall einer Verschlechterung oder eines Ausfalls einer einzelnen Region hilft Ihnen die multiregionale Replikation bei der Wiederherstellung nach einem Notfall, ohne dass sich dies auf Ihre Anwendung auswirkt. Die Wiederherstellung nach einem Notfall wird in der Regel anhand der Werte für Recovery Time Objective (RTO) und Recovery Point Objective (RPO) gemessen.

Zielvorgabe für die Wiederherstellungszeit — Die Zeit, die ein System benötigt, um nach einer Katastrophe wieder betriebsbereit zu sein. RTO misst die Menge an Ausfallzeiten, die Ihre Arbeitslast tolerieren kann, gemessen in der Zeit. Bei Disaster-Recovery-Plänen, die eine multiregionale Replikation verwenden, um ein Failover auf eine Region durchzuführen, die nicht betroffen ist, kann der RTO fast Null sein. Die RTO ist dadurch begrenzt, wie schnell Ihre Anwendung den Ausfall erkennen und den Datenverkehr in eine andere Region umleiten kann.

Recovery Point Objective — Die Datenmenge, die verloren gehen kann (gemessen in der Zeit). Bei Disaster-Recovery-Plänen, die eine multiregionale Replikation für ein Failover auf eine Region verwenden, die nicht betroffen ist, liegt das RPO in der Regel im einstelligen Sekundenbereich. Das RPO ist durch die Replikationslatenz auf das Failover-Zielreplikat begrenzt.

Im Falle eines regionalen Fehlers oder einer Verschlechterung müssen Sie keine sekundäre Region heraufstufen oder Datenbank-Failover-Verfahren durchführen, da die Replikation in Amazon Keyspaces aktiv-aktiv ist. Stattdessen können Sie Amazon Route 53 verwenden, um Ihre Anwendung an die nächstgelegene fehlerfreie Region weiterzuleiten. Weitere Informationen zu Route 53 finden Sie unter Was ist Amazon Route 53? .

Wenn eine einzelne Tabelle isoliert oder beeinträchtigt AWS-Region wird, kann Ihre Anwendung den Datenverkehr mithilfe von Route 53 in eine andere Region umleiten, um Lese- und Schreibvorgänge für eine andere Replikattabelle durchzuführen. Sie können auch benutzerdefinierte Geschäftslogik anwenden, um zu bestimmen, wann Anfragen in andere Regionen umgeleitet werden müssen. Ein

Beispiel hierfür ist, Ihre Anwendung auf die zahlreichen verfügbaren Endpunkte aufmerksam zu machen.

Wenn die Region wieder online ist, leitet Amazon Keyspaces alle ausstehenden Schreibvorgänge aus dieser Region wieder an die Replikattabellen in anderen Regionen weiter. Außerdem wird das Übertragen von Schreibvorgängen aus anderen Replikattabellen in die Region fortgesetzt, die jetzt wieder online ist.

Multi-Region Replikation in AWS-Regionen standardmäßig deaktiviert

Die multiregionale Replikation von Amazon Keyspaces wird in den folgenden Bereichen unterstützt AWS-Regionen , die standardmäßig deaktiviert sind:

- Region Afrika (Kapstadt)
- Region Naher Osten (UAE)
- Region Asien-Pazifik (Hongkong)
- Region Naher Osten (Bahrain)

Bevor Sie eine Region, die standardmäßig deaktiviert ist, mit der multiregionalen Replikation von Amazon Keyspaces verwenden können, müssen Sie zuerst die Region aktivieren. Weitere Informationen finden Sie im AWS Organizations Benutzerhandbuch unter AWS-Regionen In Ihrem Konto https://docs.aws.amazon.com/general/latest/gr/rande-manage.html#rande-manage-enable aktivieren oder deaktivieren.

Nachdem Sie eine Region aktiviert haben, können Sie in der Region neue Amazon Keyspaces-Ressourcen erstellen und die Region zu einem Schlüsselraum mit mehreren Regionen hinzufügen.

Wenn Sie eine Region deaktivieren, die von Amazon Keyspaces für die multiregionale Replikation verwendet wird, leitet Amazon Keyspaces eine 24-stündige Karenzzeit ein. Während dieses Zeitfensters können Sie das folgende Verhalten erwarten:

- Amazon Keyspaces führt weiterhin DML-Operationen (Data Manipulation Language) in aktivierten Regionen durch.
- Amazon Keyspaces unterbricht die Replikation von Datenaktualisierungen von aktivierten Regionen in die deaktivierte Region.
- Amazon Keyspaces blockiert alle DDL-Anfragen (Data Definition Language) in der deaktivierten Region.

Wenn Sie die Region irrtümlich deaktiviert haben, können Sie die Region innerhalb von 24 Stunden wieder aktivieren. Wenn Sie die Region während der 24-stündigen Übergangsfrist erneut aktivieren, ergreift Amazon Keyspaces die folgenden Maßnahmen:

- Setzt automatisch alle Replizierungen in die wieder aktivierte Region fort.
- Replizieren Sie alle Datenaktualisierungen, die in aktivierten Regionen stattgefunden haben, während die Region deaktiviert war, um die Datenkonsistenz sicherzustellen.
- Setzen Sie alle zusätzlichen Replikationsvorgänge für mehrere Regionen automatisch fort.

Falls die Region auch nach Ablauf des 24-Stunden-Fensters deaktiviert bleibt, ergreift Amazon Keyspaces die folgenden Maßnahmen, um die Region dauerhaft aus der multiregionalen Replikation zu entfernen:

- Entfernen Sie die deaktivierte Region aus allen Replikationsschlüsselräumen für mehrere Regionen.
- Konvertiert Replikate von Replikationstabellen mit mehreren Regionen in der deaktivierten Region in Schlüsselräume und Tabellen mit nur einer Region.
- Amazon Keyspaces löscht keine Ressourcen aus der deaktivierten Region.

Nachdem Amazon Keyspaces die deaktivierte Region dauerhaft aus dem Schlüsselraum für mehrere Regionen entfernt hat, können Sie die deaktivierte Region nicht wieder hinzufügen.

Multi-Region Replikation und Integration mit Point-in-Time Recovery (PITR)

Point-in-time Die Wiederherstellung wird für Tabellen mit mehreren Regionen unterstützt. Um eine Tabelle mit mehreren Regionen mit PITR erfolgreich wiederherzustellen, müssen die folgenden Bedingungen erfüllt sein.

- Die Quell- und die Zieltabelle müssen als Tabellen mit mehreren Regionen konfiguriert sein.
- Die Replikationsregionen für den Schlüsselraum der Quelltabelle und für den Schlüsselraum der Zieltabelle müssen identisch sein.
- PITR muss für alle Replikate der Quelltabelle aktiviert sein.

Sie können die Restore-Anweisung von jeder Region aus ausführen, in der die Quelltabelle verfügbar ist. Amazon Keyspaces stellt die Zieltabelle in jeder Region automatisch wieder her. Weitere Informationen zu PITR finden Sie unter the section called “Funktionsweise”.

Wenn Sie eine Tabelle mit mehreren Regionen erstellen, werden die PITR-Einstellungen, die Sie während des Erstellungsprozesses definieren, automatisch auf alle Tabellen in allen Regionen angewendet. Wenn Sie die PITR-Einstellungen mithilfe von Amazon Keyspaces ändern`ALTER TABLE`, wendet das Update nur auf die lokale Tabelle an und nicht auf die Replikate in anderen Regionen. Um PITR für eine bestehende Tabelle mit mehreren Regionen zu aktivieren, müssen Sie die `ALTER TABLE` Anweisung für alle Replikate wiederholen.

Multi-Region Replikation und Integration mit AWS service

Sie können die Replikationsleistung zwischen Tabellen in verschiedenen Tabellen mithilfe AWS-Regionen von CloudWatch Amazon-Metriken überwachen. Die folgende Metrik ermöglicht die kontinuierliche Überwachung von Schlüsselräumen mit mehreren Regionen.

- `ReplicationLatency`— Diese Metrik misst die Zeit`updates`, `inserts` die für die Replikation oder `deletes` von einer Replikattabelle zu einer anderen Replikattabelle in einem Schlüsselraum mit mehreren Regionen benötigt wurde.

Weitere Informationen zur Überwachung von Metriken finden Sie unter. CloudWatch the section called “Überwachung mit CloudWatch”

Nutzungshinweise zur Replikation in mehreren Regionen von Amazon Keyspaces

Beachten Sie Folgendes, wenn Sie die Replikation in mehreren Regionen mit Amazon Keyspaces verwenden.

- Sie können eine der verfügbaren öffentlichen Optionen auswählen. AWS-Regionen Weitere Informationen dazu AWS-Regionen , dass diese standardmäßig deaktiviert sind, finden Sie unterthe section called “Regionen sind standardmäßig deaktiviert”.
- AWS GovCloud (US) Regions und China Regionen werden nicht unterstützt.
- Ziehen Sie die folgenden Abhilfemaßnahmen in Betracht, bis die Funktionen verfügbar sind:

  Konfigurieren Sie Time to Live (TTL), wenn Sie die Tabelle mit mehreren Regionen erstellen. Sie können TTL nicht aktivieren und deaktivieren oder den TTL-Wert später anpassen. Weitere Informationen finden Sie unter the section called “Daten mit Time to Live ablaufen lassen”.
  - Verwenden Sie für die Verschlüsselung im Ruhezustand einen AWS eigenen Schlüssel. Vom Kunden verwaltete Schlüssel werden derzeit nicht für Tabellen mit mehreren Regionen unterstützt. Die Ausgabe des obigen Befehls sieht in etwa folgendermaßen aus (JSON format).

the section called “Funktionsweise”.

- Sie können sie verwendenALTER KEYSPACE, um eine Region zu einem Schlüsselraum mit einer oder mehreren Regionen hinzuzufügen. Weitere Informationen finden Sie unter the section called “Fügen Sie einem Schlüsselraum eine Region hinzu”.
  - Bevor Sie einem Schlüsselraum mit einer Region eine Region hinzufügen, stellen Sie sicher, dass keine Tabellen unter dem Schlüsselraum mit vom Kunden verwalteten Schlüsseln konfiguriert sind.
  - Alle vorhandenen Tags, die für Schlüsselräume oder Tabellen konfiguriert sind, werden nicht in die neue Region repliziert.
- Wenn Sie das bereitgestellte Kapazitätsmanagement mit Amazon Keyspaces Auto Scaling verwenden, stellen Sie sicher, dass Sie die Amazon Keyspaces-API-Operationen verwenden, um Ihre Tabellen mit mehreren Regionen zu erstellen und zu konfigurieren. Die zugrunde liegenden API-Operationen für Application Auto Scaling, die Amazon Keyspaces in Ihrem Namen aufruft, verfügen nicht über Funktionen für mehrere Regionen.

  Weitere Informationen finden Sie unter the section called “Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling für eine Tabelle mit mehreren Regionen”. Weitere Informationen zur Schätzung des Schreibkapazitätsdurchsatzes von bereitgestellten Tabellen mit mehreren Regionen finden Sie unter. the section called “Schätzen Sie die Kapazität für eine Tabelle mit mehreren Regionen”
- Obwohl Daten automatisch über die ausgewählten Regionen einer Tabelle mit mehreren Regionen repliziert werden, gibt die Abfrage nur lokale Informationen zurück, wenn ein Client eine Verbindung zu einem Endpunkt in einer Region herstellt und die `system.peers` Tabelle abfragt. Das Abfrageergebnis sieht für den Client wie ein einzelner Rechenzentrumscluster aus.
- Die multiregionale Replikation von Amazon Keyspaces ist asynchron und unterstützt `LOCAL_QUORUM` Konsistenz bei Schreibvorgängen. `LOCAL_QUORUM`Konsistenz erfordert, dass eine Aktualisierung einer Zeile dauerhaft auf zwei Replikaten in der lokalen Region gespeichert wird, bevor der Client erfolgreich zurückgemeldet wird. Die Übertragung von Schreibvorgängen in die replizierte Region (oder Regionen) erfolgt dann asynchron.

  Die multiregionale Replikation von Amazon Keyspaces unterstützt keine synchrone Replikation oder Konsistenz. `QUORUM`
- Wenn Sie einen Schlüsselraum oder eine Tabelle mit mehreren Regionen erstellen, werden alle Tags, die Sie während des Erstellungsprozesses definieren, automatisch auf alle Schlüsselräume und Tabellen in allen Regionen angewendet. Wenn Sie die vorhandenen Tags mit `ALTER`

`KEYSPACE` oder ändern`ALTER TABLE`, wird die Aktualisierung nur auf den Schlüsselraum oder die Tabelle in der Region angewendet, in der Sie die Änderung vornehmen.

- Amazon CloudWatch stellt eine `ReplicationLatency` Metrik für jede replizierte Region bereit. Es berechnet diese Metrik, indem es eingehende Zeilen verfolgt, ihre Ankunftszeit mit ihrer ursprünglichen Schreibzeit vergleicht und einen Durchschnitt berechnet. Die Zeitangaben werden innerhalb der CloudWatch Quellregion gespeichert. Weitere Informationen finden Sie unter the section called “Überwachung mit CloudWatch”.

  Es kann nützlich sein, sich die durchschnittlichen und maximalen Zeiten anzusehen, um die durchschnittliche Verzögerung bei der Replikation und im schlimmsten Fall zu ermitteln. Für diese Latenz gibt es kein SLA.

- Wenn Sie eine Tabelle mit mehreren Regionen im On-Demand-Modus verwenden, stellen Sie möglicherweise eine Erhöhung der Latenz bei der asynchronen Replikation von Schreibvorgängen fest, wenn bei einem Tabellenreplikat eine neue Datenverkehrsspitze auftritt. Ähnlich wie Amazon Keyspaces die Kapazität einer On-Demand-Tabelle mit einer Region automatisch an den empfangenen Anwendungsdatenverkehr anpasst, passt Amazon Keyspaces die Kapazität einer On-Demand-Tabellenreplikation mit mehreren Regionen automatisch an den empfangenen Datenverkehr an. Der Anstieg der Replikationslatenz ist vorübergehend, da Amazon Keyspaces automatisch mehr Kapazität zuweist, wenn Ihr Datenverkehrsvolumen zunimmt. Sobald sich alle Replikate an Ihr Datenverkehrsvolumen angepasst haben, sollte sich die Replikationslatenz wieder normalisieren. Weitere Informationen finden Sie unter the section called “Spitzenaufkommen an Datenverkehr und Skalierungseigenschaften”.

- Wenn Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus verwenden und Ihre Anwendung die bereitgestellte Durchsatzkapazität überschreitet, können Fehler bei unzureichender Kapazität und eine Erhöhung der Replikationslatenz auftreten. Um sicherzustellen, dass immer genügend Lese- und Schreibkapazität für alle Tabellenreplikate in einer Tabelle mit mehreren Regionen vorhanden ist, empfehlen wir Ihnen, Amazon Keyspaces Auto Scaling zu konfigurieren. AWS-Regionen Das Auto Scaling von Amazon Keyspaces ermöglicht die effiziente Bereitstellung der Durchsatzkapazität für variable Workloads, indem die Durchsatzkapazität automatisch in Reaktion auf den tatsächlichen Datenverkehr der Anwendung angepasst wird. Weitere Informationen finden Sie unter the section called “So funktioniert Auto Scaling für Tabellen mit mehreren Regionen”.

Konfiguration der Replikation mehrerer Regionen für Amazon Keyspaces (für Apache Cassandra)

Sie können die Konsole, Cassandra Query Language (CQL) oder die verwenden, AWS Command Line Interface um regionsübergreifende Schlüsselräume und Tabellen in Amazon Keyspaces zu erstellen und zu verwalten.

Dieser Abschnitt enthält Beispiele für die Erstellung und Verwaltung von Schlüsselräumen und Tabellen für mehrere Regionen. Alle Tabellen, die Sie in einem Schlüsselraum mit mehreren Regionen erstellen, erben automatisch die Einstellungen für mehrere Regionen aus dem Schlüsselraum.

Weitere Informationen zu unterstützten Konfigurationen und Funktionen finden Sie unter. the section called “Nutzungshinweise”

Themen

- Konfigurieren Sie die IAM-Berechtigungen, die für die Erstellung von Schlüsselräumen und Tabellen mit mehreren Regionen erforderlich sind
- Konfigurieren Sie die IAM-Berechtigungen, die zum Hinzufügen von zu einem Schlüsselraum erforderlich sind AWS-Region
- Einen multiregionalen Schlüsselraum in Amazon Keyspaces erstellen
- AWS-Region Zu einem Schlüsselraum in Amazon Keyspaces hinzufügen
- Überprüfen Sie den Replikationsfortschritt, wenn Sie einem Schlüsselraum eine neue Region hinzufügen
- Erstellen Sie eine Tabelle mit mehreren Regionen mit Standardeinstellungen in Amazon Keyspaces
- Erstellen Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus mit auto Skalierung in Amazon Keyspaces
- Aktualisieren Sie die bereitgestellten Kapazitäten und Auto-Scaling-Einstellungen für eine Tabelle mit mehreren Regionen in Amazon Keyspaces
- Sehen Sie sich die bereitgestellten Kapazitäten und Auto-Scaling-Einstellungen für eine Tabelle mit mehreren Regionen in Amazon Keyspaces an
- auto Skalierung für eine Tabelle in Amazon Keyspaces deaktivieren
- Stellen Sie die bereitgestellte Kapazität einer Tabelle mit mehreren Regionen manuell in Amazon Keyspaces ein

Konfigurieren Sie die IAM-Berechtigungen, die für die Erstellung von Schlüsselräumen und Tabellen mit mehreren Regionen erforderlich sind

Um Schlüsselräume und Tabellen mit mehreren Regionen erfolgreich zu erstellen, muss der IAM-Prinzipal in der Lage sein, eine dienstbezogene Rolle zu erstellen. Bei dieser serviceverknüpften Rolle handelt es sich um eine einzigartige Art von IAM-Rolle, die von Amazon Keyspaces vordefiniert ist. Es beinhaltet alle Berechtigungen, die Amazon Keyspaces benötigt, um Aktionen in Ihrem Namen durchzuführen. Weitere Informationen zur serviceverknüpften Rolle finden Sie unter the section called “Multi-Region Replikation”.

Um die serviceverknüpfte Rolle zu erstellen, die für die Replikation in mehreren Regionen erforderlich ist, erfordert die Richtlinie für den IAM-Prinzipal die folgenden Elemente:

- `iam:CreateServiceLinkedRole`— Die Aktion, die der Principal ausführen kann.
- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`— Die Ressource, auf der die Aktion ausgeführt werden kann.
- `iam:AWSServiceName": "replication.cassandra.amazonaws.com`— Der einzige AWS Service, an den diese Rolle angehängt werden kann, ist Amazon Keyspaces.

Im Folgenden finden Sie ein Beispiel für die Richtlinie, die einem Prinzipal die erforderlichen Mindestberechtigungen zum Erstellen von Schlüsselräumen und Tabellen für mehrere Regionen gewährt.

```
{
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication",
            "Condition": {"StringLike": {"iam:AWSServiceName":
 "replication.cassandra.amazonaws.com"}}
}
```

Zusätzliche IAM-Berechtigungen für Schlüsselräume und Tabellen mit mehreren Regionen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Keyspaces (für Apache Cassandra) in der Service Authorization Reference.

Konfigurieren Sie die IAM-Berechtigungen, die zum Hinzufügen von zu einem Schlüsselraum erforderlich sind AWS-Region

Um einem Schlüsselraum eine Region hinzuzufügen, benötigt der IAM-Prinzipal die folgenden Berechtigungen:

- `cassandra:Alter`
- `cassandra:AlterMultiRegionResource`
- `cassandra:Create`
- `cassandra:CreateMultiRegionResource`
- `cassandra:Select`
- `cassandra:SelectMultiRegionResource`
- `cassandra:Modify`
- `cassandra:ModifyMultiRegionResource`

Wenn die Tabelle im Bereitstellungsmodus mit aktiviertem Auto Scaling konfiguriert ist, sind die folgenden zusätzlichen Berechtigungen erforderlich.

- `application-autoscaling:RegisterScalableTarget`
- `application-autoscaling:DeregisterScalableTarget`
- `application-autoscaling:DescribeScalableTargets`
- `application-autoscaling:PutScalingPolicy`
- `application-autoscaling:DescribeScalingPolicies`

Um erfolgreich eine Region zu einem Single-Region-Schlüsselraum hinzuzufügen, muss der IAM-Prinzipal auch in der Lage sein, eine dienstbezogene Rolle zu erstellen. Bei dieser serviceverknüpften Rolle handelt es sich um eine einzigartige Art von IAM-Rolle, die von Amazon Keyspaces vordefiniert ist. Es beinhaltet alle Berechtigungen, die Amazon Keyspaces benötigt, um Aktionen in Ihrem Namen durchzuführen. Weitere Informationen zur serviceverknüpften Rolle finden Sie unter the section called “Multi-Region Replikation”.

Um die serviceverknüpfte Rolle zu erstellen, die für die Replikation in mehreren Regionen erforderlich ist, erfordert die Richtlinie für den IAM-Prinzipal die folgenden Elemente:

- `iam:CreateServiceLinkedRole`— Die Aktion, die der Principal ausführen kann.

- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`— Die Ressource, auf der die Aktion ausgeführt werden kann.
- `iam:AWSServiceName": "replication.cassandra.amazonaws.com`— Der einzige AWS Service, an den diese Rolle angehängt werden kann, ist Amazon Keyspaces.

Im Folgenden finden Sie ein Beispiel für die Richtlinie, die einem Prinzipal die erforderlichen Mindestberechtigungen gewährt, um einem Schlüsselraum eine Region hinzuzufügen.

```
{
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication",
            "Condition": {"StringLike": {"iam:AWSServiceName":
 "replication.cassandra.amazonaws.com"}}
}
```

Einen multiregionalen Schlüsselraum in Amazon Keyspaces erstellen

Dieser Abschnitt enthält Beispiele für die Erstellung eines Schlüsselraums für mehrere Regionen. Sie können dies in der Amazon Keyspaces-Konsole mit CQL oder dem tun. AWS CLI Alle Tabellen, die Sie in einem Schlüsselraum mit mehreren Regionen erstellen, erben automatisch die Einstellungen für mehrere Regionen aus dem Schlüsselraum.

> **Note**
>
> Wenn Sie einen Schlüsselraum für mehrere Regionen erstellen, erstellt Amazon Keyspaces eine servicebezogene Rolle mit dem Namen `AWSServiceRoleForAmazonKeyspacesReplication` in Ihrem Konto. Diese Rolle ermöglicht es Amazon Keyspaces, Schreibvorgänge in alle Replikate einer Tabelle mit mehreren Regionen in Ihrem Namen zu replizieren. Weitere Informationen hierzu finden Sie unter the section called “Multi-Region Replikation”.

Console

Erstellen Sie einen Schlüsselraum für mehrere Regionen (Konsole)

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces und anschließend Create Keyspace aus.
3. Geben Sie unter Schlüsselraumname den Namen für den Schlüsselraum ein.
4. Im Multi-Region Replikationsbereich können Sie die zusätzlichen Regionen hinzufügen, die in der Liste verfügbar sind.
5. Um den Vorgang abzuschließen, wählen Sie Create Keyspace.

Cassandra Query Language (CQL)

Erstellen Sie mithilfe von CQL einen Schlüsselraum mit mehreren Regionen

1. Um einen Schlüsselraum mit mehreren Regionen zu erstellen, geben Sie hier den an AWS-Regionen , `NetworkTopologyStrategy` in dem der Schlüsselraum repliziert werden soll. Sie müssen Ihre aktuelle Region und mindestens eine weitere Region angeben.

   Alle Tabellen im Schlüsselraum erben die Replikationsstrategie aus dem Schlüsselraum. Sie können die Replikationsstrategie auf Tabellenebene nicht ändern.

   `NetworkTopologyStrategy`— Der Replikationsfaktor für jede Region ist drei, da Amazon Keyspaces standardmäßig Daten über drei Availability Zones innerhalb derselben AWS-Region repliziert.

   Die folgende CQL-Anweisung ist ein Beispiel dafür.

   ```
   CREATE KEYSPACE mykeyspace
   WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-southeast-1':'3','eu-west-1':'3' };
   ```

2. Sie können eine CQL-Anweisung verwenden, um die `tables` Tabelle im `system_multiregion_info` Schlüsselraum abzufragen, um die Regionen und den Status der von Ihnen angegebenen Tabelle mit mehreren Regionen programmgesteuert aufzulisten. Der folgende Code ist ein Beispiel dafür.

```
SELECT * from system_multiregion_info.tables WHERE keyspace_name = 'mykeyspace'
 AND table_name = 'mytable';
```

Die Ausgabe der Anweisung sieht wie folgt aus:

```
 keyspace_name  | table_name     | region         | status
----------------+----------------+----------------+--------
 mykeyspace     | mytable        | us-east-1      | ACTIVE
 mykeyspace     | mytable        | ap-southeast-1 | ACTIVE
 mykeyspace     | mytable        | eu-west-1      | ACTIVE
```

CLI

Erstellen Sie einen neuen Schlüsselraum für mehrere Regionen mit dem AWS CLI

- Um einen Schlüsselraum für mehrere Regionen zu erstellen, können Sie die folgende CLI-Anweisung verwenden. Geben Sie Ihre aktuelle Region und mindestens eine weitere Region in der an. `regionList`

```
aws keyspaces create-keyspace --keyspace-name mykeyspace \
--replication-specification replicationStrategy=MULTI_REGION,regionList=us-
east-1,eu-west-1
```

Informationen zum Erstellen einer Tabelle mit mehreren Regionen finden Sie unter the section called “Erstellen Sie eine Tabelle mit mehreren Regionen mit Standardeinstellungen” undthe section called “Erstellen Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus”.

AWS-Region Zu einem Schlüsselraum in Amazon Keyspaces hinzufügen

Sie können einem Schlüsselraum einen neuen Schlüsselraum hinzufügen, bei dem es sich entweder AWS-Region um einen einzelnen Schlüsselraum oder um einen Schlüsselraum mit mehreren Regionen handelt. Die neue Replikat-Region wird auf alle Tabellen im Schlüsselraum angewendet.

Um einen Schlüsselraum mit einer Region in einen Schlüsselraum mit mehreren Regionen zu ändern, müssen Sie clientseitige Zeitstempel für alle Tabellen im Schlüsselraum aktivieren. Weitere Informationen finden Sie unter the section called “Client-side Zeitstempel”.

Wenn Sie eine zusätzliche Region zu einem Schlüsselraum mit mehreren Regionen hinzufügen, muss Amazon Keyspaces die vorhandenen Tabellen in die neue Region replizieren, indem für jede bestehende Tabelle eine einmalige regionsübergreifende Wiederherstellung durchgeführt wird. Die Wiederherstellungsgebühren für jede Tabelle werden pro GB abgerechnet. Weitere Informationen finden Sie unter Backup und Wiederherstellung auf der Preisseite von Amazon Keyspaces (für Apache Cassandra). Für diesen Wiederherstellungsvorgang fallen keine Gebühren für die Datenübertragung zwischen den Regionen an. Zusätzlich zu den Daten werden alle Tabelleneigenschaften mit Ausnahme von Tags in die neue Region repliziert.

Sie können die `ALTER KEYSPACE` Anweisung in CQL, den `update-keyspace` Befehl mit der oder die Konsole verwenden AWS CLI, um eine neue Region zu einem einzelnen Schlüsselraum oder einem Schlüsselraum mit mehreren Regionen in Amazon Keyspaces hinzuzufügen. Um die Abrechnung erfolgreich ausführen zu können, muss sich das von Ihnen verwendete Konto in einer der Regionen befinden, in denen der Schlüsselraum bereits verfügbar ist. Während das Replikat hinzugefügt wird, können Sie keine anderen DDL-Operationen (Data Definition Language) für die Ressourcen ausführen, die aktualisiert und repliziert werden.

Weitere Informationen zu den zum Hinzufügen einer Region erforderlichen Berechtigungen finden Sie unter. the section called “Konfigurieren Sie die IAM-Berechtigungen für das Hinzufügen einer Region”

> **Note**
>
> Wenn Sie eine zusätzliche Region zu einem Schlüsselraum mit einer einzigen Region hinzufügen, erstellt Amazon Keyspaces eine serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonKeyspacesReplication` in Ihrem Konto. Diese Rolle ermöglicht es Amazon Keyspaces, Tabellen in neue Regionen zu replizieren und Schreibvorgänge von einer Tabelle in alle Replikate einer Tabelle mit mehreren Regionen in Ihrem Namen zu replizieren. Weitere Informationen hierzu finden Sie unter the section called “Multi-Region Replikation”.

Console

Gehen Sie wie folgt vor, um mithilfe der Amazon Keyspaces-Konsole eine Region zu einem Keyspace hinzuzufügen.

Fügen Sie einem Keyspace (Konsole) eine Region hinzu

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces und dann einen Keyspace aus der Liste aus.
3. Wählen Sie die Registerkarte AWS-Regionen aus.
4. Wählen Sie auf der AWS-RegionenRegisterkarte „Region hinzufügen“ aus.
5. Wählen Sie im Dialogfeld „Region hinzufügen“ die zusätzliche Region aus, die Sie dem Schlüsselraum hinzufügen möchten.
6. Um den Vorgang abzuschließen, wählen Sie „Hinzufügen“.

Cassandra Query Language (CQL)

Fügen Sie mithilfe von CQL eine Region zu einem Schlüsselraum hinzu

- Um einem Schlüsselraum eine neue Region hinzuzufügen, können Sie die folgende Anweisung verwenden. In diesem Beispiel ist der Schlüsselraum bereits in den Regionen USA Ost (Nord-Virginia) und USA West (Oregon) verfügbar, und die CQL-Anweisung fügt die Region USA West (Nordkalifornien) hinzu.

```
ALTER KEYSPACE my_keyspace
WITH REPLICATION = {
    'class': 'NetworkTopologyStrategy',
    'us-east-1': '3',
    'us-west-2': '3',
    'us-west-1': '3'
} AND CLIENT_SIDE_TIMESTAMPS = {'status': 'ENABLED'};
```

CLI

Fügen Sie einem Schlüsselraum eine Region hinzu, indem Sie den AWS CLI

- Um mit der CLI eine neue Region zu einem Schlüsselraum hinzuzufügen, können Sie das folgende Beispiel verwenden. Der Standardwert für `client-side-timestamps` ist `DISABLED`. Mit dem `update-keyspace` Befehl müssen Sie den Wert auf `ENABLED` ändern.

```
aws keyspaces update-keyspace \
```

```
--keyspace-name my_keyspace \
--replication-specification '{"replicationStrategy": "MULTI_REGION",
 "regionList": ["us-east-1", "eu-west-1", "eu-west-3"] }' \
--client-side-timestamps '{"status": "ENABLED"}'
```

Überprüfen Sie den Replikationsfortschritt, wenn Sie einem Schlüsselraum eine neue Region hinzufügen

Das Hinzufügen einer neuen Region zu einem Amazon Keyspaces-Keyspace ist ein langwieriger Vorgang. Um den Fortschritt zu verfolgen, können Sie die in diesem Abschnitt aufgeführten Abfragen verwenden.

Cassandra Query Language (CQL)

Verwenden Sie CQL, um den Fortschritt beim Hinzufügen von Regionen zu überprüfen

- Um den Fortschritt der Erstellung der neuen Tabellenreplikate in einem bestimmten Schlüsselraum zu überprüfen, können Sie die Tabelle abfragen. `system_multiregion_info.keyspaces` Die folgende CQL-Anweisung ist ein Beispiel dafür.

  ```
  SELECT keyspace_name, region, status, tables_replication_progress
  FROM system_multiregion_info.keyspaces
  WHERE keyspace_name = 'my_keyspace';
  ```

  Während ein Replikationsvorgang läuft, zeigt der Status den Fortschritt der Tabellenerstellung in der neuen Region an. Dies ist ein Beispiel, bei dem 5 von 10 Tabellen in die neue Region repliziert wurden.

  ```
   keyspace_name | region    | status    | tables_replication_progress
  ---------------+-----------+-----------+-----------------------
     my_keyspace | us-east-1 | Updating  |
     my_keyspace | us-west-2 | Updating  |
     my_keyspace | eu-west-1 | Creating  | 50%
  ```

  Nachdem der Replikationsvorgang erfolgreich abgeschlossen wurde, sollte die Ausgabe wie in diesem Beispiel aussehen.

  ```
   keyspace_name | region    | status
  ```

```
---------------+-----------+-----------
   my_keyspace | us-east-1 | Active
   my_keyspace | us-west-2 | Active
   my_keyspace | eu-west-1 | Active
```

CLI

Verwenden Sie den AWS CLI , um den Fortschritt des Hinzufügens der Region zu überprüfen

- Um den Status der Erstellung eines Tabellenreplikats für einen bestimmten Schlüsselraum zu überprüfen, können Sie das folgende Beispiel verwenden.

```
aws keyspaces get-keyspace \
--keyspace-name my_keyspace
```

Die Ausgabe sollte diesem Beispiel ähneln.

```
{
    "keyspaceName": "my_keyspace",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
    "replicationStrategy": "MULTI_REGION",
    "replicationRegions": [
        "us-east-1",
        "eu-west-1"
    ]
    "replicationGroupStatus": [
        {
            "RegionName": "us-east-1",
            "KeyspaceStatus": "Active"
        },
        {
            "RegionName": "eu-west-1",
            "KeyspaceStatus": "Creating",
            "TablesReplicationProgress": "50.0%"
        }
    ]
}
```

Erstellen Sie eine Tabelle mit mehreren Regionen mit Standardeinstellungen in Amazon Keyspaces

Dieser Abschnitt enthält Beispiele für die Erstellung einer Tabelle mit mehreren Regionen im On-Demand-Modus mit allen Standardeinstellungen. Sie können dies in der Amazon Keyspaces-Konsole mit CQL oder dem tun. AWS CLI Alle Tabellen, die Sie in einem Schlüsselraum mit mehreren Regionen erstellen, erben automatisch die Einstellungen für mehrere Regionen aus dem Schlüsselraum.

Informationen zum Erstellen eines Schlüsselraums für mehrere Regionen finden Sie unter. the section called “Erstellen Sie einen Schlüsselraum für mehrere Regionen”

Console

Erstellen Sie eine Tabelle mit mehreren Regionen mit Standardeinstellungen (Konsole)

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie einen Schlüsselraum für mehrere Regionen.
3. Wählen Sie auf der Registerkarte Tabellen die Option Tabelle erstellen aus.
4. Geben Sie unter Tabellenname den Namen für die Tabelle ein. Die AWS-Regionen , in die diese Tabelle repliziert wird, werden im Infofeld angezeigt.
5. Fahren Sie mit dem Tabellenschema fort.
6. Fahren Sie unter Tabelleneinstellungen mit der Option Standardeinstellungen fort. Beachten Sie die folgenden Standardeinstellungen für Tabellen mit mehreren Regionen.

   - Kapazitätsmodus — Der Standardkapazitätsmodus ist On-demand. Weitere Informationen zur Konfiguration des Bereitstellungsmodus finden Sie unterthe section called “Erstellen Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus”.
   - Verwaltung von Verschlüsselungsschlüsseln — Nur die AWS-eigener SchlüsselOption wird unterstützt.
   - Client-side Zeitstempel — Diese Funktion ist für Tabellen mit mehreren Regionen erforderlich.

   - Wählen Sie Einstellungen anpassen, wenn Sie Time to Live (TTL) für die Tabelle und all ihre Replikate aktivieren müssen.

> **Note**
>
> Sie können die TTL-Einstellungen für eine bestehende Tabelle mit mehreren Regionen nicht ändern.

7. Um den Vorgang abzuschließen, wählen Sie Tabelle erstellen.

Cassandra Query Language (CQL)

Erstellen Sie eine Tabelle mit mehreren Regionen im On-Demand-Modus mit Standardeinstellungen

- Um eine Tabelle mit mehreren Regionen mit Standardeinstellungen zu erstellen, können Sie die folgende CQL-Anweisung verwenden.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
    WITH CUSTOM_PROPERTIES = {
 'capacity_mode':{
  'throughput_mode':'PAY_PER_REQUEST'
 },
 'point_in_time_recovery':{
  'status':'enabled'
 },
 'encryption_specification':{
  'encryption_type':'AWS_OWNED_KMS_KEY'
 },
 'client_side_timestamps':{
  'status':'enabled'
 }
};
```

CLI

Unter Verwendung der AWS CLI

1. Um eine Tabelle mit mehreren Regionen mit Standardeinstellungen zu erstellen, müssen Sie nur das Schema angeben. Sie können das folgende Beispiel verwenden.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable \
```

```
--schema-definition 'allColumns=[{name=pk,type=int}],partitionKeys={name= pk}'
```

Die Ausgabe des Befehls ist:

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable"
}
```

2. Um die Einstellungen der Tabelle zu bestätigen, können Sie die folgende Anweisung verwenden.

```
aws keyspaces get-table --keyspace-name mykeyspace --table-name mytable
```

Die Ausgabe zeigt alle Standardeinstellungen einer Tabelle mit mehreren Regionen.

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable",
    "creationTimestamp": "2023-12-19T16:50:37.639000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "pk",
                "type": "int"
            }
        ],
        "partitionKeys": [
            {
                "name": "pk"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2023-12-19T16:50:37.639000+00:00"
    },
```

```
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    },
    "clientSideTimestamps": {
        "status": "ENABLED"
    },
    "replicaSpecifications": [
        {
            "region": "us-east-1",
            "status": "ACTIVE",
            "capacitySpecification": {
                "throughputMode": "PAY_PER_REQUEST",
                "lastUpdateToPayPerRequestTimestamp": 1702895811.469
            }
        },
        {
            "region": "eu-north-1",
            "status": "ACTIVE",
            "capacitySpecification": {
                "throughputMode": "PAY_PER_REQUEST",
                "lastUpdateToPayPerRequestTimestamp": 1702895811.121
            }
        }
    ]
}
```

Erstellen Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus mit auto Skalierung in Amazon Keyspaces

Dieser Abschnitt enthält Beispiele für die Erstellung einer Tabelle mit mehreren Regionen im Bereitstellungsmodus mit Auto Scaling. Sie können dies in der Amazon Keyspaces-Konsole mit CQL oder dem tun. AWS CLI

Weitere Informationen zu unterstützten Konfigurationen und Replikationsfunktionen für mehrere Regionen finden Sie unter. the section called “Nutzungshinweise”

Informationen zum Erstellen eines Schlüsselraums für mehrere Regionen finden Sie unter. the section called “Erstellen Sie einen Schlüsselraum für mehrere Regionen”

Wenn Sie eine neue Tabelle mit mehreren Regionen im Bereitstellungsmodus mit Auto Scaling-Einstellungen erstellen, können Sie die allgemeinen Einstellungen für die Tabelle angeben, die für alle gelten, in AWS-Regionen denen die Tabelle repliziert wird. Sie können dann die Lesekapazitätseinstellungen und die Auto Scaling-Einstellungen für jedes Replikat überschreiben. Die Schreibkapazität bleibt jedoch zwischen allen Replikaten synchronisiert, um sicherzustellen, dass genügend Kapazität vorhanden ist, um Schreibvorgänge in allen Regionen zu replizieren.

> **Note**
>
> Die automatische Skalierung von Amazon Keyspace erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`), die automatische Skalierungsaktionen in Ihrem Namen ausführt. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “Verwenden von servicegebundenen Rollen”.

Console

Erstellen Sie eine neue Tabelle mit mehreren Regionen mit aktivierter automatischer Skalierung

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie einen Schlüsselraum für mehrere Regionen.
3. Wählen Sie auf der Registerkarte Tabellen die Option Tabelle erstellen aus.
4. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die neue Tabelle ein.
5. Erstellen Sie im Abschnitt Spalten das Schema für Ihre Tabelle.
6. Definieren Sie im Abschnitt Primärschlüssel den Primärschlüssel der Tabelle und wählen Sie optionale Clusterspalten aus.
7. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.
8. Fahren Sie mit den Read/write Kapazitätseinstellungen fort.
9. Wählen Sie für den Kapazitätsmodus die Option Bereitgestellt aus.

10. Vergewissern Sie sich, dass im Abschnitt Lesekapazität die Option Automatisch skalieren ausgewählt ist.

    Sie können festlegen, dass für alle, in die die Tabelle repliziert wird AWS-Regionen , dieselben Lesekapazitätseinheiten konfiguriert werden. Alternativ können Sie das Kontrollkästchen deaktivieren und die Lesekapazität für jede Region unterschiedlich konfigurieren.

    Wenn Sie jede Region anders konfigurieren möchten, wählen Sie die minimalen und maximalen Lesekapazitätseinheiten für jedes Tabellenreplikat sowie die Zielauslastung aus.

    - Mindestkapazitätseinheiten — Geben Sie den Wert für den Mindestdurchsatz ein, den die Tabelle immer unterstützen soll. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
    - Einheiten mit maximaler Kapazität — Geben Sie den maximalen Durchsatz ein, den Sie für die Tabelle bereitstellen möchten. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
    - Zielauslastung — Geben Sie eine Zielauslastungsrate zwischen 20 und 90% ein. Wenn der Verkehr die definierte Zielauslastungsrate überschreitet, wird die Kapazität automatisch hochskaliert. Wenn der Verkehr unter das definierte Ziel fällt, wird er automatisch wieder herunterskaliert.

    - Deaktivieren Sie das Kontrollkästchen Automatisch skalieren, wenn Sie die Lesekapazität der Tabelle manuell bereitstellen möchten. Diese Einstellung gilt für alle Replikate der Tabelle.

      > **Note**
      >
      > Um sicherzustellen, dass genügend Lesekapazität für alle Replikate vorhanden ist, empfehlen wir die automatische Skalierung von Amazon Keyspaces für bereitgestellte Tabellen mit mehreren Regionen.

    > **Note**
    >
    > Weitere Informationen zu Standardkontingenten für Ihr Konto und deren Erhöhung finden Sie unter. Kontingente

11. Vergewissern Sie sich, dass im Abschnitt Schreibkapazität die Option Automatisch skalieren ausgewählt ist. Konfigurieren Sie dann die Kapazitätseinheiten für die Tabelle. Die Schreibkapazitätseinheiten bleiben auf allen Seiten synchronisiert, AWS-Regionen um sicherzustellen, dass genügend Kapazität für die Replikation von Schreibereignissen in den Regionen vorhanden ist.
    - Deaktivieren Sie Automatisch skalieren, wenn Sie die Schreibkapazität der Tabelle manuell bereitstellen möchten. Diese Einstellung gilt für alle Replikate der Tabelle.

      > **Note**
      >
      > Um sicherzustellen, dass genügend Schreibkapazität für alle Replikate vorhanden ist, empfehlen wir die automatische Skalierung von Amazon Keyspaces für bereitgestellte Tabellen mit mehreren Regionen.

12. Wählen Sie Create table (Tabelle erstellen) aus. Ihre Tabelle wird mit den angegebenen Parametern für die automatische Skalierung erstellt.

Cassandra Query Language (CQL)

Erstellen Sie eine Tabelle mit mehreren Regionen mit bereitgestelltem Kapazitätsmodus und auto Skalierung mithilfe von CQL

- Um eine Tabelle mit mehreren Regionen im Bereitstellungsmodus mit Auto Scaling zu erstellen, müssen Sie zunächst den Kapazitätsmodus angeben, indem Sie ihn `CUSTOM_PROPERTIES` für die Tabelle definieren. Nachdem Sie den Modus für die bereitgestellte Kapazität angegeben haben, können Sie die Auto Scaling-Einstellungen für die Tabelle mithilfe von `AUTOSCALING_SETTINGS` konfigurieren.

  Ausführliche Informationen zu den Auto Scaling-Einstellungen, der Zielverfolgungsrichtlinie, dem Zielwert und optionalen Einstellungen finden Sie unterthe section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”.

  Um die Lesekapazität für ein Tabellenreplikat in einer bestimmten Region zu definieren, können Sie die folgenden Parameter als Teil der `replica_updates` Tabelle konfigurieren:

  - Die Region
  - Die bereitgestellten Lesekapazitätseinheiten (optional)
  - Einstellungen für die automatische Skalierung der Lesekapazität (optional)

Das folgende Beispiel zeigt eine `CREATE TABLE` Anweisung für eine Tabelle mit mehreren Regionen im Bereitstellungsmodus. Die allgemeinen Einstellungen für die auto Skalierung der Schreib- und Lesekapazität sind identisch. Die Einstellungen für die auto Leseskalierung spezifizieren jedoch zusätzliche Abklingzeiten von 60 Sekunden, bevor die Lesekapazität der Tabelle nach oben oder unten skaliert wird. Darüber hinaus sind die Einstellungen für die auto Skalierung der Lesekapazität für die Region USA Ost (Nord-Virginia) höher als die Einstellungen für andere Replikate. Außerdem ist der Zielwert auf 70% statt auf 50% festgelegt.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 5,
        'write_capacity_units': 5
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
            }
        }
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    },
    'replica_updates': {
        'us-east-1': {
            'provisioned_read_capacity_autoscaling_update': {
```

```
                'maximum_units': 20,
                'minimum_units': 5,
                'scaling_policy': {
                    'target_tracking_scaling_policy_configuration': {
                        'target_value': 70
                    }
                }
            }
        }
    }
};
```

CLI

Erstellen Sie eine neue Tabelle mit mehreren Regionen im Bereitstellungsmodus mit Auto Scaling mithilfe der AWS CLI

- Um eine Tabelle mit mehreren Regionen im Bereitstellungsmodus mit Auto Scaling-Konfiguration zu erstellen, können Sie die verwenden. AWS CLI Beachten Sie, dass Sie den Amazon `create-table` Keyspaces-CLI-Befehl verwenden müssen, um Auto-Scaling-Einstellungen für mehrere Regionen zu konfigurieren. Dies liegt daran, dass Application Auto Scaling, der Service, den Amazon Keyspaces verwendet, um Auto Scaling in Ihrem Namen durchzuführen, nicht mehrere Regionen unterstützt.

  Weitere Informationen zu Auto Scaling-Einstellungen, der Zielverfolgungsrichtlinie, dem Zielwert und optionalen Einstellungen finden Sie unterthe section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”.

  Um die Lesekapazität für ein Tabellenreplikat in einer bestimmten Region zu definieren, können Sie die folgenden Parameter als Teil der `replicaSpecifications` Tabelle konfigurieren:

  - Die Region
  - Die bereitgestellten Lesekapazitätseinheiten (optional)
  - Einstellungen für die automatische Skalierung der Lesekapazität (optional)

  Wenn Sie bereitgestellte Tabellen mit mehreren Regionen mit komplexen Auto Scaling-Einstellungen und unterschiedlichen Konfigurationen für Tabellenreplikate erstellen, ist es

hilfreich, die Auto Scaling-Einstellungen und Replikatkonfigurationen der Tabelle aus JSON-Dateien zu laden.

Um das folgende Codebeispiel zu verwenden, können Sie die JSON-Beispieldateien von auto-scaling.zip herunterladen und extrahieren `auto-scaling.json` und`replication.json`. Notieren Sie sich den Pfad zu den Dateien.

In diesem Beispiel befinden sich die JSON-Dateien im aktuellen Verzeichnis. Informationen zu verschiedenen Dateipfadoptionen finden Sie unter So laden Sie Parameter aus einer Datei.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable \
--schema-definition 'allColumns=[{name=pk,type=int},
{name=ck,type=int}],partitionKeys=[{name=pk},{name=ck}]' \
--capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1 \
--auto-scaling-specification file://auto-scaling.json \
--replica-specifications file://replication.json
```

Aktualisieren Sie die bereitgestellten Kapazitäten und Auto-Scaling-Einstellungen für eine Tabelle mit mehreren Regionen in Amazon Keyspaces

Dieser Abschnitt enthält Beispiele für die Verwendung der Konsole, der CQL und der AWS CLI zur Verwaltung der Amazon Keyspaces-Auto-Scaling-Einstellungen von bereitgestellten Tabellen mit mehreren Regionen. Weitere Informationen zu allgemeinen Auto Scaling-Konfigurationsoptionen und deren Funktionsweise finden Sie unterthe section called “Verwalten Sie die Durchsatzkapazität mit Auto Scaling”.

Beachten Sie, dass Sie, wenn Sie den Modus mit bereitgestellter Kapazität für Tabellen mit mehreren Regionen verwenden, immer Amazon Keyspaces-API-Aufrufe verwenden müssen, um Auto Scaling zu konfigurieren. Dies liegt daran, dass die zugrunde liegenden Application Auto Scaling Scaling-API-Operationen dies nicht sind Region-aware.

Weitere Informationen zur Schätzung des Schreibkapazitätsdurchsatzes von bereitgestellten Tabellen mit mehreren Regionen finden Sie unter. the section called “Schätzen Sie die Kapazität für eine Tabelle mit mehreren Regionen”

Weitere Informationen zur Amazon Keyspaces-API finden Sie in der Amazon Keyspaces-API-Referenz.

Wenn Sie den Bereitstellungsmodus oder die Auto Scaling-Einstellungen einer Tabelle mit mehreren Regionen aktualisieren, können Sie die Lesekapazitätseinstellungen und die auto Lesekonfiguration für jedes Replikat der Tabelle aktualisieren.

Die Schreibkapazität bleibt jedoch zwischen allen Replikaten synchronisiert, um sicherzustellen, dass genügend Kapazität vorhanden ist, um Schreibvorgänge in allen Regionen zu replizieren.

Cassandra Query Language (CQL)

Aktualisieren Sie die bereitgestellten Kapazitäten und die Einstellungen für die auto Skalierung einer Tabelle mit mehreren Regionen mithilfe von CQL

- Sie können `ALTER TABLE` es verwenden, um den Kapazitätsmodus und die Auto Scaling-Einstellungen einer vorhandenen Tabelle zu aktualisieren. Wenn Sie eine Tabelle aktualisieren, die sich derzeit im On-Demand-Kapazitätsmodus `capacity_mode` befindet, ist dies erforderlich. Wenn sich Ihre Tabelle bereits im Modus für bereitgestellte Kapazität befindet, kann dieses Feld weggelassen werden.

  Ausführliche Informationen zu den Auto Scaling-Einstellungen, der Zielverfolgungsrichtlinie, dem Zielwert und optionalen Einstellungen finden Sie unterthe section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”.

  In derselben Anweisung können Sie auch die Lesekapazität und die Einstellungen für die auto Skalierung von Tabellenreplikaten in bestimmten Regionen aktualisieren, indem Sie die `replica_updates` Eigenschaft der Tabelle aktualisieren. Die folgende Aussage ist ein Beispiel dafür.

```
ALTER TABLE mykeyspace.mytable
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    }
} AND AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50
```

```
            }
        }
    },
    'provisioned_read_capacity_autoscaling_update': {
        'maximum_units': 10,
        'minimum_units': 5,
        'scaling_policy': {
            'target_tracking_scaling_policy_configuration': {
                'target_value': 50,
                'scale_in_cooldown': 60,
                'scale_out_cooldown': 60
            }
        }
    },
    'replica_updates': {
        'us-east-1': {
            'provisioned_read_capacity_autoscaling_update': {
                'maximum_units': 20,
                'minimum_units': 5,
                'scaling_policy': {
                    'target_tracking_scaling_policy_configuration': {
                        'target_value': 70
                    }
                }
            }
        }
    }
};
```

CLI

Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling einer Tabelle mit mehreren Regionen mithilfe der AWS CLI

- Um den Bereitstellungsmodus und die Auto Scaling-Konfiguration einer vorhandenen Tabelle zu aktualisieren, können Sie den AWS CLI `update-table` Befehl verwenden.

  Beachten Sie, dass Sie die Amazon Keyspaces-CLI-Befehle verwenden müssen, um Auto-Scaling-Einstellungen für mehrere Regionen zu erstellen oder zu ändern. Dies liegt daran, dass Application Auto Scaling, der Service, den Amazon Keyspaces verwendet, um die

auto Skalierung der Tabellenkapazität in Ihrem Namen durchzuführen, nicht mehrere AWS-Regionen unterstützt.

Um die Lesekapazität für ein Tabellenreplikat in einer bestimmten Region zu aktualisieren, können Sie einen der folgenden optionalen Tabellenparameter ändern: `replicaSpecifications`

- Die bereitgestellten Lesekapazitätseinheiten (optional)
- Einstellungen für die automatische Skalierung der Lesekapazität (optional)

Wenn Sie Tabellen mit mehreren Regionen mit komplexen Auto Scaling-Einstellungen und unterschiedlichen Konfigurationen für Tabellenreplikate aktualisieren, ist es hilfreich, die Auto Scaling-Einstellungen und Replikatkonfigurationen der Tabelle aus JSON-Dateien zu laden.

Um das folgende Codebeispiel zu verwenden, können Sie die JSON-Beispieldateien von auto-scaling.zip herunterladen und extrahieren `auto-scaling.json` und`replication.json`. Notieren Sie sich den Pfad zu den Dateien.

In diesem Beispiel befinden sich die JSON-Dateien im aktuellen Verzeichnis. Informationen zu verschiedenen Dateipfadoptionen finden Sie unter So laden Sie Parameter aus einer Datei.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable \
--capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1 \
--auto-scaling-specification file://auto-scaling.json \
--replica-specifications file://replication.json
```

Sehen Sie sich die bereitgestellten Kapazitäten und Auto-Scaling-Einstellungen für eine Tabelle mit mehreren Regionen in Amazon Keyspaces an

Sie können die bereitgestellte Kapazität und die Einstellungen für die auto Skalierung einer Tabelle mit mehreren Regionen in der Amazon Keyspaces-Konsole mithilfe von CQL oder dem anzeigen. AWS CLI Dieser Abschnitt enthält Beispiele dafür, wie Sie dies mithilfe von CQL und dem tun können. AWS CLI

Cassandra Query Language (CQL)

Zeigen Sie die bereitgestellten Kapazitäten und Auto-Scaling-Einstellungen einer Tabelle mit mehreren Regionen mithilfe von CQL an

- Verwenden Sie den folgenden Befehl, um die Auto Scaling-Konfiguration einer Tabelle mit mehreren Regionen anzuzeigen.

```
SELECT * FROM system_multiregion_info.autoscaling WHERE keyspace_name =
 'mykeyspace' AND table_name = 'mytable';
```

Die Ausgabe für diesen Befehl sieht wie folgt aus:

```
 keyspace_name  | table_name | region         |
 provisioned_read_capacity_autoscaling_update

                                                                          |
 provisioned_write_capacity_autoscaling_update
---------------+------------+---------------
+-------------------------------------------------------------------------------------------
+-------------------------------------------------------------------------------------------
  mykeyspace    |  mytable   | ap-southeast-1 | {'minimum_units': 5,
 'maximum_units': 10, 'scaling_policy':
 {'target_tracking_scaling_policy_configuration': {'scale_out_cooldown':
 60, 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown':
 60}}} | {'minimum_units': 5, 'maximum_units': 10, 'scaling_policy':
 {'target_tracking_scaling_policy_configuration': {'scale_out_cooldown': 0,
 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown': 0}}}
  mykeyspace    |  mytable   | us-east-1      | {'minimum_units': 5,
 'maximum_units': 20, 'scaling_policy':
 {'target_tracking_scaling_policy_configuration': {'scale_out_cooldown':
 60, 'disable_scale_in': false, 'target_value': 70, 'scale_in_cooldown':
 60}}} | {'minimum_units': 5, 'maximum_units': 10, 'scaling_policy':
 {'target_tracking_scaling_policy_configuration': {'scale_out_cooldown': 0,
 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown': 0}}}
  mykeyspace    |  mytable   | eu-west-1      | {'minimum_units': 5,
 'maximum_units': 10, 'scaling_policy':
 {'target_tracking_scaling_policy_configuration': {'scale_out_cooldown':
 60, 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown':
 60}}} | {'minimum_units': 5, 'maximum_units': 10, 'scaling_policy':
 {'target_tracking_scaling_policy_configuration': {'scale_out_cooldown': 0,
 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown': 0}}}
```

CLI

Zeigen Sie die bereitgestellte Kapazität und die Einstellungen für die auto Skalierung einer Tabelle mit mehreren Regionen an, indem Sie AWS CLI

- Um die Auto Scaling-Konfiguration einer Tabelle mit mehreren Regionen anzuzeigen, können Sie den `get-table-auto-scaling-settings` Vorgang verwenden. Der folgende CLI-Befehl ist ein Beispiel dafür.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --
table-name mytable
```

Die Ausgabe sollte folgendermaßen aussehen.

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:777788889999:/keyspace/
mykeyspace/table/mytable",
    "autoScalingSpecification": {
        "writeCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 10,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 0,
                    "scaleOutCooldown": 0,
                    "targetValue": 50.0
                }
            }
        },
        "readCapacityAutoScaling": {
            "autoScalingDisabled": false,
            "minimumUnits": 5,
            "maximumUnits": 20,
            "scalingPolicy": {
                "targetTrackingScalingPolicyConfiguration": {
                    "disableScaleIn": false,
                    "scaleInCooldown": 60,
                    "scaleOutCooldown": 60,
```

```
                    "targetValue": 70.0
                }
            }
        }
    },
    "replicaSpecifications": [
        {
            "region": "us-east-1",
            "autoScalingSpecification": {
                "writeCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 10,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 0,
                            "scaleOutCooldown": 0,
                            "targetValue": 50.0
                        }
                    }
                },
                "readCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 20,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 60,
                            "scaleOutCooldown": 60,
                            "targetValue": 70.0
                        }
                    }
                }
            }
        },
        {
            "region": "eu-north-1",
            "autoScalingSpecification": {
                "writeCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 10,
```

```
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 0,
                            "scaleOutCooldown": 0,
                            "targetValue": 50.0
                        }
                    }
                },
                "readCapacityAutoScaling": {
                    "autoScalingDisabled": false,
                    "minimumUnits": 5,
                    "maximumUnits": 10,
                    "scalingPolicy": {
                        "targetTrackingScalingPolicyConfiguration": {
                            "disableScaleIn": false,
                            "scaleInCooldown": 60,
                            "scaleOutCooldown": 60,
                            "targetValue": 50.0
                        }
                    }
                }
            }
        ]
}
```

auto Skalierung für eine Tabelle in Amazon Keyspaces deaktivieren

Dieser Abschnitt enthält Beispiele dafür, wie Sie Auto Scaling für eine Tabelle mit mehreren Regionen im Modus für bereitgestellte Kapazität deaktivieren können. Sie können dies in der Amazon Keyspaces-Konsole mit CQL oder dem tun. AWS CLI

> **Important**
>
> Wir empfehlen die Verwendung von Auto Scaling für Tabellen mit mehreren Regionen, die den Modus für bereitgestellte Kapazität verwenden. Weitere Informationen finden Sie unter the section called “Schätzen Sie die Kapazität für eine Tabelle mit mehreren Regionen”.

> **Note**
>
> Um die dienstverknüpfte Rolle zu löschen, die Application Auto Scaling verwendet, müssen Sie die automatische Skalierung für alle Tabellen im Konto deaktivieren. AWS-Regionen

Console

Automatische Skalierung von Amazon Keyspace für eine bestehende Tabelle mit mehreren Regionen auf der Konsole ausschalten

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie die Tabelle aus, mit der Sie arbeiten möchten, und klicken Sie auf die Registerkarte Kapazität.
3. Wählen Sie im Abschnitt Kapazitätseinstellungen die Option Bearbeiten aus.
4. Um die automatische Skalierung von Amazon Keyspaces zu deaktivieren, deaktivieren Sie das Kontrollkästchen Automatisch skalieren. Wenn Sie die automatische Skalierung deaktivieren, wird die Tabelle mit Application Auto Scaling als skalierbares Ziel abgemeldet. Um die serviceverknüpfte Rolle zu löschen, die Application Auto Scaling für den Zugriff auf Ihre Amazon Keyspaces-Tabelle verwendet, folgen Sie den Schritten unter. the section called “Löschen einer serviceverknüpften Rolle für Amazon Keyspaces”
5. Wenn die Einstellungen für die automatische Skalierung definiert sind, wählen Sie Speichern.

Cassandra Query Language (CQL)

auto Skalierung für eine Tabelle mit mehreren Regionen mithilfe von CQL deaktivieren

- Sie können `ALTER TABLE` es verwenden, um Auto Scaling für eine bestehende Tabelle zu deaktivieren. Beachten Sie, dass Sie Auto Scaling für ein einzelnes Tabellenreplikat nicht deaktivieren können.

  Im folgenden Beispiel ist Auto Scaling für die Lesekapazität der Tabelle deaktiviert.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_read_capacity_autoscaling_update': {
        'autoscaling_disabled': true
```

```
    }
};
```

CLI

Deaktivieren Sie die auto Skalierung für eine Tabelle mit mehreren Regionen mithilfe des AWS CLI

- Sie können den AWS CLI `update-table` Befehl verwenden, um Auto Scaling für eine bestehende Tabelle zu deaktivieren. Beachten Sie, dass Sie Auto Scaling für ein einzelnes Tabellenreplikat nicht deaktivieren können.

  Im folgenden Beispiel ist Auto Scaling für die Lesekapazität der Tabelle deaktiviert.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
          \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

Stellen Sie die bereitgestellte Kapazität einer Tabelle mit mehreren Regionen manuell in Amazon Keyspaces ein

Wenn Sie Auto Scaling für eine Tabelle mit mehreren Regionen deaktivieren müssen, können Sie die Lesekapazität der Tabelle für eine Replikattabelle manuell mithilfe von CQL oder dem bereitstellen. AWS CLI

> **Note**
>
> Wir empfehlen die Verwendung von Auto Scaling für Tabellen mit mehreren Regionen, die den Modus für bereitgestellte Kapazität verwenden. Weitere Informationen finden Sie unter the section called “Schätzen Sie die Kapazität für eine Tabelle mit mehreren Regionen”.

Cassandra Query Language (CQL)

Manuelles Einstellen der bereitgestellten Kapazität einer Tabelle mit mehreren Regionen mithilfe von CQL

- Sie können `ALTER TABLE` es verwenden, um die Lesekapazität der Tabelle für eine Replikattabelle manuell bereitzustellen.

```
ALTER TABLE mykeyspace.mytable
WITH CUSTOM_PROPERTIES = {
    'capacity_mode': {
        'throughput_mode': 'PROVISIONED',
        'read_capacity_units': 1,
        'write_capacity_units': 1
    },
    'replica_updates': {
        'us-east-1': {
            'read_capacity_units': 2
         }
    }
};
```

CLI

Legen Sie die bereitgestellte Kapazität einer Tabelle mit mehreren Regionen manuell fest, indem Sie AWS CLI

- Wenn Sie Auto Scaling für eine Tabelle mit mehreren Regionen deaktivieren müssen, können Sie `update-table` damit die Lesekapazität der Tabelle für eine Replikattabelle manuell bereitstellen.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable \
--capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=1,writeCapacityUnits=1 \
--replica-specifications region="us-east-1",readCapacityUnits=5
```

Daten mit Point-in-Time-Recovery für Amazon Keyspaces Backup und wiederherstellen

Point-in-time Recovery (PITR) schützt Ihre Amazon Keyspaces-Tabellen vor versehentlichen Schreib- oder Löschvorgängen, indem es Ihnen kontinuierliche Backups Ihrer Tabellendaten bereitstellt.

Nehmen wir zum Beispiel an, dass ein Testskript versehentlich in eine Amazon Keyspaces-Produktionstabelle schreibt. Mit der Point-in-Time-Wiederherstellung können Sie die Daten dieser Tabelle auf eine beliebige Sekunde seit der Aktivierung von PITR innerhalb der letzten 35 Tage wiederherstellen. Wenn Sie eine Tabelle mit aktivierter Point-in-Time-Wiederherstellung löschen, können Sie die Daten der gelöschten Tabelle 35 Tage lang abfragen (ohne zusätzliche Kosten) und sie in den Zustand zurückversetzen, in dem sie sich kurz vor dem Löschen befand.

Sie können eine Amazon Keyspaces-Tabelle auf einen bestimmten Zeitpunkt zurücksetzen, indem Sie die Konsole, das AWS SDK und die AWS Command Line Interface (AWS CLI) oder Cassandra Query Language (CQL) verwenden. Weitere Informationen finden Sie unter Verwenden Sie Point-in-Time-Wiederherstellung in Amazon Keyspaces.

Point-in-time Operationen haben keine Auswirkungen auf die Leistung oder Verfügbarkeit der Basistabelle, und das Wiederherstellen einer Tabelle verbraucht keinen zusätzlichen Durchsatz.

Informationen zu PITR-Kontingenten finden Sie unterKontingente.

Preisinformationen finden Sie unter Amazon Keyspaces (für Apache Cassandra) — Preise.

Themen

- So funktioniert die Point-in-Time-Wiederherstellung in Amazon Keyspaces
- Verwenden Sie Point-in-Time-Wiederherstellung in Amazon Keyspaces

So funktioniert die Point-in-Time-Wiederherstellung in Amazon Keyspaces

Dieser Abschnitt bietet einen Überblick darüber, wie Amazon Keyspaces Point-in-Time Recovery (PITR) funktioniert. Weitere Informationen zu den Preisen finden Sie unter Amazon Keyspaces (für Apache Cassandra) — Preise.

Themen

- Zeitfenster für kontinuierliche PITR-Backups
- Einstellungen für PITR wiederherstellen

- PITR-Wiederherstellung verschlüsselter Tabellen
- PITR-Wiederherstellung von Tabellen mit mehreren Regionen
- PITR-Wiederherstellung von Tabellen mit benutzerdefinierten Typen (UDTs)
- Zeit für die Wiederherstellung der Tabelle mit PITR
- Amazon Keyspaces PITR und Integration mit Diensten AWS

Zeitfenster für kontinuierliche PITR-Backups

Amazon Keyspaces PITR verwendet zwei Zeitstempel, um den Zeitrahmen beizubehalten, für den wiederherstellbare Backups für eine Tabelle verfügbar sind.

- Frühester wiederherstellbarer Zeitpunkt — Kennzeichnet den Zeitpunkt des frühesten wiederherstellbaren Backups. Das früheste wiederherstellbare Backup reicht bis zu 35 Tage zurück oder zu dem Zeitpunkt, zu dem PITR aktiviert wurde, je nachdem, welcher Zeitpunkt der jüngste ist. Das maximale Backup-Fenster von 35 Tagen kann nicht geändert werden.
- Aktuelle Uhrzeit — Der Zeitstempel für das letzte wiederherstellbare Backup ist die aktuelle Uhrzeit. Wenn während einer Wiederherstellung kein Zeitstempel angegeben wird, wird die aktuelle Uhrzeit verwendet.

Wenn PITR aktiviert ist, können Sie zu einem beliebigen Zeitpunkt zwischen und `EarliestRestorableDateTime` wiederherstellen. `CurrentTime` Sie können Tabellendaten nur zu einem Zeitpunkt wiederherstellen, zu dem PITR aktiviert war.

Wenn Sie PITR deaktivieren und später wieder aktivieren, setzen Sie die Startzeit für das erste verfügbare Backup auf den Zeitpunkt zurück, zu dem PITR wieder aktiviert wurde. Das bedeutet, dass durch die Deaktivierung von PITR Ihr Backup-Verlauf gelöscht wird.

> **Note**
>
> DDL-Operationen (Data Definition Language) an Tabellen, wie z. B. Schemaänderungen, werden asynchron ausgeführt. In Ihren wiederhergestellten Tabellendaten können Sie nur abgeschlossene Operationen sehen. Möglicherweise werden jedoch zusätzliche Aktionen in Ihrer Quelltabelle angezeigt, wenn diese zum Zeitpunkt der Wiederherstellung ausgeführt wurden. Eine Liste der DDL-Anweisungen finden Sie unterthe section called “DDL-Anweisungen”.

Eine Tabelle muss nicht aktiv sein, um wiederhergestellt zu werden. Sie können gelöschte Tabellen auch wiederherstellen, wenn PITR für die gelöschte Tabelle aktiviert war und der Löschvorgang innerhalb des Backup-Fensters (oder innerhalb der letzten 35 Tage) erfolgte.

> **Note**
>
> Wenn eine neue Tabelle mit demselben qualifizierten Namen (z. B. mykeyspace.mytable) wie eine zuvor gelöschte Tabelle erstellt wird, kann die gelöschte Tabelle nicht mehr wiederhergestellt werden. Wenn Sie versuchen, dies von der Konsole aus zu tun, wird eine Warnung angezeigt.

Einstellungen für PITR wiederherstellen

Wenn Sie eine Tabelle mit PITR wiederherstellen, stellt Amazon Keyspaces das Schema und die Daten Ihrer Quelltabelle auf den Status zurück, der auf dem ausgewählten Zeitstempel (`day:hour:minute:second`) in einer neuen Tabelle basiert. PITR überschreibt keine vorhandenen Tabellen.

Zusätzlich zum Schema und den Daten der Tabelle stellt PITR die `custom_properties` aus der Quelltabelle wieder her. Im Gegensatz zu den Daten der Tabelle, die auf der Grundlage des ausgewählten Zeitstempels zwischen dem frühesten Wiederherstellungszeitpunkt und dem aktuellen Zeitpunkt wiederhergestellt werden, werden benutzerdefinierte Eigenschaften immer auf der Grundlage der Tabelleneinstellungen zum aktuellen Zeitpunkt wiederhergestellt.

Die Einstellungen der wiederhergestellten Tabelle entsprechen den Einstellungen der Quelltabelle mit dem Zeitstempel, zu dem die Wiederherstellung initiiert wurde. Wenn Sie diese Einstellungen während der Wiederherstellung überschreiben möchten, können Sie dies mit tun. `WITH custom_properties` Zu den benutzerdefinierten Eigenschaften gehören die folgenden Einstellungen.

- Read/write Kapazitätsmodus
- Einstellungen für die bereitgestellte Durchsatzkapazität
- PITR-Einstellungen

Wenn sich die Tabelle im Modus für bereitgestellte Kapazität befindet und Auto Scaling aktiviert ist, werden beim Wiederherstellungsvorgang auch die Auto Scaling-Einstellungen der Tabelle

wiederhergestellt. Sie können sie mit dem `autoscaling_settings` Parameter in CQL oder `autoScalingSpecification` mit der CLI überschreiben. Weitere Informationen zu den Einstellungen für die auto Skalierung finden Sie unterthe section called “Verwalten Sie die Durchsatzkapazität mit Auto Scaling”.

Wenn Sie eine vollständige Tabellenwiederherstellung durchführen, stammen alle Tabelleneinstellungen für die wiederhergestellte Tabelle aus den aktuellen Einstellungen der Quelltabelle zum Zeitpunkt der Wiederherstellung.

Beispiel: Angenommen, der bereitgestellte Durchsatz einer Tabelle wurde jüngst auf 50 Lese- und 50 Schreibkapazitätseinheiten verringert. Anschließend stellen Sie den Zustand der Tabelle auf den Stand vor drei Wochen wieder her. Zu diesem Zeitpunkt war der bereitgestellte Durchsatz auf 100 Lesekapazitätseinheiten und 100 Schreibkapazitätseinheiten festgelegt. In diesem Fall stellt Amazon Keyspaces Ihre Tabellendaten zu diesem Zeitpunkt wieder her, verwendet jedoch die aktuell bereitgestellten Durchsatzeinstellungen (50 Lesekapazitätseinheiten und 50 Schreibkapazitätseinheiten).

Die folgenden Einstellungen werden nicht wiederhergestellt, und Sie müssen sie manuell für die neue Tabelle konfigurieren.

- Amazon Keyspaces-Streams zur Erfassung von Änderungsdaten (CDC)
- AWS Identity and Access Management (IAM) -Richtlinien
- CloudWatch Amazon-Metriken und Alarme
- Tags (können der `RESTORE` CQL-Anweisung hinzugefügt werden mit`WITH TAGS`)

PITR-Wiederherstellung verschlüsselter Tabellen

Wenn Sie eine Tabelle mit PITR wiederherstellen, stellt Amazon Keyspaces die Verschlüsselungseinstellungen Ihrer Quelltabelle wieder her. Wenn die Tabelle mit einem AWS-eigener Schlüssel (Standard) verschlüsselt wurde, wird die Tabelle automatisch mit derselben Einstellung wiederhergestellt. Wenn die Tabelle, die Sie wiederherstellen möchten, mit einem vom Kunden verwalteten Schlüssel verschlüsselt wurde, muss Amazon Keyspaces auf denselben vom Kunden verwalteten Schlüssel zugreifen können, um die Tabellendaten wiederherzustellen.

Sie können die Verschlüsselungseinstellungen der Tabelle zum Zeitpunkt der Wiederherstellung ändern. Um von einem AWS-eigener Schlüssel zu einem vom Kunden verwalteten Schlüssel zu wechseln, müssen Sie zum Zeitpunkt der Wiederherstellung einen gültigen, vom Kunden verwalteten Schlüssel angeben, auf den zugegriffen werden kann.

Wenn Sie von einem vom Kunden verwalteten Schlüssel zu einem wechseln möchten AWS-eigener Schlüssel, stellen Sie sicher, dass Amazon Keyspaces Zugriff auf den vom Kunden verwalteten Schlüssel der Quelltabelle hat, um die Tabelle mit einem AWS-eigener Schlüssel wiederherzustellen. Weitere Informationen zu den Einstellungen für die Verschlüsselung im Ruhezustand für Tabellen finden Sie unterthe section called “Funktionsweise”.

> **Note**
>
> Wenn die Tabelle gelöscht wurde, weil Amazon Keyspaces den Zugriff auf Ihren vom Kunden verwalteten Schlüssel verloren hat, müssen Sie sicherstellen, dass Amazon Keyspaces auf den vom Kunden verwalteten Schlüssel zugreifen kann, bevor Sie versuchen, die Tabelle wiederherzustellen. Eine Tabelle, die mit einem vom Kunden verwalteten Schlüssel verschlüsselt wurde, kann nicht wiederhergestellt werden, wenn Amazon Keyspaces keinen Zugriff auf diesen Schlüssel hat. Weitere Informationen finden Sie unter Fehlerbehebung beim Schlüsselzugriff im AWS Key Management Service Entwicklerhandbuch.

PITR-Wiederherstellung von Tabellen mit mehreren Regionen

Sie können eine Tabelle mit mehreren Regionen mithilfe von PITR wiederherstellen. Damit der Wiederherstellungsvorgang erfolgreich ist, muss PITR für alle Replikate der Quelltabelle aktiviert sein und sowohl die Quell- als auch die Zieltabelle müssen in dieselbe repliziert werden. AWS-Regionen

Amazon Keyspaces stellt die Einstellungen der Quelltabelle in jeder der replizierten Regionen wieder her, die Teil des Schlüsselraums sind. Sie können die Einstellungen auch während des Wiederherstellungsvorgangs überschreiben. Weitere Informationen zu Einstellungen, die während der Wiederherstellung geändert werden können, finden Sie unterthe section called “Einstellungen wiederherstellen”.

Weitere Informationen zur Replikation in mehreren Regionen finden Sie unterthe section called “Funktionsweise”.

PITR-Wiederherstellung von Tabellen mit benutzerdefinierten Typen (UDTs)

Sie können eine Tabelle wiederherstellen, die UDTs verwendet. Damit der Wiederherstellungsvorgang erfolgreich ist, müssen die referenzierten UDTs im Schlüsselraum vorhanden und gültig sein.

Wenn eine erforderliche UDT fehlt, wenn Sie versuchen, eine Tabelle wiederherzustellen, versucht Amazon Keyspaces, das UDT-Schema automatisch wiederherzustellen, und fährt dann mit der Wiederherstellung der Tabelle fort.

Wenn Sie die UDT entfernt und neu erstellt haben, stellt Amazon Keyspaces die UDT mit dem neuen Schema der UDT wieder her und lehnt die Anforderung ab, die Tabelle mit dem ursprünglichen UDT-Schema wiederherzustellen. Wenn Sie in diesem Fall die Tabelle mit dem alten UDT-Schema wiederherstellen möchten, können Sie die Tabelle in einem neuen Schlüsselraum wiederherstellen. Wenn Sie eine UDT löschen und neu erstellen, wird die neu erstellte UDT als neue UDT betrachtet, auch wenn das Schema der neu erstellten UDT mit dem Schema der gelöschten UDT identisch ist. In diesem Fall lehnt Amazon Keyspaces die Anforderung ab, die Tabelle mit dem alten UDT-Schema wiederherzustellen.

Wenn die UDT fehlt und Amazon Keyspaces versucht, die UDT wiederherzustellen, schlägt der Versuch fehl, wenn Sie die maximale Anzahl von UDTs für das Konto in der Region erreicht haben.

Weitere Informationen zu UDT-Kontingenten und Standardwerten finden Sie unter. the section called “Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces” Weitere Informationen zum Arbeiten mit UDTs finden Sie unter. the section called “User-defined Typen (UDTs)”

Zeit für die Wiederherstellung der Tabelle mit PITR

Die Zeit, die Sie für die Wiederherstellung einer Tabelle benötigen, hängt von mehreren Faktoren ab und steht nicht immer in direktem Zusammenhang mit der Größe der Tabelle.

Im Folgenden finden Sie einige Überlegungen zu den Wiederherstellungszeiten.

- Die Sicherung wird in einer neuen Tabelle wiederhergestellt. Die Ausführung aller mit der Erstellung einer neuen Tabelle verbundenen Aktionen sowie die Initiierung des Wiederherstellungsprozesses kann bis zu 20 Minuten dauern, theoretisch selbst bei einer leeren Tabelle.
- Die Wiederherstellungszeiten für große Tabellen mit gut verteilten Datenmodellen können mehrere Stunden oder länger betragen.
- Wenn Ihre Quelltabelle Daten enthält, die stark verzerrt sind, kann sich die Wiederherstellungszeit verlängern. Wenn der Primärschlüssel Ihrer Tabelle beispielsweise den Monat des Jahres als Partitionsschlüssel verwendet und alle Ihre Daten aus dem Monat Dezember stammen, liegen schiefe Daten vor.

Eine bewährte Methode bei der Planung der Notfallwiederherstellung besteht darin, die durchschnittlichen Wiederherstellungszeiten regelmäßig zu dokumentieren und festzulegen, wie sich diese Zeiten auf Ihr gesamtes Recovery-Zeitziel auswirken.

Amazon Keyspaces PITR und Integration mit Diensten AWS

Die folgenden PITR-Operationen werden protokolliert AWS CloudTrail , um eine kontinuierliche Überwachung und Prüfung zu ermöglichen.

- Erstellen Sie eine neue Tabelle mit aktiviertem oder deaktiviertem PITR.
- Aktiviert oder deaktiviert PITR für eine bestehende Tabelle.
- Stellt eine aktive oder gelöschte Tabelle wieder her.

Weitere Informationen finden Sie unter Protokollieren von Amazon Keyspaces-API-Aufrufen mit AWS CloudTrail.

Sie können die folgenden PITR-Aktionen ausführen mit CloudFormation.

- Erstellen Sie eine neue Tabelle mit aktiviertem oder deaktiviertem PITR.
- Aktiviert oder deaktiviert PITR für eine bestehende Tabelle.

Weitere Informationen finden Sie in der Cassandra Resource Type Reference im CloudFormation Benutzerhandbuch.

Verwenden Sie Point-in-Time-Wiederherstellung in Amazon Keyspaces

Mit Amazon Keyspaces (für Apache Cassandra) können Sie Tabellen mithilfe von Restore (PITR) zu einem bestimmten Zeitpunkt Point-in-Time wiederherstellen. Mit PITR können Sie eine Tabelle auf einen früheren Status innerhalb der letzten 35 Tage zurücksetzen und bietet so Datenschutz- und Wiederherstellungsfunktionen. Diese Funktion ist in Fällen wie versehentlichem Löschen von Daten, Anwendungsfehlern oder zu Testzwecken nützlich. Sie können Daten schnell und effizient wiederherstellen und so Ausfallzeiten und Datenverlust minimieren. Die folgenden Abschnitte führen Sie durch den Prozess der Wiederherstellung von Tabellen mithilfe von PITR in Amazon Keyspaces, um Datenintegrität und Geschäftskontinuität sicherzustellen.

Themen

- Konfigurieren Sie die IAM-Berechtigungen zum Wiederherstellen von Tabellen für Amazon Keyspaces PITR

- PITR für eine Tabelle in Amazon Keyspaces konfigurieren
- PITR für eine Amazon Keyspaces-Tabelle ausschalten
- Eine Tabelle aus dem Backup zu einem bestimmten Zeitpunkt in Amazon Keyspaces wiederherstellen
- Eine gelöschte Tabelle mit Amazon Keyspaces PITR wiederherstellen

Konfigurieren Sie die IAM-Berechtigungen zum Wiederherstellen von Tabellen für Amazon Keyspaces PITR

In diesem Abschnitt wird zusammengefasst, wie Berechtigungen für einen AWS Identity and Access Management (IAM-) Principal zur Wiederherstellung von Amazon Keyspaces-Tabellen konfiguriert werden. In IAM `AmazonKeyspacesFullAccess` umfasst die AWS verwaltete Richtlinie die Berechtigungen zur Wiederherstellung von Amazon Keyspaces-Tabellen. Beachten Sie die im nächsten Abschnitt beschriebenen Anforderungen, um eine benutzerdefinierte Richtlinie mit den erforderlichen Mindestberechtigungen zu implementieren.

Um eine Tabelle erfolgreich wiederherzustellen, benötigt der IAM-Prinzipal die folgenden Mindestberechtigungen:

- `cassandra:Restore`— Die Wiederherstellungsaktion ist erforderlich, damit die Zieltabelle wiederhergestellt werden kann.
- `cassandra:Select`— Die Select-Aktion ist erforderlich, um aus der Quelltabelle zu lesen.
- `cassandra:TagResource`— Die Tag-Aktion ist optional und nur erforderlich, wenn beim Wiederherstellungsvorgang Tags hinzugefügt werden.

Dies ist ein Beispiel für eine Richtlinie, die einem Benutzer die erforderlichen Mindestberechtigungen zum Wiederherstellen von Tabellen im Schlüsselraum `mykeyspace` gewährt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Restore",
            "cassandra:Select"
         ],
```

```
            "Resource":[
               "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/*",
               "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
            ]
         }
      ]
}
```

Je nach anderen ausgewählten Funktionen sind möglicherweise zusätzliche Berechtigungen zum Wiederherstellen einer Tabelle erforderlich. Wenn die Quelltabelle beispielsweise im Ruhezustand mit einem vom Kunden verwalteten Schlüssel verschlüsselt ist, muss Amazon Keyspaces über Berechtigungen für den Zugriff auf den vom Kunden verwalteten Schlüssel der Quelltabelle verfügen, um die Tabelle erfolgreich wiederherzustellen. Weitere Informationen finden Sie unter the section called “PITR und verschlüsselte Tabellen”.

Wenn Sie IAM-Richtlinien mit Bedingungsschlüsseln verwenden, um den eingehenden Datenverkehr auf bestimmte Quellen zu beschränken, müssen Sie sicherstellen, dass Amazon Keyspaces berechtigt ist, im Namen Ihres Prinzipals einen Wiederherstellungsvorgang durchzuführen. Sie müssen Ihrer IAM-Richtlinie einen `aws:ViaAWSService` Bedingungsschlüssel hinzufügen, wenn Ihre Richtlinie den eingehenden Datenverkehr auf einen der folgenden Bereiche beschränkt:

- VPC-Endpunkte mit `aws:SourceVpce`
- IP-Bereiche mit `aws:SourceIp`
- VPCs mit `aws:SourceVpc`

Der `aws:ViaAWSService` Bedingungsschlüssel ermöglicht den Zugriff, wenn ein AWS Dienst eine Anfrage mit den Anmeldeinformationen des Prinzipals stellt. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingungsschlüssel im IAM-Benutzerhandbuch.

Im Folgenden finden Sie ein Beispiel für eine Richtlinie, die den Quellverkehr auf eine bestimmte IP-Adresse beschränkt und es Amazon Keyspaces ermöglicht, eine Tabelle im Namen des Prinzipals wiederherzustellen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"CassandraAccessForCustomIp",
         "Effect":"Allow",
```

```
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"false"
            },
            "ForAnyValue:IpAddress":{
               "aws:SourceIp":[
                  "123.45.167.89"
               ]
            }
         }
      },
      {
         "Sid":"CassandraAccessForAwsService",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"true"
            }
         }
      }
   ]
}
```

Ein Beispiel für eine Richtlinie, die den `aws:ViaAWSService` globalen Bedingungsschlüssel verwendet, finden Sie unter. the section called “VPC-Endpunktrichtlinien und Point-in-Time-Recovery (PITR) von Amazon Keyspaces”

PITR für eine Tabelle in Amazon Keyspaces konfigurieren

Sie können eine Tabelle in Amazon Keyspaces für Sicherungs- und Wiederherstellungsvorgänge mithilfe von PITR mit der Konsole, CQL und dem konfigurieren. AWS CLI

Wenn Sie eine neue Tabelle mit CQL oder dem erstellen AWS CLI, müssen Sie PITR in der Anweisung create table explizit aktivieren. Wenn Sie mit der Konsole eine neue Tabelle erstellen, ist PITR standardmäßig aktiviert.

Informationen zum Wiederherstellen einer Tabelle finden Sie unterthe section called “Stellen Sie eine Tabelle auf einen bestimmten Zeitpunkt wieder her”.

Console

Konfigurieren Sie PITR für eine Tabelle mithilfe der Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tabellen und dann die Tabelle aus, die Sie bearbeiten möchten.
3. Wählen Sie auf der Registerkarte Backups die Option Bearbeiten aus.
4. Wählen Sie im Abschnitt Einstellungen für die Point-in-Time-Wiederherstellung bearbeiten die Option Wiederherstellung aktivieren Point-in-time aus.
5. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Konfigurieren Sie PITR für eine Tabelle mithilfe von CQL

1. Sie können die PITR-Einstellungen für Tabellen mithilfe der `point_in_time_recovery` benutzerdefinierten Eigenschaft verwalten.

   Um PITR beim Erstellen einer neuen Tabelle zu aktivieren, müssen Sie den Status auf setzen. `point_in_time_recovery enabled` Sie können den folgenden CQL-Befehl als Beispiel verwenden.

```
CREATE TABLE "my_keyspace1"."my_table1"(
 "id" int,
 "name" ascii,
 "date" timestamp,
 PRIMARY KEY("id"))
WITH CUSTOM_PROPERTIES = {
 'capacity_mode':{'throughput_mode':'PAY_PER_REQUEST'},
 'point_in_time_recovery':{'status':'enabled'}
}
```

> **Note**
>
> Wenn keine benutzerdefinierte Eigenschaft für die Point-in-Time-Wiederherstellung angegeben ist, ist die Point-in-Time-Wiederherstellung standardmäßig deaktiviert.

2. Führen Sie den folgenden CQL-Befehl aus, um PITR für eine vorhandene Tabelle mithilfe von CQL zu aktivieren.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'enabled'}}
```

CLI

Konfigurieren Sie PITR für eine Tabelle mit dem AWS CLI

1. Sie können die PITR-Einstellungen für Tabellen mithilfe der `UpdateTable` API verwalten.

   Um PITR beim Erstellen einer neuen Tabelle zu aktivieren, müssen Sie den Befehl create table `point-in-time-recovery 'status=ENABLED'` einbeziehen. Sie können den folgenden AWS CLI Befehl als Beispiel verwenden. Der Befehl wurde in separate Zeilen aufgeteilt, um die Lesbarkeit zu verbessern.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},
{name=name,type=text},{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --point-in-time-recovery 'status=ENABLED'
```

   > **Note**
   >
   > Wenn kein Wert für die Point-in-Time-Wiederherstellung angegeben ist, ist die Point-in-Time-Wiederherstellung standardmäßig deaktiviert.

2. Um die Einstellung für die Point-in-Time-Wiederherstellung für eine Tabelle zu bestätigen, können Sie den folgenden Befehl verwenden. AWS CLI

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

3. Führen Sie den folgenden Befehl aus, um PITR für eine vorhandene Tabelle mithilfe von zu aktivieren. AWS CLI

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
 --point-in-time-recovery 'status=ENABLED'
```

PITR für eine Amazon Keyspaces-Tabelle ausschalten

Sie können PITR für eine Amazon Keyspaces-Tabelle jederzeit über die Konsole, CQL oder die ausschalten. AWS CLI

> **Important**
>
> Durch die Deaktivierung von PITR wird Ihr Backup-Verlauf sofort gelöscht, auch wenn Sie PITR in der Tabelle innerhalb von 35 Tagen wieder aktivieren.

Informationen zum Wiederherstellen einer Tabelle finden Sie unter. the section called “Stellen Sie eine Tabelle auf einen bestimmten Zeitpunkt wieder her”

Console

Deaktivieren Sie PITR für eine Tabelle mithilfe der Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tabellen und dann die Tabelle aus, die Sie bearbeiten möchten.
3. Wählen Sie auf der Registerkarte Backups die Option Bearbeiten aus.
4. Deaktivieren Sie im Abschnitt Einstellungen für die Point-in-Time-Wiederherstellung bearbeiten das Kontrollkästchen Point-in-time Wiederherstellung aktivieren.
5. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Deaktivieren Sie PITR für eine Tabelle mithilfe von CQL

- Um PITR für eine bestehende Tabelle zu deaktivieren, führen Sie den folgenden CQL-Befehl aus.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

CLI

Deaktivieren Sie PITR für eine Tabelle mit dem AWS CLI

- Führen Sie den folgenden AWS CLI Befehl aus, um PITR für eine vorhandene Tabelle zu deaktivieren.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
 --point-in-time-recovery 'status=DISABLED'
```

Eine Tabelle aus dem Backup zu einem bestimmten Zeitpunkt in Amazon Keyspaces wiederherstellen

Der folgende Abschnitt zeigt, wie Sie eine bestehende Amazon Keyspaces-Tabelle zu einem bestimmten Zeitpunkt wiederherstellen.

> **Note**
>
> Bei diesem Verfahren wird davon ausgegangen, dass die von Ihnen verwendete Tabelle mit einer Point-in-Time-Wiederherstellung konfiguriert wurde. Informationen zum Aktivieren von PITR für eine Tabelle finden Sie unter. the section called “PITR konfigurieren”

> **Important**
>
> Ändern oder löschen Sie während einer Wiederherstellung nicht die AWS Identity and Access Management (IAM-) Richtlinien, die dem IAM-Prinzipal (z. B. Benutzer, Gruppe oder Rolle) die Berechtigung zur Durchführung der Wiederherstellung gewähren. Dies könnte andernfalls zu unerwartetem Verhalten führen. Wenn Sie beispielsweise Schreibberechtigungen für eine Tabelle entfernen, während diese Tabelle wiederhergestellt wird, kann der zugrunde liegende `RestoreTableToPointInTime` Vorgang keine der wiederhergestellten Daten in die Tabelle schreiben.
> Sie können Berechtigungen erst ändern oder löschen, nachdem der Wiederherstellungsvorgang abgeschlossen ist.

Console

Stellen Sie eine Tabelle mithilfe der Konsole auf einen bestimmten Zeitpunkt wieder her

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Klicken Sie im Navigationsbereich auf der linken Seite der Konsole auf Tabellen.
3. Wählen Sie in der Tabellenliste die Tabelle aus, die Sie wiederherstellen möchten.
4. Wählen Sie auf der Registerkarte Backups der Tabelle im Bereich Point-in-time Wiederherstellung die Option Wiederherstellen aus.
5. Geben Sie für den neuen Tabellennamen beispielsweise einen neuen Namen für die wiederhergestellte Tabelle ein**`mytable_restored`**.
6. Um den Zeitpunkt für den Wiederherstellungsvorgang zu definieren, können Sie zwischen zwei Optionen wählen:
   - Wählen Sie den vorkonfigurierten frühesten Zeitpunkt aus.
   - Wählen Sie Datum und Uhrzeit angeben und geben Sie das Datum und die Uhrzeit ein, zu dem Sie die neue Tabelle wiederherstellen möchten.

   > **Note**
   >
   > Sie können zu jedem beliebigen Zeitpunkt innerhalb der frühesten Zeit und der aktuellen Uhrzeit wiederherstellen. Amazon Keyspaces stellt Ihre Tabellendaten auf den Status zurück, der auf dem ausgewählten Datum und der ausgewählten Uhrzeit (Tag:Stunde:Minute:Sekunde) basiert.
7. Wählen Sie Wiederherstellen, um den Wiederherstellungsvorgang zu starten.

   Die Tabelle, die wiederhergestellt wird, erhält den Status Restoring. Nach Abschluss des Wiederherstellungsvorgangs ändert sich der Status der wiederhergestellten Tabelle in Aktiv.

Cassandra Query Language (CQL)

Stellen Sie eine Tabelle mithilfe von CQL auf einen bestimmten Zeitpunkt wieder her

1. Sie können eine aktive Tabelle auf einen Zeitpunkt zwischen `earliest_restorable_timestamp` und der aktuellen Uhrzeit wiederherstellen. Die aktuelle Uhrzeit ist die Standardeinstellung.

Um zu überprüfen, ob die Point-in-Time-Wiederherstellung für die Tabelle aktiviert ist, fragen Sie die ab, `system_schema_mcs.tables` wie in diesem Beispiel gezeigt.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Point-in-time Die Wiederherstellung ist aktiviert, wie in der folgenden Beispielausgabe gezeigt.

```
custom_properties
-----------------
{
  ...,
    "point_in_time_recovery": {
    "earliest_restorable_timestamp":"2020-06-30T19:19:21.175Z"
    "status":"enabled"
  }
}
```

2.
   - Stellt die aktuelle Uhrzeit der Tabelle wieder her. Wenn Sie die `WITH restore_timestamp = ...` Klausel weglassen, wird der aktuelle Zeitstempel verwendet.

     ```
     RESTORE TABLE mykeyspace.mytable_restored
     FROM TABLE mykeyspace.mytable;
     ```

   - Sie können auch zu einem bestimmten Zeitpunkt wiederherstellen, der durch einen `restore_timestamp` im ISO 8601-Format definiert ist. Sie können einen beliebigen Zeitpunkt innerhalb der vergangenen 35 Tage angeben. Beispielsweise wird mit dem folgendem Befehl die Tabelle auf den Zeitpunkt `EarliestRestorableDateTime` widerhergestellt.

     ```
     RESTORE TABLE mykeyspace.mytable_restored
     FROM TABLE mykeyspace.mytable
     WITH restore_timestamp = '2020-06-30T19:19:21.175Z';
     ```

     Eine vollständige Syntaxbeschreibung finden Sie the section called “TABELLE WIEDERHERSTELLEN” in der Sprachreferenz.

3. Um zu überprüfen, ob die Wiederherstellung der Tabelle erfolgreich war, fragen Sie die ab, `system_schema_mcs.tables` um den Status der Tabelle zu bestätigen.

```
SELECT status
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable_restored'
```

Die Abfrage zeigt die folgende Ausgabe.

```
status
------
RESTORING
```

Die Tabelle, die wiederhergestellt wird, erhält den Status Restoring. Nach Abschluss des Wiederherstellungsvorgangs ändert sich der Status der Tabelle in Aktiv.

CLI

Stellen Sie eine Tabelle auf einen bestimmten Zeitpunkt wieder her, indem Sie AWS CLI

1. Erstellen Sie eine einfache Tabelle mit dem Namen`myTable`, für die PITR aktiviert ist. Der Befehl wurde aus Gründen der Lesbarkeit in separate Zeilen aufgeteilt.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},
{name=name,type=text},{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --point-in-time-recovery 'status=ENABLED'
```

2. Bestätigen Sie die Eigenschaften der neuen Tabelle und überprüfen Sie die `earliestRestorableTimestamp` PITR.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Die Ausgabe dieses Befehls gibt Folgendes zurück.

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
```

```
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
    "creationTimestamp": "2022-06-20T14:34:57.049000-07:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2022-06-20T14:34:57.049000-07:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "ENABLED",
        "earliestRestorableTimestamp": "2022-06-20T14:35:13.693000-07:00"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    }
}
```

3. • Um eine Tabelle auf einen bestimmten Zeitpunkt zurückzusetzen, geben Sie a `restore_timestamp` im ISO 8601-Format an. Sie können einen beliebigen Zeitpunkt der letzten 35 Tage in Intervallen von einer Sekunde wählen. Beispielsweise wird mit dem folgendem Befehl die Tabelle auf den Zeitpunkt `EarliestRestorableDateTime` widerhergestellt.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored' --restore-timestamp "2022-06-20 21:35:14.693"
```

Die Ausgabe dieses Befehls gibt den ARN der wiederhergestellten Tabelle zurück.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable_restored"
}
```

• Um die Tabelle auf die aktuelle Uhrzeit zurückzusetzen, können Sie den `restore-timestamp` Parameter weglassen.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored1'"
```

Eine gelöschte Tabelle mit Amazon Keyspaces PITR wiederherstellen

Das folgende Verfahren zeigt, wie eine gelöschte Tabelle aus dem Backup bis zum Zeitpunkt des Löschvorgangs wiederhergestellt wird. Sie können dies mit CQL oder dem AWS CLI tun.

> **Note**
>
> Bei diesem Verfahren wird davon ausgegangen, dass PITR für die gelöschte Tabelle aktiviert wurde.

Cassandra Query Language (CQL)

Stellen Sie eine gelöschte Tabelle mit CQL wieder her

1. Um zu überprüfen, ob die Point-in-Time-Wiederherstellung für eine gelöschte Tabelle aktiviert ist, fragen Sie die Systemtabelle ab. Es werden nur Tabellen angezeigt, für die Point-in-Time-Recovery aktiviert ist.

```
SELECT custom_properties
FROM system_schema_mcs.tables_history
WHERE keyspace_name = 'mykeyspace' AND table_name = 'my_table';
```

   Die Abfrage zeigt die folgende Ausgabe.

```
custom_properties
-----------------
{
    ...,
   "point_in_time_recovery":{
      "restorable_until_time":"2020-08-04T00:48:58.381Z",
      "status":"enabled"
   }
}
```

2. Stellen Sie die Tabelle mit der folgenden Beispielanweisung auf den Zeitpunkt des Löschvorgangs zurück.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

CLI

Stellen Sie eine gelöschte Tabelle mit dem wieder her AWS CLI

1. Löschen Sie eine Tabelle, die Sie zuvor erstellt haben und für die PITR aktiviert ist. Nachfolgend finden Sie einen Beispielbefehl.

```
aws keyspaces delete-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

2. Stellen Sie die gelöschte Tabelle mit dem folgenden Befehl auf den Zeitpunkt des Löschens zurück.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored2'
```

Die Ausgabe dieses Befehls gibt den ARN der wiederhergestellten Tabelle zurück.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable_restored2"
}
```

Daten mit Time to Live (TTL) für Amazon Keyspaces (für Apache Cassandra) ablaufen lassen

Amazon Keyspaces (für Apache Cassandra) Time to Live (TTL) hilft Ihnen, Ihre Anwendungslogik zu vereinfachen und den Speicherpreis zu optimieren, indem Daten aus Tabellen automatisch ablaufen. Daten, die Sie nicht mehr benötigen, werden basierend auf dem von Ihnen festgelegten Time-to-Live-Wert automatisch aus Ihrer Tabelle gelöscht.

Dies erleichtert die Einhaltung von Richtlinien zur Datenspeicherung, die auf Geschäfts-, Branchen- oder behördlichen Anforderungen basieren, die festlegen, wie lange Daten aufbewahrt werden müssen oder wann Daten gelöscht werden müssen.

Sie können TTL beispielsweise in einer AdTech Anwendung verwenden, um zu planen, wann Daten für bestimmte Anzeigen ablaufen und für Kunden nicht mehr sichtbar sind. Sie können TTL auch verwenden, um ältere Daten automatisch zu löschen und so Speicherkosten zu sparen.

Sie können einen Standard-TTL-Wert für die gesamte Tabelle festlegen und diesen Wert für einzelne Zeilen und Spalten überschreiben. TTL-Operationen wirken sich nicht auf die Leistung Ihrer Anwendung aus. Außerdem wirkt sich die Anzahl der Zeilen und Spalten, die mit TTL als ablaufend markiert sind, nicht auf die Verfügbarkeit Ihrer Tabelle aus.

Amazon Keyspaces filtert automatisch abgelaufene Daten heraus, sodass abgelaufene Daten nicht in Abfrageergebnissen zurückgegeben werden oder für die Verwendung in DML-Anweisungen (Data

Manipulation Language) verfügbar sind. Amazon Keyspaces löscht abgelaufene Daten in der Regel innerhalb von 10 Tagen nach dem Ablaufdatum aus dem Speicher.

In seltenen Fällen ist Amazon Keyspaces möglicherweise nicht in der Lage, Daten innerhalb von 10 Tagen zu löschen, wenn auf der zugrunde liegenden Speicherpartition zum Schutz der Verfügbarkeit anhaltende Aktivitäten stattfinden. In diesen Fällen versucht Amazon Keyspaces weiterhin, die abgelaufenen Daten zu löschen, sobald der Verkehr auf der Partition abnimmt.

Nachdem die Daten dauerhaft aus dem Speicher gelöscht wurden, fallen keine Speichergebühren mehr an.

Sie können Standard-TTL-Einstellungen für neue und bestehende Tabellen mithilfe der Konsole, Cassandra Query Language (CQL) oder der festlegen, ändern oder deaktivieren. AWS CLI

Bei Tabellen mit konfigurierter TTL-Standardeinstellung können Sie CQL-Anweisungen verwenden, um die TTL-Standardeinstellungen der Tabelle zu überschreiben und benutzerdefinierte TTL-Werte auf Zeilen und Spalten anzuwenden. Weitere Informationen erhalten Sie unter the section called “Wird verwendet`INSERT`, um benutzerdefinierte TTL für neue Zeilen festzulegen” und the section called “Wird verwendet`UPDATE`, um benutzerdefinierte TTL für Zeilen und Spalten festzulegen”.

Die TTL-Preisgestaltung basiert auf der Größe der Zeilen, die mithilfe von Time to Live gelöscht oder aktualisiert werden. TTL-Operationen werden in Einheiten von gemessen. `TTL deletes` Pro KB Daten pro Zeile, die gelöscht oder aktualisiert wird, wird ein TTL-Löschvorgang verbraucht.

Um beispielsweise eine Zeile zu aktualisieren, die 2,5 KB an Daten speichert, und um eine oder mehrere Spalten innerhalb der Zeile gleichzeitig zu löschen, sind drei TTL-Löschungen erforderlich. Oder um eine ganze Zeile zu löschen, die 3,5 KB an Daten enthält, sind vier TTL-Löschungen erforderlich.

Pro KB gelöschter Daten pro Zeile wird ein TTL-Löschvorgang verbraucht. Weitere Informationen zu den Preisen finden Sie unter Amazon Keyspaces (für Apache Cassandra) — Preise.

Themen

- Amazon Keyspaces Time to Live und Integration mit Services AWS
- Erstellen Sie eine neue Tabelle mit den Standardeinstellungen für Time to Live (TTL)
- Aktualisieren Sie den Standardwert Time to Live (TTL) einer Tabelle
- Erstellen Sie eine Tabelle mit aktivierten benutzerdefinierten Time to Live (TTL) -Einstellungen
- Tabelle mit benutzerdefinierter Gültigkeitsdauer (TTL) aktualisieren

- Verwenden Sie die INSERT Anweisung, um benutzerdefinierte TTL-Werte (Time to Live) für neue Zeilen festzulegen
- Verwenden Sie die UPDATE Anweisung, um benutzerdefinierte TTL-Einstellungen (Time to Live) für Zeilen und Spalten zu bearbeiten

Amazon Keyspaces Time to Live und Integration mit Services AWS

Die folgende TTL-Metrik ist in Amazon verfügbar CloudWatch , um eine kontinuierliche Überwachung zu ermöglichen.

- `TTLDeletes`— Die Einheiten, die zum Löschen oder Aktualisieren von Daten nacheinander mithilfe von Time to Live (TTL) verbraucht wurden.

Weitere Informationen zur Überwachung von CloudWatch Metriken finden Sie unterthe section called “Überwachung mit CloudWatch”.

Wenn Sie verwenden CloudFormation, können Sie TTL aktivieren, wenn Sie eine Amazon Keyspaces-Tabelle erstellen. Weitere Informationen finden Sie im AWS CloudFormation - Benutzerhandbuch.

Erstellen Sie eine neue Tabelle mit den Standardeinstellungen für Time to Live (TTL)

In Amazon Keyspaces können Sie bei der Erstellung der Tabelle einen Standard-TTL-Wert für alle Zeilen in einer Tabelle festlegen.

Der Standard-TTL-Wert für eine Tabelle ist Null, was bedeutet, dass Daten nicht automatisch ablaufen. Wenn der Standard-TTL-Wert für eine Tabelle größer als Null ist, wird jeder Zeile ein Ablaufzeitstempel hinzugefügt.

TTL-Werte werden in Sekunden angegeben, und der konfigurierbare Höchstwert beträgt 630.720.000 Sekunden, was 20 Jahren entspricht.

Nach der Tabellenerstellung können Sie die Standard-TTL-Einstellung der Tabelle für bestimmte Zeilen oder Spalten mit CQL-DML-Anweisungen überschreiben. Weitere Informationen erhalten Sie unter the section called “Wird verwendet`INSERT`, um benutzerdefinierte TTL für neue Zeilen festzulegen” und the section called “Wird verwendet`UPDATE`, um benutzerdefinierte TTL für Zeilen und Spalten festzulegen”.

Wenn Sie TTL für eine Tabelle aktivieren, beginnt Amazon Keyspaces, zusätzliche TTL-related Metadaten für jede Zeile zu speichern. Darüber hinaus verwendet TTL Ablaufzeitstempel, um nachzuverfolgen, wann Zeilen oder Spalten ablaufen. Die Zeitstempel werden als Zeilenmetadaten gespeichert und tragen zu den Speicherkosten für die Zeile bei.

Nachdem die TTL-Funktion aktiviert wurde, können Sie sie für eine Tabelle nicht mehr deaktivieren. Wenn Sie die Tabelle `default_time_to_live` auf 0 setzen, werden die Standardablaufzeiten für neue Daten deaktiviert, aber dadurch wird weder die TTL-Funktion deaktiviert noch die Tabelle auf die ursprünglichen Amazon Keyspaces-Speichermetadaten oder das Schreibverhalten zurückgesetzt.

Die folgenden Beispiele zeigen, wie eine neue Tabelle mit einem Standard-TTL-Wert erstellt wird.

Console

Erstellen Sie mithilfe der Konsole eine neue Tabelle mit einem Time to Live-Standardwert.

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) und anschließend Create table (Tabelle erstellen) aus.
3. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die neue Tabelle ein.
4. Erstellen Sie im Abschnitt Schema das Schema für Ihre Tabelle.
5. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.
6. Fahren Sie mit Time to Live (TTL) fort.

   In diesem Schritt wählen Sie die TTL-Standardeinstellungen für die Tabelle aus.

   Geben Sie für den Standard-TTL-Periode die Ablaufzeit ein und wählen Sie die Zeiteinheit aus, die Sie eingegeben haben, zum Beispiel Sekunden, Tage oder Jahre. Amazon Keyspaces speichert den Wert in Sekunden.
7. Wählen Sie Create table (Tabelle erstellen) aus. Ihre Tabelle wird mit dem angegebenen Standard-TTL-Wert erstellt.

Cassandra Query Language (CQL)

Erstellen Sie mithilfe von CQL eine neue Tabelle mit einem Standard-TTL-Wert

1. Die folgende Anweisung erstellt eine neue Tabelle, wobei der TTL-Standardwert auf 3.024.000 Sekunden festgelegt ist, was 35 Tagen entspricht.

```
CREATE TABLE my_table (
                userid uuid,
                time timeuuid,
                subject text,
                body text,
                user inet,
                PRIMARY KEY (userid, time)
                ) WITH default_time_to_live = 3024000;
```

2. Verwenden Sie die `cqlsh DESCRIBE` Anweisung, um die TTL-Einstellungen für die neue Tabelle zu bestätigen, wie im folgenden Beispiel gezeigt. Die Ausgabe zeigt die Standard-TTL-Einstellung für die Tabelle als. `default_time_to_live`

```
DESC TABLE my_table;
```

```
CREATE TABLE my_keyspace.my_table (
    userid uuid,
    time timeuuid,
    body text,
    subject text,
    user inet,
    PRIMARY KEY (userid, time)
) WITH CLUSTERING ORDER BY (time ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'class': 'com.amazonaws.cassandra.DefaultCaching'}
    AND comment = ''
    AND compaction = {'class': 'com.amazonaws.cassandra.DefaultCompaction'}
    AND compression = {'class': 'com.amazonaws.cassandra.DefaultCompression'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.0
    AND default_time_to_live = 3024000
    AND gc_grace_seconds = 7776000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 3600000
    AND min_index_interval = 128
```

```
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';
```

CLI

Erstellen Sie eine neue Tabelle mit einem Standard-TTL-Wert mit dem AWS CLI

1. Sie können den folgenden Befehl verwenden, um eine neue Tabelle zu erstellen, bei der der TTL-Standardwert auf ein Jahr festgelegt ist.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable' \
            --schema-definition 'allColumns=[{name=id,type=int},
{name=name,type=text},{name=date,type=timestamp}],partitionKeys=[{name=id}]' \
            --default-time-to-live '31536000'
```

2. Um den TTL-Status der Tabelle zu bestätigen, können Sie den folgenden Befehl verwenden.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Die Ausgabe des Befehls sieht wie im folgenden Beispiel aus

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
    "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
```

```
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-09-02T10:52:22.190000+00:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 31536000,
    "comment": {
        "message": ""
    },
    "replicaSpecifications": []
}
```

Aktualisieren Sie den Standardwert Time to Live (TTL) einer Tabelle

Sie können eine bestehende Tabelle mit einem neuen Standard-TTL-Wert aktualisieren. TTL-Werte werden in Sekunden festgelegt, und der konfigurierbare Höchstwert beträgt 630.720.000 Sekunden, was 20 Jahren entspricht.

Wenn Sie TTL für eine Tabelle aktivieren, beginnt Amazon Keyspaces, zusätzliche TTL-related Metadaten für jede Zeile zu speichern. Darüber hinaus verwendet TTL Ablaufzeitstempel, um nachzuverfolgen, wann Zeilen oder Spalten ablaufen. Die Zeitstempel werden als Zeilenmetadaten gespeichert und tragen zu den Speicherkosten für die Zeile bei.

Nachdem TTL für eine Tabelle aktiviert wurde, können Sie die Standard-TTL-Einstellung der Tabelle für bestimmte Zeilen oder Spalten mit CQL-DML-Anweisungen überschreiben. Weitere Informationen erhalten Sie unter the section called “Wird verwendet`INSERT`, um benutzerdefinierte TTL für neue Zeilen festzulegen” und the section called “Wird verwendet`UPDATE`, um benutzerdefinierte TTL für Zeilen und Spalten festzulegen”.

Nachdem die TTL-Funktion aktiviert wurde, können Sie sie für eine Tabelle nicht mehr deaktivieren. Wenn Sie die Tabelle `default_time_to_live` auf 0 setzen, werden die Standardablaufzeiten für neue Daten deaktiviert, aber dadurch wird weder die TTL-Funktion deaktiviert noch die Tabelle auf die ursprünglichen Amazon Keyspaces-Speichermetadaten oder das Schreibverhalten zurückgesetzt.

Gehen Sie wie folgt vor, um die Standardeinstellungen für Time to Live für bestehende Tabellen mithilfe der Konsole, CQL oder der zu aktualisieren. AWS CLI

Console

Aktualisieren Sie den TTL-Standardwert einer Tabelle mithilfe der Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie die Tabelle aus, die Sie aktualisieren möchten, und wählen Sie dann den Tab Zusätzliche Einstellungen.
3. Gehen Sie weiter zu Time to Live (TTL) und wählen Sie Bearbeiten.
4. Geben Sie für den Standard-TTL-Zeitraum die Ablaufzeit ein und wählen Sie die Zeiteinheit aus, z. B. Sekunden, Tage oder Jahre. Amazon Keyspaces speichert den Wert in Sekunden. Der TTL-Wert vorhandener Zeilen wird dadurch nicht geändert.
5. Wenn die TTL-Einstellungen definiert sind, wählen Sie Änderungen speichern.

Cassandra Query Language (CQL)

Aktualisieren Sie den Standard-TTL-Wert einer Tabelle mithilfe von CQL

1. Sie können `ALTER TABLE` es verwenden, um die Standardeinstellungen für Time to Live (TTL) einer Tabelle zu bearbeiten. Um die TTL-Standardeinstellungen der Tabelle auf 2.592.000 Sekunden zu aktualisieren, was 30 Tagen entspricht, können Sie die folgende Anweisung verwenden.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

2. Verwenden Sie die `cqlsh DESCRIBE` Anweisung, um die TTL-Einstellungen für die aktualisierte Tabelle zu bestätigen, wie im folgenden Beispiel gezeigt. Die Ausgabe zeigt die Standard-TTL-Einstellung für die Tabelle als. `default_time_to_live`

```
DESC TABLE my_table;
```

Die Ausgabe der Anweisung sollte diesem Beispiel ähneln.

```
CREATE TABLE my_keyspace.my_table (
    id int PRIMARY KEY,
    date timestamp,
    name text
) WITH bloom_filter_fp_chance = 0.01
    AND caching = {'class': 'com.amazonaws.cassandra.DefaultCaching'}
    AND comment = ''
    AND compaction = {'class': 'com.amazonaws.cassandra.DefaultCompaction'}
    AND compression = {'class': 'com.amazonaws.cassandra.DefaultCompression'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.0
    AND default_time_to_live = 2592000
    AND gc_grace_seconds = 7776000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 3600000
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';
```

CLI

Aktualisieren Sie den Standard-TTL-Wert einer Tabelle mit dem AWS CLI

1. Sie können `update-table` den Standard-TTL-Wert einer Tabelle bearbeiten. Um die TTL-Standardeinstellungen der Tabelle auf 2.592.000 Sekunden zu aktualisieren, was 30 Tagen entspricht, können Sie die folgende Anweisung verwenden.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
 --default-time-to-live '2592000'
```

2. Um den aktualisierten TTL-Standardwert zu bestätigen, können Sie die folgende Anweisung verwenden.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Die Ausgabe der Anweisung sollte wie im folgenden Beispiel aussehen.

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
    "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-09-02T10:52:22.190000+00:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
```

```
        "status": "DISABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 2592000,
    "comment": {
        "message": ""
    },
    "replicaSpecifications": []
}
```

Erstellen Sie eine Tabelle mit aktivierten benutzerdefinierten Time to Live (TTL) -Einstellungen

Um eine neue Tabelle mit benutzerdefinierten Time to Live-Einstellungen zu erstellen, die auf Zeilen und Spalten angewendet werden können, ohne die TTL-Standardeinstellungen für die gesamte Tabelle zu aktivieren, können Sie die folgenden Befehle verwenden.

> **Note**
>
> Wenn eine Tabelle mit aktivierten `ttl` benutzerdefinierten Einstellungen erstellt wird, können Sie die Einstellung später nicht mehr deaktivieren.

Cassandra Query Language (CQL)

Erstellen Sie mithilfe von CQL eine neue Tabelle mit benutzerdefinierter TTL-Einstellung

- ```
  CREATE TABLE my_keyspace.my_table (id int primary key) WITH
   CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
  ```

CLI

Erstellen Sie eine neue Tabelle mit benutzerdefinierter TTL-Einstellung mit dem AWS CLI

1. Sie können den folgenden Befehl verwenden, um eine neue Tabelle mit aktivierter TTL zu erstellen.
  ```

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable' \
                                --schema-definition
 'allColumns=[{name=id,type=int},{name=name,type=text},
 {name=date,type=timestamp}],partitionKeys=[{name=id}]' \
                                --ttl 'status=ENABLED'
```

2. Um zu bestätigen, dass TTL für die Tabelle aktiviert ist, können Sie die folgende Anweisung verwenden.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Die Ausgabe der Anweisung sollte wie im folgenden Beispiel aussehen.

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
    "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
         {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
            }
        ],
        "clusteringKeys": [],
```

```
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-09-02T11:18:55.796000+00:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    },
    "replicaSpecifications": []
}
```

Tabelle mit benutzerdefinierter Gültigkeitsdauer (TTL) aktualisieren

Um die benutzerdefinierten Time to Live-Einstellungen für eine Tabelle zu aktivieren, sodass TTL-Werte auf einzelne Zeilen und Spalten angewendet werden können, ohne einen TTL-Standardwert für die gesamte Tabelle festzulegen, können Sie die folgenden Befehle verwenden.

> **Note**
>
> After `ttl` ist aktiviert, Sie können es nicht für die Tabelle deaktivieren.

Cassandra Query Language (CQL)

Aktivieren Sie benutzerdefinierte TTL-Einstellungen für eine Tabelle mithilfe von CQL

- ```
  ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
  ```
  ```

CLI

Aktivieren Sie benutzerdefinierte TTL-Einstellungen für eine Tabelle mit dem AWS CLI

1. Sie können den folgenden Befehl verwenden, um die benutzerdefinierte TTL-Einstellung einer Tabelle zu aktualisieren.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
 --ttl 'status=ENABLED'
```

2. Um zu bestätigen, dass TTL jetzt für die Tabelle aktiviert ist, können Sie die folgende Anweisung verwenden.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Die Ausgabe der Anweisung sollte wie im folgenden Beispiel aussehen.

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
    "creationTimestamp": "2024-09-02T11:32:27.349000+00:00",
    "status": "ACTIVE",
    "schemaDefinition": {
        "allColumns": [
            {
                "name": "id",
                "type": "int"
            },
            {
                "name": "date",
                "type": "timestamp"
            },
            {
                "name": "name",
                "type": "text"
            }
        ],
        "partitionKeys": [
            {
                "name": "id"
```

```
            }
        ],
        "clusteringKeys": [],
        "staticColumns": []
    },
    "capacitySpecification": {
        "throughputMode": "PAY_PER_REQUEST",
        "lastUpdateToPayPerRequestTimestamp": "2024-09-02T11:32:27.349000+00:00"
    },
    "encryptionSpecification": {
        "type": "AWS_OWNED_KMS_KEY"
    },
    "pointInTimeRecovery": {
        "status": "DISABLED"
    },
    "ttl": {
        "status": "ENABLED"
    },
    "defaultTimeToLive": 0,
    "comment": {
        "message": ""
    },
    "replicaSpecifications": []
}
```

Verwenden Sie die **`INSERT`** Anweisung, um benutzerdefinierte TTL-Werte (Time to Live) für neue Zeilen festzulegen

> **Note**
>
> Bevor Sie mithilfe der `INSERT` Anweisung benutzerdefinierte TTL-Werte für Zeilen festlegen können, müssen Sie zunächst benutzerdefinierte TTL für die Tabelle aktivieren. Weitere Informationen finden Sie unter the section called “Tabelle mit benutzerdefinierter TTL aktualisieren”.

Um den Standard-TTL-Wert einer Tabelle zu überschreiben, indem Sie Ablaufdaten für einzelne Zeilen festlegen, können Sie die folgende Anweisung verwenden: `INSERT`

- `INSERT`— Fügt eine neue Datenzeile mit einem TTL-Wertesatz ein.

Das Setzen von TTL-Werten für neue Zeilen mithilfe der `INSERT` Anweisung hat Vorrang vor der TTL-Standardeinstellung der Tabelle.

Die folgende CQL-Anweisung fügt eine Datenzeile in die Tabelle ein und ändert die TTL-Standardeinstellung auf 259.200 Sekunden (entspricht 3 Tagen).

```
INSERT INTO my_table (userid, time, subject, body, user)
        VALUES (B79CB3BA-745E-5D9A-8903-4A02327A7E09, 96a29100-5e25-11ec-90d7-
b5d91eceda0a, 'Message', 'Hello','205.212.123.123')
        USING TTL 259200;
```

Verwenden Sie die folgende Anweisung, um die TTL-Einstellungen für die eingefügte Zeile zu bestätigen.

```
SELECT TTL (subject) from my_table;
```

Verwenden Sie die **UPDATE** Anweisung, um benutzerdefinierte TTL-Einstellungen (Time to Live) für Zeilen und Spalten zu bearbeiten

> **Note**
>
> Bevor Sie benutzerdefinierte TTL-Werte für Zeilen und Spalten festlegen können, müssen Sie TTL zunächst für die Tabelle aktivieren. Weitere Informationen finden Sie unter the section called “Tabelle mit benutzerdefinierter TTL aktualisieren”.

Sie können die `UPDATE` Anweisung verwenden, um den Standard-TTL-Wert einer Tabelle zu überschreiben, indem Sie das Ablaufdatum für einzelne Zeilen und Spalten festlegen:

- Zeilen — Sie können eine bestehende Datenzeile mit einem benutzerdefinierten TTL-Wert aktualisieren.
- Spalten — Sie können eine Teilmenge von Spalten in vorhandenen Zeilen mit einem benutzerdefinierten TTL-Wert aktualisieren.

Das Festlegen von TTL-Werten für Zeilen und Spalten hat Vorrang vor der Standard-TTL-Einstellung für die Tabelle.

Verwenden Sie die folgende Anweisung, um die TTL-Einstellungen der zuvor eingefügten Spalte „Betreff“ von 259.200 Sekunden (3 Tage) auf 86.400 Sekunden (ein Tag) zu ändern.

```
UPDATE my_table USING TTL 86400 set subject = 'Updated Message' WHERE userid =
 B79CB3BA-745E-5D9A-8903-4A02327A7E09 and time = 96a29100-5e25-11ec-90d7-b5d91eceda0a;
```

Sie können eine einfache Auswahlabfrage ausführen, um den aktualisierten Datensatz vor Ablauf der Ablaufzeit zu sehen.

```
SELECT * from my_table;
```

Die Abfrage zeigt die folgende Ausgabe.

```
userid                                | time                                 | body  |
 subject         | user
--------------------------------------+--------------------------------------+-------
+-----------------+-----------------
b79cb3ba-745e-5d9a-8903-4a02327a7e09  | 96a29100-5e25-11ec-90d7-b5d91eceda0a | Hello |
 Updated Message | 205.212.123.123
50554d6e-29bb-11e5-b345-feff819cdc9f  | cf03fb21-59b5-11ec-b371-dff626ab9620 | Hello |
        Message | 205.212.123.123
```

Um zu überprüfen, ob der Ablauf erfolgreich war, führen Sie dieselbe Abfrage nach der konfigurierten Ablaufzeit erneut aus.

```
SELECT * from my_table;
```

Die Abfrage zeigt die folgende Ausgabe, nachdem die Spalte „Betreff“ abgelaufen ist.

```
userid                                | time                                 | body  |
 subject | user
--------------------------------------+--------------------------------------+-------
+---------+-----------------
b79cb3ba-745e-5d9a-8903-4a02327a7e09  | 96a29100-5e25-11ec-90d7-b5d91eceda0a | Hello |
   null | 205.212.123.123
50554d6e-29bb-11e5-b345-feff819cdc9f  | cf03fb21-59b5-11ec-b371-dff626ab9620 | Hello |
 Message | 205.212.123.123
```

Verwenden Sie diesen Service mit einem AWS SDK

AWS Software Development Kits (SDKs) sind für viele beliebte Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++ Code-Beispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK für Go	AWS SDK für Go Code-Beispiele
AWS SDK für Java	AWS SDK für Java Code-Beispiele
AWS SDK für JavaScript	AWS SDK für JavaScript Code-Beispiele
AWS SDK für Kotlin	AWS SDK für Kotlin Code-Beispiele
AWS SDK für .NET	AWS SDK für .NET Code-Beispiele
AWS SDK für PHP	AWS SDK für PHP Code-Beispiele
AWS -Tools für PowerShell	AWS -Tools für PowerShell Code-Beispiele
AWS SDK für Python v4	AWS SDK für Python v4-Codebeispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3) Codebeispiele
AWS SDK für Ruby	AWS SDK für Ruby Code-Beispiele
AWS SDK für Rust	AWS SDK für Rust Code-Beispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Code-Beispiele
AWS SDK für Swift	AWS SDK für Swift Code-Beispiele

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

Arbeiten mit Tags und Labels für Amazon Keyspaces-Ressourcen

Sie können Amazon Keyspaces-Ressourcen (für Apache Cassandra) mithilfe von Tags kennzeichnen. Mithilfe von Tags können Sie Ihre Ressourcen auf unterschiedliche Weise kategorisieren, z. B. nach Zweck, Eigentümer, Umgebung oder anderen Kriterien. Tags können Sie bei Folgendem unterstützen:

- Eine Ressource basierend auf den Tags, die Sie ihr zugeordnet haben, schnell zu erkennen.
- Sehen Sie sich AWS Rechnungen an, die nach Schlagwörtern aufgeschlüsselt sind.
- Steuern Sie den Zugriff auf Amazon Keyspaces-Ressourcen anhand von Tags. Beispiele für IAM-Richtlinien mit Tags finden Sie unter. the section called “Autorisierung basierend auf Amazon Keyspaces-Tags”

Tagging wird von AWS Diensten wie Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Keyspaces und mehr unterstützt. Effizientes Tagging kann Kosteneinblicke bereitstellen, mit denen Sie Berichte für Services erstellen können, die ein spezifisches Tag aufweisen.

Gehen Sie wie folgt vor, um sich mit dem Taggen vertraut zu machen:

1. Lesen Sie Einschränkungen für die Verwendung von Tags zur Kennzeichnung von Ressourcen in Amazon Keyspaces.
2. Tags mit Taggen Sie Schlüsselräume, Tabellen und Streams in Amazon Keyspaces erstellen.
3. Verwenden Sie diese Kostenverteilungsberichte mithilfe von Tags für Amazon Keyspaces erstellen Option, um Ihre AWS Kosten pro aktivem Tag zu verfolgen.

Schließlich wird empfohlen, optimale Tagging-Strategien zu befolgen. Weitere Informationen finden Sie unter AWS -Markierungsstrategien.

Einschränkungen für die Verwendung von Tags zur Kennzeichnung von Ressourcen in Amazon Keyspaces

Jedes Tag besteht aus einem Schlüssel und einem Wert, die Sie beide selbst definieren. Beachten Sie die folgenden Einschränkungen:

- Jeder Amazon Keyspaces-Schlüsselraum, jede Tabelle oder jeder Stream kann nur ein Tag mit demselben Schlüssel haben. Wenn Sie versuchen, ein vorhandenes Tag (derselbe Schlüssel) hinzuzufügen, wird der vorhandene Tag-Wert auf den neuen Wert aktualisiert.
- Auf einen Schlüsselraum angewendete Tags gelten nicht automatisch für Tabellen innerhalb dieses Schlüsselraums. Um dasselbe Tag auf einen Schlüsselraum und alle zugehörigen Tabellen anzuwenden, muss jede Ressource einzeln markiert werden.
- Auf eine Tabelle angewendete Tags gelten nicht automatisch für den Stream dieser Tabelle. Um bei der Tabellenerstellung dieselben Tags auf eine Tabelle und den Stream anzuwenden, können Sie das `PropagateTagsOnEnable` Flag beim Erstellen der Tabelle verwenden. Mit diesem Flag wendet Amazon Keyspaces die Tags der Tabelle während der Stream-Erstellung auf den Stream an. Wenn der Stream aktiv ist, gelten Änderungen an den Tabellen-Tags nicht für den Stream.
- Wenn Sie einen Schlüsselraum oder eine Tabelle mit mehreren Regionen erstellen, werden alle Tags, die Sie während des Erstellungsprozesses definieren, automatisch auf alle Schlüsselräume und Tabellen in allen Regionen angewendet. Wenn Sie vorhandene Tags mithilfe von `ALTER KEYSPACE` oder ändern`ALTER TABLE`, wird die Aktualisierung nur auf den Schlüsselraum oder die Tabelle in der Region angewendet, in der Sie die Änderung vornehmen.
- Ein Wert fungiert als Deskriptor in einer Tag-Kategorie (Schlüssel). In Amazon Keyspaces darf der Wert nicht leer oder Null sein.
- Bei Tag-Schlüsseln und -Werten muss die Groß- und Kleinschreibung beachtet werden.
- Die maximale Schlüssellänge beträgt 128 Unicode-Zeichen.
- Die maximale Wertlänge beträgt 256 Unicode-Zeichen.
- Namen dürfen Buchstaben, Leerzeichen und Zahlen sowie die folgenden Sonderzeichen enthalten: `+ - = . _ : /`
- Die maximale Anzahl an Tags pro Ressource beträgt 50.
- AWS-zugewiesenen Tag-Namen und Werten wird automatisch das `aws:` Präfix zugewiesen, das Sie nicht zuweisen können. AWS-zugewiesene Tagnamen werden nicht auf das Tag-Limit von 50 angerechnet. User-assigned Tagnamen haben das Präfix `user:` im Kostenzuordnungsbericht.
- Sie können die Anwendung eines Tags nicht rückdatieren.

Taggen Sie Schlüsselräume, Tabellen und Streams in Amazon Keyspaces

Sie können Tags für Keyspaces, Tabellen und Streams mit der Amazon Keyspaces-Konsole, der oder Cassandra Query Language (CQL) hinzufügen AWS CLI, auflisten, bearbeiten oder löschen. Anschließend können Sie diese benutzerdefinierten Tags aktivieren, sodass sie auf der AWS Fakturierung und Kostenmanagement Konsole zur Nachverfolgung der Kostenzuweisung angezeigt werden. Weitere Informationen finden Sie unter Kostenverteilungsberichte mithilfe von Tags für Amazon Keyspaces erstellen.

Für die Massenbearbeitung können Sie auch den Tag-Editor auf der Konsole verwenden. Weitere Informationen findne Sie unter Arbeiten mit Tag Editor im AWS -Ressourcengruppen - Benutzerhandbuch.

Informationen zur Tag-Struktur finden Sie unter Einschränkungen für die Verwendung von Tags zur Kennzeichnung von Ressourcen in Amazon Keyspaces.

Themen

- Fügen Sie beim Erstellen eines neuen Keyspace Tags hinzu
- Fügt Tags zu einem Schlüsselraum hinzu
- Löschen Sie Tags aus einem Schlüsselraum
- Zeigt die Tags eines Schlüsselraums an
- Fügen Sie beim Erstellen einer neuen Tabelle Tags hinzu
- Fügen Sie einer Tabelle Tags hinzu
- Löschen Sie Tags aus einer Tabelle
- Zeigt die Tags einer Tabelle an
- Fügen Sie beim Erstellen einer Tabelle Tags zu einem neuen Stream hinzu
- Fügen Sie einem neuen Stream Tags für eine bestehende Tabelle hinzu
- Fügen Sie einem Stream neue Tags hinzu
- Löschen Sie Tags aus einem Stream
- Zeigt die Tags eines Streams an

Fügen Sie beim Erstellen eines neuen Keyspace Tags hinzu

Sie können die Amazon Keyspaces-Konsole, CQL oder die verwenden, AWS CLI um Tags hinzuzufügen, wenn Sie einen neuen Schlüsselraum erstellen.

Console

Legen Sie ein Tag fest, wenn Sie mit der Konsole einen neuen Schlüsselraum erstellen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces und anschließend Create Keyspace aus.
3. Geben Sie auf der Seite „Schlüsselraum erstellen“ einen Namen für den Schlüsselraum ein.
4. Wählen Sie unter Tags die Option Neues Tag hinzufügen und geben Sie einen Schlüssel und einen Wert ein.
5. Wählen Sie „Schlüsselraum erstellen“.

Cassandra Query Language (CQL)

Legen Sie ein Tag fest, wenn Sie mit CQL einen neuen Schlüsselraum erstellen

- Das folgende Beispiel erstellt einen neuen Schlüsselraum mit Tags.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

CLI

Legen Sie ein Tag fest, wenn Sie einen neuen Schlüsselraum mit dem AWS CLI

- Die folgende Anweisung erstellt einen neuen Schlüsselraum mit Tags.

```
aws keyspaces create-keyspace --keyspace-name 'myKeyspace' --tags
 'key=key1,value=val1' 'key=key2,value=val2'
```

Fügt Tags zu einem Schlüsselraum hinzu

Die folgenden Beispiele zeigen, wie Sie Tags zu einem Schlüsselraum in Amazon Keyspaces hinzufügen.

Console

Fügen Sie mithilfe der Konsole ein Tag zu einem vorhandenen Schlüsselraum hinzu

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces aus.
3. Wählen Sie einen Schlüsselraum aus der Liste aus. Wählen Sie dann die Registerkarte „Tags“, auf der Sie die Tags des Schlüsselraums anzeigen können.
4. Wählen Sie Tags verwalten, um Tags hinzuzufügen, zu bearbeiten oder zu löschen.
5. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Fügen Sie mithilfe von CQL ein Tag zu einem vorhandenen Schlüsselraum hinzu

- ```
  ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
  ```

CLI

Fügen Sie einem vorhandenen Schlüsselraum ein Tag hinzu mit dem AWS CLI

- Das folgende Beispiel zeigt, wie neue Tags zu einem vorhandenen Schlüsselraum hinzugefügt werden.

  ```
  aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-
  east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3'
   'key=key4,value=val4'
  ```

Löschen Sie Tags aus einem Schlüsselraum

Console

Löschen Sie mithilfe der Konsole ein Tag aus einem vorhandenen Schlüsselraum

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
  ```

2. Wählen Sie im Navigationsbereich Keyspaces aus.
3. Wählen Sie einen Schlüsselraum aus der Liste aus. Wählen Sie dann die Registerkarte „Tags“, auf der Sie die Tags des Schlüsselraums anzeigen können.
4. Wähle Tags verwalten und lösche die Tags, die du nicht mehr benötigst.
5. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Löschen Sie mithilfe von CQL ein Tag aus einem vorhandenen Schlüsselraum

- ```
  ALTER KEYSPACE mykeyspace DROP TAGS {'key1':'val1', 'key2':'val2'};
  ```

CLI

Löschen Sie ein Tag aus einem vorhandenen Schlüsselraum mit dem AWS CLI

- Die folgende Anweisung entfernt die angegebenen Tags aus einem Schlüsselraum.

  ```
  aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-
  east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3'
   'key=key4,value=val4'
  ```

Zeigt die Tags eines Schlüsselraums an

Die folgenden Beispiele zeigen, wie Sie Tags mit der Konsole, CQL oder dem lesen. AWS CLI

Console

Die Tags eines Keyspace mithilfe der Amazon Keyspaces-Konsole anzeigen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Keyspaces aus.
3. Wählen Sie einen Schlüsselraum aus der Liste aus. Wählen Sie dann die Registerkarte „Tags“, auf der Sie die Tags des Schlüsselraums anzeigen können.
  ```

Cassandra Query Language (CQL)

Zeigen Sie die Tags eines Schlüsselraums mit CQL an

Verwenden Sie die folgende CQL-Anweisung, um die an einen Schlüsselraum angehängten Tags zu lesen.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

Die WHERE Klausel ist erforderlich und muss eines der folgenden Formate verwenden:

- keyspace_name = **'mykeyspace'** AND resource_type = 'keyspace'
- resource_id = *arn*

- Die folgende Anweisung zeigt, ob ein Schlüsselraum Tags enthält.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_type = 'keyspace';
```

Die Ausgabe der Abfrage sieht wie folgt aus.

```
resource_id                                                    | keyspace_name
 | resource_name | resource_type | tags
-------------------------------------------------------------------
+---------------+---------------+---------------+------
arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/      | mykeyspace
    | mykeyspace    | keyspace      | {'key1': 'val1', 'key2': 'val2'}
```

CLI

Zeigen Sie die Tags eines Schlüsselraums mit dem AWS CLI

- Dieses Beispiel zeigt, wie die Tags der angegebenen Ressource aufgelistet werden.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111122223333:/keyspace/myKeyspace/'
```

Die Ausgabe des letzten Befehls sieht so aus.

```
{
    "tags": [
        {
            "key": "key1",
            "value": "val1"
        },
        {
            "key": "key2",
            "value": "val2"
        },
        {
            "key": "key3",
            "value": "val3"
        },
        {
            "key": "key4",
            "value": "val4"
        }
    ]
}
```

Fügen Sie beim Erstellen einer neuen Tabelle Tags hinzu

Sie können die Amazon Keyspaces-Konsole, CQL oder die verwenden, AWS CLI um Tags zu neuen Tabellen hinzuzufügen, wenn Sie sie erstellen.

Console

Fügen Sie ein Tag hinzu, wenn Sie eine neue Tabelle mit der (Konsole) erstellen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) und anschließend Create table (Tabelle erstellen) aus.
3. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die Tabelle ein.
4. Erstellen Sie im Abschnitt Schema das Schema für Ihre Tabelle.
5. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.

6. Fahren Sie mit dem Abschnitt Tabellentags — optional fort und wählen Sie Neues Tag hinzufügen, um neue Tags zu erstellen.
7. Wählen Sie Create table (Tabelle erstellen) aus.

Cassandra Query Language (CQL)

Fügen Sie Tags hinzu, wenn Sie eine neue Tabelle mit CQL erstellen

- Das folgende Beispiel erstellt eine neue Tabelle mit Tags.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

CLI

Fügen Sie Tags hinzu, wenn Sie eine neue Tabelle mit dem erstellen AWS CLI

- Das folgende Beispiel zeigt, wie eine neue Tabelle mit Tags erstellt wird. Der Befehl erstellt eine Tabelle MyTable in einem bereits vorhandenen Keyspace MyKeySpace. Beachten Sie, dass der Befehl aus Gründen der besseren Lesbarkeit in verschiedene Zeilen aufgeteilt wurde.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable'
            --schema-definition 'allColumns=[{name=id,type=int},
{name=name,type=text},{name=date,type=timestamp}],partitionKeys=[{name=id}]'
            --tags 'key=key1,value=val1' 'key=key2,value=val2'
```

Fügen Sie einer Tabelle Tags hinzu

Sie können einer vorhandenen Tabelle in Amazon Keyspaces mithilfe der Konsole, CQL oder der Tags hinzufügen. AWS CLI

Console

Hinzufügen von Tags zu einer Tabelle mithilfe der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.

3. Wählen Sie eine Tabelle aus der Liste aus und klicken Sie auf den Tab Tags.
4. Wählen Sie Tags verwalten, um der Tabelle Tags hinzuzufügen.
5. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Fügen Sie einer Tabelle mithilfe von CQL Tags hinzu

- Die folgende Anweisung zeigt, wie einer vorhandenen Tabelle Tags hinzugefügt werden.

```
ALTER TABLE mykeyspace.mytable ADD TAGS {'key1':'val1', 'key2':'val2'};
```

CLI

Fügen Sie einer Tabelle Tags hinzu, indem Sie den AWS CLI

- Das folgende Beispiel zeigt, wie neue Tags zu einer vorhandenen Tabelle hinzugefügt werden.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111122223333:/keyspace/myKeyspace/table/myTable' --tags
 'key=key3,value=val3' 'key=key4,value=val4'
```

Löschen Sie Tags aus einer Tabelle

Console

Löschen von Tags aus einer Tabelle mithilfe der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.
3. Wählen Sie eine Tabelle aus der Liste aus und klicken Sie auf den Tab Tags.
4. Wählen Sie Tags verwalten, um Tags aus der Tabelle zu löschen.
5. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Löschen Sie Tags mithilfe von CQL aus einer Tabelle

- Die folgende Anweisung zeigt, wie Tags aus einer vorhandenen Tabelle gelöscht werden.

```
ALTER TABLE mytable DROP TAGS {'key3':'val3', 'key4':'val4'};
```

CLI

Fügen Sie einer Tabelle Tags hinzu, indem Sie den AWS CLI

- Die folgende Anweisung entfernt die angegebenen Tags aus einem Schlüsselraum.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111122223333:/keyspace/myKeyspace/table/myTable' --tags
 'key=key3,value=val3' 'key=key4,value=val4'
```

Zeigt die Tags einer Tabelle an

Die folgenden Beispiele zeigen, wie Sie die Tags einer Tabelle in Amazon Keyspaces mithilfe der Konsole, CQL oder der anzeigen können. AWS CLI

Console

Die Tags einer Tabelle mithilfe der Konsole anzeigen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.
3. Wählen Sie eine Tabelle aus der Liste aus und klicken Sie auf den Tab Tags.

Cassandra Query Language (CQL)

Zeigen Sie die Tags einer Tabelle mit CQL an

Verwenden Sie die folgende CQL-Anweisung, um die an eine Tabelle angehängten Tags zu lesen.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

Die WHERE Klausel ist erforderlich und muss eines der folgenden Formate verwenden:

- keyspace_name = **'mykeyspace'** AND resource_name = **'mytable'**
- resource_id = *arn*

- Die folgende Abfrage gibt die Tags der angegebenen Tabelle zurück.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_name = 'mytable';
```

Die Ausgabe dieser Abfrage sieht wie folgt aus.

```
resource_id                                                                  |
 keyspace_name | resource_name | resource_type | tags
------------------------------------------------------------------------------
+---------------+---------------+---------------+------
arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/mytable|
 mykeyspace    | mytable       | table         | {'key1': 'val1', 'key2':
 'val2'}
```

CLI

Zeigen Sie die Tags einer Tabelle mit dem AWS CLI

- Dieses Beispiel zeigt, wie die Tags der angegebenen Ressource aufgelistet werden.

```
aws keyspaces list-tags-for-resource --resource-arn
 'arn:aws:arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/2025-05-11T21:21:33.291cassandra:us-east-1:111122223333:/
keyspace/myKeyspace/table/myTable'
```

Die Ausgabe des letzten Befehls sieht so aus.

```
{
```

```
    "tags": [
        {
            "key": "key1",
            "value": "val1"
        },
        {
            "key": "key2",
            "value": "val2"
        },
        {
            "key": "key3",
            "value": "val3"
        },
        {
            "key": "key4",
            "value": "val4"
        }
    ]
}
```

Fügen Sie beim Erstellen einer Tabelle Tags zu einem neuen Stream hinzu

Sie können Tags zu Streams hinzufügen, wenn Sie eine neue Tabelle mit einem Stream erstellen, indem Sie CQL verwenden oder einen Stream AWS CLI taggen.

> **Note**
>
> Amazon Keyspaces CDC erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForAmazonKeyspacesCDC`), die in Ihrem Namen Metrikdaten aus Amazon Keyspaces CDC-Streams `"cloudwatch:namespace": "AWS/Cassandra"` in Ihrem CloudWatch Konto veröffentlicht. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “CDC-Streams”.

Cassandra Query Language (CQL)

Fügen Sie Tags zu einem Stream hinzu, wenn Sie eine neue Tabelle mit CQL erstellen

1. Um eine neue Tabelle mit einem Stream zu erstellen und die Tabellen-Tags automatisch auf den Stream anzuwenden, können Sie das `'propagate_tags': 'TABLE'` Flag verwenden. Die folgende Aussage ist ein Beispiel dafür.

```
CREATE TABLE mytable (pk int, ck text, PRIMARY KEY(pk))
WITH TAGS={'key1':'val1', 'key2':'val2'}
AND cdc = TRUE
AND CUSTOM_PROPERTIES={
    'cdc_specification': {
        'view_type': 'NEW_IMAGE',
        'propagate_tags': 'TABLE'
    }
};
```

2. Um neue Tags auf den Stream anzuwenden, können Sie das folgende Beispiel verwenden.

```
CREATE TABLE mytable (pk int, ck text, PRIMARY KEY(pk))
WITH TAGS={'key1':'val1', 'key2':'val2'}
AND cdc = TRUE
AND CUSTOM_PROPERTIES={
    'cdc_specification': {
        'view_type': 'NEW_IMAGE',
        'tags': { 'key': 'string', 'value': 'string' },
    }
};
```

CLI

Fügen Sie einem Stream Tags hinzu, wenn Sie eine neue Tabelle mit dem erstellen AWS CLI

1. Um eine Tabelle mit einem Stream zu erstellen und die Tabellen-Tags automatisch auf den Stream anzuwenden, können Sie das `propagateTags=Table` Flag verwenden. Der folgende Code ist ein Beispiel dafür.

```
aws keyspaces create-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table' \
```

```
--schema-definition 'allColumns=[{name=pk,type=int},
{name=ck,type=text}],clusteringKeys=[{name=ck,orderBy=ASC}],partitionKeys=[{name=pk}]'
 \
--tags key=tag_key,value=tag_value
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Um verschiedene Tags auf den Stream anzuwenden, können Sie das folgende Beispiel verwenden.

```
aws keyspaces create-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table' \
--schema-definition 'allColumns=[{name=pk,type=int},
{name=ck,type=text}],clusteringKeys=[{name=ck,orderBy=ASC}],partitionKeys=[{name=pk}]'
 \
--tags key=tag_key,value=tag_value
--cdc-specification 'status=ENABLED,viewType=NEW_IMAGE,tags=[{key=tag_key,
 value=tag_value}]'
```

Fügen Sie einem neuen Stream Tags für eine bestehende Tabelle hinzu

Sie können Tags hinzufügen, wenn Sie einen neuen Stream für eine bestehende Tabelle erstellen. Sie können entweder das `PropagateTags` Flag verwenden, um die Tabellen-Tags auf den Stream anzuwenden, oder neue Tags für den Stream angeben. Sie können CQL oder the verwenden AWS CLI , um einen neuen Stream zu taggen.

> **Note**
>
> Amazon Keyspaces CDC erfordert das Vorhandensein einer serviceverknüpften Rolle (`AWSServiceRoleForAmazonKeyspacesCDC`), die in Ihrem Namen Metrikdaten aus Amazon Keyspaces CDC-Streams `"cloudwatch:namespace": "AWS/Cassandra"` in Ihrem CloudWatch Konto veröffentlicht. Diese Rolle wird automatisch für Sie erstellt. Weitere Informationen finden Sie unter the section called “CDC-Streams”.

Console

Fügen Sie Tags hinzu, wenn Sie mit der (Konsole) einen neuen Stream erstellen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tabellen und dann die Tabelle aus, für die Sie einen Stream hinzufügen möchten.
3. Wählen Sie die Registerkarte Streams aus.
4. Wählen Sie im Abschnitt Stream-Details die Option Bearbeiten aus.
5. Wähle Streams einschalten aus.
6. Wählen Sie den Ansichtstyp und fahren Sie mit Tags fort, um Tags für den Stream zu erstellen.
7. Sie können eine der folgenden Optionen auswählen:
   - Keine Tags — Verwenden Sie diese Option, wenn Sie keine Tags für den Stream erstellen möchten.
   - Tags aus Tabelle kopieren — Verwenden Sie diese Option, wenn Sie die Tags aus der Tabelle in den Stream kopieren möchten. Nachdem Sie die Tags kopiert haben, können Sie sie für den Stream bearbeiten. Beachten Sie, dass diese Option nur verfügbar ist, wenn die Tabelle Tags enthält.
   - Neue Tags hinzufügen — Du kannst bis zu 50 Tags für den Stream hinzufügen, indem du Neues Tag hinzufügen auswählst.
8. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Füge Tags hinzu, wenn du einen neuen Stream erstellst

1. Um einen neuen Stream für eine bestehende Tabelle zu erstellen und die Tags der Tabelle auf den Stream anzuwenden, können Sie das `'propagate_tags': 'TABLE'` Flag verwenden. Die folgenden Anweisungen sind ein Beispiel dafür.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':
 { 'view_type': 'NEW_IMAGE', 'propagate_tags': 'TABLE' } };
```

2. Um einen neuen Stream für eine bestehende Tabelle zu erstellen und neue Tags anzugeben, können Sie das folgende Beispiel verwenden.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':
 { 'view_type': 'NEW_IMAGE', 'tags': { 'key': 'string', 'value': 'string' }} };
```

CLI

Fügen Sie Tags hinzu, wenn Sie einen neuen Stream mit dem AWS CLI

1. Um einen neuen Stream mit Tags zu erstellen, können Sie das `propagateTags=TABLE` Flag verwenden, um die Tags der Tabelle automatisch auf den Stream anzuwenden. Der folgende Code ist ein Beispiel dafür.

```
aws keyspaces update-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table' \
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Um einen neuen Stream für eine bestehende Tabelle zu erstellen und neue Tags anzugeben, können Sie das folgende Beispiel verwenden.

```
aws keyspaces update-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table' \
--cdc-specification 'status=ENABLED,viewType=NEW_IMAGE,tags=[{key=tag_key,
 value=tag_value}]'
```

Fügen Sie einem Stream neue Tags hinzu

Sie können einem vorhandenen Stream in Amazon Keyspaces mithilfe der CQL oder der neue Tags hinzufügen. AWS CLI Sie können Tags nur zum neuesten Stream hinzufügen.

Console

Fügen Sie einem vorhandenen Stream (Konsole) Tags hinzu

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.

2. Wählen Sie im Navigationsbereich Tabellen und dann die Tabelle mit dem Stream aus, den Sie taggen möchten.
3. Wählen Sie die Registerkarte Streams aus.
4. Wählen Sie im Abschnitt Tags (Markierungen) die Option Manage tags (Tags (Markierungen) verwalten).
5. Wählen Sie Neues Tag hinzufügen, um ein neues Tag hinzuzufügen. Sie können bis zu 50 Tags erstellen, indem Sie diesen Schritt wiederholen.
6. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Fügen Sie einem Stream mithilfe von CQL Tags hinzu

- Die folgende Anweisung zeigt, wie einem vorhandenen Stream Tags hinzugefügt werden.

```
ALTER TABLE mykeyspace.mytable ADD TAGS_FOR_CDC {'key1':'val1', 'key2':'val2'};
```

CLI

Fügen Sie einem Stream Tags hinzu, indem Sie den AWS CLI

- Das folgende Beispiel zeigt, wie Sie einem vorhandenen Stream neue Tags hinzufügen.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-05-11T21:21:33.291' --tags 'key=key3,value=val3'
 'key=key4,value=val4'
```

Löschen Sie Tags aus einem Stream

Um Tags aus einem Stream zu löschen, können Sie CQL oder die verwenden. AWS CLI Sie können nur die Tags für den letzten Stream löschen.

Console

Löschen von Tags aus einer Tabelle mithilfe der Amazon Keyspaces-Konsole

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.
3. Wählen Sie eine Tabelle aus der Liste aus und klicken Sie auf den Tab Streams.
4. Wählen Sie im Abschnitt „Tags“ die Option „Tags verwalten“, um Tags aus der Tabelle zu löschen.
5. Wählen Sie nach dem Tag, das Sie löschen möchten, die Option Entfernen aus.
6. Wählen Sie Änderungen speichern aus.

Cassandra Query Language (CQL)

Löschen Sie Tags mithilfe von CQL aus einem Stream

- Die folgende Anweisung zeigt, wie Tags aus einem vorhandenen Stream gelöscht werden.

```
ALTER TABLE mytable DROP TAGS_FOR_CDC {'key3':'val3', 'key4':'val4'};
```

CLI

Löschen Sie Tags aus einem Stream mit dem AWS CLI

- Die folgende Anweisung entfernt die angegebenen Tags aus einem Stream.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-05-11T21:21:33.291' --tags 'key=key3,value=val3'
 'key=key4,value=val4'
```

Zeigt die Tags eines Streams an

Die folgenden Beispiele zeigen, wie Sie die Tags eines Streams in Amazon Keyspaces mithilfe von CQL oder dem anzeigen können. AWS CLI

Console

Die Tags eines Streams mithilfe der Konsole anzeigen

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.
3. Wählen Sie eine Tabelle aus der Liste aus und klicken Sie auf den Tab Streams.
4. Sie können die Tags des Streams im Abschnitt „Tags“ einsehen.

Cassandra Query Language (CQL)

Die Tags eines Streams mithilfe von CQL anzeigen

Um die an einen Stream angehängten Tags zu lesen, müssen Sie den Ressourcen-ARN des Streams in der `WHERE` Klausel angeben. Die folgende CQL-Syntax ist ein Beispiel dafür.

```
SELECT * FROM system_schema_mcs.tags WHERE resource_id = stream_arn;
```

- Die folgende Abfrage gibt die Tags für den angegebenen Stream zurück.

```
SELECT tags FROM system_schema_mcs.tags WHERE resource_id =
 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-05-06T17:17:39.800';
```

Die Ausgabe dieser Abfrage sieht wie folgt aus.

```
 resource_id
                                    | keyspace_name | resource_name           |
 resource_type | tags
 ------------------------------------------------------------------------------------------
+---------------+-------------------------+--------------
+----------------------
 arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-04-02T23:00:07.052     |      singleks | 2025-04-02T23:00:07.052 |
      stream | {'tagkey': 'tagval'}
```

CLI

Zeigen Sie die Tags eines Streams mit dem AWS CLI

- Dieses Beispiel zeigt, wie die Tags für alle Streams unter dem angegebenen Schlüsselraum aufgelistet werden.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-05-11T21:21:33.291'
```

Die Ausgabe des letzten Befehls sieht so aus.

```
{
    "tags": [
        {
            "key": "key1",
            "value": "val1"
        },
        {
            "key": "key2",
            "value": "val2"
        },
        {
            "key": "key3",
            "value": "val3"
        },
        {
            "key": "key4",
            "value": "val4"
        }
    ]
}
```

Kostenverteilungsberichte mithilfe von Tags für Amazon Keyspaces erstellen

AWS verwendet Tags, um die Ressourcenkosten in Ihrem Kostenverteilungsbericht zu organisieren. AWS bietet zwei Arten von Kostenzuordnungs-Tags:

- Ein AWS-generiertes Tag. AWS definiert, erstellt und wendet dieses Tag für Sie an.
- User-defined Schlagworte. Sie definieren und erstellen diese Tags und wenden sie an.

Sie müssen beide Arten von Tags separat aktivieren, bevor sie in Cost Explorer oder einem Kostenzuordnungsbericht angezeigt werden können.

Um AWS-generierte Tags zu aktivieren:

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Billing and Cost Management-Konsole unter https://console.aws.amazon.com/billing/home#/.
2. Wählen Sie im Navigationsbereich die Option Cost Allocation Tags (Kostenzuordnungs-Tags) aus.
3. Wählen Sie unter AWS-generierte Kostenzuordnungs-Tags die Option Aktivieren.

So aktivieren Sie benutzerdefinierte Tags:

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Billing and Cost Management-Konsole unter https://console.aws.amazon.com/billing/home#/.
2. Wählen Sie im Navigationsbereich die Option Cost Allocation Tags (Kostenzuordnungs-Tags) aus.
3. Wählen Sie unter User-Defined Cost Allocation Tags die Option Aktivieren aus.

Nachdem Sie Tags erstellt und aktiviert haben, AWS wird ein Kostenzuordnungsbericht generiert, in dem Ihre Nutzung und Kosten nach Ihren aktiven Stichwörtern gruppiert sind. Der Kostenzuordnungsbericht enthält alle Ihre AWS Kosten für jeden Abrechnungszeitraum. Der Bericht enthält getaggte und nicht getaggte Ressourcen, sodass Sie die Gebühren für Ressourcen übersichtlich verwalten können.

> **Note**
>
> Derzeit werden Daten, die von Amazon Keyspaces übertragen werden, in Kostenverteilungsberichten nicht nach Tags aufgeschlüsselt.

Weitere Informationen finden Sie unter Verwendung von Kostenzuordnungs-Tags.

Erstellen Sie Amazon Keyspaces-Ressourcen mit AWS CloudFormation

Amazon Keyspaces ist integriert AWS CloudFormation, ein Service, der Ihnen hilft, Ihre AWS Keyspaces und Tabellen zu modellieren und einzurichten, sodass Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen müssen. Sie erstellen eine Vorlage, die die gewünschten Schlüsselbereiche und Tabellen beschreibt, und CloudFormation kümmert sich um die Bereitstellung und Konfiguration dieser Ressourcen für Sie.

Wenn Sie Ihre Vorlage verwenden CloudFormation, können Sie sie wiederverwenden, um Ihre Amazon Keyspaces-Ressourcen konsistent und wiederholt einzurichten. Beschreiben Sie Ihre Ressourcen einfach einmal und stellen Sie dann dieselben Ressourcen immer wieder in mehreren Regionen AWS-Konten bereit.

Amazon Keyspaces und Vorlagen CloudFormation

Um Ressourcen für Amazon Keyspaces bereitzustellen und zu konfigurieren, müssen Sie CloudFormation Vorlagen verstehen. Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren CloudFormation Stacks bereitstellen möchten. Wenn Sie mit JSON oder YAML nicht vertraut sind, können Sie CloudFormation Designer verwenden, um Ihnen die ersten Schritte mit Vorlagen zu erleichtern. CloudFormation Weitere Informationen finden Sie unter Was ist CloudFormation Designer? im AWS CloudFormation Benutzerhandbuch.

Amazon Keyspaces unterstützt das Erstellen von Schlüsselräumen und Tabellen in. CloudFormation Für die Tabellen, die Sie mithilfe von CloudFormation Vorlagen erstellen, können Sie das Schema, den read/write Modus, die Einstellungen für den bereitgestellten Durchsatz und andere unterstützte Funktionen angeben. Weitere Informationen, einschließlich Beispielen für JSON- und YAML-Vorlagen für Schlüsselräume und Tabellen, finden Sie in der Referenz zum Ressourcentyp Amazon Keyspaces (für Apache Cassandra) in der Vorlagenreferenz.AWS CloudFormation

Erfahren Sie mehr über CloudFormation

Weitere Informationen CloudFormation dazu finden Sie in den folgenden Ressourcen:

- AWS CloudFormation
- AWS CloudFormation Benutzerhandbuch
- AWS CloudFormation Benutzerleitfaden für die Befehlszeilenschnittstelle

Verwenden von NoSQL Workbench mit Amazon Keyspaces (für Apache Cassandra)

NoSQL Workbench ist eine clientseitige Anwendung, mit der Sie nichtrelationale Datenmodelle für Amazon Keyspaces einfacher entwerfen und visualisieren können. NoSQL Workbench-Clients sind für Windows, macOS und Linux verfügbar.

Automatisches Entwerfen von Datenmodellen und Erstellen von Ressourcen

NoSQL Workbench bietet Ihnen eine Point-and-Click-Oberfläche zum Entwerfen und Erstellen von Amazon Keyspaces-Datenmodellen. Sie können ganz einfach neue Datenmodelle von Grund auf neu erstellen, indem Sie Schlüsselräume, Tabellen und Spalten definieren. Sie können auch vorhandene Datenmodelle importieren und Änderungen vornehmen (z. B. Spalten hinzufügen, bearbeiten oder entfernen), um die Datenmodelle für neue Anwendungen anzupassen. NoSQL Workbench ermöglicht es Ihnen dann, die Datenmodelle in Amazon Keyspaces oder Apache Cassandra zu übertragen und die Schlüsselbereiche und Tabellen automatisch zu erstellen. Informationen zum Erstellen von Datenmodellen finden Sie unter und. the section called “Erstellen Sie ein Datenmodell” the section called “Bearbeiten Sie ein Datenmodell”

Visualisieren von Datenmodellen

Mithilfe von NoSQL Workbench können Sie Ihre Datenmodelle visualisieren, um sicherzustellen, dass die Datenmodelle die Abfragen und Zugriffsmuster Ihrer Anwendung unterstützen.
Sie können Ihre Datenmodelle auch in einer Vielzahl von Formaten für Zusammenarbeit, Dokumentation und Präsentationen speichern und exportieren. Weitere Informationen finden Sie unter the section called “Visualisieren Sie ein Datenmodell”.

Themen

- NoSQL Workbench herunterladen
- Erste Schritte mit NoSQL Workbench
- Visualisieren Sie Datenmodelle mit NoSQL Workbench
- Erstellen Sie ein neues Datenmodell mit NoSQL Workbench
- Bestehende Datenmodelle mit NoSQL Workbench bearbeiten
- So übertragen Sie Datenmodelle an Amazon Keyspaces und Apache Cassandra
- Beispieldatenmodelle in NoSQL Workbench
- Versionsverlauf für NoSQL-Workbench

NoSQL Workbench herunterladen

Folgen Sie diesen Anweisungen, um NoSQL Workbench herunterzuladen und zu installieren.

Herunterladen und Installieren von NoSQL Workbench

1. Verwenden Sie einen der folgenden Links zum kostenlosen Herunterladen von NoSQL Workbench.

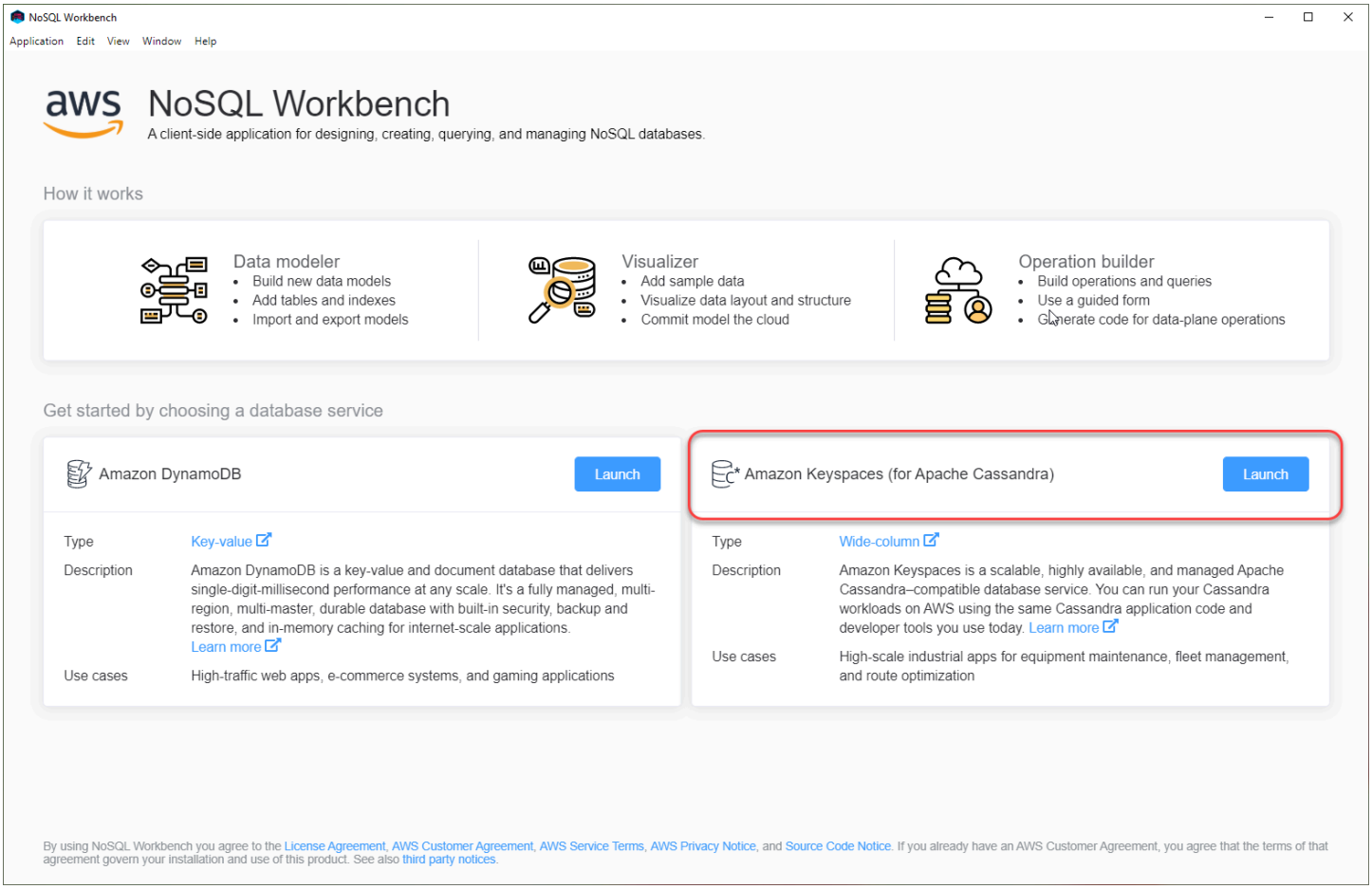
Betriebssystem	Download-Link
macOS	Für macOS herunterladen
Linux*	Download für Linux
Windows	Für Windows herunterladen

   * NoSQL-Workbench unterstützt Ubuntu 12.04, Fedora 21 und Debian 8 sowie sämtliche neueren Versionen dieser Linux-Verteilungen.

2. Starten Sie nach Abschluss des Downloads die Anwendung und folgen Sie den Anweisungen auf dem Bildschirm, um die Installation abzuschließen.

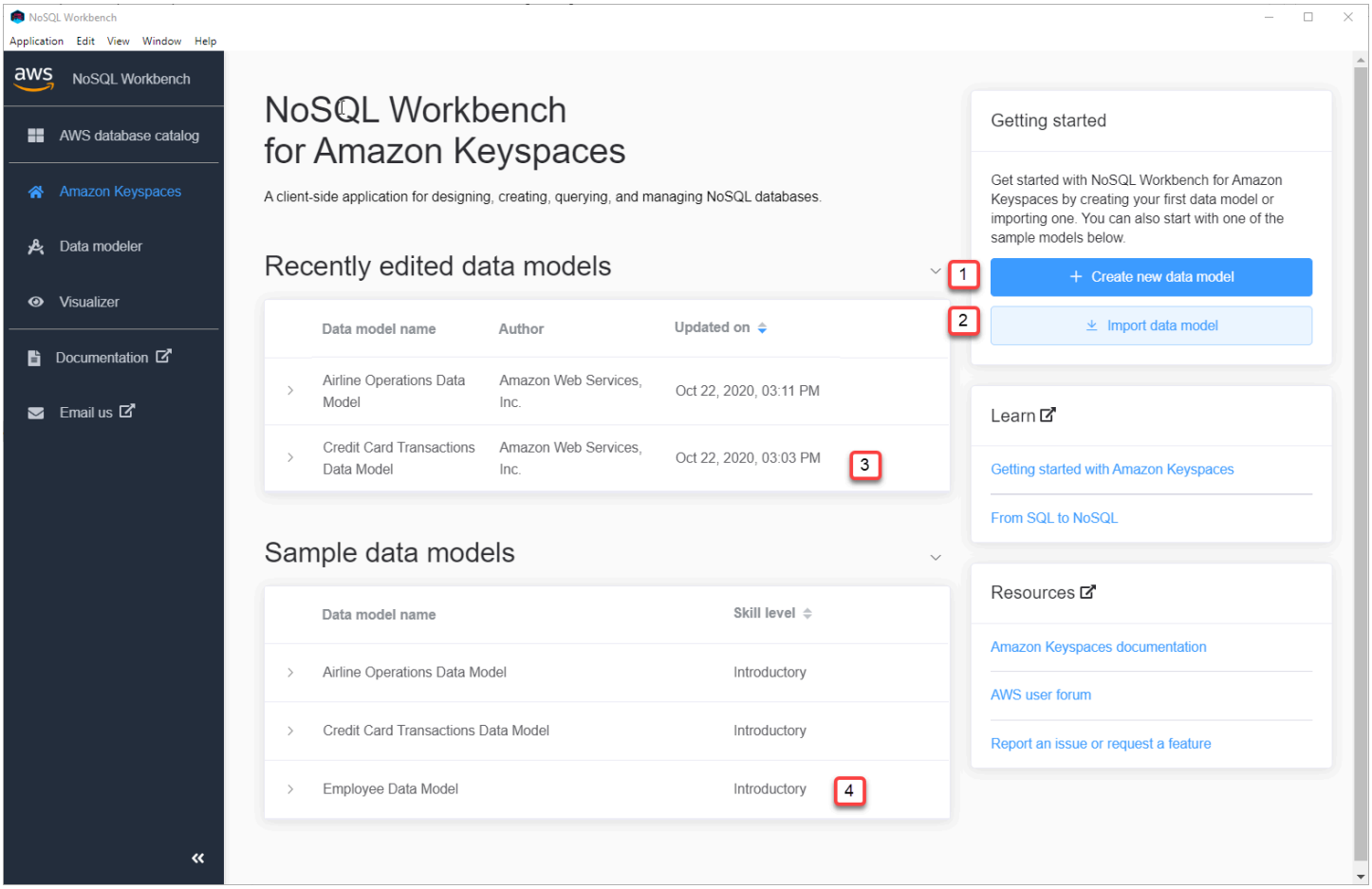
Erste Schritte mit NoSQL Workbench

Um mit NoSQL Workbench zu beginnen, wählen Sie auf der Datenbankkatalogseite in NoSQL Workbench Amazon Keyspaces und dann Launch aus.



Dadurch wird die NoSQL Workbench-Startseite für Amazon Keyspaces geöffnet, auf der Sie die folgenden Optionen haben, um loszulegen:

1. Erstellen Sie ein neues Datenmodell.
2. Importieren Sie ein vorhandenes Datenmodell im JSON-Format.
3. Öffnen Sie ein kürzlich bearbeitetes Datenmodell.
4. Öffnen Sie eines der verfügbaren Beispielmodelle.



Jede der Optionen öffnet den NoSQL Workbench-Datenmodellierer. Informationen zum weiteren Erstellen eines neuen Datenmodells finden Sie unter. the section called “Erstellen Sie ein Datenmodell” Informationen zum Bearbeiten eines vorhandenen Datenmodells finden Sie unterthe section called “Bearbeiten Sie ein Datenmodell”.

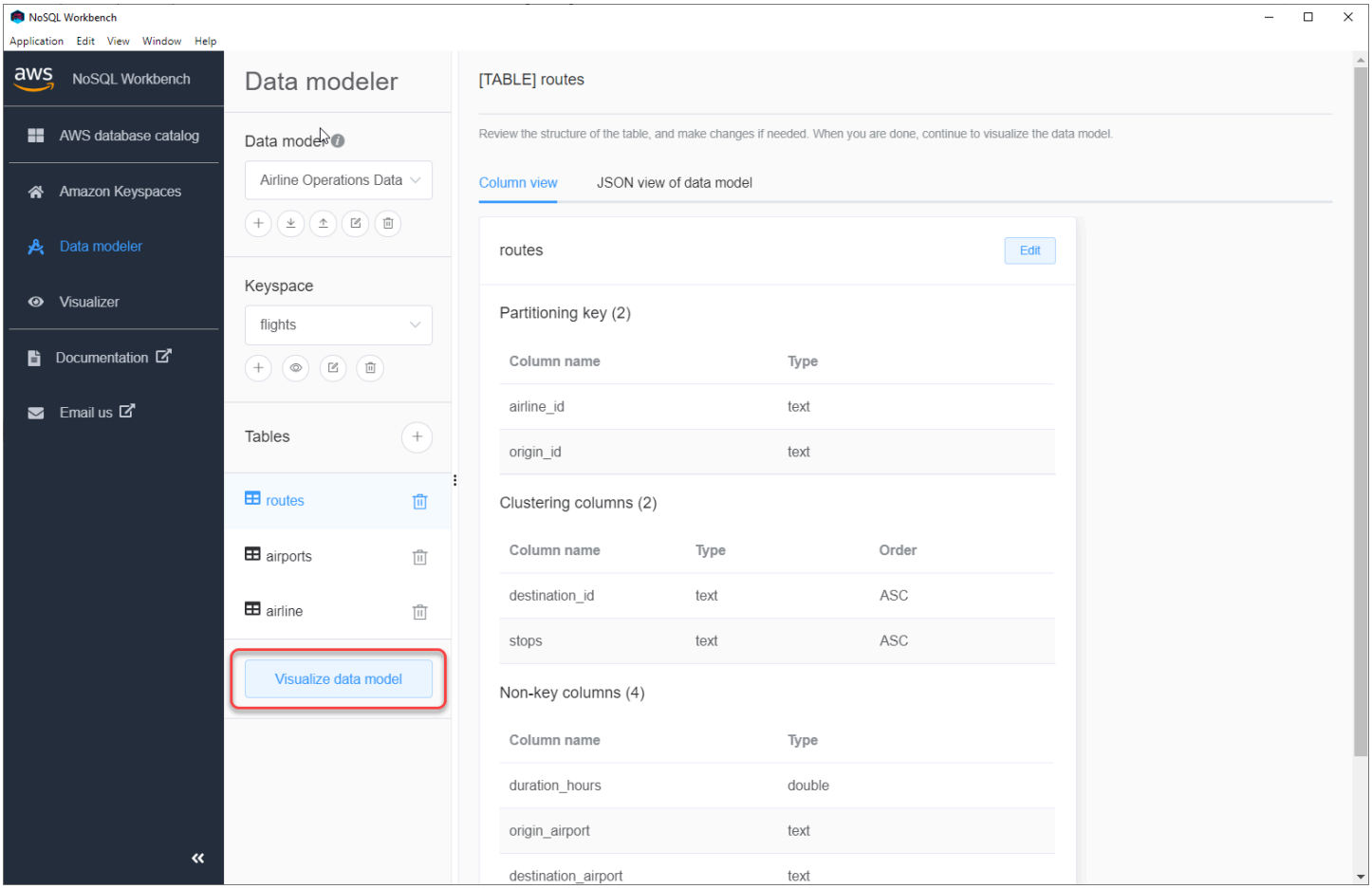
Visualisieren Sie Datenmodelle mit NoSQL Workbench

Mithilfe von NoSQL Workbench können Sie Ihre Datenmodelle visualisieren, um sicherzustellen, dass die Datenmodelle die Abfragen und Zugriffsmuster Ihrer Anwendung unterstützen. Sie können Ihre Datenmodelle auch in einer Vielzahl von Formaten für Zusammenarbeit, Dokumentation und Präsentationen speichern und exportieren.

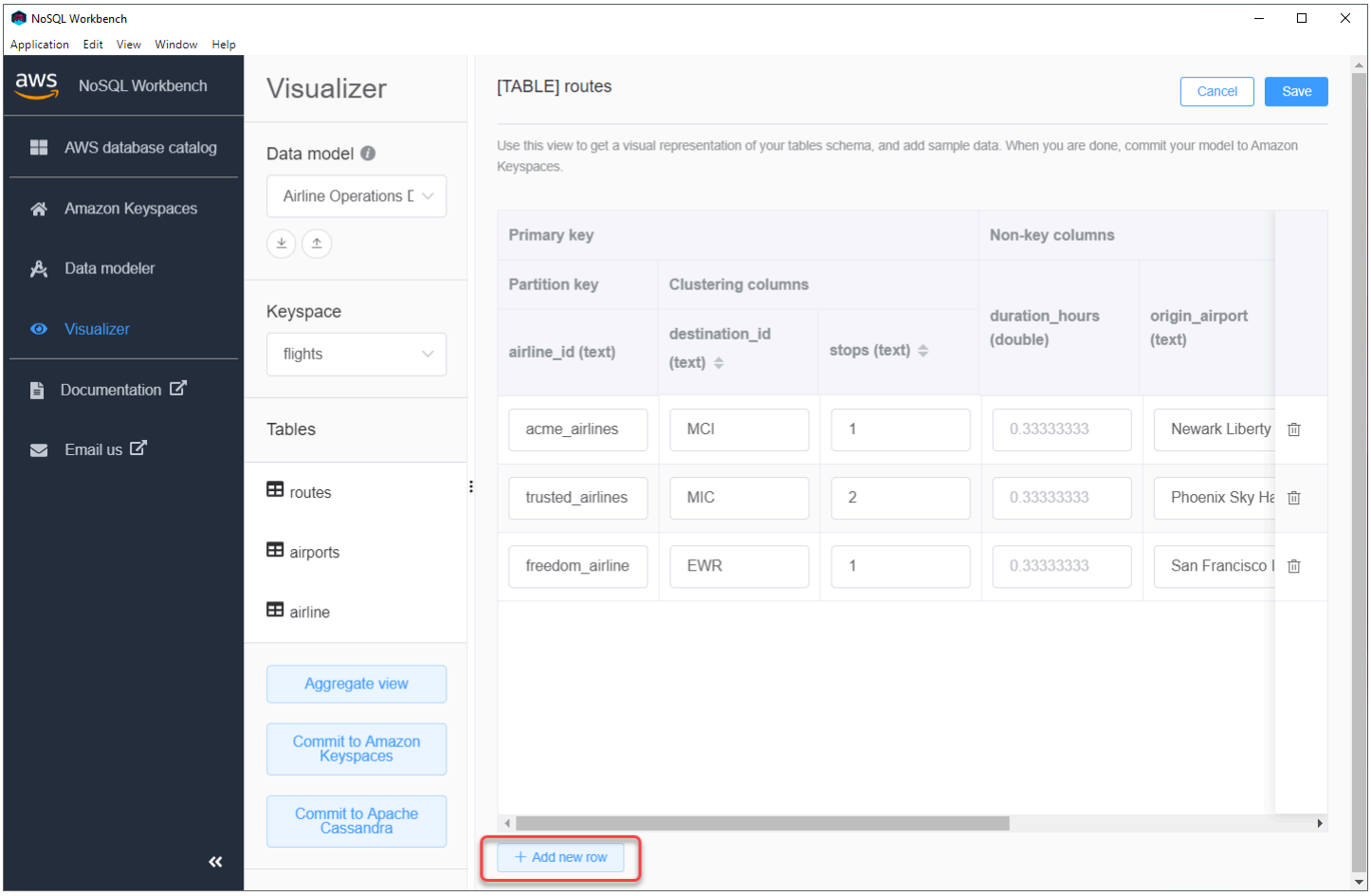
Nachdem Sie ein neues Datenmodell erstellt oder ein vorhandenes Datenmodell bearbeitet haben, können Sie das Modell visualisieren.

Visualisierung von Datenmodellen mit NoSQL Workbench

Wenn Sie das Datenmodell im Datenmodellierer fertiggestellt haben, wählen Sie Datenmodell visualisieren.

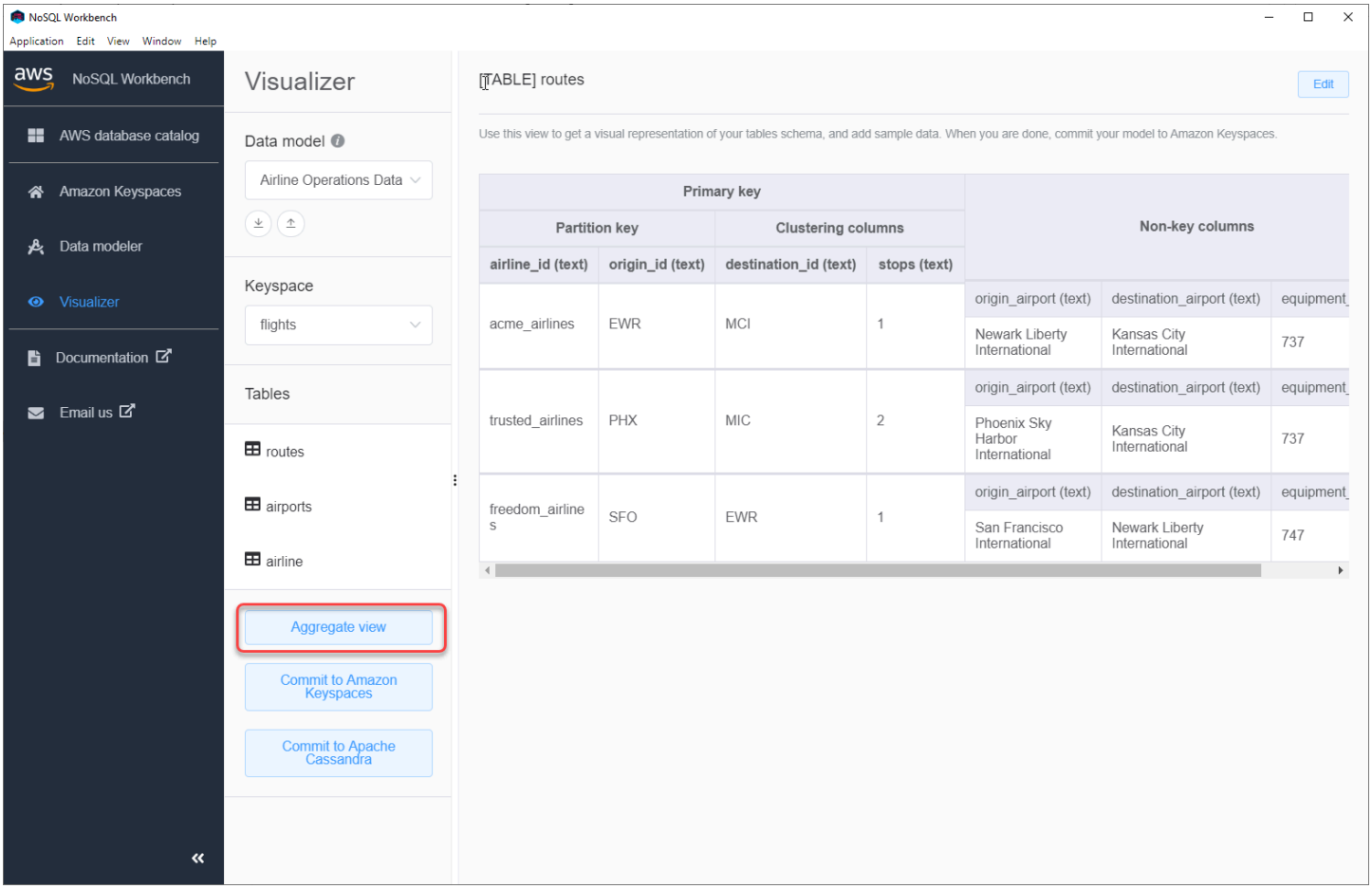


Dadurch gelangen Sie zum Datenvisualisierer in NoSQL Workbench. Der Datenvisualisierer bietet eine visuelle Darstellung des Tabellenschemas und ermöglicht das Hinzufügen von Beispieldaten. Um Beispieldaten zu einer Tabelle hinzuzufügen, wählen Sie eine Tabelle aus dem Modell aus und klicken Sie dann auf Bearbeiten. Um eine neue Datenzeile hinzuzufügen, wählen Sie am unteren Bildschirmrand die Option Neue Zeile hinzufügen aus. Klicken Sie auf Save (Speichern), wenn Sie fertig sind.



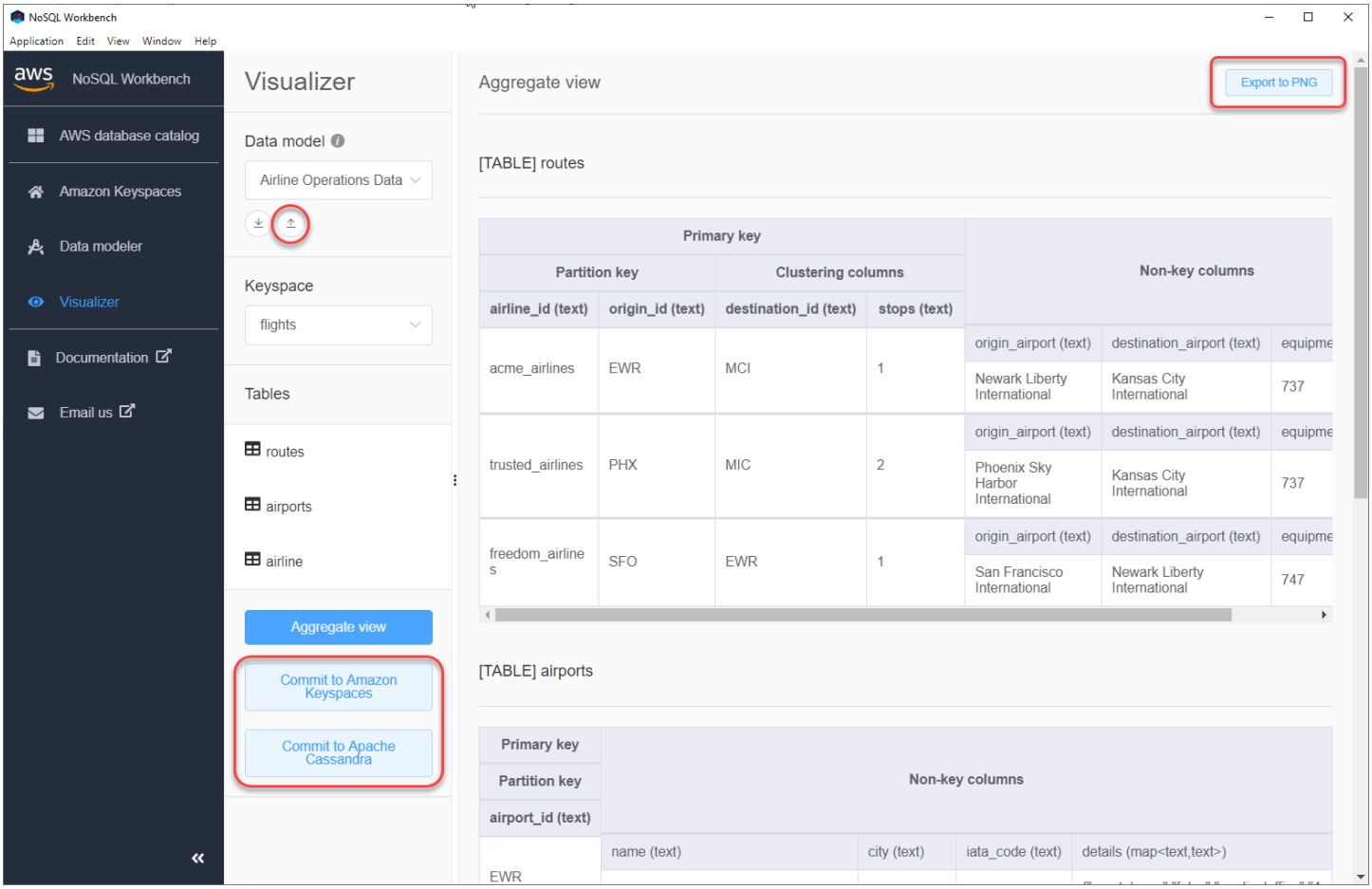
Aggregierte Ansicht

Nachdem Sie das Schema der Tabelle bestätigt haben, können Sie Datenmodellvisualisierungen aggregieren.



Nachdem Sie die Ansicht des Datenmodells aggregiert haben, können Sie die Ansicht in eine PNG-Datei exportieren. Um das Datenmodell in eine JSON-Datei zu exportieren, wählen Sie das Upload-Zeichen unter dem Namen des Datenmodells.

> **Note**
>
> Sie können das Datenmodell jederzeit während des Entwurfsprozesses im JSON-Format exportieren.



Sie haben die folgenden Optionen, um die Änderungen zu übernehmen:

- Binden Sie sich für Amazon Keyspaces ein
- Verpflichten Sie sich zu einem Apache Cassandra-Cluster

Weitere Informationen zum Commit von Änderungen finden Sie unterthe section called “Übernehmen Sie ein Datenmodell”.

Erstellen Sie ein neues Datenmodell mit NoSQL Workbench

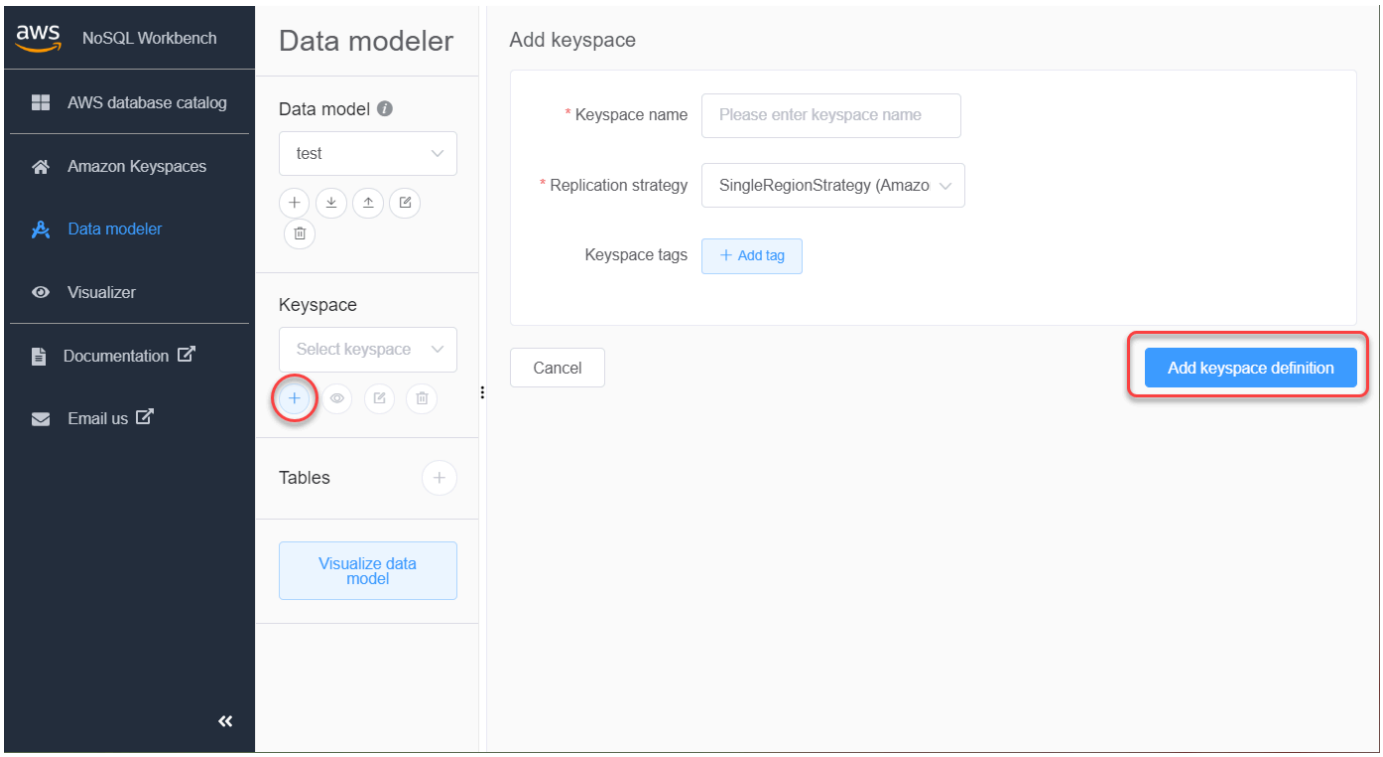
Sie können den NoSQL Workbench-Datenmodellierer verwenden, um neue Datenmodelle zu entwerfen, die auf den Datenzugriffsmustern Ihrer Anwendung basieren. Um ein neues Datenmodell für Amazon Keyspaces zu erstellen, können Sie den NoSQL Workbench-Datenmodellierer verwenden, um Schlüsselbereiche, Tabellen und Spalten zu erstellen. Folgen Sie diesen Schritten, um ein neues Datenmodell zu erstellen.

1. Um einen neuen Schlüsselraum zu erstellen, wählen Sie das Pluszeichen unter Schlüsselraum.

Wählen Sie in diesem Schritt die folgenden Eigenschaften und Einstellungen aus.

- Schlüsselraumname — Geben Sie den Namen des neuen Schlüsselraums ein.
- Replikationsstrategie — Wählen Sie die Replikationsstrategie für den Schlüsselraum. Amazon Keyspaces verwendet den SingleRegionStrategy, um Daten dreimal automatisch in mehreren AWS Availability Zones zu replizieren. Wenn Sie planen, das Datenmodell in einen Apache Cassandra-Cluster zu übertragen, können Sie oder wählen. SimpleStrategy NetworkTopologyStrategy
- Keyspaces-Tags — Ressourcen-Tags sind optional und ermöglichen es Ihnen, Ihre Ressourcen auf unterschiedliche Weise zu kategorisieren, z. B. nach Zweck, Besitzer, Umgebung oder anderen Kriterien. Weitere Informationen zu Stichwörtern für Amazon Keyspaces-Ressourcen finden Sie unter. the section called “Mit Tags arbeiten”

2. Wählen Sie Schlüsselraumdefinition hinzufügen, um den Schlüsselraum zu erstellen.

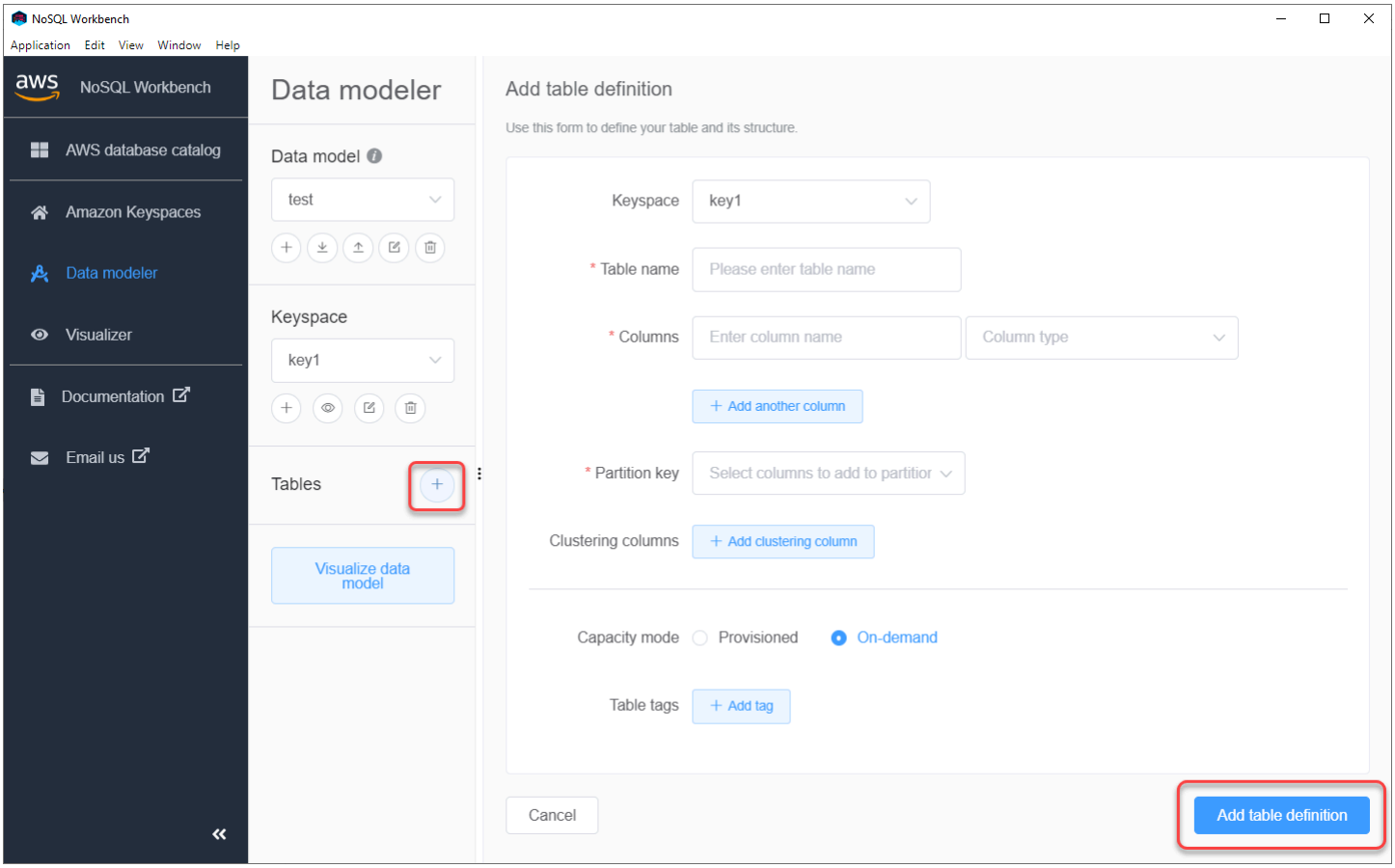


3. Um eine neue Tabelle zu erstellen, wählen Sie das Pluszeichen neben Tabellen. In diesem Schritt definieren Sie die folgenden Eigenschaften und Einstellungen.

- Tabellenname — Der Name der neuen Tabelle.
- Spalten — Fügen Sie einen Spaltennamen hinzu und wählen Sie den Datentyp. Wiederholen Sie diese Schritte für jede Spalte in Ihrem Schema.

- Partitionsschlüssel — Wählen Sie Spalten für den Partitionsschlüssel aus.
- Gruppierung von Spalten — Wählen Sie Gruppierungsspalten aus (optional).
- Kapazitätsmodus — Wählen Sie den read/write Kapazitätsmodus für die Tabelle. Sie können zwischen bereitgestellter und bedarfsgesteuerter Kapazität wählen. Weitere Informationen zu Kapazitätsmodi finden Sie unterthe section called “ read/write Kapazitätsmodi konfigurieren”.
- Tabellen-Tags — Ressourcen-Tags sind optional und ermöglichen es Ihnen, Ihre Ressourcen auf unterschiedliche Weise zu kategorisieren, z. B. nach Zweck, Besitzer, Umgebung oder anderen Kriterien. Weitere Informationen zu Stichwörtern für Amazon Keyspaces-Ressourcen finden Sie unter. the section called “Mit Tags arbeiten”

4. Wählen Sie Tabellendefinition hinzufügen, um die neue Tabelle zu erstellen.
5. Wiederholen Sie diese Schritte, um weitere Tabellen zu erstellen.
6. Fahren Sie mit the section called “Visualisieren eines Datenmodells” der Visualisierung des Datenmodells fort, das Sie erstellt haben.



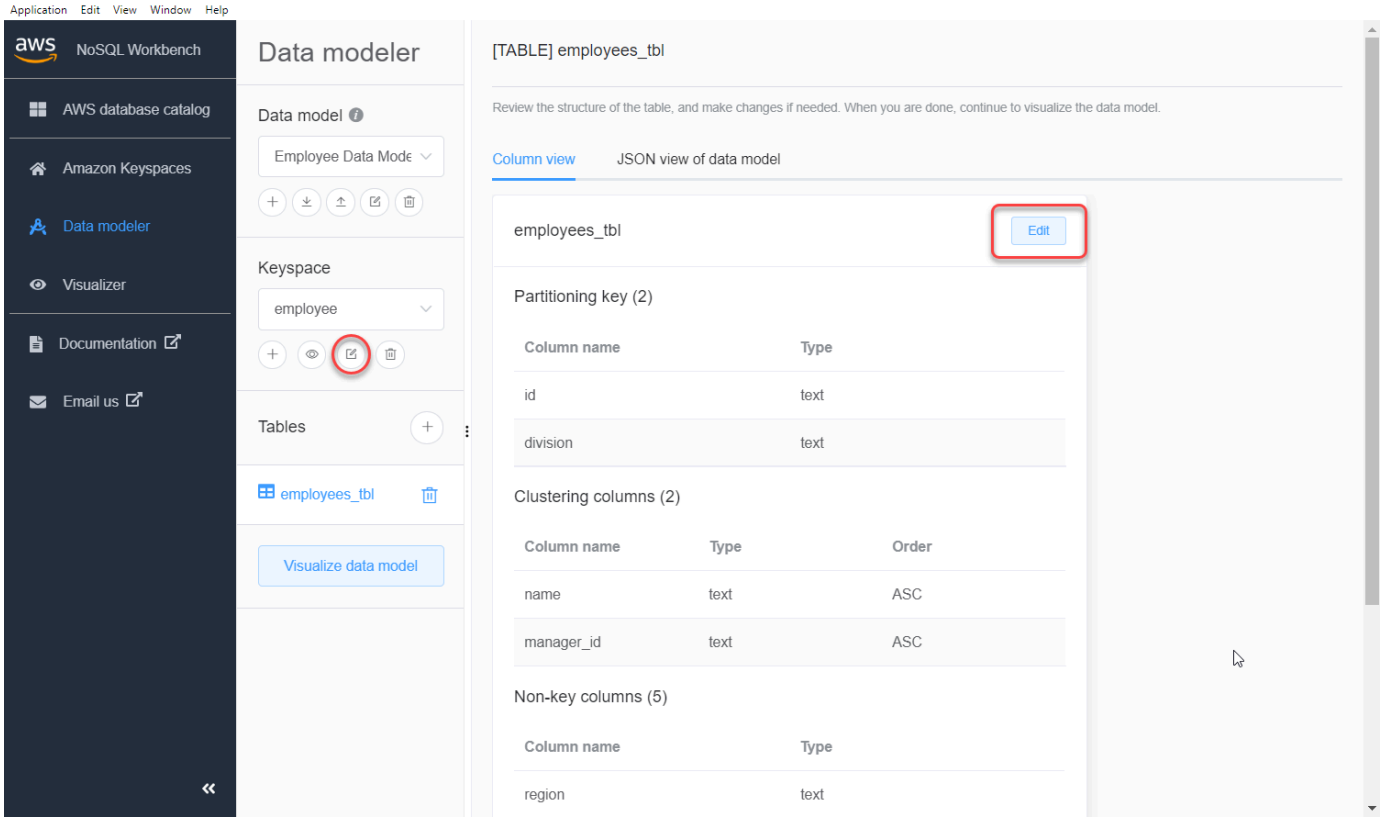
Bestehende Datenmodelle mit NoSQL Workbench bearbeiten

Sie können den Datenmodellierer verwenden, um vorhandene Datenmodelle, die mit NoSQL Workbench erstellt wurden, zu importieren und zu ändern. Der Datenmodellierer enthält auch einige Beispieldatenmodelle, die Ihnen den Einstieg in die Datenmodellierung erleichtern. Die Datenmodelle, die Sie mit NoSQL Workbench bearbeiten können, können Datenmodelle sein, die aus einer Datei importiert werden, die bereitgestellten Beispieldatenmodelle oder Datenmodelle, die Sie zuvor erstellt haben.

1. Um einen Schlüsselraum zu bearbeiten, wählen Sie das Bearbeitungssymbol unter Schlüsselraum.

   In diesem Schritt können Sie die folgenden Eigenschaften und Einstellungen bearbeiten.

   - Schlüsselraumname — Geben Sie den Namen des neuen Schlüsselraums ein.
   - Replikationsstrategie — Wählen Sie die Replikationsstrategie für den Schlüsselraum. Amazon Keyspaces verwendet den SingleRegionStrategy, um Daten dreimal automatisch in mehreren AWS Availability Zones zu replizieren. Wenn Sie planen, das Datenmodell in einen Apache Cassandra-Cluster zu übertragen, können Sie oder wählen. SimpleStrategy NetworkTopologyStrategy
   - Keyspaces-Tags — Ressourcen-Tags sind optional und ermöglichen es Ihnen, Ihre Ressourcen auf unterschiedliche Weise zu kategorisieren, z. B. nach Zweck, Besitzer, Umgebung oder anderen Kriterien. Weitere Informationen zu Stichwörtern für Amazon Keyspaces-Ressourcen finden Sie unter. the section called “Mit Tags arbeiten”
2. Wählen Sie Änderungen speichern, um den Keyspace zu aktualisieren.



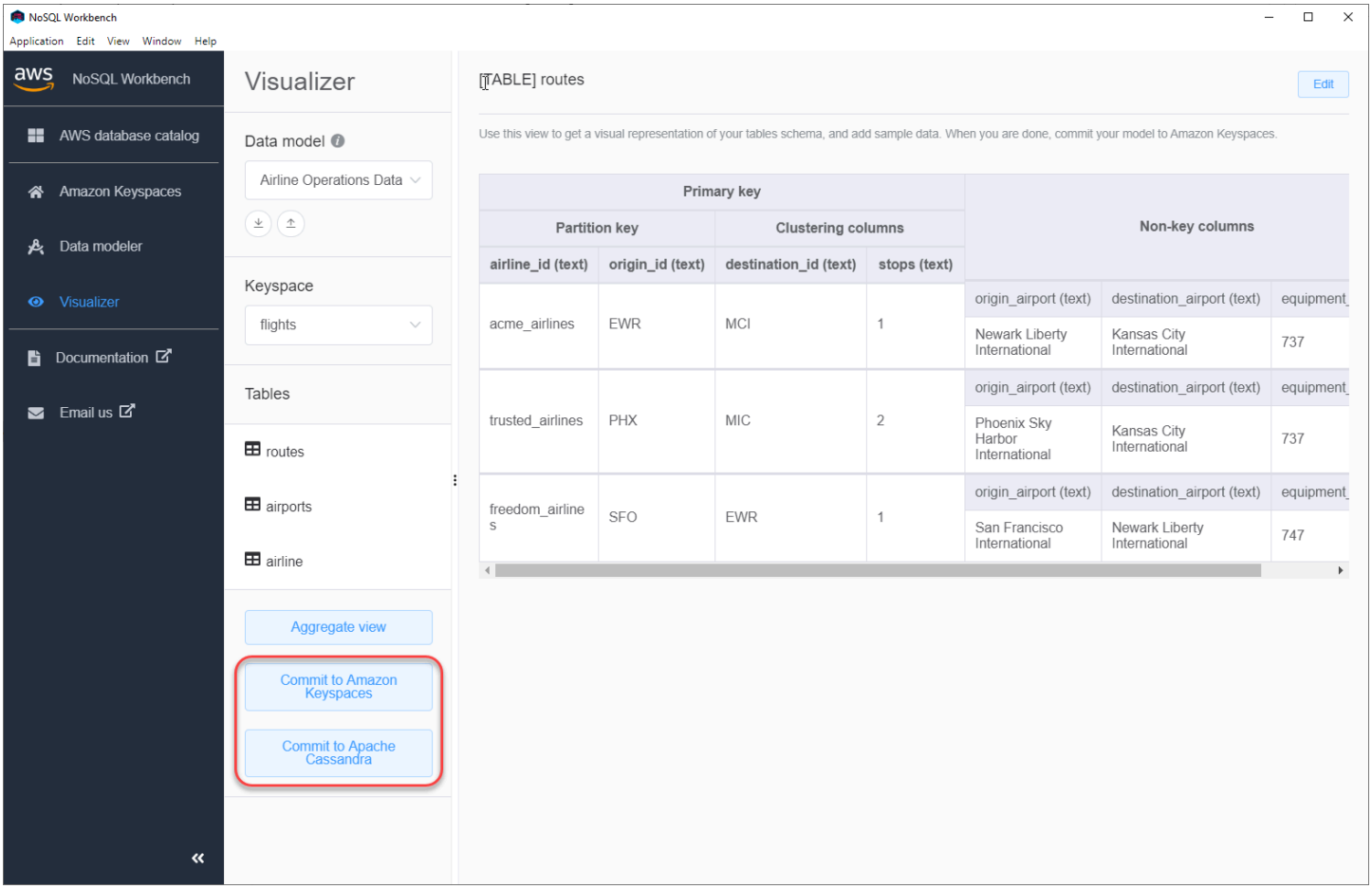
3. Um eine Tabelle zu bearbeiten, wählen Sie neben dem Tabellennamen Bearbeiten aus. In diesem Schritt können Sie die folgenden Eigenschaften und Einstellungen aktualisieren.

   - Tabellenname — Der Name der neuen Tabelle.
   - Spalten — Fügen Sie einen Spaltennamen hinzu und wählen Sie den Datentyp. Wiederholen Sie diese Schritte für jede Spalte in Ihrem Schema.
   - Partitionsschlüssel — Wählen Sie Spalten für den Partitionsschlüssel aus.
   - Gruppierung von Spalten — Wählen Sie Gruppierungsspalten aus (optional).
   - Kapazitätsmodus — Wählen Sie den read/write Kapazitätsmodus für die Tabelle. Sie können zwischen bereitgestellter und bedarfsgesteuerter Kapazität wählen. Weitere Informationen zu Kapazitätsmodi finden Sie unterthe section called “ read/write Kapazitätsmodi konfigurieren”.
   - Tabellen-Tags — Ressourcen-Tags sind optional und ermöglichen es Ihnen, Ihre Ressourcen auf unterschiedliche Weise zu kategorisieren, z. B. nach Zweck, Besitzer, Umgebung oder anderen Kriterien. Weitere Informationen zu Stichwörtern für Amazon Keyspaces-Ressourcen finden Sie unter. the section called “Mit Tags arbeiten”

4. Wählen Sie Änderungen speichern, um die Tabelle zu aktualisieren.
5. Fahren Sie mit the section called “Visualisieren eines Datenmodells” der Visualisierung des Datenmodells fort, das Sie aktualisiert haben.

So übertragen Sie Datenmodelle an Amazon Keyspaces und Apache Cassandra

In diesem Abschnitt erfahren Sie, wie Sie fertige Datenmodelle in Amazon Keyspaces- und Apache Cassandra-Cluster übertragen. Dieser Prozess erstellt automatisch die serverseitigen Ressourcen für Schlüsselräume und Tabellen auf der Grundlage der Einstellungen, die Sie im Datenmodell definiert haben.



Themen

- Bevor Sie beginnen
- Stellen Sie mit dienstspezifischen Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her
- Stellen Sie eine Verbindung zu Amazon Keyspaces her mit AWS Identity and Access Management (IAM) -Anmeldeinformationen
- Verwenden Sie eine gespeicherte Verbindung
- Binden Sie sich für Apache Cassandra ein

Bevor Sie beginnen

Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um die Verbindungen mit Kunden abzusichern. Um mithilfe von TLS eine Verbindung zu Amazon Keyspaces herzustellen, müssen Sie die folgende Aufgabe ausführen, bevor Sie beginnen können.

- Laden Sie die folgenden digitalen Zertifikate herunter und speichern Sie die Dateien lokal oder in Ihrem Home-Verzeichnis.
  1. AmazonRootCA1
  2. AmazonRootCA2
  3. AmazonRootCA3
  4. AmazonRootCA4
  5. Starfield Class 2 Root (optional — aus Gründen der Abwärtskompatibilität)

  Um die Zertifikate herunterzuladen, können Sie die folgenden Befehle verwenden.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA4.pem
curl -O https://certs.secureserver.net/repository/sf-class2-root.crt
```

> **Note**
>
> Amazon Keyspaces verwendete zuvor TLS-Zertifikate, die in der Starfield Class 2-CA verankert waren. AWS migriert alle Zertifikate AWS-Regionen , die unter Amazon Trust Services (Amazon Root CAs 1—4) ausgestellt wurden. Konfigurieren Sie während dieser Umstellung die Clients so, dass sie sowohl Amazon Root CAs 1—4 als auch dem Starfield-Root vertrauen, um die Kompatibilität in allen Regionen sicherzustellen.

  Kombinieren Sie alle heruntergeladenen Zertifikate in einer einzigen `pem` Datei mit dem Namen *`keyspaces-bundle.pem`* in unseren Beispielen. Sie erreichen dies mit dem -Befehl- Notieren Sie sich den Pfad zur Datei, diesen benötigen Sie später.

```
cat AmazonRootCA1.pem \
 AmazonRootCA2.pem \
 AmazonRootCA3.pem \
```

```
AmazonRootCA4.pem \
sf-class2-root.crt \
> keyspaces-bundle.pem
```

Nachdem Sie die Zertifikatsdatei gespeichert haben, können Sie eine Verbindung zu Amazon Keyspaces herstellen. Eine Möglichkeit besteht darin, mithilfe dienstspezifischer Anmeldeinformationen eine Verbindung herzustellen. Service-specific Anmeldeinformationen sind ein Benutzername und ein Passwort, die einem bestimmten IAM-Benutzer zugeordnet sind und nur mit dem angegebenen Dienst verwendet werden können. Die zweite Option besteht darin, eine Verbindung mit IAM-Anmeldeinformationen herzustellen, die den AWS Signature Version 4-Prozess (Sigv4) verwenden. Weitere Informationen zu diesen beiden Optionen finden Sie unter. the section called “Erstellen Sie programmatische Zugangsdaten”

Informationen zum Herstellen einer Verbindung mit dienstspezifischen Anmeldeinformationen finden Sie unterthe section called “Stellen Sie eine Verbindung mit dienstspezifischen Anmeldeinformationen her”.

Informationen zum Herstellen einer Verbindung mit IAM-Anmeldeinformationen finden Sie unter. the section called “Herstellen einer Verbindung unter Verwendung von IAM-Anmeldeinformationen”

Stellen Sie mit dienstspezifischen Anmeldeinformationen eine Verbindung zu Amazon Keyspaces her

In diesem Abschnitt wird gezeigt, wie Sie dienstspezifische Anmeldeinformationen verwenden, um das Datenmodell, das Sie mit NoSQL Workbench erstellt oder bearbeitet haben, zu bestätigen.

1. Um eine neue Verbindung mit dienstspezifischen Anmeldeinformationen herzustellen, wählen Sie die Registerkarte Mit Benutzername und Passwort verbinden.

   - Bevor Sie beginnen, müssen Sie dienstspezifische Anmeldeinformationen mithilfe des unter dokumentierten Verfahrens erstellen. the section called “Erstellen Sie dienstspezifische Anmeldeinformationen”

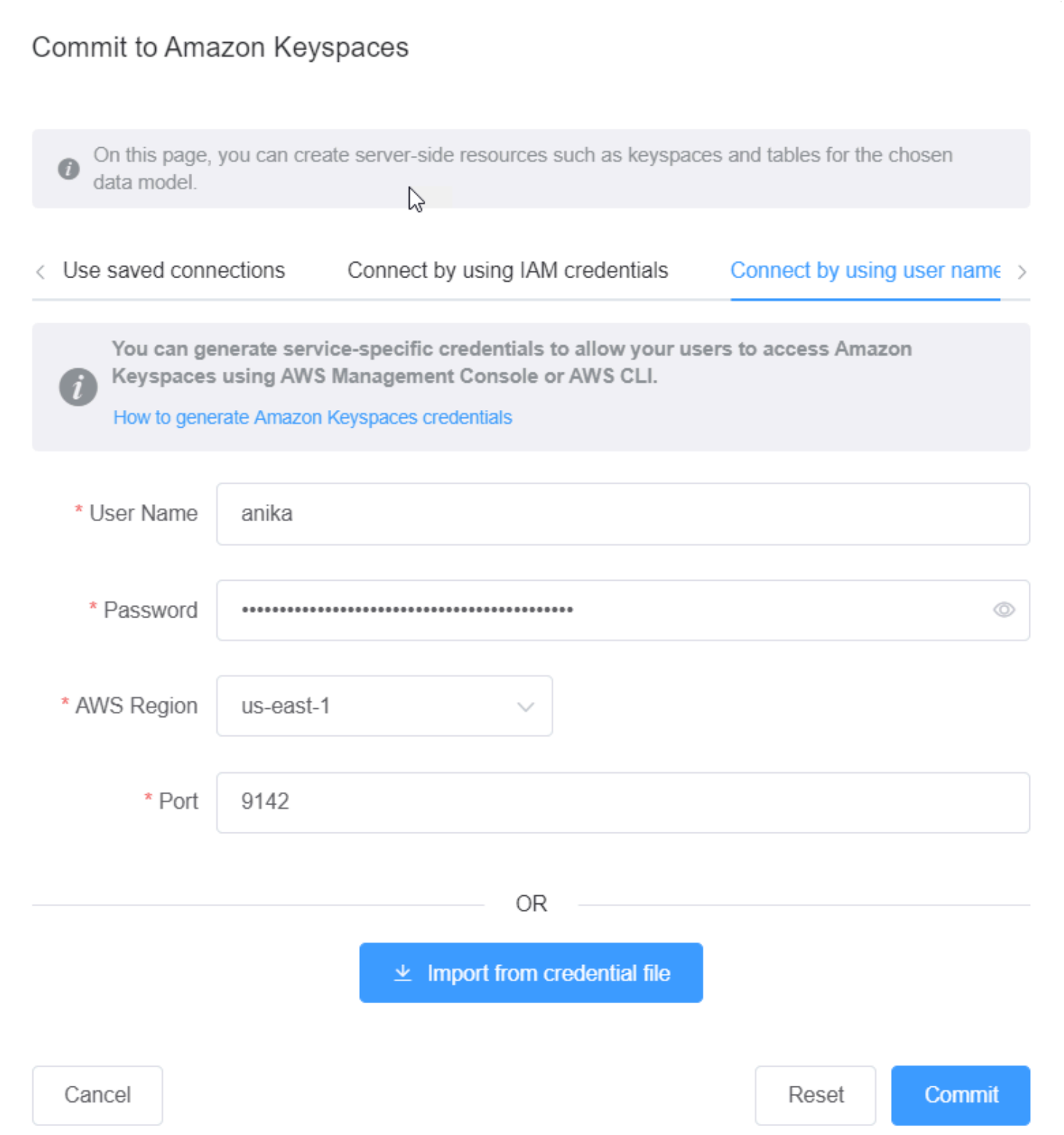
   Nachdem Sie die dienstspezifischen Anmeldeinformationen erhalten haben, können Sie mit dem Einrichten der Verbindung fortfahren. Fahren Sie mit einer der folgenden Aktionen fort:

   - Benutzername — Geben Sie den Benutzernamen ein.
   - Passwort — Geben Sie das Passwort ein.

- AWS-Region— Informationen zu verfügbaren Regionen finden Sie unterthe section called “Service-Endpunkte”.
- Port — Amazon Keyspaces verwendet Port 9142.

Alternativ können Sie gespeicherte Anmeldeinformationen aus einer Datei importieren.

2. Wählen Sie Commit, um Amazon Keyspaces mit dem Datenmodell zu aktualisieren.


Commit to Amazon Keyspaces
On this page, you can create server-side resources such as keyspaces and tables for the chosen data model.
Use saved connections
Connect by using IAM credentials
Connect by using user name
You can generate service-specific credentials to allow your users to access Amazon Keyspaces using AWS Management Console or AWS CLI.
How to generate Amazon Keyspaces credentials
* User Name
anika
* Password
* AWS Region
us-east-1
* Port
9142
OR
Import from credential file
Cancel
Reset
Commit

Stellen Sie eine Verbindung zu Amazon Keyspaces her mit AWS Identity and Access Management (IAM) -Anmeldeinformationen

In diesem Abschnitt wird gezeigt, wie Sie IAM-Anmeldeinformationen verwenden, um das mit NoSQL Workbench erstellte oder bearbeitete Datenmodell festzuschreiben.

1. Um eine neue Verbindung mithilfe von IAM-Anmeldeinformationen herzustellen, wählen Sie die Registerkarte Mit IAM-Anmeldeinformationen verbinden.

   - Bevor Sie beginnen, müssen Sie IAM-Anmeldeinformationen mit einer der folgenden Methoden erstellen.

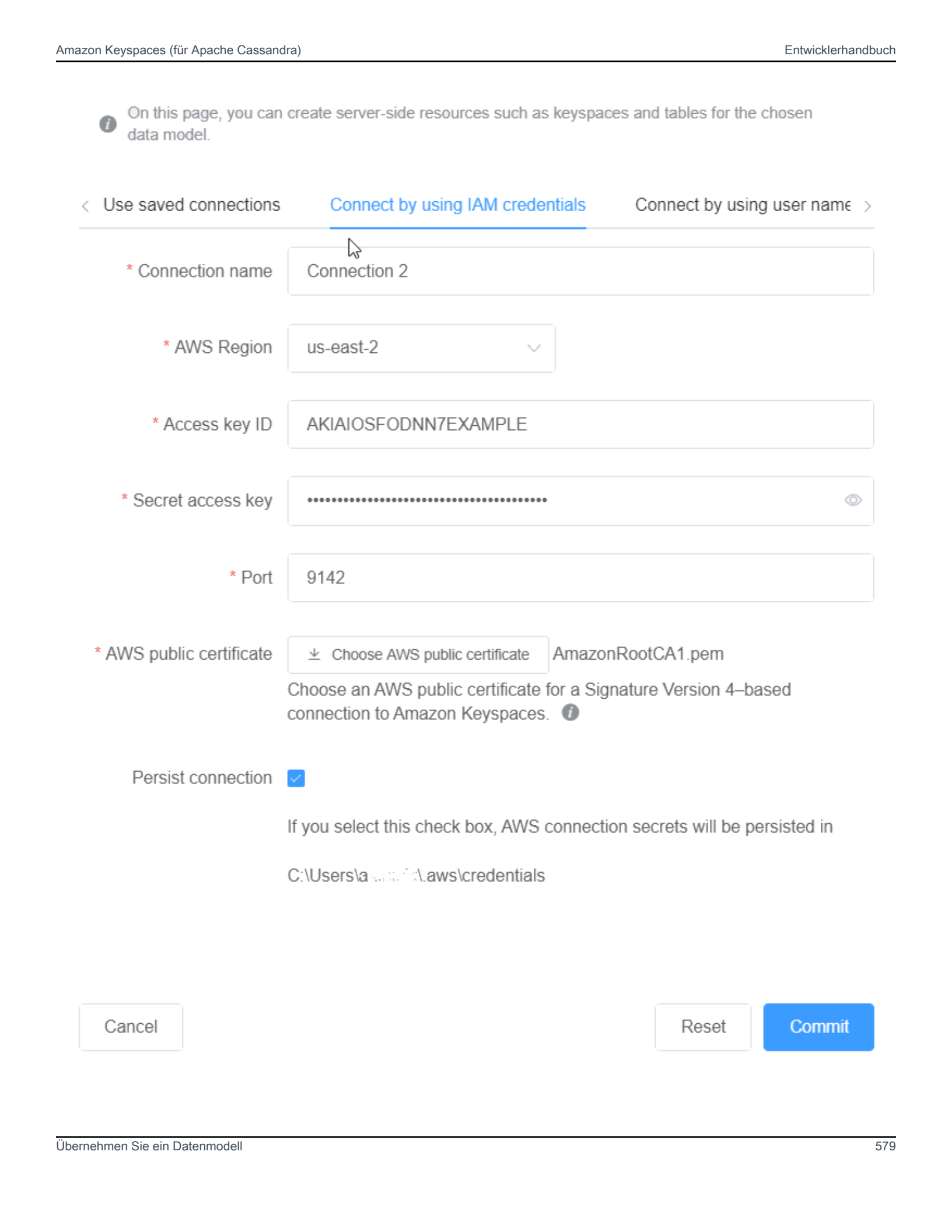
     - Verwenden Sie für den Konsolenzugriff Ihren IAM-Benutzernamen und Ihr Passwort, um sich über die IAM-Anmeldeseite bei AWS-Managementkonsole anzumelden. Informationen zu AWS Sicherheitsanmeldeinformationen, einschließlich programmatischem Zugriff und Alternativen zu langfristigen Anmeldeinformationen, finden Sie im IAM-Benutzerhandbuch unter AWS Sicherheitsanmeldeinformationen. Einzelheiten zur Anmeldung bei Ihrem AWS-Konto finden Sie im AWS-Anmeldung Benutzerhandbuch unter So melden Sie sich AWS an mit.
     - Für CLI-Zugriff benötigen Sie eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel. Verwenden Sie möglichst temporäre Anmeldeinformationen anstelle langfristiger Zugriffsschlüssel. Temporäre Anmeldeinformationen bestehen aus einer Zugriffsschlüssel-ID, einem geheimen Zugriffsschlüssel und einem Sicherheits-Token, das angibt, wann die Anmeldeinformationen ablaufen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Verwenden temporärer Anmeldeinformationen mit AWS Ressourcen.
     - Für den API-Zugriff benötigen Sie eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel. Verwenden Sie IAM-Benutzer-Zugriffsschlüssel anstelle von Root-Benutzer des AWS-Kontos -Zugriffsschlüsseln. Weitere Informationen zum Erstellen von Zugriffsschlüsseln finden Sie im IAM-Benutzerhandbuch unter Zugriffsschlüssel für IAM-Benutzer verwalten.

     Weitere Informationen finden Sie unter Zugriffsschlüssel für IAM-Benutzer verwalten.

   Nachdem Sie die IAM-Anmeldeinformationen erhalten haben, können Sie mit dem Einrichten der Verbindung fortfahren.

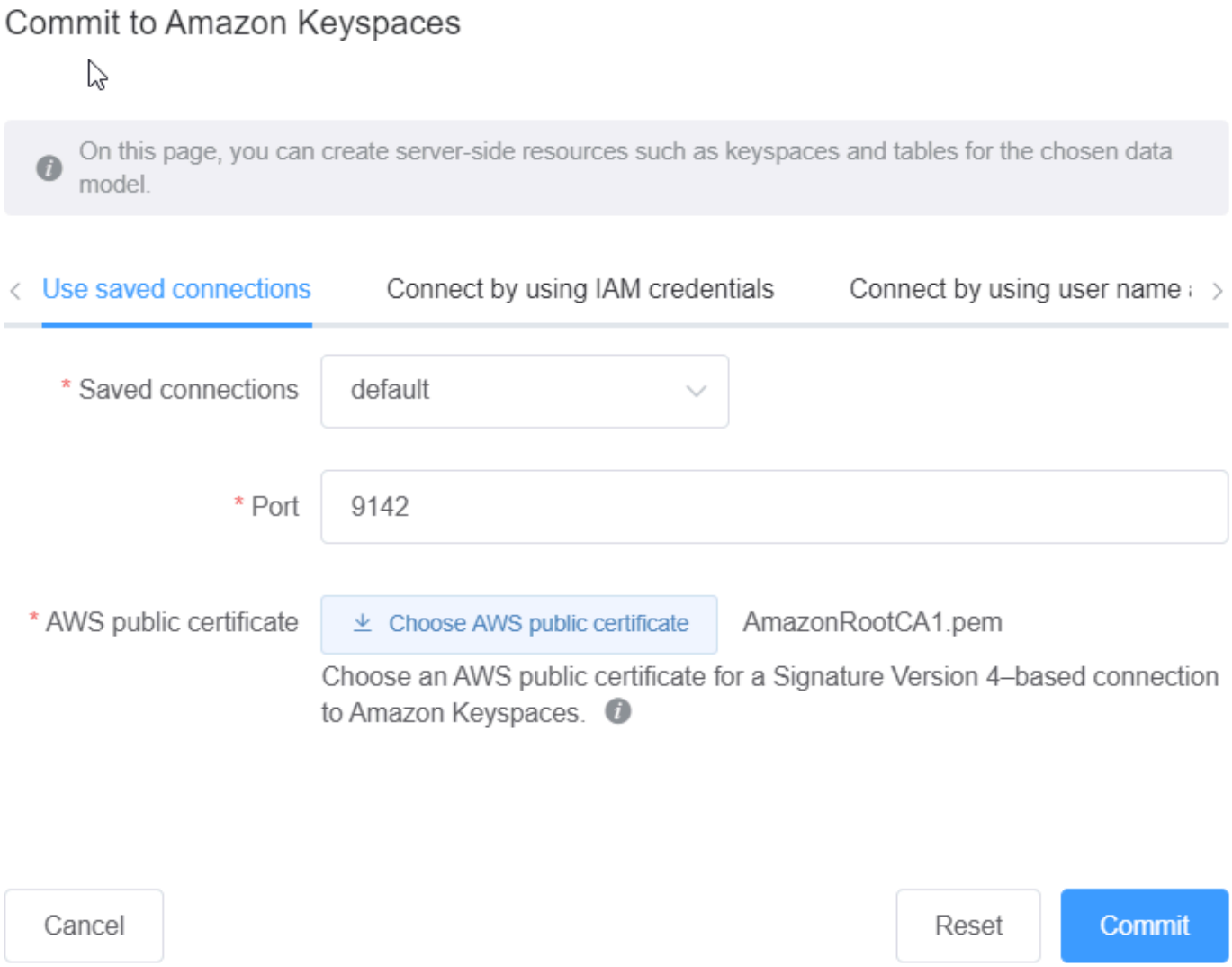
- Verbindungsname — Der Name der Verbindung.
- AWS-Region— Verfügbare Regionen finden Sie unterthe section called “Service-Endpunkte”.
- Zugriffsschlüssel-ID — Geben Sie die Zugriffsschlüssel-ID ein.
- Geheimer Zugangsschlüssel — Geben Sie den geheimen Zugangsschlüssel ein.
- Port — Amazon Keyspaces verwendet Port 9142.
- AWS öffentliches Zertifikat — Verweisen Sie auf das AWS Zertifikat, das im ersten Schritt heruntergeladen wurde.
- Verbindung aufrechterhalten — Wählen Sie dieses Kontrollkästchen, wenn Sie die AWS Verbindungsgeheimnisse lokal speichern möchten.

2. Wählen Sie Commit, um Amazon Keyspaces mit dem Datenmodell zu aktualisieren.


On this page, you can create server-side resources such as keyspaces and tables for the chosen data model.
Use saved connections
Connect by using IAM credentials
Connect by using user name
* Connection name
Connection 2
* AWS Region
us-east-2
* Access key ID
AKIAIOSFODNN7EXAMPLE
* Secret access key
* Port
9142
* AWS public certificate
Choose AWS public certificate
AmazonRootCA1.pem
Choose an AWS public certificate for a Signature Version 4–based connection to Amazon Keyspaces.
Persist connection
If you select this check box, AWS connection secrets will be persisted in
C:\Users\a
\.aws\credentials
Cancel
Reset
Commit

Verwenden Sie eine gespeicherte Verbindung

Wenn Sie zuvor eine Verbindung zu Amazon Keyspaces eingerichtet haben, können Sie diese als Standardverbindung verwenden, um Änderungen am Datenmodell zu bestätigen. Wählen Sie den Tab Gespeicherte Verbindungen verwenden und fahren Sie mit der Übertragung der Aktualisierungen fort.

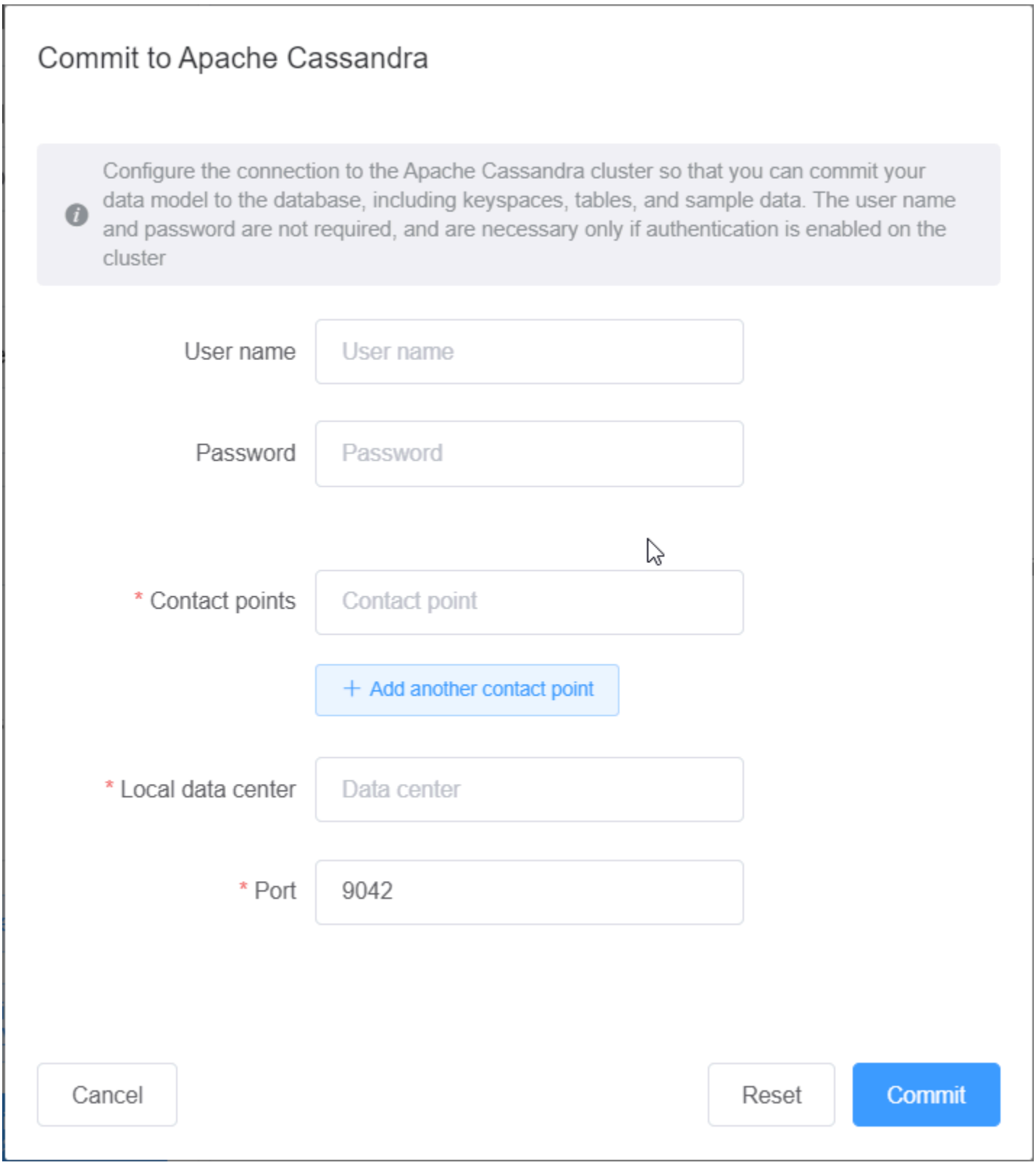


Binden Sie sich für Apache Cassandra ein

In diesem Abschnitt erfahren Sie, wie Sie Verbindungen zu einem Apache Cassandra-Cluster herstellen, um das mit NoSQL Workbench erstellte oder bearbeitete Datenmodell zu übertragen.

> **Note**
>
> Nur Datenmodelle, die mit Apache Cassandra-Clustern erstellt wurden `SimpleStrategy` oder in diese übertragen werden `NetworkTopologyStrategy` können. Um die Replikationsstrategie zu ändern, bearbeiten Sie den Schlüsselraum im Datenmodellierer.

1.
   - Benutzername — Geben Sie den Benutzernamen ein, wenn die Authentifizierung auf dem Cluster aktiviert ist.
   - Passwort — Geben Sie das Passwort ein, wenn die Authentifizierung auf dem Cluster aktiviert ist.
   - Kontaktpunkte — Geben Sie die Kontaktpunkte ein.
   - Lokales Rechenzentrum — Geben Sie den Namen des lokalen Rechenzentrums ein.
   - Port — Die Verbindung verwendet Port 9042.
2. Wählen Sie Commit, um den Apache Cassandra-Cluster mit dem Datenmodell zu aktualisieren.


Commit to Apache Cassandra
Configure the connection to the Apache Cassandra cluster so that you can commit your data model to the database, including keyspaces, tables, and sample data. The user name and password are not required, and are necessary only if authentication is enabled on the cluster
User name
User name
Password
Password
* Contact points
Contact point
+ Add another contact point
* Local data center
Data center
* Port
9042
Cancel
Reset
Commit

Beispieldatenmodelle in NoSQL Workbench

Auf der Startseite für den Modeler und Visualizer werden eine Reihe von Beispielmodellen angezeigt, die im Lieferumfang von NoSQL Workbench enthalten sind. In diesem Abschnitt werden diese Modelle und ihre möglichen Verwendungsmöglichkeiten beschrieben.

Themen

- Mitarbeiterdatenmodell
- Datenmodell für Kreditkartentransaktionen
- Datenmodell für den Flugbetrieb

Mitarbeiterdatenmodell

Dieses Datenmodell stellt ein Amazon Keyspaces-Schema für eine Mitarbeiterdatenbankanwendung dar.

Anwendungen, die auf Mitarbeiterinformationen für ein bestimmtes Unternehmen zugreifen, können dieses Datenmodell verwenden.

Die von diesem Datenmodell unterstützten Zugriffsmuster sind:

- Abrufen eines Mitarbeiterdatensatzes mit einer bestimmten ID.
- Abrufen eines Mitarbeiterdatensatzes mit einer bestimmten ID und Abteilung.
- Abrufen eines Mitarbeiterdatensatzes mit einer bestimmten ID und einem bestimmten Namen.

Datenmodell für Kreditkartentransaktionen

Dieses Datenmodell stellt ein Amazon Keyspaces-Schema für Kreditkartentransaktionen in Einzelhandelsgeschäften dar.

Die Speicherung von Kreditkartentransaktionen hilft den Geschäften nicht nur bei der Buchhaltung, sondern hilft den Filialleitern auch dabei, Kauftrends zu analysieren, was ihnen bei der Prognose und Planung helfen kann.

Die von diesem Datenmodell unterstützten Zugriffsmuster sind:

- Abrufen von Transaktionen nach Kreditkartennummer, Monat und Jahr sowie Datum.
- Abrufen von Transaktionen nach Kreditkartennummer, Kategorie und Datum.

- Abrufen von Transaktionen nach Kategorie, Standort und Kreditkartennummer.
- Abrufen von Transaktionen nach Kreditkartennummer und Streitwert.

Datenmodell für den Flugbetrieb

Dieses Datenmodell zeigt Daten zu Flügen, einschließlich Flughäfen, Fluggesellschaften und Flugrouten.

Zu den wichtigsten Komponenten der Amazon Keyspaces-Modellierung, die demonstriert werden, gehören Schlüssel-Wert-Paare, großspaltige Datenspeicher, zusammengesetzte Schlüssel und komplexe Datentypen wie Maps zur Demonstration gängiger NoSQL-Datenzugriffsmuster.

Die von diesem Datenmodell unterstützten Zugriffsmuster sind:

- Abrufen von Strecken, die von einer bestimmten Fluggesellschaft an einem bestimmten Flughafen ausgehen.
- Abrufen von Strecken mit einem bestimmten Zielflughafen.
- Abruf von Flughäfen mit direkten Flügen.
- Abrufen von Flughafen- und Airline-Daten.

Versionsverlauf für NoSQL-Workbench

In der folgenden Tabelle werden die wichtigen Änderungen in den einzelnen Versionen der clientseitigen NoSQL Workbench-Anwendung beschrieben.

Änderungen	Beschreibung	Date
NoSQL Workbench für Amazon Keyspaces — GA.	NoSQL Workbench für Amazon Keyspaces ist allgemein verfügbar.	28. Oktober 2020
NoSQL-Workbench (Vorversion) veröffentlicht.	NoSQL Workbench ist eine clientseitige Anwendung , mit der Sie nichtrelationale Datenmodelle für Amazon Keyspaces einfacher entwerfen und visualisieren	5. Oktober 2020

Änderungen	Beschreibung	Date
	können. NoSQL Workbench -Clients sind für Windows, macOS und Linux verfügbar. Weitere Informationen finden Sie unter NoSQL Workbench for Amazon Keyspaces.	

Codebeispiele für Amazon Keyspaces mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Amazon Keyspaces mit einem AWS Software Development Kit (SDK) verwendet wird.

Bei Grundlagen handelt es sich um Codebeispiele, die Ihnen zeigen, wie Sie die wesentlichen Vorgänge innerhalb eines Services ausführen.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien anzeigen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Grundlegende Beispiele für Amazon Keyspaces mit AWS SDKs
  - Hello Amazon Keyspaces
  - Lernen Sie die Grundlagen von Amazon Keyspaces mit einem AWS SDK kennen
  - Aktionen für Amazon Keyspaces mithilfe von AWS SDKs
    - Verwenden Sie es CreateKeyspace mit einem AWS SDK
    - CreateTableMit einem AWS SDK verwenden
    - DeleteKeyspaceMit einem AWS SDK verwenden
    - DeleteTableMit einem AWS SDK verwenden
    - GetKeyspaceMit einem AWS SDK verwenden
    - GetTableMit einem AWS SDK verwenden
    - ListKeyspacesMit einem AWS SDK verwenden
    - ListTablesMit einem AWS SDK verwenden
    - RestoreTableMit einem AWS SDK verwenden
    - UpdateTableMit einem AWS SDK verwenden

Grundlegende Beispiele für Amazon Keyspaces mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Grundlagen von Amazon Keyspaces (für Apache Cassandra) mit AWS SDKs verwendet werden.

Beispiele

- Hello Amazon Keyspaces
- Lernen Sie die Grundlagen von Amazon Keyspaces mit einem AWS SDK kennen
- Aktionen für Amazon Keyspaces mithilfe von AWS SDKs
  - Verwenden Sie es CreateKeyspace mit einem AWS SDK
  - CreateTableMit einem AWS SDK verwenden
  - DeleteKeyspaceMit einem AWS SDK verwenden
  - DeleteTableMit einem AWS SDK verwenden
  - GetKeyspaceMit einem AWS SDK verwenden
  - GetTableMit einem AWS SDK verwenden
  - ListKeyspacesMit einem AWS SDK verwenden
  - ListTablesMit einem AWS SDK verwenden
  - RestoreTableMit einem AWS SDK verwenden
  - UpdateTableMit einem AWS SDK verwenden

Hello Amazon Keyspaces

Die folgenden Codebeispiele veranschaulichen, wie Sie mit der Verwendung von Amazon Keyspaces beginnen.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
namespace KeyspacesActions;

public class HelloKeyspaces
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Keyspaces (for Apache
 Cassandra).
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonKeyspaces>()
                .AddTransient<KeyspacesWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<HelloKeyspaces>();

        var keyspacesClient =
 host.Services.GetRequiredService<IAmazonKeyspaces>();
        var keyspacesWrapper = new KeyspacesWrapper(keyspacesClient);

        Console.WriteLine("Hello, Amazon Keyspaces! Let's list your keyspaces:");
        await keyspacesWrapper.ListKeyspaces();
    }
}
```

- Einzelheiten zur API finden Sie ListKeyspacesin der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.KeyspaceSummary;
import software.amazon.awssdk.services.keyspaces.model.KeyspacesException;
import software.amazon.awssdk.services.keyspaces.model.ListKeyspacesRequest;
import software.amazon.awssdk.services.keyspaces.model.ListKeyspacesResponse;
import java.util.List;

/**
 * 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 HelloKeyspaces {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        KeyspacesClient keyClient = KeyspacesClient.builder()
                .region(region)
                .build();

        listKeyspaces(keyClient);
    }

    public static void listKeyspaces(KeyspacesClient keyClient) {
        try {
            ListKeyspacesRequest keyspacesRequest =
 ListKeyspacesRequest.builder()
                    .maxResults(10)
```

```
                    .build();

            ListKeyspacesResponse response =
 keyClient.listKeyspaces(keyspacesRequest);
            List<KeyspaceSummary> keyspaces = response.keyspaces();
            for (KeyspaceSummary keyspace : keyspaces) {
                System.out.println("The name of the keyspace is " +
 keyspace.keyspaceName());
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListKeyspacesin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/**
Before running this Kotlin 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-kotlin/latest/developer-guide/setup.html
*/

suspend fun main() {
    listKeyspaces()
```

```
}

suspend fun listKeyspaces() {
    val keyspacesRequest =
        ListKeyspacesRequest {
            maxResults = 10
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.listKeyspaces(keyspacesRequest)
        response.keyspaces?.forEach { keyspace ->
            println("The name of the keyspace is ${keyspace.keyspaceName}")
        }
    }
}
```

- Einzelheiten zur API finden Sie ListKeyspacesin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import boto3


def hello_keyspaces(keyspaces_client):
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon Keyspaces (for Apache
 Cassandra)
    client and list the keyspaces in your account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param keyspaces_client: A Boto3 Amazon Keyspaces Client object. This object
 wraps
```

```
                            the low-level Amazon Keyspaces service API.
    """
    print("Hello, Amazon Keyspaces! Let's list some of your keyspaces:\n")
    for ks in keyspaces_client.list_keyspaces(maxResults=5).get("keyspaces", []):
        print(ks["keyspaceName"])
        print(f"\t{ks['resourceArn']}")


if __name__ == "__main__":
    hello_keyspaces(boto3.client("keyspaces"))
```

- Einzelheiten zur API finden Sie ListKeyspacesin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Lernen Sie die Grundlagen von Amazon Keyspaces mit einem AWS SDK kennen

Die folgenden Code-Beispiele veranschaulichen Folgendes:

- Erstellen Sie einen Keyspace und eine Tabelle. Das Tabellenschema enthält Filmdaten und die zeitpunktbezogene Wiederherstellung ist aktiviert.
- Stellen Sie über eine sichere TLS-Verbindung mit SigV4-Authentifizierung eine Verbindung zum Keyspace her.
- Fragen Sie die Tabelle ab. Fügen Sie Filmdaten hinzu, rufen Sie sie ab und aktualisieren Sie sie.
- Aktualisieren Sie die Tabelle. Fügen Sie eine Spalte hinzu, um bereits gesehene Filme zu verfolgen.
- Stellen Sie den vorherigen Zustand der Tabelle wieder her und bereinigen Sie die Ressourcen.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
global using System.Security.Cryptography.X509Certificates;
global using Amazon.Keyspaces;
global using Amazon.Keyspaces.Model;
global using KeyspacesActions;
global using KeyspacesScenario;
global using Microsoft.Extensions.Configuration;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;
global using Newtonsoft.Json;


namespace KeyspacesBasics;

/// <summary>
/// Amazon Keyspaces (for Apache Cassandra) scenario. Shows some of the basic
/// actions performed with Amazon Keyspaces.
/// </summary>
public class KeyspacesBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
```

```
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonKeyspaces>()
            .AddTransient<KeyspacesWrapper>()
            .AddTransient<CassandraWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<KeyspacesBasics>();

        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var keyspacesWrapper =
host.Services.GetRequiredService<KeyspacesWrapper>();
        var uiMethods = new UiMethods();

        var keyspaceName = configuration["KeyspaceName"];
        var tableName = configuration["TableName"];

        bool success; // Used to track the results of some operations.

        uiMethods.DisplayOverview();
        uiMethods.PressEnter();

        // Create the keyspace.
        var keyspaceArn = await keyspacesWrapper.CreateKeyspace(keyspaceName);

        // Wait for the keyspace to be available. GetKeyspace results in a
        // resource not found error until it is ready for use.
        try
        {
            var getKeyspaceArn = "";
            Console.Write($"Created {keyspaceName}. Waiting for it to become
available. ");
            do
            {
```

```
                getKeyspaceArn = await
keyspacesWrapper.GetKeyspace(keyspaceName);
                Console.Write(". ");
            } while (getKeyspaceArn != keyspaceArn);
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine("Waiting for keyspace to be created.");
        }

        Console.WriteLine($"\nThe keyspace {keyspaceName} is ready for use.");

        uiMethods.PressEnter();

        // Create the table.
        // First define the schema.
        var allColumns = new List<ColumnDefinition>
        {
            new ColumnDefinition { Name = "title", Type = "text" },
            new ColumnDefinition { Name = "year", Type = "int" },
            new ColumnDefinition { Name = "release_date", Type = "timestamp" },
            new ColumnDefinition { Name = "plot", Type = "text" },
        };

        var partitionKeys = new List<PartitionKey>
        {
            new PartitionKey { Name = "year", },
            new PartitionKey { Name = "title" },
        };

        var tableSchema = new SchemaDefinition
        {
            AllColumns = allColumns,
            PartitionKeys = partitionKeys,
        };

        var tableArn = await keyspacesWrapper.CreateTable(keyspaceName,
tableSchema, tableName);

        // Wait for the table to be active.
        try
        {
            var resp = new GetTableResponse();
            Console.Write("Waiting for the new table to be active. ");
```

```
            do
            {
                try
                {
                    resp = await keyspacesWrapper.GetTable(keyspaceName,
tableName);
                    Console.Write(".");
                }
                catch (ResourceNotFoundException)
                {
                    Console.Write(".");
                }
            } while (resp.Status != TableStatus.ACTIVE);

            // Display the table's schema.
            Console.WriteLine($"\nTable {tableName} has been created in
{keyspaceName}");
            Console.WriteLine("Let's take a look at the schema.");
            uiMethods.DisplayTitle("All columns");
            resp.SchemaDefinition.AllColumns.ForEach(column =>
            {
                Console.WriteLine($"{column.Name,-40}\t{column.Type,-20}");
            });

            uiMethods.DisplayTitle("Cluster keys");
            resp.SchemaDefinition.ClusteringKeys.ForEach(clusterKey =>
            {

Console.WriteLine($"{clusterKey.Name,-40}\t{clusterKey.OrderBy,-20}");
            });

            uiMethods.DisplayTitle("Partition keys");
            resp.SchemaDefinition.PartitionKeys.ForEach(partitionKey =>
            {
                Console.WriteLine($"{partitionKey.Name}");
            });

            uiMethods.PressEnter();
        }
        catch (ResourceNotFoundException ex)
        {
            Console.WriteLine($"Error: {ex.Message}");
        }
```

```
        // Access Apache Cassandra using the Cassandra drive for C#.
        var cassandraWrapper =
host.Services.GetRequiredService<CassandraWrapper>();
        var movieFilePath = configuration["MovieFile"];

        Console.WriteLine("Let's add some movies to the table we created.");
        var inserted = await cassandraWrapper.InsertIntoMovieTable(keyspaceName,
tableName, movieFilePath);

        uiMethods.PressEnter();

        Console.WriteLine("Added the following movies to the table:");
        var rows = await cassandraWrapper.GetMovies(keyspaceName, tableName);
        uiMethods.DisplayTitle("All Movies");

        foreach (var row in rows)
        {
            var title = row.GetValue<string>("title");
            var year = row.GetValue<int>("year");
            var plot = row.GetValue<string>("plot");
            var release_date = row.GetValue<DateTime>("release_date");
            Console.WriteLine($"{release_date}\t{title}\t{year}\n{plot}");
            Console.WriteLine(uiMethods.SepBar);
        }

        // Update the table schema
        uiMethods.DisplayTitle("Update table schema");
        Console.WriteLine("Now we will update the table to add a boolean field
called watched.");

        // First save the current time as a UTC Date so the original
        // table can be restored later.
        var timeChanged = DateTime.UtcNow;

        // Now update the schema.
        var resourceArn = await keyspacesWrapper.UpdateTable(keyspaceName,
tableName);
        uiMethods.PressEnter();

        Console.WriteLine("Now let's mark some of the movies as watched.");

        // Pick some files to mark as watched.
        var movieToWatch = rows[2].GetValue<string>("title");
        var watchedMovieYear = rows[2].GetValue<int>("year");
```

```
        var changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[6].GetValue<string>("title");
        watchedMovieYear = rows[6].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[9].GetValue<string>("title");
        watchedMovieYear = rows[9].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[10].GetValue<string>("title");
        watchedMovieYear = rows[10].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        movieToWatch = rows[13].GetValue<string>("title");
        watchedMovieYear = rows[13].GetValue<int>("year");
        changedRows = await cassandraWrapper.MarkMovieAsWatched(keyspaceName,
tableName, movieToWatch, watchedMovieYear);

        uiMethods.DisplayTitle("Watched movies");
        Console.WriteLine("These movies have been marked as watched:");
        rows = await cassandraWrapper.GetWatchedMovies(keyspaceName, tableName);
        foreach (var row in rows)
        {
            var title = row.GetValue<string>("title");
            var year = row.GetValue<int>("year");
            Console.WriteLine($"{title,-40}\t{year,8}");
        }
        uiMethods.PressEnter();

        Console.WriteLine("We can restore the table to its previous state but
that can take up to 20 minutes to complete.");
        string answer;
        do
        {
            Console.WriteLine("Do you want to restore the table? (y/n)");
            answer = Console.ReadLine();
        } while (answer.ToLower() != "y" && answer.ToLower() != "n");

        if (answer == "y")
```

```
        {
            var restoredTableName = $"{tableName}_restored";
            var restoredTableArn = await keyspacesWrapper.RestoreTable(
                keyspaceName,
                tableName,
                restoredTableName,
                timeChanged);
            // Loop and call GetTable until the table is gone. Once it has been
            // deleted completely, GetTable will raise a
ResourceNotFoundException.
            bool wasRestored = false;

            try
            {
                do
                {
                    var resp = await keyspacesWrapper.GetTable(keyspaceName,
restoredTableName);
                    wasRestored = (resp.Status == TableStatus.ACTIVE);
                } while (!wasRestored);
            }
            catch (ResourceNotFoundException)
            {
                // If the restored table raised an error, it isn't
                // ready yet.
                Console.Write(".");
            }
        }

        uiMethods.DisplayTitle("Clean up resources.");

        // Delete the table.
        success = await keyspacesWrapper.DeleteTable(keyspaceName, tableName);

        Console.WriteLine($"Table {tableName} successfully deleted from
{keyspaceName}.");
        Console.WriteLine("Waiting for the table to be removed completely. ");

        // Loop and call GetTable until the table is gone. Once it has been
        // deleted completely, GetTable will raise a ResourceNotFoundException.
        bool wasDeleted = false;

        try
        {
```

```
            do
            {
                var resp = await keyspacesWrapper.GetTable(keyspaceName,
 tableName);
            } while (!wasDeleted);
        }
        catch (ResourceNotFoundException ex)
        {
            wasDeleted = true;
            Console.WriteLine($"{ex.Message} indicates that the table has been
 deleted.");
        }

        // Delete the keyspace.
        success = await keyspacesWrapper.DeleteKeyspace(keyspaceName);
        Console.WriteLine("The keyspace has been deleted and the demo is now
 complete.");
    }
}
```

```
namespace KeyspacesActions;

/// <summary>
/// Performs Amazon Keyspaces (for Apache Cassandra) actions.
/// </summary>
public class KeyspacesWrapper
{
    private readonly IAmazonKeyspaces _amazonKeyspaces;

    /// <summary>
    /// Constructor for the KeyspaceWrapper.
    /// </summary>
    /// <param name="amazonKeyspaces">An Amazon Keyspaces client object.</param>
    public KeyspacesWrapper(IAmazonKeyspaces amazonKeyspaces)
    {
        _amazonKeyspaces = amazonKeyspaces;
    }

    /// <summary>
    /// Create a new keyspace.
    /// </summary>
```

```
    /// <param name="keyspaceName">The name for the new keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new keyspace.</returns>
    public async Task<string> CreateKeyspace(string keyspaceName)
    {
        var response =
            await _amazonKeyspaces.CreateKeyspaceAsync(
                new CreateKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }


    /// <summary>
    /// Create a new Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace where the table will be
created.</param>
    /// <param name="schema">The schema for the new table.</param>
    /// <param name="tableName">The name of the new table.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new table.</returns>
    public async Task<string> CreateTable(string keyspaceName, SchemaDefinition
schema, string tableName)
    {
        var request = new CreateTableRequest
        {
            KeyspaceName = keyspaceName,
            SchemaDefinition = schema,
            TableName = tableName,
            PointInTimeRecovery = new PointInTimeRecovery { Status =
PointInTimeRecoveryStatus.ENABLED }
        };

        var response = await _amazonKeyspaces.CreateTableAsync(request);
        return response.ResourceArn;
    }


    /// <summary>
    /// Delete an existing keyspace.
    /// </summary>
    /// <param name="keyspaceName"></param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyspace(string keyspaceName)
    {
        var response = await _amazonKeyspaces.DeleteKeyspaceAsync(
```

```
            new DeleteKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTable(string keyspaceName, string tableName)
    {
        var response = await _amazonKeyspaces.DeleteTableAsync(
            new DeleteTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Get data about a keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name of the keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the keyspace.</returns>
    public async Task<string> GetKeyspace(string keyspaceName)
    {
        var response = await _amazonKeyspaces.GetKeyspaceAsync(
            new GetKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }


    /// <summary>
    /// Get information about an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the Amazon Keyspaces table.</param>
    /// <returns>The response containing data about the table.</returns>
    public async Task<GetTableResponse> GetTable(string keyspaceName, string
tableName)
    {
        var response = await _amazonKeyspaces.GetTableAsync(
```

```
            new GetTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response;
    }


    /// <summary>
    /// Lists all keyspaces for the account.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task ListKeyspaces()
    {
        var paginator = _amazonKeyspaces.Paginators.ListKeyspaces(new
ListKeyspacesRequest());

        Console.WriteLine("{0, -30}\t{1}", "Keyspace name", "Keyspace ARN");
        Console.WriteLine(new string('-', Console.WindowWidth));
        await foreach (var keyspace in paginator.Keyspaces)
        {

Console.WriteLine($"{keyspace.KeyspaceName,-30}\t{keyspace.ResourceArn}");
        }
    }


    /// <summary>
    /// Lists the Amazon Keyspaces tables in a keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name of the keyspace.</param>
    /// <returns>A list of TableSummary objects.</returns>
    public async Task<List<TableSummary>> ListTables(string keyspaceName)
    {
        var response = await _amazonKeyspaces.ListTablesAsync(new
ListTablesRequest { KeyspaceName = keyspaceName });
        response.Tables.ForEach(table =>
        {

Console.WriteLine($"{table.KeyspaceName}\t{table.TableName}\t{table.ResourceArn}");
        });

        return response.Tables;
    }
```

```
    /// <summary>
    /// Restores the specified table to the specified point in time.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to restore.</param>
    /// <param name="timestamp">The time to which the table will be restored.</
param>
    /// <returns>The Amazon Resource Name (ARN) of the restored table.</returns>
    public async Task<string> RestoreTable(string keyspaceName, string tableName,
 string restoredTableName, DateTime timestamp)
    {
        var request = new RestoreTableRequest
        {
            RestoreTimestamp = timestamp,
            SourceKeyspaceName = keyspaceName,
            SourceTableName = tableName,
            TargetKeyspaceName = keyspaceName,
            TargetTableName = restoredTableName
        };

        var response = await _amazonKeyspaces.RestoreTableAsync(request);
        return response.RestoredTableARN;
    }


    /// <summary>
    /// Updates the movie table to add a boolean column named watched.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to change.</param>
    /// <returns>The Amazon Resource Name (ARN) of the updated table.</returns>
    public async Task<string> UpdateTable(string keyspaceName, string tableName)
    {
        var newColumn = new ColumnDefinition { Name = "watched", Type =
 "boolean" };
        var request = new UpdateTableRequest
        {
            KeyspaceName = keyspaceName,
            TableName = tableName,
            AddColumns = new List<ColumnDefinition> { newColumn }
        };
        var response = await _amazonKeyspaces.UpdateTableAsync(request);
        return response.ResourceArn;
    }
```

```
}
```

```
using System.Net;
using Cassandra;

namespace KeyspacesScenario;

/// <summary>
/// Class to perform CRUD methods on an Amazon Keyspaces (for Apache Cassandra)
 database.
///
/// NOTE: This sample uses a plain text authenticator for example purposes only.
/// Recommended best practice is to use a SigV4 authentication plugin, if
 available.
/// </summary>
public class CassandraWrapper
{
    private readonly IConfiguration _configuration;
    private readonly string _localPathToFile;
    private const string _certLocation = "https://certs.secureserver.net/
repository/sf-class2-root.crt";
    private const string _certFileName = "sf-class2-root.crt";
    private readonly X509Certificate2Collection _certCollection;
    private X509Certificate2 _amazoncert;
    private Cluster _cluster;

    // User name and password for the service.
    private string _userName = null!;
    private string _pwd = null!;

    public CassandraWrapper()
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        _localPathToFile = Path.GetTempPath();
```

```
        // Get the Starfield digital certificate and save it locally.
        var client = new WebClient();
        client.DownloadFile(_certLocation, $"{_localPathToFile}/
{_certFileName}");

        //var httpClient = new HttpClient();
        //var httpResult = httpClient.Get(fileUrl);
        //using var resultStream = await httpResult.Content.ReadAsStreamAsync();
        //using var fileStream = File.Create(pathToSave);
        //resultStream.CopyTo(fileStream);

        _certCollection = new X509Certificate2Collection();
        _amazoncert = new X509Certificate2($"{_localPathToFile}/
{_certFileName}");

        // Get the user name and password stored in the configuration file.
        _userName = _configuration["UserName"]!;
        _pwd = _configuration["Password"]!;

        // For a list of Service Endpoints for Amazon Keyspaces, see:
        // https://docs.aws.amazon.com/keyspaces/latest/devguide/
programmatic.endpoints.html
        var awsEndpoint = _configuration["ServiceEndpoint"];

        _cluster = Cluster.Builder()
            .AddContactPoints(awsEndpoint)
            .WithPort(9142)
            .WithAuthProvider(new PlainTextAuthProvider(_userName, _pwd))
            .WithSSL(new SSLOptions().SetCertificateCollection(_certCollection))
            .WithQueryOptions(
                new QueryOptions()
                    .SetConsistencyLevel(ConsistencyLevel.LocalQuorum)
                    .SetSerialConsistencyLevel(ConsistencyLevel.LocalSerial))
            .Build();
    }

    /// <summary>
    /// Loads the contents of a JSON file into a list of movies to be
    /// added to the Apache Cassandra table.
    /// </summary>
    /// <param name="movieFileName">The full path to the JSON file.</param>
    /// <returns>A list of movie objects.</returns>
```

```
    public List<Movie> ImportMoviesFromJson(string movieFileName, int numToImport
= 0)
    {
        if (!File.Exists(movieFileName))
        {
            return null!;
        }

        using var sr = new StreamReader(movieFileName);
        string json = sr.ReadToEnd();

        var allMovies = JsonConvert.DeserializeObject<List<Movie>>(json);

        // If numToImport = 0, return all movies in the collection.
        if (numToImport == 0)
        {
            // Now return the entire list of movies.
            return allMovies;
        }
        else
        {
            // Now return the first numToImport entries.
            return allMovies.GetRange(0, numToImport);
        }
    }

    /// <summary>
    /// Insert movies into the movie table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="movieTableName">The Amazon Keyspaces table.</param>
    /// <param name="movieFilePath">The path to the resource file containing
    /// movie data to insert into the table.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> InsertIntoMovieTable(string keyspaceName, string
movieTableName, string movieFilePath, int numToImport = 20)
    {
        // Get some movie data from the movies.json file
        var movies = ImportMoviesFromJson(movieFilePath, numToImport);

        var session = _cluster.Connect(keyspaceName);

        string insertCql;
```

```
        RowSet rs;

        // Now we insert the numToImport movies into the table.
        foreach (var movie in movies)
        {
            // Escape single quote characters in the plot.
            insertCql = $"INSERT INTO {keyspaceName}.{movieTableName}
(title, year, release_date, plot) values($${movie.Title}$$, {movie.Year},
'{movie.Info.Release_Date.ToString("yyyy-MM-dd")}', $${movie.Info.Plot}$$)";
            rs = await session.ExecuteAsync(new SimpleStatement(insertCql));
        }

        return true;
    }

    /// <summary>
    /// Gets all of the movies in the movies table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table.</param>
    /// <returns>A list of row objects containing movie data.</returns>
    public async Task<List<Row>> GetMovies(string keyspaceName, string tableName)
    {
        var session = _cluster.Connect();
        RowSet rs;
        try
        {
            rs = await session.ExecuteAsync(new SimpleStatement($"SELECT * FROM
{keyspaceName}.{tableName}"));

            // Extract the row data from the returned RowSet.
            var rows = rs.GetRows().ToList();
            return rows;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
            return null!;
        }
    }

    /// <summary>
    /// Mark a movie in the movie table as watched.
    /// </summary>
```

```
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table.</param>
    /// <param name="title">The title of the movie to mark as watched.</param>
    /// <param name="year">The year the movie was released.</param>
    /// <returns>A set of rows containing the changed data.</returns>
    public async Task<List<Row>> MarkMovieAsWatched(string keyspaceName, string
tableName, string title, int year)
    {
        var session = _cluster.Connect();
        string updateCql = $"UPDATE {keyspaceName}.{tableName} SET watched=true
WHERE title = $${title}$$ AND year = {year};";
        var rs = await session.ExecuteAsync(new SimpleStatement(updateCql));
        var rows = rs.GetRows().ToList();
        return rows;
    }

    /// <summary>
    /// Retrieve the movies in the movies table where watched is true.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table.</param>
    /// <returns>A list of row objects containing information about movies
    /// where watched is true.</returns>
    public async Task<List<Row>> GetWatchedMovies(string keyspaceName, string
tableName)
    {
        var session = _cluster.Connect();
        RowSet rs;
        try
        {
            rs = await session.ExecuteAsync(new SimpleStatement($"SELECT
title, year, plot FROM {keyspaceName}.{tableName} WHERE watched = true ALLOW
FILTERING"));

            // Extract the row data from the returned RowSet.
            var rows = rs.GetRows().ToList();
            return rows;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
            return null!;
        }
    }
```

```
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der AWS SDK für .NET - API-Referenz.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/**
 * 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
 *
 * Before running this Java code example, you must create a
```

```
 * Java keystore (JKS) file and place it in your project's resources folder.
 *
 * This file is a secure file format used to hold certificate information for
 * Java applications. This is required to make a connection to Amazon Keyspaces.
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/keyspaces/latest/devguide/using_java_driver.html
 *
 * This Java example performs the following tasks:
 *
 * 1. Create a keyspace.
 * 2. Check for keyspace existence.
 * 3. List keyspaces using a paginator.
 * 4. Create a table with a simple movie data schema and enable point-in-time
 * recovery.
 * 5. Check for the table to be in an Active state.
 * 6. List all tables in the keyspace.
 * 7. Use a Cassandra driver to insert some records into the Movie table.
 * 8. Get all records from the Movie table.
 * 9. Get a specific Movie.
 * 10. Get a UTC timestamp for the current time.
 * 11. Update the table schema to add a ‘watched’ Boolean column.
 * 12. Update an item as watched.
 * 13. Query for items with watched = True.
 * 14. Restore the table back to the previous state using the timestamp.
 * 15. Check for completion of the restore action.
 * 16. Delete the table.
 * 17. Confirm that both tables are deleted.
 * 18. Delete the keyspace.
 */

public class ScenarioKeyspaces {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    /*
     * Usage:
     * fileName - The name of the JSON file that contains movie data. (Get this
 file
     * from the GitHub repo at resources/sample_file.)
     * keyspaceName - The name of the keyspace to create.
     */
    public static void main(String[] args) throws InterruptedException,
 IOException {
```

```
        String fileName = "<Replace with the JSON file that contains movie
data>";
        String keyspaceName = "<Replace with the name of the keyspace to
create>";
        String titleUpdate = "The Family";
        int yearUpdate = 2013;
        String tableName = "Movie";
        String tableNameRestore = "MovieRestore";
        Region region = Region.US_EAST_1;
        KeyspacesClient keyClient = KeyspacesClient.builder()
                .region(region)
                .build();

        DriverConfigLoader loader =
DriverConfigLoader.fromClasspath("application.conf");
        CqlSession session = CqlSession.builder()
                .withConfigLoader(loader)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon Keyspaces example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create a keyspace.");
        createKeySpace(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        Thread.sleep(5000);
        System.out.println("2. Check for keyspace existence.");
        checkKeyspaceExistence(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. List keyspaces using a paginator.");
        listKeyspacesPaginator(keyClient);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Create a table with a simple movie data schema and
enable point-in-time recovery.");
        createTable(keyClient, keyspaceName, tableName);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("5. Check for the table to be in an Active state.");
        Thread.sleep(6000);
        checkTable(keyClient, keyspaceName, tableName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. List all tables in the keyspace.");
        listTables(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Use a Cassandra driver to insert some records into
the Movie table.");
        Thread.sleep(6000);
        loadData(session, fileName, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Get all records from the Movie table.");
        getMovieData(session, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. Get a specific Movie.");
        getSpecificMovie(session, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. Get a UTC timestamp for the current time.");
        ZonedDateTime utc = ZonedDateTime.now(ZoneOffset.UTC);
        System.out.println("DATETIME = " + Date.from(utc.toInstant()));
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Update the table schema to add a watched Boolean
column.");
        updateTable(keyClient, keyspaceName, tableName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Update an item as watched.");
        Thread.sleep(10000); // Wait 10 secs for the update.
```

```
        updateRecord(session, keyspaceName, titleUpdate, yearUpdate);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Query for items with watched = True.");
        getWatchedData(session, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Restore the table back to the previous state
using the timestamp.");
        System.out.println("Note that the restore operation can take up to 20
minutes.");
        restoreTable(keyClient, keyspaceName, utc);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("15. Check for completion of the restore action.");
        Thread.sleep(5000);
        checkRestoredTable(keyClient, keyspaceName, "MovieRestore");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("16. Delete both tables.");
        deleteTable(keyClient, keyspaceName, tableName);
        deleteTable(keyClient, keyspaceName, tableNameRestore);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Confirm that both tables are deleted.");
        checkTableDelete(keyClient, keyspaceName, tableName);
        checkTableDelete(keyClient, keyspaceName, tableNameRestore);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("18. Delete the keyspace.");
        deleteKeyspace(keyClient, keyspaceName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The scenario has completed successfully.");
        System.out.println(DASHES);
    }
```

```
   public static void deleteKeyspace(KeyspacesClient keyClient, String
keyspaceName) {
       try {
           DeleteKeyspaceRequest deleteKeyspaceRequest =
DeleteKeyspaceRequest.builder()
                   .keyspaceName(keyspaceName)
                   .build();

           keyClient.deleteKeyspace(deleteKeyspaceRequest);

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void checkTableDelete(KeyspacesClient keyClient, String
keyspaceName, String tableName)
           throws InterruptedException {
       try {
           String status;
           GetTableResponse response;
           GetTableRequest tableRequest = GetTableRequest.builder()
                   .keyspaceName(keyspaceName)
                   .tableName(tableName)
                   .build();

           // Keep looping until table cannot be found and a
ResourceNotFoundException is
           // thrown.
           while (true) {
               response = keyClient.getTable(tableRequest);
               status = response.statusAsString();
               System.out.println(". The table status is " + status);
               Thread.sleep(500);
           }

       } catch (ResourceNotFoundException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
       }
       System.out.println("The table is deleted");
   }
```

```
   public static void deleteTable(KeyspacesClient keyClient, String
keyspaceName, String tableName) {
       try {
           DeleteTableRequest tableRequest = DeleteTableRequest.builder()
                   .keyspaceName(keyspaceName)
                   .tableName(tableName)
                   .build();

           keyClient.deleteTable(tableRequest);

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void checkRestoredTable(KeyspacesClient keyClient, String
keyspaceName, String tableName)
           throws InterruptedException {
       try {
           boolean tableStatus = false;
           String status;
           GetTableResponse response = null;
           GetTableRequest tableRequest = GetTableRequest.builder()
                   .keyspaceName(keyspaceName)
                   .tableName(tableName)
                   .build();

           while (!tableStatus) {
               response = keyClient.getTable(tableRequest);
               status = response.statusAsString();
               System.out.println("The table status is " + status);

               if (status.compareTo("ACTIVE") == 0) {
                   tableStatus = true;
               }
               Thread.sleep(500);
           }

           List<ColumnDefinition> cols =
response.schemaDefinition().allColumns();
           for (ColumnDefinition def : cols) {
               System.out.println("The column name is " + def.name());
               System.out.println("The column type is " + def.type());
```

```
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void restoreTable(KeyspacesClient keyClient, String
keyspaceName, ZonedDateTime utc) {
        try {
            Instant myTime = utc.toInstant();
            RestoreTableRequest restoreTableRequest =
RestoreTableRequest.builder()
                    .restoreTimestamp(myTime)
                    .sourceTableName("Movie")
                    .targetKeyspaceName(keyspaceName)
                    .targetTableName("MovieRestore")
                    .sourceKeyspaceName(keyspaceName)
                    .build();

            RestoreTableResponse response =
keyClient.restoreTable(restoreTableRequest);
            System.out.println("The ARN of the restored table is " +
response.restoredTableARN());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getWatchedData(CqlSession session, String keyspaceName) {
        ResultSet resultSet = session
                .execute("SELECT * FROM \"" + keyspaceName + "\".\"Movie\" WHERE
watched = true ALLOW FILTERING;");
        resultSet.forEach(item -> {
            System.out.println("The Movie title is " + item.getString("title"));
            System.out.println("The Movie year is " + item.getInt("year"));
            System.out.println("The plot is " + item.getString("plot"));
        });
    }
```

```
   public static void updateRecord(CqlSession session, String keySpace, String
titleUpdate, int yearUpdate) {
       String sqlStatement = "UPDATE \"" + keySpace
               + "\".\"Movie\" SET watched=true WHERE title = :k0 AND year
= :k1;";
       BatchStatementBuilder builder =
BatchStatement.builder(DefaultBatchType.UNLOGGED);
       builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM);
       PreparedStatement preparedStatement = session.prepare(sqlStatement);
       builder.addStatement(preparedStatement.boundStatementBuilder()
               .setString("k0", titleUpdate)
               .setInt("k1", yearUpdate)
               .build());

       BatchStatement batchStatement = builder.build();
       session.execute(batchStatement);
   }

   public static void updateTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
       try {
           ColumnDefinition def = ColumnDefinition.builder()
                   .name("watched")
                   .type("boolean")
                   .build();

           UpdateTableRequest tableRequest = UpdateTableRequest.builder()
                   .keyspaceName(keySpace)
                   .tableName(tableName)
                   .addColumns(def)
                   .build();

           keyClient.updateTable(tableRequest);

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void getSpecificMovie(CqlSession session, String keyspaceName)
{
       ResultSet resultSet = session.execute(
```

```
                "SELECT * FROM \"" + keyspaceName + "\".\"Movie\" WHERE title =
'The Family' ALLOW FILTERING ;");
        resultSet.forEach(item -> {
            System.out.println("The Movie title is " + item.getString("title"));
            System.out.println("The Movie year is " + item.getInt("year"));
            System.out.println("The plot is " + item.getString("plot"));
        });
    }

    // Get records from the Movie table.
    public static void getMovieData(CqlSession session, String keyspaceName) {
        ResultSet resultSet = session.execute("SELECT * FROM \"" + keyspaceName +
"\".\"Movie\";");
        resultSet.forEach(item -> {
            System.out.println("The Movie title is " + item.getString("title"));
            System.out.println("The Movie year is " + item.getInt("year"));
            System.out.println("The plot is " + item.getString("plot"));
        });
    }

    // Load data into the table.
    public static void loadData(CqlSession session, String fileName, String
keySpace) throws IOException {
        String sqlStatement = "INSERT INTO \"" + keySpace + "\".\"Movie\" (title,
year, plot) values (:k0, :k1, :k2)";
        JsonParser parser = new JsonFactory().createParser(new File(fileName));
        com.fasterxml.jackson.databind.JsonNode rootNode = new
ObjectMapper().readTree(parser);
        Iterator<JsonNode> iter = rootNode.iterator();
        ObjectNode currentNode;
        int t = 0;
        while (iter.hasNext()) {

            // Add 20 movies to the table.
            if (t == 20)
                break;
            currentNode = (ObjectNode) iter.next();

            int year = currentNode.path("year").asInt();
            String title = currentNode.path("title").asText();
            String plot = currentNode.path("info").path("plot").toString();

            // Insert the data into the Amazon Keyspaces table.
```

```
            BatchStatementBuilder builder =
BatchStatement.builder(DefaultBatchType.UNLOGGED);
            builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM);
            PreparedStatement preparedStatement = session.prepare(sqlStatement);
            builder.addStatement(preparedStatement.boundStatementBuilder()
                    .setString("k0", title)
                    .setInt("k1", year)
                    .setString("k2", plot)
                    .build());

            BatchStatement batchStatement = builder.build();
            session.execute(batchStatement);
            t++;
        }

        System.out.println("You have added " + t + " records successfully!");
    }

    public static void listTables(KeyspacesClient keyClient, String keyspaceName)
{
        try {
            ListTablesRequest tablesRequest = ListTablesRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            ListTablesIterable listRes =
keyClient.listTablesPaginator(tablesRequest);
            listRes.stream()
                    .flatMap(r -> r.tables().stream())
                    .forEach(content -> System.out.println(" ARN: " +
content.resourceArn() +
                            " Table name: " + content.tableName()));

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkTable(KeyspacesClient keyClient, String keyspaceName,
String tableName)
            throws InterruptedException {
        try {
            boolean tableStatus = false;
```

```
            String status;
            GetTableResponse response = null;
            GetTableRequest tableRequest = GetTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            while (!tableStatus) {
                response = keyClient.getTable(tableRequest);
                status = response.statusAsString();
                System.out.println(". The table status is " + status);

                if (status.compareTo("ACTIVE") == 0) {
                    tableStatus = true;
                }
                Thread.sleep(500);
            }

            List<ColumnDefinition> cols =
response.schemaDefinition().allColumns();
            for (ColumnDefinition def : cols) {
                System.out.println("The column name is " + def.name());
                System.out.println("The column type is " + def.type());
            }

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
        try {
            // Set the columns.
            ColumnDefinition defTitle = ColumnDefinition.builder()
                    .name("title")
                    .type("text")
                    .build();

            ColumnDefinition defYear = ColumnDefinition.builder()
                    .name("year")
                    .type("int")
                    .build();
```

```
        ColumnDefinition defReleaseDate = ColumnDefinition.builder()
                .name("release_date")
                .type("timestamp")
                .build();

        ColumnDefinition defPlot = ColumnDefinition.builder()
                .name("plot")
                .type("text")
                .build();

        List<ColumnDefinition> colList = new ArrayList<>();
        colList.add(defTitle);
        colList.add(defYear);
        colList.add(defReleaseDate);
        colList.add(defPlot);

        // Set the keys.
        PartitionKey yearKey = PartitionKey.builder()
                .name("year")
                .build();

        PartitionKey titleKey = PartitionKey.builder()
                .name("title")
                .build();

        List<PartitionKey> keyList = new ArrayList<>();
        keyList.add(yearKey);
        keyList.add(titleKey);

        SchemaDefinition schemaDefinition = SchemaDefinition.builder()
                .partitionKeys(keyList)
                .allColumns(colList)
                .build();

        PointInTimeRecovery timeRecovery = PointInTimeRecovery.builder()
                .status(PointInTimeRecoveryStatus.ENABLED)
                .build();

        CreateTableRequest tableRequest = CreateTableRequest.builder()
                .keyspaceName(keySpace)
                .tableName(tableName)
                .schemaDefinition(schemaDefinition)
                .pointInTimeRecovery(timeRecovery)
```

```
                    .build();

            CreateTableResponse response = keyClient.createTable(tableRequest);
            System.out.println("The table ARN is " + response.resourceArn());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void listKeyspacesPaginator(KeyspacesClient keyClient) {
        try {
            ListKeyspacesRequest keyspacesRequest =
ListKeyspacesRequest.builder()
                    .maxResults(10)
                    .build();

            ListKeyspacesIterable listRes =
keyClient.listKeyspacesPaginator(keyspacesRequest);
            listRes.stream()
                    .flatMap(r -> r.keyspaces().stream())
                    .forEach(content -> System.out.println(" Name: " +
content.keyspaceName()));

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkKeyspaceExistence(KeyspacesClient keyClient, String
keyspaceName) {
        try {
            GetKeyspaceRequest keyspaceRequest = GetKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            GetKeyspaceResponse response =
keyClient.getKeyspace(keyspaceRequest);
            String name = response.keyspaceName();
            System.out.println("The " + name + " KeySpace is ready");

        } catch (KeyspacesException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createKeySpace(KeyspacesClient keyClient, String
 keyspaceName) {
        try {
            CreateKeyspaceRequest keyspaceRequest =
 CreateKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            CreateKeyspaceResponse response =
 keyClient.createKeyspace(keyspaceRequest);
            System.out.println("The ARN of the KeySpace is " +
 response.resourceArn());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der AWS SDK for Java 2.x - API-Referenz.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/**
 Before running this Kotlin 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-kotlin/latest/developer-guide/setup.html

 This example uses a secure file format to hold certificate information for
 Kotlin applications. This is required to make a connection to Amazon Keyspaces.
 For more information, see the following documentation topic:

 https://docs.aws.amazon.com/keyspaces/latest/devguide/using_java_driver.html

 This Kotlin example performs the following tasks:

 1. Create a keyspace.
 2. Check for keyspace existence.
 3. List keyspaces using a paginator.
 4. Create a table with a simple movie data schema and enable point-in-time
 recovery.
 5. Check for the table to be in an Active state.
 6. List all tables in the keyspace.
 7. Use a Cassandra driver to insert some records into the Movie table.
 8. Get all records from the Movie table.
 9. Get a specific Movie.
 10. Get a UTC timestamp for the current time.
 11. Update the table schema to add a ‘watched’ Boolean column.
 12. Update an item as watched.
 13. Query for items with watched = True.
 14. Restore the table back to the previous state using the timestamp.
```

```
 15. Check for completion of the restore action.
 16. Delete the table.
 17. Confirm that both tables are deleted.
 18. Delete the keyspace.
 */

/*
   Usage:
     fileName - The name of the JSON file that contains movie data. (Get this
 file from the GitHub repo at resources/sample_file.)
     keyspaceName - The name of the keyspace to create.
 */
val DASHES: String = String(CharArray(80)).replace("\u0000", "-")

suspend fun main() {
    val fileName = "<Replace with the JSON file that contains movie data>"
    val keyspaceName = "<Replace with the name of the keyspace to create>"
    val titleUpdate = "The Family"
    val yearUpdate = 2013
    val tableName = "MovieKotlin"
    val tableNameRestore = "MovieRestore"

    val loader = DriverConfigLoader.fromClasspath("application.conf")
    val session =
        CqlSession
            .builder()
            .withConfigLoader(loader)
            .build()

    println(DASHES)
    println("Welcome to the Amazon Keyspaces example scenario.")
    println(DASHES)

    println(DASHES)
    println("1. Create a keyspace.")
    createKeySpace(keyspaceName)
    println(DASHES)

    println(DASHES)
    delay(5000)
    println("2. Check for keyspace existence.")
    checkKeyspaceExistence(keyspaceName)
    println(DASHES)
```

```
    println(DASHES)
    println("3. List keyspaces using a paginator.")
    listKeyspacesPaginator()
    println(DASHES)

    println(DASHES)
    println("4. Create a table with a simple movie data schema and enable point-
in-time recovery.")
    createTable(keyspaceName, tableName)
    println(DASHES)

    println(DASHES)
    println("5. Check for the table to be in an Active state.")
    delay(6000)
    checkTable(keyspaceName, tableName)
    println(DASHES)

    println(DASHES)
    println("6. List all tables in the keyspace.")
    listTables(keyspaceName)
    println(DASHES)

    println(DASHES)
    println("7. Use a Cassandra driver to insert some records into the Movie
 table.")
    delay(6000)
    loadData(session, fileName, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("8. Get all records from the Movie table.")
    getMovieData(session, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("9. Get a specific Movie.")
    getSpecificMovie(session, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("10. Get a UTC timestamp for the current time.")
    val utc = ZonedDateTime.now(ZoneOffset.UTC)
    println("DATETIME = ${Date.from(utc.toInstant())}")
    println(DASHES)
```

```
    println(DASHES)
    println("11. Update the table schema to add a watched Boolean column.")
    updateTable(keyspaceName, tableName)
    println(DASHES)

    println(DASHES)
    println("12. Update an item as watched.")
    delay(10000) // Wait 10 seconds for the update.
    updateRecord(session, keyspaceName, titleUpdate, yearUpdate)
    println(DASHES)

    println(DASHES)
    println("13. Query for items with watched = True.")
    getWatchedData(session, keyspaceName)
    println(DASHES)

    println(DASHES)
    println("14. Restore the table back to the previous state using the
timestamp.")
    println("Note that the restore operation can take up to 20 minutes.")
    restoreTable(keyspaceName, utc)
    println(DASHES)

    println(DASHES)
    println("15. Check for completion of the restore action.")
    delay(5000)
    checkRestoredTable(keyspaceName, "MovieRestore")
    println(DASHES)

    println(DASHES)
    println("16. Delete both tables.")
    deleteTable(keyspaceName, tableName)
    deleteTable(keyspaceName, tableNameRestore)
    println(DASHES)

    println(DASHES)
    println("17. Confirm that both tables are deleted.")
    checkTableDelete(keyspaceName, tableName)
    checkTableDelete(keyspaceName, tableNameRestore)
    println(DASHES)

    println(DASHES)
    println("18. Delete the keyspace.")
```

```
    deleteKeyspace(keyspaceName)
    println(DASHES)

    println(DASHES)
    println("The scenario has completed successfully.")
    println(DASHES)
}

suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest =
        DeleteKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}

suspend fun checkTableDelete(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var status: String
    var response: GetTableResponse
    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    try {
        KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient
 ->
            // Keep looping until the table cannot be found and a
 ResourceNotFoundException is thrown.
            while (true) {
                response = keyClient.getTable(tableRequest)
                status = response.status.toString()
                println(". The table status is $status")
                delay(500)
            }
        }
    } catch (e: ResourceNotFoundException) {
```

```
        println(e.message)
    }
    println("The table is deleted")
}

suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}

suspend fun checkRestoredTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println("The table status is $status")

            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
```

```
        }

        val cols = response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}

suspend fun restoreTable(
    keyspaceName: String?,
    utc: ZonedDateTime,
) {
    // Create an aws.smithy.kotlin.runtime.time.Instant value.
    val timeStamp =
        aws.smithy.kotlin.runtime.time
            .Instant(utc.toInstant())
    val restoreTableRequest =
        RestoreTableRequest {
            restoreTimestamp = timeStamp
            sourceTableName = "MovieKotlin"
            targetKeyspaceName = keyspaceName
            targetTableName = "MovieRestore"
            sourceKeyspaceName = keyspaceName
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.restoreTable(restoreTableRequest)
        println("The ARN of the restored table is ${response.restoredTableArn}")
    }
}

fun getWatchedData(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet = session.execute("SELECT * FROM \"$keyspaceName\".
\"MovieKotlin\" WHERE watched = true ALLOW FILTERING;")
    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
```

```
        println("The plot is ${item.getString("plot")}")
    }
}

fun updateRecord(
    session: CqlSession,
    keySpace: String,
    titleUpdate: String?,
    yearUpdate: Int,
) {
    val sqlStatement =
        "UPDATE \"$keySpace\".\"MovieKotlin\" SET watched=true WHERE title = :k0
 AND year = :k1;"
    val builder = BatchStatement.builder(DefaultBatchType.UNLOGGED)
    builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
    val preparedStatement = session.prepare(sqlStatement)
    builder.addStatement(
        preparedStatement
            .boundStatementBuilder()
            .setString("k0", titleUpdate)
            .setInt("k1", yearUpdate)
            .build(),
    )
    val batchStatement = builder.build()
    session.execute(batchStatement)
}

suspend fun updateTable(
    keySpace: String?,
    tableNameVal: String?,
) {
    val def =
        ColumnDefinition {
            name = "watched"
            type = "boolean"
        }

    val tableRequest =
        UpdateTableRequest {
            keyspaceName = keySpace
            tableName = tableNameVal
            addColumns = listOf(def)
        }
```

```
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.updateTable(tableRequest)
    }
}

fun getSpecificMovie(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet =
        session.execute("SELECT * FROM \"$keyspaceName\".\"MovieKotlin\" WHERE
 title = 'The Family' ALLOW FILTERING ;")

    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
    }
}

// Get records from the Movie table.
fun getMovieData(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet = session.execute("SELECT * FROM \"$keyspaceName\".
\"MovieKotlin\";")
    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
    }
}

// Load data into the table.
fun loadData(
    session: CqlSession,
    fileName: String,
    keySpace: String,
) {
    val sqlStatement =
        "INSERT INTO \"$keySpace\".\"MovieKotlin\" (title, year, plot) values
 (:k0, :k1, :k2)"
    val parser = JsonFactory().createParser(File(fileName))
```

```
    val rootNode = ObjectMapper().readTree<JsonNode>(parser)
    val iter: Iterator<JsonNode> = rootNode.iterator()
    var currentNode: ObjectNode

    var t = 0
    while (iter.hasNext()) {
        if (t == 50) {
            break
        }

        currentNode = iter.next() as ObjectNode
        val year = currentNode.path("year").asInt()
        val title = currentNode.path("title").asText()
        val info = currentNode.path("info").toString()

        // Insert the data into the Amazon Keyspaces table.
        val builder = BatchStatement.builder(DefaultBatchType.UNLOGGED)
        builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
        val preparedStatement: PreparedStatement = session.prepare(sqlStatement)
        builder.addStatement(
            preparedStatement
                .boundStatementBuilder()
                .setString("k0", title)
                .setInt("k1", year)
                .setString("k2", info)
                .build(),
        )

        val batchStatement = builder.build()
        session.execute(batchStatement)
        t++
    }
}

suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest =
        ListTablesRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
```

```
            .collect { obj ->
                println(" ARN: ${obj.resourceArn} Table name: ${obj.tableName}")
            }
    }
}

suspend fun checkTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println(". The table status is $status")
            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
        val cols: List<ColumnDefinition>? =
 response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}

suspend fun createTable(
    keySpaceVal: String?,
    tableNameVal: String?,
) {
```

```
    // Set the columns.
    val defTitle =
        ColumnDefinition {
            name = "title"
            type = "text"
        }

    val defYear =
        ColumnDefinition {
            name = "year"
            type = "int"
        }

    val defReleaseDate =
        ColumnDefinition {
            name = "release_date"
            type = "timestamp"
        }

    val defPlot =
        ColumnDefinition {
            name = "plot"
            type = "text"
        }

    val colList = ArrayList<ColumnDefinition>()
    colList.add(defTitle)
    colList.add(defYear)
    colList.add(defReleaseDate)
    colList.add(defPlot)

    // Set the keys.
    val yearKey =
        PartitionKey {
            name = "year"
        }

    val titleKey =
        PartitionKey {
            name = "title"
        }

    val keyList = ArrayList<PartitionKey>()
    keyList.add(yearKey)
```

```
    keyList.add(titleKey)

    val schemaDefinitionOb =
        SchemaDefinition {
            partitionKeys = keyList
            allColumns = colList
        }

    val timeRecovery =
        PointInTimeRecovery {
            status = PointInTimeRecoveryStatus.Enabled
        }

    val tableRequest =
        CreateTableRequest {
            keyspaceName = keySpaceVal
            tableName = tableNameVal
            schemaDefinition = schemaDefinitionOb
            pointInTimeRecovery = timeRecovery
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createTable(tableRequest)
        println("The table ARN is ${response.resourceArn}")
    }
}

suspend fun listKeyspacesPaginator() {
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listKeyspacesPaginated(ListKeyspacesRequest {})
            .transform { it.keyspaces?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println("Name: ${obj.keyspaceName}")
            }
    }
}

suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyspaceRequest =
        GetKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
```

```
        val response: GetKeyspaceResponse =
 keyClient.getKeyspace(keyspaceRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}

suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest =
        CreateKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Führen Sie ein interaktives Szenario an einem Prompt aus.

```
class KeyspaceScenario:
    """Runs an interactive scenario that shows how to get started using Amazon
 Keyspaces."""

    def __init__(self, ks_wrapper):
        """
        :param ks_wrapper: An object that wraps Amazon Keyspace actions.
        """
        self.ks_wrapper = ks_wrapper

    @demo_func
    def create_keyspace(self):
        """
        1. Creates a keyspace.
        2. Lists up to 10 keyspaces in your account.
        """
        print("Let's create a keyspace.")
        ks_name = q.ask(
            "Enter a name for your new keyspace.\nThe name can contain only
 letters, "
            "numbers and underscores: ",
            q.non_empty,
        )
        if self.ks_wrapper.exists_keyspace(ks_name):
            print(f"A keyspace named {ks_name} exists.")
        else:
            ks_arn = self.ks_wrapper.create_keyspace(ks_name)
            ks_exists = False
            while not ks_exists:
                wait(3)
                ks_exists = self.ks_wrapper.exists_keyspace(ks_name)
```

```
            print(f"Created a new keyspace.\n\t{ks_arn}.")
        print("The first 10 keyspaces in your account are:\n")
        self.ks_wrapper.list_keyspaces(10)

    @demo_func
    def create_table(self):
        """
        1. Creates a table in the keyspace. The table is configured with a schema
to hold
           movie data and has point-in-time recovery enabled.
        2. Waits for the table to be in an active state.
        3. Displays schema information for the table.
        4. Lists tables in the keyspace.
        """
        print("Let's create a table for movies in your keyspace.")
        table_name = q.ask("Enter a name for your table: ", q.non_empty)
        table = self.ks_wrapper.get_table(table_name)
        if table is not None:
            print(
                f"A table named {table_name} already exists in keyspace "
                f"{self.ks_wrapper.ks_name}."
            )
        else:
            table_arn = self.ks_wrapper.create_table(table_name)
            print(f"Created table {table_name}:\n\t{table_arn}")
            table = {"status": None}
            print("Waiting for your table to be ready...")
            while table["status"] != "ACTIVE":
                wait(5)
                table = self.ks_wrapper.get_table(table_name)
        print(f"Your table is {table['status']}. Its schema is:")
        pp(table["schemaDefinition"])
        print("\nThe tables in your keyspace are:\n")
        self.ks_wrapper.list_tables()

    @demo_func
    def ensure_tls_cert(self):
        """
        Ensures you have a TLS certificate available to use to secure the
connection
        to the keyspace. This function downloads a default certificate or lets
you
        specify your own.
        """
```

```
        print("To connect to your keyspace, you must have a TLS certificate.")
        print("Checking for TLS certificate...")
        cert_path = os.path.join(
            os.path.dirname(__file__), QueryManager.DEFAULT_CERT_FILE
        )
        if not os.path.exists(cert_path):
            cert_choice = q.ask(
                f"Press enter to download a certificate from
{QueryManager.CERT_URL} "
                f"or enter the full path to the certificate you want to use: "
            )
            if cert_choice:
                cert_path = cert_choice
            else:
                cert = requests.get(QueryManager.CERT_URL).text
                with open(cert_path, "w") as cert_file:
                    cert_file.write(cert)
        else:
            q.ask(f"Certificate {cert_path} found. Press Enter to continue.")
        print(
            f"Certificate {cert_path} will be used to secure the connection to
your keyspace."
        )
        return cert_path

    @demo_func
    def query_table(self, qm, movie_file):
        """
        1. Adds movies to the table from a sample movie data file.
        2. Gets a list of movies from the table and lets you select one.
        3. Displays more information about the selected movie.
        """
        qm.add_movies(self.ks_wrapper.table_name, movie_file)
        movies = qm.get_movies(self.ks_wrapper.table_name)
        print(f"Added {len(movies)} movies to the table:")
        sel = q.choose("Pick one to learn more about it: ", [m.title for m in
movies])
        movie_choice = qm.get_movie(
            self.ks_wrapper.table_name, movies[sel].title, movies[sel].year
        )
        print(movie_choice.title)
        print(f"\tReleased: {movie_choice.release_date}")
        print(f"\tPlot: {movie_choice.plot}")
```

```
    @demo_func
    def update_and_restore_table(self, qm):
        """
        1. Updates the table by adding a column to track watched movies.
        2. Marks some of the movies as watched.
        3. Gets the list of watched movies from the table.
        4. Restores to a movies_restored table at a previous point in time.
        5. Gets the list of movies from the restored table.
        """
        print("Let's add a column to record which movies you've watched.")
        pre_update_timestamp = datetime.utcnow()
        print(
            f"Recorded the current UTC time of {pre_update_timestamp} so we can
restore the table later."
        )
        self.ks_wrapper.update_table()
        print("Waiting for your table to update...")
        table = {"status": "UPDATING"}
        while table["status"] != "ACTIVE":
            wait(5)
            table = self.ks_wrapper.get_table(self.ks_wrapper.table_name)
        print("Column 'watched' added to table.")
        q.ask(
            "Let's mark some of the movies as watched. Press Enter when you're
ready.\n"
        )
        movies = qm.get_movies(self.ks_wrapper.table_name)
        for movie in movies[:10]:
            qm.watched_movie(self.ks_wrapper.table_name, movie.title, movie.year)
            print(f"Marked {movie.title} as watched.")
        movies = qm.get_movies(self.ks_wrapper.table_name, watched=True)
        print("-" * 88)
        print("The watched movies in our table are:\n")
        for movie in movies:
            print(movie.title)
        print("-" * 88)
        if q.ask(
            "Do you want to restore the table to the way it was before all of
these\n"
            "updates? Keep in mind, this can take up to 20 minutes. (y/n) ",
            q.is_yesno,
        ):
            starting_table_name = self.ks_wrapper.table_name
```

```
            table_name_restored =
self.ks_wrapper.restore_table(pre_update_timestamp)
            table = {"status": "RESTORING"}
            while table["status"] != "ACTIVE":
                wait(10)
                table = self.ks_wrapper.get_table(table_name_restored)
            print(
                f"Restored {starting_table_name} to {table_name_restored} "
                f"at a point in time of {pre_update_timestamp}."
            )
            movies = qm.get_movies(table_name_restored)
            print("Now the movies in our table are:")
            for movie in movies:
                print(movie.title)

    def cleanup(self, cert_path):
        """
        1. Deletes the table and waits for it to be removed.
        2. Deletes the keyspace.

        :param cert_path: The path of the TLS certificate used in the demo. If
the
                          certificate was downloaded during the demo, it is
removed.
        """
        if q.ask(
            f"Do you want to delete your {self.ks_wrapper.table_name} table and "
            f"{self.ks_wrapper.ks_name} keyspace? (y/n) ",
            q.is_yesno,
        ):
            table_name = self.ks_wrapper.table_name
            self.ks_wrapper.delete_table()
            table = self.ks_wrapper.get_table(table_name)
            print("Waiting for the table to be deleted.")
            while table is not None:
                wait(5)
                table = self.ks_wrapper.get_table(table_name)
            print("Table deleted.")
            self.ks_wrapper.delete_keyspace()
            print(
                "Keyspace deleted. If you chose to restore your table during the
"
                "demo, the original table is also deleted."
            )
```

```
            if cert_path == os.path.join(
                os.path.dirname(__file__), QueryManager.DEFAULT_CERT_FILE
            ) and os.path.exists(cert_path):
                os.remove(cert_path)
                print("Removed certificate that was downloaded for this demo.")

    def run_scenario(self):
        logging.basicConfig(level=logging.INFO, format="%(levelname)s:
 %(message)s")

        print("-" * 88)
        print("Welcome to the Amazon Keyspaces (for Apache Cassandra) demo.")
        print("-" * 88)

        self.create_keyspace()
        self.create_table()
        cert_file_path = self.ensure_tls_cert()
        # Use a context manager to ensure the connection to the keyspace is
 closed.
        with QueryManager(
            cert_file_path, boto3.DEFAULT_SESSION, self.ks_wrapper.ks_name
        ) as qm:
            self.query_table(qm, "../../../resources/sample_files/movies.json")
            self.update_and_restore_table(qm)
        self.cleanup(cert_file_path)

        print("\nThanks for watching!")
        print("-" * 88)


if __name__ == "__main__":
    try:
        scenario = KeyspaceScenario(KeyspaceWrapper.from_client())
        scenario.run_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

Definieren Sie eine Klasse, die Keyspace- und Tabellenaktionen umschließt.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""
```

```
    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def create_keyspace(self, name):
        """
        Creates a keyspace.

        :param name: The name to give the keyspace.
        :return: The Amazon Resource Name (ARN) of the new keyspace.
        """
        try:
            response = self.keyspaces_client.create_keyspace(keyspaceName=name)
            self.ks_name = name
            self.ks_arn = response["resourceArn"]
        except ClientError as err:
            logger.error(
                "Couldn't create %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.ks_arn


    def exists_keyspace(self, name):
        """
        Checks whether a keyspace exists.

        :param name: The name of the keyspace to look up.
```

```
        :return: True when the keyspace exists. Otherwise, False.
        """
        try:
            response = self.keyspaces_client.get_keyspace(keyspaceName=name)
            self.ks_name = response["keyspaceName"]
            self.ks_arn = response["resourceArn"]
            exists = True
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Keyspace %s does not exist.", name)
                exists = False
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return exists


    def list_keyspaces(self, limit):
        """
        Lists the keyspaces in your account.

        :param limit: The maximum number of keyspaces to list.
        """
        try:
            ks_paginator = self.keyspaces_client.get_paginator("list_keyspaces")
            for page in ks_paginator.paginate(PaginationConfig={"MaxItems":
limit}):
                for ks in page["keyspaces"]:
                    print(ks["keyspaceName"])
                    print(f"\t{ks['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list keyspaces. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
    def create_table(self, table_name):
        """
        Creates a table in the  keyspace.
        The table is created with a schema for storing movie data
        and has point-in-time recovery enabled.

        :param table_name: The name to give the table.
        :return: The ARN of the new table.
        """
        try:
            response = self.keyspaces_client.create_table(
                keyspaceName=self.ks_name,
                tableName=table_name,
                schemaDefinition={
                    "allColumns": [
                        {"name": "title", "type": "text"},
                        {"name": "year", "type": "int"},
                        {"name": "release_date", "type": "timestamp"},
                        {"name": "plot", "type": "text"},
                    ],
                    "partitionKeys": [{"name": "year"}, {"name": "title"}],
                },
                pointInTimeRecovery={"status": "ENABLED"},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["resourceArn"]


    def get_table(self, table_name):
        """
        Gets data about a table in the keyspace.

        :param table_name: The name of the table to look up.
        :return: Data about the table.
        """
        try:
```

```
            response = self.keyspaces_client.get_table(
                keyspaceName=self.ks_name, tableName=table_name
            )
            self.table_name = table_name
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Table %s does not exist.", table_name)
                self.table_name = None
                response = None
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    table_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return response


    def list_tables(self):
        """
        Lists the tables in the keyspace.
        """
        try:
            table_paginator = self.keyspaces_client.get_paginator("list_tables")
            for page in table_paginator.paginate(keyspaceName=self.ks_name):
                for table in page["tables"]:
                    print(table["tableName"])
                    print(f"\t{table['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list tables in keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def update_table(self):
        """
        Updates the schema of the table.
```

```
        This example updates a table of movie data by adding a new column
        that tracks whether the movie has been watched.
        """
        try:
            self.keyspaces_client.update_table(
                keyspaceName=self.ks_name,
                tableName=self.table_name,
                addColumns=[{"name": "watched", "type": "boolean"}],
            )
        except ClientError as err:
            logger.error(
                "Couldn't update table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def restore_table(self, restore_timestamp):
        """
        Restores the table to a previous point in time. The table is restored
        to a new table in the same keyspace.

        :param restore_timestamp: The point in time to restore the table. This
time
                                  must be in UTC format.
        :return: The name of the restored table.
        """
        try:
            restored_table_name = f"{self.table_name}_restored"
            self.keyspaces_client.restore_table(
                sourceKeyspaceName=self.ks_name,
                sourceTableName=self.table_name,
                targetKeyspaceName=self.ks_name,
                targetTableName=restored_table_name,
                restoreTimestamp=restore_timestamp,
            )
        except ClientError as err:
            logger.error(
                "Couldn't restore table %s. Here's why: %s: %s",
                restore_timestamp,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
```

```
            )
            raise
        else:
            return restored_table_name


    def delete_table(self):
        """
        Deletes the table from the keyspace.
        """
        try:
            self.keyspaces_client.delete_table(
                keyspaceName=self.ks_name, tableName=self.table_name
            )
            self.table_name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_keyspace(self):
        """
        Deletes the keyspace.
        """
        try:
            self.keyspaces_client.delete_keyspace(keyspaceName=self.ks_name)
            self.ks_name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

Definieren Sie eine Klasse, die eine TLS-Verbindung zu einem Keyspace herstellt, sich mit SigV4 authentifiziert und CQL-Abfragen an eine Tabelle im Keyspace sendet.

```
class QueryManager:
    """
    Manages queries to an Amazon Keyspaces (for Apache Cassandra) keyspace.
    Queries are secured by TLS and authenticated by using the Signature V4
 (SigV4)
    AWS signing protocol. This is more secure than sending username and password
    with a plain-text authentication provider.

    This example downloads a default certificate to secure TLS, or lets you
 specify
    your own.

    This example uses a table of movie data to demonstrate basic queries.
    """

    DEFAULT_CERT_FILE = "sf-class2-root.crt"
    CERT_URL = f"https://certs.secureserver.net/repository/sf-class2-root.crt"

    def __init__(self, cert_file_path, boto_session, keyspace_name):
        """
        :param cert_file_path: The path and file name of the certificate used for
 TLS.
        :param boto_session: A Boto3 session. This is used to acquire your AWS
 credentials.
        :param keyspace_name: The name of the keyspace to connect.
        """
        self.cert_file_path = cert_file_path
        self.boto_session = boto_session
        self.ks_name = keyspace_name
        self.cluster = None
        self.session = None

    def __enter__(self):
        """
        Creates a session connection to the keyspace that is secured by TLS and
        authenticated by SigV4.
        """
        ssl_context = SSLContext(PROTOCOL_TLSv1_2)
```

```
        ssl_context.load_verify_locations(self.cert_file_path)
        ssl_context.verify_mode = CERT_REQUIRED
        auth_provider = SigV4AuthProvider(self.boto_session)
        contact_point = f"cassandra.
{self.boto_session.region_name}.amazonaws.com"
        exec_profile = ExecutionProfile(
            consistency_level=ConsistencyLevel.LOCAL_QUORUM,
            load_balancing_policy=DCAwareRoundRobinPolicy(),
        )
        self.cluster = Cluster(
            [contact_point],
            ssl_context=ssl_context,
            auth_provider=auth_provider,
            port=9142,
            execution_profiles={EXEC_PROFILE_DEFAULT: exec_profile},
            protocol_version=4,
        )
        self.cluster.__enter__()
        self.session = self.cluster.connect(self.ks_name)
        return self

    def __exit__(self, *args):
        """
        Exits the cluster. This shuts down all existing session connections.
        """
        self.cluster.__exit__(*args)

    def add_movies(self, table_name, movie_file_path):
        """
        Gets movies from a JSON file and adds them to a table in the keyspace.

        :param table_name: The name of the table.
        :param movie_file_path: The path and file name of a JSON file that
 contains movie data.
        """
        with open(movie_file_path, "r") as movie_file:
            movies = json.loads(movie_file.read())
        stmt = self.session.prepare(
            f"INSERT INTO {table_name} (year, title, release_date, plot) VALUES
 (?, ?, ?, ?);"
        )
        for movie in movies[:20]:
            self.session.execute(
                stmt,
```

```
                parameters=[
                    movie["year"],
                    movie["title"],
                    date.fromisoformat(movie["info"]
["release_date"].partition("T")[0]),
                    movie["info"]["plot"],
                ],
            )

    def get_movies(self, table_name, watched=None):
        """
        Gets the title and year of the full list of movies from the table.

        :param table_name: The name of the movie table.
        :param watched: When specified, the returned list of movies is filtered
 to
                        either movies that have been watched or movies that have
 not
                        been watched. Otherwise, all movies are returned.
        :return: A list of movies in the table.
        """
        if watched is None:
            stmt = SimpleStatement(f"SELECT title, year from {table_name}")
            params = None
        else:
            stmt = SimpleStatement(
                f"SELECT title, year from {table_name} WHERE watched = %s ALLOW
 FILTERING"
            )
            params = [watched]
        return self.session.execute(stmt, parameters=params).all()

    def get_movie(self, table_name, title, year):
        """
        Gets a single movie from the table, by title and year.

        :param table_name: The name of the movie table.
        :param title: The title of the movie.
        :param year: The year of the movie's release.
        :return: The requested movie.
        """
        return self.session.execute(
            SimpleStatement(
                f"SELECT * from {table_name} WHERE title = %s AND year = %s"
```

```
            ),
            parameters=[title, year],
        ).one()

    def watched_movie(self, table_name, title, year):
        """
        Updates a movie as having been watched.

        :param table_name: The name of the movie table.
        :param title: The title of the movie.
        :param year: The year of the movie's release.
        """
        self.session.execute(
            SimpleStatement(
                f"UPDATE {table_name} SET watched=true WHERE title = %s AND year
= %s"
            ),
            parameters=[title, year],
        )
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für Python (Boto3).
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Aktionen für Amazon Keyspaces mithilfe von AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne Amazon Keyspaces-Aktionen mit AWS SDKs durchgeführt werden. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der API-Referenz für Amazon Keyspaces (für Apache Cassandra).

Beispiele

- Verwenden Sie es CreateKeyspace mit einem AWS SDK
- CreateTableMit einem AWS SDK verwenden
- DeleteKeyspaceMit einem AWS SDK verwenden
- DeleteTableMit einem AWS SDK verwenden
- GetKeyspaceMit einem AWS SDK verwenden
- GetTableMit einem AWS SDK verwenden
- ListKeyspacesMit einem AWS SDK verwenden
- ListTablesMit einem AWS SDK verwenden
- RestoreTableMit einem AWS SDK verwenden
- UpdateTableMit einem AWS SDK verwenden

Verwenden Sie es **`CreateKeyspace`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `CreateKeyspace` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Create a new keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name for the new keyspace.</param>
    /// <returns>The Amazon Resource Name (ARN) of the new keyspace.</returns>
    public async Task<string> CreateKeyspace(string keyspaceName)
    {
        var response =
            await _amazonKeyspaces.CreateKeyspaceAsync(
                new CreateKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.ResourceArn;
    }
```

- Einzelheiten zur API finden Sie CreateKeyspacein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    public static void createKeySpace(KeyspacesClient keyClient, String
keyspaceName) {
        try {
```

```
            CreateKeyspaceRequest keyspaceRequest =
 CreateKeyspaceRequest.builder()
                    .keyspaceName(keyspaceName)
                    .build();

            CreateKeyspaceResponse response =
 keyClient.createKeyspace(keyspaceRequest);
            System.out.println("The ARN of the KeySpace is " +
 response.resourceArn());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie CreateKeyspacein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest =
        CreateKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Einzelheiten zur API finden Sie CreateKeyspacein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def create_keyspace(self, name):
        """
        Creates a keyspace.

        :param name: The name to give the keyspace.
        :return: The Amazon Resource Name (ARN) of the new keyspace.
        """
        try:
            response = self.keyspaces_client.create_keyspace(keyspaceName=name)
```

```
            self.ks_name = name
            self.ks_arn = response["resourceArn"]
        except ClientError as err:
            logger.error(
                "Couldn't create %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.ks_arn
```

- Einzelheiten zur API finden Sie CreateKeyspacein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        oo_result = lo_kys->createkeyspace(
          iv_keyspacename = iv_keyspace_name ).
        MESSAGE 'Keyspace created successfully.' TYPE 'I'.
      CATCH /aws1/cx_kysconflictexception.
        MESSAGE 'Keyspace already exists.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie CreateKeyspacein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`CreateTable`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `CreateTable` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /// <summary>
   /// Create a new Amazon Keyspaces table.
   /// </summary>
   /// <param name="keyspaceName">The keyspace where the table will be
created.</param>
   /// <param name="schema">The schema for the new table.</param>
   /// <param name="tableName">The name of the new table.</param>
   /// <returns>The Amazon Resource Name (ARN) of the new table.</returns>
   public async Task<string> CreateTable(string keyspaceName, SchemaDefinition
schema, string tableName)
   {
       var request = new CreateTableRequest
       {
           KeyspaceName = keyspaceName,
```

```
            SchemaDefinition = schema,
            TableName = tableName,
            PointInTimeRecovery = new PointInTimeRecovery { Status =
PointInTimeRecoveryStatus.ENABLED }
        };

        var response = await _amazonKeyspaces.CreateTableAsync(request);
        return response.ResourceArn;
    }
```

- Einzelheiten zur API finden Sie CreateTablein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    public static void createTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
        try {
            // Set the columns.
            ColumnDefinition defTitle = ColumnDefinition.builder()
                    .name("title")
                    .type("text")
                    .build();

            ColumnDefinition defYear = ColumnDefinition.builder()
                    .name("year")
                    .type("int")
                    .build();

            ColumnDefinition defReleaseDate = ColumnDefinition.builder()
                    .name("release_date")
                    .type("timestamp")
                    .build();
```

```
        ColumnDefinition defPlot = ColumnDefinition.builder()
                .name("plot")
                .type("text")
                .build();

        List<ColumnDefinition> colList = new ArrayList<>();
        colList.add(defTitle);
        colList.add(defYear);
        colList.add(defReleaseDate);
        colList.add(defPlot);

        // Set the keys.
        PartitionKey yearKey = PartitionKey.builder()
                .name("year")
                .build();

        PartitionKey titleKey = PartitionKey.builder()
                .name("title")
                .build();

        List<PartitionKey> keyList = new ArrayList<>();
        keyList.add(yearKey);
        keyList.add(titleKey);

        SchemaDefinition schemaDefinition = SchemaDefinition.builder()
                .partitionKeys(keyList)
                .allColumns(colList)
                .build();

        PointInTimeRecovery timeRecovery = PointInTimeRecovery.builder()
                .status(PointInTimeRecoveryStatus.ENABLED)
                .build();

        CreateTableRequest tableRequest = CreateTableRequest.builder()
                .keyspaceName(keySpace)
                .tableName(tableName)
                .schemaDefinition(schemaDefinition)
                .pointInTimeRecovery(timeRecovery)
                .build();

        CreateTableResponse response = keyClient.createTable(tableRequest);
        System.out.println("The table ARN is " + response.resourceArn());
```

```
        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie CreateTablein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun createTable(
    keySpaceVal: String?,
    tableNameVal: String?,
) {
    // Set the columns.
    val defTitle =
        ColumnDefinition {
            name = "title"
            type = "text"
        }

    val defYear =
        ColumnDefinition {
            name = "year"
            type = "int"
        }

    val defReleaseDate =
        ColumnDefinition {
            name = "release_date"
            type = "timestamp"
        }
```

```
    val defPlot =
        ColumnDefinition {
            name = "plot"
            type = "text"
        }

    val colList = ArrayList<ColumnDefinition>()
    colList.add(defTitle)
    colList.add(defYear)
    colList.add(defReleaseDate)
    colList.add(defPlot)

    // Set the keys.
    val yearKey =
        PartitionKey {
            name = "year"
        }

    val titleKey =
        PartitionKey {
            name = "title"
        }

    val keyList = ArrayList<PartitionKey>()
    keyList.add(yearKey)
    keyList.add(titleKey)

    val schemaDefinitionOb =
        SchemaDefinition {
            partitionKeys = keyList
            allColumns = colList
        }

    val timeRecovery =
        PointInTimeRecovery {
            status = PointInTimeRecoveryStatus.Enabled
        }

    val tableRequest =
        CreateTableRequest {
            keyspaceName = keySpaceVal
            tableName = tableNameVal
            schemaDefinition = schemaDefinitionOb
            pointInTimeRecovery = timeRecovery
```

```
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createTable(tableRequest)
        println("The table ARN is ${response.resourceArn}")
    }
}
```

- Einzelheiten zur API finden Sie CreateTablein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def create_table(self, table_name):
        """
```

```
        Creates a table in the  keyspace.
        The table is created with a schema for storing movie data
        and has point-in-time recovery enabled.

        :param table_name: The name to give the table.
        :return: The ARN of the new table.
        """
        try:
            response = self.keyspaces_client.create_table(
                keyspaceName=self.ks_name,
                tableName=table_name,
                schemaDefinition={
                    "allColumns": [
                        {"name": "title", "type": "text"},
                        {"name": "year", "type": "int"},
                        {"name": "release_date", "type": "timestamp"},
                        {"name": "plot", "type": "text"},
                    ],
                    "partitionKeys": [{"name": "year"}, {"name": "title"}],
                },
                pointInTimeRecovery={"status": "ENABLED"},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["resourceArn"]
```

- Einzelheiten zur API finden Sie CreateTablein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        " Define schema with columns
        DATA(lt_columns) = VALUE /aws1/
cl_kyscolumndefinition=>tt_columndefinitionlist(
          ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'title' iv_type =
 'text' ) )
          ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'year' iv_type =
 'int' ) )
          ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'release_date' iv_type =
 'timestamp' ) )
          ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'plot' iv_type =
 'text' ) )
        ).

        " Define partition keys
        DATA(lt_partition_keys) = VALUE /aws1/
cl_kyspartitionkey=>tt_partitionkeylist(
          ( NEW /aws1/cl_kyspartitionkey( iv_name = 'year' ) )
          ( NEW /aws1/cl_kyspartitionkey( iv_name = 'title' ) )
        ).

        " Create schema definition
        DATA(lo_schema) = NEW /aws1/cl_kysschemadefinition(
          it_allcolumns = lt_columns
          it_partitionkeys = lt_partition_keys ).

        " Enable point-in-time recovery
        DATA(lo_pitr) = NEW /aws1/cl_kyspointintimerec(
          iv_status = 'ENABLED' ).

        oo_result = lo_kys->createtable(
          iv_keyspacename = iv_keyspace_name
```

```
          iv_tablename = iv_table_name
          io_schemadefinition = lo_schema
          io_pointintimerecovery = lo_pitr ).
        MESSAGE 'Table created successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie CreateTablein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`DeleteKeyspace`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `DeleteKeyspace` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Delete an existing keyspace.
    /// </summary>
```

```
    /// <param name="keyspaceName"></param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyspace(string keyspaceName)
    {
        var response = await _amazonKeyspaces.DeleteKeyspaceAsync(
            new DeleteKeyspaceRequest { KeyspaceName = keyspaceName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteKeyspacein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   public static void deleteKeyspace(KeyspacesClient keyClient, String
keyspaceName) {
       try {
           DeleteKeyspaceRequest deleteKeyspaceRequest =
DeleteKeyspaceRequest.builder()
                   .keyspaceName(keyspaceName)
                   .build();

           keyClient.deleteKeyspace(deleteKeyspaceRequest);

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie DeleteKeyspacein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest =
        DeleteKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}
```

- Einzelheiten zur API finden Sie DeleteKeyspacein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""
```

```
    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def delete_keyspace(self):
        """
        Deletes the keyspace.
        """
        try:
            self.keyspaces_client.delete_keyspace(keyspaceName=self.ks_name)
            self.ks_name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie DeleteKeyspacein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        lo_kys->deletekeyspace(
          iv_keyspacename = iv_keyspace_name ).
        MESSAGE 'Keyspace deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie DeleteKeyspacein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`DeleteTable`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `DeleteTable` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Delete an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTable(string keyspaceName, string tableName)
    {
        var response = await _amazonKeyspaces.DeleteTableAsync(
            new DeleteTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteTablein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    public static void deleteTable(KeyspacesClient keyClient, String
keyspaceName, String tableName) {
```

```
        try {
            DeleteTableRequest tableRequest = DeleteTableRequest.builder()
                    .keyspaceName(keyspaceName)
                    .tableName(tableName)
                    .build();

            keyClient.deleteTable(tableRequest);

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie DeleteTablein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}
```

- Einzelheiten zur API finden Sie DeleteTablein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def delete_table(self):
        """
        Deletes the table from the keyspace.
        """
        try:
            self.keyspaces_client.delete_table(
                keyspaceName=self.ks_name, tableName=self.table_name
            )
            self.table_name = None
```

```
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie DeleteTablein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        lo_kys->deletetable(
          iv_keyspacename = iv_keyspace_name
          iv_tablename = iv_table_name ).
        MESSAGE 'Table deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie DeleteTablein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`GetKeyspace`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `GetKeyspace` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Get data about a keyspace.
/// </summary>
/// <param name="keyspaceName">The name of the keyspace.</param>
/// <returns>The Amazon Resource Name (ARN) of the keyspace.</returns>
public async Task<string> GetKeyspace(string keyspaceName)
{
    var response = await _amazonKeyspaces.GetKeyspaceAsync(
        new GetKeyspaceRequest { KeyspaceName = keyspaceName });
    return response.ResourceArn;
}
```

- Einzelheiten zur API finden Sie GetKeyspacein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   public static void checkKeyspaceExistence(KeyspacesClient keyClient, String
keyspaceName) {
       try {
           GetKeyspaceRequest keyspaceRequest = GetKeyspaceRequest.builder()
                   .keyspaceName(keyspaceName)
                   .build();

           GetKeyspaceResponse response =
keyClient.getKeyspace(keyspaceRequest);
           String name = response.keyspaceName();
           System.out.println("The " + name + " KeySpace is ready");

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie GetKeyspacein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyspaceRequest =
        GetKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response: GetKeyspaceResponse =
 keyClient.getKeyspace(keyspaceRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}
```

- Einzelheiten zur API finden Sie GetKeyspacein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
```

```
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def exists_keyspace(self, name):
        """
        Checks whether a keyspace exists.

        :param name: The name of the keyspace to look up.
        :return: True when the keyspace exists. Otherwise, False.
        """
        try:
            response = self.keyspaces_client.get_keyspace(keyspaceName=name)
            self.ks_name = response["keyspaceName"]
            self.ks_arn = response["resourceArn"]
            exists = True
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Keyspace %s does not exist.", name)
                exists = False
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return exists
```

- Einzelheiten zur API finden Sie GetKeyspacein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        oo_result = lo_kys->getkeyspace(
          iv_keyspacename = iv_keyspace_name ).
        MESSAGE 'Keyspace retrieved successfully.' TYPE 'I'.
      CATCH /aws1/cx_kysresourcenotfoundex.
        MESSAGE 'Keyspace does not exist.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie GetKeyspacein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`GetTable`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `GetTable` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Get information about an Amazon Keyspaces table.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the Amazon Keyspaces table.</param>
    /// <returns>The response containing data about the table.</returns>
    public async Task<GetTableResponse> GetTable(string keyspaceName, string
tableName)
    {
        var response = await _amazonKeyspaces.GetTableAsync(
            new GetTableRequest { KeyspaceName = keyspaceName, TableName =
tableName });
        return response;
    }
```

- Einzelheiten zur API finden Sie GetTablein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   public static void checkTable(KeyspacesClient keyClient, String keyspaceName,
String tableName)
           throws InterruptedException {
       try {
           boolean tableStatus = false;
           String status;
           GetTableResponse response = null;
           GetTableRequest tableRequest = GetTableRequest.builder()
                   .keyspaceName(keyspaceName)
                   .tableName(tableName)
                   .build();

           while (!tableStatus) {
               response = keyClient.getTable(tableRequest);
               status = response.statusAsString();
               System.out.println(". The table status is " + status);

               if (status.compareTo("ACTIVE") == 0) {
                   tableStatus = true;
               }
               Thread.sleep(500);
           }

           List<ColumnDefinition> cols =
response.schemaDefinition().allColumns();
           for (ColumnDefinition def : cols) {
               System.out.println("The column name is " + def.name());
               System.out.println("The column type is " + def.type());
           }

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie GetTablein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun checkTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println(". The table status is $status")
            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
        val cols: List<ColumnDefinition>? =
 response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
```

```
}
```

- Einzelheiten zur API finden Sie GetTablein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def get_table(self, table_name):
        """
        Gets data about a table in the keyspace.

        :param table_name: The name of the table to look up.
        :return: Data about the table.
        """
        try:
```

```
            response = self.keyspaces_client.get_table(
                keyspaceName=self.ks_name, tableName=table_name
            )
            self.table_name = table_name
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info("Table %s does not exist.", table_name)
                self.table_name = None
                response = None
            else:
                logger.error(
                    "Couldn't verify %s exists. Here's why: %s: %s",
                    table_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return response
```

- Einzelheiten zur API finden Sie GetTablein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        oo_result = lo_kys->gettable(
          iv_keyspacename = iv_keyspace_name
          iv_tablename = iv_table_name ).
        MESSAGE 'Table information retrieved successfully.' TYPE 'I'.
      CATCH /aws1/cx_kysresourcenotfoundex.
        MESSAGE 'Table does not exist.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
```

```
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie GetTablein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`ListKeyspaces`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `ListKeyspaces` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Lists all keyspaces for the account.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task ListKeyspaces()
    {
        var paginator = _amazonKeyspaces.Paginators.ListKeyspaces(new
ListKeyspacesRequest());
```

```
        Console.WriteLine("{0, -30}\t{1}", "Keyspace name", "Keyspace ARN");
        Console.WriteLine(new string('-', Console.WindowWidth));
        await foreach (var keyspace in paginator.Keyspaces)
        {

Console.WriteLine($"{keyspace.KeyspaceName,-30}\t{keyspace.ResourceArn}");
        }
    }
```

- Einzelheiten zur API finden Sie ListKeyspacesin der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    public static void listKeyspacesPaginator(KeyspacesClient keyClient) {
        try {
            ListKeyspacesRequest keyspacesRequest =
ListKeyspacesRequest.builder()
                    .maxResults(10)
                    .build();

            ListKeyspacesIterable listRes =
keyClient.listKeyspacesPaginator(keyspacesRequest);
            listRes.stream()
                    .flatMap(r -> r.keyspaces().stream())
                    .forEach(content -> System.out.println(" Name: " +
content.keyspaceName()));

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie ListKeyspacesin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun listKeyspacesPaginator() {
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listKeyspacesPaginated(ListKeyspacesRequest {})
            .transform { it.keyspaces?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println("Name: ${obj.keyspaceName}")
            }
    }
}
```

- Einzelheiten zur API finden Sie ListKeyspacesin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
```

```
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def list_keyspaces(self, limit):
        """
        Lists the keyspaces in your account.

        :param limit: The maximum number of keyspaces to list.
        """
        try:
            ks_paginator = self.keyspaces_client.get_paginator("list_keyspaces")
            for page in ks_paginator.paginate(PaginationConfig={"MaxItems":
limit}):
                for ks in page["keyspaces"]:
                    print(ks["keyspaceName"])
                    print(f"\t{ks['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list keyspaces. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie ListKeyspacesin AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        oo_result = lo_kys->listkeyspaces(
          iv_maxresults = iv_max_results ).
        MESSAGE 'Keyspaces listed successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie ListKeyspacesin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`ListTables`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `ListTables` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Lists the Amazon Keyspaces tables in a keyspace.
    /// </summary>
    /// <param name="keyspaceName">The name of the keyspace.</param>
    /// <returns>A list of TableSummary objects.</returns>
    public async Task<List<TableSummary>> ListTables(string keyspaceName)
    {
        var response = await _amazonKeyspaces.ListTablesAsync(new
ListTablesRequest { KeyspaceName = keyspaceName });
        response.Tables.ForEach(table =>
        {

Console.WriteLine($"{table.KeyspaceName}\t{table.TableName}\t{table.ResourceArn}");
        });

        return response.Tables;
    }
```

- Einzelheiten zur API finden Sie ListTablesin der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   public static void listTables(KeyspacesClient keyClient, String keyspaceName)
{
       try {
           ListTablesRequest tablesRequest = ListTablesRequest.builder()
                   .keyspaceName(keyspaceName)
                   .build();

           ListTablesIterable listRes =
keyClient.listTablesPaginator(tablesRequest);
           listRes.stream()
                   .flatMap(r -> r.tables().stream())
                   .forEach(content -> System.out.println(" ARN: " +
content.resourceArn() +
                           " Table name: " + content.tableName()));

       } catch (KeyspacesException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie ListTablesin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest =
        ListTablesRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
```

```
        keyClient
            .listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println(" ARN: ${obj.resourceArn} Table name: ${obj.tableName}")
            }
    }
}
```

- Einzelheiten zur API finden Sie ListTablesin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def list_tables(self):
```

```
        """
        Lists the tables in the keyspace.
        """
        try:
            table_paginator = self.keyspaces_client.get_paginator("list_tables")
            for page in table_paginator.paginate(keyspaceName=self.ks_name):
                for table in page["tables"]:
                    print(table["tableName"])
                    print(f"\t{table['resourceArn']}")
        except ClientError as err:
            logger.error(
                "Couldn't list tables in keyspace %s. Here's why: %s: %s",
                self.ks_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie ListTablesin AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        oo_result = lo_kys->listtables(
          iv_keyspacename = iv_keyspace_name ).
        MESSAGE 'Tables listed successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie ListTablesin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`RestoreTable`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `RestoreTable` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Restores the specified table to the specified point in time.
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to restore.</param>
    /// <param name="timestamp">The time to which the table will be restored.</
param>
    /// <returns>The Amazon Resource Name (ARN) of the restored table.</returns>
    public async Task<string> RestoreTable(string keyspaceName, string tableName,
 string restoredTableName, DateTime timestamp)
    {
```

```
        var request = new RestoreTableRequest
        {
            RestoreTimestamp = timestamp,
            SourceKeyspaceName = keyspaceName,
            SourceTableName = tableName,
            TargetKeyspaceName = keyspaceName,
            TargetTableName = restoredTableName
        };

        var response = await _amazonKeyspaces.RestoreTableAsync(request);
        return response.RestoredTableARN;
    }
```

- Einzelheiten zur API finden Sie RestoreTablein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   public static void restoreTable(KeyspacesClient keyClient, String
keyspaceName, ZonedDateTime utc) {
        try {
            Instant myTime = utc.toInstant();
            RestoreTableRequest restoreTableRequest =
RestoreTableRequest.builder()
                    .restoreTimestamp(myTime)
                    .sourceTableName("Movie")
                    .targetKeyspaceName(keyspaceName)
                    .targetTableName("MovieRestore")
                    .sourceKeyspaceName(keyspaceName)
                    .build();

            RestoreTableResponse response =
keyClient.restoreTable(restoreTableRequest);
```

```
            System.out.println("The ARN of the restored table is " +
 response.restoredTableARN());

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie RestoreTablein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun restoreTable(
    keyspaceName: String?,
    utc: ZonedDateTime,
) {
    // Create an aws.smithy.kotlin.runtime.time.Instant value.
    val timeStamp =
        aws.smithy.kotlin.runtime.time
            .Instant(utc.toInstant())
    val restoreTableRequest =
        RestoreTableRequest {
            restoreTimestamp = timeStamp
            sourceTableName = "MovieKotlin"
            targetKeyspaceName = keyspaceName
            targetTableName = "MovieRestore"
            sourceKeyspaceName = keyspaceName
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.restoreTable(restoreTableRequest)
        println("The ARN of the restored table is ${response.restoredTableArn}")
```

```
    }
}
```

- Einzelheiten zur API finden Sie RestoreTablein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def restore_table(self, restore_timestamp):
        """
        Restores the table to a previous point in time. The table is restored
        to a new table in the same keyspace.

        :param restore_timestamp: The point in time to restore the table. This
 time
```

```
                                must be in UTC format.
        :return: The name of the restored table.
        """
        try:
            restored_table_name = f"{self.table_name}_restored"
            self.keyspaces_client.restore_table(
                sourceKeyspaceName=self.ks_name,
                sourceTableName=self.table_name,
                targetKeyspaceName=self.ks_name,
                targetTableName=restored_table_name,
                restoreTimestamp=restore_timestamp,
            )
        except ClientError as err:
            logger.error(
                "Couldn't restore table %s. Here's why: %s: %s",
                restore_timestamp,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return restored_table_name
```

- Einzelheiten zur API finden Sie RestoreTablein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        oo_result = lo_kys->restoretable(
          iv_sourcekeyspacename = iv_source_keyspace_name
```

```
          iv_sourcetablename = iv_source_table_name
          iv_targetkeyspacename = iv_target_keyspace_name
          iv_targettablename = iv_target_table_name
          iv_restoretimestamp = iv_restore_timestamp ).
        MESSAGE 'Table restore initiated successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie RestoreTablein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`UpdateTable`Mit einem AWS SDK verwenden

Die folgenden Code-Beispiele zeigen, wie `UpdateTable` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Kennenlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Updates the movie table to add a boolean column named watched.
```

```
    /// </summary>
    /// <param name="keyspaceName">The keyspace containing the table.</param>
    /// <param name="tableName">The name of the table to change.</param>
    /// <returns>The Amazon Resource Name (ARN) of the updated table.</returns>
    public async Task<string> UpdateTable(string keyspaceName, string tableName)
    {
        var newColumn = new ColumnDefinition { Name = "watched", Type =
"boolean" };
        var request = new UpdateTableRequest
        {
            KeyspaceName = keyspaceName,
            TableName = tableName,
            AddColumns = new List<ColumnDefinition> { newColumn }
        };
        var response = await _amazonKeyspaces.UpdateTableAsync(request);
        return response.ResourceArn;
    }
```

- Einzelheiten zur API finden Sie UpdateTablein der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    public static void updateTable(KeyspacesClient keyClient, String keySpace,
String tableName) {
        try {
            ColumnDefinition def = ColumnDefinition.builder()
                    .name("watched")
                    .type("boolean")
                    .build();

            UpdateTableRequest tableRequest = UpdateTableRequest.builder()
                    .keyspaceName(keySpace)
```

```
                    .tableName(tableName)
                    .addColumns(def)
                    .build();

            keyClient.updateTable(tableRequest);

        } catch (KeyspacesException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie UpdateTablein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun updateTable(
    keySpace: String?,
    tableNameVal: String?,
) {
    val def =
        ColumnDefinition {
            name = "watched"
            type = "boolean"
        }

    val tableRequest =
        UpdateTableRequest {
            keyspaceName = keySpace
            tableName = tableNameVal
            addColumns = listOf(def)
        }
```

```
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.updateTable(tableRequest)
    }
}
```

- Einzelheiten zur API finden Sie UpdateTablein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class KeyspaceWrapper:
    """Encapsulates Amazon Keyspaces (for Apache Cassandra) keyspace and table
 actions."""

    def __init__(self, keyspaces_client):
        """
        :param keyspaces_client: A Boto3 Amazon Keyspaces client.
        """
        self.keyspaces_client = keyspaces_client
        self.ks_name = None
        self.ks_arn = None
        self.table_name = None

    @classmethod
    def from_client(cls):
        keyspaces_client = boto3.client("keyspaces")
        return cls(keyspaces_client)


    def update_table(self):
        """
        Updates the schema of the table.

        This example updates a table of movie data by adding a new column
```

```
        that tracks whether the movie has been watched.
        """
        try:
            self.keyspaces_client.update_table(
                keyspaceName=self.ks_name,
                tableName=self.table_name,
                addColumns=[{"name": "watched", "type": "boolean"}],
            )
        except ClientError as err:
            logger.error(
                "Couldn't update table %s. Here's why: %s: %s",
                self.table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie UpdateTablein AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        " Add a new column to track watched movies
        DATA(lt_add_columns) = VALUE /aws1/
cl_kyscolumndefinition=>tt_columndefinitionlist(
          ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'watched' iv_type =
 'boolean' ) )
        ).

        oo_result = lo_kys->updatetable(
          iv_keyspacename = iv_keyspace_name
          iv_tablename = iv_table_name
```

```
          it_addcolumns = lt_add_columns ).
        MESSAGE 'Table updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie UpdateTablein der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Bibliotheken und Tools von Amazon Keyspaces (für Apache Cassandra)

Dieser Abschnitt enthält Informationen zu Bibliotheken, Codebeispielen und Tools von Amazon Keyspaces (für Apache Cassandra).

Topics

- Bibliotheken und Beispiele
- Hervorgehobene Beispiel- und Entwickler-Tool-Repos

Bibliotheken und Beispiele

Die Open-Source-Bibliotheken und Entwicklertools von Amazon Keyspaces finden Sie GitHub in den Repos AWSund den AWS Beispiel-Repos.

Amazon Keyspaces (für Apache Cassandra) Entwickler-Toolkit

Dieses Repository bietet ein Docker-Image mit hilfreichen Entwicklertools für Amazon Keyspaces. Es enthält beispielsweise eine CQLSHRC-Datei mit bewährten Methoden, eine optionale AWS Authentifizierungserweiterung für cqlsh und Hilfstools zur Ausführung allgemeiner Aufgaben. Das Toolkit ist für Amazon Keyspaces optimiert, funktioniert aber auch mit Apache Cassandra-Clustern.

https://github.com/aws-samples/amazon-keyspaces-toolkit.

Beispiele für Amazon Keyspaces (für Apache Cassandra)

Dieses Repo ist unsere offizielle Liste von Amazon Keyspaces-Beispielcodes. Das Repo ist nach Sprachen in Abschnitte unterteilt (siehe Beispiele). Jede Sprache hat ihren eigenen Unterabschnitt mit Beispielen. Diese Beispiele zeigen gängige Amazon Keyspaces-Serviceimplementierungen und Muster, die Sie beim Erstellen von Anwendungen verwenden können.

https://github.com/aws-samples/amazon-keyspaces-examples/.

AWS Authentifizierungs-Plugins für Signature Version 4 (Sigv4)

Mit den Plug-ins können Sie den Zugriff auf Amazon Keyspaces mithilfe von AWS Identity and Access Management (IAM-) Benutzern und -Rollen verwalten.

Java:. https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin

Node.js: https://github.com/aws/aws-sigv4-auth-cassandra-nodejs-driver-plugin.

Python: https://github.com/aws/aws-sigv4-auth-cassandra-python-driver-plugin.

Geh:https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Hervorgehobene Beispiel- und Entwickler-Tool-Repos

Im Folgenden finden Sie eine Auswahl hilfreicher Community-Tools für Amazon Keyspaces (für Apache Cassandra).

Amazon Keyspaces-Protokollpuffer

Sie können Protocol Buffers (Protobuf) mit Amazon Keyspaces verwenden, um eine Alternative zu Apache Cassandra User Defined Types (UDTs) bereitzustellen. Protobuf ist ein kostenloses, plattformübergreifendes Open-Source-Datenformat, das zur Serialisierung strukturierter Daten verwendet wird. Sie können Protobuf-Daten mithilfe des CQL-Datentyps speichern und UDTs umgestalten und gleichzeitig strukturierte `BLOB` Daten in allen Anwendungen und Programmiersprachen beibehalten.

Dieses Repository bietet ein Codebeispiel, das eine Verbindung zu Amazon Keyspaces herstellt, eine neue Tabelle erstellt und eine Zeile mit einer Protobuf-Nachricht einfügt. Dann wird die Zeile mit starker Konsistenz gelesen.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/protobuf-user-defined-types

CloudFormation Vorlage zur Erstellung eines CloudWatch Amazon-Dashboards für Amazon Keyspaces-Metriken (für Apache Cassandra)

Dieses Repository bietet CloudFormation Vorlagen für die schnelle Einrichtung von CloudWatch Metriken für Amazon Keyspaces. Die Verwendung dieser Vorlage erleichtert Ihnen den Einstieg, indem Sie bereitstellbare vorgefertigte CloudWatch Dashboards mit häufig verwendeten Metriken bereitstellen.

https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates.

Verwenden von Amazon Keyspaces (für Apache Cassandra) mit AWS Lambda

Das Repository enthält Beispiele, die zeigen, wie Sie von Lambda aus eine Verbindung zu Amazon Keyspaces herstellen können. Im Folgenden finden Sie einige Beispiele.

C#/.NET:. https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/dotnet/datastax-v3/connection-lambda

Java:. https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/connection-lambda

Ein weiteres Lambda-Beispiel, das zeigt, wie Amazon Keyspaces von einem Python-Lambda aus bereitgestellt und verwendet werden, ist im folgenden Repo verfügbar.

https://github.com/aws-samples/aws-keyspaces-lambda-python

Verwenden von Amazon Keyspaces (für Apache Cassandra) mit Spring

Dies ist ein Beispiel, das Ihnen zeigt, wie Sie Amazon Keyspaces mit Spring Boot verwenden.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring

Verwenden von Amazon Keyspaces (für Apache Cassandra) mit Scala

Dies ist ein Beispiel, das zeigt, wie Sie mithilfe des SigV4-Authentifizierungs-Plug-ins mit Scala eine Verbindung zu Amazon Keyspaces herstellen.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/connection-sigv4

Verwenden von Amazon Keyspaces (für Apache Cassandra) mit AWS Glue

Dies ist ein Beispiel, das zeigt, wie Amazon Keyspaces mit AWS Glue verwendet wird.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue

Amazon Keyspaces (für Apache Cassandra) Cassandra Query Language (CQL) zum Konverter CloudFormation

Dieses Paket implementiert ein Befehlszeilentool zur Konvertierung von Apache Cassandra Query Language (CQL) -Skripts in CloudFormation (CloudFormation) -Vorlagen, wodurch Amazon Keyspaces-Schemas einfach stapelweise verwaltet werden können. CloudFormation

https://github.com/aws/amazon-keyspaces-cql-to-cfn-converter.

Amazon Keyspaces (für Apache Cassandra) Helfer für den Apache Cassandra-Treiber für Java

Dieses Repository enthält Treiberrichtlinien, Beispiele und bewährte Methoden für die Verwendung des DataStax Java-Treibers mit Amazon Keyspaces (für Apache Cassandra).

https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers.

Schnelle Komprimierungsdemo von Amazon Keyspaces (für Apache Cassandra)

Dieses Repository zeigt, wie Sie Objekte komprimieren, speichern und read/write groß machen können, um eine schnellere Leistung zu erzielen und den Durchsatz und die Speicherkosten zu senken.

https://github.com/aws-samples/amazon-keyspaces-compression-example.

Amazon Keyspaces (für Apache Cassandra) und Amazon S3 S3-Codec-Demo

Der benutzerdefinierte Amazon S3 S3-Codec unterstützt die transparente, vom Benutzer konfigurierbare Zuordnung von UUID-Zeigern zu Amazon S3 S3-Objekten.

https://github.com/aws-samples/amazon-keyspaces-large-object-s3-demo.

Bewährte Methoden für Design und Architektur mit Amazon Keyspaces (für Apache Cassandra)

In diesem Abschnitt finden Sie schnell Empfehlungen zur Maximierung der Leistung und Minimierung der Durchsatzkosten bei der Arbeit mit Amazon Keyspaces.

Inhalt

- Hauptunterschiede und Gestaltungsprinzipien des NoSQL-Designs
  - Unterschiede zwischen einem relationalen Datendesign und NoSQL
  - Zwei Schlüsselkonzepte für das NoSQL-Design
  - Ansatz für ein NoSQL-Design
- Optimieren Sie die Client-Treiberverbindungen für die serverlose Umgebung
  - So funktionieren Verbindungen in Amazon Keyspaces
  - So konfigurieren Sie Verbindungen in Amazon Keyspaces
  - So konfigurieren Sie die Wiederholungsrichtlinie für Verbindungen in Amazon Keyspaces
  - So konfigurieren Sie Verbindungen über VPC-Endpunkte in Amazon Keyspaces
  - So überwachen Sie Verbindungen in Amazon Keyspaces
  - So behandeln Sie Verbindungsfehler in Amazon Keyspaces
- Bewährte Methoden zur Datenmodellierung: Empfehlungen für den Entwurf von Datenmodellen
  - So verwenden Sie Partitionsschlüssel effektiv in Amazon Keyspaces
    - Verwenden Sie Write-Sharding, um Workloads gleichmäßig auf die Partitionen zu verteilen
      - Sharding mit zusammengesetzten Partitionsschlüsseln und Zufallswerten
      - Sharding mit zusammengesetzten Partitionsschlüsseln und berechneten Werten
- Optimierung der Kosten von Amazon Keyspaces-Tabellen
  - Auswerten Ihrer Kosten auf Tabellenebene
    - So zeigen Sie die Kosten einer einzelnen Amazon Keyspaces-Tabelle an
    - Standardansicht von Cost Explorer
    - So verwenden Sie Tabellen-Tags in Cost Explorer
  - Evaluieren Sie den Kapazitätsmodus Ihrer Tabelle
    - Verfügbare Tabellenkapazitätsmodi
    - Fälle, in denen der On-Demand-Modus ausgewählt werden sollte

    - Fälle, in denen der Modus bereitgestellter Kapazität ausgewählt werden sollte
    - Zusätzliche Faktoren, die bei der Auswahl eines Tabellenkapazitätsmodus zu berücksichtigen sind
- Evaluieren Sie die Application Auto Scaling Scaling-Einstellungen Ihrer Tabelle
    - Grundlegendes zu Ihren Application Auto Scaling Scaling-Einstellungen
    - Wie identifiziert man Tabellen mit geringer Zielauslastung (<= 50%)
    - So bewältigen Sie Workloads mit saisonalen Schwankungen
    - So bewältigen Sie stark schwankende Workloads mit unbekannten Mustern
    - So bewältigen Sie Workloads mit verknüpften Anwendungen
- Identifizieren Sie Ihre ungenutzten Ressourcen, um die Kosten in Amazon Keyspaces zu optimieren
    - So ermitteln Sie nicht verwendete Ressourcen
    - Ermitteln von nicht verwendeten Tabellenressourcen
    - Bereinigen von nicht verwendeten Tabellenressourcen
    - Bereinigen ungenutzter Point-in-Time Recovery (PITR) -Backups
- Bewerten Sie Ihre Tabellennutzungsmuster, um Leistung und Kosten zu optimieren
    - Ausführen von weniger strikt konsistenten Lesevorgängen
    - Aktivieren von Time to Live (TTL)
- Auswerten Ihrer bereitgestellten Kapazität, um eine angemessene Bereitstellung zu erzielen
    - So rufen Sie Verbrauchsmetriken aus Ihren Amazon Keyspaces-Tabellen ab
    - So identifizieren Sie Amazon Keyspaces-Tabellen mit unzureichender Bereitstellung
    - So identifizieren Sie übermäßig bereitgestellte Amazon Keyspaces-Tabellen

Hauptunterschiede und Gestaltungsprinzipien des NoSQL-Designs

NoSQL-Datenbanksysteme wie Amazon Keyspaces verwenden alternative Modelle für das Datenmanagement, z. B. Schlüssel-Wert-Paare oder die Speicherung von Dokumenten. Wenn Sie von einem relationalen Datenbankmanagementsystem zu einem NoSQL-Datenbanksystem wie Amazon Keyspaces wechseln, ist es wichtig, die wichtigsten Unterschiede und spezifischen Entwurfsansätze zu verstehen.

Themen

- Unterschiede zwischen einem relationalen Datendesign und NoSQL
- Zwei Schlüsselkonzepte für das NoSQL-Design
- Ansatz für ein NoSQL-Design

Unterschiede zwischen einem relationalen Datendesign und NoSQL

Relationale Datenbankmanagementsysteme (RDBMS) und nNoSQL-Datenbanken besitzen unterschiedliche Vor- und Nachteile:

- In RDBMS können Daten flexibel abgerufen werden, Abfragen sind jedoch vergleichsweise kostspielig und können bei einer großen Zahl von Zugriffen nicht gut skaliert werden (siehe the section called “Datenmodellierung”).
- In einer NoSQL-Datenbank wie Amazon Keyspaces können Daten auf eine begrenzte Anzahl von Arten effizient abgefragt werden. Andernfalls können Abfragen teuer und langsam sein.

Diese Unterschiede führen dazu, dass das Datenbankdesign der beiden Systeme verschieden ist:

- In RDBMS entwerfen Sie das Design im Hinblick auf Flexibilität, ohne sich um Implementierungsdetails oder Leistung zu kümmern. Die Optimierung von Abfragen wirkt sich im Allgemeinen nicht auf das Schemadesign aus, eine Standardisierung ist jedoch wichtig.
- In Amazon Keyspaces entwerfen Sie Ihr Schema speziell, um die häufigsten und wichtigsten Abfragen so schnell und kostengünstig wie möglich zu gestalten. Die Datenstrukturen sind an die spezifischen Anforderungen Ihrer geschäftlichen Anwendungsfälle angepasst.

Zwei Schlüsselkonzepte für das NoSQL-Design

Das NoSQL-Design erfordert einen anderen Ansatz als das RDBMS-Design. Sie können für ein RDBMS ein standardisiertes Datenmodell entwickeln, ohne sich Gedanken über Zugriffsmuster machen zu müssen. Anschließend können Sie es erweitern, wenn neue Fragen und Abfrageanforderungen entstehen. Sie können jeden einzelnen Typ von Daten in einer eigenen Tabelle organisieren.

Wie sich das NoSQL-Design unterscheidet

- Im Gegensatz dazu sollten Sie nicht mit dem Entwerfen Ihres Schemas für Amazon Keyspaces beginnen, bis Sie wissen, welche Fragen es beantworten muss. Es ist äußerst wichtig, die geschäftlichen Probleme und Anwendungsfälle vor der Entwicklung des Schemas zu kennen.
- Sie sollten so wenige Tabellen wie möglich in einer Amazon Keyspaces-Anwendung verwalten. Weniger Tabellen sorgen für mehr Skalierbarkeit, erfordern weniger Rechteverwaltung und reduzieren den Overhead für Ihre Amazon Keyspaces-Anwendung. Dies kann auch dazu beitragen, die Backup-Kosten insgesamt niedrig zu halten.

Ansatz für ein NoSQL-Design

Der erste Schritt beim Entwerfen Ihrer Amazon Keyspaces-Anwendung besteht darin, die spezifischen Abfragemuster zu identifizieren, die das System erfüllen muss.

Insbesondere ist es wichtig, drei Basiseigenschaften der Zugriffsmuster Ihrer Anwendung zu kennen, bevor Sie mit dem Design beginnen:

- Datengröße: Wenn Sie wissen, wie viele Daten gleichzeitig gespeichert und angefordert werden, können Sie ermitteln, wie die Daten am effektivsten partitioniert werden können.
- Datenform: Statt Daten neu zu gestalten, wenn eine Abfrage verarbeitet wird (wie im Fall von RDBMS-Systemen), organisieren NoSQL-Datenbanken Daten so, dass ihre Form in der Datenbank dem entspricht, was abgefragt werden wird. Dies ist ein entscheidender Faktor, um Geschwindigkeit und Skalierbarkeit zu verbessern.
- Datengeschwindigkeit: Amazon Keyspaces skaliert, indem es die Anzahl der physischen Partitionen erhöht, die für die Verarbeitung von Abfragen verfügbar sind, und indem Daten effizient auf diese Partitionen verteilt werden. Wenn Sie im Voraus wissen, wie hoch die maximale Abfragelast sein wird, kann dies dazu beitragen, festzustellen, wie Daten partitioniert werden, um die I/O Kapazität optimal zu nutzen.

Nach der Identifizierung spezifischer Abfrageanforderungen können Sie die Daten nach allgemeinen Grundsätzen organisieren, denen die Leistung unterliegt:

- Speichern Sie verwandte Daten zusammen. Studien zur Optimierung von Routing-Tabellen, die vor 20 Jahren durchgeführt wurden, hatten zum Ergebnis, dass der "Referenzspeicherort" den wichtigsten Faktor für die Verkürzung der Reaktionszeit darstellt: Daten sollten daher zentral gespeichert werden. Das gilt auch heute für NoSQL-Systeme: Wenn verwandte Daten nahe

beieinander gespeichert werden, hat dies große Auswirkungen auf Kosten und Leistung. Statt verwandte Datenelemente über mehrere Tabellen zu verteilen, sollten Sie verwandte Elemente in Ihrem NoSQL-System so nahe beieinander wie möglich speichern.

In der Regel sollten Sie in einer Amazon Keyspaces-Anwendung so wenige Tabellen wie möglich verwalten.

Ausnahmen hiervon sind Fälle, in denen große Mengen von Zeitreihendaten oder Datensätze mit sehr unterschiedlichen Zugriffsmustern vorhanden sind. Eine einzelne Tabelle mit umgekehrten Indizes kann in der Regel einfache Abfragen unterstützen, um die komplexen hierarchischen Datenstrukturen zu erstellen und abzurufen, die Ihre Anwendung benötigt.

- Verwenden Sie eine Sortierreihenfolge. Verwandte Elemente können zusammen gruppiert und effizient abgefragt werden, wenn ihr Schlüsseldesign dazu führt, dass sie gemeinsam sortiert werden. Dies stellt eine wichtige Strategie für das NoSQL-Design dar.
- Verteilen Sie Abfragen. Es ist auch wichtig, dass sich eine große Anzahl von Abfragen nicht auf einen Teil der Datenbank konzentriert, wo sie die I/O Kapazität überschreiten könnten. Sie sollten stattdessen Datenschlüssel entwerfen, um die Zugriffe so gleichmäßig wie möglich über die Partitionen zu verteilen und Konzentrationspunkte zu vermeiden.

Diese allgemeinen Prinzipien lassen sich in einigen gängigen Entwurfsmustern umsetzen, die Sie verwenden können, um Daten in Amazon Keyspaces effizient zu modellieren.

Optimieren Sie die Client-Treiberverbindungen für die serverlose Umgebung

Für die Kommunikation mit Amazon Keyspaces können Sie einen der vorhandenen Apache Cassandra-Client-Treiber Ihrer Wahl verwenden. Da Amazon Keyspaces ein serverloser Service ist, empfehlen wir Ihnen, die Verbindungskonfiguration Ihres Client-Treibers für die Durchsatzanforderungen Ihrer Anwendung zu optimieren. In diesem Thema werden bewährte Methoden vorgestellt, darunter die Berechnung, wie viele Verbindungen Ihre Anwendung benötigt, sowie die Überwachung und Fehlerbehandlung von Verbindungen.

Themen

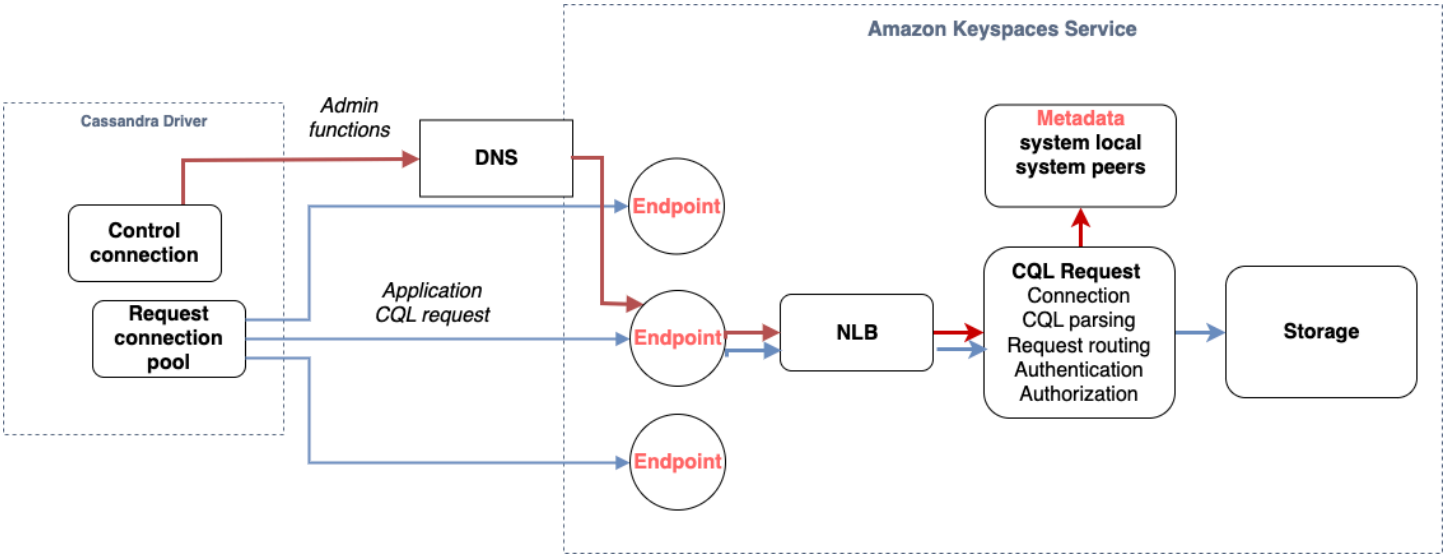
- So funktionieren Verbindungen in Amazon Keyspaces
- So konfigurieren Sie Verbindungen in Amazon Keyspaces
- So konfigurieren Sie die Wiederholungsrichtlinie für Verbindungen in Amazon Keyspaces

- So konfigurieren Sie Verbindungen über VPC-Endpunkte in Amazon Keyspaces
- So überwachen Sie Verbindungen in Amazon Keyspaces
- So behandeln Sie Verbindungsfehler in Amazon Keyspaces

So funktionieren Verbindungen in Amazon Keyspaces

Dieser Abschnitt gibt einen Überblick darüber, wie Client-Treiberverbindungen in Amazon Keyspaces funktionieren. Da eine Fehlkonfiguration des Cassandra-Clienttreibers zu `PerConnectionRequestExceeded` Ereignissen in Amazon Keyspaces führen kann, ist die Konfiguration der richtigen Anzahl von Verbindungen in der Client-Treiberkonfiguration erforderlich, um diese und ähnliche Verbindungsfehler zu vermeiden.

Bei der Verbindung mit Amazon Keyspaces benötigt der Treiber einen Seed-Endpunkt, um eine erste Verbindung herzustellen. Amazon Keyspaces verwendet DNS, um die erste Verbindung zu einem der vielen verfügbaren Endpunkte weiterzuleiten. Die Endpunkte sind mit Netzwerk-Loadbalancern verbunden, die wiederum eine Verbindung zu einem der Request-Handler in der Flotte herstellen. Nachdem die erste Verbindung hergestellt wurde, sammelt der Client-Treiber Informationen über alle verfügbaren Endpunkte aus der Tabelle. `system.peers` Mit diesen Informationen kann der Client-Treiber zusätzliche Verbindungen zu den aufgelisteten Endpunkten herstellen. Die Anzahl der Verbindungen, die der Clienttreiber herstellen kann, ist durch die Anzahl der lokalen Verbindungen begrenzt, die in den Einstellungen des Clienttreibers angegeben ist. Standardmäßig stellen die meisten Clienttreiber eine Verbindung pro Endpunkt her und richten einen Verbindungspool zu Cassandra ein und verteilen Abfragen über diesen Verbindungspool. Obwohl mehrere Verbindungen zu demselben Endpunkt hergestellt werden können, können sie hinter dem Network Load Balancer mit vielen verschiedenen Request-Handlern verbunden sein. Wenn Sie eine Verbindung über den öffentlichen Endpunkt herstellen, führt der Aufbau einer Verbindung zu jedem der neun in der `system.peers` Tabelle aufgeführten Endpunkte zu neun Verbindungen zu unterschiedlichen Request-Handlern.



So konfigurieren Sie Verbindungen in Amazon Keyspaces

Amazon Keyspaces unterstützt bis zu 3.000 CQL-Abfragen pro TCP-Verbindung pro Sekunde. Da die Anzahl der Verbindungen, die ein Treiber herstellen kann, unbegrenzt ist, empfehlen wir, nur 500 CQL-Anfragen pro Sekunde und Verbindung anzuvisieren, um Overhead, Traffic-Bursts und einen besseren Lastenausgleich zu berücksichtigen. Gehen Sie wie folgt vor, um sicherzustellen, dass die Treiberverbindung korrekt für die Anforderungen Ihrer Anwendung konfiguriert ist.

Erhöhen Sie die Anzahl der Verbindungen pro IP-Adresse, die Ihr Treiber in seinem Verbindungspool verwaltet.

- Die meisten Cassandra-Treiber richten einen Verbindungspool zu Cassandra ein und verteilen Abfragen über diesen Verbindungspool. Das Standardverhalten der meisten Treiber besteht darin, eine einzige Verbindung zu jedem Endpunkt herzustellen. Amazon Keyspaces stellt Treibern neun Peer-IP-Adressen zur Verfügung. Basierend auf dem Standardverhalten der meisten Treiber führt dies zu 9 Verbindungen. Amazon Keyspaces unterstützt bis zu 3.000 CQL-Abfragen pro TCP-Verbindung pro Sekunde. Daher beträgt der maximale CQL-Abfragedurchsatz eines Treibers mit den Standardeinstellungen 27.000 CQL-Abfragen pro Sekunde. Wenn Sie die Standardeinstellungen des Treibers verwenden, muss eine einzelne Verbindung möglicherweise mehr als den maximalen CQL-Abfragedurchsatz von 3.000 CQL-Abfragen pro Sekunde verarbeiten. Dies könnte zu Ereignissen führen. `PerConnectionRequestExceeded`
- Um `PerConnectionRequestExceeded` Ereignisse zu vermeiden, müssen Sie den Treiber so konfigurieren, dass zusätzliche Verbindungen pro Endpunkt hergestellt werden, um den Durchsatz zu verteilen.

- Als bewährte Methode in Amazon Keyspaces wird davon ausgegangen, dass jede Verbindung 500 CQL-Abfragen pro Sekunde unterstützen kann.
- Das bedeutet, dass Sie für eine Produktionsanwendung, die schätzungsweise 27.000 CQL-Abfragen pro Sekunde, verteilt auf die neun verfügbaren Endpunkte, unterstützen muss, sechs Verbindungen pro Endpunkt konfigurieren müssen. Dadurch wird sichergestellt, dass jede Verbindung nicht mehr als 500 Anfragen pro Sekunde verarbeitet.

Berechnen Sie die Anzahl der Verbindungen pro IP-Adresse, die Sie für Ihren Treiber konfigurieren müssen, basierend auf den Anforderungen Ihrer Anwendung.

Betrachten Sie das folgende Beispiel, um die Anzahl der Verbindungen zu ermitteln, die Sie pro Endpunkt für Ihre Anwendung konfigurieren müssen. Sie haben eine Anwendung, die 20.000 CQL-Abfragen pro Sekunde unterstützen muss, die aus 10.000 `INSERTSELECT`, 5.000 und 5.000 `DELETE` Vorgängen bestehen. Die Java-Anwendung wird auf drei Instances auf Amazon Elastic Container Service (Amazon ECS) ausgeführt, wobei jede Instance eine einzelne Sitzung mit Amazon Keyspaces einrichtet. Die Berechnung, mit der Sie abschätzen können, wie viele Verbindungen Sie für Ihren Treiber konfigurieren müssen, verwendet die folgende Eingabe.

1. Die Anzahl der Anfragen pro Sekunde, die Ihre Anwendung unterstützen muss.
2. Die Anzahl der verfügbaren Instances, wobei eine Instanz abgezogen wird, um Wartungs- oder Ausfallzeiten zu berücksichtigen.
3. Die Anzahl der verfügbaren Endpunkte. Wenn Sie eine Verbindung über öffentliche Endpunkte herstellen, stehen Ihnen neun Endpunkte zur Verfügung. Wenn Sie VPC-Endpunkte verwenden, stehen Ihnen je nach Region zwischen zwei und fünf Endpunkte zur Verfügung.
4. Verwenden Sie 500 CQL-Abfragen pro Sekunde und Verbindung als bewährte Methode für Amazon Keyspaces.
5. Runden Sie das Ergebnis zusammen.

In diesem Beispiel sieht die Formel so aus.

```
20,000 CQL queries / (3 instances - 1 failure) / 9 public endpoints / 500 CQL queries
 per second = ROUND(2.22) = 3
```

Basierend auf dieser Berechnung müssen Sie in der Treiberkonfiguration drei lokale Verbindungen pro Endpunkt angeben. Für Remoteverbindungen konfigurieren Sie nur eine Verbindung pro Endpunkt.

So konfigurieren Sie die Wiederholungsrichtlinie für Verbindungen in Amazon Keyspaces

Bei der Konfiguration der Wiederholungsrichtlinie für eine Verbindung zu Amazon Keyspaces empfehlen wir, die Amazon Keyspaces-Wiederholungsrichtlinie zu implementieren. `AmazonKeyspacesExponentialRetryPolicy` Diese Wiederholungsrichtlinie eignet sich besser für Wiederholungsversuche über verschiedene Verbindungen zu Amazon Keyspaces als die des Treibers. `DefaultRetryPolicy`

Mit der `AmazonKeyspacesExponentialRetryPolicy` können Sie die Anzahl der Wiederholungsversuche für die Verbindung konfigurieren, die Ihren Anforderungen entspricht. Standardmäßig ist die Anzahl der Wiederholungsversuche für auf `AmazonKeyspacesExponentialRetryPolicy` 3 festgelegt.

Ein weiterer Vorteil besteht darin, dass die Amazon Keyspaces-Wiederholungsrichtlinie die ursprüngliche vom Service zurückgegebene Ausnahme zurückgibt, die angibt, warum der Anforderungsversuch fehlgeschlagen ist. Die standardmäßige Wiederholungsrichtlinie gibt nur die generische Antwort zurück`NoHostAvailableException`, wodurch möglicherweise Erkenntnisse über den Fehler bei der Anfrage verborgen bleiben.

Um die Richtlinie für die Wiederholung von Anfragen mithilfe von zu konfigurieren`AmazonKeyspacesExponentialRetryPolicy`, empfehlen wir Ihnen, eine kleine Anzahl von Wiederholungsversuchen zu konfigurieren und alle zurückgegebenen Ausnahmen in Ihrem Anwendungscode zu behandeln.

Codebeispiele zur Implementierung von Wiederholungsrichtlinien finden Sie unter Amazon Keyspaces-Wiederholungsrichtlinien auf Github.

So konfigurieren Sie Verbindungen über VPC-Endpunkte in Amazon Keyspaces

Wenn Sie eine Verbindung über private VPC-Endpunkte herstellen, stehen Ihnen höchstwahrscheinlich 3 Endpunkte zur Verfügung. Die Anzahl der VPC-Endpunkte kann je nach Region unterschiedlich sein, basierend auf der Anzahl der Availability Zones und der Anzahl der Subnetze in der zugewiesenen VPC. Die Region USA Ost (Nord-Virginia) hat fünf Availability Zones und Sie können bis zu fünf Amazon Keyspaces-Endpunkte haben. Die Region USA West (Nordkalifornien) hat zwei Availability Zones und Sie können bis zu zwei Amazon Keyspaces-Endpunkte haben. Die Anzahl der Endpunkte hat keinen Einfluss auf die Skalierung, erhöht jedoch

die Anzahl der Verbindungen, die Sie in der Treiberkonfiguration einrichten müssen. Betrachten Sie das folgende Beispiel. Ihre Anwendung muss 20.000 CQL-Abfragen unterstützen und wird auf drei Instances auf Amazon ECS ausgeführt, wobei jede Instance eine einzelne Sitzung zu Amazon Keyspaces einrichtet. Der einzige Unterschied besteht darin, wie viele Endpunkte in den verschiedenen verfügbar sind. AWS-Regionen

In der Region USA Ost (Nord-Virginia) sind Verbindungen erforderlich:

```
20,000 CQL queries / (3 instances - 1 failure) / 5 private VPC endpoints / 500 CQL
 queries per second = 4 local connections
```

In der Region USA West (Nordkalifornien) sind Verbindungen erforderlich:

```
20,000 CQL queries / (3 instances - 1 failure) / 2 private VPC endpoints / 500 CQL
 queries per second = 10 local connections
```

> **Important**
>
> Bei der Verwendung von privaten VPC-Endpunkten sind zusätzliche Berechtigungen erforderlich, damit Amazon Keyspaces die verfügbaren VPC-Endpunkte dynamisch ermitteln und die Tabelle auffüllen kann. `system.peers` Weitere Informationen finden Sie unter the section called “Auffüllen von `system.peers` Tabelleneinträgen mit VPC-Endpunktinformationen der Schnittstelle”.

Wenn Sie über einen privaten VPC-Endpunkt mit einem anderen auf Amazon Keyspaces zugreifen, ist es wahrscheinlich AWS-Konto, dass Sie nur einen einzigen Amazon Keyspaces-Endpunkt sehen. Auch dies hat keinen Einfluss auf den Umfang des möglichen Durchsatzes zu Amazon Keyspaces, aber es kann erforderlich sein, dass Sie die Anzahl der Verbindungen in Ihrer Treiberkonfiguration erhöhen. Dieses Beispiel zeigt dieselbe Berechnung für einen einzelnen verfügbaren Endpunkt.

```
20,000 CQL queries / (3 instances - 1 failure) / 1 private VPC endpoints / 500 CQL
 queries per second = 20 local connections
```

Weitere Informationen zum kontoübergreifenden Zugriff auf Amazon Keyspaces über eine gemeinsam genutzte VPC finden Sie unter. the section called “Kontenübergreifenden Zugriff in einer gemeinsam genutzten VPC konfigurieren”

So überwachen Sie Verbindungen in Amazon Keyspaces

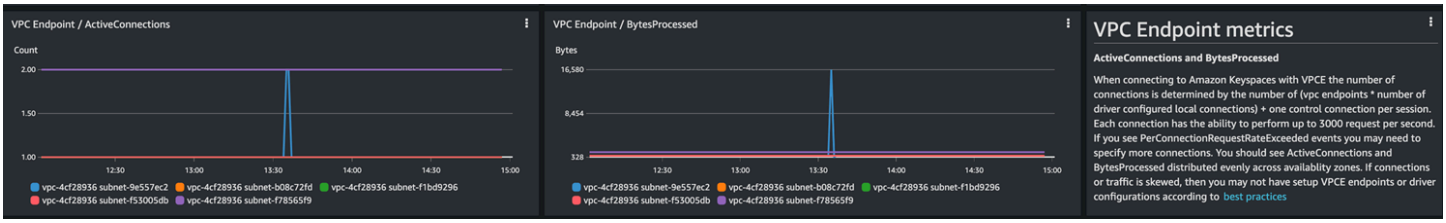
Um die Anzahl der Endpunkte zu ermitteln, mit denen Ihre Anwendung verbunden ist, können Sie die Anzahl der erkannten Peers in der `system.peers` Tabelle protokollieren. Das folgende Beispiel ist ein Beispiel für Java-Code, der die Anzahl der Peers ausgibt, nachdem die Verbindung hergestellt wurde.

```
ResultSet result = session.execute(new SimpleStatement("SELECT * FROM system.peers"));

logger.info("number of Amazon Keyspaces endpoints:" + result.all().stream().count());
```

> **Note**
>
> Die CQL-Konsole oder AWS -Konsole werden nicht in einer VPC bereitgestellt und verwenden daher den öffentlichen Endpunkt. Wenn die `system.peers` Abfrage von Anwendungen aus ausgeführt wird, die sich außerhalb des VPCE befinden, führt dies daher häufig zu 9 Peers. Es kann auch hilfreich sein, die IP-Adressen der einzelnen Peers auszudrucken.

Sie können auch die Anzahl der Peers beobachten, wenn Sie einen VPC-Endpunkt verwenden, indem Sie CloudWatch VPCE-Amazon-Metriken einrichten. In können Sie die Anzahl der Verbindungen sehen CloudWatch, die zum VPC-Endpunkt hergestellt wurden. Die Cassandra-Treiber stellen für jeden Endpunkt eine Verbindung her, um CQL-Abfragen zu senden, und eine Kontrollverbindung, um Systemtabelleninformationen zu sammeln. Das Bild unten zeigt die CloudWatch VPC-Endpunktmetriken nach dem Herstellen einer Verbindung zu Amazon Keyspaces mit einer in den Treibereinstellungen konfigurierten Verbindung. Die Metrik zeigt sechs aktive Verbindungen, bestehend aus einer Kontrollverbindung und fünf Verbindungen (eine pro Endpunkt in allen Availability Zones).



Um mit der Überwachung der Anzahl der Verbindungen mithilfe eines CloudWatch Diagramms zu beginnen, können Sie diese CloudFormation Vorlage bereitstellen, die GitHub im Amazon Keyspaces-Vorlagen-Repository verfügbar ist.

So behandeln Sie Verbindungsfehler in Amazon Keyspaces

Wenn das Kontingent von 3.000 Anfragen pro Verbindung überschritten wird, gibt Amazon Keyspaces ein `PerConnectionRequestExceeded` Ereignis zurück und der Cassandra-Treiber erhält eine `WriteTimeout` `ReadTimeout` OR-Ausnahme. Sie sollten diese Ausnahme mit exponentiellem Backoff in Ihrer Cassandra-Wiederholungsrichtlinie oder in Ihrer Anwendung erneut versuchen. Sie sollten einen exponentiellen Backoff angeben, um zu vermeiden, dass zusätzliche Anfragen gesendet werden.

Die standardmäßige Wiederholungsrichtlinie versucht, dies `try next host` im Abfrageplan zu tun. Da Amazon Keyspaces beim Herstellen einer Verbindung zum VPC-Endpunkt über ein bis drei verfügbare Endpunkte verfügen kann, werden `NoHostAvailableException` in Ihren Anwendungsprotokollen möglicherweise auch die `ReadTimeout` Ausnahmen zusätzlich zu den `WriteTimeout` und angezeigt. Sie können die von Amazon Keyspaces bereitgestellten Wiederholungsrichtlinien verwenden, die es auf demselben Endpunkt, aber über verschiedene Verbindungen hinweg wiederholen.

Beispiele für exponentielle Wiederholungsrichtlinien für Java finden Sie GitHub im Amazon Keyspaces-Repository für Java-Codebeispiele. Weitere Sprachbeispiele finden Sie auf Github im Amazon Keyspaces-Repository für Codebeispiele.

Bewährte Methoden zur Datenmodellierung: Empfehlungen für den Entwurf von Datenmodellen

Effektive Datenmodellierung ist entscheidend für die Optimierung der Leistung und Minimierung der Kosten bei der Arbeit mit Amazon Keyspaces (für Apache Cassandra). Dieses Thema behandelt wichtige Überlegungen und Empfehlungen für den Entwurf von Datenmodellen, die den Datenzugriffsmustern Ihrer Anwendung entsprechen.

- Design des Partitionsschlüssels — Der Partitionsschlüssel spielt eine entscheidende Rolle bei der Bestimmung, wie Daten auf die Partitionen in Amazon Keyspaces verteilt werden. Die Auswahl eines geeigneten Partitionsschlüssels kann sich erheblich auf die Abfrageleistung und die Durchsatzkosten auswirken. In diesem Abschnitt werden Strategien für die Gestaltung von Partitionsschlüsseln erörtert, die eine gleichmäßige Verteilung der Lese- und Schreibaktivität auf die Partitionen fördern.
- Wichtige Überlegungen:

- Einheitliche Verteilung der Aktivitäten — Streben Sie eine einheitliche Lese- und Schreibaktivität auf allen Partitionen an, um die Durchsatzkosten zu minimieren und die Burst-Kapazität effektiv zu nutzen.
- Zugriffsmuster — Stimmen Sie das Design Ihres Partitionsschlüssels auf die primären Datenzugriffsmuster Ihrer Anwendung ab.
- Partitionsgröße — Vermeiden Sie es, zu große Partitionen zu erstellen, da dies die Leistung beeinträchtigen und die Kosten in die Höhe treiben kann.

Um Datenmodelle einfacher zu visualisieren und zu entwerfen, können Sie die NoSQL Workbench verwenden.

Themen

- So verwenden Sie Partitionsschlüssel effektiv in Amazon Keyspaces

So verwenden Sie Partitionsschlüssel effektiv in Amazon Keyspaces

Der Primärschlüssel, der jede Zeile in einer Amazon Keyspaces-Tabelle eindeutig identifiziert, kann aus einer oder mehreren Partitionsschlüsselspalten bestehen, die bestimmen, in welchen Partitionen die Daten gespeichert werden, und einer oder mehreren optionalen Clusterspalten, die definieren, wie Daten innerhalb einer Partition geclustert und sortiert werden.

Da der Partitionsschlüssel die Anzahl der Partitionen festlegt, in denen Ihre Daten gespeichert sind und wie die Daten auf diese Partitionen verteilt werden, kann die Art und Weise, wie Sie Ihren Partitionsschlüssel auswählen, erhebliche Auswirkungen auf die Leistung Ihrer Abfragen haben. Im Allgemeinen sollten Sie Ihre Anwendung so entwerfen, dass sie auf allen Partitionen auf der Festplatte einheitlich abläuft.

Die gleichmäßige Verteilung der Lese- und Schreibaktivitäten Ihrer Anwendung auf alle Partitionen trägt zur Minimierung der Durchsatzkosten bei. Dies gilt sowohl für bedarfsgesteuerte als auch für bereitgestellte read/write Kapazitätsmodi. Wenn Sie beispielsweise den Modus mit bereitgestellter Kapazität verwenden, können Sie die Zugriffsmuster ermitteln, die Ihre Anwendung benötigt, und die Gesamtzahl der Lesekapazitätseinheiten (RCU) und Schreibkapazitätseinheiten (WCU) schätzen, die jede Tabelle benötigt. Amazon Keyspaces unterstützt Ihre Zugriffsmuster mithilfe des von Ihnen bereitgestellten Durchsatzes, sofern der Datenverkehr auf einer bestimmten Partition 3.000 RCUs und 1.000 WCUs nicht überschreitet.

Wenn eine Partition einen anhaltend hohen Lese- oder Schreibdurchsatz aufweist, kann Amazon Keyspaces die Partition je nach Verkehrsmuster automatisch in zwei neue Partitionen aufteilen. Jede neue Partition enthält eine Teilmenge der Zeilen der ursprünglichen Partition, wodurch der Durchsatz gleichmäßig auf beide Partitionen verteilt wird.

Amazon Keyspaces bietet zusätzliche Flexibilität bei der Bereitstellung von Durchsätzen pro Partition, indem es Burst-Kapazität bereitstellt. Weitere Informationen finden Sie unter. the section called “Verwenden Sie Burst-Kapazität”

Themen

- Verwenden Sie Write-Sharding, um Workloads gleichmäßig auf die Partitionen zu verteilen

Verwenden Sie Write-Sharding, um Workloads gleichmäßig auf die Partitionen zu verteilen

Eine Möglichkeit, Schreibvorgänge besser auf eine Partition in Amazon Keyspaces zu verteilen, besteht darin, den Speicherplatz zu erweitern. Dies kann auf verschiedene Arten geschehen. Sie können eine zusätzliche Partitionsschlüsselspalte hinzufügen, in die Sie Zufallszahlen schreiben, um die Zeilen auf die Partitionen zu verteilen. Oder Sie können eine Zahl verwenden, die basierend auf einer Abfrage berechnet wird.

Sharding mit zusammengesetzten Partitionsschlüsseln und Zufallswerten

Eine Strategie, um Lasten gleichmäßiger auf eine Partition zu verteilen, besteht darin, eine zusätzliche Partitionsschlüsselspalte hinzuzufügen, in die Sie Zufallszahlen schreiben. Anschließend werden die Schreibvorgänge zufällig auf den größeren Raum verteilt.

Stellen Sie sich zum Beispiel die folgende Tabelle vor, in der ein einziger Partitionsschlüssel ein Datum darstellt.

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   title text,
   description int,
   PRIMARY KEY (publish_date));
```

Um diese Tabelle gleichmäßiger auf die Partitionen zu verteilen, könnten Sie eine zusätzliche Partitionsschlüsselspalte hinzufügen`shard`, in der Zufallszahlen gespeichert werden. Beispiel:

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   shard int,
   title text,
   description int,
   PRIMARY KEY ((publish_date, shard)));
```

Beim Einfügen von Daten können Sie eine Zufallszahl zwischen 1 und 200 für die `shard` Spalte wählen. Dies ergibt zusammengesetzte Partitionsschlüsselwerte wie`(2020-07-09, 1)`,`(2020-07-09, 2)`, und so weiter, bis`(2020-07-09, 200)`. Durch die Randomisierung des Partitionsschlüssels werden die Schreibvorgänge in die Tabelle jeden Tag gleichmäßig auf mehrere Partitionen verteilt. Dies führt zu einer besseren Parallelverarbeitung und einem höheren Gesamtdurchsatz.

Um jedoch alle Zeilen für einen bestimmten Tag zu lesen, müssten Sie die Zeilen nach allen Shards abfragen und dann die Ergebnisse zusammenführen. Beispielsweise würden Sie zuerst eine `SELECT` Anweisung für den Wert `(2020-07-09, 1)` des Partitionsschlüssels ausgeben. Geben Sie dann eine weitere `SELECT` Anweisung für `(2020-07-09, 2)` usw. aus`(2020-07-09, 200)`. Schließlich müsste Ihre Anwendung die Ergebnisse all dieser `SELECT` Anweisungen zusammenführen.

Sharding mit zusammengesetzten Partitionsschlüsseln und berechneten Werten

Mit einer Randomisierungsstrategie kann der Schreibdurchsatz erheblich verbessert werden. Es ist jedoch schwierig, eine bestimmte Zeile zu lesen, da Sie nicht wissen, welcher Wert in die `shard` Spalte geschrieben wurde, als die Zeile geschrieben wurde. Um das Lesen einzelner Zeilen zu erleichtern, können Sie eine andere Strategie verwenden. Anstatt eine Zufallszahl zu verwenden, um die Zeilen auf die Partitionen zu verteilen, verwenden Sie eine Zahl, die Sie auf der Grundlage eines Objekts berechnen können, das Sie abfragen möchten.

Denken Sie an das vorherige Beispiel, in dem eine Tabelle das Datum des aktuellen Tages im Partitionsschlüssel verwendet. Nehmen wir nun an, dass jede Zeile über eine `title` Spalte verfügt, auf die zugegriffen werden kann, und dass Sie Zeilen in den meisten Fällen nicht nur nach Datum, sondern auch nach Titel suchen müssen. Bevor Ihre Anwendung die Zeile in die Tabelle schreibt, könnte sie anhand des Titels einen Hashwert berechnen und diesen zum Füllen der `shard` Spalte verwenden. Die Berechnung könnte eine Zahl zwischen 1 und 200 ergeben, die ähnlich wie bei der Zufallsstrategie relativ gleichmäßig verteilt ist.

Eine einfache Berechnung würde wahrscheinlich ausreichen, z. B. das Produkt der UTF-8 Codepunktwerte für die Zeichen im Titel, Modulo 200, + 1. Der Wert des zusammengesetzten Partitionsschlüssels wäre dann die Kombination aus Datum und Berechnungsergebnis.

Mit dieser Strategie werden die Schreibvorgänge gleichmäßig auf die Partitions-Schlüsselwerte und somit auch über die physischen Partitionen verteilt. Sie können problemlos eine `SELECT` Anweisung für eine bestimmte Zeile und ein bestimmtes Datum ausführen, da Sie den Partitionsschlüsselwert für einen bestimmten `title` Wert berechnen können.

Um alle Zeilen für einen bestimmten Tag zu lesen, müssen Sie trotzdem `SELECT` jeden der `(2020-07-09, N)` Schlüssel (wo `N` 1—200) benötigen, und Ihre Anwendung muss dann alle Ergebnisse zusammenführen. Der Vorteil dabei ist, dass Sie so vermeiden, dass ein einzelner „Hot Partition”-Schlüsselwert die gesamte Workload übernimmt.

Optimierung der Kosten von Amazon Keyspaces-Tabellen

In diesem Abschnitt werden bewährte Methoden zur Optimierung der Kosten für Ihre vorhandenen Amazon Keyspaces-Tabellen beschrieben. Sie sollten sich die folgenden Strategien zur Kostenoptimierung ansehen, um herauszufinden, welche davon Ihren Anforderungen am besten entspricht, und sie iterativ angehen. Jede Strategie bietet einen Überblick darüber, was sich auf Ihre Kosten auswirken könnte, wie Sie nach Möglichkeiten zur Kostenoptimierung suchen können, sowie präskriptive Anleitungen zur Implementierung dieser bewährten Verfahren, um Ihnen beim Sparen zu helfen.

Themen

- Auswerten Ihrer Kosten auf Tabellenebene
- Evaluieren Sie den Kapazitätsmodus Ihrer Tabelle
- Evaluieren Sie die Application Auto Scaling Scaling-Einstellungen Ihrer Tabelle
- Identifizieren Sie Ihre ungenutzten Ressourcen, um die Kosten in Amazon Keyspaces zu optimieren
- Bewerten Sie Ihre Tabellennutzungsmuster, um Leistung und Kosten zu optimieren
- Auswerten Ihrer bereitgestellten Kapazität, um eine angemessene Bereitstellung zu erzielen

Auswerten Ihrer Kosten auf Tabellenebene

Mit dem Cost Explorer Explorer-Tool im AWS-Managementkonsole können Sie die Kosten nach Typ aufgeschlüsselt anzeigen, z. B. Lese-, Schreib-, Speicher- und Backup-Gebühren. Sie können diese Kosten auch nach Zeitraum zusammengefasst sehen, beispielsweise die Kosten für einen Monat oder einen Tag.

Eine häufige Herausforderung bei Cost Explorer besteht darin, dass Sie die Kosten nur einer bestimmten Tabelle nicht einfach überprüfen können, da Sie mit Cost Explorer nicht nach den Kosten einer bestimmten Tabelle filtern oder gruppieren können. Sie können die metrische Größe der fakturierbaren Tabelle (Byte) jeder Tabelle in der Amazon Keyspaces-Konsole auf der Registerkarte Monitor der Tabelle einsehen. Wenn Sie mehr kostenbezogene Informationen pro Tabelle benötigen, erfahren Sie in diesem Abschnitt, wie Sie mithilfe von Tagging eine Kostenanalyse für einzelne Tabellen im Cost Explorer durchführen können.

Themen

- So zeigen Sie die Kosten einer einzelnen Amazon Keyspaces-Tabelle an
- Standardansicht von Cost Explorer
- So verwenden Sie Tabellen-Tags in Cost Explorer

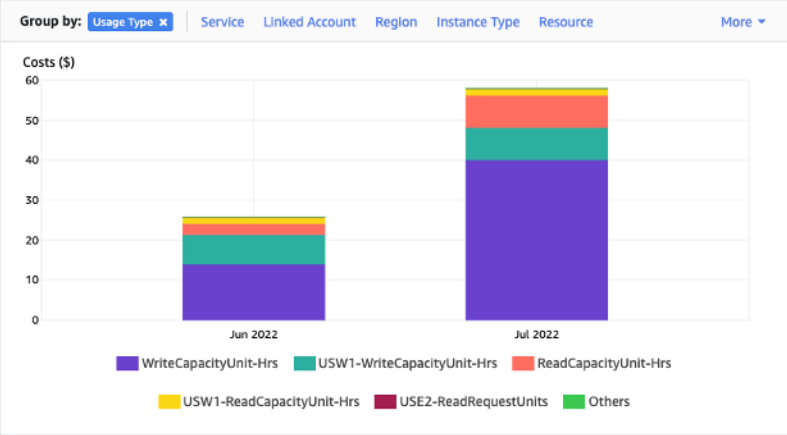
So zeigen Sie die Kosten einer einzelnen Amazon Keyspaces-Tabelle an

In der Konsole können Sie grundlegende Informationen zu einer Amazon Keyspaces-Tabelle einsehen, darunter das Primärschlüsselschema, die Größe der fakturierbaren Tabelle und kapazitätsbezogene Kennzahlen. Sie können die Größe der Tabelle verwenden, um die monatlichen Speicherkosten für die Tabelle zu berechnen. Zum Beispiel 0,25$ pro GB im Osten der USA (Nord-Virginia) AWS-Region.

Wenn die Tabelle den Modus für bereitgestellte Kapazität verwendet, werden auch die aktuellen Einstellungen für die Lesekapazitätseinheit (RCU) und die Schreibkapazitätseinheit (WCU) zurückgegeben. Sie können diese Informationen verwenden, um die aktuellen Lese- und Schreibkosten für die Tabelle zu berechnen. Beachten Sie, dass sich diese Kosten ändern können, insbesondere wenn Sie die Tabelle mit der automatischen Skalierung von Amazon Keyspaces konfiguriert haben.

Standardansicht von Cost Explorer

Die Standardansicht im Cost Explorer enthält Diagramme, in denen die Kosten der verbrauchten Ressourcen, beispielsweise Durchsatz und Speicherplatz, dargestellt werden. Sie können diese Kosten nach Zeitraum gruppieren, z. B. nach Gesamtkosten nach Monat oder Tag. Die Kosten für Speicher, Lese- und Schreibvorgänge und andere Kategorien können ebenfalls aufgeschlüsselt und verglichen werden.



So verwenden Sie Tabellen-Tags in Cost Explorer

Standardmäßig bietet der Cost Explorer keine Zusammenfassung der Kosten für eine bestimmte Tabelle, da er die Kosten mehrerer Tabellen zu einer Summe zusammenfasst. Sie können jedoch AWS -Ressourcen-Tagging verwenden, um jede Tabelle mit einem Metadaten-Tag zu identifizieren. Tags sind Schlüssel-Wert-Paare, die Sie für eine Vielzahl von Zwecken verwenden können, z. B. um alle Ressourcen zu identifizieren, die zu einem Projekt oder einer Abteilung gehören. Weitere Informationen finden Sie unter the section called “Mit Tags arbeiten”.

In diesem Beispiel verwenden wir eine Tabelle mit dem Namen. MyTable

1. Setze ein Tag mit dem Schlüssel von table_name und dem Wert von. MyTable
2. Aktivieren Sie das Tag in Cost Explorer und filtern Sie dann nach dem Tag-Wert, um einen besseren Einblick in die Kosten der einzelnen Tabellen zu erhalten.

> **Note**
>
> Es kann ein oder zwei Tage dauern, bis das Tag in Cost Explorer angezeigt wird.

Sie können Metadaten-Tags selbst in der Konsole oder programmgesteuert mit CQL, dem oder dem AWS CLI SDK festlegen. AWS Überlegen Sie sich, im Rahmen des neuen Tabellenerstellungsprozesses Ihrer Organisation die Festlegung eines table_name-Tags zu erfordern. Weitere Informationen finden Sie unter the section called “Erstellen Sie Berichte zur Kostenverteilung”.

Evaluieren Sie den Kapazitätsmodus Ihrer Tabelle

Dieser Abschnitt bietet einen Überblick darüber, wie Sie den geeigneten Kapazitätsmodus für Ihre Amazon Keyspaces-Tabelle auswählen. Jeder Modus ist auf die Anforderungen einer anderen Workload abgestimmt, sowohl was die Reaktionsfähigkeit auf Durchsatzänderungen, als auch was die Abrechnung der Nutzung anbelangt. Sie müssen diese Faktoren bei Ihrer Entscheidung abwägen.

Themen

- Verfügbare Tabellenkapazitätsmodi
- Fälle, in denen der On-Demand-Modus ausgewählt werden sollte
- Fälle, in denen der Modus bereitgestellter Kapazität ausgewählt werden sollte
- Zusätzliche Faktoren, die bei der Auswahl eines Tabellenkapazitätsmodus zu berücksichtigen sind

Verfügbare Tabellenkapazitätsmodi

Wenn Sie eine Amazon Keyspaces-Tabelle erstellen, müssen Sie entweder den Modus „On-Demand-Kapazität“ oder „Bereitgestellte Kapazität“ wählen. Weitere Informationen finden Sie unter the section called “ read/write Kapazitätsmodi konfigurieren”.

On-demand Kapazitätsmodus

Der On-Demand-Kapazitätsmodus ist so konzipiert, dass Sie die Kapazität Ihrer Amazon Keyspaces-Tabelle nicht mehr planen oder bereitstellen müssen. In diesem Modus bearbeitet Ihre Tabelle Anfragen sofort, ohne dass Ressourcen nach oben oder unten skaliert werden müssen (bis zum Doppelten des vorherigen Spitzendurchsatzes der Tabelle).

On-demand Tabellen werden abgerechnet, indem die Anzahl der tatsächlichen Anfragen mit der Tabelle verglichen wird. Sie zahlen also nur für das, was Sie tatsächlich nutzen, und nicht für das, was bereitgestellt wurde.

Modus bereitgestellter Kapazität

Der Modus für bereitgestellte Kapazität ist ein traditionelleres Modell, bei dem Sie entweder direkt oder mit Hilfe von Application Auto Scaling definieren können, wie viel Kapazität die Tabelle für Anfragen zur Verfügung hat. Da zu jedem Zeitpunkt eine bestimmte Kapazität für die Tabelle bereitgestellt wird, basiert die Abrechnung auf der bereitgestellten Kapazität und nicht auf der Anzahl der Anforderungen. Eine Überschreitung der zugewiesenen Kapazität kann auch dazu führen, dass die Tabelle Anfragen ablehnt und die Benutzererfahrung der Benutzer Ihrer Anwendung beeinträchtigt.

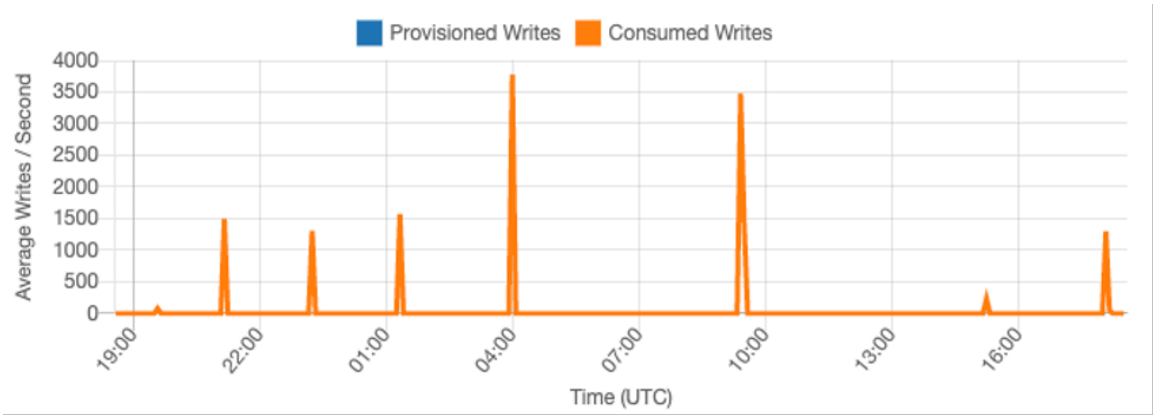
Der Modus „Bereitgestellte Kapazität“ erfordert ein ausgewogenes Verhältnis zwischen der Nichtbereitstellung und der Unterbereitstellung der Tabelle, um sowohl das geringe Auftreten von Fehlern bei unzureichender Durchsatzkapazität als auch die Optimierung der Kosten zu erreichen.

Fälle, in denen der On-Demand-Modus ausgewählt werden sollte

Bei der Kostenoptimierung ist der On-Demand-Modus die beste Wahl, wenn Sie eine unvorhersehbare Arbeitslast haben, die der in der folgenden Grafik dargestellten ähnelt.

Diese Faktoren tragen zu dieser Art von Arbeitslast bei:

- Unvorhersehbares Timing der Anforderungen (dadurch Datenverkehrsspitzen)
- Variables Volumen der Anforderungen (aufgrund von Batch-Workloads)
- Sinkt in einer bestimmten Stunde auf Null oder unter 18% des Spitzenwerts (aufgrund von Entwicklungs- oder Testumgebungen)



Bei Workloads mit den oben genannten Merkmalen kann die Verwendung von Application Auto Scaling, um genügend Kapazität aufrechtzuerhalten, damit die Tabelle auf Datenverkehrsspitzen reagieren kann, zu unerwünschten Ergebnissen führen. Entweder ist die Tabelle überdimensioniert und kostet mehr als nötig, oder die Tabelle ist zu wenig bereitgestellt und Anfragen führen zu

unnötigen Durchsatzfehlern bei niedriger Kapazität. In solchen Fällen sind On-Demand-Tabellen die bessere Wahl.

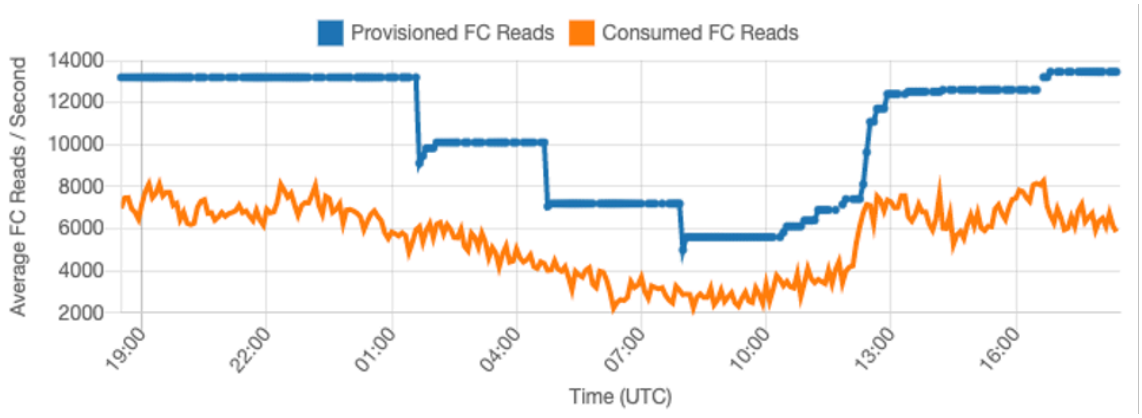
Da On-Demand-Tabellen auf Anfrage abgerechnet werden, müssen Sie auf Tabellenebene nichts weiter tun, um die Kosten zu optimieren. Sie sollten Ihre On-Demand-Tabellen regelmäßig überprüfen, um sicherzustellen, dass der Workload weiterhin die oben genannten Merkmale aufweist. Wenn sich die Arbeitslast stabilisiert hat, sollten Sie erwägen, zum Bereitstellungsmodus zu wechseln, um die Kostenoptimierung aufrechtzuerhalten.

Fälle, in denen der Modus bereitgestellter Kapazität ausgewählt werden sollte

Ein idealer Workload für den Modus mit bereitgestellter Kapazität ist ein Workload mit einem besser vorhersehbaren Nutzungsmuster, wie in der folgenden Grafik dargestellt.

Die folgenden Faktoren tragen zu einer vorhersehbaren Arbeitslast bei:

- Predicable/cyclical Verkehr für eine bestimmte Stunde oder einen bestimmten Tag
- Begrenzte kurzfristige Datenverkehrsspitzen



Da das Verkehrsvolumen innerhalb einer bestimmten Zeit oder eines bestimmten Tages stabiler ist, können Sie die bereitgestellte Kapazität relativ nahe an der tatsächlich verbrauchten Kapazität der Tabelle festlegen. Bei der Kostenoptimierung einer Tabelle mit bereitgestellter Kapazität geht es letztlich darum, die bereitgestellte Kapazität (blaue Linie) so nah wie möglich an die verbrauchte Kapazität (orange Linie) heranzuführen, ohne die Anzahl der `ThrottledRequests` Ereignisse in der Tabelle zu erhöhen. Der Abstand zwischen den beiden Zeilen ist sowohl verschwendete Kapazität als auch eine Versicherung gegen eine schlechte Benutzererfahrung aufgrund unzureichender Durchsatzkapazitätsfehler.

Amazon Keyspaces bietet Application Auto Scaling für bereitgestellte Kapazitätstabellen, das diese automatisch in Ihrem Namen ausgleicht. Sie können Ihre verbrauchte Kapazität im Laufe des Tages verfolgen und die bereitgestellte Kapazität der Tabelle anhand einer Handvoll Variablen konfigurieren.

Einheiten für minimale Kapazität

Sie können die Mindestkapazität einer Tabelle festlegen, um das Auftreten von Fehlern bei unzureichender Durchsatzkapazität zu begrenzen. Die Kosten der Tabelle werden dadurch jedoch nicht gesenkt. Wenn Ihre Tabelle Perioden mit geringer Auslastung aufweist, gefolgt von einem plötzlichen Anstieg hoher Auslastung, kann die Festlegung des Minimums verhindern, dass Application Auto Scaling die Tabellenkapazität zu niedrig festlegt.

Einheiten für maximale Kapazität

Sie können die maximale Kapazität einer Tabelle festlegen, um zu verhindern, dass eine Tabelle stärker als beabsichtigt skaliert wird. Erwägen Sie, ein Maximum für Entwicklungs- oder Testtabellen festzulegen, bei denen umfangreiche Auslastungstests nicht erwünscht sind. Sie können für jede Tabelle einen Höchstwert festlegen. Achten Sie jedoch darauf, diese Einstellung regelmäßig mit der Tabellenbasislinie abzugleichen, wenn Sie sie in der Produktion verwenden, um versehentliche Fehler bei unzureichender Durchsatzkapazität zu vermeiden.

Zielauslastung

Die Festlegung einer Zielauslastung der Tabelle ist das primäre Mittel zur Kostenoptimierung für eine Tabelle mit bereitgestellter Kapazität. Wenn Sie hier einen niedrigeren Prozentwert angeben, erhöht sich die Überbelegung der Tabelle, was die Kosten erhöht, aber das Risiko von Fehlern bei unzureichender Durchsatzkapazität verringert. Wenn Sie einen höheren Prozentwert festlegen, verringert sich die Überbelegung der Tabelle, erhöht sich jedoch das Risiko von Fehlern bei unzureichender Durchsatzkapazität.

Zusätzliche Faktoren, die bei der Auswahl eines Tabellenkapazitätsmodus zu berücksichtigen sind

Bei der Entscheidung zwischen den beiden Kapazitätsmodi sollten einige zusätzliche Faktoren berücksichtigt werden.

Berücksichtigen Sie bei der Entscheidung zwischen den beiden Tabellenmodi, wie stark sich dieser zusätzliche discount auf die Kosten des Tisches auswirkt. In vielen Fällen kann selbst eine relativ unvorhersehbare Arbeitslast kostengünstiger sein, wenn sie auf einer Tabelle mit überdimensionierter Kapazität und reservierter Kapazität ausgeführt wird.

Verbessern der Vorhersehbarkeit Ihrer Workload

In manchen Situationen kann ein Workload scheinbar sowohl ein vorhersehbares als auch ein unvorhersehbares Muster aufweisen. Dies kann zwar mit einer On-Demand-Tabelle problemlos unterstützt werden, aber die Kosten werden wahrscheinlich niedriger sein, wenn die unvorhersehbaren Muster in der Arbeitslast verbessert werden können.

Eine der häufigsten Ursachen für diese Muster sind Batch-Importe. Diese Art von Verkehr kann die Basiskapazität der Tabelle oft so stark überschreiten, dass bei der Ausführung Fehler mit unzureichender Durchsatzkapazität auftreten würden. Um eine Workload wie diese in einer Tabelle mit bereitgestellter Kapazität auszuführen, sollten Sie die folgenden Optionen in Betracht ziehen:

- Wenn der Batch zu geplanten Zeiten ausgeführt wird, können Sie eine Erhöhung der Kapazität Ihrer Anwendung für die auto Skalierung planen, bevor sie ausgeführt wird.
- Wenn der Batch zufällig ausgeführt wird, sollten Sie versuchen, die für die Ausführung benötigte Zeit zu verlängern, anstatt ihn so schnell wie möglich auszuführen.
- Fügen Sie dem Import eine Anlaufphase hinzu, in der die Geschwindigkeit des Imports gering beginnt, aber über einige Minuten langsam erhöht wird, bis Application Auto Scaling die Möglichkeit hatte, mit der Anpassung der Tabellenkapazität zu beginnen.

Evaluieren Sie die Application Auto Scaling Scaling-Einstellungen Ihrer Tabelle

Dieser Abschnitt bietet einen Überblick darüber, wie Sie die Application Auto Scaling Scaling-Einstellungen Ihrer Amazon Keyspaces-Tabellen auswerten können. Amazon Keyspaces Application Auto Scaling ist eine Funktion, die den Tabellendurchsatz auf der Grundlage Ihres Anwendungsdatenverkehrs und Ihrer Zielnutzungsmetrik verwaltet. Dadurch wird sichergestellt, dass Ihre Tabellen über die für Ihre Anwendungsmuster erforderliche Kapazität verfügen.

Der Application Auto Scaling Scaling-Dienst überwacht Ihre aktuelle Tabellenauslastung und vergleicht sie mit dem Zielnutzungswert:`TargetValue`. Er benachrichtigt Sie, wenn es an der Zeit ist, die zugewiesene Kapazität zu erhöhen oder zu verringern.

Themen

- Grundlegendes zu Ihren Application Auto Scaling Scaling-Einstellungen
- Wie identifiziert man Tabellen mit geringer Zielauslastung (<= 50%)
- So bewältigen Sie Workloads mit saisonalen Schwankungen

- So bewältigen Sie stark schwankende Workloads mit unbekannten Mustern
- So bewältigen Sie Workloads mit verknüpften Anwendungen

Grundlegendes zu Ihren Application Auto Scaling Scaling-Einstellungen

Die Festlegung des richtigen Werts für die Zielauslastung, den ersten Schritt und die Endwerte erfordert die Beteiligung Ihres Operations-Teams. Auf diese Weise können Sie die Werte auf der Grundlage der historischen Anwendungsnutzung, die zum Auslösen der Application Auto Scaling Scaling-Richtlinien verwendet wird, richtig definieren. Das Auslastungsziel ist der Prozentsatz Ihrer Gesamtkapazität, der während eines bestimmten Zeitraums erreicht werden muss, bevor die Application Auto Scaling Scaling-Regeln gelten.

Wenn Sie ein hohes Auslastungsziel festlegen (ein Ziel von etwa 90%), bedeutet dies, dass Ihr Traffic für einen bestimmten Zeitraum höher als 90% sein muss, bevor Application Auto Scaling aktiviert wird. Ein hohes Auslastungsziel sollten Sie nur verwenden, wenn Ihre Anwendung sehr konstant arbeitet und keine Datenverkehrsspitzen verzeichnet.

Wenn Sie eine sehr niedrige Auslastung festlegen (ein Ziel von weniger als 50%), bedeutet dies, dass Ihre Anwendung 50% der bereitgestellten Kapazität erreichen müsste, bevor sie eine Application Auto Scaling Scaling-Richtlinie auslöst. Sofern Ihr Anwendungsdatenverkehr nicht extrem schnell zunimmt, führt dies in der Regel zu ungenutzter Kapazität und einer Ressourcenverschwendung.

Wie identifiziert man Tabellen mit geringer Zielauslastung (<= 50%)

Sie können entweder das AWS CLI oder verwenden AWS-Managementkonsole , um die Auto Scaling-Richtlinien `TargetValues` für Ihre Anwendung in Ihren Amazon Keyspaces-Ressourcen zu überwachen und zu identifizieren.

> **Note**
>
> Wenn Sie Tabellen mit mehreren Regionen im Modus für bereitgestellte Kapazität mit Amazon Keyspaces Auto Scaling verwenden, stellen Sie sicher, dass Sie die Amazon Keyspaces-API-Operationen verwenden, um Auto Scaling zu konfigurieren. Die zugrunde liegenden API-Operationen für Application Auto Scaling, die Amazon Keyspaces in Ihrem Namen aufruft, verfügen nicht über Funktionen für mehrere Regionen. Weitere Informationen finden Sie unter the section called “Bereitgestellte Kapazität und Auto-Scaling-Einstellungen für eine Tabelle mit mehreren Regionen anzeigen”.

AWS CLI

1. Verwenden Sie den folgenden Befehl, um die gesamte Ressourcenliste zurückzugeben:

```
aws application-autoscaling describe-scaling-policies --service-namespace
 cassandra
```

Dieser Befehl gibt die gesamte Liste der Application Auto Scaling Scaling-Richtlinien zurück, die für jede Amazon Keyspaces-Ressource ausgestellt wurden. Wenn Sie nur die Ressourcen aus einer bestimmten Tabelle abrufen möchten, können Sie den `–resource-id parameter` hinzufügen. Beispiel:

```
aws application-autoscaling describe-scaling-policies --service-namespace
 cassandra --resource-id "keyspace/keyspace-name/table/table-name”
```

2. Geben Sie nur die Auto Scaling-Richtlinien für eine bestimmte Tabelle zurück, indem Sie den folgenden Befehl ausführen

```
aws application-autoscaling describe-scaling-policies --service-namespace
 cassandra --resource-id "keyspace/keyspace-name/table/table-name”
```

Die Werte für die Application Auto Scaling Scaling-Richtlinien sind unten hervorgehoben. Sie müssen sicherstellen, dass der Zielwert über 50% liegt, um eine übermäßige Bereitstellung zu vermeiden. Das Ergebnis sollte etwa wie folgt aussehen:

```
{
    "ScalingPolicies": [
        {
            "PolicyARN": "arn:aws:autoscaling:us-
east-1:111122223333:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-
scaling-policy",
            "PolicyName": $<full-table-name>”,
            "ServiceNamespace": "cassandra",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:table:WriteCapacityUnits",
            "PolicyType": "TargetTrackingScaling",
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "KeyspacesWriteCapacityUtilization"
```

```
                }
            },
            "Alarms": [
                ...
            ],
            "CreationTime": "2022-03-04T16:23:48.641000+10:00"
        },
        {
            "PolicyARN": "arn:aws:autoscaling:us-
east-1:111122223333:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-
scaling-policy",
            "PolicyName":$<full-table-name>”,
            "ServiceNamespace": "cassandra",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:table:ReadCapacityUnits",
            "PolicyType": "TargetTrackingScaling",
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "KeyspacesReadCapacityUtilization"
                }
            },
            "Alarms": [
                ...
            ],
            "CreationTime": "2022-03-04T16:23:47.820000+10:00"
        }
    ]
}
```

AWS-Managementkonsole

1. Melden Sie sich bei der an AWS-Managementkonsole und rufen Sie die CloudWatch Serviceseite unter Erste Schritte mit dem auf. AWS-Managementkonsole Wählen Sie bei AWS-Region Bedarf das entsprechende aus.
2. Wählen Sie in der linken Navigationsleiste die Option Tables (Tabellen) aus. Wählen Sie auf der Seite Tables (Tabellen) den Namen der Tabelle aus.
3. Überprüfen Sie auf der Seite Tabellendetails auf der Registerkarte Kapazität die Einstellungen für Application Auto Scaling Ihrer Tabelle.

Bei Zielauslastungswerten von maximal 50 % sollten Sie Ihre Tabellenauslastungsmetriken prüfen, um festzustellen, ob eine zu geringe oder eine übermäßige Bereitstellung vorliegt.

So bewältigen Sie Workloads mit saisonalen Schwankungen

Stellen Sie sich folgendes Szenario vor: Ihre Anwendung läuft die meiste Zeit unter einem minimalen Durchschnittswert, doch das Auslastungsziel ist niedrig. Somit kann Ihre Anwendung schnell auf Ereignisse reagieren, die zu bestimmten Tageszeiten auftreten, Sie verfügen über genügend Kapazität und es kommt nicht zu Drosselungen. Ein solches Szenario kommt häufig vor, wenn eine Anwendung während der normalen Bürozeiten (9 bis 17 Uhr) sehr stark ausgelastet ist, außerhalb der Geschäftszeiten jedoch auf niedrigem Niveau läuft. Da einige Benutzer vor 9 Uhr eine Verbindung herstellen, verwendet die Anwendung diesen niedrigen Schwellenwert, um schnell hochzufahren und zu Spitzenzeiten die erforderliche Kapazität zu erreichen.

Dieses Szenario könnte wie folgt aussehen:

- Zwischen 17 Uhr und 9 Uhr bleiben die `ConsumedWriteCapacityUnits`-Einheiten zwischen 90 und 100.
- Benutzer beginnen vor 9 Uhr, eine Verbindung zu der Anwendung herzustellen, und die Kapazitätseinheiten steigen erheblich an (der maximale Wert, den Sie gesehen haben, beträgt 1 500 WCU).
- Im Durchschnitt variiert Ihre Anwendungsnutzung während der Arbeitszeit zwischen 800 und 1 200.

Wenn das vorherige Szenario auf Ihre Anwendung zutrifft, sollten Sie die Application Auto Scaling nach Zeitplan in Betracht ziehen, bei der in Ihrer Tabelle immer noch eine Regel für Application Auto Scaling konfiguriert sein könnte, aber mit einer weniger aggressiven Zielauslastung, die die zusätzliche Kapazität nur in den von Ihnen benötigten Intervallen bereitstellt.

Mit dem können Sie die AWS CLI folgenden Schritte ausführen, um eine geplante Auto Scaling-Regel zu erstellen, die auf der Grundlage der Tageszeit und des Wochentags ausgeführt wird.

1. Registrieren Sie Ihre Amazon Keyspaces-Tabelle als skalierbares Ziel bei Application Auto Scaling. Ein skalierbares Ziel ist eine Ressource, die Application Auto Scaling auf- und abskalieren kann.

```
aws application-autoscaling register-scalable-target \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:WriteCapacityUnits \
```

```
    --resource-id keyspace/keyspace-name/table/table-name \
    --min-capacity 90 \
    --max-capacity 1500
```

2. Richten Sie geplante Aktionen entsprechend Ihren Anforderungen ein.

   Sie benötigen zwei Regeln, um das Szenario abzudecken: eine für die Hochskalierung und eine weitere für die Verkleinerung. Die erste Regel zum Hochskalieren der geplanten Aktion wird im folgenden Beispiel gezeigt.

   ```
   aws application-autoscaling put-scheduled-action \
       --service-namespace cassandra \
       --scalable-dimension cassandra:table:WriteCapacityUnits \
       --resource-id keyspace/keyspace-name/table/table-name \
       --scheduled-action-name my-8-5-scheduled-action \
       --scalable-target-action MinCapacity=800,MaxCapacity=1500 \
       --schedule "cron(45 8 ? * MON-FRI *)" \
       --timezone "Australia/Brisbane"
   ```

   In diesem Beispiel wird die zweite Regel zum Verkleinern der geplanten Aktion gezeigt.

   ```
   aws application-autoscaling put-scheduled-action \
       --service-namespace cassandra \
       --scalable-dimension cassandra:table:WriteCapacityUnits \
       --resource-id keyspace/keyspace-name/table/table-name \
       --scheduled-action-name my-5-8-scheduled-down-action \
       --scalable-target-action MinCapacity=90,MaxCapacity=1500 \
       --schedule "cron(15 17 ? * MON-FRI *)" \
       --timezone "Australia/Brisbane"
   ```

3. Führen Sie den folgenden Befehl aus, um zu bestätigen, dass beide Regeln aktiviert wurden:

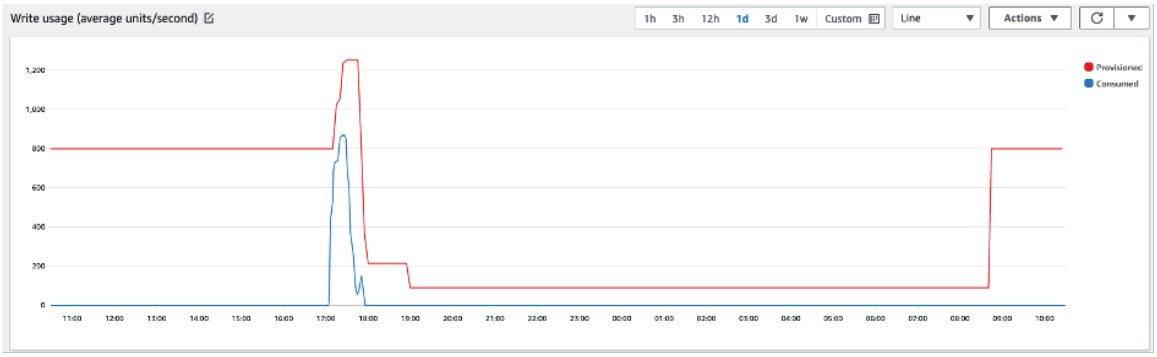
   ```
   aws application-autoscaling describe-scheduled-actions --service-namespace
    cassandra
   ```

   Das Ergebnis sollte ungefähr wie folgt aussehen:

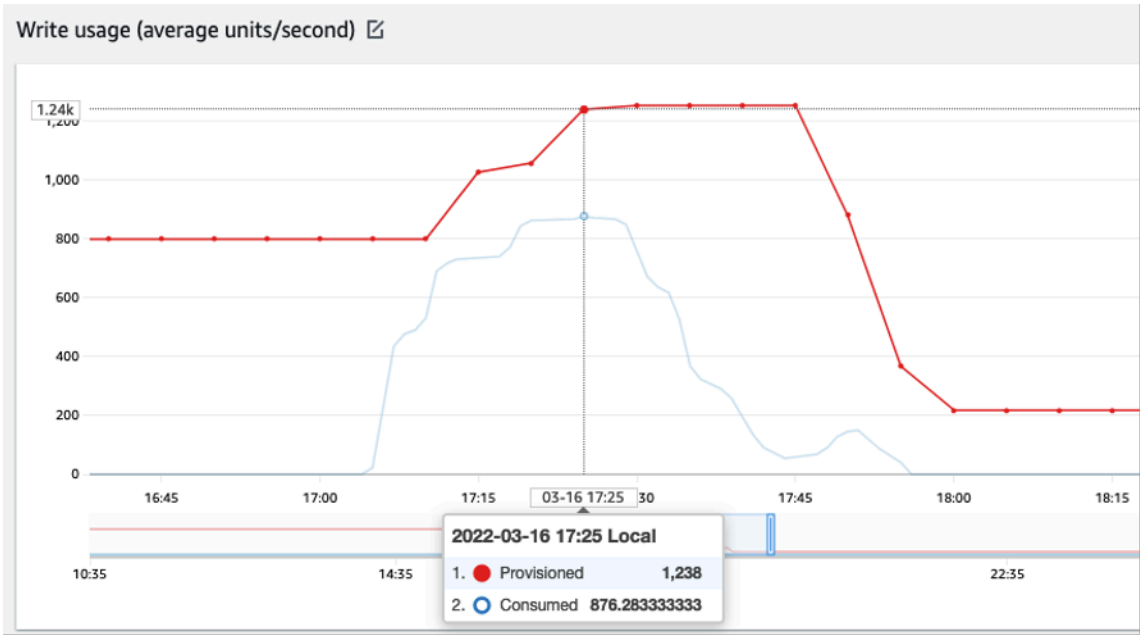
   ```
   {
       "ScheduledActions": [
           {
               "ScheduledActionName": "my-5-8-scheduled-down-action",
   ```

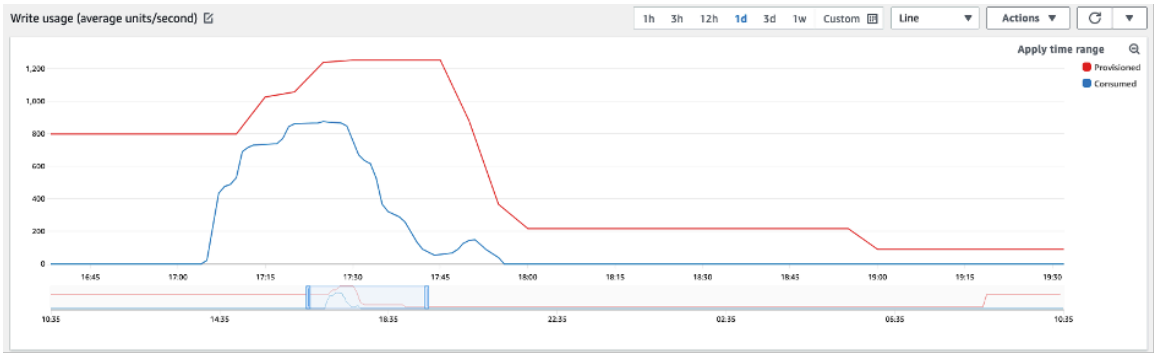
```
            "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:111122223333:scheduledAction:<uuid>:resource/keyspaces/table/table-
name:scheduledActionName/my-5-8-scheduled-down-action",
            "ServiceNamespace": "cassandra",
            "Schedule": "cron(15 17 ? * MON-FRI *)",
            "Timezone": "Australia/Brisbane",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:table:WriteCapacityUnits",
            "ScalableTargetAction": {
                "MinCapacity": 90,
                "MaxCapacity": 1500
            },
            "CreationTime": "2022-03-15T17:30:25.100000+10:00"
        },
        {
            "ScheduledActionName": "my-8-5-scheduled-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:111122223333:scheduledAction:<uuid>:resource/keyspaces/table/table-
name:scheduledActionName/my-8-5-scheduled-action",
            "ServiceNamespace": "cassandra",
            "Schedule": "cron(45 8 ? * MON-FRI *)",
            "Timezone": "Australia/Brisbane",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:table:WriteCapacityUnits",
            "ScalableTargetAction": {
                "MinCapacity": 800,
                "MaxCapacity": 1500
            },
            "CreationTime": "2022-03-15T17:28:57.816000+10:00"
        }
    ]
}
```

Im folgenden Bild ist ein Workload zu sehen, bei dem immer die Zielauslastung von 70 % beibehalten wird. Beachten Sie, dass die Auto Scaling-Regeln immer noch gelten und der Durchsatz nicht reduziert wird.



Wenn wir die Ansicht vergrößern, sehen wir, dass es in der Anwendung einen Spitzenwert gegeben hat, der den 70%-Schwellenwert für Auto Scaling ausgelöst hat, sodass Auto Scaling aktiviert und die für die Tabelle benötigte zusätzliche Kapazität bereitgestellt wurde. Die geplante Auto Scaling-Aktion wirkt sich auf Höchst- und Minimalwerte aus, und es liegt in Ihrer Verantwortung, sie einzurichten.





So bewältigen Sie stark schwankende Workloads mit unbekannten Mustern

In diesem Szenario verwendet die Anwendung ein sehr niedriges Auslastungsziel, da Sie die Anwendungsmuster noch nicht kennen und Sie sicherstellen möchten, dass es bei Ihrem Workload nicht zu Durchsatzfehlern mit niedriger Kapazität kommt.

Erwägen Sie stattdessen den On-Demand-Kapazitätsmodus. On-demand Tabellen eignen sich perfekt für stark beanspruchte Workloads, bei denen Sie die Datenverkehrsmuster nicht kennen. Im On-Demand-Kapazitätsmodus zahlen Sie pro Anforderung für die Lese- und Schreibvorgänge, die Ihre Anwendung in Ihren Tabellen ausführt. Sie müssen nicht angeben, wie viel Lese- und Schreibdurchsatz Sie von Ihrer Anwendung erwarten, da Amazon Keyspaces Ihre Workloads sofort berücksichtigt, wenn sie steigen oder fallen.

So bewältigen Sie Workloads mit verknüpften Anwendungen

In diesem Szenario hängt die Anwendung von anderen Systemen ab. Dies kann beispielsweise in Batchverarbeitungsszenarien der Fall sein, bei denen es abhängig von Ereignissen in der Anwendungslogik zu starken Datenverkehrsspitzen kommen kann.

Erwägen Sie die Entwicklung einer benutzerdefinierten Logik zur auto-scaling von Anwendungen, die auf Ereignisse reagiert, bei denen Sie die Tabellenkapazität erhöhen können `TargetValues` und die Ihren spezifischen Anforderungen entspricht. Sie könnten von einer Kombination von AWS Diensten wie Λ Amazon EventBridge und Step Functions profitieren und diese nutzen, um auf Ihre spezifischen Anwendungsanforderungen zu reagieren.

Identifizieren Sie Ihre ungenutzten Ressourcen, um die Kosten in Amazon Keyspaces zu optimieren

Dieser Abschnitt gibt einen Überblick darüber, wie Sie Ihre nicht verwendeten Ressourcen regelmäßig bewerten können. Wenn sich Ihre Anwendungsanforderungen weiterentwickeln, sollten Sie sicherstellen, dass keine Ressourcen ungenutzt bleiben und zu unnötigen Amazon Keyspaces-Kosten beitragen. Die unten beschriebenen Verfahren verwenden CloudWatch Amazon-Metriken, um ungenutzte Ressourcen zu identifizieren und Maßnahmen zur Kostensenkung zu ergreifen.

Sie können Amazon Keyspaces mithilfe von Amazon Keyspaces überwachen CloudWatch, das Rohdaten von Amazon Keyspaces sammelt und zu lesbaren, nahezu in Echtzeit verfügbaren Metriken verarbeitet. Diese Statistiken werden eine gewisse Zeit aufbewahrt, damit Sie zum besseren Verständnis Ihrer Nutzung Verlaufsdaten zur Verfügung haben. Standardmäßig werden Amazon Keyspaces-Metrikdaten CloudWatch automatisch an gesendet. Weitere Informationen finden Sie

unter Was ist Amazon CloudWatch? und Aufbewahrung von Kennzahlen im CloudWatch Amazon-Benutzerhandbuch.

Themen

- So ermitteln Sie nicht verwendete Ressourcen
- Ermitteln von nicht verwendeten Tabellenressourcen
- Bereinigen von nicht verwendeten Tabellenressourcen
- Bereinigen ungenutzter Point-in-Time Recovery (PITR) -Backups

So ermitteln Sie nicht verwendete Ressourcen

Um ungenutzte Tabellen zu identifizieren, können Sie sich die folgenden CloudWatch Kennzahlen über einen Zeitraum von 30 Tagen ansehen, um festzustellen, ob in einer bestimmten Tabelle aktive Lese- oder Schreibvorgänge vorhanden sind:

`ConsumedReadCapacityUnits`

Die Anzahl der in einem bestimmten Zeitraum verbrauchten Lesekapazitätseinheiten, um nachverfolgen zu können, wie viel Kapazität Sie genutzt haben. Sie können die gesamte verbrauchte Lesekapazität für eine Tabelle abrufen.

`ConsumedWriteCapacityUnits`

Die Anzahl der in einem bestimmten Zeitraum verbrauchten Schreibkapazitätseinheiten, um nachverfolgen zu können, wie viel Kapazität Sie genutzt haben. Sie können die gesamte verbrauchte Schreibkapazität für eine Tabelle abrufen.

Ermitteln von nicht verwendeten Tabellenressourcen

Amazon CloudWatch ist ein Überwachungs- und Beobachtbarkeitsservice, der die Amazon Keyspaces-Tabellenmetriken bereitstellt, anhand derer Sie ungenutzte Ressourcen identifizieren können. CloudWatch Metriken können sowohl über AWS-Managementkonsole als auch über die eingesehen werden. AWS Command Line Interface

AWS Command Line Interface

Um die Metriken Ihrer Tabellen über anzuzeigen AWS Command Line Interface, können Sie die folgenden Befehle verwenden.

1. Werten Sie zunächst die Lesevorgänge Ihrer Tabelle aus:

> **Note**
>
> Wenn der Tabellenname in Ihrem Konto nicht eindeutig ist, müssen Sie auch den Namen des Schlüsselraums angeben.

```
aws cloudwatch get-metric-statistics --metric-name
ConsumedReadCapacityUnits --start-time <start-time> --end-time <end-
time> --period <period> --namespace AWS/Cassandra --statistics Sum --
dimensions Name=TableName,Value=<table-name>
```

Damit Tabellen nicht fälschlicherweise als nicht verwendet ermittelt werden, sollten Sie die Metriken über einen längeren Zeitraum auswerten. Wählen Sie einen geeigneten Start- und Endzeitbereich, z. B. 30 Tage, und einen geeigneten Zeitraum, z. B. 86400.

In den zurückgegebenen Daten zeigt eine Summe von mehr als 0 an, dass die auszuwertende Tabelle während dieses Zeitraums Lesedatenverkehr empfangen hat.

Das folgende Ergebnis zeigt eine Tabelle, die im ausgewerteten Zeitraum Lesedatenverkehr empfangen hat:

```
        {
            "Timestamp": "2022-08-25T19:40:00Z",
            "Sum": 36023355.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-12T19:40:00Z",
            "Sum": 38025777.5,
            "Unit": "Count"
        },
```

Das folgende Ergebnis zeigt eine Tabelle, die im ausgewerteten Zeitraum keinen Lesedatenverkehr empfangen hat:

```
        {
            "Timestamp": "2022-08-01T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
```

```
        },
        {
            "Timestamp": "2022-08-20T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

2. Werten Sie als Nächstes die Schreibvorgänge Ihrer Tabelle aus:

```
aws cloudwatch get-metric-statistics --metric-name
ConsumedWriteCapacityUnits --start-time <start-time> --end-time <end-
time> --period <period> --namespace AWS/Cassandra --statistics Sum --
dimensions Name=TableName,Value=<table-name>
```

Damit Tabellen nicht fälschlicherweise als nicht verwendet ermittelt werden, sollten Sie die Metriken über einen längeren Zeitraum auswerten. Wählen Sie einen geeigneten Startzeit- und Endzeitbereich, beispielsweise 30 Tage, und einen geeigneten Zeitraum, wie z. B. 86400.

In den zurückgegebenen Daten zeigt eine Summe von mehr als 0 an, dass die auszuwertende Tabelle während dieses Zeitraums Lesedatenverkehr empfangen hat.

Das folgende Ergebnis zeigt eine Tabelle, die im ausgewerteten Zeitraum Schreibdatenverkehr empfangen hat:

```
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 41014457.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-18T20:15:00Z",
            "Sum": 40048531.0,
            "Unit": "Count"
        },
```

Das folgende Ergebnis zeigt eine Tabelle, die im ausgewerteten Zeitraum keinen Schreibdatenverkehr empfangen hat:

```
        {
            "Timestamp": "2022-07-31T20:15:00Z",
```

```
            "Sum": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

AWS-Managementkonsole

Mit den folgenden Schritten können Sie Ihre Ressourcennutzung anhand der bewerten. AWS-Managementkonsole

1. Melden Sie sich bei an AWS-Managementkonsole und navigieren Sie zur CloudWatch Serviceseite unter https://console.aws.amazon.com/cloudwatch/. Wählen Sie bei Bedarf oben rechts AWS-Region in der Konsole die entsprechende Option aus.
2. Suchen Sie in der linken Navigationsleiste den Abschnitt Metriken und wählen Sie Alle Metriken aus.
3. Die obige Aktion öffnet ein Dashboard mit zwei Bereichen. Im oberen Bereich sehen Sie die aktuellen Kennzahlen grafisch dargestellt. Unten können Sie die Metriken auswählen, die grafisch dargestellt werden können. Wählen Sie im unteren Bereich Amazon Keyspaces aus.
4. Wählen Sie im Auswahlfeld für Amazon Keyspaces-Metriken die Kategorie Tabellenmetriken aus, um die Metriken für Ihre Tabellen in der aktuellen Region anzuzeigen.
5. Identifizieren Sie Ihren Tabellennamen, indem Sie im Menü nach unten scrollen und dann die Metriken `ConsumedReadCapacityUnits` und `ConsumedWriteCapacityUnits` für Ihre Tabelle auswählen.
6. Wählen Sie den Tab Graphed Metrics (2) und passen Sie die Spalte Statistik auf Summe an.
7. Um zu vermeiden, dass eine Tabelle fälschlicherweise als unbenutzt identifiziert wird, werten Sie die Tabellenmetriken über einen längeren Zeitraum aus. Wählen Sie oben im Grafikfenster einen geeigneten Zeitraum, z. B. einen Monat, für die Auswertung Ihrer Tabelle aus. Wählen Sie Benutzerdefiniert, wählen Sie im Dropdownmenü die Option 1 Monat aus und wählen Sie Anwenden aus.
8. Werten Sie die grafisch dargestellten Metriken für Ihre Tabelle aus, um festzustellen, ob die Tabelle genutzt wird. Metriken, die über 0 hinausgehen, weisen darauf hin, dass eine Tabelle

während des ausgewerteten Zeitraums genutzt wurde. Ein flaches Diagramm bei 0 für Lese- und Schreibvorgänge zeigt an, dass eine Tabelle unbenutzt ist.

Bereinigen von nicht verwendeten Tabellenressourcen

Wenn Sie nicht verwendete Tabellenressourcen ermittelt haben, können Sie die laufenden Kosten für diese Ressourcen auf folgende Weise reduzieren.

> **Note**
>
> Wenn Sie eine nicht verwendete Tabelle ermittelt haben, die jedoch verfügbar bleiben soll, falls in Zukunft darauf zugegriffen werden muss, sollten Sie eine Umstellung auf den On-Demand-Modus in Betracht ziehen. Andernfalls können Sie erwägen, die Tabelle zu löschen.

Kapazitätsmodi

Amazon Keyspaces berechnet Gebühren für das Lesen, Schreiben und Speichern von Daten in Ihren Amazon Keyspaces-Tabellen.

Amazon Keyspaces bietet zwei Kapazitätsmodi mit spezifischen Abrechnungsoptionen für die Verarbeitung von Lese- und Schreibvorgängen in Ihren Tabellen: auf Abruf und bereitgestellt. Der read/write Kapazitätsmodus steuert, wie Ihnen der Lese- und Schreibdurchsatz in Rechnung gestellt wird und wie Sie die Kapazität verwalten.

Für On-Demand-Modustabellen müssen Sie nicht angeben, wie viel Lese- und Schreibdurchsatz Sie von Ihrer Anwendung erwarten. Amazon Keyspaces berechnet Ihnen die Lese- und Schreibvorgänge, die Ihre Anwendung an Ihren Tabellen durchführt, in Form von Leseanforderungseinheiten und Schreibanforderungseinheiten. Wenn auf Ihrer Tabelle keine Aktivität stattfindet, zahlen Sie nicht für den Durchsatz, aber es fallen trotzdem Speichergebühren an.

Löschen von Tabellen

Wenn Sie eine unbenutzte Tabelle entdeckt haben und diese löschen möchten, sollten Sie zunächst eine Sicherungskopie erstellen oder die Daten exportieren.

Durchgeführte Backups AWS Backup können Cold Storage Tiering nutzen und so die Kosten weiter senken. In der Dokumentation zur Verwaltung von Backup-Plänen finden Sie Informationen dazu, wie Sie einen Lebenszyklus verwenden können, um Ihr Backup in einen Cold Storage zu verschieben.

Nachdem Ihre Tabelle gesichert wurde, können Sie sie entweder über die AWS-Managementkonsole oder über die AWS Command Line Interface löschen.

Bereinigen ungenutzter Point-in-Time Recovery (PITR) -Backups

Amazon Keyspaces bietet Point-in-time Recovery, d. h. kontinuierliche Backups für 35 Tage, um Sie vor versehentlichen Schreib- oder Löschvorgängen zu schützen. Mit PITR-Backups sind Kosten verbunden.

In der Dokumentation unter finden Sie Informationen the section called “Backup und Wiederherstellung mit Point-in-Time-Recovery” darüber, ob für Ihre Tabellen Backups aktiviert sind, die möglicherweise nicht mehr benötigt werden.

Bewerten Sie Ihre Tabellennutzungsmuster, um Leistung und Kosten zu optimieren

Dieser Abschnitt bietet einen Überblick darüber, wie Sie beurteilen können, ob Sie Ihre Amazon Keyspaces-Tabellen effizient verwenden. Es gibt bestimmte Nutzungsmuster, die für Amazon Keyspaces nicht optimal sind, und sie bieten Raum für Optimierungen sowohl in Bezug auf die Leistung als auch in Bezug auf die Kosten.

Themen

- Ausführen von weniger strikt konsistenten Lesevorgängen
- Aktivieren von Time to Live (TTL)

Ausführen von weniger strikt konsistenten Lesevorgängen

Amazon Keyspaces ermöglicht es Ihnen, die Lesekonsistenz pro Anfrage zu konfigurieren. Leseanforderungen sind standardmäßig letztendlich konsistent. Eventuell werden konsistente Lesevorgänge mit 0,5 RCU für bis zu 4 KB Daten berechnet.

Die meisten Teile von verteilten Workloads sind flexibel und können letztendliche Konsistenz tolerieren. Es kann jedoch Zugriffsmuster geben, die strikt konsistente Lesevorgänge erfordern. Für stark konsistente Lesevorgänge werden 1 RCU für bis zu 4 KB Daten berechnet, wodurch sich Ihre Lesekosten im Wesentlichen verdoppeln. Amazon Keyspaces bietet Ihnen die Flexibilität, beide Konsistenzmodelle in derselben Tabelle zu verwenden.

Durch Auswertung Ihres Workloads und Anwendungscodes können Sie prüfen, ob strikt konsistente Lesevorgänge nur bei Bedarf verwendet werden.

Aktivieren von Time to Live (TTL)

Time to Live (TTL) hilft Ihnen dabei, Ihre Anwendungslogik zu vereinfachen und den Speicherpreis zu optimieren, indem Daten aus Tabellen automatisch ablaufen. Daten, die Sie nicht mehr benötigen, werden basierend auf dem von Ihnen festgelegten Time-to-Live-Wert automatisch aus Ihrer Tabelle gelöscht.

Auswerten Ihrer bereitgestellten Kapazität, um eine angemessene Bereitstellung zu erzielen

Dieser Abschnitt bietet einen Überblick darüber, wie Sie beurteilen können, ob Ihre Amazon Keyspaces-Tabellen über die richtige Größe für die Bereitstellung verfügen. Wenn sich Ihre Arbeitslast weiterentwickelt, sollten Sie Ihre Betriebsabläufe entsprechend ändern, insbesondere wenn Ihre Amazon Keyspaces-Tabelle im Bereitstellungsmodus konfiguriert ist und Sie das Risiko einer Über- oder Unterbereitstellung Ihrer Tabellen eingehen.

Die in diesem Abschnitt beschriebenen Verfahren erfordern statistische Informationen, die aus den Amazon Keyspaces-Tabellen erfasst werden sollten, die Ihre Produktionsanwendung unterstützen. Um Ihr Anwendungsverhalten zu verstehen, sollten Sie einen Zeitraum definieren, der signifikant genug ist, um die saisonale Datenabhängigkeit Ihrer Anwendung zu erfassen. Wenn Ihre Anwendung beispielsweise wöchentliche Muster aufweist, sollte ein Zeitraum von drei Wochen ausreichen, um die Anforderungen an den Anwendungsdurchsatz zu analysieren.

Wenn Sie nicht wissen, wo Sie anfangen sollen, verwenden Sie für die folgenden Berechnungen die Datennutzung von mindestens einem Monat.

Bei der Bewertung der Kapazität können Sie für Amazon Keyspaces-Tabellen Read Capacity Units (RCUs) und Write Capacity Units (WCU) unabhängig voneinander konfigurieren.

Themen

- So rufen Sie Verbrauchsmetriken aus Ihren Amazon Keyspaces-Tabellen ab
- So identifizieren Sie Amazon Keyspaces-Tabellen mit unzureichender Bereitstellung
- So identifizieren Sie übermäßig bereitgestellte Amazon Keyspaces-Tabellen

So rufen Sie Verbrauchsmetriken aus Ihren Amazon Keyspaces-Tabellen ab

Um die Tabellenkapazität zu bewerten, überwachen Sie die folgenden CloudWatch Kennzahlen und wählen Sie die entsprechende Dimension aus, um Tabelleninformationen abzurufen:

Lesekapazitätseinheiten	Schreibkapazitätseinheiten
`ConsumedReadCapacityUnits`	`ConsumedWriteCapacityUnits`
`ProvisionedReadCapacityUnits`	`ProvisionedWriteCapacityUnits`
`ReadThrottleEvents`	`WriteThrottleEvents`

Sie können dies entweder über die AWS CLI oder die tun AWS-Managementkonsole.

AWS CLI

Bevor Sie die Kennzahlen zum Tabellenverbrauch abrufen, müssen Sie zunächst einige historische Datenpunkte mithilfe der CloudWatch API erfassen.

Erstellen Sie zunächst zwei Dateien: `write-calc.json` und `read-calc.json`. Diese Dateien stellen die Berechnungen für die Tabelle dar. Sie müssen einige der Felder aktualisieren, wie in der Tabelle unten angegeben, damit sie zu Ihrer Umgebung passen.

> **Note**
>
> Wenn der Tabellenname in Ihrem Konto nicht eindeutig ist, müssen Sie auch den Namen des Schlüsselraums angeben.

Feldname	Definition	Beispiel
`<table-name>`	Der Name der Tabelle, die Sie analysieren	SampleTable
`<period>`	Der Zeitraum, den Sie zur Bewertung des Nutzungsziels verwenden, basierend auf Sekunden	Für einen Zeitraum von 1 Stunde müssen Sie Folgendes angeben: 3600
`<start-time>`	Der Beginn Ihres Auswertungsintervalls, im ISO8601-Format angegeben	21.02.2022 T23:00:00

Feldname	Definition	Beispiel
<end-time>	Das Ende Ihres Auswertungsintervalls, im ISO8601-Format angegeben	22.02.2022 T06:00:00

Die Datei mit Schreibberechnungen ruft die Anzahl der WCU ab, die im angegebenen Zeitraum für den angegebenen Zeitraum bereitgestellt und verbraucht wurden. Außerdem wird ein Nutzungsprozentsatz generiert, der für Analysen verwendet werden kann. Der vollständige Inhalt der `write-calc.json` Datei sollte wie im folgenden Beispiel aussehen.

```
{
  "MetricDataQueries": [
    {
      "Id": "provisionedWCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ProvisionedWriteCapacityUnits",
          "Dimensions": [
            {
              "Name": "TableName",
              "Value": "<table-name>"
            }
          ]
        },
        "Period": <period>,
        "Stat": "Average"
      },
      "Label": "Provisioned",
      "ReturnData": false
    },
    {
      "Id": "consumedWCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ConsumedWriteCapacityUnits",
          "Dimensions": [
            {
              "Name": "TableName",
```

```
              "Value": "<table-name>""
            }
          ]
        },
        "Period": <period>,
        "Stat": "Sum"
      },
      "Label": "",
      "ReturnData": false
    },
    {
      "Id": "m1",
      "Expression": "consumedWCU/PERIOD(consumedWCU)",
      "Label": "Consumed WCUs",
      "ReturnData": false
    },
    {
      "Id": "utilizationPercentage",
      "Expression": "100*(m1/provisionedWCU)",
      "Label": "Utilization Percentage",
      "ReturnData": true
    }
  ],
  "StartTime": "<start-time>",
  "EndTime": "<end-time>",
  "ScanBy": "TimestampDescending",
  "MaxDatapoints": 24
}
```

Die gelesene Berechnungsdatei verwendet ähnliche Metriken. Diese Datei ruft ab, wie viele RCUs während des angegebenen Zeitraums bereitgestellt und genutzt wurden. Der Inhalt der `read-calc.json` Datei sollte wie in diesem Beispiel aussehen.

```
{
  "MetricDataQueries": [
    {
      "Id": "provisionedRCU",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/Cassandra",
          "MetricName": "ProvisionedReadCapacityUnits",
          "Dimensions": [
            {
```

```
            "Name": "TableName",
            "Value": "<table-name>"
          }
        ]
      },
      "Period": <period>,
      "Stat": "Average"
    },
    "Label": "Provisioned",
    "ReturnData": false
  },
  {
    "Id": "consumedRCU",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/Cassandra",
        "MetricName": "ConsumedReadCapacityUnits",
        "Dimensions": [
          {
            "Name": "TableName",
            "Value": "<table-name>"
          }
        ]
      },
      "Period": <period>,
      "Stat": "Sum"
    },
    "Label": "",
    "ReturnData": false
  },
  {
    "Id": "m1",
    "Expression": "consumedRCU/PERIOD(consumedRCU)",
    "Label": "Consumed RCUs",
    "ReturnData": false
  },
  {
    "Id": "utilizationPercentage",
    "Expression": "100*(m1/provisionedRCU)",
    "Label": "Utilization Percentage",
    "ReturnData": true
  }
],
"StartTime": "<start-time>",
```

```
  "EndTime": "<end-time>",
  "ScanBy": "TimestampDescending",
  "MaxDatapoints": 24
}
```

Sobald Sie die Dateien erstellt haben, können Sie mit dem Abrufen der Nutzungsdaten beginnen.

1. Geben Sie den folgenden Befehl ein, um die Daten zur Schreibauslastung abzurufen:

   ```
   aws cloudwatch get-metric-data --cli-input-json file://write-calc.json
   ```

2. Geben Sie den folgenden Befehl ein, um die Lesenutzungsdaten abzurufen:

   ```
   aws cloudwatch get-metric-data --cli-input-json file://read-calc.json
   ```

Das Ergebnis für beide Abfragen ist eine Reihe von Datenpunkten im JSON-Format, die für Analysen verwendet werden können. Ihre Ergebnisse hängen von der Anzahl der von Ihnen angegebenen Datenpunkte, dem Zeitraum und Ihren eigenen spezifischen Workload-Daten ab. Es könnte wie im folgenden Beispiel aussehen.

```
{
    "MetricDataResults": [
        {
            "Id": "utilizationPercentage",
            "Label": "Utilization Percentage",
            "Timestamps": [
                "2022-02-22T05:00:00+00:00",
                "2022-02-22T04:00:00+00:00",
                "2022-02-22T03:00:00+00:00",
                "2022-02-22T02:00:00+00:00",
                "2022-02-22T01:00:00+00:00",
                "2022-02-22T00:00:00+00:00",
                "2022-02-21T23:00:00+00:00"
            ],
            "Values": [
                91.55364583333333,
                55.066631944444445,
                2.6114930555555556,
                24.9496875,
                40.94725694444445,
                25.61819444444444,
```

```
                0.0
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

> **Note**
>
> Wenn Sie einen kurzen Zeitraum und einen langen Zeitraum angeben, müssen Sie möglicherweise den `MaxDatapoints` Wert ändern, der im Skript standardmäßig auf 24 gesetzt ist. Dies entspricht einem Datenpunkt pro Stunde und 24 pro Tag.

AWS-Managementkonsole

1. Melden Sie sich bei an AWS-Managementkonsole und navigieren Sie zur CloudWatch Serviceseite unter Erste Schritte mit dem AWS-Managementkonsole. Wählen Sie bei AWS-Region Bedarf das entsprechende aus.
2. Suchen Sie in der linken Navigationsleiste den Abschnitt Metriken und wählen Sie Alle Metriken aus.
3. Dadurch wird ein Dashboard mit zwei Bereichen geöffnet. Im oberen Bereich wird die Grafik angezeigt, und im unteren Bereich befinden sich die Kennzahlen, die Sie grafisch darstellen möchten. Wählen Sie das Amazon Keyspaces-Bedienfeld.
4. Wählen Sie in den Unterfenstern die Kategorie „Tabellenmetriken“ aus. Dies zeigt Ihnen die Tabellen in Ihrer aktuellen Version AWS-Region.
5. Ermitteln Sie Ihren Tabellennamen, indem Sie im Menü nach unten scrollen und die Metriken für Schreibvorgänge auswählen: `ConsumedWriteCapacityUnits` und `ProvisionedWriteCapacityUnits`.

   > **Note**
   >
   > In diesem Beispiel geht es um Metriken für Schreibvorgänge, Sie können diese Schritte jedoch auch verwenden, um die Metriken für Lesevorgänge grafisch darzustellen.

6. Wählen Sie die Registerkarte Graphed metrics (2) (Grafisch dargestellte Metriken (2)) aus, um die Formeln zu ändern. CloudWatch Wählt standardmäßig die statistische Funktion Durchschnitt für die Grafiken.
7. Wenn beide grafisch dargestellten Metriken ausgewählt sind (Kontrollkästchen auf der linken Seite), wählen Sie das Menü Add math (Math. hinzufügen), dann Common (Allgemein) und schließlich die Funktion Percentage (Prozent) aus. Wiederholen Sie den Vorgang zweimal.

   Wählen Sie zum ersten Mal die Prozentfunktion aus.

   Wählen Sie zum zweiten Mal die Funktion Percentage aus.
8. Zu diesem Zeitpunkt sollten vier Metriken im unteren Menü aufgeführt sein. Befassen wir uns nun mit der Berechnung von `ConsumedWriteCapacityUnits`. Um konsistent zu sein, müssen Sie die Namen mit denen abgleichen, die Sie in dem AWS CLI Abschnitt verwendet haben. Klicken Sie auf die m1 ID und ändern Sie diesen Wert in consumedWCU.
9. Ändern Sie die Statistik von Average (Durchschnitt) in Sum (Summe). Diese Aktion erstellt automatisch eine weitere Metrik namens ANOMALY_DETECTION_BAND. Im Rahmen dieses Verfahrens können Sie dies ignorieren, indem Sie das Kontrollkästchen für die neu generierte ad1-Metrik entfernen.
10. Wiederholen Sie Schritt 8, um die m2 ID in provisionedWCU umzubenennen. Lassen Sie die Statistik auf Average (Durchschnitt) eingestellt.
11. Wählen Sie das Label Expression1 und aktualisieren Sie den Wert auf m1 und das Label auf Consumed WCUs.

    > **Note**
    >
    > Stellen Sie sicher, dass Sie nur m1 (Kontrollkästchen links) und provisionedWCU ausgewählt haben, damit die Daten richtig angezeigt werden. Aktualisieren Sie die Formel, indem Sie auf Details klicken und die Formel in consumedWCU/PERIOD(ConsumedWCU) ändern. Durch diesen Schritt wird möglicherweise auch eine weitere ANOMALY_DETECTION_BAND-Metrik generiert, die Sie im Rahmen dieses Verfahrens jedoch ignorieren können.
12. Sie sollten jetzt über zwei Grafiken verfügen: eine, die Ihre bereitgestellten WCUs in der Tabelle anzeigt, und eine andere, die die verbrauchten WCUs anzeigt.

13. Aktualisieren Sie die Prozentformel, indem Sie die Grafik für Expression2 (e2) auswählen. Benennen Sie die Labels und IDs in utilizationPercentage um. Benennen Sie die Formel so um, dass sie 100* () entspricht. m1/provisionedWCU
14. Entfernen Sie das Kontrollkästchen aus allen Metriken mit Ausnahme von UtilizationPercentage, um Ihre Nutzungsmuster zu visualisieren. Das Standardintervall ist auf 1 Minute festgelegt, Sie können es jedoch nach Bedarf ändern.

Die Ergebnisse, die Sie erhalten, hängen von den tatsächlichen Daten Ihres Workloads ab. Intervalle mit einer Auslastung von mehr als 100% sind anfällig für Fehler bei geringer Durchsatzkapazität. Amazon Keyspaces bietet Burst-Kapazität, aber sobald die Burst-Kapazität erschöpft ist, kommt es bei einem Wert über 100% zu Fehlern bei niedriger Durchsatzkapazität.

So identifizieren Sie Amazon Keyspaces-Tabellen mit unzureichender Bereitstellung

Bei den meisten Workloads gilt eine Tabelle als unzureichend bereitgestellt, wenn sie ständig mehr als 80% der bereitgestellten Kapazität beansprucht.

Burst-Kapazität ist eine Funktion von Amazon Keyspaces, mit der Kunden vorübergehend mehr RCUs/WCUs als ursprünglich bereitgestellt nutzen können (mehr als den bereitgestellten Durchsatz pro Sekunde, der für die Tabelle definiert wurde). Die Burst-Kapazität soll plötzliche Zunahmen des Datenverkehrs aufgrund von besonderen Ereignissen oder Auslastungsspitzen auffangen. Diese Burst-Kapazität ist begrenzt. Weitere Informationen finden Sie unter. the section called “Verwenden Sie Burst-Kapazität” Sobald die ungenutzten RCUs und WCUs aufgebraucht sind, kann es zu Fehlern beim Durchsatz mit niedriger Kapazität kommen, wenn Sie versuchen, mehr Kapazität als vorgesehen zu verbrauchen. Wenn sich Ihr Anwendungsdatenverkehr der Auslastungsrate von 80% nähert, ist das Risiko, dass Fehler bei geringem Kapazitätsdurchsatz auftreten, erheblich höher.

Die Regel der Auslastungsrate von 80 % hängt von der Saisonalität Ihrer Daten und dem Wachstums Ihres Datenverkehrs ab. Betrachten Sie folgende Szenarien:

- Wenn Ihr Datenverkehr in den letzten 12 Monaten bei einer Auslastung von ~ 90 % stabil war, hat Ihre Tabelle genau die richtige Kapazität.
- Wenn Ihr Anwendungsdatenverkehr monatlich um 8 % wächst, werden Sie in weniger als 3 Monaten 100 % erreichen.
- Auch wenn Ihr Anwendungsdatenverkehr monatlich um 5 % wächst, werden Sie in etwas mehr als 4 Monaten 100 % erreichen.

Die Ergebnisse der obigen Abfragen vermitteln ein Bild Ihrer Auslastungsrate. Verwenden Sie sie als Orientierungshilfe für die Auswertung weiterer Metriken, die bei der Entscheidung, Ihre Tabellenkapazität nach Bedarf zu erhöhen, hilfreich sein können (z. B. monatliche oder wöchentliche Wachstumsrate). Legen Sie gemeinsam mit Ihrem Operations-Team fest, welcher Prozentsatz für Ihren Workload und Ihre Tabellen geeignet ist.

Es gibt spezielle Szenarien, in denen die Daten verzerrt sind, wenn Sie sie täglich oder wöchentlich analysieren. Beispielsweise könnten Sie bei saisonalen Anwendungen, die während der Arbeitszeit stark ausgelastet sind (aber außerhalb der Geschäftszeiten auf nahezu Null sinken), von der Planung der auto-scaling der Anwendung profitieren, bei der Sie die Tageszeiten (und Wochentage) angeben, um die bereitgestellte Kapazität zu erhöhen, sowie den Zeitpunkt, zu dem sie reduziert werden soll. Anstatt eine höhere Kapazität anzustreben, um die geschäftigen Zeiten abzudecken, können Sie auch von den auto-scaling Skalierungskonfigurationen der Amazon Keyspaces-Tabellen profitieren, wenn Ihre Saisonalität weniger ausgeprägt ist.

So identifizieren Sie übermäßig bereitgestellte Amazon Keyspaces-Tabellen

Die mit den obigen Skripts erhaltenen Abfrageergebnisse liefern die für eine erste Analyse erforderlichen Datenpunkte. Wenn Ihr Datensatz für mehrere Intervalle Auslastungswerte von weniger als 20 % aufweist, wurde für Ihre Tabelle möglicherweise zu viel Kapazität bereitgestellt. Um genauer herauszufinden, ob Sie die Anzahl der WCUs und RCUs reduzieren müssen, sollten Sie die anderen Messwerte in den Intervallen erneut überprüfen.

Wenn Ihre Tabelle mehrere niedrige Nutzungsintervalle enthält, können Sie von der Verwendung von Application Auto Scaling-Richtlinien profitieren, indem Sie entweder Application Auto Scaling planen oder einfach die standardmäßigen Application Auto Scaling Scaling-Richtlinien für die Tabelle konfigurieren, die auf der Auslastung basieren.

Wenn Sie eine Arbeitslast mit einem niedrigen Verhältnis zwischen Auslastung und hoher Drosselung haben (Max (ThrottleEventsThrottleEvents) /Min () im Intervall), kann dies passieren, wenn Sie eine sehr hohe Arbeitslast haben, bei der der Verkehr an bestimmten Tagen (oder Tageszeiten) erheblich zunimmt, ansonsten aber konstant niedrig ist. In diesen Szenarien kann es von Vorteil sein, geplantes Application Auto Scaling zu verwenden.

Fehlerbehebung bei Amazon Keyspaces (für Apache Cassandra)

Dieses Handbuch behandelt Schritte zur Fehlerbehebung für verschiedene Szenarien bei der Arbeit mit Amazon Keyspaces (für Apache Cassandra). Es enthält Informationen zur Behebung allgemeiner Fehler, Verbindungsprobleme, Kapazitätsverwaltungsprobleme und DDL-Fehler (Data Definition Language).

- Allgemeine Fehler
  - Behebung von Ausnahmen auf oberster Ebene wie `NoNodeAvailableExceptionNoHostAvailableException`, und`AllNodesFailedException`.
  - Isolierung der zugrunde liegenden Fehler von Java-Treiberausnahmen.
  - Implementieren von Wiederholungsrichtlinien und korrekte Konfiguration von Verbindungen.
- Probleme mit der Verbindung
  - Behebung von Fehlern beim Herstellen einer Verbindung zu Amazon Keyspaces-Endpunkten mithilfe von Apache `cqlsh` Cassandra-Clienttreibern.
  - Behebung von VPC-Endpunktverbindungen, Cassandra-stress Verbindungen und IAM-Konfigurationsfehlern.
  - Umgang mit Verbindungsverlusten bei Datenimporten.
- Fehler bei der Kapazitätsverwaltung
  - Erkennen und Beheben unzureichender Kapazitätsfehler im Zusammenhang mit Tabellen, Partitionen und Verbindungen.
  - Überwachung relevanter Amazon Keyspaces-Metriken in Amazon CloudWatch Logs.
  - Optimierung von Verbindungen und Durchsatz für eine verbesserte Leistung.
- DDL-Fehler (Data Definition Language)
  - Behebung von Fehlern beim Erstellen, Zugreifen auf oder Wiederherstellen von Schlüsselräumen und Tabellen.
  - Behandlung von Fehlern im Zusammenhang mit benutzerdefinierten TTL-Einstellungen (Time to Live), Spaltenlimits und Bereichslöschungen.
  - Überlegungen zu umfangreichen Löscharbeitslasten.

Anleitungen zur Fehlerbehebung speziell für den IAM-Zugriff finden Sie unter. the section called “Fehlerbehebung” Weitere Informationen zu bewährten Sicherheitsmethoden finden Sie unterthe section called “Bewährte Methoden für die Gewährleistung der Sicherheit”.

Topics

- Behebung allgemeiner Fehler in Amazon Keyspaces
- Behebung von Verbindungsfehlern in Amazon Keyspaces
- Behebung von Kapazitätsverwaltungsfehlern in Amazon Keyspaces
- Behebung von Sprachfehlern bei der Datendefinition in Amazon Keyspaces

Behebung allgemeiner Fehler in Amazon Keyspaces

Erhalten Sie allgemeine Fehler? Im Folgenden finden Sie einige häufig auftretende Probleme und wie Sie sie lösen können.

Allgemeine Fehler

Sie erhalten eine der folgenden Ausnahmen auf oberster Ebene, die aus vielen verschiedenen Gründen auftreten können.

- `NoNodeAvailableException`
- `NoHostAvailableException`
- `AllNodesFailedException`

Diese Ausnahmen werden vom Client-Treiber generiert und können entweder auftreten, wenn Sie die Kontrollverbindung herstellen oder wenn Sie read/write/prepare/execute/batch-Anfragen ausführen.

Wenn der Fehler beim Herstellen der Kontrollverbindung auftritt, ist dies ein Zeichen dafür, dass alle in Ihrer Anwendung angegebenen Kontaktpunkte nicht erreichbar sind. Wenn der Fehler bei der Ausführung von read/write/prepare/execute -Abfragen auftritt, bedeutet dies, dass alle Wiederholungsversuche für diese Anfrage erschöpft sind. Jeder Wiederholungsversuch wird auf einem anderen Knoten versucht, wenn Sie die Standard-Wiederholungsrichtlinie verwenden.

Wie isoliert man den zugrundeliegenden Fehler von Ausnahmen für Java-Treiber der obersten Ebene

Diese allgemeinen Fehler können entweder durch Verbindungsprobleme oder durch die Ausführung von read/write prepare/execute /-Vorgängen verursacht werden. In verteilten Systemen muss mit vorübergehenden Ausfällen gerechnet werden, und diese sollten behoben werden, indem die Anfrage erneut versucht wird. Der Java-Treiber versucht es nicht automatisch erneut, wenn Verbindungsfehler auftreten. Es wird daher empfohlen, die Wiederholungsrichtlinie zu implementieren, wenn Sie die Treiberverbindung in Ihrer Anwendung herstellen. Eine ausführliche Übersicht über bewährte Verbindungsmethoden finden Sie unter. the section called “Verbindungen”

Standardmäßig ist der Java-Treiber für alle Anfragen `idempotence` auf False gesetzt, was bedeutet, dass der Java-Treiber die fehlgeschlagene read/write /prepare-Anfrage nicht automatisch wiederholt. Um die Einstellung `idempotence` auf fehlgeschlagene Anfragen einzustellen `true` und den Treiber anzuweisen, es erneut zu versuchen, können Sie dies auf verschiedene Arten tun. Hier ist ein Beispiel, wie Sie Idempotenz programmgesteuert für eine einzelne Anfrage in Ihrer Java-Anwendung festlegen können.

```
Statement s = new SimpleStatement("SELECT * FROM my_table WHERE id = 1");
s.setIdempotent(true);
```

Oder Sie können die Standard-Idempotenz für Ihre gesamte Java-Anwendung programmgesteuert festlegen, wie im folgenden Beispiel gezeigt.

```
// Make all statements idempotent by default:
cluster.getConfiguration().getQueryOptions().setDefaultIdempotence(true);
//Set the default idempotency to true in your Cassandra configuration
basic.request.default-idempotence = true
```

Eine weitere Empfehlung besteht darin, eine Wiederholungsrichtlinie auf Anwendungsebene zu erstellen. In diesem Fall muss die Anwendung die Anfrage catch `NoNodeAvailableException` und erneut versuchen. Wir empfehlen 10 Wiederholungen mit exponentiellem Backoff, beginnend bei 10 ms und bis zu 100 ms mit einer Gesamtzeit von 1 Sekunde für alle Wiederholungen.

Eine weitere Option besteht darin, die Amazon Keyspaces-Richtlinie für exponentielle Wiederholungen anzuwenden, wenn die auf Github verfügbare Java-Treiberverbindung hergestellt wird.

Vergewissern Sie sich, dass Sie Verbindungen zu mehr als einem Knoten hergestellt haben, wenn Sie die standardmäßige Wiederholungsrichtlinie verwenden. Sie können dies mit der folgenden Abfrage in Amazon Keyspaces tun.

```
SELECT * FROM system.peers;
```

Wenn die Antwort für diese Abfrage leer ist, bedeutet dies, dass Sie mit einem einzigen Knoten für Amazon Keyspaces arbeiten. Wenn Sie die standardmäßige Wiederholungsrichtlinie verwenden, gibt es keine Wiederholungsversuche, da die Standardwiederholung immer auf einem anderen Knoten erfolgt. Weitere Informationen zum Herstellen von Verbindungen über VPC-Endpunkte finden Sie unter. the section called “VPC-Endpunktverbindungen”

Eine schrittweise Anleitung, die zeigt, wie Sie mithilfe des Datastax 4.x Cassandra-Treibers eine Verbindung zu Amazon Keyspaces herstellen, finden Sie unter. the section called “Authentifizierungs-Plugin für Java 4.x”

Behebung von Verbindungsfehlern in Amazon Keyspaces

Haben Sie Probleme beim Herstellen einer Verbindung? Hier finden Sie einige häufig auftretende Probleme und wie Sie sie lösen können.

Fehler beim Herstellen einer Verbindung zu einem Amazon Keyspaces-Endpunkt

Fehlgeschlagene Verbindungen und Verbindungsfehler können zu unterschiedlichen Fehlermeldungen führen. Der folgende Abschnitt behandelt die häufigsten Szenarien.

Themen

- Ich kann mit cqlsh keine Verbindung zu Amazon Keyspaces herstellen
- Ich kann mit einem Cassandra-Client-Treiber keine Verbindung zu Amazon Keyspaces herstellen

Ich kann mit cqlsh keine Verbindung zu Amazon Keyspaces herstellen

Sie versuchen, mit cqlsh eine Verbindung zu einem Amazon Keyspaces-Endpunkt herzustellen, und die Verbindung schlägt mit a fehl. **`Connection error`**

Wenn Sie versuchen, eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen und cqlsh nicht richtig konfiguriert wurde, schlägt die Verbindung fehl. Der folgende Abschnitt enthält Beispiele

für die häufigsten Konfigurationsprobleme, die zu Verbindungsfehlern führen, wenn Sie versuchen, eine Verbindung mit cqlsh herzustellen.

> **Note**
>
> Wenn Sie versuchen, von einer VPC aus eine Verbindung zu Amazon Keyspaces herzustellen, sind zusätzliche Berechtigungen erforderlich. Um eine Verbindung mithilfe von VPC-Endpunkten erfolgreich zu konfigurieren, folgen Sie den Schritten in. the section called “Verbindung mit VPC-Endpunkten herstellen”

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber einen Verbindungsfehler. **`timed out`**

Dies kann der Fall sein, wenn Sie nicht den richtigen Port angegeben haben, was zu dem folgenden Fehler führt.

```
#  cqlsh cassandra.us-east-1.amazonaws.com 9140 -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.199': error(None,
 "Tried connecting to [('3.234.248.199', 9140)]. Last error: timed out")})
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Sie Port 9142 für die Verbindung verwenden.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`Name or service not known`**

Dies kann der Fall sein, wenn Sie einen Endpunkt verwendet haben, der falsch geschrieben ist oder nicht existiert. Im folgenden Beispiel ist der Name des Endpunkts falsch geschrieben.

```
#  cqlsh cassandra.us-east-1.amazon.com 9142 -u "USERNAME" -p "PASSWORD" --ssl
Traceback (most recent call last):
  File "/usr/bin/cqlsh.py", line 2458, in >module>
    main(*read_options(sys.argv[1:], os.environ))
  File "/usr/bin/cqlsh.py", line 2436, in main
    encoding=options.encoding)
  File "/usr/bin/cqlsh.py", line 484, in __init__
    load_balancing_policy=WhiteListRoundRobinPolicy([self.hostname]),
  File "/usr/share/cassandra/lib/cassandra-driver-internal-only-3.11.0-bb96859b.zip/
cassandra-driver-3.11.0-bb96859b/cassandra/policies.py", line 417, in __init__
socket.gaierror: [Errno -2] Name or service not known
```

Um dieses Problem zu beheben, wenn Sie öffentliche Endpunkte für die Verbindung verwenden, wählen Sie einen verfügbaren Endpunkt aus und stellen Sie sicherthe section called “Service-Endpunkte”, dass der Name des Endpunkts keine Fehler enthält. Wenn Sie VPC-Endpunkte für die Verbindung verwenden, stellen Sie sicher, dass die VPC-Endpunktinformationen in Ihrer cqlsh-Konfiguration korrekt sind.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`OperationTimedOut`**

Amazon Keyspaces setzt voraus, dass SSL für Verbindungen aktiviert ist, um eine hohe Sicherheit zu gewährleisten. Der SSL-Parameter fehlt möglicherweise, wenn Sie die folgende Fehlermeldung erhalten.

```
# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD"
Connection error: ('Unable to connect to any servers', {'3.234.248.192':
 OperationTimedOut('errors=Timed out creating connection (5 seconds),
 last_host=None',)})
#
```

Um dieses Problem zu beheben, fügen Sie dem Befehl cqlsh connection das folgende Flag hinzu.

```
--ssl
```

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, und erhalten eine Fehlermeldung. **`SSL transport factory requires a valid certfile to be specified`**

In diesem Fall fehlt der Pfad zum SSL/TLS Zertifikat, was zu dem folgenden Fehler führt.

```
# cat .cassandra/cqlshrc
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory
#


# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Validation is enabled; SSL transport factory requires a valid certfile to be specified.
 Please provide path to the certfile in [ssl] section as 'certfile' option in /
root/.cassandra/cqlshrc (or use [certfiles] section) or set SSL_CERTFILE environment
 variable.
```

```
#
```

Um dieses Problem zu beheben, fügen Sie den Pfad zur Zertifikatsdatei auf Ihrem Computer hinzu. Weitere Informationen finden Sie unter the section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”.

```
certfile =  path_to_file/keyspaces-bundle.pem
```

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`No such file or directory`**

Dies kann der Fall sein, wenn der Pfad zur Zertifikatsdatei auf Ihrem Computer falsch ist, was zu dem folgenden Fehler führt.

```
# cat .cassandra/cqlshrc
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory

[ssl]
validate = true
certfile = /root/wrong_path/keyspaces-bundle.pem
#



# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.192': IOError(2, 'No
 such file or directory')})
#
```

Um dieses Problem zu beheben, stellen Sie sicher, dass der Pfad zur Zertifikatsdatei auf Ihrem Computer korrekt ist. Weitere Informationen finden Sie unter the section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`[X509] PEM lib`**

Dies kann der Fall sein, wenn die SSL/TLS `pem` Zertifikatsdatei nicht gültig ist, was zu dem folgenden Fehler führt.

```
# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.241':
 error(185090057, u"Tried connecting to [('3.234.248.241', 9142)]. Last error: [X509]
 PEM lib (_ssl.c:3063)")})
#
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Sie die erforderlichen digitalen Zertifikate heruntergeladen haben. Weitere Informationen finden Sie unter the section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber einen **unknown** SSL-Fehler.

Dies kann der Fall sein, wenn die SSL/TLS `pem` Zertifikatsdatei leer ist, was zu dem folgenden Fehler führt.

```
# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.220': error(0,
 u"Tried connecting to [('3.234.248.220', 9142)]. Last error: unknown error
 (_ssl.c:3063)")})
#
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Sie die erforderlichen digitalen Zertifikate heruntergeladen haben. Sie können dies anhand der Schritte im folgenden Thema überprüfenthe section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`SSL: CERTIFICATE_VERIFY_FAILED`**

Dies kann der Fall sein, wenn die SSL/TLS Zertifikatsdatei nicht verifiziert werden konnte, was zu dem folgenden Fehler führt.

```
Connection error: ('Unable to connect to any servers', {'3.234.248.223':
 error(1, u"Tried connecting to [('3.234.248.223', 9142)]. Last error: [SSL:
 CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:727)")})
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Sie die erforderlichen digitalen Zertifikate heruntergeladen haben. Sie können dies anhand der Schritte im folgenden Thema überprüfenthe section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell”.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`Last error: timed out`**

Dies kann der Fall sein, wenn Sie in Ihrer Amazon EC2-Sicherheitsgruppe keine ausgehende Regel für Amazon Keyspaces konfiguriert haben, was zu dem folgenden Fehler führt.

```
# cqlsh cassandra.us-east-1.amazonaws.com 9142 -u  "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.206': error(None,
 "Tried connecting to [('3.234.248.206', 9142)]. Last error: timed out")})
#
```

Um zu überprüfen, ob dieses Problem durch die Konfiguration der Amazon EC2 EC2-Instance verursacht wird und nicht`cqlsh`, können Sie versuchen, eine Verbindung zu Ihrem Keyspace herzustellen AWS CLI, indem Sie beispielsweise den folgenden Befehl verwenden.

```
aws keyspaces list-tables --keyspace-name 'my_keyspace'
```

Wenn auch bei diesem Befehl ein Timeout auftritt, ist die Amazon EC2 EC2-Instance nicht korrekt konfiguriert.

Um zu bestätigen, dass Sie über ausreichende Berechtigungen für den Zugriff auf Amazon Keyspaces verfügen, können Sie die verwenden, AWS CloudShell um eine Verbindung `cqlsh` herzustellen. Wenn diese Verbindungen hergestellt werden, müssen Sie die Amazon EC2 EC2-Instance konfigurieren.

Um dieses Problem zu beheben, stellen Sie sicher, dass Ihre Amazon EC2 EC2-Instance über eine ausgehende Regel verfügt, die Datenverkehr zu Amazon Keyspaces zulässt. Ist dies nicht der Fall, müssen Sie eine neue Sicherheitsgruppe für die EC2-Instance erstellen und eine Regel hinzufügen, die ausgehenden Datenverkehr zu Amazon Keyspaces-Ressourcen zulässt. Um die Regel für ausgehenden Datenverkehr zu aktualisieren, sodass Datenverkehr zu Amazon Keyspaces zugelassen wird, wählen Sie CQLSH/CASSANDRAaus dem Drop-down-Menü Typ eine Option aus.

Nachdem Sie die neue Sicherheitsgruppe mit der Regel für ausgehenden Datenverkehr erstellt haben, müssen Sie sie der Instance hinzufügen. Wählen Sie die Instance aus und wählen Sie dann Aktionen, Sicherheit und dann Sicherheitsgruppen ändern aus. Fügen Sie die neue Sicherheitsgruppe mit der Regel für ausgehenden Datenverkehr hinzu, stellen Sie jedoch sicher, dass die Standardgruppe ebenfalls verfügbar bleibt.

Weitere Informationen zum Anzeigen und Bearbeiten von EC2-Regeln für ausgehenden Datenverkehr finden Sie unter Regeln zu einer Sicherheitsgruppe hinzufügen im Amazon EC2 EC2-Benutzerhandbuch.

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`Unauthorized`**

Dies kann der Fall sein, wenn Ihnen Amazon Keyspaces-Berechtigungen in der IAM-Benutzerrichtlinie fehlen, was zu dem folgenden Fehler führt.

```
# cqlsh cassandra.us-east-1.amazonaws.com 9142 -u  "testuser-at-12345678910" -p
 "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.241':
 AuthenticationFailed('Failed to authenticate to 3.234.248.241: Error from server:
 code=2100 [Unauthorized] message="User arn:aws:iam::12345678910:user/testuser has no
 permissions."',)})
#
```

Um dieses Problem zu beheben, stellen Sie sicher, dass der IAM-Benutzer `testuser-at-12345678910` über Zugriffsberechtigungen für Amazon Keyspaces verfügt. Beispiele für IAM-Richtlinien, die Zugriff auf Amazon Keyspaces gewähren, finden Sie unter. the section called “Identity-based Beispiele für Richtlinien”

Anleitungen zur Fehlerbehebung, die sich speziell auf den IAM-Zugriff beziehen, finden Sie unter. the section called “Fehlerbehebung”

Sie versuchen, mit cqlsh eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine Fehlermeldung. **`Bad credentials`**

Dies kann der Fall sein, wenn der Benutzername oder das Passwort falsch ist, was zu dem folgenden Fehler führt.

```
# cqlsh cassandra.us-east-1.amazonaws.com 9142 -u  "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.248':
 AuthenticationFailed('Failed to authenticate to 3.234.248.248: Error from server:
 code=0100 [Bad credentials] message="Provided username USERNAME and/or password are
 incorrect"',)})
#
```

Um dieses Problem zu beheben, stellen Sie sicher, dass die *USERNAME* Zeichen und *PASSWORD* in Ihrem Code mit dem Benutzernamen und dem Passwort übereinstimmen, die Sie bei der Generierung der dienstspezifischen Anmeldeinformationen erhalten haben.

> **Important**
>
> Wenn beim Versuch, eine Verbindung mit cqlsh herzustellen, weiterhin Fehler auftreten, führen Sie den Befehl mit der `--debug` Option erneut aus und fügen Sie die detaillierte Ausgabe bei der Kontaktaufnahme hinzu. Support

Ich kann mit einem Cassandra-Client-Treiber keine Verbindung zu Amazon Keyspaces herstellen

In den folgenden Abschnitten werden die häufigsten Fehler beim Herstellen einer Verbindung mit einem Cassandra-Client-Treiber beschrieben.

Sie versuchen, mithilfe des DataStax Java-Treibers eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen, erhalten aber eine **`NodeUnavailableException`** Fehlermeldung.

Wenn die Verbindung, über die die Anfrage versucht wird, unterbrochen ist, führt dies zu dem folgenden Fehler.

```
[com.datastax.oss.driver.api.core.NodeUnavailableException: No connection
 was available to Node(endPoint=vpce-22ff22f2f22222fff-aa1bb234.cassandra.us-
west-2.vpce.amazonaws.com/11.1.1111.222:9142, hostId=1a23456b-
c77d-8888-9d99-146cb22d6ef6, hashCode=123ca4567)]
```

Um dieses Problem zu beheben, ermitteln Sie den Heartbeat-Wert und senken Sie ihn auf 30 Sekunden, falls er höher ist.

```
advanced.heartbeat.interval = 30 seconds
```

Suchen Sie dann nach dem zugehörigen Timeout und stellen Sie sicher, dass der Wert auf mindestens 5 Sekunden eingestellt ist.

```
advanced.connection.init-query-timeout = 5 seconds
```

Sie versuchen, mithilfe eines Treibers und des Sigv4-Plug-ins eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen, erhalten aber eine **`AttributeError`** Fehlermeldung.

Wenn die Anmeldeinformationen nicht korrekt konfiguriert sind, führt dies zu dem folgenden Fehler.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
 {'44.234.22.154:9142': AttributeError("'NoneType' object has no attribute
 'access_key'")})
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Sie die Ihrem IAM-Benutzer oder Ihrer IAM-Rolle zugewiesenen Anmeldeinformationen weitergeben, wenn Sie das SigV4-Plug-In verwenden. Das SigV4-Plugin benötigt die folgenden Anmeldeinformationen.

- `AWS_ACCESS_KEY_ID`— Gibt einen AWS Zugriffsschlüssel an, der einem IAM-Benutzer oder einer IAM-Rolle zugeordnet ist.
- `AWS_SECRET_ACCESS_KEY`— Gibt den geheimen Schlüssel an, der dem Zugriffsschlüssel zugeordnet ist. Dies ist im Wesentlichen das "Passwort" für den Zugriffsschlüssel.

Weitere Informationen über Zugriffsschlüssel und das SigV4-Plugin finden Sie unterthe section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Sie versuchen, mithilfe eines Treibers eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen, erhalten aber eine **`PartialCredentialsError`** Fehlermeldung.

Wenn der `AWS_SECRET_ACCESS_KEY` fehlt, kann dies zu dem folgenden Fehler führen.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
 {'44.234.22.153:9142':
 PartialCredentialsError('Partial credentials found in config-file, missing:
 aws_secret_access_key')})
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Sie bei der Verwendung des SigV4-Plug-ins `AWS_ACCESS_KEY_ID` sowohl das `AWS_SECRET_ACCESS_KEY` als auch das übergeben. Weitere Informationen zu Zugriffsschlüsseln und dem SigV4-Plugin finden Sie unter. the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”

Sie versuchen, mithilfe eines Treibers eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen, erhalten aber eine **`Invalid signature`** Fehlermeldung.

Dies kann der Fall sein, wenn eine der für die Signatur erforderlichen Komponenten falsch oder für die Sitzung nicht richtig definiert ist.

- `AWS_ACCESS_KEY_ID`
- `AWS_SECRET_ACCESS_KEY`
- `AWS_DEFAULT_REGION`

Der folgende Fehler ist ein Beispiel für ungültige Zugriffsschlüssel.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
 {'11.234.11.234:9142':
 AuthenticationFailed('Failed to authenticate to 11.234.11.234:9142: Error from server:
 code=0100
 [Bad credentials] message="Authentication failure: Invalid signature"')})
```

Um dieses Problem zu beheben, stellen Sie sicher, dass die Zugriffsschlüssel und die AWS-Region korrekt für das SigV4-Plugin für den Zugriff auf Amazon Keyspaces konfiguriert wurden. Weitere Informationen über Zugriffsschlüssel und das SigV4-Plugin finden Sie unter. the section called "Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS "

Meine VPC-Endpunktverbindung funktioniert nicht richtig

Sie versuchen, über VPC-Endpunkte eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber Tokenzuordnungsfehler oder Sie haben einen niedrigen Durchsatz.

Dies kann der Fall sein, wenn die VPC-Endpunktverbindung nicht korrekt konfiguriert ist.

Überprüfen Sie die folgenden Konfigurationsdetails, um diese Probleme zu beheben. Eine schrittweise Anleitung zur Konfiguration einer Verbindung über VPC-Schnittstellen-Endpunkte für Amazon Keyspaces finden Sie unter. the section called "Verbindung mit VPC-Endpunkten herstellen"

1. Stellen Sie sicher, dass die IAM-Entität, die für die Verbindung mit Amazon Keyspaces verwendet wird, read/write Zugriff auf die Benutzertabelle und Lesezugriff auf die Systemtabellen hat, wie im folgenden Beispiel gezeigt.

   ```
   {
      "Version":"2012-10-17",
      "Statement":[
         {
   ```

```
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

2. Vergewissern Sie sich, dass die für die Verbindung mit Amazon Keyspaces verwendete IAM-Entität über die erforderlichen Leseberechtigungen für den Zugriff auf die VPC-Endpunktinformationen auf Ihrer Amazon EC2 EC2-Instance verfügt, wie im folgenden Beispiel gezeigt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

> **Note**
>
> Die verwalteten Richtlinien `AmazonKeyspacesReadOnlyAccess_v2` `AmazonKeyspacesFullAccess` enthalten die erforderlichen Berechtigungen, damit Amazon Keyspaces auf die Amazon EC2 EC2-Instance zugreifen kann, um Informationen über verfügbare Schnittstellen-VPC-Endpunkte zu lesen.

Weitere Informationen zu VPC-Endpunkten finden Sie unter the section called “Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces”

3. Stellen Sie sicher, dass die SSL-Konfiguration des Java-Treibers die Hostnamenvalidierung auf False setzt, wie in diesem Beispiel gezeigt.

```
hostname-validation = false
```

Weitere Hinweise zur Treiberkonfiguration finden Sie unterthe section called “Schritt 2: Konfigurieren Sie den Treiber”.

4. Um zu überprüfen, ob der VPC-Endpunkt korrekt konfiguriert wurde, können Sie die folgende Anweisung in Ihrer VPC ausführen.

> **Note**
>
> Sie können Ihre lokale Entwicklerumgebung oder den Amazon Keyspaces CQL-Editor nicht verwenden, um diese Konfiguration zu bestätigen, da sie den öffentlichen Endpunkt verwenden.

```
SELECT peer FROM system.peers;
```

Die Ausgabe sollte ähnlich wie in diesem Beispiel aussehen und je nach VPC-Setup und Region zwischen 2 und 6 Knoten mit privaten IPv4-Adressen zurückgeben, wenn eine Verbindung über ein IPv4-Netzwerk hergestellt wird. AWS

```
peer
---------------
 192.0.2.0.15
 192.0.2.0.24
 192.0.2.0.13
 192.0.2.0.7
 192.0.2.0.8

(5 rows)
```

Ich kann keine Verbindung herstellen mit `cassandra-stress`

Sie versuchen, mit dem **`cassandra-stress`** Befehl eine Verbindung zu Amazon Keyspaces herzustellen, erhalten aber eine **`SSL context`** Fehlermeldung.

Dies passiert, wenn Sie versuchen, eine Verbindung zu Amazon Keyspaces herzustellen, der TrustStore aber nicht richtig eingerichtet ist. Amazon Keyspaces erfordert die Verwendung von Transport Layer Security (TLS), um Verbindungen mit Clients zu sichern.

In diesem Fall wird der folgende Fehler angezeigt.

```
Error creating the initializing the SSL Context
```

Um dieses Problem zu lösen, folgen Sie den Anweisungen zur Einrichtung eines TrustStores, wie in diesem Thema the section called “Bevor Sie beginnen” beschrieben.

Sobald der TrustStore eingerichtet ist, sollten Sie mit dem folgenden Befehl eine Verbindung herstellen können.

```
./cassandra-stress user profile=./profile.yaml n=100 "ops(insert=1,select=1)"
 cl=LOCAL_QUORUM -node "cassandra.eu-north-1.amazonaws.com" -port native=9142
 -transport ssl-alg="PKIX" truststore="./cassandra_truststore.jks" truststore-
password="trustStore_pw" -mode native cql3 user="user_name" password="password"
```

Ich kann mit IAM-Identitäten keine Verbindung herstellen

Sie versuchen, mithilfe einer IAM-Identität eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen, erhalten aber eine **`Unauthorized`** Fehlermeldung.

Dies passiert, wenn Sie versuchen, mithilfe einer IAM-Identität (z. B. eines IAM-Benutzers) eine Verbindung zu einer Amazon Keyspaces-Tabelle herzustellen, ohne die Richtlinie zu implementieren und dem Benutzer zuerst die erforderlichen Berechtigungen zu erteilen.

In diesem Fall wird der folgende Fehler angezeigt.

```
Connection error: ('Unable to connect to any servers', {'3.234.248.202':
 AuthenticationFailed('Failed to authenticate to 3.234.248.202:
Error from server: code=2100 [Unauthorized] message="User
 arn:aws:iam::1234567890123:user/testuser has no permissions."',)})
```

Überprüfen Sie die Berechtigungen des IAM-Benutzers, um dieses Problem zu beheben. Um eine Verbindung mit einem Standardtreiber herzustellen, muss ein Benutzer mindestens SELECT Zugriff auf die Systemtabellen haben, da die meisten Treiber das System lesen, keyspaces/tables wenn sie die Verbindung herstellen.

Beispiele für IAM-Richtlinien, die Zugriff auf das System und Benutzertabellen von Amazon Keyspaces gewähren, finden Sie unter. the section called “Zugreifen auf Amazon Keyspaces-Tabellen”

Den Abschnitt zur Fehlerbehebung speziell für IAM finden Sie unter. the section called “Fehlerbehebung”

Ich versuche, Daten mit cqlsh zu importieren und die Verbindung zu meiner Amazon Keyspaces-Tabelle ist unterbrochen

Sie versuchen, Daten mit cqlsh auf Amazon Keyspaces hochzuladen, erhalten aber Verbindungsfehler.

Die Verbindung zu Amazon Keyspaces schlägt fehl, nachdem der cqlsh-Client drei aufeinanderfolgende Fehler beliebiger Art vom Server empfangen hat. Der Cqlsh-Client schlägt mit der folgenden Meldung fehl.

```
Failed to import 1 rows: NoHostAvailable - , will retry later, attempt 3 of 100
```

Um diesen Fehler zu beheben, müssen Sie sicherstellen, dass die zu importierenden Daten dem Tabellenschema in Amazon Keyspaces entsprechen. Überprüfen Sie die Importdatei auf Analysefehler. Sie können versuchen, eine einzelne Datenzeile zu verwenden, indem Sie eine INSERT-Anweisung verwenden, um den Fehler zu isolieren.

Der Client versucht automatisch, die Verbindung wiederherzustellen.

Behebung von Kapazitätsverwaltungsfehlern in Amazon Keyspaces

Haben Sie Probleme mit der serverlosen Kapazität? Im Folgenden finden Sie einige häufig auftretende Probleme und deren Lösung.

Kapazitätsfehler bei serverlosen Servern

In diesem Abschnitt wird beschrieben, wie Fehler im Zusammenhang mit der serverlosen Kapazitätsverwaltung erkannt und behoben werden können. Beispielsweise könnten Sie

Ereignisse mit unzureichender Kapazität beobachten, wenn Ihre Anwendung die bereitgestellte Durchsatzkapazität überschreitet.

Da es sich bei Apache Cassandra um clusterbasierte Software handelt, die für die Ausführung auf einer Flotte von Knoten konzipiert ist, enthält sie keine Ausnahmemeldungen, die sich auf serverlose Funktionen wie die Durchsatzkapazität beziehen. Die meisten Treiber verstehen nur die Fehlercodes, die in Apache Cassandra verfügbar sind, sodass Amazon Keyspaces dieselben Fehlercodes verwendet, um die Kompatibilität aufrechtzuerhalten.

Um Cassandra-Fehler den zugrunde liegenden Kapazitätsereignissen zuzuordnen, können Sie Amazon verwenden, CloudWatch um die relevanten Amazon Keyspaces-Metriken zu überwachen. Insufficient-capacity Ereignisse, die zu clientseitigen Fehlern führen, können basierend auf der Ressource, die das Ereignis verursacht, in diese drei Gruppen eingeteilt werden:

- Tabelle — Wenn Sie den Modus Bereitgestellte Kapazität für eine Tabelle wählen und Ihre Anwendung den bereitgestellten Durchsatz überschreitet, treten möglicherweise Fehler aufgrund unzureichender Kapazität auf. Weitere Informationen finden Sie unter the section called “ read/write Kapazitätsmodi konfigurieren”.
- Partition — Es kann zu Ereignissen mit unzureichender Kapazität kommen, wenn der Datenverkehr auf einer bestimmten Partition 3.000 RCUs oder 1.000 WCUs überschreitet. Als bewährte Methode empfehlen wir, den Datenverkehr gleichmäßig auf die Partitionen zu verteilen. Weitere Informationen finden Sie unter the section called “Datenmodellierung”.
- Verbindung — Möglicherweise ist der Durchsatz unzureichend, wenn Sie das Kontingent für die maximale Anzahl von Vorgängen pro Sekunde pro Verbindung überschreiten. Um den Durchsatz zu erhöhen, können Sie bei der Konfiguration der Verbindung mit dem Treiber die Anzahl der Standardverbindungen erhöhen.

  Informationen zur Konfiguration von Verbindungen für Amazon Keyspaces finden Sie unterthe section called “Wie konfiguriert man Verbindungen”. Weitere Informationen zur Optimierung von Verbindungen über VPC-Endpunkte finden Sie unter. the section called “VPC-Endpunktverbindungen”

Um festzustellen, welche Ressource das Ereignis mit unzureichender Kapazität verursacht, das den clientseitigen Fehler zurückgibt, können Sie das Dashboard in der Amazon Keyspaces-Konsole überprüfen. Standardmäßig bietet die Konsole im Abschnitt Kapazität und zugehörige CloudWatch Metriken auf der Registerkarte Kapazität der Tabelle eine aggregierte Ansicht der gängigsten Kapazitäts- und Datenverkehrsmetriken.

Um Ihr eigenes Dashboard mit Amazon zu erstellen CloudWatch, überprüfen Sie die folgenden Amazon Keyspaces-Metriken.

- `PerConnectionRequestRateExceeded`— Anfragen an Amazon Keyspaces, die das Kontingent für die Anforderungsrate pro Verbindung überschreiten. Jede Clientverbindung mit Amazon Keyspaces kann bis zu 3 000 CQL-Anfragen pro Sekunde unterstützen. Sie können mehr als 3 000 Anfragen pro Sekunde ausführen, indem Sie mehrere Verbindungen herstellen.
- `ReadThrottleEvents`— Anfragen an Amazon Keyspaces, die die Lesekapazität für eine Tabelle überschreiten.
- `StoragePartitionThroughputCapacityExceeded`— Anfragen an eine Amazon Keyspaces-Speicherpartition, die die Durchsatzkapazität der Partition überschreiten. Amazon Keyspaces-Speicherpartitionen können bis zu 1000 WCU/WRU pro Sekunde und 3000 RCU/RRU pro Sekunde unterstützen. Um diese Ausnahmen zu vermeiden, empfehlen wir Ihnen, Ihr Datenmodell zu überprüfen, um den read/write Datenverkehr auf mehrere Partitionen zu verteilen.
- `WriteThrottleEvents`— Anfragen an Amazon Keyspaces, die die Schreibkapazität für eine Tabelle überschreiten.

Weitere Informationen dazu finden Sie CloudWatch unterthe section called “Überwachung mit CloudWatch”. Eine Liste aller verfügbaren CloudWatch Metriken für Amazon Keyspaces finden Sie unterthe section called “Metriken und Dimensionen”.

> **Note**
>
> Um mit einem benutzerdefinierten Dashboard zu beginnen, das alle häufig beobachteten Metriken für Amazon Keyspaces anzeigt, können Sie eine vorgefertigte CloudWatch Vorlage verwenden, die GitHub im AWS Beispiel-Repository verfügbar ist.

Themen

- Ich erhalte von meinem Client-Treiber Fehler mit NoHostAvailable unzureichender Kapazität
- Ich erhalte beim Datenimport Schreib-Timeout-Fehler
- Ich kann die tatsächliche Speichergröße eines Schlüsselraums oder einer Tabelle nicht sehen

Ich erhalte von meinem Client-Treiber Fehler mit **`NoHostAvailable`** unzureichender Kapazität

Sie sehen **`Read_Timeout`** unsere Ausnahmen für eine Tabelle. **`Write_Timeout`**

Wiederholter Versuch, in eine Amazon Keyspaces-Tabelle mit unzureichender Kapazität zu schreiben oder aus ihr zu lesen, kann zu clientseitigen Treiberfehlern führen.

Wird verwendet CloudWatch , um Ihre bereitgestellten und tatsächlichen Durchsatzmetriken sowie Ereignisse zu unzureichender Kapazität für die Tabelle zu überwachen. Beispielsweise schlägt eine Leseanforderung, die nicht über genügend Durchsatzkapazität verfügt, mit einer `Read_Timeout` Ausnahme fehl und wird der `ReadThrottleEvents` Metrik zugewiesen. Eine Schreibanforderung, die nicht über genügend Durchsatzkapazität verfügt, schlägt mit einer `Write_Timeout` Ausnahme fehl und wird der `WriteThrottleEvents` Metrik zugewiesen. Weitere Informationen zu diesen Metriken finden Sie unter the section called “Metriken und Dimensionen”.

Um diese Probleme zu lösen, sollten Sie eine der folgenden Optionen in Betracht ziehen.

- Erhöhen Sie den bereitgestellten Durchsatz für die Tabelle. Dies ist die maximale Menge an Durchsatzkapazität, die eine Anwendung verbrauchen kann. Weitere Informationen finden Sie unter the section called “Lesekapazitätseinheiten und Schreibkapazitätseinheiten”.
- Lassen Sie den Service die Durchsatzkapazität mit automatischer Skalierung in Ihrem Namen verwalten. Weitere Informationen finden Sie unter the section called “Verwalten Sie die Durchsatzkapazität mit Auto Scaling”.
- Wählen Sie den On-demandKapazitätsmodus für die Tabelle. Weitere Informationen finden Sie unter the section called “Konfigurieren Sie den Kapazitätsmodus auf Anforderung”.

Informationen dazu, wie Sie das Standardkapazitätskontingent für Ihr Konto erhöhen müssen, finden Sie unterKontingente.

Es werden Fehler im Zusammenhang mit der Überschreitung der Partitionskapazität angezeigt.

Wenn Sie den Fehler sehen, ist `StoragePartitionThroughputCapacityExceeded` die Partitionskapazität vorübergehend überschritten. Dies kann automatisch durch Adaptive Capacity oder On-Demand-Kapazität behoben werden. Wir empfehlen, Ihr Datenmodell zu überprüfen, um den read/write Datenverkehr auf mehrere Partitionen zu verteilen, um diese Fehler zu vermeiden. Amazon Keyspaces-Speicherpartitionen können bis zu 1000 WCU/WRU pro Sekunde und 3000 RCU/RRU pro Sekunde unterstützen. Weitere Informationen darüber, wie Sie Ihr Datenmodell

verbessern können, um den read/write Datenverkehr auf mehr Partitionen zu verteilen, finden Sie unterthe section called “Datenmodellierung”.

`Write_Timeout`Ausnahmen können auch durch eine erhöhte Anzahl gleichzeitiger Schreibvorgänge verursacht werden, die statische und nicht statische Daten in derselben logischen Partition umfassen. Wenn erwartet wird, dass der Datenverkehr mehrere gleichzeitige Schreibvorgänge ausführt, die statische und nicht statische Daten innerhalb derselben logischen Partition umfassen, empfehlen wir, statische und nicht statische Daten getrennt zu schreiben. Das separate Schreiben der Daten trägt auch zur Optimierung der Durchsatzkosten bei.

Es werden Fehler im Zusammenhang mit der Überschreitung der Verbindungsanforderungsrate angezeigt.

Die Anzeige ist auf eine der folgenden Ursachen `PerConnectionRequestRateExceeded` zurückzuführen.

- Möglicherweise haben Sie nicht genügend Verbindungen pro Sitzung konfiguriert.
- Möglicherweise erhalten Sie weniger Verbindungen als verfügbare Peers, weil Sie die VPC-Endpunktberechtigungen nicht richtig konfiguriert haben. Weitere Informationen zu VPC-Endpunktrichtlinien finden Sie unterthe section called “Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces”.
- Wenn Sie einen 4.x-Treiber verwenden, überprüfen Sie, ob Sie die Hostnamenüberprüfung aktiviert haben. Der Treiber aktiviert standardmäßig die Überprüfung des TLS-Hostnamens. Diese Konfiguration führt dazu, dass Amazon Keyspaces für den Treiber als Cluster mit einem Knoten erscheint. Wir empfehlen, dass Sie die Hostnamen-Verifizierung ausschalten.

Wir empfehlen Ihnen, die folgenden bewährten Methoden zu befolgen, um sicherzustellen, dass Ihre Verbindungen und Ihr Durchsatz optimiert sind:

- Konfigurieren Sie die Durchsatzoptimierung für CQL-Abfragen.

  Amazon Keyspaces unterstützt bis zu 3.000 CQL-Abfragen pro TCP-Verbindung pro Sekunde, aber die Anzahl der Verbindungen, die ein Treiber herstellen kann, ist unbegrenzt.

  Die meisten Open-Source-Cassandra-Treiber richten einen Verbindungspool zu Cassandra ein und verteilen Abfragen über diesen Verbindungspool. Amazon Keyspaces stellt Treibern 9 Peer-IP-Adressen zur Verfügung. Das Standardverhalten der meisten Treiber besteht darin, eine einzige Verbindung zu jeder Peer-IP-Adresse herzustellen. Daher beträgt der maximale CQL-

Abfragedurchsatz eines Treibers, der die Standardeinstellungen verwendet, 27.000 CQL-Abfragen pro Sekunde.

Um diese Zahl zu erhöhen, empfehlen wir Ihnen, die Anzahl der Verbindungen pro IP-Adresse zu erhöhen, die Ihr Treiber in seinem Verbindungspool verwaltet. Wenn Sie beispielsweise die maximale Anzahl an Verbindungen pro IP-Adresse auf 2 festlegen, wird der maximale Durchsatz Ihres Treibers auf 54.000 CQL-Abfragen pro Sekunde verdoppelt.

- Optimieren Sie Ihre Einzelknotenverbindungen.

  Standardmäßig stellen die meisten Open-Source-Cassandra-Treiber beim Einrichten einer Sitzung eine oder mehrere Verbindungen zu jeder in der `system.peers` Tabelle angegebenen IP-Adresse her. Bestimmte Konfigurationen können jedoch dazu führen, dass ein Treiber eine Verbindung zu einer einzigen Amazon Keyspaces-IP-Adresse herstellt. Dies kann passieren, wenn der Treiber versucht, den SSL-Hostnamen der Peer-Knoten (z. B. DataStax Java-Treiber) zu überprüfen, oder wenn er eine Verbindung über einen VPC-Endpunkt herstellt.

  Um dieselbe Verfügbarkeit und Leistung wie ein Treiber mit Verbindungen zu mehreren IP-Adressen zu erhalten, empfehlen wir Ihnen, wie folgt vorzugehen:

  - Erhöhen Sie die Anzahl der Verbindungen pro IP je nach gewünschtem Client-Durchsatz auf 9 oder höher.
  - Erstellen Sie eine benutzerdefinierte Wiederholungsrichtlinie, die sicherstellt, dass Wiederholungsversuche für denselben Knoten ausgeführt werden. Die Ausgabe des obigen Befehls sieht in etwa folgendermaßen aus (JSON format).

    the section called “Wie konfiguriert man Richtlinien für Wiederholungsversuche”.
  - Wenn Sie VPC-Endpunkte verwenden, gewähren Sie der IAM-Entität, die für die Verbindung mit Amazon Keyspaces verwendet wird, Zugriffsberechtigungen, um Ihre VPC nach den Endpunkt- und Netzwerkschnittstelleninformationen abzufragen. Dies verbessert den Lastausgleich und erhöht den Durchsatz. read/write Weitere Informationen finden Sie unter ???.

Ich erhalte beim Datenimport Schreib-Timeout-Fehler

Beim Hochladen von Daten mithilfe des Befehls wird ein Timeout-Fehler angezeigt. **`cqlsh COPY`**

```
Failed to import 1 rows: WriteTimeout - Error from server: code=1100 [Coordinator node
 timed out waiting for replica nodes' responses]
 message="Operation timed out - received only 0 responses." info={'received_responses':
 0, 'required_responses': 2, 'write_type': 'SIMPLE', 'consistency':
```

```
'LOCAL_QUORUM'}, will retry later, attempt 1 of 100
```

Amazon Keyspaces verwendet die `WriteTimeout` Ausnahmen `ReadTimeout` und, um anzuzeigen, wenn eine Schreibanforderung aufgrund unzureichender Durchsatzkapazität fehlschlägt. Um bei der Diagnose von Ausnahmen mit unzureichender Kapazität zu helfen, veröffentlicht Amazon Keyspaces die folgenden Kennzahlen in Amazon CloudWatch.

- `WriteThrottleEvents`
- `ReadThrottledEvents`
- `StoragePartitionThroughputCapacityExceeded`

Um Fehler mit unzureichender Kapazität beim Laden von Daten zu beheben, verringern Sie die Schreibrate pro Worker oder die Gesamtaufnahmerate und versuchen Sie dann erneut, die Zeilen hochzuladen. Weitere Informationen finden Sie unter the section called “Schritt 4: Konfigurieren Sie die Einstellungen von `cqlsh COPY FROM`”. Für eine robustere Option zum Hochladen von Daten sollten Sie DSBulk verwenden, das im Repository verfügbar ist. GitHub Schrittweise Anleitungen hierzu finden Sie unter the section called “Daten mit DSBulk laden”.

Ich kann die tatsächliche Speichergröße eines Schlüsselraums oder einer Tabelle nicht sehen

Sie können die tatsächliche Speichergröße des Schlüsselraums oder der Tabelle nicht sehen.

Weitere Informationen zur Speichergröße Ihrer Tabelle finden Sie unterthe section called “Auswerten Ihrer Kosten auf Tabellenebene”. Sie können die Speichergröße auch schätzen, indem Sie mit der Berechnung der Zeilengröße in einer Tabelle beginnen. Detaillierte Anweisungen zur Berechnung der Zeilengröße finden Sie unterthe section called “Schätzen Sie die Zeilengröße”.

Behebung von Sprachfehlern bei der Datendefinition in Amazon Keyspaces

Haben Sie Probleme beim Erstellen von Ressourcen? Im Folgenden finden Sie einige häufig auftretende Probleme und wie Sie sie lösen können.

Sprachfehler bei der Datendefinition

Amazon Keyspaces führt DDL-Operationen (Data Definition Language) asynchron aus, z. B. das Erstellen und Löschen von Schlüsselräumen und Tabellen. Wenn eine Anwendung versucht, die Ressource zu verwenden, bevor sie bereit ist, schlägt der Vorgang fehl.

Sie können den Erstellungsstatus neuer Schlüsselräume und Tabellen in der überwachen. Dies gibt an AWS-Managementkonsole, wann ein Schlüsselraum oder eine Tabelle ausstehend oder aktiv ist. Sie können den Erstellungsstatus eines neuen Schlüsselraums oder einer neuen Tabelle auch programmgesteuert überwachen, indem Sie die Systemschematabelle abfragen. Ein Schlüsselraum oder eine Tabelle wird im Systemschema sichtbar, wenn sie einsatzbereit ist.

> **Note**
>
> Um die Erstellung von Schlüsselräumen mithilfe von zu optimieren CloudFormation, können Sie dieses Hilfsprogramm verwenden, um CQL-Skripten in Vorlagen zu konvertieren. CloudFormation Das Tool ist im Repository verfügbar. GitHub

Themen

- Ich habe einen neuen Schlüsselraum erstellt, kann ihn aber nicht anzeigen oder darauf zugreifen
- Ich habe eine neue Tabelle erstellt, kann sie aber nicht anzeigen oder darauf zugreifen
- Ich versuche, eine Tabelle mithilfe von Amazon Keyspaces Point-in-Time Recovery (PITR) wiederherzustellen, aber die Wiederherstellung schlägt fehl
- Ich versuche, benutzerdefinierte TTL-Einstellungen (Time to Live) zu bearbeiten, aber der Vorgang schlägt fehl INSERT/UPDATE
- Ich versuche, Daten in meine Amazon Keyspaces-Tabelle hochzuladen, und erhalte die Fehlermeldung, dass die Anzahl der Spalten überschritten wird
- Ich versuche, Daten in meiner Amazon Keyspaces-Tabelle zu löschen, und das Löschen schlägt für den Bereich fehl

Ich habe einen neuen Schlüsselraum erstellt, kann ihn aber nicht anzeigen oder darauf zugreifen

Sie erhalten Fehler von Ihrer Anwendung, die versucht, auf einen neuen Schlüsselraum zuzugreifen.

Wenn Sie versuchen, auf einen neu erstellten Amazon Keyspaces-Schlüsselraum zuzugreifen, der immer noch asynchron erstellt wird, erhalten Sie eine Fehlermeldung. Der folgende Fehler ist ein Beispiel.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 keyspace mykeyspace"
```

Das empfohlene Entwurfsmuster, um zu überprüfen, wann ein neuer Keyspace einsatzbereit ist, besteht darin, die Amazon Keyspaces-Systemschematabellen (system_schema_mcs.*) abzufragen.

Weitere Informationen finden Sie unter the section called “Überprüfen Sie den Status der Keyspace-Erstellung”.

Ich habe eine neue Tabelle erstellt, kann sie aber nicht anzeigen oder darauf zugreifen

Sie erhalten Fehler von Ihrer Anwendung, die versucht, auf eine neue Tabelle zuzugreifen.

Wenn Sie versuchen, auf eine neu erstellte Amazon Keyspaces-Tabelle zuzugreifen, die immer noch asynchron erstellt wird, erhalten Sie eine Fehlermeldung. Beispielsweise schlägt der Versuch, eine Tabelle abzufragen, die noch nicht verfügbar ist, mit einem `unconfigured table` Fehler fehl.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 table mykeyspace.mytable"
```

Der Versuch, die Tabelle mit anzuzeigen, `sync_table()` schlägt mit einem fehl`KeyError`.

```
KeyError: 'mytable'
```

Das empfohlene Entwurfsmuster, um zu überprüfen, wann eine neue Tabelle einsatzbereit ist, besteht darin, die Amazon Keyspaces-Systemschematabellen (system_schema_mcs.*) abzufragen.

Dies ist die Beispielausgabe für eine Tabelle, die gerade erstellt wird.

```
user-at-123@cqlsh:system_schema_mcs> select table_name,status from
 system_schema_mcs.tables where keyspace_name='example_keyspace' and
 table_name='example_table';

table_name | status

------------+----------
```

```
example_table | CREATING

(1 rows)
```

Dies ist die Beispielausgabe für eine Tabelle, die aktiv ist.

```
user-at-123@cqlsh:system_schema_mcs> select table_name,status from
 system_schema_mcs.tables where keyspace_name='example_keyspace' and
 table_name='example_table';

table_name | status

------------+----------

example_table | ACTIVE

(1 rows)
```

Weitere Informationen finden Sie unter the section called “Überprüfen Sie den Status der Tabellenerstellung”.

Ich versuche, eine Tabelle mithilfe von Amazon Keyspaces Point-in-Time Recovery (PITR) wiederherzustellen, aber die Wiederherstellung schlägt fehl

Wenn Sie versuchen, eine Amazon Keyspaces-Tabelle mit Point-in-Time Recovery (PITR) wiederherzustellen, und Sie sehen, dass der Wiederherstellungsvorgang beginnt, aber nicht erfolgreich abgeschlossen wird, haben Sie möglicherweise nicht alle erforderlichen Berechtigungen konfiguriert, die für den Wiederherstellungsprozess für diese bestimmte Tabelle erforderlich sind.

Zusätzlich zu den Benutzerberechtigungen benötigt Amazon Keyspaces möglicherweise Berechtigungen, um während des Wiederherstellungsvorgangs im Namen Ihres Auftraggebers Aktionen durchzuführen. Dies ist der Fall, wenn die Tabelle mit einem vom Kunden verwalteten Schlüssel verschlüsselt ist oder wenn Sie IAM-Richtlinien verwenden, die den eingehenden Datenverkehr einschränken.

Wenn Sie beispielsweise Bedingungsschlüssel in Ihrer IAM-Richtlinie verwenden, um den Quelldatenverkehr auf bestimmte Endpunkte oder IP-Bereiche zu beschränken, schlägt der Wiederherstellungsvorgang fehl. Damit Amazon Keyspaces die Tabellenwiederherstellung im Namen Ihres Prinzipals durchführen kann, müssen Sie der IAM-Richtlinie einen `aws:ViaAWSService` globalen Bedingungsschlüssel hinzufügen.

Weitere Informationen zu Berechtigungen zum Wiederherstellen von Tabellen finden Sie unter. the section called “Konfigurieren Sie die IAM-Berechtigungen für die Wiederherstellung”

Ich versuche, benutzerdefinierte TTL-Einstellungen (Time to Live) zu bearbeiten, aber der Vorgang schlägt fehl INSERT/UPDATE

Wenn Sie versuchen, einen benutzerdefinierten TTL-Wert einzufügen oder zu aktualisieren, schlägt der Vorgang möglicherweise mit dem folgenden Fehler fehl.

```
TTL is not yet supported.
```

Um mithilfe `INSERT` von `UPDATE` OR-Operationen benutzerdefinierte TTL-Werte für Zeilen oder Spalten anzugeben, müssen Sie zuerst TTL für die Tabelle aktivieren. Sie können TTL für eine Tabelle mithilfe der `ttl` benutzerdefinierten Eigenschaft aktivieren.

Weitere Informationen zum Aktivieren benutzerdefinierter TTL-Einstellungen für Tabellen finden Sie unter. the section called “Tabelle mit benutzerdefinierter TTL aktualisieren”

Ich versuche, Daten in meine Amazon Keyspaces-Tabelle hochzuladen, und erhalte die Fehlermeldung, dass die Anzahl der Spalten überschritten wird

Sie laden Daten hoch und haben die Anzahl der Spalten überschritten, die gleichzeitig aktualisiert werden können.

Dieser Fehler tritt auf, wenn Ihr Tabellenschema die maximale Größe von 350 KB überschreitet. Weitere Informationen finden Sie unter Kontingente.

Ich versuche, Daten in meiner Amazon Keyspaces-Tabelle zu löschen, und das Löschen schlägt für den Bereich fehl

Sie versuchen, Daten anhand des Partitionsschlüssels zu löschen, und erhalten einen Fehler beim Löschen des Bereichs.

Dieser Fehler tritt auf, wenn Sie versuchen, mehr als 1.000 Zeilen in einem Löschvorgang zu löschen.

```
Range delete requests are limited by the amount of items that can be deleted in a
 single range.
```

Weitere Informationen finden Sie unter the section called “Bereich löschen”.

Um mehr als 1.000 Zeilen innerhalb einer einzelnen Partition zu löschen, sollten Sie die folgenden Optionen in Betracht ziehen.

- Partitionsweise löschen — Wenn die Mehrheit der Partitionen weniger als 1.000 Zeilen enthält, können Sie versuchen, Daten partitionsweise zu löschen. Wenn die Partitionen mehr als 1.000 Zeilen enthalten, versuchen Sie stattdessen, sie anhand der Clusterspalte zu löschen.
- Nach Clusterspalte löschen — Wenn Ihr Modell mehrere Clusterspalten enthält, können Sie die Spaltenhierarchie verwenden, um mehrere Zeilen zu löschen. Clusterspalten sind eine verschachtelte Struktur, und Sie können viele Zeilen löschen, indem Sie mit der Spalte der obersten Ebene arbeiten.
- Nach einzelnen Zeilen löschen — Sie können durch die Zeilen iterieren und jede Zeile anhand ihres vollständigen Primärschlüssels (Partitionsspalten und Clusterspalten) löschen.
- Als bewährte Methode sollten Sie erwägen, Ihre Zeilen auf mehrere Partitionen aufzuteilen. In Amazon Keyspaces empfehlen wir Ihnen, Ihren Durchsatz auf Tabellenpartitionen zu verteilen. Dadurch werden Daten und Zugriff gleichmäßig auf die physischen Ressourcen verteilt, wodurch der beste Durchsatz erzielt wird. Weitere Informationen finden Sie unter the section called “Datenmodellierung”.

Beachten Sie auch die folgenden Empfehlungen, wenn Sie Löschvorgänge für hohe Arbeitslasten planen.

- Mit Amazon Keyspaces können Partitionen eine praktisch unbegrenzte Anzahl von Zeilen enthalten. Auf diese Weise können Sie Partitionen „breiter“ skalieren als die herkömmliche Cassandra-Richtlinie von 100 MB. Es ist nicht ungewöhnlich, dass Zeitreihen oder Ledger im Laufe der Zeit um mehr als ein Gigabyte an Daten anwachsen.
- Mit Amazon Keyspaces müssen Sie keine Verdichtungsstrategien oder Grabsteine in Betracht ziehen, wenn Sie Löschvorgänge für umfangreiche Workloads durchführen müssen. Sie können so viele Daten löschen, wie Sie möchten, ohne die Leseleistung zu beeinträchtigen.

Überwachung von Amazon Keyspaces (für Apache Cassandra)

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon Keyspaces und Ihren anderen AWS Lösungen. AWS bietet die folgenden Überwachungstools, um Amazon Keyspaces zu beobachten, zu melden, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen zu ergreifen:

- Amazon Keyspaces bietet ein vorkonfiguriertes Dashboard, in dem die Latenz und die Fehler zusammengefasst für alle Tabellen im Konto AWS-Managementkonsole angezeigt werden.
- Amazon CloudWatch überwacht Ihre AWS Ressourcen und die Anwendungen, auf denen Sie laufen, AWS in Echtzeit. Mit maßgeschneiderten Dashboards können Sie Kennzahlen sammeln und verfolgen. Sie können beispielsweise eine Ausgangsbasis für die normale Leistung von Amazon Keyspaces in Ihrer Umgebung erstellen, indem Sie die Leistung zu verschiedenen Zeiten und unter verschiedenen Lastbedingungen messen. Speichern Sie bei der Überwachung von Amazon Keyspaces historische Überwachungsdaten, damit Sie sie mit aktuellen Leistungsdaten vergleichen, normale Leistungsmuster und Leistungsanomalien identifizieren und Methoden zur Behebung von Problemen entwickeln können. Um eine Ausgangsbasis festzulegen, sollten Sie mindestens nach Systemfehlern Ausschau halten. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.
- CloudWatch Amazon-Alarme überwachen eine einzelne Metrik über einen von Ihnen angegebenen Zeitraum und führen eine oder mehrere Aktionen aus, die auf dem Wert der Metrik im Verhältnis zu einem bestimmten Schwellenwert über mehrere Zeiträume basieren. Wenn Sie Amazon Keyspaces beispielsweise im Bereitstellungsmodus mit Application Auto Scaling verwenden, handelt es sich bei der Aktion um eine Benachrichtigung, die vom Amazon Simple Notification Service (Amazon SNS) gesendet wird, um eine Application Auto Scaling-Richtlinie auszuwerten.

  CloudWatch Alarme lösen keine Aktionen aus, nur weil sie sich in einem bestimmten Zustand befinden. Der Status muss sich geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Überwachung von Amazon Keyspaces mit Amazon CloudWatch.
- Mit Amazon CloudWatch Logs können Sie Ihre Protokolldateien aus Amazon Keyspaces-Tabellen und anderen Quellen überwachen CloudTrail, speichern und darauf zugreifen. CloudWatch Logs kann Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr

robusten Speicher archivieren. Weitere Informationen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.

- AWS CloudTrail erfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS-Konto -Kontos erfolgten, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon-S3-Bucket. Sie können die Benutzer und Konten, die AWS aufgerufen haben, identifizieren, sowie die Quell-IP-Adresse, von der diese Aufrufe stammen, und den Zeitpunkt der Aufrufe ermitteln. Weitere Informationen finden Sie im AWS CloudTrail - Benutzerhandbuch.

Amazon EventBridge ist ein serverloser Event-Bus-Service, der es einfach macht, Ihre Anwendungen mit Daten aus einer Vielzahl von Quellen zu verbinden. EventBridge liefert einen Stream von Echtzeitdaten aus Ihren eigenen Anwendungen, Software-as-a-Service (SaaS-) Anwendungen und AWS Diensten und leitet diese Daten an Ziele wie Lambda weiter. Auf diese Weise können Sie Ereignisse überwachen, die in Services auftreten, und ereignisgesteuerte Architekturen erstellen. Weitere Informationen finden Sie im EventBridge Amazon-Benutzerhandbuch.

Themen

- Überwachung von Amazon Keyspaces mit Amazon CloudWatch
- Protokollieren von Amazon Keyspaces-API-Aufrufen mit AWS CloudTrail

Überwachung von Amazon Keyspaces mit Amazon CloudWatch

Sie können Amazon Keyspaces mithilfe von Amazon überwachen. Amazon CloudWatch sammelt Rohdaten und verarbeitet sie zu lesbaren Metriken, die nahezu in Echtzeit ablaufen. Diese Statistiken werden 15 Monate gespeichert, damit Sie auf Verlaufsinformationen zugreifen können und einen besseren Überblick darüber erhalten, wie Ihre Webanwendung oder der Service ausgeführt werden.

Sie können auch Alarme einrichten, die auf bestimmte Grenzwerte achten und Benachrichtigungen senden oder Aktivitäten auslösen, wenn diese Grenzwerte erreicht werden. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.

> **Note**
>
> Um schnell mit einem vorkonfigurierten CloudWatch Dashboard zu beginnen, das allgemeine Kennzahlen für Amazon Keyspaces anzeigt, können Sie eine CloudFormation Vorlage

verwenden, die unter verfügbar ist. https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates

Themen

- Wie verwende ich Amazon Keyspaces-Metriken?
- Amazon Keyspaces-Metriken und -Dimensionen
- Amazon Keyspaces-Metriken anzeigen in CloudWatch
- CloudWatch Alarme zur Überwachung von Amazon Keyspaces erstellen

Wie verwende ich Amazon Keyspaces-Metriken?

Die von Amazon Keyspaces gemeldeten Metriken liefern Informationen, die Sie auf unterschiedliche Weise analysieren können. In der folgenden Liste finden Sie einige häufige Verwendungszwecke für die Metriken. Es handelt sich dabei um Vorschläge für den Einstieg und nicht um eine umfassende Liste. Weitere Informationen zu Kennzahlen und Aufbewahrung finden Sie unter Metriken.

Wie kann ich ...?	Relevante Metriken
Wie kann ich feststellen, wenn ein Systemfehler aufgetreten ist?	Sie können überwachen`SystemErrors` , ob Anfragen zu einem Serverfehlercode geführt haben. In der Regel sollte diese Metrik gleich Null sein. Ist dies nicht der Fall, sollten Sie das genauer untersuchen.
Wie kann ich die durchschnittliche bereitgestellte Lesekapazität mit der verbrauchten Lesekapazität vergleichen?	Um die durchschnittliche bereitgestellte Lesekapazität und die verbrauchte Lesekapazität zu überwachen 1. Stellen Sie den Zeitraum für `ConsumedReadCapacityUnits` und `ProvisionedReadCapacityUnits` auf das Intervall ein, das Sie überwachen möchten. 2. Ändern Sie die Statistik für `ConsumedReadCapacityUnits` von `Average` bis`Sum`. 3. Erstellen Sie einen neuen leeren Math-Ausdruck. 4. Geben Sie im Abschnitt Details des neuen mathematischen Ausdrucks die ID von ein **`ConsumedReadCapacityUnits`** und dividieren Sie die Metrik durch die

Wie kann ich ...?	Relevante Metriken
	CloudWatch PERIOD-Funktion der Metrik (metric_id/(PERIOD (metric_id)). 5. Auswahl aufheben. `ConsumedReadCapacityUnits` Sie können jetzt Ihre durchschnittlich verbrauchte Lesekapazität mit Ihrer bereitgestellten Kapazität vergleichen. Weitere Informationen zu den grundlegenden Rechenfunktionen und zum Erstellen einer Zeitreihe finden Sie unter Metrische Mathematik verwenden.
Wie kann ich die durchschnittliche bereitgestellte Schreibkapazität mit der verbrauchten Schreibkapazität vergleichen?	Um die durchschnittliche bereitgestellte Schreibkapazität und die verbrauchte Schreibkapazität zu überwachen 1. Stellen Sie den Zeitraum für `ConsumedWriteCapacityUnits` und `ProvisionedWriteCapacityUnits` auf das Intervall ein, das Sie überwachen möchten. 2. Ändern Sie die Statistik für `ConsumedWriteCapacityUnits` von `Average` bis`Sum`. 3. Erstellen Sie einen neuen leeren Math-Ausdruck. 4. Geben Sie im Abschnitt Details des neuen mathematischen Ausdrucks die ID von ein **`ConsumedWriteCapacityUnits`** und dividieren Sie die Metrik durch die CloudWatch PERIOD-Funktion der Metrik (metric_id/(PERIOD (metric_id)). 5. Auswahl aufheben. `ConsumedWriteCapacityUnits` Sie können jetzt Ihre durchschnittlich verbrauchte Schreibkapazität mit Ihrer bereitgestellten Kapazität vergleichen. Weitere Informationen zu grundlegenden arithmetischen Funktionen und zum Erstellen einer Zeitreihe finden Sie unter Metrische Mathematik verwenden.

Amazon Keyspaces-Metriken und -Dimensionen

Wenn Sie mit Amazon Keyspaces interagieren, werden Metriken und Dimensionen an Amazon CloudWatch gesendet. Alle Metriken werden aggregiert und jede Minute gemeldet. Die folgenden Abschnitte geben einen Überblick über alle verfügbaren Metriken für Amazon Keyspaces.

Themen

- Amazon-Keyspaces-Metriken
- Dimensionen für Amazon Keyspaces-Metriken
- Metriken für Amazon Keyspaces Change Data Capture (CDC)
- Dimensionen für Amazon Keyspaces-Metriken zur Datenerfassung (CDC)

Amazon-Keyspaces-Metriken

Amazon CloudWatch aggregiert Amazon Keyspaces-Metriken in Intervallen von einer Minute.

Nicht alle Statistiken, wie `Average` oder `Sum` sind für jede Metrik anwendbar. All diese Werte sind jedoch über die Amazon Keyspaces-Konsole oder über die CloudWatch Konsole oder AWS SDKs für alle Metriken verfügbar. AWS CLI In der folgenden Tabelle enthält jede Metrik eine Liste gültiger Statistiken, die für diese Metrik gelten.

Metrik	Description
`AccountMaxTableLevelReads`	Die maximale Anzahl von Lesekapazitätseinheiten, die von einer Tabelle eines Kontos verwendet werden können. Bei On-Demand-Tabellen begrenzt dieses Limit die maximale Anzahl von Leseanforderungseinheiten, die eine Tabelle verwenden kann. Einheiten: `Count` Gültige Statistiken: • `Maximum`— Die maximale Anzahl von Lesekapaz itätseinheiten, die von einer Tabelle des Kontos verwendet werden können.

Metrik	Description
`AccountMaxTableLevelWrites`	Die maximale Anzahl von Schreibkapazitätseinheiten, die von einer Tabelle eines Kontos verwendet werden können. Bei On-Demand-Tabellen begrenzt dieses Limit die maximale Anzahl von Schreibanforderungseinheiten, die eine Tabelle verwenden kann. Einheiten: `Count` Gültige Statistiken: • `Maximum`— Die maximale Anzahl von Schreibkapazitätseinheiten, die von einer Tabelle des Kontos verwendet werden können.
`AccountProvisionedReadCapacityUtilization`	Prozentsatz der bereitgestellten Lesekapazitätseinheiten, die vom Konto genutzt werden. Einheiten: `Percent` Gültige Statistiken: • `Maximum` – Der maximale Prozentsatz der bereitgestellten Lesekapazitätseinheiten, die vom Konto genutzt werden. • `Minimum` – Der Mindestprozentsatz der bereitgestellten Lesekapazitätseinheiten, die vom Konto genutzt werden. • `Average` – Der durchschnittliche Prozentsatz der bereitgestellten Lesekapazitätseinheiten, die vom Konto genutzt werden. Die Metrik wird für Intervallen von fünf Minuten veröffentlicht. Wenn Sie die bereitgestellten Lesekapazitätseinheiten schnell anpassen, spiegelt diese Statistik möglicherweise nicht den tatsächlichen Durchschnitt wider.

Metrik	Description
`AccountProvisioned WriteCapacityUtilization`	Prozentsatz der bereitgestellten Schreibkapazitätse inheiten, die vom Konto genutzt werden. Einheiten: `Percent` Gültige Statistiken: • `Maximum` – Der maximale Prozentsatz der bereitges tellten Schreibkapazitätseinheiten, die vom Konto genutzt werden. • `Minimum` – Der minimale Prozentsatz der bereitges tellten Schreibkapazitätseinheiten, die vom Konto genutzt werden. • `Average` – Der durchschnittliche Prozentsatz der bereitgestellten Schreibkapazitätseinheiten, die vom Konto genutzt werden. Die Metrik wird für Intervallen von fünf Minuten veröffentlicht. Wenn Sie die bereitges tellten Schreibkapazitätseinheiten schnell anpassen, spiegelt diese Statistik möglicherweise nicht den tatsächlichen Durchschnitt wider.

Metrik	Description
`BillableTableSizeInBytes`	Die fakturierbare Größe der Tabelle in Byte. Dies ist die Summe der codierten Größen aller Zeilen in der Tabelle. Mit dieser Metrik können Sie die Speicherkosten Ihrer Tabelle im Laufe der Zeit verfolgen. Einheiten: `Bytes` Maße: `Keyspace, TableName` Gültige Statistiken: • `Maximum`— Die maximale Speichergröße der Tabelle. • `Minimum`— Die minimale Speichergröße der Tabelle. • `Average`— Die durchschnittliche Speichergröße der Tabelle. Diese Metrik wird in Intervallen von 4 bis 6 Stunden berechnet.

Metrik	Description
`ConditionalCheckFailedRequests`	Die Anzahl der fehlgeschlagenen LWT-Schreibanforderungen (Lightweight Transaction). Mit den `INSERT`-, `UPDATE`- und `DELETE`-Operationen können Sie eine logische Bedingung angeben, die als wahr ausgewertet werden muss, bevor die Operation fortgesetzt werden kann. Wenn diese Bedingung als falsch ausgewertet `ConditionalCheckFailedRequests` wird, wird sie um eins erhöht. Bedingungsprüfungen, die den Wert falsch ergeben, verbrauchen Schreibkapazitätseinheiten auf der Grundlage der Zeilengröße. Weitere Informationen finden Sie unter the section called “Schätzen Sie den Kapazitätsverbrauch von LWT”. Einheiten: `Count` Maße: `Keyspace, TableName` Gültige Statistiken: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `Sum`

Metrik	Description
`ConsumedReadCapacityUnits`	Die Anzahl der im angegebenen Zeitraum verbrauchten Lesekapazitätseinheiten. Weitere Informationen finden Sie unter Read/Write Kapazitätsmodus. **Note** Um Ihre durchschnittliche Durchsatzauslastung pro Sekunde zu verstehen, verwenden Sie die `Sum` Statistik, um den verbrauchten Durchsatz für den Zeitraum von einer Minute zu berechnen . Teilen Sie dann die Summe durch die Anzahl der Sekunden in einer Minute (60), um den Durchschnitt `ConsumedReadCapacityUnits` pro Sekunde zu berechnen (wobei zu berücksic htigen ist, dass dieser Durchschnitt keine großen, sondern nur kurze Spitzen der Leseaktivität hervorhebt, die während dieser Minute aufgetret en sind). Weitere Informationen zum Vergleich der durchschnittlich verbrauchten Lesekapaz ität mit der bereitgestellten Lesekapazität finden Sie unter the section called “Verwenden von - Metriken” Einheiten: `Count` Maße: `Keyspace, TableName` Gültige Statistiken: • `Minimum`— Die Mindestanzahl an Lesekapazitätseinh eiten, die von einer einzelnen Anforderung an die Tabelle verbraucht werden. • `Maximum`— Die maximale Anzahl von Lesekapaz itätseinheiten, die von einer einzelnen Anfrage an die Tabelle verbraucht werden.

Metrik	Description
	• `Average` – Die durchschnittliche Lesekapazität pro Anforderung. **Note** Der `Average`-Wert wird von Perioden der Inaktivität beeinflusst, in denen der Stichprob enwert Null ist. • `Sum` – Die verbrauchten Einheiten der gesamten Lesekapazität. Die nützlichste Statistik für diese Metrik ist `ConsumedReadCapacityUnits` . • `SampleCount` — Die Anzahl der Anfragen an Amazon Keyspaces, auch wenn keine Lesekapazität verbraucht wurde. **Note** Der `SampleCount` -Wert wird von Perioden der Inaktivität beeinflusst, in denen der Stichprobenwert Null ist.

Metrik	Description
`ConsumedWriteCapacityUnits`	Die Anzahl der Schreibkapazitätseinheiten, die im angegebenen Zeitraum verbraucht wurden. Sie können die gesamte verbrauchte Schreibkapazität für eine Tabelle abrufen. Weitere Informationen finden Sie unter Read/Write Kapazitätsmodus. **Note** Um Ihre durchschnittliche Durchsatzauslastung pro Sekunde zu verstehen, verwenden Sie die `Sum` Statistik, um den verbrauchten Durchsatz für den Zeitraum von einer Minute zu berechnen. Teilen Sie dann die Summe durch die Anzahl der Sekunden in einer Minute (60), um den Durchschnitt `ConsumedWriteCapacityUnits` pro Sekunde zu berechnen (wobei zu berücksichtigen ist, dass dieser Durchschnitt keine großen, sondern nur kurze Spitzen der Schreibaktivität hervorhebt, die während dieser Minute aufgetreten sind). Weitere Informationen zum Vergleich der durchschnittlich verbrauchten Schreibkapazität mit der bereitgestellten Schreibkapazität finden Sie unter the section called “Verwenden von -Metriken” Einheiten: `Count` Maße: `Keyspace, TableName` Gültige Statistiken: • `Minimum`— Die Mindestanzahl von Schreibkapazitätseinheiten, die von einer einzelnen Anforderung an die Tabelle verbraucht werden.

Metrik	Description
	• `Maximum`— Die maximale Anzahl von Schreibka pazitätseinheiten, die von einer einzelnen Anfrage an die Tabelle verbraucht werden. • `Average` – Die durchschnittliche Schreibkapazität pro Anforderung. **Note** Der `Average`-Wert wird von Perioden der Inaktivität beeinflusst, in denen der Stichprob enwert Null ist. • `Sum` – Die verbrauchten Schreibkapazitätseinheiten. Die nützlichste Statistik für diese Metrik ist `ConsumedW riteCapacityUnits` . • `SampleCount` — Die Anzahl der Anfragen an Amazon Keyspaces, auch wenn keine Schreibkapazität verbraucht wurde. **Note** Der `SampleCount` -Wert wird von Perioden der Inaktivität beeinflusst, in denen der Stichprobenwert Null ist.

Metrik	Description
`MaxProvisionedTableReadCapacityUtilization`	Der Prozentsatz der bereitgestellten Lesekapazitätseinheiten, der von der Tabelle mit der höchsten bereitgestellten Lesekapazität eines Kontos genutzt wird. Einheiten: `Percent` Gültige Statistiken: • `Maximum` – : Der maximale Prozentsatz der bereitgestellten Lesekapazitätseinheiten, die von der höchsten bereitgestellten Lesetabelle des Kontos genutzt werden. • `Minimum` – Der Mindestprozentsatz der bereitgestellten Lesekapazitätseinheiten, die von der höchsten bereitgestellten Lesetabelle des Kontos genutzt werden. • `Average` – Der durchschnittliche Prozentsatz der bereitgestellten Lesekapazitätseinheiten, die von der höchsten bereitgestellten Lesetabelle des Kontos genutzt werden. Die Metrik wird für Intervallen von fünf Minuten veröffentlicht. Wenn Sie die bereitgestellten Lesekapazitätseinheiten schnell anpassen, spiegelt diese Statistik möglicherweise nicht den tatsächlichen Durchschnitt wider.

Metrik	Description
`MaxProvisionedTableWriteCapacityUtilization`	Der Prozentsatz der bereitgestellten Schreibkapazität, der von der Tabelle mit der höchsten bereitgestellten Schreibkapazität eines Kontos genutzt wird. Einheiten: `Percent` Gültige Statistiken: • `Maximum`— Der maximale Prozentsatz der bereitgestellten Schreibkapazitätseinheiten, der von der Tabelle mit der höchsten bereitgestellten Schreibkapazität des Kontos genutzt wird. • `Minimum`— Der Mindestprozentsatz der bereitgestellten Schreibkapazitätseinheiten, der von der Tabelle mit der höchsten bereitgestellten Schreibkapazität des Kontos genutzt wird. • `Average`— Der durchschnittliche Prozentsatz der bereitgestellten Schreibkapazitätseinheiten, der von der Tabelle mit der höchsten bereitgestellten Schreibkapazität des Kontos genutzt wird. Die Metrik wird für Intervallen von fünf Minuten veröffentlicht. Wenn Sie die bereitgestellten Schreibkapazitätseinheiten schnell anpassen, spiegelt diese Statistik möglicherweise nicht den tatsächlichen Durchschnitt wider.

Metrik	Description
`PerConnectionRequestRateExceeded`	Anfragen an Amazon Keyspaces, die das Kontingent für die Anforderungsrate pro Verbindung überschreiten. Jede Clientverbindung mit Amazon Keyspaces kann bis zu 3 000 CQL-Anfragen pro Sekunde unterstützen. Kunden können mehrere Verbindungen herstellen, um den Durchsatz zu erhöhen. Wenn Sie die Replikation mit mehreren Regionen verwenden, trägt jeder replizierte Schreibvorgang ebenfalls zu diesem Kontingent bei. Als bewährte Methode empfehlen wir, die Anzahl der Verbindungen zu Ihren Tabellen zu erhöhen, um Fehler zu vermeiden `PerConnectionRequestRateExceeded`. Die Anzahl der Verbindungen, die Sie in Amazon Keyspaces haben können, ist unbegrenzt. Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `SampleCount` • `Sum`

Metrik	Description
`ProvisionedReadCapacityUnits`	Die Anzahl der bereitgestellten Lesekapazitätseinheiten für eine Tabelle. Die `TableName` Dimension gibt den `ProvisionedReadCapacityUnits` für die Tabelle zurück. Einheiten: `Count` Maße: `Keyspace, TableName` Gültige Statistiken: • `Minimum` – Die niedrigste Einstellung für bereitgestellte Lesekapazität. Wenn Sie `ALTER TABLE` verwenden, um die Lesekapazität zu erhöhen, zeigt diese Metrik den niedrigsten Wert der bereitgestellten `ReadCapacityUnits` während dieses Zeitraums an. • `Maximum` – Die höchste Einstellung für bereitgestellte Lesekapazität. Wenn Sie `ALTER TABLE` verwenden, um die Lesekapazität zu verringern, zeigt diese Metrik den höchsten Wert der bereitgestellten `ReadCapacityUnits` während dieses Zeitraums an. • `Average` – Die durchschnittliche bereitgestellte Lesekapazität. Die `ProvisionedReadCapacityUnits` -Metrik wird in Intervallen von fünf Minuten veröffentlicht. Wenn Sie die bereitgestellten Lesekapazitätseinheiten schnell anpassen, spiegelt diese Statistik möglicherweise nicht den tatsächlichen Durchschnitt wider.

Metrik	Description
`ProvisionedWriteCapacityUnits`	Die Anzahl der bereitgestellten Schreibkapazitätseinheiten für eine Tabelle. Die `TableName` Dimension gibt den `ProvisionedWriteCapacityUnits` für die Tabelle zurück. Einheiten: `Count` Maße: `Keyspace, TableName` Gültige Statistiken: • `Minimum` – Die niedrigste Einstellung für bereitgestellte Schreibkapazität. Wenn Sie `ALTER TABLE` verwenden, um die Schreibkapazität zu erhöhen, zeigt diese Metrik den niedrigsten Wert der bereitgestellten `WriteCapacityUnits` während dieses Zeitraums an. • `Maximum` – Die höchste Einstellung für bereitgestellte Schreibkapazität. Wenn Sie `ALTER TABLE` verwenden, um die Schreibkapazität zu verringern, zeigt diese Metrik den höchsten Wert der bereitgestellten `WriteCapacityUnits` während dieses Zeitraums an. • `Average` – Die durchschnittliche bereitgestellte Schreibkapazität. Die `ProvisionedWriteCapacityUnits` -Metrik wird in Intervallen von fünf Minuten veröffentlicht. Wenn Sie die bereitgestellten Schreibkapazitätseinheiten schnell anpassen, spiegelt diese Statistik möglicherweise nicht den tatsächlichen Durchschnitt wider.

Metrik	Description
`ReadThrottleEvents`	Anfragen an Amazon Keyspaces, die die bereitgestellte Lesekapazität für eine Tabelle oder Kontingente auf Kontoebene, Quoten pro Anfrage pro Verbindung oder Kontingente auf Partitionsebene überschreiten. Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `SampleCount` • `Sum`
`ReplicationLatency`	Diese Metrik gilt nur für Schlüsselräume mit mehreren Regionen und misst die Zeit`updates`, `inserts` die für die Replikation oder `deletes` von einer Replikattabelle zu einer anderen Replikattabelle in einem Schlüsselraum mit mehreren Regionen benötigt wurde. Einheiten: `Millisecond` Maße: `TableName, ReceivingRegion` Gültige Statistiken: • `Average` • `Maximum` • `Minimum`

Metrik	Description
ReturnedItemCountBySelect	Die Anzahl der Zeilen, die von SELECT mehrzeiligen Abfragen während des angegebenen Zeitraums zurückgegeben wurden. Multi-row SELECTAbfragen sind Abfragen, die nicht den vollqualifizierten Primärschlüssel enthalten, wie z. B. vollständige Tabellenscans und Bereichsabfragen. Die Anzahl der zurückgegebenen Zeilen entspricht nicht unbedingt der Anzahl der Zeilen, die ausgewertet wurden. Nehmen wir zum Beispiel an, Sie haben für eine Tabelle SELECT * mit ALLOW FILTERING 100 Zeilen ein mit angefordert, aber eine WHERE Klausel angegeben, die die Ergebnisse so einschränkt, dass nur 15 Zeilen zurückgegeben werden. In diesem Fall SELECT würde die Antwort von a 100 und a ScanCount Count von 15 zurückgegebenen Zeilen enthalten. Einheiten: Count Maße: Keyspace, TableName, Operation Gültige Statistiken: • Minimum • Maximum • Average • SampleCount • Sum

Metrik	Description
`StoragePartitionThroughputCapacityExceeded`	Anfragen an eine Amazon Keyspaces-Speicherpartition, die die Durchsatzkapazität der Partition überschreiten. Amazon Keyspaces-Speicherpartitionen können bis zu 1000 WCU/WRU pro Sekunde und 3000 RCU/RRU pro Sekunde unterstützen. Wir empfehlen, Ihr Datenmodell zu überprüfen, um den read/write Datenverkehr auf mehrere Partitionen zu verteilen, um diese Ausnahmen zu vermeiden. **Note** Logische Amazon Keyspaces-Partitionen können sich über mehrere Speicherpartitionen erstrecken und sind praktisch unbegrenzt groß. Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `SampleCount` • `Sum`
`SuccessfulRequestCount`	Die Anzahl der erfolgreichen Anfragen, die im angegebenen Zeitraum verarbeitet wurden. Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `SampleCount`

Metrik	Description
SuccessfulRequestLatency	Die erfolgreichen Anfragen an Amazon Keyspaces während des angegebenen Zeitraums. SuccessfulRequestLatency kann zwei verschiedene Arten von Informationen bereitstellen: • Die verstrichene Zeit für erfolgreiche Anfragen (Minimum, Maximum, Sum oder Average). • Die Anzahl erfolgreicher Anforderungen (SampleCount). SuccessfulRequestLatency spiegelt nur Aktivitäten innerhalb von Amazon Keyspaces wider und berücksichtigt weder Netzwerklatenz noch clientseitige Aktivitäten. Einheiten: Milliseconds Maße: Keyspace, TableName, Operation Gültige Statistiken: • Minimum • Maximum • Average • SampleCount

Metrik	Description
SystemErrors	Die Anfragen an Amazon Keyspaces, die ServerError während des angegebenen Zeitraums eine generieren. A weist ServerError normalerweise auf einen internen Servicefehler hin. Einheiten: Count Maße: Keyspace, TableName, Operation Gültige Statistiken: • Sum • SampleCount
SystemReconciliationDeletes	Die Einheiten, die zum Löschen von als veraltet eingestuften Daten verwendet werden, wenn clientseitige Zeitstempel aktiviert sind. Jede SystemReconciliationDelete bietet ausreichend Kapazität, um bis zu 1 KB Daten pro Zeile zu löschen oder zu aktualisieren. Um beispielsweise eine Zeile zu aktualisieren, die 2,5 KB an Daten speichert, und um eine oder mehrere Spalten in der Zeile gleichzeitig zu löschen, sind 3 SystemReconciliationDeletes erforderlich. Oder um eine ganze Zeile zu löschen, die 3,5 KB an Daten enthält, sind 4 erforderlichSystemReconciliationDeletes . Einheiten: Count Maße: Keyspace, TableName Gültige Statistiken: • Sum— Die Gesamtzahl der in einem Zeitraum SystemReconciliationDeletes verbrauchten.

Metrik	Description
`TTLDeletes`	Die Einheiten, die zum Löschen oder Aktualisieren von Daten nacheinander mithilfe von Time to Live (TTL) verbraucht wurden. Jede `TTLDelete` bietet ausreiche nd Kapazität, um bis zu 1 KB Daten pro Zeile zu löschen oder zu aktualisieren. Um beispielsweise eine Zeile zu aktualisieren, die 2,5 KB an Daten speichert, und um eine oder mehrere Spalten innerhalb der Zeile gleichzeitig zu löschen, sind 3 TTL-Löschungen erforderlich. Oder um eine ganze Zeile zu löschen, die 3,5 KB an Daten enthält, sind 4 TTL-Löschungen erforderlich. Einheiten: `Count` Maße: `Keyspace, TableName` Gültige Statistiken: • `Sum`— Die Gesamtzahl der in einem Zeitraum `TTLDeletes` verbrauchten.

Metrik	Description
`UserErrors`	Anfragen an Amazon Keyspaces, die während des angegebenen Zeitraums einen `InvalidRequest` Fehler generieren. `InvalidRequest` Fehler stellen normalerweise einen clientseitigen Fehler dar, z. B. eine ungültige Kombinati on von Parametern, einen Versuch, eine nicht existiere nde Tabelle zu aktualisieren, oder eine falsche Anforderu ngssignatur. `UserErrors` stellen die Summe der ungültigen Anfragen für die aktuelle AWS-Region und die aktuelle Anfrage dar. AWS-Konto Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `Sum` • `SampleCount`
`WriteThrottleEvents`	Anfragen an Amazon Keyspaces, die die bereitgestellte Schreibkapazität für eine Tabelle oder Kontingente auf Kontoebene, Quoten pro Anfrage pro Verbindung oder Kontingente auf Partitionsebene überschreiten. Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `SampleCount` • `Sum`

Dimensionen für Amazon Keyspaces-Metriken

Die Metriken für Amazon Keyspaces werden anhand der Werte für das Konto, den Tabellennamen oder den Vorgang qualifiziert. Sie können die CloudWatch Konsole verwenden, um Amazon Keyspaces-Daten für jede der Dimensionen in der folgenden Tabelle abzurufen.

Dimension	Description
`Keyspace`	Diese Dimension beschränkt die Daten auf einen bestimmten Schlüsselraum. Dieser Wert kann ein beliebiger Schlüsselraum in der aktuellen Region und in der aktuellen Region sein. AWS-Konto
`Operation`	Diese Dimension beschränkt die Daten auf eine der Amazon Keyspaces-CQL-Operationen, wie `INSERT` z. B. Oder-Oper ationen. `SELECT`
`TableName`	Diese Dimension schränkt die Daten auf einen spezifische Tabelle ein. Dieser Wert kann ein beliebiger Tabellenname in der aktuellen Region und in der aktuellen Region sein. AWS-Konto Wenn der Tabellenname innerhalb des Kontos nicht eindeutig ist, müssen Sie ihn auch angeben`Keyspace`.

Metriken für Amazon Keyspaces Change Data Capture (CDC)

Die folgenden Metriken sind in Ihrem CloudWatch Konto verfügbar und überwachen API-Aufrufe an den Amazon Keyspaces-Endpunkt zur Erfassung von Änderungsdaten (CDC).

Metrik	Description
`ReturnedRecordsCount`	Die Anzahl der Stream-Datensätze, die während des angegebenen `GetRecords` Zeitraums von Vorgängen zurückgegeben wurden. Einheiten: `Count` Maße: `Keyspace, TableName, Operation`

Metrik	Description
	Gültige Statistiken: • Minimum • Maximum • Average • SampleCount • Sum
SuccessfulRequestLatency	Die erfolgreichen Anfragen an Amazon Keyspaces während des angegebenen Zeitraums. SuccessfulRequestLatency kann zwei verschiedene Arten von Informationen bereitstellen: • Die verstrichene Zeit für erfolgreiche Anfragen (Minimum, Maximum, Sum oder Average). • Die Anzahl erfolgreicher Anforderungen (SampleCount). SuccessfulRequestLatency spiegelt nur Aktivitäten innerhalb der CDC-APIs von Amazon Keyspaces wider und berücksichtigt weder Netzwerklatenz noch clientseitige Aktivitäten. Einheiten: Milliseconds Maße: Keyspace, TableName, Operation Gültige Statistiken: • Minimum • Maximum • Average • SampleCount

Metrik	Description
`SystemErrors`	Die Anfragen an Amazon Keyspaces CDC, die während des angegebenen Zeitraums einen `HTTP 500` Statuscode generieren. Ein `HTTP 500` Statuscode weist normalerweise auf einen internen Servicefehler hin. Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `Sum` • `SampleCount`
`UserErrors`	Anfragen an Amazon Keyspaces CDC, die während des angegebenen Zeitraums einen `HTTP 400` Statuscode generieren. `UserErrors` stellt die Summe der `HTTP 400` Fehler für Amazon Keyspaces CDC in der aktuellen AWS-Region und der aktuellen Version dar. AWS-Konto Einheiten: `Count` Maße: `Keyspace, TableName, Operation` Gültige Statistiken: • `Sum` • `SampleCount`

Dimensionen für Amazon Keyspaces-Metriken zur Datenerfassung (CDC)

Die Metriken für Amazon Keyspaces werden anhand der Werte für das Konto, den Tabellennamen oder den Vorgang qualifiziert. Sie können die CloudWatch Konsole verwenden, um Amazon Keyspaces-Daten für jede der Dimensionen in der folgenden Tabelle abzurufen.

Dimension	Description
`StreamLabel`	Diese Dimension beschränkt die Daten auf einen bestimmten CDC-Stream (Change Data Capture), sofern `KeyspaceName` und `TableName` sofern angegeben.
`Keyspace`	Diese Dimension beschränkt die Daten auf einen bestimmten Schlüsselraum. Dieser Wert kann ein beliebiger Schlüsselraum in der aktuellen Region und in der aktuellen Region sein. AWS-Konto
`TableName`	Diese Dimension schränkt die Daten auf einen spezifische Tabelle ein. Dieser Wert kann ein beliebiger Tabellenname in der aktuellen Region und der aktuellen AWS-Konto Region sein. Wenn der Tabellenname innerhalb des Kontos nicht eindeutig ist, müssen Sie ihn auch angeben`Keyspace`.
`Operation`	Diese Dimension beschränkt die Daten auf eine der folgenden Amazon Keyspaces Streams API-Operationen: `GetRecords` `GetStream` `GetShardIterator` , und`ListStreams` .

Amazon Keyspaces-Metriken anzeigen in CloudWatch

Wenn Sie mit Amazon Keyspaces interagieren, werden die folgenden Metriken und Dimensionen an Amazon CloudWatch gesendet. Alle Metriken werden aggregiert und jede Minute gemeldet. Sie können die folgenden Verfahren verwenden, um die Metriken für Amazon Keyspaces anzuzeigen.

Um Metriken mit der CloudWatch Konsole anzuzeigen

Metriken werden zunächst nach dem Service-Namespace und anschließend nach den verschiedenen Dimensionskombinationen in den einzelnen Namespaces gruppiert.

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Ändern Sie, falls erforderlich, die AWS-Region. Wählen Sie in der Navigationsleiste die Region aus, in der sich Ihre AWS Ressourcen befinden. Weitere Informationen finden Sie unter AWS - Service-Endpunkte.
3. Wählen Sie im Navigationsbereich Metriken aus.

4. Wählen Sie unter dem Tab Alle Metriken `AWS/Cassandra`.

So zeigen Sie Metriken mit der AWS CLI an

- Geben Sie als Eingabeaufforderung den folgenden Befehl ein.

```
aws cloudwatch list-metrics --namespace "AWS/Cassandra"
```

CloudWatch Alarme zur Überwachung von Amazon Keyspaces erstellen

Sie können einen CloudWatch Amazon-Alarm für Amazon Keyspaces erstellen, der eine Amazon Simple Notification Service (Amazon SNS) -Nachricht sendet, wenn sich der Status des Alarms ändert. Ein Alarm überwacht eine Metrik über einen bestimmten, von Ihnen definierten Zeitraum. Der Alarm führt eine oder mehrere Aktionen durch, basierend auf dem Wert der Metrik im Vergleich zu einem bestimmten Schwellenwert in einer Reihe von Zeiträumen. Die Aktion ist eine Benachrichtigung, die an ein Amazon SNS SNS-Thema oder eine Application Auto Scaling Scaling-Richtlinie gesendet wird.

Wenn Sie Amazon Keyspaces im Bereitstellungsmodus mit Application Auto Scaling verwenden, erstellt der Service in Ihrem Namen zwei CloudWatch Alarmpaare. Jedes Paar steht für Ihre Ober- und Untergrenzen für bereitgestellte und verbrauchte Durchsatzeinstellungen. Diese CloudWatch Alarme werden ausgelöst, wenn die tatsächliche Auslastung der Tabelle über einen längeren Zeitraum von Ihrer Zielauslastung abweicht. Weitere Informationen zu CloudWatch Alarmen, die von Application Auto Scaling erstellt wurden, finden Sie unterthe section called “So funktioniert die automatische Skalierung von Amazon Keyspaces”.

Alarme rufen nur Aktionen für anhaltende Statusänderungen auf. CloudWatch Alarme lösen keine Aktionen aus, nur weil sie sich in einem bestimmten Zustand befinden. Der Status muss sich geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein.

Weitere Informationen zum Erstellen von CloudWatch Alarmen finden Sie unter Verwenden von CloudWatch Amazon-Alarmen im CloudWatch Amazon-Benutzerhandbuch.

Protokollieren von Amazon Keyspaces-API-Aufrufen mit AWS CloudTrail

Amazon Keyspaces ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS Dienstes in Amazon Keyspaces bereitstellt. CloudTrail erfasst Data Definition Language (DDL) -API-Aufrufe und Data Manipulation Language (DML) -API-Aufrufe für Amazon Keyspaces als Ereignisse. Zu den aufgezeichneten Aufrufen gehören Aufrufe von der Amazon Keyspaces-Konsole und programmatische Aufrufe der Amazon Keyspaces-API-Operationen.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon Simple Storage Service (Amazon S3) -Bucket aktivieren, einschließlich Ereignissen für Amazon Keyspaces.

Wenn Sie keinen Trail konfigurieren, können Sie die zuletzt unterstützten Ereignisse trotzdem auf der CloudTrail Konsole im Event-Verlauf einsehen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon Keyspaces gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

Themen

- Konfiguration von Amazon Keyspaces-Protokolldateieinträgen in CloudTrail
- Informationen zur Amazon Keyspaces-Datendefinitionssprache (DDL) in CloudTrail
- Informationen zur Amazon Keyspaces Data Manipulation Language (DML) in CloudTrail
- Grundlegendes zu Amazon Keyspaces-Protokolldateieinträgen

Konfiguration von Amazon Keyspaces-Protokolldateieinträgen in CloudTrail

Jede angemeldete Amazon Keyspaces-API-Aktion CloudTrail enthält Anforderungsparameter, die in der CQL-Abfragesprache ausgedrückt werden. Weitere Informationen hierzu finden Sie unter CQL-Sprachreferenz.

Sie können in Ihrem AWS-Konto die neusten Ereignisse anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, einschließlich Veranstaltungen für Amazon Keyspaces, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole erstellen, gilt der Trail standardmäßig für alle AWS Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren.

Weitere Informationen finden Sie in folgenden Themen im AWS CloudTrail -Benutzerhandbuch:

- Übersicht zum Erstellen eines Trails
- CloudTrail unterstützte Dienste und Integrationen
- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen
- Empfangen von CloudTrail Protokolldateien von mehreren Konten

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Root- oder AWS Identity and Access Management (IAM-) Benutzeranmeldedaten gestellt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter CloudTrail -Element userIdentity.

Informationen zur Amazon Keyspaces-Datendefinitionssprache (DDL) in CloudTrail

CloudTrail ist auf Ihrem aktiviert AWS-Konto , wenn Sie das Konto erstellen. Wenn eine DDL-Aktivität in Amazon Keyspaces auftritt, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen im CloudTrail Eventverlauf automatisch als Ereignis aufgezeichnet. Die folgende Tabelle zeigt die DDL-Anweisungen, die für Amazon Keyspaces protokolliert werden.

CloudTrail **`eventName`**	Statement	CQL-Aktion	AWS SDK-Aktion
CreateKeyspace	DDL	`CREATE KEYSPACE`	`CreateKeyspace`
AlterKeyspace	DDL	`ALTER KEYSPACE`	`UpdateKeyspace`
DropKeyspace	DDL	`DROP KEYSPACE`	`DeleteKeyspace`
CreateTable	DDL	`CREATE TABLE`	`CreateTable`
DropTable	DDL	`DROP TABLE`	`DeleteTable`
AlterTable	DDL	`ALTER TABLE`	`UpdateTable` , `TagResource` , `UntagResource`
CreateUdt	DDL	`CREATE TYPE`	`CreateType`
DropUdt	DDL	`DROP TYPE`	`DeleteType`
GetStream	DDL	keine CQL-Aktion verfügbar	`GetStream`
ListStreams	DDL	keine CQL-Aktion verfügbar	`ListStreams`

Informationen zur Amazon Keyspaces Data Manipulation Language (DML) in CloudTrail

Um die Protokollierung von Amazon Keyspaces-DML-Anweisungen mit zu aktivieren CloudTrail, müssen Sie zuerst die Protokollierung der API-Aktivität auf der Datenebene aktivieren. CloudTrail Sie können mit der Protokollierung von Amazon Keyspaces-DML-Ereignissen in neuen oder bestehenden Trails beginnen, indem Sie festlegen, dass Aktivitäten für den Datenereignistyp Cassandra-Tabelle über die CloudTrail Konsole protokolliert werden sollen, oder indem `AWS::Cassandra::Table` Sie den `resources.type` Wert auf AWS CLI- oder CloudTrail API-Operationen setzen. Weitere Informationen finden Sie unter Protokollieren von Datenereignissen.

Weitere Informationen und ein Beispiel, das zeigt, wie Alarme für Datenereignisse erstellt werden, finden Sie im folgenden Beitrag im AWS Datenbank-Blog Using DML Auditing for Amazon Keyspaces (für Apache Cassandra).

Die folgende Tabelle zeigt die Datenereignisse, die von for protokolliert werden. CloudTrail `Cassandra table`

CloudTrail **`eventName`**	Statement	CQL-Aktion	AWS SDK-Aktionen
Select	DML	`SELECT`	`GetKeyspace` , `GetTable`, `GetType`, `ListKeyspaces` , `ListTables` , `ListTypes` , `ListTagsForResource`
Einfügen	DML	`INSERT`	keine AWS SDK-Aktionen verfügbar
Aktualisierung	DML	`UPDATE`	keine AWS SDK-Aktionen verfügbar
Delete	DML	`DELETE`	keine AWS SDK-Aktionen verfügbar
`GetShardIterator`	DML	keine CQL-Aktion verfügbar	`GetShardIterator`
`GetRecords`	DML	keine CQL-Aktion verfügbar	`GetRecords`

Grundlegendes zu Amazon Keyspaces-Protokolldateieinträgen

CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion, Anforderungsparameter usw. CloudTrail

Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `DropTable` Aktionen`CreateKeyspace`, `DropKeyspaceCreateTable`, und demonstriert:


```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
        "accountId": "111122223333",
        "sessionContext": {
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::111122223333:role/Admin",
            "accountId": "111122223333",
            "userName": "Admin"
          },
          "webIdFederationData": {},
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
          }
        }
      },
      "eventTime": "2020-01-15T18:53:04Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "CreateKeyspace",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "\n\tCREATE KEYSPACE \"mykeyspace\"\n\tWITH\n\t\tREPLICATION =
 {'class': 'SingleRegionStrategy'}\n\t\t",
        "keyspaceName": "mykeyspace"
      },
      "responseElements": null,
      "requestID": "bfa3e75d-bf4d-4fc0-be5e-89d15850eb41",
```

```
    "eventID": "d25beae8-f611-4229-877a-921557a07bb9",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::Cassandra::Keyspace",
        "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
      }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "recipientAccountId": "111122223333",
    "managementEvent": true,
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
  },
  {
    "eventVersion": "1.05",
    "userIdentity": {
      "type": "AssumedRole",
      "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
      "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
      "accountId": "111122223333",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "AKIAIOSFODNN7EXAMPLE",
          "arn": "arn:aws:iam::111122223333:role/Admin",
          "accountId": "111122223333",
          "userName": "Admin"
        },
        "webIdFederationData": {},
        "attributes": {
          "mfaAuthenticated": "false",
          "creationDate": "2020-01-15T18:47:56Z"
        }
      }
    },
    "eventTime": "2020-01-15T19:28:39Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "DropKeyspace",
```

```
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
      "rawQuery": "DROP KEYSPACE \"mykeyspace\"",
      "keyspaceName": "mykeyspace"
    },
    "responseElements": null,
    "requestID": "66f3d86a-56ae-4c29-b46f-abcd489ed86b",
    "eventID": "e5aebeac-e1dd-41e3-a515-84fe6aaabd7b",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::Cassandra::Keyspace",
        "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
      }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "recipientAccountId": "111122223333",
    "managementEvent": true,
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
  },
  {
    "eventVersion": "1.05",
    "userIdentity": {
      "type": "AssumedRole",
      "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
      "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
      "accountId": "111122223333",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "AKIAIOSFODNN7EXAMPLE",
          "arn": "arn:aws:iam::111122223333:role/Admin",
          "accountId": "111122223333",
          "userName": "Admin"
        },
        "webIdFederationData": {},
```

```
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
          }
        }
      },
      "eventTime": "2020-01-15T18:55:24Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "CreateTable",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "\n\tCREATE TABLE \"mykeyspace\".\"mytable\"(\n\t\t\"ID\" int,
\n\t\t\"username\" text,\n\t\t\"email\" text,\n\t\t\"post_type\" text,\n\t\tPRIMARY
 KEY((\"ID\", \"username\", \"email\")))",
        "keyspaceName": "mykeyspace",
        "tableName": "mytable"
      },
      "responseElements": null,
      "requestID": "5f845963-70ea-4988-8a7a-2e66d061aacb",
      "eventID": "fe0dbd2b-7b34-4675-a30c-740f9d8d73f9",
      "readOnly": false,
      "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Table",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable"
        }
      ],
      "eventType": "AwsApiCall",
      "apiVersion": "3.4.4",
      "recipientAccountId": "111122223333",
      "managementEvent": true,
      "eventCategory": "Management",
      "tlsDetails": {
          "tlsVersion": "TLSv1.2",
          "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
          "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    },
    {
      "eventVersion": "1.05",
      "userIdentity": {
```

```
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
        "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
        "accountId": "111122223333",
        "sessionContext": {
          "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::111122223333:role/Admin",
            "accountId": "111122223333",
            "userName": "Admin"
          },
          "webIdFederationData": {},
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2020-01-15T18:47:56Z"
          }
        }
      },
      "eventTime": "2020-01-15T19:27:59Z",
      "eventSource": "cassandra.amazonaws.com",
      "eventName": "DropTable",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "10.24.34.01",
      "userAgent": "Cassandra Client/ProtocolV4",
      "requestParameters": {
        "rawQuery": "DROP TABLE \"mykeyspace\".\"mytable\"",
        "keyspaceName": "mykeyspace",
        "tableName": "mytable"
      },
      "responseElements": null,
      "requestID": "025501b0-3582-437e-9d18-8939e9ef262f",
      "eventID": "1a5cbedc-4e38-4889-8475-3eab98de0ffd",
      "readOnly": false,
      "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Table",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable"
        }
      ],
      "eventType": "AwsApiCall",
      "apiVersion": "3.4.4",
```

```
      "recipientAccountId": "111122223333",
      "managementEvent": true,
      "eventCategory": "Management",
      "tlsDetails": {
          "tlsVersion": "TLSv1.2",
          "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
          "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
  ]
}
```

Die folgende Protokolldatei zeigt ein Beispiel für eine SELECT Anweisung.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/alice",
        "accountId": "111122223333",
        "userName": "alice"
    },
    "eventTime": "2023-11-17T10:38:04Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Select",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
          "keyspaceName": "my_keyspace",
          "tableName": "my_table",
          "conditions": [
              "pk = **(Redacted)",
              "ck < 3**(Redacted)0",
              "region = 't**(Redacted)t'"
          ],
          "select": [
              "pk",
              "ck",
              "region"
          ],
          "allowFiltering": true
    },
```

```
    "responseElements": null,
    "requestID": "6d83bbf0-a3d0-4d49-b1d9-e31779a28628",
    "eventID": "e00552d3-34e9-4092-931a-912c4e08ba17",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::Cassandra::Table",
            "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table"
        }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
}
```

Die folgende Protokolldatei zeigt ein Beispiel für eine INSERT Anweisung.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/alice",
        "accountId": "111122223333",
        "userName": "alice"
    },
    "eventTime": "2023-12-01T22:11:43Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Insert",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
          "keyspaceName": "my_keyspace",
```

```
            "tableName": "my_table",
            "primaryKeys": {
                "pk": "**(Redacted)",
                "ck": "1**(Redacted)8"
            },
            "columnNames": [
                "pk",
                "ck",
                "region"
            ],
            "updateParameters": {
                "TTL": "2**(Redacted)0"
            }
        }
    },
    "responseElements": null,
    "requestID": "edf8af47-2f87-4432-864d-a960ac35e471",
    "eventID": "81b56a1c-9bdd-4c92-bb8e-92776b5a3bf1",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::Cassandra::Table",
            "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/
my_table"
        }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
}
```

Die folgende Protokolldatei zeigt ein Beispiel für eine UPDATE Anweisung.

```
{
    "eventVersion": "1.09",
```

```
    "userIdentity": {
          "type": "IAMUser",
          "principalId": "AKIAIOSFODNN7EXAMPLE",
          "arn": "arn:aws:iam::111122223333:user/alice",
          "accountId": "111122223333",
          "userName": "alice"
    },
    "eventTime": "2023-12-01T22:11:43Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Update",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
          "keyspaceName": "my_keyspace",
          "tableName": "my_table",
          "primaryKeys": {
              "pk": "'t**(Redacted)t'",
              "ck": "'s**(Redacted)g'"
          },
          "assignmentColumnNames": [
             "nonkey"
          ],
          "conditions": [
              "nonkey < 1**(Redacted)7"
          ]
    },
    "responseElements": null,
    "requestID": "edf8af47-2f87-4432-864d-a960ac35e471",
    "eventID": "81b56a1c-9bdd-4c92-bb8e-92776b5a3bf1",
    "readOnly": false,
    "resources": [
        {
          "accountId": "111122223333",
          "type": "AWS::Cassandra::Table",
          "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/
my_table"
        }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
```

```
    "tlsDetails": {
          "tlsVersion": "TLSv1.3",
          "cipherSuite": "TLS_AES_128_GCM_SHA256",
          "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
}
```

Die folgende Protokolldatei zeigt ein Beispiel für eine DELETE Anweisung.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/alice",
        "accountId": "111122223333",
        "userName": "alice",
    },
    "eventTime": "2023-10-23T13:59:05Z",
    "eventSource": "cassandra.amazonaws.com",
    "eventName": "Delete",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "10.24.34.01",
    "userAgent": "Cassandra Client/ProtocolV4",
    "requestParameters": {
        "keyspaceName": "my_keyspace",
        "tableName": "my_table",
        "primaryKeys": {
            "pk": "**(Redacted)",
            "ck": "**(Redacted)"
        },
        "conditions": [],
        "deleteColumnNames": [
            "m",
            "s"
        ],
        "updateParameters": {}
    },
    "responseElements": null,
    "requestID": "3d45e63b-c0c8-48e2-bc64-31afc5b4f49d",
    "eventID": "499da055-c642-4762-8775-d91757f06512",
    "readOnly": false,
    "resources": [
```

```
      {
        "accountId": "111122223333",
        "type": "AWS::Cassandra::Table",
        "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/
my_table"
      }
    ],
    "eventType": "AwsApiCall",
    "apiVersion": "3.4.4",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
    }
}
```

Sicherheit in Amazon Keyspaces (für Apache Cassandra)

Cloud-Sicherheit hat AWS höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, auf der AWS Dienste in der ausgeführt AWS Cloud werden. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Die Wirksamkeit unserer Sicherheitsfunktionen wird regelmäßig von externen Prüfern im Rahmen des AWS -Compliance-Programms getestet und überprüft. Weitere Informationen zu den Compliance-Programmen, die für Amazon Keyspaces gelten, finden Sie unter AWS Services im Umfang nach Compliance-Programmen.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. In Ihre Verantwortung fallen außerdem weitere Faktoren, wie z. B. die Vertraulichkeit der Daten, die Anforderungen Ihrer Organisation sowie geltende Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von Amazon Keyspaces anwenden können. In den folgenden Themen erfahren Sie, wie Sie Amazon Keyspaces konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, die Ihnen helfen können, Ihre Amazon Keyspaces-Ressourcen zu überwachen und zu sichern.

Topics

- Datenschutz in Amazon Keyspaces
- AWS Identity and Access Management für Amazon Keyspaces
- Konformitätsüberprüfung für Amazon Keyspaces (für Apache Cassandra)
- Resilienz und Notfallwiederherstellung in Amazon Keyspaces
- Infrastruktursicherheit in Amazon Keyspaces
- Konfiguration und Schwachstellenanalyse für Amazon Keyspaces
- Bewährte Sicherheitsmethoden für Amazon Keyspaces

Datenschutz in Amazon Keyspaces

Das AWS Modell der gilt für den Datenschutz in Amazon Keyspaces (für Apache Cassandra). Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der alle Systeme laufen. AWS Cloud Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz . Informationen zum Datenschutz in Europa finden Sie im General Data Protection Regulation (GDPR) Center.

Aus Datenschutzgründen empfehlen wir, dass Sie Ihre AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Wird verwendet SSL/TLS , um mit AWS Ressourcen zu kommunizieren. Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein AWS CloudTrail. Informationen zur Verwendung von CloudTrail Pfaden zur Erfassung von AWS Aktivitäten finden Sie unter Arbeiten mit CloudTrail Pfaden im AWS CloudTrail Benutzerhandbuch.
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-3-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-3.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit Amazon Keyspaces oder anderen AWS-Services über die Konsole AWS CLI, API oder AWS SDKs arbeiten. Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen

externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Themen

- Verschlüsselung im Ruhezustand in Amazon Keyspaces
- Verschlüsselung bei der Übertragung in Amazon Keyspaces
- Datenschutz im Netzwerkverkehr in Amazon Keyspaces

Verschlüsselung im Ruhezustand in Amazon Keyspaces

Die Verschlüsselung im Ruhezustand von Amazon Keyspaces (für Apache Cassandra) bietet erhöhte Sicherheit, indem alle Ihre Daten im Ruhezustand mit Verschlüsselungsschlüsseln verschlüsselt werden, die in AWS Key Management Service () gespeichert sind.AWS KMS Diese Funktionalität trägt zur Verringerung des Betriebsaufwands und der Komplexität bei, die mit dem Schutz sensibler Daten einhergeht. Mit Encryption at Rest können Sie sicherheitsrelevante Anwendungen entwickeln, die strenge Compliance- und behördliche Anforderungen an den Datenschutz erfüllen.

Die Amazon Keyspaces-Verschlüsselung im Ruhezustand verschlüsselt Ihre Daten mit dem 256-Bit-Advanced Encryption Standard (). AES-256 Dies trägt dazu bei, Ihre Daten vor unbefugtem Zugriff auf den zugrunde liegenden Speicher zu schützen.

Amazon Keyspaces verschlüsselt und entschlüsselt die Daten in Tabellen und Streams transparent. Amazon Keyspaces verwendet Umschlagverschlüsselung und eine Schlüsselhierarchie, um Datenverschlüsselungsschlüssel zu schützen. Es lässt sich AWS KMS zum Speichern und Verwalten des Root-Verschlüsselungsschlüssels integrieren. Weitere Informationen zur Hierarchie der Verschlüsselungsschlüssel finden Sie unterthe section called “Funktionsweise”. Weitere Informationen zu AWS KMS Konzepten wie der Envelope-Verschlüsselung finden Sie unter AWS KMS Management Service Concepts im AWS Key Management Service Developer Guide.

Wenn Sie eine neue Tabelle erstellen, können Sie einen der folgenden AWS KMS Schlüssel (KMS-Schlüssel) wählen:

- AWS-eigener Schlüssel — Dies ist der Standardverschlüsselungstyp. Der Schlüssel gehört Amazon Keyspaces (ohne zusätzliche Kosten).
- Vom Kunden verwalteter Schlüssel — Dieser Schlüssel wird in Ihrem Konto gespeichert und wird von Ihnen erstellt, gehört und verwaltet. Sie haben die volle Kontrolle über den vom Kunden verwalteten Schlüssel (es AWS KMS fallen Gebühren an).

Amazon Keyspaces verschlüsselt automatisch Change Data Capture (CDC) -Streams mit demselben Schlüssel wie die zugrunde liegende Tabelle. Weitere Informationen zu CDC finden Sie unter. the section called “Arbeiten mit CDC-Streams”

Sie können jederzeit zwischen dem AWS-eigener Schlüssel und dem vom Kunden verwalteten Schlüssel wechseln. Sie können einen vom Kunden verwalteten Schlüssel angeben, wenn Sie eine neue Tabelle erstellen oder den KMS-Schlüssel einer vorhandenen Tabelle mithilfe der Konsole oder programmgesteuert mithilfe von CQL-Anweisungen ändern. Um zu erfahren wie dies geht, vgl. Verschlüsselung in Ruhe: So verwenden Sie kundenseitig verwaltete Schlüssel zum Verschlüsseln von Tabellen in Amazon Keyspaces.

Die Verschlüsselung im Ruhezustand mit der Standardoption von AWS-eigene Schlüssel wird ohne zusätzliche Kosten angeboten. Es werden jedoch AWS KMS -Gebühren für kundenseitig verwaltete Schlüssel erhoben. Weitere Informationen zur Preisgestaltung finden Sie unter AWS KMS – Preise.

Die Amazon Keyspaces-Verschlüsselung im Ruhezustand ist in allen Regionen verfügbar AWS-Regionen, auch in den AWS Regionen China (Peking) und AWS China (Ningxia). Weitere Informationen finden Sie unter Verschlüsselung im Ruhezustand: So funktioniert sie in Amazon Keyspaces.

Themen

- Verschlüsselung im Ruhezustand: So funktioniert sie in Amazon Keyspaces
- Verschlüsselung in Ruhe: So verwenden Sie kundenseitig verwaltete Schlüssel zum Verschlüsseln von Tabellen in Amazon Keyspaces

Verschlüsselung im Ruhezustand: So funktioniert sie in Amazon Keyspaces

Bei der Amazon Keyspaces-Verschlüsselung (für Apache Cassandra) werden Ihre Daten im Ruhezustand mit dem 256-Bit-Advanced Encryption Standard () verschlüsselt. AES-256 Dies hilft, Ihre Daten vor unbefugtem Zugriff auf den zugrunde liegenden Speicher zu schützen. Alle Kundendaten in Amazon Keyspaces-Tabellen werden standardmäßig im Ruhezustand verschlüsselt, und die serverseitige Verschlüsselung ist transparent, sodass keine Änderungen an Anwendungen erforderlich sind.

Encryption at Rest ist in AWS Key Management Service (AWS KMS) integriert, um den Verschlüsselungsschlüssel zu verwalten, der zum Verschlüsseln Ihrer Tabellen verwendet wird. Wenn Sie eine neue Tabelle erstellen oder eine vorhandene Tabelle aktualisieren, können Sie eine der folgenden AWS KMS Schlüsseloptionen wählen:

- AWS-eigener Schlüssel — Dies ist der Standardverschlüsselungstyp. Der Schlüssel gehört Amazon Keyspaces (ohne zusätzliche Kosten).
- Vom Kunden verwalteter Schlüssel — Dieser Schlüssel wird in Ihrem Konto gespeichert und wird von Ihnen erstellt, gehört und verwaltet. Sie haben die volle Kontrolle über den vom Kunden verwalteten Schlüssel (es AWS KMS fallen Gebühren an).

AWS KMS Schlüssel (KMS-Schlüssel)

Die Verschlüsselung im Ruhezustand schützt alle Ihre Amazon Keyspaces-Daten mit einem AWS KMS Schlüssel. Standardmäßig verwendet Amazon Keyspaces einen, einen mehrinstanzigen Verschlüsselungsschlüssel AWS-eigener Schlüssel, der in einem Amazon Keyspaces-Dienstkonto erstellt und verwaltet wird.

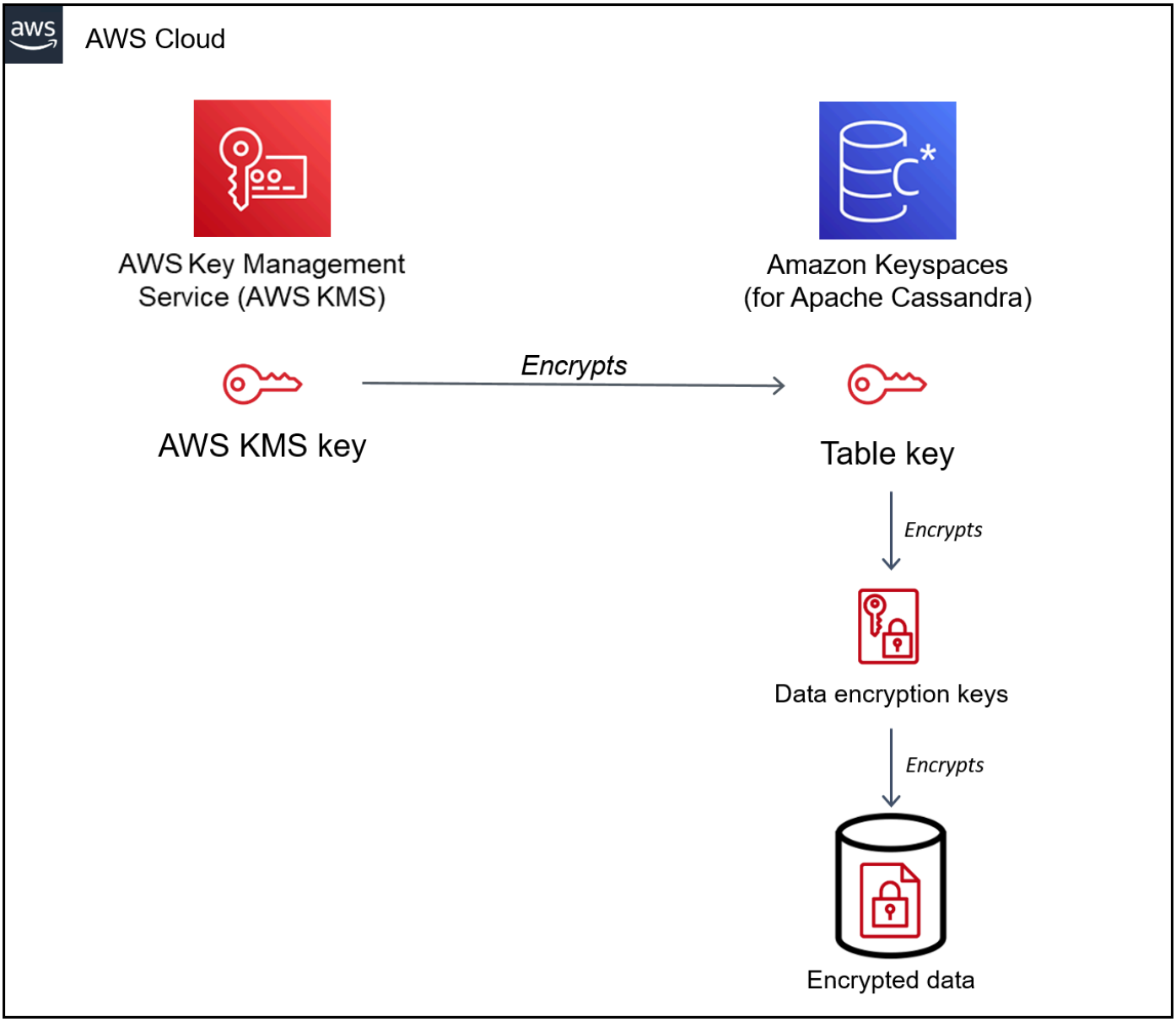
Sie können Ihre Amazon Keyspaces-Tabellen jedoch mit einem vom Kunden verwalteten Schlüssel in Ihrem verschlüsseln. AWS-Konto Sie können für jede Tabelle in einem Schlüsselraum einen anderen KMS-Schlüssel auswählen. Der KMS-Schlüssel, den Sie für eine Tabelle auswählen, wird auch zum Verschlüsseln aller Metadaten und wiederherstellbaren Backups verwendet.

Sie wählen den KMS-Schlüssel für eine Tabelle aus, wenn Sie die Tabelle erstellen oder aktualisieren. Sie können den KMS-Schlüssel für eine Tabelle jederzeit ändern, entweder in der Amazon Keyspaces-Konsole oder mithilfe der ALTER TABLE-Anweisung. Das Umschalten der KMS-Schlüssel erfolgt reibungslos und erfordert keine Ausfallzeiten oder Serviceverschlechterungen.

Schlüsselhierarchie

Amazon Keyspaces verwendet eine Schlüsselhierarchie, um Daten zu verschlüsseln. In dieser Schlüsselhierarchie ist der KMS-Schlüssel der Stammschlüssel. Er wird zum Verschlüsseln und Entschlüsseln des Verschlüsselungsschlüssels für die Amazon Keyspaces-Tabelle verwendet. Der Tabellenverschlüsselungsschlüssel wird zum Verschlüsseln der Verschlüsselungsschlüssel verwendet, die von Amazon Keyspaces intern zum Verschlüsseln und Entschlüsseln von Daten bei Lese- und Schreibvorgängen verwendet werden.

Mit der Verschlüsselungsschlüsselhierarchie können Sie Änderungen am KMS-Schlüssel vornehmen, ohne Daten erneut verschlüsseln zu müssen oder Anwendungen und laufende Datenvorgänge zu beeinträchtigen.



Schlüssel der Tabelle

Der Amazon Keyspaces-Tabellenschlüssel wird als Schlüsselverschlüsselungsschlüssel verwendet. Amazon Keyspaces verwendet den Tabellenschlüssel, um die internen Datenverschlüsselungsschlüssel zu schützen, die zum Verschlüsseln der in Tabellen, Protokolldateien und wiederherstellbaren Backups gespeicherten Daten verwendet werden. Amazon Keyspaces generiert einen eindeutigen Datenverschlüsselungsschlüssel für jede zugrunde liegende Struktur in einer Tabelle. Es können jedoch mehrere Tabellenzeilen durch denselben Datenverschlüsselungsschlüssel geschützt werden.

Wenn Sie den KMS-Schlüssel zum ersten Mal auf einen vom Kunden verwalteten Schlüssel festlegen, AWS KMS wird ein Datenschlüssel generiert. Der AWS KMS Datenschlüssel bezieht sich auf den Tabellenschlüssel in Amazon Keyspaces.

Wenn Sie auf eine verschlüsselte Tabelle zugreifen, sendet Amazon Keyspaces eine Anfrage an, den KMS-Schlüssel AWS KMS zum Entschlüsseln des Tabellenschlüssels zu verwenden. Dann verwendet es den Klartext-Tabellenschlüssel, um die Amazon Keyspaces-Datenverschlüsselungsschlüssel zu entschlüsseln, und es verwendet die Klartext-Datenverschlüsselungsschlüssel, um Tabellendaten zu entschlüsseln.

Amazon Keyspaces verwendet und speichert den Tabellenschlüssel und die Datenverschlüsselungsschlüssel außerhalb von. AWS KMS Alle Schlüssel werden mit Advanced Encryption Standard (AES)-Verschlüsselung und 256-Bit-Verschlüsselungsschlüsseln geschützt. Dann speichert es die verschlüsselten Schlüssel zusammen mit den verschlüsselten Daten, sodass sie verfügbar sind, um die Tabellendaten bei Bedarf zu entschlüsseln.

Tabellenschlüssel-Caching

Um zu vermeiden, dass AWS KMS jede Amazon Keyspaces-Operation aufgerufen wird, speichert Amazon Keyspaces die Klartext-Tabellenschlüssel für jede Verbindung im Speicher zwischen. Wenn Amazon Keyspaces nach fünf Minuten Inaktivität eine Anfrage für den zwischengespeicherten Tabellenschlüssel erhält, sendet es eine neue Anforderung zur Entschlüsselung des Tabellenschlüssels. AWS KMS Dieser Aufruf erfasst alle Änderungen, die seit der letzten Anforderung zur Entschlüsselung des Tabellenschlüssels an den Zugriffsrichtlinien für den KMS-Schlüssel in AWS KMS oder AWS Identity and Access Management (IAM) vorgenommen wurden.

Umschlagverschlüsselung

Wenn Sie den vom Kunden verwalteten Schlüssel für Ihre Tabelle ändern, generiert Amazon Keyspaces einen neuen Tabellenschlüssel. Anschließend verwendet es den neuen Tabellenschlüssel, um die Datenverschlüsselungsschlüssel erneut zu verschlüsseln. Außerdem verwendet es den neuen Tabellenschlüssel, um frühere Tabellenschlüssel zu verschlüsseln, die zum Schutz wiederherstellbarer Backups verwendet werden. Dieser Vorgang wird als Umschlagverschlüsselung bezeichnet. Dadurch wird sichergestellt, dass Sie auch dann auf wiederherstellbare Backups zugreifen können, wenn Sie den vom Kunden verwalteten Schlüssel wechseln. Weitere Informationen zur Umschlagverschlüsselung finden Sie im AWS Key Management Service Entwicklerhandbuch unter Envelope Encryption.

Themen

- AWS eigene Schlüssel
- Kundenseitig verwaltete Schlüssel
- Nutzungshinweise zur Verschlüsselung im Ruhezustand

AWS eigene Schlüssel

AWS-eigene Schlüssel sind nicht in Ihrem AWS-Konto gespeichert. Sie sind Teil einer Sammlung von KMS-Schlüsseln, die AWS Eigentümer und Verwalter für die Verwendung in mehreren Ordnern sind AWS-Konten. AWS Dienste, die Sie AWS-eigene Schlüssel zum Schutz Ihrer Daten verwenden können.

Sie können ihre Nutzung AWS-eigene Schlüssel nicht einsehen, verwalten oder verwenden oder überprüfen. Sie müssen jedoch keine Arbeiten ausführen oder Programme ändern, um die Schlüssel zu schützen, mit denen Ihre Daten verschlüsselt werden.

Für die Nutzung von wird Ihnen weder eine monatliche Gebühr noch eine Nutzungsgebühr berechnet AWS-eigene Schlüssel, und sie werden nicht auf die AWS KMS Kontingente für Ihr Konto angerechnet.

Kundenseitig verwaltete Schlüssel

Vom Kunden verwaltete Schlüssel sind Schlüssel in Ihrem AWS-Konto Konto, die Sie erstellen, besitzen und verwalten. Sie haben die volle Kontrolle über diese KMS-Schlüssel.

Verwenden Sie einen kundenverwalteten KMS-Schlüssel, um die folgenden Funktionen zu erhalten:

- Sie erstellen und verwalten den vom Kunden verwalteten Schlüssel, einschließlich der Festlegung und Verwaltung der wichtigsten Richtlinien, IAM-Richtlinien und Genehmigungen, um den Zugriff auf den vom Kunden verwalteten Schlüssel zu kontrollieren. Sie können den vom Kunden verwalteten Schlüssel aktivieren und deaktivieren, die automatische Schlüsselrotation aktivieren und deaktivieren und den vom Kunden verwalteten Schlüssel so planen, dass er gelöscht wird, wenn er nicht mehr verwendet wird. Sie können Tags und Aliase für die vom Kunden verwalteten Schlüssel erstellen.
- Sie können einen kundenverwalteten Schlüssel mit importiertem Schlüsselmaterial oder einen kundenverwalteten Schlüssel in einem benutzerdefinierten Schlüsselspeicher verwenden, den Sie besitzen und verwalten.

- Sie können Amazon AWS CloudTrail CloudWatch Logs verwenden, um die Anfragen zu verfolgen, an die Amazon Keyspaces in AWS KMS Ihrem Namen sendet. Weitere Informationen finden Sie unter the section called “Schritt 6: Konfigurieren Sie die Überwachung mit AWS CloudTrail”.

Für vom Kunden verwaltete Schlüssel fällt für jeden API-Aufruf eine Gebühr an, und für diese KMS-Schlüssel gelten AWS KMS Kontingente. Weitere Informationen finden Sie unter AWS KMS Ressourcen- oder Anforderungskontingente.

Wenn Sie einen vom Kunden verwalteten Schlüssel als Stammverschlüsselungsschlüssel für eine Tabelle angeben, werden wiederherstellbare Backups mit demselben Verschlüsselungsschlüssel verschlüsselt, der für die Tabelle zum Zeitpunkt der Erstellung des Backups angegeben wurde. Wenn der KMS-Schlüssel für die Tabelle rotiert wird, stellt die Schlüsselumhüllung sicher, dass der neueste KMS-Schlüssel Zugriff auf alle wiederherstellbaren Backups hat.

Amazon Keyspaces muss Zugriff auf Ihren vom Kunden verwalteten Schlüssel haben, um Ihnen Zugriff auf Ihre Tabellendaten zu gewähren. Wenn der Status des Verschlüsselungsschlüssels auf deaktiviert gesetzt ist oder dessen Löschung geplant ist, kann Amazon Keyspaces keine Daten verschlüsseln oder entschlüsseln. Daher können Sie keine Lese- und Schreibvorgänge für die Tabelle ausführen. Sobald der Service feststellt, dass auf Ihren Verschlüsselungscode nicht zugegriffen werden kann, sendet Amazon Keyspaces eine E-Mail-Benachrichtigung, um Sie darauf hinzuweisen.

Sie müssen den Zugriff auf Ihren Verschlüsselungsschlüssel innerhalb von sieben Tagen wiederherstellen. Andernfalls löscht Amazon Keyspaces Ihre Tabelle automatisch. Als Vorsichtsmaßnahme erstellt Amazon Keyspaces vor dem Löschen der Tabelle eine wiederherstellbare Sicherungskopie Ihrer Tabellendaten. Amazon Keyspaces bewahrt das wiederherstellbare Backup 35 Tage lang auf. Nach 35 Tagen können Sie Ihre Tabellendaten nicht mehr wiederherstellen. Das wiederherstellbare Backup wird Ihnen nicht in Rechnung gestellt, es fallen jedoch die Standardgebühren für die Wiederherstellung an.

Sie können dieses wiederherstellbare Backup verwenden, um Ihre Daten in einer neuen Tabelle wiederherzustellen. Um die Wiederherstellung einzuleiten, muss der letzte vom Kunden verwaltete Schlüssel, der für die Tabelle verwendet wurde, aktiviert werden, und Amazon Keyspaces muss Zugriff darauf haben.

> **Note**
>
> Wenn Sie eine Tabelle erstellen, die mit einem vom Kunden verwalteten Schlüssel verschlüsselt ist, auf den nicht zugegriffen werden kann oder der gelöscht werden soll, bevor

der Erstellungsvorgang abgeschlossen ist, tritt ein Fehler auf. Der Vorgang zum Erstellen einer Tabelle schlägt fehl und Sie erhalten eine E-Mail-Benachrichtigung.

Nutzungshinweise zur Verschlüsselung im Ruhezustand

Beachten Sie Folgendes, wenn Sie die Verschlüsselung im Ruhezustand in Amazon Keyspaces verwenden.

- Server-side Die Verschlüsselung im Ruhezustand ist für alle Amazon Keyspaces-Tabellen aktiviert und kann nicht deaktiviert werden. Die gesamte Tabelle ist im Ruhezustand verschlüsselt. Sie können keine bestimmten Spalten oder Zeilen für die Verschlüsselung auswählen.
- Standardmäßig verwendet Amazon Keyspaces einen Single-Service-Standardschlüssel (AWS-eigener Schlüssel) zum Verschlüsseln all Ihrer Tabellen. Wenn dieser Schlüssel nicht existiert, wird er für Sie erstellt. Die Standardschlüssel des Dienstes können nicht deaktiviert werden.
- Bei der Verschlüsselung im Ruhezustand werden Daten nur dann verschlüsselt, wenn sie statisch (im Ruhezustand) auf einem persistenten Speichermedium gespeichert sind. Wenn die Datensicherheit bei der Übertragung oder bei der Verwendung von Daten ein Problem darstellt, müssen Sie zusätzliche Maßnahmen ergreifen:
  - Daten während der Übertragung: Alle Ihre Daten in Amazon Keyspaces werden während der Übertragung verschlüsselt. Standardmäßig wird die Kommunikation zu und von Amazon Keyspaces durch die Verschlüsselung Secure Sockets Layer (SSL) /Transport Layer Security (TLS) geschützt.
  - Verwendete Daten: Schützen Sie Ihre Daten, bevor Sie sie an Amazon Keyspaces senden, indem Sie eine clientseitige Verschlüsselung verwenden.
  - Kundenverwaltete Schlüssel: Daten, die in Ihren Tabellen gespeichert sind, werden immer mit Ihren vom Kunden verwalteten Schlüsseln verschlüsselt. Operationen, die atomare Aktualisierungen mehrerer Zeilen durchführen, verschlüsseln jedoch Daten, die vorübergehend AWS-eigene Schlüssel während der Verarbeitung verwendet werden. Dazu gehören Operationen zum Löschen von Bereichen und Operationen, die gleichzeitig auf statische und nicht statische Daten zugreifen.
- Für einen einzelnen, vom Kunden verwalteten Schlüssel können bis zu 50.000 Zuschüsse gewährt werden. Für jede Amazon Keyspaces-Tabelle, die einem vom Kunden verwalteten Schlüssel zugeordnet ist, werden 2 Zuschüsse verbraucht. Ein Grant wird freigegeben, wenn die Tabelle gelöscht wird. Der zweite Zuschuss wird verwendet, um einen automatischen Snapshot der Tabelle zu erstellen, um sich vor Datenverlust zu schützen, falls Amazon Keyspaces unbeabsichtigt den

Zugriff auf den vom Kunden verwalteten Schlüssel verliert. Dieser Zuschuss wird 42 Tage nach dem Löschen der Tabelle freigegeben.

Verschlüsselung in Ruhe: So verwenden Sie kundenseitig verwaltete Schlüssel zum Verschlüsseln von Tabellen in Amazon Keyspaces

Sie können die Konsolen- oder CQL-Anweisungen verwenden, um die AWS KMS key für neue Tabellen anzugeben und die Verschlüsselungsschlüssel vorhandener Tabellen in Amazon Keyspaces zu aktualisieren. Im folgenden Thema wird beschrieben, wie vom Kunden verwaltete Schlüssel für neue und bestehende Tabellen implementiert werden.

Themen

- Voraussetzungen: Erstellen Sie einen vom Kunden verwalteten Schlüssel mithilfe von Amazon Keyspaces AWS KMS und gewähren Sie ihm Berechtigungen
- Schritt 3: Geben Sie einen vom Kunden verwalteten Schlüssel für eine neue Tabelle an
- Schritt 4: Aktualisieren Sie den Verschlüsselungsschlüssel einer vorhandenen Tabelle
- Schritt 5: Verwenden Sie den Amazon Keyspaces-Verschlüsselungskontext in Protokollen
- Schritt 6: Konfigurieren Sie die Überwachung mit AWS CloudTrail

Voraussetzungen: Erstellen Sie einen vom Kunden verwalteten Schlüssel mithilfe von Amazon Keyspaces AWS KMS und gewähren Sie ihm Berechtigungen

Bevor Sie eine Amazon Keyspaces-Tabelle mit einem vom Kunden verwalteten Schlüssel schützen können, müssen Sie zuerst den Schlüssel in AWS Key Management Service (AWS KMS) erstellen und dann Amazon Keyspaces autorisieren, diesen Schlüssel zu verwenden.

Schritt 1: Erstellen Sie einen vom Kunden verwalteten Schlüssel mit AWS KMS

Um einen vom Kunden verwalteten Schlüssel zu erstellen, der zum Schutz einer Amazon Keyspaces-Tabelle verwendet werden soll, können Sie die Schritte unter KMS-Schlüssel mit symmetrischer Verschlüsselung mithilfe der Konsole oder der AWS API erstellen befolgen.

Schritt 2: Autorisieren Sie die Verwendung Ihres vom Kunden verwalteten Schlüssels

Bevor Sie einen vom Kunden verwalteten Schlüssel zum Schutz einer Amazon Keyspaces-Tabelle auswählen können, müssen die Richtlinien für diesen vom Kunden verwalteten Schlüssel Amazon Keyspaces die Erlaubnis geben, ihn in Ihrem Namen zu verwenden. Sie haben die volle Kontrolle

über die Richtlinien und Zuschüsse für den vom Kunden verwalteten Schlüssel. Sie können diese Berechtigungen in einer Schlüsselrichtlinie, einer IAM-Richtlinie oder einer Erteilung bereitstellen.

Amazon Keyspaces benötigt keine zusätzliche Autorisierung, um den Standard AWS-eigener Schlüsselzum Schutz der Amazon Keyspaces-Tabellen in Ihrem AWS Konto zu verwenden.

Die folgenden Themen zeigen, wie Sie die erforderlichen Berechtigungen mithilfe von IAM-Richtlinien und -Zuschüssen konfigurieren, die es Amazon Keyspaces-Tabellen ermöglichen, einen vom Kunden verwalteten Schlüssel zu verwenden.

Themen

- Wichtige Richtlinie für vom Kunden verwaltete Schlüssel
- Beispiel für eine Schlüsselrichtlinie
- Verwendung von Zuschüssen zur Autorisierung von Amazon Keyspaces

Wichtige Richtlinie für vom Kunden verwaltete Schlüssel

Wenn Sie einen vom Kunden verwalteten Schlüssel zum Schutz einer Amazon Keyspaces-Tabelle auswählen, erhält Amazon Keyspaces die Erlaubnis, den vom Kunden verwalteten Schlüssel im Namen des Prinzipals zu verwenden, der die Auswahl trifft. Dieser Principal, ein Benutzer oder eine Rolle, muss über die Berechtigungen für den vom Kunden verwalteten Schlüssel verfügen, die Amazon Keyspaces benötigt.

Amazon Keyspaces benötigt mindestens die folgenden Berechtigungen für einen vom Kunden verwalteten Schlüssel:

- kms:Encrypt
- kms:Decrypt
- kms: ReEncrypt * (für kms: ReEncryptFrom und kms:ReEncryptTo)
- km: GenerateDataKey * (für km: GenerateDataKey und km: GenerateDataKeyWithoutPlaintext)
- km: DescribeKey
- km: CreateGrant

Beispiel für eine Schlüsselrichtlinie

Beispielsweise bietet die folgende Beispiel-Schlüsselrichtlinie nur die erforderlichen Berechtigungen. Die Richtlinie hat folgende Auswirkungen:

- Erlaubt Amazon Keyspaces, den vom Kunden verwalteten Schlüssel für kryptografische Operationen zu verwenden und Zuschüsse zu gewähren — aber nur, wenn Amazon Keyspaces im Namen von Prinzipalen des Kontos handelt, die die Erlaubnis haben, Amazon Keyspaces zu verwenden. Wenn die in der Datenschutzerklärung angegebenen Principals nicht berechtigt sind, Amazon Keyspaces zu verwenden, schlägt der Anruf fehl, auch wenn er vom Amazon Keyspaces-Service stammt.
- Der ViaService Bedingungsschlüssel kms: erlaubt die Berechtigungen nur, wenn die Anfrage von Amazon Keyspaces im Namen der in der Datenschutzerklärung aufgeführten Prinzipale kommt. Diese Prinzipale können diese Operationen nicht direkt aufrufen. Beachten Sie, dass der `kms:ViaService`-Wert, `cassandra.*.amazonaws.com`, in der Region-Position ein Sternchen (*) hat. Amazon Keyspaces benötigt die Genehmigung, unabhängig von einer bestimmten AWS-Region Person zu sein.
- Gewährt den vom Kunden verwalteten Schlüsseladministratoren (Benutzern, die die `db-team` Rolle übernehmen können) schreibgeschützten Zugriff auf den vom Kunden verwalteten Schlüssel und die Berechtigung, Zuschüsse zu widerrufen, einschließlich der Zuweisungen, die Amazon Keyspaces zum Schutz der Tabelle benötigt.
- Gewährt Amazon Keyspaces schreibgeschützten Zugriff auf den vom Kunden verwalteten Schlüssel. In diesem Fall kann Amazon Keyspaces diese Operationen direkt aufrufen. Es muss nicht im Namen eines Kontoinhabers handeln.

Bevor Sie ein Beispiel für eine Schlüsselrichtlinie verwenden, ersetzen Sie die Beispielprinzipale durch tatsächliche Prinzipale aus Ihrem. AWS-Konto

```
{
  "Id": "key-policy-cassandra",
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid" : "Allow access through Amazon Keyspaces for all principals in the account
 that are authorized to use Amazon Keyspaces",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:user/db-lead"},
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey",
```

```
        "kms:CreateGrant"
      ],
      "Resource": "*",
      "Condition": {
         "StringLike": {
           "kms:ViaService" : "cassandra.*.amazonaws.com"
         }
      }
    },
    {
      "Sid":  "Allow administrators to view the customer managed key and revoke
 grants",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/db-team"
       },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
      "Resource": "*"
    }
  ]
}
```

Verwendung von Zuschüssen zur Autorisierung von Amazon Keyspaces

Zusätzlich zu den wichtigsten Richtlinien verwendet Amazon Keyspaces Zuschüsse, um Berechtigungen für einen vom Kunden verwalteten Schlüssel festzulegen. Um die Erteilungen für den kundenverwalteten Schlüssel in Ihrem Konto anzuzeigen, verwenden Sie die ListGrants-Operation. Amazon Keyspaces benötigt keine Zuschüsse oder zusätzliche Berechtigungen, um Ihre Tabelle AWS-eigener Schlüsselzu schützen.

Amazon Keyspaces verwendet die gewährten Berechtigungen, wenn es Systemwartungs- und kontinuierliche Datenschutzaufgaben im Hintergrund ausführt. Außerdem werden Erteilungen verwenden, um Tabellenschlüssel zu generieren.

Jede Erteilung ist tabellenspezifisch. Wenn das Konto mehrere Tabellen enthält, die unter demselben vom Kunden verwalteten Schlüssel verschlüsselt sind, gibt es für jede Tabelle eine Genehmigung jedes Typs. Die Gewährung wird durch den Amazon Keyspaces-Verschlüsselungskontext

eingeschränkt, der den Tabellennamen und die AWS-Konto ID umfasst. Der Zuschuss beinhaltet die Erlaubnis, den Zuschuss zurückzuziehen, wenn er nicht mehr benötigt wird.

Um die Grants zu erstellen, muss Amazon Keyspaces über die Berechtigung verfügen, `CreateGrant` im Namen des Benutzers, der die verschlüsselte Tabelle erstellt hat, anzurufen.

Die Schlüsselrichtlinie kann es dem Konto auch erlauben, die Erteilung für den kundenverwalteten Schlüssel zu widerrufen. Wenn Sie jedoch den Zuschuss für eine aktive verschlüsselte Tabelle widerrufen, kann Amazon Keyspaces die Tabelle nicht schützen und verwalten.

Schritt 3: Geben Sie einen vom Kunden verwalteten Schlüssel für eine neue Tabelle an

Gehen Sie wie folgt vor, um den vom Kunden verwalteten Schlüssel in einer neuen Tabelle mithilfe der Amazon Keyspaces-Konsole oder CQL anzugeben.

Erstellen Sie eine verschlüsselte Tabelle mit einem vom Kunden verwalteten Schlüssel (Konsole)

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) und anschließend Create table (Tabelle erstellen) aus.
3. Wählen Sie auf der Seite Tabelle erstellen im Abschnitt Tabellendetails einen Schlüsselraum aus und geben Sie einen Namen für die neue Tabelle ein.
4. Erstellen Sie im Abschnitt Schema das Schema für Ihre Tabelle.
5. Wählen Sie im Abschnitt Tabelleneinstellungen die Option Einstellungen anpassen aus.
6. Fahren Sie mit den Verschlüsselungseinstellungen fort.

   In diesem Schritt wählen Sie die Verschlüsselungseinstellungen für die Tabelle aus.

   Wählen Sie im Abschnitt Verschlüsselung im Ruhezustand unter Wählen Sie eine AWS KMS key die Option Wählen Sie einen anderen KMS-Schlüssel (erweitert) und wählen Sie im Suchfeld einen Amazon-Ressourcennamen (ARN) aus AWS KMS key oder geben Sie einen ein.

   > **Note**
   >
   > Wenn auf den ausgewählten Schlüssel nicht zugegriffen werden kann oder die erforderlichen Berechtigungen fehlen, finden Sie weitere Informationen unter

Problembehandlung beim Schlüsselzugriff im AWS Key Management Service Entwicklerhandbuch.

7. Wählen Sie Create (Erstellen) , um die verschlüsselte Tabelle zu erstellen.

Erstellen Sie eine neue Tabelle mit einem vom Kunden verwalteten Schlüssel für Verschlüsselung im Ruhezustand (CQL)

Um eine neue Tabelle zu erstellen, die einen vom Kunden verwalteten Schlüssel für die Verschlüsselung im Ruhezustand verwendet, können Sie die `CREATE TABLE` Anweisung verwenden, wie im folgenden Beispiel gezeigt. Stellen Sie sicher, dass Sie den Schlüssel ARN durch einen ARN für einen gültigen Schlüssel mit Amazon Keyspaces erteilten Berechtigungen ersetzen.

```
CREATE TABLE my_keyspace.my_table(id bigint, name text, place text STATIC, PRIMARY
 KEY(id, name)) WITH CUSTOM_PROPERTIES = {
        'encryption_specification':{
                'encryption_type': 'CUSTOMER_MANAGED_KMS_KEY',
                'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
            }
    };
```

Wenn Sie einen erhalten`Invalid Request Exception`, müssen Sie bestätigen, dass der vom Kunden verwaltete Schlüssel gültig ist und Amazon Keyspaces über die erforderlichen Berechtigungen verfügt. Informationen zur Bestätigung, dass der Schlüssel korrekt konfiguriert wurde, finden Sie unter Fehlerbehebung beim Schlüsselzugriff im AWS Key Management Service Entwicklerhandbuch.

Schritt 4: Aktualisieren Sie den Verschlüsselungsschlüssel einer vorhandenen Tabelle

Sie können auch die Amazon Keyspaces-Konsole oder CQL verwenden, um die Verschlüsselungsschlüssel einer vorhandenen Tabelle jederzeit zwischen einem AWS-eigener Schlüssel und einem vom Kunden verwalteten KMS-Schlüssel zu ändern.

Aktualisieren Sie eine bestehende Tabelle mit dem neuen vom Kunden verwalteten Schlüssel (Konsole)

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.

3. Wählen Sie die Tabelle aus, die Sie aktualisieren möchten, und klicken Sie dann auf den Tab Zusätzliche Einstellungen.
4. Wählen Sie im Abschnitt Verschlüsselung im Ruhezustand die Option Verschlüsselung verwalten aus, um die Verschlüsselungseinstellungen für die Tabelle zu bearbeiten.

   Wählen Sie unter Wählen Sie die Option Anderen KMS-Schlüssel auswählen (erweitert) und wählen Sie im Suchfeld einen Amazon-Ressourcennamen (ARN) aus AWS KMS key oder geben Sie einen ein. AWS KMS key

   > **Note**
   >
   > Wenn der von Ihnen gewählte Schlüssel nicht gültig ist, finden Sie weitere Informationen unter Fehlerbehebung beim Schlüsselzugriff im AWS Key Management Service Entwicklerhandbuch.

   Alternativ können Sie eine AWS-eigener Schlüssel für eine Tabelle wählen, die mit einem vom Kunden verwalteten Schlüssel verschlüsselt ist.
5. Wählen Sie Änderungen speichern, um Ihre Änderungen an der Tabelle zu speichern.

Aktualisieren Sie den Verschlüsselungsschlüssel, der für eine bestehende Tabelle verwendet wurde

Um den Verschlüsselungsschlüssel einer vorhandenen Tabelle zu ändern, verwenden Sie die `ALTER TABLE` Anweisung, um einen vom Kunden verwalteten Schlüssel für die Verschlüsselung im Ruhezustand anzugeben. Stellen Sie sicher, dass Sie den Schlüssel ARN durch einen ARN für einen gültigen Schlüssel mit Amazon Keyspaces erteilten Berechtigungen ersetzen.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
              'encryption_specification':{
                      'encryption_type': 'CUSTOMER_MANAGED_KMS_KEY',
                      'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
                  }
         };
```

Wenn Sie einen erhalten`Invalid Request Exception`, müssen Sie bestätigen, dass der vom Kunden verwaltete Schlüssel gültig ist und Amazon Keyspaces über die erforderlichen Berechtigungen verfügt. Informationen zur Bestätigung, dass der Schlüssel korrekt konfiguriert

wurde, finden Sie unter Fehlerbehebung beim Schlüsselzugriff im AWS Key Management Service Entwicklerhandbuch.

Um den Verschlüsselungsschlüssel wieder auf die Standardoption Verschlüsselung im Ruhezustand zurückzusetzen AWS-eigene Schlüssel, können Sie die `ALTER TABLE` Anweisung verwenden, wie im folgenden Beispiel gezeigt.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
                'encryption_specification':{
                      'encryption_type' : 'AWS_OWNED_KMS_KEY'
                    }
         };
```

Schritt 5: Verwenden Sie den Amazon Keyspaces-Verschlüsselungskontext in Protokollen

Ein Verschlüsselungskontext ist eine Gruppe von Schlüssel/Wert-Paaren mit willkürlichen, nicht geheimen Daten. Wenn Sie einen Verschlüsselungskontext in eine Anfrage zur Verschlüsselung von Daten einbeziehen, wird der Verschlüsselungskontext AWS KMS kryptografisch an die verschlüsselten Daten gebunden. Zur Entschlüsselung der Daten müssen Sie denselben Verschlüsselungskontext übergeben.

Amazon Keyspaces verwendet bei allen AWS KMS kryptografischen Vorgängen denselben Verschlüsselungskontext. Wenn Sie einen vom Kunden verwalteten Schlüssel verwenden, um Ihre Amazon Keyspaces-Tabelle zu schützen, können Sie den Verschlüsselungskontext verwenden, um die Verwendung des vom Kunden verwalteten Schlüssels in Prüfaufzeichnungen und Protokollen zu identifizieren. Es erscheint auch im Klartext in Protokollen, z. B. in Protokollen für AWS CloudTrailund Amazon CloudWatch Logs.

In seinen Anfragen verwendet Amazon Keyspaces einen Verschlüsselungskontext mit drei Schlüssel-Wert-Paaren. AWS KMS

```
"encryptionContextSubset": {
    "aws:cassandra:keyspaceName": "my_keyspace",
    "aws:cassandra:tableName": "mytable"
    "aws:cassandra:subscriberId": "111122223333"
}
```

- Keyspace — Das erste Schlüssel-Wert-Paar identifiziert den Schlüsselraum, der die Tabelle enthält, die Amazon Keyspaces verschlüsselt. Der Schlüssel lautet `aws:cassandra:keyspaceName`. Der Wert ist der Name des Schlüsselraums.

```
"aws:cassandra:keyspaceName": "<keyspace-name>"
```

Beispiel:

```
"aws:cassandra:keyspaceName": "my_keyspace"
```

- Tabelle — Das zweite Schlüssel-Wert-Paar identifiziert die Tabelle, die Amazon Keyspaces verschlüsselt. Der Schlüssel lautet `aws:cassandra:tableName`. Der Wert ist der Name der Tabelle.

  ```
  "aws:cassandra:tableName": "<table-name>"
  ```

  Beispiel:

  ```
  "aws:cassandra:tableName": "my_table"
  ```

- Konto — Das dritte Schlüssel-Wert-Paar identifiziert die. AWS-Konto Der Schlüssel lautet `aws:cassandra:subscriberId`. Der Wert ist die Konto-ID.

  ```
  "aws:cassandra:subscriberId": "<account-id>"
  ```

  Beispiel:

  ```
  "aws:cassandra:subscriberId": "111122223333"
  ```

Schritt 6: Konfigurieren Sie die Überwachung mit AWS CloudTrail

Wenn Sie einen vom Kunden verwalteten Schlüssel verwenden, um Ihre Amazon Keyspaces-Tabellen zu schützen, können Sie AWS CloudTrail Protokolle verwenden, um die Anfragen zu verfolgen, an die Amazon Keyspaces in AWS KMS Ihrem Namen sendet.

Die `CreateGrant` Anfragen `GenerateDataKeyDescribeKey`,`Decrypt`, und werden in diesem Abschnitt behandelt. Darüber hinaus verwendet Amazon Keyspaces eine RetireGrantOperation, um einen Zuschuss zu entfernen, wenn Sie eine Tabelle löschen.

> **Note**
>
> Wenn Sie mit Amazon Keyspaces arbeiten, können bei einigen Vorgängen CloudTrail Ereignisse mit dem `invokedBy` Feld von `dynamodb.amazonaws.com` generiert werden. Dies ist zu erwarten und tritt auf, weil Amazon Keyspaces in Amazon DynamoDB integriert ist, um seinen Service bereitzustellen.

GenerateDataKey

Amazon Keyspaces erstellt einen eindeutigen Tabellenschlüssel, um Daten im Ruhezustand zu verschlüsseln. Es sendet eine GenerateDataKeyAnfrage an AWS KMS , die den KMS-Schlüssel für die Tabelle spezifiziert.

Das Ereignis, das die `GenerateDataKey`-Operation aufzeichnet, ähnelt dem folgenden Beispielereignis. Der Benutzer ist das Amazon Keyspaces-Servicekonto. Zu den Parametern gehören der Amazon-Ressourcenname (ARN) des vom Kunden verwalteten Schlüssels, ein Schlüsselspezifizierer, für den ein 256-Bit-Schlüssel erforderlich ist, und der Verschlüsselungskontext, der den Schlüsselraum, die Tabelle und den identifiziert. AWS-Konto

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T04:56:05Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keySpec": "AES_256",
        "encryptionContext": {
            "aws:cassandra:keyspaceName": "my_keyspace",
            "aws:cassandra:tableName": "my_table",
            "aws:cassandra:subscriberId": "123SAMPLE012"
        },
        "keyId": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
```

```
    },
    "responseElements": null,
    "requestID": "5e8e9cb5-9194-4334-aacc-9dd7d50fe246",
    "eventID": "49fccab9-2448-4b97-a89d-7d5c39318d6f",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012",
    "sharedEventID": "84fbaaf0-9641-4e32-9147-57d2cb08792e"
}
```

DescribeKey

Amazon Keyspaces verwendet einen DescribeKeyVorgang, um festzustellen, ob der von Ihnen ausgewählte KMS-Schlüssel im Konto und in der Region vorhanden ist.

Das Ereignis, das die `DescribeKey`-Operation aufzeichnet, ähnelt dem folgenden Beispielereignis. Der Benutzer ist das Amazon Keyspaces-Servicekonto. Zu den Parametern gehören der ARN des vom Kunden verwalteten Schlüssels und ein Schlüsselspezifizierer, für den ein 256-Bit-Schlüssel erforderlich ist.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAZ3FNIIVIZZ6H7CFQG",
        "arn": "arn:aws:iam::123SAMPLE012:user/admin",
        "accountId": "123SAMPLE012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "admin",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
```

```
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-16T04:55:42Z"
            }
        },
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T04:55:58Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
    },
    "responseElements": null,
    "requestID": "c25a8105-050b-4f52-8358-6e872fb03a6c",
    "eventID": "0d96420e-707e-41b9-9118-56585a669658",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012"
}
```


Decrypt

Wenn Sie auf eine Amazon Keyspaces-Tabelle zugreifen, muss Amazon Keyspaces den Tabellenschlüssel entschlüsseln, damit es die Schlüssel entschlüsseln kann, die sich in der Hierarchie darunter befinden. Anschließend werden die Daten in der Tabelle entschlüsselt. Um den Tabellenschlüssel zu entschlüsseln, sendet Amazon Keyspaces eine Decrypt-Anfrage an AWS KMS , die den KMS-Schlüssel für die Tabelle angibt.

Das Ereignis, das die `Decrypt`-Operation aufzeichnet, ähnelt dem folgenden Beispielereignis. Der Benutzer ist Ihr Hauptbenutzer, der auf AWS-Konto die Tabelle zugreift. Zu den Parametern gehören der verschlüsselte Tabellenschlüssel (als Chiffretext-Blob) und der Verschlüsselungskontext, der die Tabelle und den identifiziert. AWS-Konto AWS KMS leitet die ID des vom Kunden verwalteten Schlüssels aus dem Chiffretext ab.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T05:29:44Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "encryptionContext": {
            "aws:cassandra:keyspaceName": "my_keyspace",
            "aws:cassandra:tableName": "my_table",
            "aws:cassandra:subscriberId": "123SAMPLE012"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "50e80373-83c9-4034-8226-5439e1c9b259",
    "eventID": "8db9788f-04a5-4ae2-90c9-15c79c411b6b",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012",
    "sharedEventID": "7ed99e2d-910a-4708-a4e3-0180d8dbb68e"
```

```
}
```

CreateGrant

Wenn Sie einen vom Kunden verwalteten Schlüssel verwenden, um Ihre Amazon Keyspaces-Tabelle zu schützen, verwendet Amazon Keyspaces Zuschüsse, damit der Service kontinuierliche Datenschutz-, Wartungs- und Haltbarkeitsaufgaben durchführen kann. Diese Zuschüsse sind am nicht erforderlich. AWS-eigene Schlüssel

Die Zuschüsse, die Amazon Keyspaces erstellt, sind tabellenspezifisch. Der Principal in der CreateGrantAnfrage ist der Benutzer, der die Tabelle erstellt hat.

Das Ereignis, das die `CreateGrant`-Operation aufzeichnet, ähnelt dem folgenden Beispielereignis. Zu den Parametern gehören der ARN des vom Kunden verwalteten Schlüssels für die Tabelle, der Principal des Empfängers und der ausscheidende Principal (der Amazon Keyspaces-Service) sowie die Vorgänge, die durch den Zuschuss abgedeckt werden. Es enthält auch eine Einschränkung, nach der alle Verschlüsselungsvorgänge den angegebenen Verschlüsselungskontext verwenden müssen.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAZ3FNIIVIZZ6H7CFQG",
        "arn": "arn:aws:iam::arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111:user/admin",
        "accountId": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "admin",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-16T04:55:42Z"
            }
        },
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-04-16T05:11:10Z",
    "eventSource": "kms.amazonaws.com",
```

```
    "eventName": "CreateGrant",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "a7d328af-215e-4661-9a69-88c858909f20",
        "operations": [
            "DescribeKey",
            "GenerateDataKey",
            "Decrypt",
            "Encrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "RetireGrant"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:cassandra:keyspaceName": "my_keyspace",
                "aws:cassandra:tableName": "my_table",
                "aws:cassandra:subscriberId": "123SAMPLE012"
            }
        },
        "retiringPrincipal": "cassandratest.us-east-1.amazonaws.com",
        "granteePrincipal": "cassandratest.us-east-1.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "18e4235f1b07f289762a31a1886cb5efd225f069280d4f76cd83b9b9b5501013"
    },
    "requestID": "b379a767-1f9b-48c3-b731-fb23e865e7f7",
    "eventID": "29ee1fd4-28f2-416f-a419-551910d20291",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123SAMPLE012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123SAMPLE012"
```

```
}
```

Verschlüsselung bei der Übertragung in Amazon Keyspaces

Amazon Keyspaces akzeptiert nur sichere Verbindungen mit Transport Layer Security (TLS). Die Verschlüsselung bei der Übertragung bietet eine zusätzliche Datenschutzebene, indem Ihre Daten auf dem Weg zu und von Amazon Keyspaces verschlüsselt werden. Unternehmensrichtlinien, Branchen- oder Regierungsvorschriften und Compliance-Anforderungen erfordern häufig die Verwendung von Verschlüsselung bei der Übertragung, um die Datensicherheit Ihrer Anwendungen bei der Übertragung von Daten über das Netzwerk zu erhöhen.

Informationen zum Verschlüsseln von `cqlsh` Verbindungen zu Amazon Keyspaces mithilfe von TLS finden Sie unter. the section called “Wie konfiguriert man `cqlsh` Verbindungen für TLS manuell” Informationen zur Verwendung der TLS-Verschlüsselung mit Client-Treibern finden Sie unter. the section called “Verwenden eines Cassandra-Client-Treibers”

Datenschutz im Netzwerkverkehr in Amazon Keyspaces

In diesem Thema wird beschrieben, wie Amazon Keyspaces (für Apache Cassandra) Verbindungen von lokalen Anwendungen zu Amazon Keyspaces und zwischen Amazon Keyspaces und anderen Ressourcen innerhalb derselben sichert. AWS AWS-Region

Datenverkehr zwischen Service und On-Premises-Clients und -Anwendungen

Sie haben zwei Verbindungsoptionen zwischen Ihrem privaten Netzwerk und: AWS

- Eine AWS Site-to-Site VPN Verbindung. Weitere Informationen finden Sie unter Was ist AWS Site-to-Site VPN? im AWS Site-to-Site VPN -Benutzerhandbuch.
- Eine Direct Connect Verbindung. Weitere Informationen finden Sie unter Was ist Direct Connect? im Direct Connect -Benutzerhandbuch.

Als verwalteter Service ist Amazon Keyspaces (für Apache Cassandra) durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon Keyspaces zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Cipher-Suites mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Amazon Keyspaces unterstützt zwei Methoden zur Authentifizierung von Kundenanfragen. Die erste Methode verwendet dienstspezifische Anmeldeinformationen, bei denen es sich um passwortbasierte Anmeldeinformationen handelt, die für einen bestimmten IAM-Benutzer generiert wurden. Sie können das Passwort mithilfe der IAM-Konsole, der oder der API erstellen AWS CLI und verwalten. AWS Weitere Informationen finden Sie unter Verwenden von IAM mit Amazon Keyspaces.

Die zweite Methode verwendet ein Authentifizierungs-Plugin für den DataStax Open-Source-Java-Treiber für Cassandra. Mit diesem Plugin können IAM-Benutzer, -Rollen und föderierte Identitäten mithilfe des AWS Signature Version 4-Prozesses (Sigv4) Authentifizierungsinformationen zu API-Anfragen von Amazon Keyspaces (für Apache Cassandra) hinzufügen. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Verkehr zwischen Ressourcen in derselben Region AWS

Schnittstellen-VPC-Endpunkte ermöglichen die private Kommunikation zwischen Ihrer Virtual Private Cloud (VPC), die in Amazon VPC läuft, und Amazon Keyspaces. Schnittstelle, über die VPC-Endpunkte betrieben werden. Dabei handelt es sich um einen AWS Dienst AWS PrivateLink, der die private Kommunikation zwischen VPCs und Diensten ermöglicht. AWS AWS PrivateLink ermöglicht dies durch die Verwendung einer elastic network interface mit privaten IPs in Ihrer VPC, sodass der Netzwerkverkehr das Amazon-Netzwerk nicht verlässt. Schnittstellen-VPC-Endpunkte benötigen kein Internet-Gateway, kein NAT-Gerät, keine VPN-Verbindung oder Direct Connect Verbindung. Weitere Informationen finden Sie unter Amazon Virtual Private Cloud und Interface VPC-Endpoints ()AWS PrivateLink. Beispiele für Richtlinien finden Sie unter the section called “Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces”.

AWS Identity and Access Management für Amazon Keyspaces

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu kontrollieren. AWS IAM-Administratoren kontrollieren, wer authentifiziert

(angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Amazon Keyspaces-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- So funktioniert Amazon Keyspaces mit IAM
- Beispiele für identitätsbasierte Richtlinien von Amazon Keyspaces
- AWS verwaltete Richtlinien für Amazon Keyspaces
- Fehlerbehebung bei Identität und Zugriff auf Amazon Keyspaces
- Verwenden von serviceverknüpften Rollen für Amazon Keyspaces

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von Ihrer Rolle ab:

- Servicebenutzer – Fordern Sie von Ihrem Administrator Berechtigungen an, wenn Sie nicht auf Features zugreifen können (siehe Fehlerbehebung bei Identität und Zugriff auf Amazon Keyspaces).
- Serviceadministrator – Bestimmen Sie den Benutzerzugriff und stellen Sie Berechtigungsanfragen (siehe So funktioniert Amazon Keyspaces mit IAM).
- IAM-Administrator – Schreiben Sie Richtlinien zur Zugriffsverwaltung (siehe Beispiele für identitätsbasierte Richtlinien von Amazon Keyspaces).

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen sich als IAM-Benutzer authentifizieren oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich als föderierte Identität anmelden, indem Sie Anmeldeinformationen aus einer Identitätsquelle wie AWS IAM Identity Center (IAM Identity Center), Single Sign-On-Authentifizierung

oder Anmeldeinformationen verwenden. Google/Facebook Weitere Informationen zum Anmelden finden Sie unter So melden Sie sich bei Ihrem AWS-Konto an im Benutzerhandbuch für AWS-Anmeldung .

AWS Bietet für den programmatischen Zugriff ein SDK und eine CLI zum kryptografischen Signieren von Anfragen. Weitere Informationen finden Sie unter AWS Signature Version 4 for API requests im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, dem sogenannten AWS-Konto Root-Benutzer, der vollständigen Zugriff auf alle AWS-Services Ressourcen hat. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Eine Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Tasks that require root user credentials im IAM-Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität mit bestimmten Berechtigungen für eine einzelne Person oder Anwendung. Wir empfehlen die Verwendung temporärer Anmeldeinformationen anstelle von IAM-Benutzern mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erfordern, dass menschliche Benutzer für den Zugriff AWS mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden müssen.

Eine IAM-Gruppe spezifiziert eine Sammlung von IAM-Benutzern und erleichtert die Verwaltung von Berechtigungen für große Gruppen von Benutzern. Weitere Informationen finden Sie unter Anwendungsfälle für IAM-Benutzer im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität mit spezifischen Berechtigungen, die temporäre Anmeldeinformationen bereitstellt. Sie können eine Rolle übernehmen, indem Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) wechseln AWS CLI oder einen AWS API-Vorgang aufrufen. Weitere Informationen finden Sie unter Methoden, um eine Rolle zu übernehmen im IAM-Benutzerhandbuch.

IAM-Rollen sind nützlich für den Verbundbenutzer-Zugriff, temporäre IAM-Benutzerberechtigungen, kontoübergreifenden Zugriff, serviceübergreifenden Zugriff und Anwendungen, die auf Amazon EC2 laufen. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie definiert Berechtigungen, wenn sie mit einer Identität oder Ressource verknüpft sind. AWS bewertet diese Richtlinien, wenn ein Principal eine Anfrage stellt. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Mit Hilfe von Richtlinien legen Administratoren fest, wer Zugriff auf was hat, indem sie definieren, welches Prinzipal welche Aktionen auf welchen Ressourcenund unter welchen Bedingungendurchführen darf.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator erstellt IAM-Richtlinien und fügt sie zu Rollen hinzu, die die Benutzer dann übernehmen können. IAM-Richtlinien definieren Berechtigungen unabhängig von der Methode, die zur Ausführung der Operation verwendet wird.

Identity-based Richtlinien

Identity-based Richtlinien sind Richtliniendokumente für JSON-Berechtigungen, die Sie an eine Identität (Benutzer, Gruppe oder Rolle) anhängen. Diese Richtlinien steuern, welche Aktionen Identitäten für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Identity-based Richtlinien können Inline-Richtlinien (direkt in eine einzelne Identität eingebettet) oder verwaltete Richtlinien (eigenständige Richtlinien, die mehreren Identitäten zugeordnet sind) sein. Informationen dazu, wie Sie zwischen verwalteten und Inline-Richtlinien wählen, finden Sie unter Choose between managed policies and inline policies im IAM-Benutzerhandbuch.

Resource-based Richtlinien

Resource-based Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anhängen. Beispiele hierfür sind Vertrauensrichtlinien für IAM-Rollen und Amazon S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben.

Resource-based Richtlinien sind Inline-Richtlinien, die sich in diesem Dienst befinden. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche Richtlinientypen, mit denen die maximalen Berechtigungen festgelegt werden können, die durch gängigere Richtlinientypen gewährt werden:

- Berechtigungsgrenzen – Eine Berechtigungsgrenze legt die maximalen Berechtigungen fest, die eine identitätsbasierte Richtlinie einer IAM-Entität erteilen kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im -IAM-Benutzerhandbuch.
- Service-Kontrollrichtlinien (SCPs) – SCPs legen die maximalen Berechtigungen für eine Organisation oder Organisationseinheit in AWS Organizations fest. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Ressourcen-Kontrollrichtlinien (RCPs) – RCPs definieren die maximale Anzahl an Berechtigungen, die Ressourcen in Ihren Konten zur Verfügung stehen. Weitere Informationen finden Sie unter Ressourcen-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die als Parameter übergeben werden, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere Arten von Richtlinien für eine Anfrage gelten, sind die sich daraus ergebenden Berechtigungen schwieriger zu verstehen. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie unter Bewertungslogik für Richtlinien im IAM-Benutzerhandbuch.

So funktioniert Amazon Keyspaces mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Amazon Keyspaces zu verwalten, sollten Sie wissen, welche IAM-Funktionen für Amazon Keyspaces verfügbar sind. Einen allgemeinen Überblick darüber, wie Amazon Keyspaces und andere AWS Dienste mit IAM funktionieren, finden Sie im AWS IAM-Benutzerhandbuch unter Dienste, die mit IAM funktionieren.

Themen

- Identitätsbasierte Richtlinien von Amazon Keyspaces
- Ressourcenbasierte Richtlinien von Amazon Keyspaces
- Autorisierung basierend auf Amazon Keyspaces-Tags
- IAM-Rollen in Amazon Keyspaces

Identitätsbasierte Richtlinien von Amazon Keyspaces

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Amazon Keyspaces unterstützt bestimmte Aktionen und Ressourcen sowie Bedingungsschlüssel. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Die servicespezifischen Ressourcen und Aktionen von Amazon Keyspaces sowie Bedingungskontextschlüssel, die für IAM-Berechtigungsrichtlinien verwendet werden können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Keyspaces (für Apache Cassandra) in der Service Authorization Reference.

Aktionen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Nehmen Sie Aktionen in eine Richtlinie auf, um Berechtigungen zur Ausführung des zugehörigen Vorgangs zu erteilen.

Richtlinienaktionen in Amazon Keyspaces verwenden das folgende Präfix vor der Aktion:`cassandra:`. Um beispielsweise jemandem die Erlaubnis zu erteilen, einen Amazon Keyspaces-Schlüsselraum mit der Amazon `CREATE` Keyspaces-CQL-Anweisung zu erstellen, nehmen Sie die `cassandra:Create` Aktion in seine Richtlinie auf. Richtlinienanweisungen müssen entweder ein – `Action`oder ein `NotAction`-Element enthalten. Amazon Keyspaces definiert eigene Aktionen, die Aufgaben beschreiben, die Sie mit diesem Service ausführen können.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie wie folgt durch Kommata:

```
"Action": [
      "cassandra:CREATE",
      "cassandra:MODIFY"
          ]
```

Eine Liste der Amazon Keyspaces-Aktionen finden Sie unter Von Amazon Keyspaces definierte Aktionen (für Apache Cassandra) in der Service Authorization Reference.

Ressourcen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Als Best Practice geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

In Amazon Keyspaces können Schlüsselräume, Tabellen und Streams im `Resource` Element der IAM-Berechtigungen verwendet werden.

> **Note**
>
> Um auf Benutzerschlüsselbereiche und Tabellen in Amazon Keyspaces zuzugreifen, muss Ihre IAM-Richtlinie `cassandra:Select` Berechtigungen für Systemtabellen enthalten:
>
> ```
> arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
> ```

Dies gilt für die folgenden Szenarien:

- AWS Zugriff auf die Managementkonsole
- SDK-Ressourcenoperationen, zum Beispiel `GetKeyspaceGetTable`,`ListKeyspaces`, und `ListTables`
- Standardmäßige Apache Cassandra-Clienttreiberverbindungen, da Treiber bei der Verbindungsinitialisierung automatisch Systemtabellen lesen

Systemtabellen sind schreibgeschützt und können nicht geändert werden.

Die Amazon Keyspace-Ressource hat den folgenden ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/
```

Die Amazon Keyspaces-Tabellenressource hat den folgenden ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}
```

Die Amazon Keyspaces-Stream-Ressource hat den folgenden ARN:

```
arn:${Partition}:cassandra:{Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

Weitere Informationen zum Format von ARNs finden Sie unter Amazon Resource Names (ARNs) and AWS Service Namespaces.

Um beispielsweise den `mykeyspace` Schlüsselraum in Ihrer Anweisung anzugeben, verwenden Sie den folgenden ARN:

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
```

Um alle Schlüsselräume anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie den Platzhalter (*):

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/*"
```

Einige Amazon Keyspaces-Aktionen, z. B. zum Erstellen von Ressourcen, können nicht für eine bestimmte Ressource ausgeführt werden. In diesen Fällen müssen Sie den Platzhalter (*) verwenden.

```
"Resource": "*"
```

Um beispielsweise einem IAM-Prinzipal `SELECT` Berechtigungen für `mytable` in zu erteilen`mykeyspace`, muss der Principal sowohl über Leseberechtigungen als auch verfügen. `mytable` `keyspace/system*` Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie die ARNs durch Kommata voneinander.

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
```

Eine Liste der Amazon Keyspaces-Ressourcentypen und ihrer ARNs finden Sie unter Von Amazon Keyspaces definierte Ressourcen (für Apache Cassandra) in der Service Authorization Reference. Informationen darüber, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter Von Amazon Keyspaces definierte Aktionen (für Apache Cassandra).

Bedingungsschlüssel

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Condition` gibt an, wann Anweisungen auf der Grundlage definierter Kriterien ausgeführt werden. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Amazon Keyspaces definiert seinen eigenen Satz von Bedingungsschlüsseln und unterstützt auch die Verwendung einiger globaler Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontext-Schlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Alle Amazon Keyspaces-Aktionen unterstützen die Schlüssel `aws:RequestTag/${TagKey}aws:ResourceTag/${TagKey}`, die und die `aws:TagKeys` Bedingungstasten. Weitere Informationen finden Sie unter the section called “Amazon Keyspaces-Ressourcenzugriff basierend auf Tags”.

Eine Liste der Amazon Keyspaces-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Keyspaces (für Apache Cassandra) in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von Amazon Keyspaces definierte Aktionen (für Apache Cassandra).

Beispiele

Beispiele für identitätsbasierte Richtlinien von Amazon Keyspaces finden Sie unter. Beispiele für identitätsbasierte Richtlinien von Amazon Keyspaces

Ressourcenbasierte Richtlinien von Amazon Keyspaces

Amazon Keyspaces unterstützt keine ressourcenbasierten Richtlinien. Ein Beispiel für eine detaillierte Seite zu ressourcenbasierten Richtlinien finden Sie unter https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html.

Autorisierung basierend auf Amazon Keyspaces-Tags

Sie können den Zugriff auf Ihre Amazon Keyspaces-Ressourcen mithilfe von Tags verwalten. Um den Ressourcenzugriff auf der Grundlage von Tags zu verwalten, geben Sie Tag-Informationen im Bedingungselement einer Richtlinie mithilfe der `aws:TagKeys` Bedingungsschlüssel `cassandra:ResourceTag/key-name`aws:RequestTag/*key-name*, oder ein. Weitere Informationen zum Taggen von Amazon Keyspaces-Ressourcen finden Sie unter. the section called “Mit Tags arbeiten”

Ein Beispiel für eine identitätsbasierte Richtlinie zur Einschränkung des Zugriffs auf eine Ressource auf der Grundlage der Markierungen dieser Ressource finden Sie unter Amazon Keyspaces-Ressourcenzugriff basierend auf Tags.

IAM-Rollen in Amazon Keyspaces

Eine IAM-Rolle ist eine Entität innerhalb von Ihnen AWS-Konto , die über bestimmte Berechtigungen verfügt.

Temporäre Anmeldeinformationen mit Amazon Keyspaces verwenden

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Sie erhalten temporäre Sicherheitsanmeldedaten, indem Sie AWS STS API-Operationen wie AssumeRoleoder GetFederationTokenaufrufen.

Amazon Keyspaces unterstützt die Verwendung temporärer Anmeldeinformationen mit dem AWS Signature Version 4 (Sigv4) -Authentifizierungs-Plugin, das im Github-Repo für die folgenden Sprachen verfügbar ist:

- Java:. https://github.com/aws/aws-sigv4-auth-cassandra-java-driver-plugin
- Node.js: https://github.com/aws/aws-sigv4-auth-cassandra-nodejs-driver-plugin.
- Python: https://github.com/aws/aws-sigv4-auth-cassandra-python-driver-plugin.
- Geh:https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Beispiele und Tutorials, die das Authentifizierungs-Plugin für den programmgesteuerten Zugriff auf Amazon Keyspaces implementieren, finden Sie unter. the section called “Verwenden eines Cassandra-Client-Treibers”

Service-linked Rollen

Service-linked Rollen ermöglichen es AWS Diensten, auf Ressourcen in anderen Diensten zuzugreifen, um eine Aktion in Ihrem Namen auszuführen. Service-linked Rollen werden in Ihrem IAM-Konto angezeigt und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.

Einzelheiten zum Erstellen oder Verwalten von serviceverknüpften Amazon Keyspaces-Rollen finden Sie unter. the section called “Verwenden von servicegebundenen Rollen”

Servicerollen

Amazon Keyspaces unterstützt keine Servicerollen.

Beispiele für identitätsbasierte Richtlinien von Amazon Keyspaces

Standardmäßig sind IAM-Benutzer und -Rollen nicht berechtigt, Amazon Keyspaces-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben über die Konsole, CQLSH oder API ausführen. AWS CLI AWS Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen

Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von Richtlinien auf der JSON-Registerkarte im IAM-Benutzerhandbuch.

Themen

- Best Practices für Richtlinien
- Verwenden der Amazon Keyspaces-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Zugreifen auf Amazon Keyspaces-Tabellen
- Amazon Keyspaces-Ressourcenzugriff basierend auf Tags

Best Practices für Richtlinien

Identity-based Richtlinien legen fest, ob jemand Amazon Keyspaces-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder diese löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Wenn Sie identitätsbasierte Richtlinien erstellen oder bearbeiten, befolgen Sie diese Richtlinien und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien oder Von AWS verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.

- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung mit IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Best Practices für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der Amazon Keyspaces-Konsole

Amazon Keyspaces benötigt keine speziellen Berechtigungen für den Zugriff auf die Amazon Keyspaces-Konsole. Sie benötigen mindestens Leseberechtigungen, um Details zu den Amazon Keyspaces-Ressourcen in Ihrem aufzulisten und anzuzeigen. AWS-Konto Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (IAM-Benutzer oder -Rollen) mit dieser Richtlinie.

Den Entitäten stehen zwei AWS verwaltete Richtlinien für den Zugriff auf die Amazon Keyspaces-Konsole zur Verfügung.

- AmazonKeyspacesReadOnlyAccess_v2 — Diese Richtlinie gewährt nur Lesezugriff auf Amazon Keyspaces.

- AmazonKeyspacesFullAccess— Diese Richtlinie gewährt Berechtigungen zur Nutzung von Amazon Keyspaces mit vollem Zugriff auf alle Funktionen.

Weitere Informationen zu den verwalteten Richtlinien von Amazon Keyspaces finden Sie unterthe section called “AWS verwaltete Richtlinien”.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der AWS CLI API oder. AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Zugreifen auf Amazon Keyspaces-Tabellen

> **Note**
>
> Um auf Benutzerschlüsselbereiche und Tabellen in Amazon Keyspaces zuzugreifen, muss Ihre IAM-Richtlinie `cassandra:Select` Berechtigungen für Systemtabellen enthalten:
>
> ```
> arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
> ```
>
> Dies gilt für die folgenden Szenarien:
>
> - AWS Zugriff auf die Managementkonsole
> - SDK-Ressourcenoperationen, zum Beispiel `GetKeyspaceGetTable`,`ListKeyspaces`, und `ListTables`
> - Standardmäßige Apache Cassandra-Clienttreiberverbindungen, da Treiber bei der Verbindungsinitialisierung automatisch Systemtabellen lesen
>
> Systemtabellen sind schreibgeschützt und können nicht geändert werden.

Im Folgenden finden Sie eine Beispielrichtlinie, die nur Lesezugriff (`SELECT`) auf die Amazon Keyspaces-Systemtabellen gewährt. Ersetzen Sie für alle Beispiele die Region und die Konto-ID im Amazon-Ressourcennamen (ARN) durch Ihre eigenen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
```

```
      }
   ]
}
```

Die folgende Beispielrichtlinie fügt schreibgeschützten Zugriff auf die Benutzertabelle `mytable` im Schlüsselraum hinzu. `mykeyspace`

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Die folgende Beispielrichtlinie weist read/write Zugriff auf eine Benutzertabelle und Lesezugriff auf die Systemtabellen zu.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select",
            "cassandra:Modify"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
```

```
   ]
}
```

Die folgende Beispielrichtlinie ermöglicht es einem Benutzer, Tabellen im `mykeyspace` Schlüsselraum zu erstellen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Create",
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/*",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

Die folgende Beispielrichtlinie weist den Systemtabellen Lesezugriff zu, schränkt jedoch den `SELECT` (Lese-) und `MODIFY` (Schreib-) Zugriff auf die Benutzertabelle ein. `mytable`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": [
        "cassandra:Select",
```

```
        "cassandra:Modify"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/mytable"
      ]
    }
  ]
}
```

Amazon Keyspaces-Ressourcenzugriff basierend auf Tags

Sie können Bedingungen in Ihrer identitätsbasierten Richtlinie verwenden, um den Zugriff auf Amazon Keyspaces-Ressourcen anhand von Tags zu steuern. Diese Richtlinien steuern die Sichtbarkeit der Schlüsselbereiche und Tabellen im Konto. Beachten Sie, dass sich tagbasierte Berechtigungen für Systemtabellen anders verhalten, wenn Anfragen über das AWS SDK gestellt werden, als bei API-Aufrufen der Cassandra Query Language (CQL) über Cassandra-Treiber und Entwicklertools.

- Um Anfragen mit dem AWS SDK stellen `List` und `Get` Ressourcen bereitstellen zu können, wenn der Tag-basierte Zugriff verwendet wird, benötigt der Aufrufer Lesezugriff auf Systemtabellen. Beispielsweise sind `Select` Aktionsberechtigungen erforderlich, um mithilfe der Operation Daten aus Systemtabellen zu lesen. `GetTable` Wenn der Aufrufer nur Tag-basierten Zugriff auf eine bestimmte Tabelle hat, schlägt ein Vorgang fehl, der zusätzlichen Zugriff auf eine Systemtabelle erfordert.
- Aus Gründen der Kompatibilität mit dem etablierten Cassandra-Treiberverhalten werden tagbasierte Autorisierungsrichtlinien nicht durchgesetzt, wenn Operationen an Systemtabellen mithilfe von Cassandra Query Language (CQL) -API-Aufrufen über Cassandra-Treiber und Entwicklertools ausgeführt werden.

Das folgende Beispiel zeigt, wie Sie eine Richtlinie erstellen können, die einem Benutzer Berechtigungen zum Anzeigen einer Tabelle gewährt, wenn die Tabelle den Wert des Benutzernamens dieses Benutzers `Owner` enthält. In diesem Beispiel gewähren Sie auch Lesezugriff auf die Systemtabellen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ReadOnlyAccessTaggedTables",
```

```
         "Effect":"Allow",
         "Action":"cassandra:Select",
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/*",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ],
         "Condition":{
            "StringEquals":{
               "aws:ResourceTag/Owner":"${aws:username}"
            }
         }
      }
   ]
}
```

Sie können diese Richtlinie den IAM-Benutzern in Ihrem Konto anfügen. Wenn ein benannter Benutzer `richard-roe` versucht, eine Amazon Keyspaces-Tabelle aufzurufen, muss die Tabelle mit `Owner=richard-roe` oder `owner=richard-roe` gekennzeichnet sein. Andernfalls wird der Zugriff abgelehnt. Der Tag-Schlüssel `Owner` der Bedingung stimmt sowohl mit `Owner` als auch mit `owner` überein, da die Namen von Bedingungsschlüsseln nicht zwischen Groß- und Kleinschreibung unterscheiden. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.

Die folgende Richtlinie gewährt einem Benutzer die Erlaubnis, Tabellen mit Tags zu erstellen, wenn die Tabelle den Wert des Benutzernamens dieses Benutzers `Owner` enthält.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
          "Sid": "CreateTagTableUser",
          "Effect": "Allow",
          "Action": [
              "cassandra:Create",
              "cassandra:TagResource"
          ],
          "Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/*",
          "Condition":{
             "StringEquals":{
                "aws:RequestTag/Owner":"${aws:username}"
            }
```

```
            }
        }
    ]
}
```

AWS verwaltete Richtlinien für Amazon Keyspaces

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet wird AWS. AWS Verwaltete Richtlinien sind so konzipiert, dass sie Berechtigungen für viele gängige Anwendungsfälle bereitstellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie für alle AWS Kunden verfügbar sind. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn eine neue Richtlinie eingeführt AWS-Service wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS verwaltete Richtlinie: AmazonKeyspacesReadOnlyAccess _v2

Sie können die `AmazonKeyspacesReadOnlyAccess_v2`-Richtlinie an Ihre IAM-Identitäten anfügen.

Diese Richtlinie gewährt nur Lesezugriff auf Amazon Keyspaces und beinhaltet die erforderlichen Berechtigungen, wenn eine Verbindung über private VPC-Endpunkte hergestellt wird.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen.

- `Amazon Keyspaces`— Bietet schreibgeschützten Zugriff auf Amazon Keyspaces.
- `Amazon Keyspaces CDC streams`— Ermöglicht Prinzipalen das Ansehen von Amazon Keyspaces CDC-Streams.
- `Application Auto Scaling` – Ermöglicht Prinzipalen, Konfigurationen Application Auto Scaling anzuzeigen. Dies ist erforderlich, damit Benutzer Richtlinien für die automatische Skalierung anzeigen können, die an eine Tabelle angehängt sind.
- `CloudWatch`— Ermöglicht Prinzipalen das Anzeigen von Metrikdaten und Alarmen, die in konfiguriert sind. CloudWatch Dies ist erforderlich, damit Benutzer die Größe der Fakturierungstabelle und die CloudWatch Alarme, die für eine Tabelle konfiguriert wurden, einsehen können.
- `AWS KMS`— Ermöglicht es den Hauptbenutzern, die in konfigurierten Schlüssel anzuzeigen. AWS KMS Dies ist erforderlich, damit Benutzer die AWS KMS Schlüssel einsehen können, die sie in ihrem Konto erstellen und verwalten, um zu bestätigen, dass es sich bei dem Amazon Keyspaces zugewiesenen Schlüssel um einen aktivierten symmetrischen Verschlüsselungsschlüssel handelt.
- `Amazon EC2`— Ermöglicht Principals, die sich über VPC-Endpunkte mit Amazon Keyspaces verbinden, die VPC auf Ihrer Amazon EC2 EC2-Instance nach Endpunkt- und Netzwerkschnittstelleninformationen abzufragen. Dieser schreibgeschützte Zugriff auf die Amazon EC2 EC2-Instance ist erforderlich, damit Amazon Keyspaces verfügbare Schnittstellen-VPC-Endpunkte in der `system.peers` Tabelle suchen und speichern kann, die für den Verbindungslastenausgleich verwendet wird.

Um die Richtlinie im Format zu überprüfen, siehe _v2. `JSON` AmazonKeyspacesReadOnlyAccess

AWS verwaltete Richtlinie: AmazonKeyspacesReadOnlyAccess

Sie können die `AmazonKeyspacesReadOnlyAccess`-Richtlinie an Ihre IAM-Identitäten anfügen.

Diese Richtlinie gewährt nur Lesezugriff auf Amazon Keyspaces.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen.

- `Amazon Keyspaces`— Bietet schreibgeschützten Zugriff auf Amazon Keyspaces.
- `Amazon Keyspaces CDC streams`— Ermöglicht Prinzipalen das Ansehen von Amazon Keyspaces CDC-Streams.
- `Application Auto Scaling` – Ermöglicht Prinzipalen, Konfigurationen Application Auto Scaling anzuzeigen. Dies ist erforderlich, damit Benutzer Richtlinien für die automatische Skalierung anzeigen können, die an eine Tabelle angehängt sind.
- `CloudWatch`— Ermöglicht Prinzipalen das Anzeigen von Metrikdaten und Alarmen, die in konfiguriert sind. CloudWatch Dies ist erforderlich, damit Benutzer die Größe der Fakturierungstabelle und die CloudWatch Alarme, die für eine Tabelle konfiguriert wurden, einsehen können.
- `AWS KMS`— Ermöglicht es den Hauptbenutzern, die in konfigurierten Schlüssel anzuzeigen. AWS KMS Dies ist erforderlich, damit Benutzer die AWS KMS Schlüssel einsehen können, die sie in ihrem Konto erstellen und verwalten, um zu bestätigen, dass es sich bei dem Amazon Keyspaces zugewiesenen Schlüssel um einen aktivierten symmetrischen Verschlüsselungsschlüssel handelt.

Informationen zur `JSON` Formatierung der Richtlinie finden Sie unter.
AmazonKeyspacesReadOnlyAccess

AWS verwaltete Richtlinie: AmazonKeyspacesFullAccess

Sie können die `AmazonKeyspacesFullAccess`-Richtlinie an Ihre IAM-Identitäten anfügen.

Diese Richtlinie gewährt Administratorberechtigungen, die Ihren Administratoren uneingeschränkten Zugriff auf Amazon Keyspaces ermöglichen.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen.

- `Amazon Keyspaces`— Ermöglicht Prinzipalen den Zugriff auf jede Amazon Keyspaces-Ressource und die Ausführung aller Aktionen.
- `Application Auto Scaling`— Ermöglicht Prinzipalen das Erstellen, Anzeigen und Löschen automatischer Skalierungsrichtlinien für Amazon Keyspaces-Tabellen. Dies ist erforderlich, damit Administratoren automatische Skalierungsrichtlinien für Amazon Keyspaces-Tabellen verwalten können.
- `CloudWatch`— Ermöglicht Prinzipalen, die Größe der fakturierbaren Tabelle zu sehen sowie CloudWatch Alarme für automatische Skalierungsrichtlinien von Amazon Keyspace zu erstellen, anzuzeigen und zu löschen. Dies ist erforderlich, damit Administratoren die Größe der fakturierbaren Tabelle einsehen und ein Dashboard erstellen können. CloudWatch
- `IAM`— Ermöglicht Amazon Keyspaces, automatisch serviceverknüpfte Rollen mit IAM zu erstellen, wenn die folgenden Funktionen aktiviert sind:
  - `Amazon Keyspaces CDC streams`— Wenn ein Administrator einen Stream für eine Tabelle aktiviert, erstellt Amazon Keyspaces die serviceverknüpfte Rolle, AWSServiceRoleForAmazonKeyspacesCDCum in Ihrem Namen CloudWatch Metriken in Ihrem Konto zu veröffentlichen.
  - `Application Auto Scaling`— Wenn ein Administrator Application Auto Scaling für eine Tabelle aktiviert, erstellt Amazon Keyspaces die serviceverknüpfte Rolle AWSServiceRoleForApplicationAutoScaling_, CassandraTable um automatische Skalierungsaktionen in Ihrem Namen durchzuführen.
  - `Amazon Keyspaces multi-Region replication`— Wenn ein Administrator einen neuen Schlüsselraum für mehrere Regionen erstellt oder einem vorhandenen Schlüsselraum mit einer Region einen neuen AWS-Region hinzufügt, erstellt Amazon Keyspaces die serviceverknüpfte Rolle, AWSServiceRoleForAmazonKeyspacesReplicationum die Replikation von Tabellen, Daten und Metadaten in den ausgewählten Regionen in Ihrem Namen durchzuführen.
- `AWS KMS` – Ermöglicht es Prinzipalen, in AWS KMS konfigurierte Schlüssel anzuzeigen. Dies ist erforderlich, damit Benutzer die AWS KMS Schlüssel einsehen können, die sie in ihrem Konto erstellen und verwalten, um zu bestätigen, dass es sich bei dem Amazon Keyspaces zugewiesenen Schlüssel um einen aktivierten symmetrischen Verschlüsselungsschlüssel handelt.
- `Amazon EC2`— Ermöglicht Principals, die sich über VPC-Endpunkte mit Amazon Keyspaces verbinden, die VPC auf Ihrer Amazon EC2 EC2-Instance nach Endpunkt- und Netzwerkschnittstelleninformationen abzufragen. Dieser schreibgeschützte Zugriff auf die Amazon EC2 EC2-Instance ist erforderlich, damit Amazon Keyspaces verfügbare Schnittstellen-VPC-Endpunkte in der `system.peers` Tabelle suchen und speichern kann, die für den Verbindungslastenausgleich verwendet wird.

Informationen zur Formatierung der Richtlinie finden Sie unter. `JSON` AmazonKeyspacesFullAccess

AWS verwaltete Richtlinie: KeyspacesCDCServiceRolePolicy

Sie können `KeyspacesCDCServiceRolePolicy` nicht an Ihre IAM-Entitäten anhängen. Diese Richtlinie ist mit einer dienstbezogenen Rolle verknüpft, die es Amazon Keyspaces ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “CDC-Streams”.

Diese Richtlinie gewährt der serviceverknüpften Rolle die erforderlichen BerechtigungenAWSServiceRoleForAmazonKeyspacesCDC, um Amazon Keyspaces CDC-Stream-Metrikdaten in Ihrem Namen zu CloudWatch veröffentlichen.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen.

- `CloudWatch`— Ermöglicht der serviceverknüpften Rolle AWSServiceRoleForAmazonKeyspacesCDC, in Ihrem Namen Metrikdaten aus Amazon Keyspaces CDC-Streams `"cloudwatch:namespace": "AWS/Cassandra"` in Ihrem CloudWatch Konto zu veröffentlichen.

Informationen zum Format der Richtlinie finden Sie unter. `JSON` KeyspacesCDCServiceRolePolicy

Amazon Keyspaces-Aktualisierungen der AWS verwalteten Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Amazon Keyspaces an, seit dieser Service begonnen hat, diese Änderungen zu verfolgen. Um automatische Warnungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der Dokumentverlauf-Seite.

Änderungen	Beschreibung	Date
KeyspacesCDCServiceRolePolicy – Neue Richtlinie	Amazon Keyspaces hat eine neue verwaltete Richtlinie hinzugefügt `KeyspacesCDCServiceRolePolicy` , die der serviceverknüpften Rolle die erforderlichen Berechtigungen erteilt, `AWSServiceRoleForAmazonKeyspacesCDC` um Amazon Keyspaces CDC-Stream-Metrikdaten in Ihrem Namen zu CloudWatch veröffentlichen. Weitere Informationen finden Sie unter the section called “CDC-Streams”.	02. Juli 2025
AmazonKeyspacesReadOnlyAccess_v2 — Aktualisierung einer bestehenden Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, um es IAM-Prinzipalen zu ermöglichen, Amazon Keyspaces CDC-Streams anzusehen. Weitere Informationen finden Sie unter the section called “CDC-Streams anzeigen”.	02. Juli 2025
AmazonKeyspacesReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, um es IAM-Prinzipalen zu ermöglichen, Amazon Keyspaces CDC-Streams anzusehen. Weitere Informationen finden Sie unter the	02. Juli 2025

Änderungen	Beschreibung	Date
	section called “CDC-Streams anzeigen”.	
AmazonKeyspacesFullAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat die `KeyspacesCDCServiceRolePolicy` verwaltete Richtlinie für die serviceverknüpfte Rolle erstellt AWSServiceRoleForAmazonKeyspacesCDC, um die Berechtigungen hinzuzufügen, die erforderlich sind, wenn ein Administrator einen Stream für eine Tabelle aktiviert. Amazon Keyspaces verwendet die serviceverknüpfte Rolle`AWSServiceRoleForAmazonKeyspacesCDC` , um in Ihrem CloudWatch Namen Kennzahlen in Ihrem Konto zu veröffentlichen. Weitere Informationen finden Sie unter the section called “CDC-Streams”.	02. Juli 2025

Änderungen	Beschreibung	Date
AmazonKeyspacesFullAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat die mit dem `KeyspacesReplicationServiceRolePolicy` Service verknüpfte Rolle aktualisiert AWSServiceRoleForAmazonKeyspacesReplication, um die Berechtigungen hinzuzufügen, die erforderlich sind, wenn ein Administrator einem einzelnen oder mehreren Regionen einen neuen AWS-Region Schlüsselraum hinzufügt. Amazon Keyspaces verwendet die serviceverknüpfte Rolle, `AWSServiceRoleForAmazonKeyspacesReplication` um Tabellen, deren Einstellungen und Daten in Ihrem Namen zu replizieren. Weitere Informationen finden Sie unter the section called “Multi-Region Replikation”.	19. November 2024

Änderungen	Beschreibung	Date
AmazonKeyspacesFullAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, damit Amazon Keyspaces eine dienstbezogene Rolle erstellen kann, wenn ein Administrator einem Schlüsselraum mit einer oder mehreren Regionen eine neue Region hinzufügt. Amazon Keyspaces verwendet die serviceverknüpfte Rolle, um Datenreplikationsaufgaben in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “Multi-Region Replikation”.	3. Oktober 2023

Änderungen	Beschreibung	Date
AmazonKeyspacesReadOnlyAccess_v2 — Neue Richtlinie	Amazon Keyspaces hat eine neue Richtlinie erstellt, um Nur-Lese-Berechtigungen für Clients hinzuzufügen, die sich über VPC-Schnittstellen -Endpunkte mit Amazon Keyspaces verbinden, um auf die Amazon EC2 EC2-Instance zuzugreifen, um Netzwerkinformationen abzurufen. Amazon Keyspaces speichert verfügbare VPC-Schnittstellen-Endpunkte in der `system.peers` Tabelle für den Verbindungslastenausgleich. Weitere Informationen finden Sie unter the section called “Verwendung von -Schnittstellen-VPC-Endpunkten”.	12. September 2023

Änderungen	Beschreibung	Date
AmazonKeyspacesFullAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, die es Amazon Keyspaces ermöglichen, eine servicebezogene Rolle zu erstellen, wenn ein Administrator einen Schlüsselraum mit mehreren Regionen erstellt. Amazon Keyspaces verwendet die serviceverknüpfte Rolle`AWSServiceRoleForAmazonKeyspacesReplication` , um Datenreplikationsaufgaben in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “Multi-Region Replikation”.	5. Juni 2023
AmazonKeyspacesReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, mit denen Benutzer die abrechnungsfähige Größe einer Tabelle mithilfe von anzeigen können. CloudWatch Amazon Keyspaces ist in Amazon integriert CloudWatch , sodass Sie die Größe der fakturierbaren Tabelle überwachen können. Weitere Informationen finden Sie unter the section called “Amazon-Keyspaces-Metriken”.	7. Juli 2022

Änderungen	Beschreibung	Date
AmazonKeyspacesFullAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, mit denen Benutzer die abrechnungsfähige Größe einer Tabelle mithilfe von anzeigen können. CloudWatch Amazon Keyspaces ist in Amazon integriert CloudWatch , sodass Sie die Größe der fakturierbaren Tabelle überwachen können. Weitere Informationen finden Sie unter the section called “Amazon-Keyspaces-Metriken”.	7. Juli 2022

Änderungen	Beschreibung	Date
AmazonKeyspacesReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, um Benutzern das Anzeigen von AWS KMS Schlüsseln zu ermöglichen, die für die Amazon Keyspaces-Verschlüsselung im Ruhezustand konfiguriert wurden. Amazon Keyspaces Encryption at Rest lässt sich integrieren, um die Verschlüsselungsschlüssel zu schützen und zu verwalten, die zur Verschlüsselung von Daten im Ruhezustand verwendet werden. AWS KMS Um den für Amazon Keyspaces konfigurierten AWS KMS Schlüssel anzuzeigen, wurden Nur-Lese-Berechtigungen hinzugefügt.	1. Juni 2021

Änderungen	Beschreibung	Date
AmazonKeyspacesFullAccess – Aktualisierung auf eine bestehende Richtlinie	Amazon Keyspaces hat neue Berechtigungen hinzugefügt, um Benutzern das Anzeigen von AWS KMS Schlüsseln zu ermöglichen, die für die Amazon Keyspaces-Verschlüsselung im Ruhezustand konfiguriert wurden. Amazon Keyspaces Encryption at Rest lässt sich integrieren, um die Verschlüsselungsschlüssel zu schützen und zu verwalten, die zur Verschlüsselung von Daten im Ruhezustand verwendet werden. AWS KMS Um den für Amazon Keyspaces konfigurierten AWS KMS Schlüssel anzuzeigen, wurden Nur-Lese-Berechtigungen hinzugefügt.	1. Juni 2021
Amazon Keyspaces hat begonnen, Änderungen zu verfolgen	Amazon Keyspaces hat damit begonnen, Änderungen an seinen AWS verwalteten Richtlinien nachzuverfolgen.	1. Juni 2021

Fehlerbehebung bei Identität und Zugriff auf Amazon Keyspaces

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit Amazon Keyspaces und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion in Amazon Keyspaces durchzuführen

- Ich habe einen IAM-Benutzer oder eine IAM-Rolle geändert und die Änderungen wurden nicht sofort wirksam
- Ich kann eine Tabelle mit Amazon Keyspaces Point-in-Time Recovery (PITR) nicht wiederherstellen
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich bin Administrator und möchte anderen den Zugriff auf Amazon Keyspaces ermöglichen
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Amazon Keyspaces-Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion in Amazon Keyspaces durchzuführen

Wenn Ihnen AWS-Managementkonsole mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion durchzuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator ist die Person, die Ihnen Ihren Benutzernamen und Ihr Passwort zur Verfügung gestellt hat.

Der folgende Beispielfehler tritt auf, wenn der `mateojackson` IAM-Benutzer versucht, die Konsole zu verwenden, um Details zu einer Tabelle anzuzeigen, *`table`* aber keine `cassandra:`*`Select`* Berechtigungen für die Tabelle besitzt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cassandra:Select on resource: mytable
```

In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion *`mytable`* auf die Ressource `cassandra:`*`Select`* zugreifen zu können.

Ich habe einen IAM-Benutzer oder eine IAM-Rolle geändert und die Änderungen wurden nicht sofort wirksam

Es kann bis zu 10 Minuten dauern, bis Änderungen der IAM-Richtlinien für Anwendungen mit bestehenden, etablierten Verbindungen zu Amazon Keyspaces wirksam werden. Änderungen der IAM-Richtlinien werden sofort wirksam, wenn Anwendungen eine neue Verbindung herstellen. Wenn Sie Änderungen an einem vorhandenen IAM-Benutzer oder einer vorhandenen IAM-Rolle vorgenommen haben und diese nicht sofort wirksam wurden, warten Sie entweder 10 Minuten oder trennen Sie die Verbindung und stellen Sie erneut eine Verbindung zu Amazon Keyspaces her.

Ich kann eine Tabelle mit Amazon Keyspaces Point-in-Time Recovery (PITR) nicht wiederherstellen

Wenn Sie versuchen, eine Amazon Keyspaces-Tabelle mit Point-in-Time Recovery (PITR) wiederherzustellen, und Sie sehen, dass der Wiederherstellungsvorgang beginnt, aber nicht erfolgreich abgeschlossen wird, haben Sie möglicherweise nicht alle erforderlichen Berechtigungen konfiguriert, die für den Wiederherstellungsprozess erforderlich sind. Sie müssen sich an Ihren Administrator wenden, um Unterstützung zu erhalten, und diese Person bitten, Ihre Richtlinien zu aktualisieren, damit Sie eine Tabelle in Amazon Keyspaces wiederherstellen können.

Zusätzlich zu den Benutzerberechtigungen benötigt Amazon Keyspaces möglicherweise Berechtigungen, um während des Wiederherstellungsvorgangs im Namen Ihres Auftraggebers Aktionen durchzuführen. Dies ist der Fall, wenn die Tabelle mit einem vom Kunden verwalteten Schlüssel verschlüsselt ist oder wenn Sie IAM-Richtlinien verwenden, die den eingehenden Datenverkehr einschränken. Wenn Sie beispielsweise Bedingungsschlüssel in Ihrer IAM-Richtlinie verwenden, um den Quelldatenverkehr auf bestimmte Endpunkte oder IP-Bereiche zu beschränken, schlägt der Wiederherstellungsvorgang fehl. Damit Amazon Keyspaces die Tabellenwiederherstellung im Namen Ihres Prinzipals durchführen kann, müssen Sie der IAM-Richtlinie einen `aws:ViaAWSService` globalen Bedingungsschlüssel hinzufügen.

Weitere Informationen zu Berechtigungen zum Wiederherstellen von Tabellen finden Sie unter. the section called “Konfigurieren Sie die IAM-Berechtigungen für die Wiederherstellung”

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion durchzuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Amazon Keyspaces übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Service zu übergeben, anstatt eine neue Servicerolle oder eine dienstbezogene Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amazon Keyspaces auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin Administrator und möchte anderen den Zugriff auf Amazon Keyspaces ermöglichen

Um anderen den Zugriff auf Amazon Keyspaces zu ermöglichen, müssen Sie den Personen oder Anwendungen, die Zugriff benötigen, die Erlaubnis erteilen. Wenn Sie Personen und Anwendungen verwalten, weisen Sie Benutzern oder Gruppen Berechtigungssätze zu, um deren Zugriffsebene zu definieren. AWS IAM Identity Center Mit Berechtigungssätzen werden automatisch IAM-Richtlinien erstellt und den IAM-Rollen zugewiesen, die der Person oder Anwendung zugeordnet sind. Weitere Informationen finden Sie im AWS IAM Identity Center Benutzerhandbuch unter Berechtigungssätze.

Wenn Sie IAM Identity Center nicht verwenden, müssen Sie IAM-Entitäten (Benutzer oder Rollen) für die Personen oder Anwendungen erstellen, die Zugriff benötigen. Anschließend müssen Sie der Entität eine Richtlinie beifügen, die ihr die richtigen Berechtigungen in Amazon Keyspaces gewährt. Nachdem die Berechtigungen erteilt wurden, stellen Sie dem Benutzer oder Anwendungsentwickler die Anmeldeinformationen zur Verfügung. Sie werden diese Anmeldeinformationen für den Zugriff verwenden AWS. Weitere Informationen zum Erstellen von IAM-Benutzern, -Gruppen, -Richtlinien und -Berechtigungen finden Sie im IAM-Benutzerhandbuch unter IAM-Identitäten sowie Richtlinien und Berechtigungen in IAM.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Amazon Keyspaces-Ressourcen ermöglichen

Sie können eine Rolle erstellen, mit der Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation auf Ihre Ressourcen zugreifen können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob Amazon Keyspaces diese Funktionen unterstützt, finden Sie unterSo funktioniert Amazon Keyspaces mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Verwenden von serviceverknüpften Rollen für Amazon Keyspaces

Amazon Keyspaces (für Apache Cassandra) verwendet AWS Identity and Access Management (IAM) service-verknüpfte Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon Keyspaces verknüpft ist. Service-linked Rollen sind von Amazon Keyspaces vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Informationen zu anderen Diensten, die dienstbezogene Rollen unterstützen, finden Sie unter AWS Dienste, die mit IAM funktionieren. Suchen Sie dort nach den Diensten, für die in der Service-linked Rollenspalte Ja steht. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Themen

- Rollen für die auto Skalierung von Amazon Keyspaces-Anwendungen verwenden
- Rollen für Amazon Keyspaces Multi-Region Replication verwenden
- Rollen für Amazon Keyspaces CDC-Streams verwenden

Rollen für die auto Skalierung von Amazon Keyspaces-Anwendungen verwenden

Amazon Keyspaces (für Apache Cassandra) verwendet AWS Identity and Access Management (IAM) service-verknüpfte Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle,

die direkt mit Amazon Keyspaces verknüpft ist. Service-linked Rollen sind von Amazon Keyspaces vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle erleichtert die Einrichtung von Amazon Keyspaces, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon Keyspaces definiert die Berechtigungen seiner serviceverknüpften Rollen, und sofern nicht anders definiert, kann nur Amazon Keyspaces seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauensrichtlinie und die Berechtigungsrichtlinie, und diese Berechtigungsrichtlinie kann keiner anderen juristischen Stelle von IAM zugeordnet werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem die zugehörigen Ressourcen gelöscht wurden. Dies schützt Ihre Amazon Keyspaces-Ressourcen, da Sie die Zugriffsberechtigung für die Ressourcen nicht versehentlich entfernen können.

Service-linked Rollenberechtigungen für Amazon Keyspaces

Amazon Keyspaces verwendet die serviceverknüpfte Rolle mit dem Namen AWSServiceRoleForApplicationAutoScaling_ CassandraTable, damit Application Auto Scaling Amazon Keyspaces und Amazon in CloudWatch Ihrem Namen aufrufen kann.

Die mit dem AWSServiceRoleForApplicationAutoScaling CassandraTable Dienst verknüpfte Rolle vertraut darauf, dass die folgenden Dienste die Rolle übernehmen:

- `cassandra.application-autoscaling.amazonaws.com`

Die Rollenberechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für die angegebenen Amazon Keyspaces-Ressourcen durchzuführen:

- Aktion: `cassandra:Select` für `arn:*:cassandra:*:*:/keyspace/system/table/*`
- Aktion:`cassandra:Select` auf die Ressource `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
- Aktion:`cassandra:Select` auf die Ressource `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Aktion:`cassandra:Alter` auf Ressource`arn:*:cassandra:*:*:"*"`

Eine serviceverknüpfte Rolle für Amazon Keyspaces erstellen

Sie müssen keine serviceverknüpfte Rolle für die automatische Skalierung von Amazon Keyspaces manuell erstellen. Wenn Sie Amazon Keyspaces Auto Scaling für eine Tabelle mit der AWS-Managementkonsole, CQL, der oder der AWS API aktivieren AWS CLI, erstellt Application Auto Scaling die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie Amazon Keyspaces Auto Scaling für eine Tabelle aktivieren, erstellt Application Auto Scaling die serviceverknüpfte Rolle erneut für Sie.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet. Weitere Informationen finden Sie unter Eine neue Rolle ist erschienen in meinem. AWS-Konto

Bearbeitung einer serviceverknüpften Rolle für Amazon Keyspaces

Amazon Keyspaces erlaubt es Ihnen nicht, die mit dem CassandraTable Dienst AWSServiceRoleForApplicationAutoScaling _ verknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon Keyspaces

Wenn Sie ein Feature oder einen Service, die bzw. der eine serviceverknüpfte Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise ist keine ungenutzte Entität vorhanden, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch zuerst die automatische Skalierung für alle Tabellen im Konto für alle deaktivieren, AWS-Regionen bevor Sie die serviceverknüpfte Rolle manuell löschen können. Informationen zum Deaktivieren der automatischen Skalierung für Amazon Keyspaces-Tabellen finden Sie unterthe section called “auto Skalierung von Amazon Keyspaces für eine Tabelle ausschalten”.

> **Note**
>
> Wenn die automatische Skalierung von Amazon Keyspace die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu ändern, schlägt die Abmeldung möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

So löschen Sie die serviceverknüpfte Rolle mit IAM

Verwenden Sie die IAM-Konsole, die oder die AWS API AWS CLI, um die mit dem Dienst _ verknüpfte Rolle zu löschen. AWSServiceRoleForApplicationAutoScaling CassandraTable Weitere Informationen finden Sie unter Löschen einer Service-Linked Rolle im IAM-Benutzerhandbuch.

> **Note**
>
> Um die serviceverknüpfte Rolle zu löschen, die von der automatischen Skalierung von Amazon Keyspace verwendet wird, müssen Sie zunächst die automatische Skalierung für alle Tabellen im Konto deaktivieren.

Unterstützte Regionen für serviceverknüpfte Amazon Keyspaces-Rollen

Amazon Keyspaces unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter Service-Endpunkte für Amazon Keyspaces.

Rollen für Amazon Keyspaces Multi-Region Replication verwenden

Amazon Keyspaces (für Apache Cassandra) verwendet AWS Identity and Access Management (IAM) service-verknüpfte Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon Keyspaces verknüpft ist. Service-linked Rollen sind von Amazon Keyspaces vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle erleichtert die Einrichtung von Amazon Keyspaces, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon Keyspaces definiert die Berechtigungen seiner serviceverknüpften Rollen, und sofern nicht anders definiert, kann nur Amazon Keyspaces seine Rollen übernehmen. Die definierten Berechtigungen umfassen die

Vertrauensrichtlinie und die Berechtigungsrichtlinie, und diese Berechtigungsrichtlinie kann keiner anderen juristischen Stelle von IAM zugeordnet werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem die zugehörigen Ressourcen gelöscht wurden. Dies schützt Ihre Amazon Keyspaces-Ressourcen, da Sie die Zugriffsberechtigung für die Ressourcen nicht versehentlich entfernen können.

Service-linked Rollenberechtigungen für Amazon Keyspaces

Amazon Keyspaces verwendet die mit dem Service verknüpfte Rolle namens AWSServiceRoleForAmazonKeyspacesReplication, um Amazon Keyspaces zu ermöglichen, in Ihrem Namen neue Daten AWS-Regionen zu einem Schlüsselraum hinzuzufügen und Tabellen mit all ihren Daten und Einstellungen in die neue Region zu replizieren. Die Rolle ermöglicht es Amazon Keyspaces auch, Schreibvorgänge in Tabellen in allen Regionen in Ihrem Namen zu replizieren.

Die AWSServiceRoleForAmazonKeyspacesReplication serviceverknüpfte Rolle vertraut darauf, dass die folgenden Dienste die Rolle übernehmen:

- `replication.cassandra.amazonaws.com`

Die genannte Rollenberechtigungsrichtlinie KeyspacesReplicationServiceRolePolicy ermöglicht es Amazon Keyspaces, die folgenden Aktionen durchzuführen:

- Aktion: `cassandra:Select`
- Aktion: `cassandra:SelectMultiRegionResource`
- Aktion: `cassandra:Modify`
- Aktion: `cassandra:ModifyMultiRegionResource`
- Aktion: `cassandra:AlterMultiRegionResource`
- Aktion: `application-autoscaling:RegisterScalableTarget` — Amazon Keyspaces verwendet die Auto Scaling-Berechtigungen der Anwendung, wenn Sie im Bereitstellungsmodus mit aktiviertem Auto Scaling ein Replikat zu einer einzelnen Regionstabelle hinzufügen.
- Aktion: `application-autoscaling:DeregisterScalableTarget`
- Aktion: `application-autoscaling:DescribeScalableTargets`
- Aktion: `application-autoscaling:PutScalingPolicy`
- Aktion: `application-autoscaling:DescribeScalingPolicies`
- Aktion: `cassandra:Alter`

- Aktion: `cloudwatch:DeleteAlarms`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`

Obwohl die mit dem Service verknüpfte Amazon Keyspaces-Rolle die folgenden Berechtigungen AWSServiceRoleForAmazonKeyspacesReplication bereitstellt: „Action:“ für den angegebenen Amazon-Ressourcennamen (ARN) „arn: *“ in der Richtlinie, Amazon Keyspaces stellt den ARN Ihres Kontos bereit.

Die Berechtigungen zum Erstellen der dienstbezogenen Rolle AWSServiceRoleForAmazonKeyspacesReplication sind in der verwalteten Richtlinie enthalten. `AmazonKeyspacesFullAccess` Weitere Informationen finden Sie unter the section called “AmazonKeyspacesFullAccess”.

Sie müssen Berechtigungen konfigurieren, damit eine Benutzer, Gruppen oder Rollen eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen können. Weitere Informationen finden Sie unter Service-linked Rollenberechtigungen im IAM-Benutzerhandbuch.

Eine serviceverknüpfte Rolle für Amazon Keyspaces erstellen

Sie können eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie in der AWS-Managementkonsole, der oder der AWS API einen Schlüsselraum für mehrere Regionen erstellen AWS CLI, erstellt Amazon Keyspaces die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Schlüsselraum für mehrere Regionen erstellen, erstellt Amazon Keyspaces die serviceverknüpfte Rolle erneut für Sie.

Bearbeitung einer serviceverknüpften Rolle für Amazon Keyspaces

Amazon Keyspaces erlaubt es Ihnen nicht, die AWSServiceRoleForAmazonKeyspacesReplication serviceverknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon Keyspaces

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch zuerst alle Schlüsselräume für mehrere Regionen im Konto für alle löschen, AWS-Regionen bevor Sie die serviceverknüpfte Rolle manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor Sie IAM verwenden können, um eine dienstverknüpfte Rolle zu löschen, müssen Sie zunächst alle von der Rolle verwendeten Schlüsselräume und Tabellen mit mehreren Regionen löschen.

> **Note**
>
> Wenn der Amazon Keyspaces-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

Um Amazon Keyspaces-Ressourcen zu löschen, die von der AWSServiceRoleForAmazonKeyspacesReplication (Konsole) verwendet werden

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon Keyspaces-Konsole unter https://console.aws.amazon.com/keyspaces/home.
2. Wählen Sie in der linken Seitenleiste die Option „Keyspaces“.
3. Wählen Sie alle Schlüsselräume mit mehreren Regionen aus der Liste aus.
4. Wählen Sie Löschen, bestätigen Sie den Löschvorgang und wählen Sie Schlüsselräume löschen.

Sie können Schlüsselräume mit mehreren Regionen auch programmgesteuert löschen, indem Sie eine der folgenden Methoden verwenden.

- Die Cassandra Query Language (CQL) -Anweisung. ???
- Der Delete-Keyspace-Vorgang der CLI. AWS
- Der DeleteKeyspaceBetrieb der Amazon Keyspaces-API.

Manuelles Löschen der serviceverknüpften Rolle

Verwenden Sie die IAM-Konsole, die oder die AWS API AWS CLI, um die AWSServiceRoleForAmazonKeyspacesReplication serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte Amazon Keyspaces-Rollen

Amazon Keyspaces unterstützt nicht die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Sie können die AWSServiceRoleForAmazonKeyspacesReplication Rolle in den folgenden Regionen verwenden.

Name der Region	Regions-ID	Support in Amazon Keyspaces
USA Ost (Nord-Virginia)	us-east-1	Ja
USA Ost (Ohio)	us-east-2	Ja
USA West (Nordkalifornien)	us-west-1	Ja
USA West (Oregon)	us-west-2	Ja
Asien-Pazifik (Mumbai)	ap-south-1	Ja
Asien-Pazifik (Osaka)	ap-northeast-3	Ja
Asien-Pazifik (Seoul)	ap-northeast-2	Ja
Asien-Pazifik (Singapore)	ap-southeast-1	Ja
Asien-Pazifik (Sydney)	ap-southeast-2	Ja
Asien-Pazifik (Tokyo)	ap-northeast-1	Ja
Kanada (Zentral)	ca-central-1	Ja
Europa (Frankfurt)	eu-central-1	Ja
Europa (Ireland)	eu-west-1	Ja
Europa (London)	eu-west-2	Ja

Name der Region	Regions-ID	Support in Amazon Keyspaces
Europa (Paris)	eu-west-3	Ja
Afrika (Kapstadt)	af-south-1	Ja
Südamerika (São Paulo)	sa-east-1	Ja
AWS GovCloud (US-East)	us-gov-east-1	Nein
AWS GovCloud (US-West)	us-gov-west-1	Nein

Rollen für Amazon Keyspaces CDC-Streams verwenden

Amazon Keyspaces (für Apache Cassandra) verwendet AWS Identity and Access Management (IAM) service-verknüpfte Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon Keyspaces verknüpft ist. Service-linked Rollen sind von Amazon Keyspaces vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle erleichtert die Einrichtung von Amazon Keyspaces, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon Keyspaces definiert die Berechtigungen seiner serviceverknüpften Rollen, und sofern nicht anders definiert, kann nur Amazon Keyspaces seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können die serviceverknüpfte Rolle nicht löschen.

Service-linked Rollenberechtigungen für Amazon Keyspaces

Amazon Keyspaces verwendet die benannte serviceverknüpfte Rolle AWSServiceRoleForAmazonKeyspacesCDC, um Amazon Keyspaces CDC-Streams zu ermöglichen, in Ihrem Namen CloudWatch Metriken auf Ihrem Konto zu veröffentlichen.

Die AWSServiceRoleForAmazonKeyspacesCDC serviceverknüpfte Rolle vertraut darauf, dass der folgende Service die Rolle übernimmt:

- `cassandra-streams.amazonaws.com`

Die genannte Rollenberechtigungsrichtlinie KeyspacesCDCServiceRolePolicyermöglicht es Amazon Keyspaces, die folgende Aktion für Ressourcen im CloudWatch `AWS/Cassandra` Namespace durchzuführen:

- Aktion: `cloudwatch:PutMetricData` für `*`

  Das AWSServiceRoleForAmazonKeyspacesCDC stellt die Berechtigungen bereit: Aktion: Cloudwatch: PutMetricData für alle Ressourcen, die die folgende Bedingung erfüllen:. `"cloudwatch:namespace": "AWS/Cassandra"`

Weitere Informationen zu finden Sie KeyspacesCDCServiceRolePolicy unterthe section called “KeyspacesCDCServiceRolePolicy”.

Um CDC-Streams für eine Tabelle zu aktivieren, wodurch die serviceverknüpfte Rolle automatisch erstellt wird AWSServiceRoleForAmazonKeyspacesCDC, benötigt der IAM-Prinzipal die folgenden Berechtigungen.

```
{
    "Sid": "KeyspacesCDCServiceLinkedRole",
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/cassandra-streams.amazonaws.com/
AWSServiceRoleForAmazonKeyspacesCDC",
    "Condition": {
    "StringLike": {
        "iam:AWSServiceName": "cassandra-streams.amazonaws.com"
    }
}
```

Die Berechtigungen zum Erstellen der serviceverknüpften Rolle AWSServiceRoleForAmazonKeyspacesCDC sind in der verwalteten Richtlinie enthalten. `AmazonKeyspacesFullAccess` Weitere Informationen finden Sie unter the section called “AmazonKeyspacesFullAccess”.

Eine serviceverknüpfte Rolle für Amazon Keyspaces erstellen

Sie müssen keine serviceverknüpfte Rolle für Amazon Keyspaces CDC-Streams manuell erstellen. Wenn Sie Amazon Keyspaces CDC-Streams für eine Tabelle mit der AWS-Managementkonsole, CQL, der oder der AWS API aktivieren AWS CLI, erstellt Amazon Keyspaces die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie Amazon Keyspaces CDC-Streams für eine Tabelle aktivieren, erstellt Amazon Keyspaces die serviceverknüpfte Rolle erneut für Sie.

Bearbeitung einer serviceverknüpften Rolle für Amazon Keyspaces

Amazon Keyspaces erlaubt es Ihnen nicht, die AWSServiceRoleForAmazonKeyspacesCDC serviceverknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte Amazon Keyspaces-Rollen

Amazon Keyspaces unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter AWS -Regionen und Endpunkte.

Konformitätsüberprüfung für Amazon Keyspaces (für Apache Cassandra)

Third-party Auditoren bewerten die Sicherheit und Konformität von Amazon Keyspaces (für Apache Cassandra) im Rahmen mehrerer AWS Compliance-Programme. Dazu zählen:

- ISO/IEC 27001:2013, 27017:2015, 27018:2019und 9001:2015. ISO/IEC Weitere Informationen finden Sie unter ISO- und CSA STAR-Zertifizierungen und -Dienstleistungen. AWS
- System and Organization Controls (SOC)
- Payment Card Industry (PCI)
- Das Federal Risk and Authorization Management Program (FedRAMP) ist hoch
- Health Insurance Portability and Accountability Act (HIPAA)

Unser neues AWS Anmeldeerlebnis ist nicht für regulierte Workloads konzipiert. Wenn Sie unser neues AWS Anmeldeerlebnis nutzen, es aber AWS für regulierte Workloads nutzen möchten, können Sie sich für erweiterte Funktionen für AWS Ihre Umgebung anmelden oder diese aktivieren. AWS

Um zu erfahren, ob AWS-Service ein Programm AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm und wählen Sie

das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme, .

Sie können Auditberichte von Drittanbietern mit herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter Berichte AWS Artifact .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens sowie den geltenden Gesetzen und Vorschriften ab. Weitere Informationen zu Ihrer Compliance-Verantwortung bei der Verwendung AWS-Services finden Sie in der AWS Sicherheitsdokumentation.

Resilienz und Notfallwiederherstellung in Amazon Keyspaces

Die AWS globale Infrastruktur basiert AWS-Regionen auf Availability Zones. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Amazon Keyspaces repliziert Daten automatisch dreimal in mehreren AWS Availability Zones innerhalb derselben, AWS-Region um Haltbarkeit und hohe Verfügbarkeit zu gewährleisten.

Weitere Informationen zu Availability Zones AWS-Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Zusätzlich zur AWS globalen Infrastruktur bietet Amazon Keyspaces mehrere Funktionen, um Ihre Datenstabilität und Backup-Anforderungen zu erfüllen.

Replikation in mehreren Regionen

> Amazon Keyspaces bietet Replikation in mehreren Regionen, wenn Sie Ihre Daten oder Anwendungen über größere geografische Entfernungen replizieren müssen. Sie können Ihre Amazon Keyspaces-Tabellen auf verschiedene AWS-Regionen Ihrer Wahl replizieren. Weitere Informationen finden Sie unter the section called “Multi-Region Replikation”.

Point-in-time Wiederherstellung (PITR)

PITR schützt Ihre Amazon Keyspaces-Tabellen vor versehentlichen Schreib- oder Löschvorgängen, indem es Ihnen kontinuierliche Backups Ihrer Tabellendaten bereitstellt. Weitere Informationen finden Sie unter Point-in-timeWiederherstellung für Amazon Keyspaces.

Infrastruktursicherheit in Amazon Keyspaces

Als verwalteter Service ist Amazon Keyspaces (für Apache Cassandra) durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon Keyspaces zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Cipher-Suites mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Amazon Keyspaces unterstützt zwei Methoden zur Authentifizierung von Kundenanfragen. Die erste Methode verwendet dienstspezifische Anmeldeinformationen, bei denen es sich um passwortbasierte Anmeldeinformationen handelt, die für einen bestimmten IAM-Benutzer generiert wurden. Sie können das Passwort mithilfe der IAM-Konsole, der oder der API erstellen AWS CLI und verwalten. AWS Weitere Informationen finden Sie unter Verwenden von IAM mit Amazon Keyspaces.

Die zweite Methode verwendet ein Authentifizierungs-Plugin für den DataStax Open-Source-Java-Treiber für Cassandra. Mit diesem Plugin können IAM-Benutzer, -Rollen und föderierte Identitäten mithilfe des AWS Signature Version 4-Prozesses (Sigv4) Authentifizierungsinformationen zu API-Anfragen von Amazon Keyspaces (für Apache Cassandra) hinzufügen. Weitere Informationen finden Sie unter the section called “Erstellen Sie IAM-Anmeldeinformationen für die Authentifizierung AWS ”.

Sie können diese API-Operationen von jedem Netzwerkstandort aus aufrufen, Amazon Keyspaces unterstützt jedoch ressourcenbasierte Zugriffsrichtlinien, die Einschränkungen auf der Grundlage der Quell-IP-Adresse beinhalten können. Sie können auch Amazon Keyspaces-Richtlinien verwenden, um den Zugriff von bestimmten Amazon Virtual Private Cloud (Amazon VPC) -Endpunkten oder

bestimmten VPCs aus zu kontrollieren. Dadurch wird der Netzwerkzugriff auf eine bestimmte Amazon Keyspaces-Ressource effektiv nur von der spezifischen VPC innerhalb des Netzwerks isoliert. AWS

Sie können einen VPC-Schnittstellen-Endpunkt verwenden, um zu verhindern, dass der Datenverkehr zwischen Ihrer Amazon VPC und Amazon Keyspaces das Amazon-Netzwerk verlässt. Interface VPC-Endpoints basieren auf einer AWS Technologie AWS PrivateLink, die private Kommunikation zwischen AWS Services über eine elastic network interface mit privaten IPs in Ihrer Amazon VPC ermöglicht. Weitere Informationen finden Sie unter the section called “Verwendung von -Schnittstellen-VPC-Endpunkten”.

Verwenden von Amazon Keyspaces mit Schnittstellen-VPC-Endpunkten

Schnittstellen-VPC-Endpunkte ermöglichen die private Kommunikation zwischen Ihrer Virtual Private Cloud (VPC), die in Amazon VPC läuft, und Amazon Keyspaces. Schnittstelle, über die VPC-Endpunkte betrieben werden. Dabei handelt es sich um einen AWS Dienst AWS PrivateLink, der die private Kommunikation zwischen VPCs und Diensten ermöglicht. AWS

AWS PrivateLink ermöglicht dies durch die Verwendung einer elastic network interface mit privaten IP-Adressen in Ihrer VPC, sodass der Netzwerkverkehr das Amazon-Netzwerk nicht verlässt. Schnittstellen-VPC-Endpunkte benötigen keinen Internet-Gateway, kein NAT-Gerät, keine VPN-Verbindung und keine Direct Connect -Verbindung. Weitere Informationen finden Sie unter Amazon Virtual Private Cloud und Interface VPC-Endpoints ()AWS PrivateLink.

Themen

- Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces
- Auffüllen von system.peers Tabelleneinträgen mit VPC-Endpunktinformationen der Schnittstelle
- Steuerung des Zugriffs auf VPC-Schnittstellen-Endpunkte für Amazon Keyspaces
- Verfügbarkeit
- VPC-Endpunktrichtlinien und Point-in-Time-Recovery (PITR) von Amazon Keyspaces
- Häufige Fehler und Warnungen

Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces

Sie können einen VPC-Schnittstellen-Endpunkt erstellen, sodass der Datenverkehr zwischen Amazon Keyspaces und Ihren Amazon VPC-Ressourcen über den VPC-Endpunkt der Schnittstelle fließt. Folgen Sie zunächst den Schritten zum Erstellen eines Schnittstellen-Endpunkts. Bearbeiten Sie als Nächstes die Sicherheitsgruppe, die dem Endpunkt zugeordnet ist, den Sie im vorherigen Schritt

erstellt haben, und konfigurieren Sie eine Regel für eingehenden Datenverkehr für Port 9142. Weitere Informationen finden Sie unter Regeln hinzufügen, entfernen und aktualisieren.

Eine schrittweise Anleitung zur Konfiguration einer Verbindung zu Amazon Keyspaces über einen VPC-Endpunkt finden Sie unter. the section called “Verbindung mit VPC-Endpunkten herstellen” Informationen zum Konfigurieren des kontoübergreifenden Zugriffs für Amazon Keyspaces-Ressourcen getrennt von Anwendungen AWS-Konten in verschiedenen VPC finden Sie unter. the section called “Konfigurieren Sie den kontoübergreifenden Zugriff”

Auffüllen von `system.peers` Tabelleneinträgen mit VPC-Endpunktinformationen der Schnittstelle

Apache Cassandra-Treiber verwenden die `system.peers` Tabelle, um Knoteninformationen über den Cluster abzufragen. Cassandra-Treiber verwenden die Knoteninformationen für den Lastenausgleich bei Verbindungen und für Wiederholungsvorgänge. Amazon Keyspaces füllt automatisch neun Einträge in der `system.peers` Tabelle für Clients, die sich über den öffentlichen Endpunkt verbinden.

Um Kunden, die sich über VPC-Endpunkte mit Schnittstellen verbinden, ähnliche Funktionen zu bieten, füllt Amazon Keyspaces die `system.peers` Tabelle in Ihrem Konto mit einem Eintrag für jede Availability Zone, in der ein VPC-Endpunkt verfügbar ist. Um verfügbare Schnittstellen-VPC-Endpunkte in der `system.peers` Tabelle zu suchen und zu speichern, verlangt Amazon Keyspaces, dass Sie der IAM-Entität, die für die Verbindung mit Amazon Keyspaces verwendet wird, Zugriffsberechtigungen gewähren, um Ihre VPC nach den Endpunkt- und Netzwerkschnittstelleninformationen abzufragen.

> **Important**
>
> Wenn Sie die `system.peers` Tabelle mit Ihren verfügbaren Schnittstellen-VPC-Endpunkten füllen, wird der Lastenausgleich verbessert und der Durchsatz erhöht. read/write Es wird für alle Clients empfohlen, die über VPC-Endpunkte auf Amazon Keyspaces zugreifen, und ist für Apache Spark erforderlich.

Um der IAM-Entität, die für die Verbindung mit Amazon Keyspaces verwendet wird, Berechtigungen zum Abrufen der erforderlichen VPC-Endpunktinformationen für die Schnittstelle zu gewähren, können Sie Ihre bestehende IAM-Rolle oder Benutzerrichtlinie aktualisieren oder eine neue IAM-Richtlinie erstellen, wie im folgenden Beispiel gezeigt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

> **Note**
>
> Die verwalteten Richtlinien `AmazonKeyspacesReadOnlyAccess_v2` `AmazonKeyspacesFullAccess` enthalten die erforderlichen Berechtigungen, damit Amazon Keyspaces auf die Amazon EC2 EC2-Instance zugreifen kann, um Informationen über verfügbare Schnittstellen-VPC-Endpunkte zu lesen.

Um zu überprüfen, ob die Richtlinie korrekt eingerichtet wurde, fragen Sie die `system.peers` Tabelle nach Netzwerkinformationen ab. Wenn die `system.peers` Tabelle leer ist, könnte dies darauf hinweisen, dass die Richtlinie nicht erfolgreich konfiguriert wurde oder dass Sie das Kontingent für die Anforderungsrate `DescribeNetworkInterfaces` und `DescribeVPCEndpoints` API-Aktionen überschritten haben. `DescribeVPCEndpoints`fällt in die `Describe*` Kategorie und wird als nicht mutierende Aktion angesehen. `DescribeNetworkInterfaces`fällt in die Untergruppe der ungefilterten und nicht paginierten, nicht mutierenden Aktionen, für die unterschiedliche Quoten gelten. Weitere Informationen finden Sie in der Amazon EC2 EC2-API-Referenz unter Bucket-Größen und Nachfüllraten anfordern.

Wenn Sie eine leere Tabelle sehen, versuchen Sie es einige Minuten später erneut, um Probleme mit der Quote für Anfragen auszuschließen. Informationen dazu, ob Sie die VPC-Endpoints korrekt konfiguriert haben, finden Sie unter. the section called “Verbindungsfehler am VPC-Endpunkt” Wenn Ihre Abfrage Ergebnisse aus der Tabelle zurückgibt, wurde Ihre Richtlinie korrekt konfiguriert.

Steuerung des Zugriffs auf VPC-Schnittstellen-Endpunkte für Amazon Keyspaces

Mit VPC-Endpunktrichtlinien können Sie den Zugriff auf Ressourcen auf zwei Arten steuern:

- IAM-Richtlinie — Sie können die Anfragen, Benutzer oder Gruppen steuern, die über einen bestimmten VPC-Endpunkt auf Amazon Keyspaces zugreifen dürfen. Sie können dies tun, indem Sie einen Bedingungsschlüssel in der Richtlinie verwenden, der einem IAM-Benutzer, einer IAM-Gruppe oder einer IAM-Rolle zugeordnet ist.
- VPC-Richtlinie — Sie können kontrollieren, welche VPC-Endpunkte Zugriff auf Ihre Amazon Keyspaces-Ressourcen haben, indem Sie ihnen Richtlinien zuordnen. Um den Zugriff auf einen bestimmten Schlüsselraum oder eine Tabelle einzuschränken, sodass nur Datenverkehr über einen bestimmten VPC-Endpunkt zugelassen wird, bearbeiten Sie die bestehende IAM-Richtlinie, die den Ressourcenzugriff einschränkt, und fügen Sie diesen VPC-Endpunkt hinzu.

Im Folgenden finden Sie Beispiele für Endpunktrichtlinien für den Zugriff auf Amazon Keyspaces-Ressourcen.

- Beispiel für eine IAM-Richtlinie: Beschränken Sie den gesamten Zugriff auf eine bestimmte Amazon Keyspaces-Tabelle, sofern der Datenverkehr nicht vom angegebenen VPC-Endpunkt stammt — Diese Beispielrichtlinie kann einem IAM-Benutzer, einer IAM-Rolle oder einer IAM-Gruppe zugewiesen werden. Es schränkt den Zugriff auf eine bestimmte Amazon Keyspaces-Tabelle ein, es sei denn, der eingehende Verkehr stammt von einem bestimmten VPC-Endpunkt.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "UserOrRolePolicyToDenyAccess",
         "Action": "cassandra:*",
         "Effect": "Deny",
         "Resource": [
                        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable",
                        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
           ],
         "Condition": { "StringNotEquals" : { "aws:sourceVpce": "vpce-abc123" } }
      }
   ]
}
```

> **Note**
>
> Um den Zugriff auf eine bestimmte Tabelle einzuschränken, müssen Sie auch den Zugriff auf die Systemtabellen einbeziehen. Systemtabellen sind schreibgeschützt.

- Beispiel für eine VPC-Richtlinie: Read-only Zugriff — Diese Beispielrichtlinie kann an einen VPC-Endpunkt angehängt werden. (Weitere Informationen finden Sie unter Steuern des Zugriffs auf Amazon VPC-Ressourcen). Es beschränkt Aktionen auf den schreibgeschützten Zugriff auf Amazon Keyspaces-Ressourcen über den VPC-Endpunkt, mit dem es verbunden ist.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadOnly",
      "Principal": "*",
      "Action": [
        "cassandra:Select"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

- Beispiel für eine VPC-Richtlinie: Beschränken Sie den Zugriff auf eine bestimmte Amazon Keyspaces-Tabelle — Diese Beispielrichtlinie kann an einen VPC-Endpunkt angehängt werden. Es schränkt den Zugriff auf eine bestimmte Tabelle über den VPC-Endpunkt ein, an den sie angehängt ist.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Sid": "RestrictAccessToTable",
            "Principal": "*",
            "Action": "cassandra:*",
            "Effect": "Allow",
            "Resource": [
```

```
                        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable",
                        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
            ]
        }
    ]
}
```

> **Note**
>
> Um den Zugriff auf eine bestimmte Tabelle einzuschränken, müssen Sie auch den Zugriff auf die Systemtabellen einbeziehen. Systemtabellen sind schreibgeschützt.

Verfügbarkeit

Amazon Keyspaces unterstützt die Verwendung von VPC-Endpunkten mit Schnittstellen überall dort, AWS-Regionen wo der Service verfügbar ist. Weitere Informationen finden Sie unter ???.

VPC-Endpunktrichtlinien und Point-in-Time-Recovery (PITR) von Amazon Keyspaces

Wenn Sie IAM-Richtlinien mit Bedingungsschlüsseln verwenden, um den eingehenden Datenverkehr einzuschränken, schlägt der Vorgang zur Wiederherstellung der Tabelle möglicherweise fehl. Wenn Sie beispielsweise den Quelldatenverkehr mithilfe von `aws:SourceVpce` Bedingungsschlüsseln auf bestimmte VPC-Endpunkte beschränken, schlägt die Tabellenwiederherstellung fehl. Damit Amazon Keyspaces im Namen Ihres Principals einen Wiederherstellungsvorgang durchführen kann, müssen Sie Ihrer IAM-Richtlinie einen `aws:ViaAWSService` Bedingungsschlüssel hinzufügen. Der `aws:ViaAWSService` Bedingungsschlüssel ermöglicht den Zugriff, wenn ein AWS Dienst unter Verwendung der Anmeldeinformationen des Prinzipals eine Anfrage stellt. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingungsschlüssel im IAM-Benutzerhandbuch. Die folgende Richtlinie ist ein Beispiel dafür.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"CassandraAccessForVPCE",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
```

```
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"false"
            },
            "StringEquals":{
               "aws:SourceVpce":[
                  "vpce-12345678901234567"
               ]
            }
         }
      },
      {
         "Sid":"CassandraAccessForAwsService",
         "Effect":"Allow",
         "Action":"cassandra:*",
         "Resource":"*",
         "Condition":{
            "Bool":{
               "aws:ViaAWSService":"true"
            }
         }
      }
   ]
}
```

Häufige Fehler und Warnungen

Wenn Sie Amazon Virtual Private Cloud verwenden und eine Verbindung zu Amazon Keyspaces herstellen, wird möglicherweise die folgende Warnung angezeigt.

```
Control node cassandra.us-east-1.amazonaws.com/1.111.111.111:9142 has an entry
 for itself in system.peers: this entry will be ignored. This is likely due to a
 misconfiguration;
please verify your rpc_address configuration in cassandra.yaml on all nodes in your
 cluster.
```

Diese Warnung tritt auf, weil die `system.peers` Tabelle Einträge für alle Amazon VPC-Endpoints enthält, zu deren Anzeige Amazon Keyspaces berechtigt ist, einschließlich des Amazon VPC-Endpunkts, über den Sie verbunden sind. Sie können diese Warnung getrost ignorieren.

Informationen zu anderen Fehlern finden Sie unterthe section called “Verbindungsfehler am VPC-Endpunkt”.

Verwendung von Amazon Keyspaces CDC-Streams mit Schnittstellen-VPC-Endpunkten

Schnittstellen-VPC-Endpunkte ermöglichen die private Kommunikation zwischen Ihrer Virtual Private Cloud (VPC), die in Amazon VPC ausgeführt wird, und Amazon Keyspaces. Schnittstelle, über die VPC-Endpunkte betrieben werden. Dabei handelt es sich um einen AWS Dienst AWS PrivateLink, der die private Kommunikation zwischen VPCs und Diensten ermöglicht. AWS

AWS PrivateLink ermöglicht dies durch die Verwendung einer elastic network interface mit privaten IP-Adressen in Ihrer VPC, sodass der Netzwerkverkehr das Amazon-Netzwerk nicht verlässt. Schnittstellen-VPC-Endpunkte benötigen keinen Internet-Gateway, kein NAT-Gerät, keine VPN-Verbindung und keine Direct Connect -Verbindung. Weitere Informationen finden Sie unter Amazon Virtual Private Cloud und Interface VPC-Endpoints ()AWS PrivateLink.

Themen

- Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces CDC-Streams
- Amazon Keyspaces: CDC-Streaming-Schnittstelle, VPC-Endpunkte
- VPC-Endpunkt für Amazon Keyspaces CDC-Streams-Schnittstelle erstellen
- Aktualisieren Sie einen VPC-Endpunkt der Amazon Keyspaces CDC-Streams-Schnittstelle
- Streams mithilfe eines VPC-Endpunkts der Amazon Keyspaces CDC-Streams-Schnittstelle auflisten
- Erstellen Sie eine Richtlinie für einen VPC-Endpunkt der Amazon Keyspaces-CDC-Streams-Schnittstelle

Verwenden von Schnittstellen-VPC-Endpunkten für Amazon Keyspaces CDC-Streams

Sie können einen VPC-Schnittstellen-Endpunkt verwenden, sodass der Datenverkehr zwischen Amazon Keyspaces CDC-Streams und Ihren Amazon VPC-Ressourcen über den VPC-Endpunkt der Schnittstelle fließt. Sie können VPC-Endpunktrichtlinien verwenden, um den Zugriff auf Ihre CDC-Streams einzuschränken.

Weitere Informationen zu Amazon Keyspaces CDC-Streams finden Sie unter. the section called “Arbeiten mit CDC-Streams”

Amazon Keyspaces: CDC-Streaming-Schnittstelle, VPC-Endpunkte

Wenn Sie einen Schnittstellenendpunkt erstellen, generiert Amazon Keyspaces CDC Streams zwei Arten von endpunktspezifischen DNS-Namen für den Stream: Regional und Zonal.

Regional

Der regionale DNS-Name enthält die folgenden Informationen:

- eine eindeutige Amazon VPC-Endpunkt-ID
- eine Service-ID
- der AWS-Region
- das `vpce.amazonaws.com` Suffix

Für einen Amazon VPC-Endpunkt mit der ID `vpce-1a2b3c4d` könnte der generierte DNS-Name dem folgenden Beispiel ähneln:`vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com`.

Zonal

Der zonale DNS-Name umfasst zusätzlich zu den Informationen im regionalen DNS-Namen die Availability Zone. Der generierte DNS-Name für den Amazon VPC-Endpunkt mit der ID `vpce-1a2b3c4d` würde wie im folgenden Beispiel aussehen. Beachten Sie, dass der AWS-Region jetzt die Availability Zone umfasst: `vpce-1a2b3c4d-5e6f-us-east-1a.cassandra-streams.us-east-1.vpce.amazonaws.com`

Sie können diese Option verwenden, wenn Ihre Architektur Availability Zones isoliert. Sie könnten sie beispielsweise zur Fehlereingrenzung oder zur Senkung der regionalen Datenübertragungskosten verwenden.

> **Note**
>
> Um eine optimale Zuverlässigkeit zu erreichen, empfehlen wir, Ihren Service in mindestens drei Availability Zones bereitzustellen.

VPC-Endpunkt für Amazon Keyspaces CDC-Streams-Schnittstelle erstellen

Sie können das AWS CLI oder das AWS SDK verwenden, um auf Amazon Keyspaces CDC Streams-API-Operationen über die Amazon Keyspaces CDC Streams-Schnittstellenendpunkte zuzugreifen.

Eine vollständige Liste aller verfügbaren API-Operationen finden Sie unter Amazon Keyspaces Streams API-Referenz.

Weitere Informationen zum Erstellen von VPC-Endpunkten finden Sie unter Erstellen eines Schnittstellenendpunkts im Amazon VPC-Benutzerhandbuch.

Um einen VPC-Endpunkt zu erstellen, können Sie die Syntax im folgenden Beispiel verwenden.

```
aws ec2 create-vpc-endpoint \
  --region us-east-1 \
  --service-name api.aws.us-east-1.cassandra-streams \
  --vpc-id client-vpc-id \
  --subnet-ids client-subnet-id \
  --vpc-endpoint-type Interface \
  --security-group-ids client-sg-id
```

Aktualisieren Sie einen VPC-Endpunkt der Amazon Keyspaces CDC-Streams-Schnittstelle

Um einen VPC-Endpunkt zu aktualisieren, können Sie die Syntax im folgenden Beispiel verwenden.

```
aws ec2 modify-vpc-endpoint \
  --region us-east-1 \
  --vpc-endpoint-id client-vpc-id \
  --policy-document policy-document \ #example optional parameter
  --add-security-group-ids security-group-ids \ #example optional parameter
```

Streams mithilfe eines VPC-Endpunkts der Amazon Keyspaces CDC-Streams-Schnittstelle auflisten

Um die Streams aufzulisten, die einen VPC-Endpunkt verwenden, können Sie die Syntax im folgenden Beispiel verwenden. Stellen Sie sicher, dass Sie die Region und den DNS-Namen der VPC-Endpunkt-ID durch Ihre eigenen Informationen ersetzen.

```
aws keyspacesstreams \
  --endpoint https://vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com
 \
  --region us-east-1 \
  list-streams
```

Erstellen Sie eine Richtlinie für einen VPC-Endpunkt der Amazon Keyspaces-CDC-Streams-Schnittstelle

Sie können Ihrem Amazon VPC-Endpunkt eine Endpunktrichtlinie hinzufügen, die den Zugriff auf Amazon Keyspaces CDC-Streams steuert. Die Richtlinie gibt die folgenden Informationen an:

- Der AWS Identity and Access Management (IAM-) Principal, der Aktionen ausführen kann
- Aktionen, die ausgeführt werden können
- Ressourcen, für die Aktionen ausgeführt werden können

Um den Zugriff auf bestimmte Amazon Keyspaces CDC-Streams so zu beschränken, dass nur bestimmte AWS Dienste in Ihrem Amazon VPC-Zugriff zugelassen werden, können Sie das folgende Beispiel verwenden.

Die folgende Stream-Richtlinie gewährt Zugriff auf jeden IAM-Principal für die Aktionen `cassandra:GetStream` und `cassandra:GetRecords` für den angegebenen Stream, der der zum Konto gehörenden Ressource `2025-02-20T11:22:33.444` `/keyspace/mykeyspace/table/mytable/` zugeordnet ist. `123456788901` Um diese Endpunktrichtlinie zu verwenden, stellen Sie sicher, dass Sie die Region, die Konto-ID und die Ressource durch das Stream-Label ersetzen.

```
{
"Version": "2012-10-17",
  "Id": "Policy1216114807515",
  "Statement": [
    { "Sid": "Access-to-specific-stream-only",
      "Principal": "*",
      "Action": [
        "cassandra:GetStream",
        "cassandra:GetRecords"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable/stream/2025-02-20T11:22:33.444"]
    }
  ]
}
```

> **Note**
>
> Amazon Keyspaces unterstützt keine Gateway-Endpunkte für CDC-Streams.

Konfiguration und Schwachstellenanalyse für Amazon Keyspaces

AWS kümmert sich um grundlegende Sicherheitsaufgaben wie das Patchen von Gastbetriebssystemen (OS) und Datenbanken, die Firewall-Konfiguration und die Notfallwiederherstellung. Diese Verfahren wurden von qualifizierten Dritten überprüft und zertifiziert. Weitere Informationen finden Sie in den folgenden Ressourcen:

- Modell der gemeinsamen Verantwortung Modell der der
- Amazon Web Services: Übersicht über Sicherheitsverfahren (Whitepaper)

Bewährte Sicherheitsmethoden für Amazon Keyspaces

Amazon Keyspaces (für Apache Cassandra) bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden sind allgemeine Richtlinien und keine vollständige Sicherheitslösung. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

Themen

- Bewährte Methoden zur präventiven Sicherheit für Amazon Keyspaces
- Bewährte Methoden zur Detektivsicherheit für Amazon Keyspaces

Bewährte Methoden zur präventiven Sicherheit für Amazon Keyspaces

Die folgenden bewährten Sicherheitsmethoden gelten als präventiv, da sie Ihnen helfen können, Sicherheitsvorfälle in Amazon Keyspaces zu antizipieren und zu verhindern.

Verwenden Sie Verschlüsselung im Ruhezustand

Amazon Keyspaces verschlüsselt im Ruhezustand alle Benutzerdaten, die in Tabellen gespeichert sind, mithilfe von Verschlüsselungsschlüsseln, die in AWS Key Management Service ()AWS

KMS gespeichert sind. Dies bietet eine zusätzliche Datenschutzebene, indem Ihre Daten vor unbefugtem Zugriff auf den zugrunde liegenden Speicher geschützt werden.

Standardmäßig verwendet Amazon Keyspaces eine, AWS-eigener Schlüssel um all Ihre Tabellen zu verschlüsseln. Wenn dieser Schlüssel nicht existiert, wird er für Sie erstellt. Die Standardschlüssel des Dienstes können nicht deaktiviert werden.

Alternativ können Sie einen vom Kunden verwalteten Schlüssel für die Verschlüsselung im Ruhezustand verwenden. Weitere Informationen finden Sie unter Amazon Keyspaces Encryption at Rest.

Verwenden Sie IAM-Rollen, um den Zugriff auf Amazon Keyspaces zu authentifizieren

Damit Benutzer, Anwendungen und andere AWS Dienste auf Amazon Keyspaces zugreifen können, müssen sie gültige AWS Anmeldeinformationen in ihren AWS API-Anfragen angeben. Sie sollten AWS Anmeldeinformationen nicht direkt in der Anwendung oder EC2-Instance speichern. Dies sind langfristige Anmeldeinformationen, die nicht automatisch rotiert werden und daher erhebliche geschäftliche Auswirkungen haben können, wenn sie kompromittiert werden. Eine IAM-Rolle ermöglicht es Ihnen, temporäre Zugriffsschlüssel zu erhalten, mit denen Sie auf die AWS -Services und Ressourcen zugreifen können.

Weitere Informationen finden Sie unter IAM-Rollen.

Verwenden Sie IAM-Richtlinien für die Amazon Keyspaces-Basisautorisierung

Bei der Erteilung von Berechtigungen entscheiden Sie, wer sie erhält, für welche Amazon Keyspaces-APIs sie Berechtigungen erhalten und welche spezifischen Aktionen Sie für diese Ressourcen zulassen möchten. Die Implementierung der geringsten Rechte ist entscheidend für die Reduzierung von Sicherheitsrisiken und der Auswirkungen, die sich aus Fehlern oder böswilligen Absichten ergeben können.

Hängen Sie Berechtigungsrichtlinien an IAM-Identitäten (d. h. Benutzer, Gruppen und Rollen) an und gewähren Sie so Berechtigungen zur Ausführung von Vorgängen auf Amazon Keyspaces-Ressourcen.

Sie können dies wie folgt tun:

- AWS verwaltete (vordefinierte) Richtlinien
- Kundenverwaltete Richtlinien

Verwenden von IAM-Richtlinienbedingungen für eine differenzierte Zugriffssteuerung

Wenn Sie Berechtigungen in Amazon Keyspaces gewähren, können Sie Bedingungen angeben, die bestimmen, wie eine Berechtigungsrichtlinie wirksam wird. Die Implementierung der geringsten Rechte ist entscheidend für die Reduzierung von Sicherheitsrisiken und der Auswirkungen, die sich aus Fehlern oder böswilligen Absichten ergeben können.

Sie können Bedingungen angeben, wenn Sie Berechtigungen mithilfe einer IAM-Richtlinie erteilen. Sie können z. B. Folgendes tun:

- Erteilen Sie Berechtigungen, um Benutzern nur Lesezugriff auf bestimmte Schlüsselräume oder Tabellen zu gewähren.
- Gewähren Sie einem Benutzer je nach Identität des Benutzers Schreibzugriff auf eine bestimmte Tabelle.

Weitere Informationen finden Sie unter Identity-Based Richtlinienbeispiele.

Clientseitige Verschlüsselung in Betracht ziehen

Wenn Sie sensible oder vertrauliche Daten in Amazon Keyspaces speichern, möchten Sie diese Daten möglicherweise so nah wie möglich an ihrem Ursprung verschlüsseln, damit Ihre Daten während ihres gesamten Lebenszyklus geschützt sind. Die Verschlüsselung Ihrer sensiblen Daten während der Übertragung und im Ruhezustand stellt sicher, dass Ihre Klartextdaten nicht für Dritte verfügbar sind.

Bewährte Methoden zur Detektivsicherheit für Amazon Keyspaces

Die folgenden bewährten Sicherheitsmethoden gelten als detektiv, da sie Ihnen helfen können, potenzielle Sicherheitslücken und Sicherheitsvorfälle zu erkennen.

Wird verwendet AWS CloudTrail , um die Verwendung von AWS Key Management Service (AWS KMS) AWS KMS Schlüsseln zu überwachen

Wenn Sie einen vom Kunden verwalteten AWS KMS Schlüssel für die Verschlüsselung im Ruhezustand verwenden, wird die Verwendung dieses Schlüssels angemeldet AWS CloudTrail. CloudTrail bietet Einblick in die Benutzeraktivitäten, indem es die auf Ihrem Konto ausgeführten Aktionen aufzeichnet. CloudTrail zeichnet wichtige Informationen zu jeder Aktion auf, einschließlich der Person, die die Anfrage gestellt hat, die verwendeten Dienste, die durchgeführten Aktionen, die Parameter für die Aktionen und die vom AWS Dienst zurückgegebenen Antwortelemente. Diese Informationen helfen Ihnen dabei, Änderungen an

Ihren AWS Ressourcen nachzuverfolgen und betriebliche Probleme zu beheben. CloudTrail macht es einfacher, die Einhaltung interner Richtlinien und regulatorischer Standards sicherzustellen.

Sie können es verwenden CloudTrail , um die Verwendung von Schlüsseln zu überprüfen. CloudTrail erstellt Protokolldateien, die einen Verlauf der AWS API-Aufrufe und verwandter Ereignisse für Ihr Konto enthalten. Diese Protokolldateien enthalten alle AWS KMS API-Anfragen, die über die Konsole, AWS SDKs und Befehlszeilentools gestellt wurden, sowie alle API-Anfragen, die über integrierte AWS Dienste gestellt wurden. Sie können diese Protokolldateien verwenden, um Informationen darüber zu erhalten, wann der AWS KMS Schlüssel verwendet wurde, welcher Vorgang angefordert wurde, die Identität des Anfragenden, die IP-Adresse, von der die Anfrage kam, usw. Weitere Informationen finden Sie unter Protokollierung von AWS Key Management Service -API-Aufrufen mit AWS CloudTrail im AWS CloudTrail -Benutzerhandbuch.

Wird CloudTrail zur Überwachung von DDL-Vorgängen (Data Definition Language) von Amazon Keyspaces verwendet

CloudTrail bietet Einblick in die Benutzeraktivitäten, indem die auf Ihrem Konto durchgeführten Aktionen aufgezeichnet werden. CloudTrail zeichnet wichtige Informationen zu jeder Aktion auf, einschließlich der Person, die die Anfrage gestellt hat, die verwendeten Dienste, die durchgeführten Aktionen, die Parameter für die Aktionen und die vom AWS Dienst zurückgegebenen Antwortelemente. Diese Informationen helfen Ihnen dabei, Änderungen an Ihren AWS Ressourcen nachzuverfolgen und betriebliche Probleme zu beheben. CloudTrail macht es einfacher, die Einhaltung interner Richtlinien und regulatorischer Standards sicherzustellen.

Alle Amazon Keyspaces DDL-Operationen werden automatisch angemeldet CloudTrail . Mit DDL-Operationen können Sie Amazon Keyspaces und -Tabellen erstellen und verwalten.

Wenn Aktivitäten in Amazon Keyspaces auftreten, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Weitere Informationen finden Sie unter Protokollieren von Amazon Keyspaces-Vorgängen mithilfe AWS CloudTrail von. Sie können aktuelle Ereignisse in Ihrem AWS-Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie im AWS CloudTrail Benutzerhandbuch unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, einschließlich Veranstaltungen für Amazon Keyspaces, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon Simple Storage Service (Amazon S3) -

Bucket. Wenn Sie einen Trail auf der Konsole erstellen, gilt der Trail standardmäßig für alle AWS-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS -Partition und stellt die Protokolldateien für den von Ihnen angegebenen S3 Bucket bereit. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren.

Kennzeichnen Sie Ihre Amazon Keyspaces-Ressourcen zur Identifizierung und Automatisierung

Sie können Ihren AWS Ressourcen Metadaten in Form von Tags zuweisen. Jedes Tag ist eine einfache Bezeichnung, die aus einem vom Kunden definierten Schlüssel und einem optionalen Wert besteht, der die Verwaltung, Suche und Filterung von Ressourcen erleichtern kann.

Tagging ermöglicht die Implementierung gruppierter Steuerelemente. Obwohl es keine inhärenten Typen von Tags gibt, können Sie Ressourcen nach Zweck, Besitzer, Umgebung oder anderen Kriterien kategorisieren. Im Folgenden sind einige Beispiele aufgeführt:

- Zugriff — Wird verwendet, um den Zugriff auf Amazon Keyspaces-Ressourcen anhand von Tags zu steuern. Weitere Informationen finden Sie unter the section called “Autorisierung basierend auf Amazon Keyspaces-Tags”.
- Sicherheit — Wird verwendet, um Anforderungen wie Datenschutzeinstellungen festzulegen.
- Vertraulichkeit — Eine Kennung für die spezifische Datenvertraulichkeitsstufe, die eine Ressource unterstützt.
- Umgebung – Wird verwendet, um zwischen Entwicklungs-, Test- und Produktionsinfrastruktur zu unterscheiden.

Weitere Informationen finden Sie unter AWS Tagging-Strategien und Hinzufügen von Tags und Labels zu Ressourcen.

CQL-Sprachreferenz für Amazon Keyspaces (für Apache Cassandra)

Nachdem Sie eine Verbindung zu einem Amazon Keyspaces-Endpunkt hergestellt haben, verwenden Sie Cassandra Query Language (CQL), um mit Ihrer Datenbank zu arbeiten. CQL ähnelt in vielerlei Hinsicht der Structured Query Language (SQL).

- CQL-Elemente — Dieser Abschnitt behandelt die grundlegenden Elemente von CQL, die in Amazon Keyspaces unterstützt werden, darunter Bezeichner, Konstanten, Begriffe und Datentypen. Er erklärt Konzepte wie Zeichenfolgentypen, numerische Typen, Sammlungstypen und mehr.
- Data Definition Language (DDL) — DDL-Anweisungen werden verwendet, um Datenstrukturen wie Schlüsselräume und Tabellen in Amazon Keyspaces zu verwalten. Dieser Abschnitt behandelt Anweisungen zum Erstellen, Ändern und Löschen von Schlüsselräumen und Tabellen sowie zum Wiederherstellen von Tabellen aus einem Point-in-Time-Backup.
- Data Manipulation Language (DML) — DML-Anweisungen werden verwendet, um Daten in Tabellen zu verwalten. Dieser Abschnitt behandelt Anweisungen zum Auswählen, Einfügen, Aktualisieren und Löschen von Daten. Außerdem werden erweiterte Abfragefunktionen wie die Verwendung des `IN` Operators, das Sortieren von Ergebnissen und die Seitennummerierung erläutert.
- Built-in Funktionen — Amazon Keyspaces unterstützt eine Vielzahl integrierter Skalarfunktionen, die Sie in CQL-Anweisungen verwenden können. Dieser Abschnitt bietet einen Überblick über diese Funktionen, einschließlich Anwendungsbeispielen.

In diesem Thema finden Sie ausführliche Syntax, Beispiele und bewährte Methoden für die effektive Verwendung von CQL in Amazon Keyspaces.

Themen

- Elemente der Cassandra Query Language (CQL) in Amazon Keyspaces
- DDL-Anweisungen (Datendefinitionssprache) in Amazon Keyspaces
- DML-Anweisungen (Datenmanipulationssprache) in Amazon Keyspaces
- Built-in Funktionen in Amazon Keyspaces

Elemente der Cassandra Query Language (CQL) in Amazon Keyspaces

Erfahren Sie mehr über die Cassandra Query Language (CQL) -Elemente, die von Amazon Keyspaces unterstützt werden, darunter Bezeichner, Konstanten, Begriffe und Datentypen.

Themen

- IDs
- Konstanten
- Bedingungen
- Datentypen
- JSON-Kodierung von Amazon Keyspaces-Datentypen

IDs

Identifikatoren (oder Namen) werden verwendet, um Tabellen, Spalten und andere Objekte zu identifizieren. Ein Bezeichner kann in Anführungszeichen gesetzt werden oder nicht. Folgendes gilt.

```
identifier          ::=  unquoted_identifier | quoted_identifier
unquoted_identifier ::=  re('[a-zA-Z][a-zA-Z0-9_]*')
quoted_identifier   ::=  '"' (any character where " can appear if doubled)+ '"'
```

Konstanten

Die folgenden Konstanten sind definiert.

```
constant ::=  string | integer | float | boolean | uuid | blob | NULL
string   ::=  '\'' (any character where ' can appear if doubled)+ '\''
              '$$' (any character other than '$$') '$$'
integer  ::=  re('-?[0-9]+')
float    ::=  re('-?[0-9]+(\.[0-9]*)?([eE][+-]?[0-9+])?') | NAN | INFINITY
boolean  ::=  TRUE | FALSE
uuid     ::=  hex{8}-hex{4}-hex{4}-hex{4}-hex{12}
hex      ::=  re("[0-9a-fA-F]")
blob     ::=  '0' ('x' | 'X') hex+
```

Bedingungen

Ein Begriff bezeichnet die Art von Werten, die unterstützt werden. Begriffe werden wie folgt definiert.

```
term                ::=  constant | literal | function_call | arithmetic_operation |
 type_hint | bind_marker
literal             ::=  collection_literal | tuple_literal
function_call       ::=  identifier '(' [ term (',' term)* ] ')'
arithmetic_operation ::=  '-' term | term ('+' | '-' | '*' | '/' | '%') term
```

Datentypen

Amazon Keyspaces unterstützt die folgenden Datentypen:

Zeichenfolgen-Typen

Datentyp	Description
`ascii`	Stellt eine ASCII-Zeichenfolge dar.
`text`	Stellt eine UTF-8 codierte Zeichenfolge dar.
`varchar`	Stellt eine UTF-8 kodierte Zeichenfolge dar (`varchar`ist ein Alias für`text`).

Numerische Typen

Datentyp	Description
`bigint`	Stellt einen 64-Bit-Wert vom Typ Long mit Vorzeichen dar.
`counter`	Stellt einen 64-Bit-Integer-Zähler mit Vorzeichen dar. Weitere Informationen finden Sie unter the section called “Zähler”.
`decimal`	Stellt eine Dezimalzahl mit variabler Genauigkeit dar.

Datentyp	Description
`double`	Stellt einen 64-Bit-IEEE-754-Gleitkommawert dar.
`float`	Stellt einen 32-Bit-Gleitkommawert nach IEEE 754 dar.
`int`	Stellt einen 32-Bit-Int-Wert mit Vorzeichen dar.
`varint`	Stellt einen Ganzzahlwert im Bereich +/-10^38 dar.

Zähler

Eine `counter` Spalte enthält eine 64-Bit-Ganzzahl mit Vorzeichen. Der Zählerwert wird mithilfe der the section called “UPDATE” Anweisung erhöht oder verringert und kann nicht direkt festgelegt werden. Das macht `counter` Spalten nützlich, um Zählungen nachzuverfolgen. Mithilfe von Zählern können Sie beispielsweise die Anzahl der Einträge in einer Protokolldatei oder die Anzahl der Aufrufe eines Beitrags in einem sozialen Netzwerk verfolgen. Für `counter` Spalten gelten die folgenden Einschränkungen:

- Eine Spalte vom Typ `counter` kann nicht Teil `primary key` einer Tabelle sein.
- In einer Tabelle, die eine oder mehrere Spalten des Typs enthält`counter`, müssen alle Spalten in dieser Tabelle vom Typ sein`counter`.

In Fällen, in denen eine Zähleraktualisierung fehlschlägt (z. B. aufgrund von Timeouts oder Verbindungsverlust mit Amazon Keyspaces), weiß der Client nicht, ob der Zählerwert aktualisiert wurde. Wenn die Aktualisierung erneut versucht wird, wird die Aktualisierung des Zählerwerts möglicherweise ein zweites Mal angewendet.

Blob-Typ

Datentyp	Description
`blob`	Stellt beliebige Byte dar.

Typ BOOLEAN

Datentyp	Description
`boolean`	Steht für `true` oder`false`.

Time-related Typen

Datentyp	Description
`date`	Eine Zeichenfolge im Format<yyyy>-<mm>-<dd>.
`timestamp`	64-Bit-Ganzzahl mit Vorzeichen, die das Datum und die Uhrzeit seit der Epoche (1. Januar 1970 um 00:00:00 GMT) in Millisekunden darstellt.
`timeuuid`	Stellt eine UUID der Version 1 dar.

Sammlungstypen

Datentyp	Description
`list`	Stellt eine geordnete Sammlung literaler Elemente dar.
`map`	Stellt eine ungeordnete Sammlung von Schlüssel-Wert-Paaren dar.
`set`	Stellt eine ungeordnete Sammlung von einem oder mehreren literalen Elementen dar.

Sie deklarieren eine Auflistungsspalte, indem Sie den Sammlungstyp verwenden, gefolgt von einem anderen Datentyp (z. B. `TEXT` oder`INT`) in spitzen Klammern. Sie können eine Spalte mit einem `SET`

of erstellen`TEXT`, oder Sie können Paare aus `MAP` of `TEXT` und `INT` Schlüssel-Wert-Paaren erstellen, wie im folgenden Beispiel gezeigt.

```
SET <TEXT>
MAP <TEXT, INT>
```

Eine nicht eingefrorene Sammlung ermöglicht es Ihnen, Aktualisierungen an jedem einzelnen Sammlungselement vorzunehmen. Client-side Zeitstempel und Einstellungen für Time to Live (TTL) werden für einzelne Elemente gespeichert.

Wenn Sie das `FROZEN` Schlüsselwort für einen Sammlungstyp verwenden, werden die Werte der Sammlung zu einem einzigen unveränderlichen Wert serialisiert, und Amazon Keyspaces behandelt sie wie einen. `BLOB` Dies ist eine eingefrorene Sammlung. Eine `INSERT` `UPDATE` Oder-Anweisung überschreibt die gesamte eingefrorene Sammlung. Sie können einzelne Elemente in einer eingefrorenen Sammlung nicht aktualisieren.

Client-side Zeitstempel und TTL-Einstellungen (Time to Live) gelten für die gesamte eingefrorene Sammlung, nicht für einzelne Elemente. `Frozen`Sammlungsspalten können Teil `PRIMARY KEY` einer Tabelle sein.

Sie können eingefrorene Sammlungen verschachteln. Sie können beispielsweise `MAP` innerhalb von a definieren, `SET` wenn der `MAP` das `FROZEN` Schlüsselwort verwendet, wie im folgenden Beispiel gezeigt.

```
SET <FROZEN> <MAP <TEXT, INT>>>
```

Amazon Keyspaces unterstützt standardmäßig die Verschachtelung von bis zu 8 Ebenen eingefrorener Sammlungen. Weitere Informationen finden Sie unter the section called “Amazon Keyspaces-Servicekontingente”. Weitere Informationen zu den funktionalen Unterschieden mit Apache Cassandra finden Sie unter. the section called “`FROZEN`Sammlungen” Weitere Hinweise zur CQL-Syntax finden Sie unter the section called “CREATE TABLE” und. the section called “ALTER TABLE”

Typ „Tupel“

Der `tuple` Datentyp steht für eine begrenzte Gruppe literaler Elemente. Sie können ein Tupel als Alternative zu einem verwenden. `user defined type` Sie müssen das `FROZEN` Schlüsselwort nicht für Tupel verwenden. Das liegt daran, dass ein Tupel immer eingefroren ist und Sie Elemente nicht einzeln aktualisieren können.

Andere Typen

Datentyp	Description
`inet`	Eine Zeichenfolge, die eine IP-Adresse darstellt , entweder im IPv4- oder IPv6-Format.

Statisch

In einer Amazon Keyspaces-Tabelle mit Clusterspalten können Sie das `STATIC` Schlüsselwort verwenden, um eine statische Spalte beliebigen Typs zu erstellen.

Die folgende Aussage ist ein Beispiel dafür.

```
my_column INT STATIC
```

Weitere Hinweise zum Arbeiten mit statischen Spalten finden Sie unterthe section called “Schätzen Sie den Kapazitätsverbrauch statischer Spalten”.

User-defined Typen (UDTs)

Amazon Keyspaces unterstützt benutzerdefinierte Typen (UDTs). Sie können jeden gültigen Amazon Keyspaces-Datentyp verwenden, um eine UDT zu erstellen, einschließlich Sammlungen und anderer vorhandener UDTs. Sie erstellen UDTs in einem Schlüsselraum und können sie verwenden, um Spalten in einer beliebigen Tabelle im Schlüsselraum zu definieren.

Weitere Hinweise zur CQL-Syntax finden Sie unter. the section called “Typen” Weitere Informationen zur Arbeit mit UDTs finden Sie unter. the section called “User-defined Typen (UDTs)”

Informationen darüber, wie viele UDTs pro Schlüsselraum unterstützt werden, welche Verschachtelungsebenen unterstützt werden und welche anderen Standardwerte und Kontingente für UDTs gelten, finden Sie unter. the section called “Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces”

JSON-Kodierung von Amazon Keyspaces-Datentypen

Amazon Keyspaces bietet dieselben JSON-Datentypzuordnungen wie Apache Cassandra. In der folgenden Tabelle werden die Datentypen beschrieben, die Amazon Keyspaces in `INSERT JSON`

Anweisungen akzeptiert, und die Datentypen, die Amazon Keyspaces bei der Rückgabe von Daten mit der `SELECT JSON` Anweisung verwendet.

Für Einzelfeld-Datentypen wie`float`, `int` `UUID``date`, und können Sie Daten auch als einfügen. `string` Für zusammengesetzte Datentypen und Sammlungen wie `tuple` `map``list`, und können Sie Daten auch als JSON oder codiert `JSON string` einfügen.

JSON-Datentyp	In **`INSERT JSON`** Anweisungen akzeptierte Datentypen	In **`SELECT JSON`** Anweisungen zurückgegebene Datentypen	Hinweise
`ascii`	`string`	`string`	Verwendet JSON-Zeichen-Escape-Zeichen `\u`.
`bigint`	`integer, string`	`integer`	Die Zeichenfolge muss eine gültige 64-Bit-Ganzzahl sein.
`blob`	`string`	`string`	Die Zeichenfolge sollte mit beginnen, `0x` gefolgt von einer geraden Anzahl von Hex-Ziffern.
`boolean`	`boolean, string`	`boolean`	Die Zeichenfolge muss entweder `true` oder sein`false`.
`date`	`string`	`string`	Datum im Format`YYYY-MM-DD` , Zeitzone UTC.
`decimal`	`integer, float, string`	`float`	Kann im clientseitigen Decoder die IEEE-754 Gleitkommagenauigk

JSON-Datentyp	In **`INSERT JSON`** Anweisungen akzeptierte Datentypen	In **`SELECT JSON`** Anweisungen zurückgegebene Datentypen	Hinweise
			eit von 32 Bit oder 64 Bit überschreiten.
`double`	`integer, float, string`	`float`	Die Zeichenfolge muss eine gültige Ganzzahl oder eine Fließkommazahl sein.
`float`	`integer, float, string`	`float`	Die Zeichenfolge muss eine gültige Ganzzahl oder eine Fließkommazahl sein.
`inet`	`string`	`string`	IPv4- oder IPv6-Adresse.
`int`	`integer, string`	`integer`	Die Zeichenfolge muss eine gültige 32-Bit-Ganzzahl sein.
`list`	`list, string`	`list`	Verwendet die native JSON-Listendarstellung.
`map`	`map, string`	`map`	Verwendet die native JSON-Kartendarstellung.
`smallint`	`integer, string`	`integer`	Die Zeichenfolge muss eine gültige 16-Bit-Ganzzahl sein.

JSON-Datentyp	In **`INSERT JSON`** Anweisungen akzeptierte Datentypen	In **`SELECT JSON`** Anweisungen zurückgegebene Datentypen	Hinweise
`set`	`list, string`	`list`	Verwendet die native JSON-Listendarstellung.
`text`	`string`	`string`	Verwendet JSON-Zeichen-Escape\u.
`time`	`string`	`string`	Tageszeit im Format`HH-MM-SS[.fffffffff]` .
`timestamp`	`integer, string`	`string`	Ein Zeitstempel. Zeichenkettenkonstanten ermöglichen es Ihnen, Zeitstempel als Datumsangaben zu speichern. Datumsstempel mit Format `YYYY-MM-DD HH:MM:SS.SSS` werden zurückgegeben.
`timeuuid`	`string`	`string`	Geben Sie 1 UUID ein. Informationen constants zum UUID-Format finden Sie unter.
`tinyint`	`integer, string`	`integer`	Die Zeichenfolge muss eine gültige 8-Bit-Ganzzahl sein.

JSON-Datentyp	In **`INSERT JSON`** Anweisungen akzeptierte Datentypen	In **`SELECT JSON`** Anweisungen zurückgegebene Datentypen	Hinweise
`tuple`	`list, string`	`list`	Verwendet die native JSON-Listendarstellung.
`UDT`	`map, string`	`map`	Verwendet die native JSON-Kartendarstellung mit Feldnamen als Schlüssel.
`uuid`	`string`	`string`	Informationen constants zum UUID-Format finden Sie unter.
`varchar`	`string`	`string`	Verwendet JSON-Zeichen-Escape. \u
`varint`	`integer, string`	`integer`	Variable Länge; kann 32-Bit- oder 64-Bit-Ganzzahlen im clientseitigen Decoder überlaufen.

DDL-Anweisungen (Datendefinitionssprache) in Amazon Keyspaces

Data Definition Language (DDL) ist der Satz von Cassandra Query Language (CQL) -Anweisungen, die Sie verwenden, um Datenstrukturen in Amazon Keyspaces (für Apache Cassandra) zu verwalten, z. B. Schlüsselräume und Tabellen. Sie verwenden DDL, um diese Datenstrukturen zu erstellen, sie zu ändern, nachdem sie erstellt wurden, und sie zu entfernen, wenn sie nicht mehr verwendet werden. Amazon Keyspaces führt DDL-Operationen asynchron aus. Weitere Informationen darüber,

wie Sie überprüfen können, ob ein asynchroner Vorgang abgeschlossen wurde, finden Sie unter. the section called “Asynchrone Erstellung und Löschung von Schlüsselräumen und Tabellen”

Die folgenden DDL-Anweisungen werden unterstützt:

- SCHLÜSSELRAUM ERSTELLEN
- SCHLÜSSELRAUM ÄNDERN
- KEYSPACE LÖSCHEN
- BENUTZE
- TABELLE ERSTELLEN
- ALTER TABLE
- TABELLE WIEDERHERSTELLEN
- TABELLE LÖSCHEN
- TYP ERSTELLEN
- TYP LÖSCHEN

Themen

- Keyspaces
- Tabellen
- User-defined Typen (UDTs)

Keyspaces

Ein Schlüsselraum gruppiert verwandte Tabellen, die für eine oder mehrere Anwendungen relevant sind. In Bezug auf ein relationales Datenbankmanagementsystem (RDBMS) ähneln Schlüsselräume in etwa Datenbanken, Tablespaces oder ähnlichen Konstrukten.

> **Note**
>
> In Apache Cassandra bestimmen Schlüsselräume, wie Daten zwischen mehreren Speicherknoten repliziert werden. Amazon Keyspaces ist jedoch ein vollständig verwalteter Service: Die Details der Speicherebene werden in Ihrem Namen verwaltet. Aus diesem Grund sind Schlüsselräume in Amazon Keyspaces nur logische Konstrukte und haben nichts mit dem zugrunde liegenden physischen Speicher zu tun.

Informationen zu Kontingentbeschränkungen und Einschränkungen für Amazon Keyspaces-Keyspaces finden Sie unter. Kontingente

Anweisungen für Schlüsselräume

- SCHLÜSSELRAUM ERSTELLEN
- SCHLÜSSELRAUM ÄNDERN
- SCHLÜSSELRAUM LÖSCHEN
- USE

SCHLÜSSELRAUM ERSTELLEN

Verwenden Sie die `CREATE KEYSPACE` Anweisung, um einen neuen Schlüsselraum zu erstellen.

Syntax

```
create_keyspace_statement ::=
    CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name
    WITH options
```

Wobei Folgendes gilt:

- *keyspace_name*ist der Name des zu erstellenden Schlüsselraums.
- Optionen sind eine oder mehrere der folgenden Optionen:
  - `REPLICATION`— Eine Map, die die Replikationsstrategie für den Schlüsselraum angibt:
    - `SingleRegionStrategy`— Für einen Schlüsselraum mit nur einer Region. (Erforderlich)
    - `NetworkTopologyStrategy`— Geben Sie mindestens zwei an. AWS-Regionen Der Replikationsfaktor für jede Region ist drei. (Optional)
  - `DURABLE_WRITES`— Schreibvorgänge in Amazon Keyspaces sind immer dauerhaft, sodass diese Option nicht erforderlich ist. Falls angegeben, muss der Wert jedoch sein. `true`
  - `TAGS`— Eine Liste von Schlüssel-Wert-Paar-Tags, die bei der Erstellung an die Ressource angehängt werden sollen. (Optional)

Beispiel

Erstellen Sie einen Schlüsselraum wie folgt.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class': 'SingleRegionStrategy'} and TAGS ={'key1':'val1',
 'key2':'val2'} ;
```

Um einen Schlüsselraum für mehrere Regionen zu erstellen, geben Sie mindestens zwei an `NetworkTopologyStrategy` und schließen Sie sie ein. AWS-Regionen Der Replikationsfaktor für jede Region ist drei.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3'};
```

SCHLÜSSELRAUM ÄNDERN

Sie können die *`ALTER KEYSPACE WITH`* Anweisung für die folgenden Optionen verwenden

- `REPLICATION`— Verwenden Sie diese Option, um einem Schlüsselraum ein neues AWS-Region Replikat hinzuzufügen. Sie können eine neue Region zu einem Schlüsselraum mit einer Region oder einem Schlüsselraum mit mehreren Regionen hinzufügen.
- `TAGS`— Verwenden Sie diese Option, um Tags zu einem Schlüsselraum hinzuzufügen oder daraus zu entfernen.

Syntax

```
alter_keyspace_statement ::=
    ALTER KEYSPACE keyspace_name
    WITH options
```

Wobei Folgendes gilt:

- *`keyspace_name`*ist der Name des Schlüsselraums, der geändert werden soll.
- Es gibt eine der folgenden Optionen:
  - `ADD | DROP TAGS`— Eine Liste von Schlüssel-Wert-Paar-Tags, die dem Schlüsselraum hinzugefügt oder daraus entfernt werden sollen.
  - `REPLICATION`— Eine Karte, die die Replikationsstrategie für den Schlüsselraum angibt;
    - `class`— `NetworkTopologyStrategy` definiert den Schlüsselraum als einen Schlüsselraum mit mehreren Regionen.

- `region`— Geben Sie einen zusätzlichen AWS-Region für diesen Schlüsselraum an. Der Replikationsfaktor für jede Region ist drei.
- `CLIENT_SIDE_TIMESTAMPS`— Die Standardeinstellung ist`DISABLED`. Sie können den Status nur auf ändern`ENABLED`.

Beispiele

Ändern Sie einen Schlüsselraum wie im folgenden Beispiel gezeigt, um Tags hinzuzufügen.

```
ALTER KEYSPACE my_keyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

Um einem Schlüsselraum mit mehreren Regionen eine dritte Region hinzuzufügen, können Sie die folgende Anweisung verwenden.

```
ALTER KEYSPACE my_keyspace
WITH REPLICATION = {
    'class': 'NetworkTopologyStrategy',
    'us-east-1': '3',
    'us-west-2': '3',
    'us-west-1': '3'
} AND CLIENT_SIDE_TIMESTAMPS = {'status': 'ENABLED'};
```

SCHLÜSSELRAUM LÖSCHEN

Verwenden Sie die `DROP KEYSPACE` Anweisung, um einen Schlüsselraum einschließlich seines gesamten Inhalts, z. B. Tabellen, zu entfernen.

Syntax

```
drop_keyspace_statement ::=
    DROP KEYSPACE [ IF EXISTS ] keyspace_name
```

Wobei Folgendes gilt:

- keyspace_name ist der Name des Schlüsselraums, der gelöscht werden soll.

Beispiel

```
DROP KEYSPACE my_keyspace;
```

USE

Verwenden Sie die USE Anweisung, um den aktuellen Schlüsselraum zu definieren. Auf diese Weise können Sie auf Objekte verweisen, die an einen bestimmten Schlüsselraum gebunden sind, z. B. Tabellen und Typen, ohne den vollqualifizierten Namen zu verwenden, der das Schlüsselraumpräfix enthält.

Syntax

```
use_statement ::=
    USE keyspace_name
```

Wobei Folgendes gilt:

- keyspace_name ist der Name des zu verwendenden Schlüsselraums.

Beispiel

```
USE my_keyspace;
```

Tabellen

Tabellen sind die primären Datenstrukturen in Amazon Keyspaces. Daten in einer Tabelle sind in Zeilen und Spalten organisiert. Eine Teilmenge dieser Spalten wird verwendet, um die Partitionierung (und letztlich die Datenplatzierung) durch die Angabe eines Partitionsschlüssels zu bestimmen.

Eine weitere Gruppe von Spalten kann in Clusterspalten definiert werden, was bedeutet, dass sie als Prädikate an der Abfrageausführung teilnehmen können.

Standardmäßig werden neue Tabellen mit On-Demand-Durchsatzkapazität erstellt. Sie können den Kapazitätsmodus für neue und bestehende Tabellen ändern. Weitere Informationen zu den Modi für den read/write Kapazitätsdurchsatz finden Sie unterthe section called “ read/write Kapazitätsmodi konfigurieren”.

Für Tabellen im Bereitstellungsmodus können Sie eine optionale AUTOSCALING_SETTINGS Konfiguration vornehmen. Weitere Informationen zur auto Skalierung von Amazon Keyspaces und den verfügbaren Optionen finden Sie unterthe section called “Konfigurieren Sie die automatische Skalierung für eine bestehende Tabelle”.

Informationen zu Kontingentbeschränkungen und Einschränkungen für Amazon Keyspaces-Tabellen finden Sie unterKontingente.

Aussagen für Tabellen

- CREATE TABLE
- ALTER TABLE
- TABELLE WIEDERHERSTELLEN
- DROP TABLE

CREATE TABLE

Verwenden Sie die `CREATE TABLE` Anweisung, um eine neue Tabelle zu erstellen.

Syntax

```
create_table_statement ::=  CREATE TABLE [ IF NOT EXISTS ] table_name
    '('
        column_definition
        ( ',' column_definition )*
        [ ',' PRIMARY KEY '(' primary_key ')' ]
    ')' [ WITH table_options ]

column_definition      ::=  column_name cql_type [ FROZEN ][ STATIC ][ PRIMARY KEY]

primary_key            ::=  partition_key [ ',' clustering_columns ]

partition_key          ::=  column_name
                              | '(' column_name ( ',' column_name )* ')'

clustering_columns     ::=  column_name ( ',' column_name )*

table_options          ::=  [table_options]
                              | CLUSTERING ORDER BY '(' clustering_order
 ')' [ AND table_options ]
                              | cdc
                              | CUSTOM_PROPERTIES
                              | AUTOSCALING_SETTINGS
                              | default_time_to_live
                              | TAGS
```

```
clustering_order       ::=  column_name (ASC | DESC) ( ',' column_name (ASC | DESC) )*
```

Wobei Folgendes gilt:

- *table_name*ist der Name der Tabelle, die erstellt werden soll. Der vollqualifizierte Name beinhaltet das Schlüsselraumpräfix. Alternativ können Sie den aktuellen Schlüsselraum mit der USE Keyspace-Anweisung festlegen.
- *column_definition*besteht aus den folgenden Komponenten:
  - *column_name*— Der Name der Spalte.
  - *cql_type*— Ein Amazon Keyspaces-Datentyp (siheDatentypen).
  - *FROZEN*— Kennzeichnet diese Spalte, die benutzerdefiniert ist oder deren Typ collection (z. B.,, oderMAP) ist LISTSET, als eingefroren. Eine eingefrorene Sammlung wird in einen einzigen unveränderlichen Wert serialisiert und wie eine behandelt. BLOB Weitere Informationen finden Sie unter the section called “Sammlungstypen”.
  - *STATIC*— Kennzeichnet diese Spalte als statisch. Statische Spalten speichern Werte, die von allen Zeilen in derselben Partition gemeinsam genutzt werden.
  - *PRIMARY KEY*— Bezeichnet diese Spalte als Primärschlüssel der Tabelle.
- *primary_key*besteht aus den folgenden Komponenten:
  - partition_key
  - clustering_columns
- *partition_key*:
  - Der Partitionsschlüssel kann eine einzelne Spalte oder ein zusammengesetzter Wert sein, der aus zwei oder mehr Spalten besteht. Der Partitionsschlüsselteil des Primärschlüssels ist erforderlich und bestimmt, wie Amazon Keyspaces Ihre Daten speichert.
- *clustering_columns*:
  - Der optionale Teil der Clustering-Spalte Ihres Primärschlüssels bestimmt, wie die Daten innerhalb der einzelnen Partitionen gruppiert und sortiert werden.
- *table_options*bestehen aus den folgenden Komponenten:
  - *CLUSTERING ORDER BY*— Die standardmäßige CLUSTERING-REIHENFOLGE für eine Tabelle setzt sich aus Ihren Gruppierungsschlüsseln in ASC (aufsteigender) Sortierrichtung zusammen. Geben Sie es an, um das Standard-Sortierverhalten zu überschreiben.
  - *cdc*— Ein boolescher Wert, der angibt, ob Amazon Keyspaces einen CDC-Stream (Change Data Capture) für die Tabelle erstellt. Der Standardwert ist false. Um bei der Aktivierung eines

`view type` Streams den anzugeben, legen Sie den Wert mit fest. `cdc_specification` `CUSTOM_PROPERTIES`

- *`CUSTOM_PROPERTIES`*— Eine Karte mit Einstellungen, die für Amazon Keyspaces spezifisch sind.
  - `capacity_mode`: Gibt den read/write Durchsatzkapazitätsmodus für die Tabelle an. Die Optionen sind `throughput_mode:PAY_PER_REQUEST` und `throughput_mode:PROVISIONED`. Der bereitgestellte Kapazitätsmodus erfordert `read_capacity_units` und `write_capacity_units` als Eingaben. Der Standardwert ist `throughput_mode:PAY_PER_REQUEST`.
  - `cdc_specification`: Gibt den Wert `view_type` des CDC-Streams an. Die Optionen sind:
    - `NEW_AND_OLD_IMAGES`— beide Versionen der Zeile, vor und nach der Änderung. Dies ist die Standardeinstellung.
    - `NEW_IMAGE`— die Version der Zeile nach der Änderung.
    - `OLD_IMAGE`— die Version der Zeile vor der Änderung.
    - `KEYS_ONLY`— die Partitions- und Clusterschlüssel der Zeile, die geändert wurde.

    Weitere Hinweise zu CDC-Streams finden Sie unter the section called “Arbeiten mit CDC-Streams” Codebeispiele finden Sie unter the section called “Aktivieren Sie einen CDC-Stream für eine neue Tabelle”.
  - `client_side_timestamps`: Gibt an, ob clientseitige Zeitstempel für die Tabelle aktiviert oder deaktiviert sind. Die Optionen sind `{'status': 'enabled'}` und `{'status': 'disabled'}`. Wenn es nicht angegeben ist, ist die Standardeinstellung. `status:disabled` Nachdem clientseitige Zeitstempel für eine Tabelle aktiviert wurden, kann diese Einstellung nicht deaktiviert werden.
  - `encryption_specification`: Gibt die Verschlüsselungsoptionen für die Verschlüsselung im Ruhezustand an. Wenn es nicht angegeben ist, ist die Standardeinstellung`encryption_type:AWS_OWNED_KMS_KEY`. Die vom Kunden verwaltete Verschlüsselungsoption erfordert den AWS KMS Schlüssel im Format Amazon Resource Name (ARN) als Eingabe:`kms_key_identifier:ARN`:`kms_key_identifier:ARN`.
  - `point_in_time_recovery`: Gibt an, ob die Point-in-Time-Wiederherstellung für die Tabelle aktiviert oder deaktiviert ist. Die Optionen sind `status:enabled` und `status:disabled`. Wenn es nicht angegeben ist, ist die Standardeinstellung. `status:disabled`
  - `replica_updates`: Gibt die Einstellungen einer Tabelle mit mehreren Regionen an, die spezifisch für eine AWS-Region sind. Bei einer Tabelle mit mehreren Regionen können Sie

die Lesekapazität der Tabelle je nach Region unterschiedlich konfigurieren. AWS-Region Sie können dies tun, indem Sie die folgenden Parameter konfigurieren. Weitere Informationen und Beispiele finden Sie unter the section called “Erstellen Sie eine Tabelle mit mehreren Regionen im Bereitstellungsmodus”.

- `region`— Das AWS-Region des Tabellenreplikats mit den folgenden Einstellungen:
  - `read_capacity_units`

- TTL: Aktiviert benutzerdefinierte Time to Live-Einstellungen für die Tabelle. Verwenden Sie zum Aktivieren`status:enabled`. Der Standardwert ist `status:disabled`. Nachdem TTL es aktiviert wurde, können Sie es für die Tabelle nicht mehr deaktivieren.

- *`AUTOSCALING_SETTINGS`*beinhaltet die folgenden optionalen Einstellungen für Tabellen im Bereitstellungsmodus. Weitere Informationen und Beispiele finden Sie unter the section called “Erstellen Sie eine neue Tabelle mit automatischer Skalierung”.
  - `provisioned_write_capacity_autoscaling_update`:
    - `autoscaling_disabled`— Um die auto Skalierung für die Schreibkapazität zu aktivieren, setzen Sie den Wert auf`false`. Der Standardwert ist `true`. (Optional)
    - `minimum_units`— Der Mindestwert für den Schreibdurchsatz, den die Tabelle immer unterstützen sollte. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
    - `maximum_units`— Der maximale Schreibdurchsatz, den die Tabelle immer unterstützen sollte. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
    - `scaling_policy`— Amazon Keyspaces unterstützt die Zielverfolgungsrichtlinie. Das Auto Scaling-Ziel ist die bereitgestellte Schreibkapazität der Tabelle.
      - `target_tracking_scaling_policy_configuration`— Um die Zielverfolgungsrichtlinie zu definieren, müssen Sie den Zielwert definieren. Weitere Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.
        - `target_value`— Die Zielauslastungsrate der Tabelle. Die auto Skalierung von Amazon Keyspaces stellt sicher, dass das Verhältnis von verbrauchter Kapazität zu bereitgestellter Kapazität auf oder nahe diesem Wert bleibt. Sie definieren `target_value` als Prozentsatz. Ein Doppelter zwischen 20 und 90. (Erforderlich)
        - `scale_in_cooldown`— Eine Abklingzeit in Sekunden zwischen den Skalierungsaktivitäten, sodass sich die Tabelle stabilisiert, bevor eine weitere

Aktivitätsskala beginnt. Wenn kein Wert angegeben wird, ist der Standardwert 0. (Optional)

- `scale_out_cooldown`— Eine Abklingzeit in Sekunden zwischen Skalierungsaktivitäten, durch die sich die Tabelle stabilisiert, bevor eine weitere Scale-Out-Aktivität beginnt. Wenn kein Wert angegeben wird, ist der Standardwert 0. (Optional)
- `disable_scale_in`: A`boolean`, das angibt, ob `scale-in` es für die Tabelle deaktiviert oder aktiviert ist. Dieser Parameter ist standardmäßig deaktiviert. Stellen Sie zum `scale-in` Einschalten den `boolean` Wert auf ein`FALSE`. Das bedeutet, dass die Kapazität für eine Tabelle in Ihrem Namen automatisch herunterskaliert wird. (Optional)

- `provisioned_read_capacity_autoscaling_update`:
  - `autoscaling_disabled`— Um die auto Skalierung für die Lesekapazität zu aktivieren, setzen Sie den Wert auf`false`. Der Standardwert ist `true`. (Optional)
  - `minimum_units`— Der Mindestdurchsatz, für den die Tabelle immer bereit sein sollte, ihn zu unterstützen. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
  - `maximum_units`— Der maximale Durchsatz, den die Tabelle immer unterstützen sollte. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
  - `scaling_policy`— Amazon Keyspaces unterstützt die Zielverfolgungsrichtlinie. Das Auto Scaling-Ziel ist die bereitgestellte Lesekapazität der Tabelle.
    - `target_tracking_scaling_policy_configuration`— Um die Zielverfolgungsrichtlinie zu definieren, müssen Sie den Zielwert definieren. Weitere Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.
      - `target_value`— Die Zielauslastungsrate der Tabelle. Die auto Skalierung von Amazon Keyspaces stellt sicher, dass das Verhältnis von verbrauchter Kapazität zu bereitgestellter Kapazität auf oder nahe diesem Wert bleibt. Sie definieren `target_value` als Prozentsatz. Ein Doppelter zwischen 20 und 90. (Erforderlich)
      - `scale_in_cooldown`— Eine Abklingzeit in Sekunden zwischen den Skalierungsaktivitäten, sodass sich die Tabelle stabilisiert, bevor eine weitere Aktivitätsskala beginnt. Wenn kein Wert angegeben wird, ist der Standardwert 0. (Optional)

- `scale_out_cooldown`— Eine Abklingzeit in Sekunden zwischen Skalierungsaktivitäten, durch die sich die Tabelle stabilisiert, bevor eine weitere Scale-Out-Aktivität beginnt. Wenn kein Wert angegeben wird, ist der Standardwert 0. (Optional)
- `disable_scale_in`: A`boolean`, das angibt, ob `scale-in` es für die Tabelle deaktiviert oder aktiviert ist. Dieser Parameter ist standardmäßig deaktiviert. Stellen Sie zum `scale-in` Einschalten den `boolean` Wert auf ein`FALSE`. Das bedeutet, dass die Kapazität für eine Tabelle in Ihrem Namen automatisch herunterskaliert wird. (Optional)

- `replica_updates`: Gibt die AWS-Region spezifischen Auto-Scaling-Einstellungen einer Tabelle mit mehreren Regionen an. Bei einer Tabelle mit mehreren Regionen können Sie die Lesekapazität der Tabelle je nach Region unterschiedlich konfigurieren. AWS-Region Sie können dies tun, indem Sie die folgenden Parameter konfigurieren. Weitere Informationen und Beispiele finden Sie unter the section called “Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling für eine Tabelle mit mehreren Regionen”.
  - `region`— Das AWS-Region des Tabellenreplikats mit den folgenden Einstellungen:
    - `provisioned_read_capacity_autoscaling_update`
      - `autoscaling_disabled`— Um die auto Skalierung für die Lesekapazität der Tabelle zu aktivieren, setzen Sie den Wert auf`false`. Der Standardwert ist `true`. (Optional)

        > **Note**
        >
        > Auto Scaling für eine Tabelle mit mehreren Regionen muss für alle Replikate der Tabelle entweder aktiviert oder deaktiviert sein.

      - `minimum_units`— Der Mindestwert an Lesedurchsatz, den die Tabelle immer unterstützen sollte. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
      - `maximum_units`— Der maximale Lesedurchsatz, den die Tabelle immer unterstützen sollte. Der Wert muss zwischen 1 und dem maximalen Kontingent für den Durchsatz pro Sekunde für Ihr Konto liegen (standardmäßig 40.000).
      - `scaling_policy`— Amazon Keyspaces unterstützt die Zielverfolgungsrichtlinie. Das Auto Scaling-Ziel ist die bereitgestellte Lesekapazität der Tabelle.
        - `target_tracking_scaling_policy_configuration`— Um die Zielverfolgungsrichtlinie zu definieren, müssen Sie den Zielwert definieren. Weitere

Informationen zu Target Tracking und Cooldown-Perioden finden Sie unter Target Tracking Scaling Policies im Application Auto Scaling User Guide.

- `target_value`— Die Zielauslastungsrate der Tabelle. Die auto Skalierung von Amazon Keyspaces stellt sicher, dass das Verhältnis zwischen verbrauchter Lesekapazität und bereitgestellter Lesekapazität bei oder nahe diesem Wert bleibt. Sie definieren `target_value` als Prozentsatz. Ein Doppelter zwischen 20 und 90. (Erforderlich)
- `scale_in_cooldown`— Eine Abklingzeit in Sekunden zwischen den Skalierungsaktivitäten, sodass sich die Tabelle stabilisiert, bevor eine weitere Aktivitätsskala beginnt. Wenn kein Wert angegeben wird, ist der Standardwert 0. (Optional)
- `scale_out_cooldown`— Eine Abklingzeit in Sekunden zwischen Skalierungsaktivitäten, durch die sich die Tabelle stabilisiert, bevor eine weitere Scale-Out-Aktivität beginnt. Wenn kein Wert angegeben wird, ist der Standardwert 0. (Optional)
- `disable_scale_in`: A`boolean`, das angibt, ob `scale-in` es für die Tabelle deaktiviert oder aktiviert ist. Dieser Parameter ist standardmäßig deaktiviert. Stellen Sie zum `scale-in` Einschalten den `boolean` Wert auf ein`FALSE`. Das bedeutet, dass die Lesekapazität für eine Tabelle in Ihrem Namen automatisch herunterskaliert wird. (Optional)

- `default_time_to_live`— Die Standardeinstellung für die Gültigkeitsdauer der Tabelle in Sekunden.
- `TAGS`— Eine Liste von Schlüssel-Wert-Paar-Tags, die bei ihrer Erstellung an die Ressource angehängt werden sollen.
- *`clustering_order`*besteht aus den folgenden Teilen:
  - *`column_name`*— Der Name der Spalte.
  - *`ASC | DESC`*— Legt den Modifikator für die aufsteigende (`ASC`) oder absteigende (`DESC`) Reihenfolge fest. Wenn er nicht angegeben ist, ist die Standardreihenfolge ASC.

Beispiel

```
CREATE TABLE IF NOT EXISTS my_keyspace.my_table (
                                        id text,
                                        name text,
                                        region text,
```

```
                                            division text,
                                            project text,
                                            role text,
                                            pay_scale int,
                                            vacation_hrs float,
                                            manager_id text,
                                            PRIMARY KEY (id,division))
                                            WITH CUSTOM_PROPERTIES={
                                                'capacity_mode':{
                                                        'throughput_mode':
 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units': 20
                                                    },
                                                'point_in_time_recovery':{'status':
 'enabled'},
                                                'encryption_specification':{
                                                        'encryption_type':
 'CUSTOMER_MANAGED_KMS_KEY',

 'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
                                                    }
                                            }
                                            AND CLUSTERING ORDER BY (division ASC)
                                            AND TAGS={'key1':'val1', 'key2':'val2'}
                                            AND default_time_to_live = 3024000;
```

In einer Tabelle, die Clusterspalten verwendet, können Spalten, die keine Clusterbildung bilden, in der Tabellendefinition als statisch deklariert werden. Weitere Hinweise zu statischen Spalten finden Sie unter. the section called “Schätzen Sie den Kapazitätsverbrauch statischer Spalten”

Beispiel

```
CREATE TABLE my_keyspace.my_table (
                                            id int,
                                            name text,
                                            region text,
                                            division text,
                                            project text STATIC,
                                            PRIMARY KEY (id,division));
```

Sie können eine Tabelle mit einer Spalte erstellen, die einen benutzerdefinierten Typ (UDT) verwendet. Die erste Anweisung in den Beispielen erstellt einen Typ, die zweite Anweisung erstellt eine Tabelle mit einer Spalte, die diesen Typ verwendet.

Beispiel

```
CREATE TYPE my_keyspace."udt""N@ME" (my_field int);
CREATE TABLE my_keyspace.my_table (my_col1 int pri key, my_col2 "udt""N@ME");
```

ALTER TABLE

Verwenden Sie die `ALTER TABLE` Anweisung, um neue Spalten hinzuzufügen, Tags hinzuzufügen oder die benutzerdefinierten Eigenschaften der Tabelle zu ändern.

Syntax

```
alter_table_statement ::=  ALTER TABLE table_name

        [ ADD ( column_definition | column_definition_list)  ]
        [[ADD | DROP] TAGS {'key1':'val1', 'key2':'val2'}]
        [ WITH table_options [ , ... ] ] ;

column_definition      ::=  column_name cql_type
```

Wobei Folgendes gilt:

- *`table_name`*ist der Name der Tabelle, die geändert werden soll.
- *`column_definition`*ist der Name der Spalte und des Datentyps, die hinzugefügt werden sollen.
- *`column_definition_list`*ist eine durch Kommas getrennte Liste von Spalten in Klammern.
- *`table_options`*besteht aus den folgenden Komponenten:
  - *`AUTOSCALING_SETTINGS`*beinhaltet die optionalen Auto Scaling-Einstellungen für bereitgestellte Tabellen. Die Syntax und ausführliche Beschreibungen finden Sie unterthe section called “CREATE TABLE”. Beispiele finden Sie unter the section called “Konfigurieren Sie die automatische Skalierung für eine bestehende Tabelle”.
  - *`cdc`*— Ein boolescher Wert, der angibt, ob Amazon Keyspaces einen CDC-Stream (Change Data Capture) für die Tabelle erstellt. Der Standardwert ist `false`. Um bei der Aktivierung eines

`view type` Streams den anzugeben, legen Sie den Wert mit fest. `cdc_specification` `CUSTOM_PROPERTIES`

- *`CUSTOM_PROPERTIES`*— Eine Karte mit Einstellungen, die für Amazon Keyspaces spezifisch sind.
  - `capacity_mode`: Gibt den read/write Durchsatzkapazitätsmodus für die Tabelle an. Die Optionen sind `throughput_mode:PAY_PER_REQUEST` und `throughput_mode:PROVISIONED`. Der bereitgestellte Kapazitätsmodus erfordert `read_capacity_units` und `write_capacity_units` als Eingaben. Der Standardwert ist `throughput_mode:PAY_PER_REQUEST`.
  - `cdc_specification`: Gibt den Wert `view_type` des CDC-Streams an. Die Optionen sind:
    - `NEW_AND_OLD_IMAGES`— beide Versionen der Zeile, vor und nach der Änderung. Dies ist die Standardeinstellung.
    - `NEW_IMAGE`— die Version der Zeile nach der Änderung.
    - `OLD_IMAGE`— die Version der Zeile vor der Änderung.
    - `KEYS_ONLY`— die Partitions- und Clusterschlüssel der Zeile, die geändert wurde.

    Weitere Hinweise zu CDC-Streams finden Sie unter the section called “Arbeiten mit CDC-Streams” Codebeispiele finden Sie unter the section called “Aktivieren Sie einen CDC-Stream für eine vorhandene Tabelle”.
  - `client_side_timestamps`: Gibt an, ob clientseitige Zeitstempel für die Tabelle aktiviert oder deaktiviert sind. Die Optionen sind `{'status': 'enabled'}` und `{'status': 'disabled'}`. Wenn es nicht angegeben ist, ist die Standardeinstellung. `status:disabled` Nachdem clientseitige Zeitstempel für eine Tabelle aktiviert wurden, kann diese Einstellung nicht deaktiviert werden.
  - `encryption_specification`: Gibt die Verschlüsselungsoption für die Verschlüsselung im Ruhezustand an. Die Optionen sind `encryption_type:AWS_OWNED_KMS_KEY` und `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. Die vom Kunden verwaltete Verschlüsselungsoption erfordert den AWS KMS Schlüssel im Format Amazon Resource Name (ARN) als Eingabe:`kms_key_identifier:ARN`.
  - `point_in_time_recovery`: Gibt an, ob die Point-in-Time-Wiederherstellung für die Tabelle aktiviert oder deaktiviert ist. Die Optionen sind `status:enabled` und `status:disabled`. Der Standardwert ist `status:disabled`.
  - `replica_updates`: Gibt die AWS-Region spezifischen Einstellungen einer Tabelle mit mehreren Regionen an. Bei einer Tabelle mit mehreren Regionen können Sie die

Lesekapazität der Tabelle je nach Region unterschiedlich konfigurieren. AWS-Region Sie können dies tun, indem Sie die folgenden Parameter konfigurieren. Weitere Informationen und Beispiele finden Sie unter the section called “Aktualisieren Sie die Einstellungen für bereitgestellte Kapazität und Auto Scaling für eine Tabelle mit mehreren Regionen”.

- `region`— Das AWS-Region des Tabellenreplikats mit den folgenden Einstellungen:
  - `read_capacity_units`

- `ttl`: Aktiviert benutzerdefinierte Time to Live-Einstellungen für die Tabelle. Verwenden Sie zum Aktivieren`status:enabled`. Der Standardwert ist `status:disabled`. Nachdem `ttl` es aktiviert wurde, können Sie es für die Tabelle nicht mehr deaktivieren.

- `default_time_to_live`: Die Standardeinstellung „Time to Live“ in Sekunden für die Tabelle.
- *`TAGS`*ist eine Liste von Schlüssel-Wert-Paar-Tags, die an die Ressource angehängt werden sollen.

> **Note**
>
> Mit ALTER TABLE können Sie nur eine einzige benutzerdefinierte Eigenschaft ändern. Sie können nicht mehr als einen ALTER TABLE-Befehl in derselben Anweisung kombinieren.

Beispiele

Die folgende Anweisung zeigt, wie einer vorhandenen Tabelle eine Spalte hinzugefügt wird.

```
ALTER TABLE mykeyspace.mytable ADD (ID int);
```

Diese Anweisung zeigt, wie zwei Sammlungsspalten zu einer vorhandenen Tabelle hinzugefügt werden:

- Eine eingefrorene Sammlungsspalte`col_frozen_list`, die eine verschachtelte eingefrorene Sammlung enthält
- Eine nicht eingefrorene Sammlungsspalte`col_map`, die eine verschachtelte eingefrorene Sammlung enthält

```
ALTER TABLE my_Table ADD(col_frozen_list FROZEN<LIST<FROZEN<SET<TEXT>>>>, col_map
 MAP<INT, FROZEN<SET<INT>>>);
```

Das folgende Beispiel zeigt, wie einer Tabelle eine Spalte hinzugefügt wird, die einen benutzerdefinierten Typ (UDT) verwendet.

```
ALTER TABLE my_keyspace.my_table ADD (my_column, my_udt;);
```

Um den Kapazitätsmodus einer Tabelle zu ändern und Lese- und Schreibkapazitätseinheiten anzugeben, können Sie die folgende Anweisung verwenden.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units':
 20}};
```

Die folgende Anweisung gibt einen vom Kunden verwalteten KMS-Schlüssel für die Tabelle an.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={
              'encryption_specification':{
                     'encryption_type': 'CUSTOMER_MANAGED_KMS_KEY',
                     'kms_key_identifier':'arn:aws:kms:eu-
west-1:5555555555555:key/11111111-1111-111-1111-111111111111'
                  }
         };
```

Um die Point-in-Time-Wiederherstellung für eine Tabelle zu aktivieren, können Sie die folgende Anweisung verwenden.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'point_in_time_recovery':
 {'status': 'enabled'}};
```

Um einen Standardwert für die Gültigkeitsdauer in Sekunden für eine Tabelle festzulegen, können Sie die folgende Anweisung verwenden.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Diese Anweisung ermöglicht benutzerdefinierte Time-to-Live-Einstellungen für eine Tabelle.

```
ALTER TABLE mytable WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

TABELLE WIEDERHERSTELLEN

Verwenden Sie die `RESTORE TABLE` Anweisung, um eine Tabelle auf einen bestimmten Zeitpunkt zurückzusetzen. Für diese Anweisung muss die Point-in-Time-Wiederherstellung für eine Tabelle aktiviert sein. Weitere Informationen finden Sie unter the section called “Backup und Wiederherstellung mit Point-in-Time-Recovery”.

Syntax

```
restore_table_statement ::=
    RESTORE TABLE restored_table_name FROM TABLE source_table_name
                    [ WITH table_options [ , ... ] ];
```

Wobei Folgendes gilt:

- *`restored_table_name`*ist der Name der wiederhergestellten Tabelle.
- *`source_table_name`*ist der Name der Quelltabelle.
- *`table_options`*besteht aus den folgenden Teilen:
  - *`restore_timestamp`*ist die Uhrzeit des Wiederherstellungspunkts im Format ISO 8601. Wenn er nicht angegeben ist, wird der aktuelle Zeitstempel verwendet.
  - *`CUSTOM_PROPERTIES`*— Eine Karte mit Einstellungen, die für Amazon Keyspaces spezifisch sind.
    - `capacity_mode`: Gibt den read/write Durchsatzkapazitätsmodus für die Tabelle an. Die Optionen sind `throughput_mode:PAY_PER_REQUEST` und `throughput_mode:PROVISIONED`. Der bereitgestellte Kapazitätsmodus erfordert `read_capacity_units` und `write_capacity_units` als Eingaben. Die Standardeinstellung ist die aktuelle Einstellung aus der Quelltabelle.
    - `encryption_specification`: Gibt die Verschlüsselungsoption für die Verschlüsselung im Ruhezustand an. Die Optionen sind `encryption_type:AWS_OWNED_KMS_KEY` und `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. Die vom Kunden verwaltete Verschlüsselungsoption erfordert den AWS KMS Schlüssel im Format Amazon Resource Name (ARN) als Eingabe:`kms_key_identifier:ARN`. Um eine mit einem vom Kunden verwalteten Schlüssel verschlüsselte Tabelle in eine mit einem verschlüsselte Tabelle wiederherzustellen AWS-eigener Schlüssel, benötigt Amazon Keyspaces Zugriff auf den AWS KMS Schlüssel der Quelltabelle.
    - `point_in_time_recovery`: Gibt an, ob die Point-in-Time-Wiederherstellung für die Tabelle aktiviert oder deaktiviert ist. Die Optionen sind `status:enabled` und `status:disabled`.

Anders als beim Erstellen neuer Tabellen ist der Standardstatus für wiederhergestellte Tabellen darauf zurückzuführen, `status:enabled` dass die Einstellung von der Quelltabelle übernommen wird. Um PITR für wiederhergestellte Tabellen zu deaktivieren, müssen Sie `status:disabled` dies explizit festlegen.

- `replica_updates`: Gibt die AWS-Region spezifischen Einstellungen einer Tabelle mit mehreren Regionen an. Bei einer Tabelle mit mehreren Regionen können Sie die Lesekapazität der Tabelle je nach Region unterschiedlich konfigurieren. AWS-Region Sie können dies tun, indem Sie die folgenden Parameter konfigurieren.
  - `region`— Das AWS-Region des Tabellenreplikats mit den folgenden Einstellungen:
    - `read_capacity_units`

- *`AUTOSCALING_SETTINGS`*beinhaltet die optionalen Auto Scaling-Einstellungen für bereitgestellte Tabellen. Eine ausführliche Syntax und Beschreibungen finden Sie unterthe section called “CREATE TABLE”.
- *`TAGS`*ist eine Liste von Schlüssel-Wert-Paar-Tags, die an die Ressource angehängt werden sollen.

> **Note**
>
> Gelöschte Tabellen können nur zum Zeitpunkt des Löschvorgangs wiederhergestellt werden.

Beispiel

```
RESTORE TABLE mykeyspace.mytable_restored from table mykeyspace.my_table
WITH restore_timestamp = '2020-06-30T04:05:00+0000'
AND custom_properties = {'point_in_time_recovery':{'status':'disabled'},
 'capacity_mode':{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10,
 'write_capacity_units': 20}}
AND TAGS={'key1':'val1', 'key2':'val2'};
```

DROP TABLE

Verwenden Sie die `DROP TABLE` Anweisung, um eine Tabelle aus dem Schlüsselraum zu entfernen.

Syntax

```
drop_table_statement ::=
```

```
    DROP TABLE [ IF EXISTS ] table_name
```

Wobei Folgendes gilt:

- `IF EXISTSDROP TABLE`verhindert, dass ein Fehler auftritt, wenn die Tabelle nicht existiert. (Optional)
- *`table_name`*ist der Name der Tabelle, die gelöscht werden soll.

Beispiel

```
DROP TABLE my_keyspace.my_table;
```

User-defined Typen (UDTs)

UDT — Eine Gruppierung von Feldern und Datentypen, die Sie verwenden können, um eine einzelne Spalte in Amazon Keyspaces zu definieren. Gültige Datentypen für UDTs sind alle unterstützten Cassandra-Datentypen, einschließlich Sammlungen und anderer UDTs, die Sie bereits im selben Schlüsselraum erstellt haben. Weitere Informationen zu den unterstützten Cassandra-Datentypen finden Sie unter. the section called “Unterstützung des Cassandra-Datentyps”

```
user_defined_type::= udt_name
udt_name::= [ keyspace_name '.' ] identifier
```

Anweisungen für Typen

- CREATE TYPE
- DROP TYPE

CREATE TYPE

Verwenden Sie die `CREATE  TYPE` Anweisung, um einen neuen Typ zu erstellen.

Syntax

```
create_type_statement ::=  CREATE TYPE [ IF NOT EXISTS ] udt_name
    '('field_definition ( ',' field_definition)* ')'
            field_definition::= identifier cql_type
```

Wobei Folgendes gilt:

- `IF NOT EXISTSCREATE TYPE`verhindert, dass ein Fehler auftritt, wenn der Typ bereits existiert. (Optional)
- *`udt_name`*ist beispielsweise der vollständig qualifizierte Name der UDT im Typformat. `my_keyspace.my_type` Wenn Sie den aktuellen Schlüsselraum mit der `USE` Anweisung definieren, müssen Sie den Schlüsselraumnamen nicht angeben.
- *`field_definition`*besteht aus einem Namen und einem Typ.

Die folgende Tabelle zeigt Beispiele für zulässige UDT-Namen. Die erste Spalte zeigt, wie Sie den Namen eingeben, wenn Sie den Typ erstellen, die zweite Spalte zeigt, wie Amazon Keyspaces den Namen intern formatiert. Amazon Keyspaces erwartet den formatierten Namen für Operationen wie. `GetType`

Eingegebener Name	Formatierter Name	Hinweis
`MY_UDT`	`my_udt`	Ohne doppelte Anführungszeichen konvertiert Amazon Keyspaces alle Großbuchstaben in Kleinbuchstaben.
`"MY_UDT"`	`MY_UDT`	Bei doppelten Anführungszeichen respektiert Amazon Keyspaces die Großbuchstaben und entfernt die doppelten Anführungszeichen aus dem formatierten Namen.
`"1234"`	`1234`	Bei doppelten Anführungszeichen kann der Name mit einer Zahl beginnen, und Amazon Keyspaces entfernt die doppelten Anführungszeichen aus dem formatierten Namen.
`"Special_ Ch@r@cter s<>!!"`	`Special_C h@r@cters <>!!`	Bei doppelten Anführungszeichen kann der Name Sonderzeichen enthalten, und Amazon Keyspaces entfernt die doppelten Anführungszeichen aus dem formatierten Namen.
`"nested"" """"quote s"`	`nested""" quotes`	Amazon Keyspaces entfernt die äußeren doppelten Anführungszeichen und die doppelten Escape-An führungszeichen aus dem formatierten Namen.

Beispiele

```
CREATE TYPE my_keyspace.phone (
    country_code int,
    number text
);
```

Sie können UDTs verschachteln, wenn die verschachtelte UDT eingefroren ist. Weitere Informationen zu Standardwerten und Kontingenten für Typen finden Sie unter. the section called “Amazon Keyspaces UDT-Kontingente und Standardwerte”

```
CREATE TYPE my_keyspace.user (
    first_name text,
    last_name text,
    phones FROZEN<phone>
);
```

Weitere Codebeispiele, die zeigen, wie UDTs erstellt werden, finden Sie unterthe section called “User-defined Typen (UDTs)”.

DROP TYPE

Verwenden Sie die `DROP TYPE` Anweisung, um eine UDT zu löschen. Sie können nur einen Typ löschen, der nicht von einem anderen Typ oder einer anderen Tabelle verwendet wird.

Syntax

```
drop_type_statement ::=  DROP TYPE [ IF EXISTS ] udt_name
```

Wobei Folgendes gilt:

- `IF EXISTSDROP TYPE`verhindert, dass ein Fehler auftritt, wenn der Typ nicht existiert. (Optional)
- *`udt_name`*ist beispielsweise der vollständig qualifizierte Name der UDT im Typformat. `my_keyspace.my_type` Wenn Sie den aktuellen Schlüsselraum mit der `USE` Anweisung definieren, müssen Sie den Schlüsselraumnamen nicht angeben.

Beispiel

```
DROP TYPE udt_name;
```

DML-Anweisungen (Datenmanipulationssprache) in Amazon Keyspaces

Data Manipulation Language (DML) ist der Satz von Cassandra Query Language (CQL) -Anweisungen, mit denen Sie Daten in Amazon Keyspaces-Tabellen (für Apache Cassandra) verwalten. Sie verwenden DML-Anweisungen, um Daten in einer Tabelle hinzuzufügen, zu ändern oder zu löschen.

Sie verwenden DML-Anweisungen auch, um Daten in einer Tabelle abzufragen. (Beachten Sie, dass CQL keine Verknüpfungen oder Unterabfragen unterstützt.)

Themen

- SELECT
- INSERT
- UPDATE
- DELETE

SELECT

Verwenden Sie eine SELECT-Anweisung, um Daten abzufragen.

Syntax

```
select_statement ::=  SELECT  [ JSON ] ( select_clause | '*' )
                      FROM table_name
                      [ WHERE 'where_clause' ]
                      [ ORDER BY 'ordering_clause' ]
                      [ LIMIT (integer | bind_marker) ]
                      [ ALLOW FILTERING ]
select_clause    ::=  selector [ AS identifier ] ( ',' selector [ AS identifier ] )
selector         ::=  column_name
                      | term
                      | CAST '(' selector AS cql_type ')'
                      | function_name '(' [ selector ( ',' selector )* ] ')'
where_clause     ::=  relation ( AND relation )*
relation         ::=  column_name operator term
                      TOKEN
operator         ::=  '=' | '<' | '>' | '<=' | '>=' | IN | CONTAINS | CONTAINS KEY
```

```
ordering_clause  ::=  column_name [ ASC | DESC ] ( ',' column_name [ ASC | DESC ] )*
```

Beispiele

```
SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;

SELECT JSON name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

Eine Tabelle, die JSON-encoded Datentypen Amazon Keyspaces-Datentypen zuordnet, finden Sie unterthe section called “JSON-Kodierung von Amazon Keyspaces-Datentypen”.

Verwenden Sie das Schlüsselwort **IN**

Das IN Schlüsselwort gibt die Gleichheit für einen oder mehrere Werte an. Es kann auf den Partitionsschlüssel und die Clusterspalte angewendet werden. Die Ergebnisse werden in der Reihenfolge zurückgegeben, in der die Schlüssel in der SELECT Anweisung dargestellt werden.

Beispiele

```
SELECT * from mykeyspace.mytable WHERE primary.key1 IN (1,2) and clustering.key1 = 2;
SELECT * from mykeyspace.mytable WHERE primary.key1 IN (1,2) and clustering.key1 <= 2;
SELECT * from mykeyspace.mytable WHERE primary.key1 = 1 and clustering.key1 IN (1, 2);
SELECT * from mykeyspace.mytable WHERE primary.key1 <= 2 and clustering.key1 IN (1, 2)
 ALLOW FILTERING;
```

Weitere Informationen zum IN Schlüsselwort und zur Verarbeitung der Anweisung durch Amazon Keyspaces finden Sie unterthe section called “Verwenden IN SELECT”.

Ergebnisse bestellen

Die ORDER BY Klausel gibt die Sortierreihenfolge der zurückgegebenen Ergebnisse an. Sie verwendet als Argumente eine Liste von Spaltennamen zusammen mit der Sortierreihenfolge für jede Spalte. Sie können Clusterspalten nur in Sortierklauseln angeben. Non-clusteringSpalten sind nicht zulässig. Die Optionen für die Sortierreihenfolge gelten ASC für die aufsteigende und DESC für die absteigende Sortierreihenfolge. Wenn die Sortierreihenfolge weggelassen wird, wird die Standardreihenfolge der Clusterspalte verwendet. Mögliche Sortierreihenfolgen finden Sie unterthe section called “Ergebnisse bestellen”.

Beispiel

```
SELECT name, id, division, manager_id FROM "myGSGKeyspace".employees_tbl WHERE id =
 '012-34-5678' ORDER BY division;
```

Bei Verwendung `ORDER BY` mit dem `IN` Schlüsselwort werden die Ergebnisse innerhalb einer Seite angeordnet. Eine vollständige Neuanordnung mit deaktivierter Paginierung wird nicht unterstützt.

TOKEN

Sie können die `TOKEN` Funktion auf die `PARTITION KEY` Spalten in `SELECT` und `WHERE` -Klauseln anwenden. Mit dieser `TOKEN` Funktion gibt Amazon Keyspaces Zeilen zurück, die auf dem zugewiesenen Token-Wert von basieren `PARTITION_KEY` und nicht auf dem Wert von. `PARTITION KEY`

`TOKEN`Relationen werden mit dem Schlüsselwort nicht unterstützt. `IN`

Beispiele

```
SELECT TOKEN(id) from my_table;

SELECT TOKEN(id) from my_table WHERE TOKEN(id) > 100 and TOKEN(id) < 10000;
```

TTL-Funktion

Sie können die `TTL` Funktion mit der `SELECT` Anweisung verwenden, um die für eine Spalte gespeicherte Ablaufzeit in Sekunden abzurufen. Wenn kein `TTL` Wert festgelegt ist, kehrt die Funktion zurück`null`.

Beispiel

```
SELECT TTL(my_column) from my_table;
```

Die `TTL` Funktion kann nicht für Spalten mit mehreren Zellen wie Sammlungen verwendet werden.

WRITETIMEFunktion

Sie können die `WRITETIME` Funktion mit der `SELECT` Anweisung nur dann verwenden, um den Zeitstempel abzurufen, der als Metadaten für den Wert einer Spalte gespeichert ist, wenn die Tabelle clientseitige Zeitstempel verwendet. Weitere Informationen finden Sie unter the section called “Client-side Zeitstempel”.

```
SELECT WRITETIME(my_column) from my_table;
```

Die `WRITETIME` Funktion kann nicht für Spalten mit mehreren Zellen wie Sammlungen verwendet werden.

> **Note**
>
> Aus Gründen der Kompatibilität mit dem gängigen Verhalten von Cassandra-Treibern werden tagbasierte Autorisierungsrichtlinien nicht durchgesetzt, wenn Sie mithilfe von Cassandra Query Language (CQL) -API-Aufrufen über Cassandra-Treiber und Entwicklertools Operationen an Systemtabellen ausführen. Weitere Informationen finden Sie unter the section called “Amazon Keyspaces-Ressourcenzugriff basierend auf Tags”.

INSERT

Verwenden Sie die `INSERT` Anweisung, um einer Tabelle eine Zeile hinzuzufügen.

Syntax

```
insert_statement ::=  INSERT INTO table_name ( names_values | json_clause )
                      [ IF NOT EXISTS ]
                      [ USING update_parameter ( AND update_parameter )* ]
names_values     ::=  names VALUES tuple_literal
json_clause      ::=  JSON string [ DEFAULT ( NULL | UNSET ) ]
names            ::=  '(' column_name ( ',' column_name )* ')'
```

Beispiel

```
INSERT INTO "myGSGKeyspace".employees_tbl (id, name, project, region, division, role,
 pay_scale, vacation_hrs, manager_id)
VALUES ('012-34-5678','Russ','NightFlight','US','Engineering','IC',3,12.5,
 '234-56-7890') ;
```

Parameter aktualisieren

`INSERT`unterstützt die folgenden Werte als`update_parameter`:

- TTL— Ein Zeitwert in Sekunden. Der maximal konfigurierbare Wert ist 630 720 000 Sekunden, was 20 Jahren entspricht.

- `TIMESTAMP`— Ein `bigint` Wert, der die Anzahl der Mikrosekunden seit der Standardbasiszeit darstellt, die alsepoch: 1. Januar 1970 um 00:00:00 Uhr GMT bekannt ist. Ein Zeitstempel in Amazon Keyspaces muss zwischen 2 Tagen in der Vergangenheit und 5 Minuten in der future liegen.

Beispiel

```
INSERT INTO my_table (userid, time, subject, body, user)
        VALUES (B79CB3BA-745E-5D9A-8903-4A02327A7E09, 96a29100-5e25-11ec-90d7-
b5d91eceda0a, 'Message', 'Hello','205.212.123.123')
        USING TTL 259200;
```

JSON-Unterstützung

Eine Tabelle, die JSON-encoded Datentypen Amazon Keyspaces-Datentypen zuordnet, finden Sie unterthe section called “JSON-Kodierung von Amazon Keyspaces-Datentypen”.

Sie können das `JSON` Schlüsselwort verwenden, um eine `JSON` -kodierte Map als einzelne Zeile einzufügen. Verwenden Sie für Spalten, die in der Tabelle vorhanden sind, aber in der JSON-Einfügeanweisung weggelassen werden, `DEFAULT UNSET` um die vorhandenen Werte beizubehalten. Wird verwendet`DEFAULT NULL`, um in jede Zeile ausgelassener Spalten einen NULL-Wert zu schreiben und die vorhandenen Werte zu überschreiben (es fallen die üblichen Schreibgebühren an). `DEFAULT NULL`ist die Standardoption.

Beispiel

```
INSERT INTO "myGSGKeyspace".employees_tbl JSON '{"id":"012-34-5678",
                                                 "name": "Russ",
                                                 "project": "NightFlight",
                                                 "region": "US",
                                                 "division": "Engineering",
                                                 "role": "IC",
                                                 "pay_scale": 3,
                                                 "vacation_hrs": 12.5,
                                                 "manager_id": "234-56-7890"}';
```

Wenn die JSON-Daten doppelte Schlüssel enthalten, speichert Amazon Keyspaces den letzten Wert für den Schlüssel (ähnlich wie Apache Cassandra). Im folgenden Beispiel, in dem sich der doppelte Schlüssel befinde`id`, wird der Wert verwendet`234-56-7890`.

Beispiel

```
INSERT INTO "myGSGKeyspace".employees_tbl JSON '{"id":"012-34-5678",
                                                 "name": "Russ",
                                                 "project": "NightFlight",
                                                 "region": "US",
                                                 "division": "Engineering",
                                                 "role": "IC",
                                                 "pay_scale": 3,
                                                 "vacation_hrs": 12.5,
                                                 "id": "234-56-7890"}';
```

UPDATE

Verwenden Sie die UPDATE Anweisung, um eine Zeile in einer Tabelle zu ändern.

Syntax

```
update_statement ::=  UPDATE table_name
                      [ USING update_parameter ( AND update_parameter )* ]
                      SET assignment ( ',' assignment )*
                      WHERE where_clause
                      [ IF ( EXISTS | condition ( AND condition )*) ]
update_parameter ::=  ( integer | bind_marker )
assignment       ::=  simple_selection '=' term
                     | column_name '=' column_name ( '+' | '-' ) term
                     | column_name '=' list_literal '+' column_name
simple_selection ::=  column_name
                     | column_name '[' term ']'
                     | column_name '.' `field_name
condition        ::=  simple_selection operator term
```

Beispiel

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale = 5 WHERE id = '567-89-0123' AND
 division = 'Marketing' ;
```

Verwenden Sie die folgende Syntax`counter`, um a zu inkrementieren. Weitere Informationen finden Sie unter the section called “Zähler”.

```
UPDATE ActiveUsers SET counter = counter + 1  WHERE user = A70FE1C0-5408-4AE3-
BE34-8733E5K09F14 AND action = 'click';
```

Parameter aktualisieren

UPDATEunterstützt die folgenden Werte alsupdate_parameter:

- TTL— Ein Zeitwert in Sekunden. Der maximal konfigurierbare Wert ist 630 720 000 Sekunden, was 20 Jahren entspricht.
- TIMESTAMP— Ein bigint Wert, der die Anzahl der Mikrosekunden seit der Standardbasiszeit darstellt, die alsepoch: 1. Januar 1970 um 00:00:00 Uhr GMT bekannt ist. Ein Zeitstempel in Amazon Keyspaces muss zwischen 2 Tagen in der Vergangenheit und 5 Minuten in der future liegen.

Beispiel

```
UPDATE my_table (userid, time, subject, body, user)
        VALUES (B79CB3BA-745E-5D9A-8903-4A02327A7E09, 96a29100-5e25-11ec-90d7-
b5d91eceda0a, 'Message', 'Hello again','205.212.123.123')
        USING TIMESTAMP '2022-11-03 13:30:54+0400';
```

DELETE

Verwenden Sie die DELETE Anweisung, um eine Zeile aus einer Tabelle zu entfernen.

Syntax

```
delete_statement ::=  DELETE [ simple_selection ( ',' simple_selection ) ]
                      FROM table_name
                      [ USING update_parameter ( AND update_parameter )* ]
                      WHERE where_clause
                      [ IF ( EXISTS | condition ( AND condition )*) ]

simple_selection ::=  column_name
                     | column_name '[' term ']'
                     | column_name '.' `field_name

condition        ::=  simple_selection operator term
```

Wobei Folgendes gilt:

- *table_name*ist die Tabelle, die die Zeile enthält, die Sie löschen möchten.

Beispiel

```
DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
 division='Executive' ;
```

`DELETE`unterstützt den folgenden Wert als`update_parameter`:

- `TIMESTAMP`— Ein `bigint` Wert, der die Anzahl der Mikrosekunden seit der Standardbasiszeit darstellt, die alsepoch: 1. Januar 1970 um 00:00:00 GMT bekannt ist.

Built-in Funktionen in Amazon Keyspaces

Amazon Keyspaces (für Apache Cassandra) unterstützt eine Vielzahl integrierter Funktionen, die Sie in Cassandra Query Language (CQL) -Anweisungen verwenden können.

Themen

- Skalarfunktionen

Skalarfunktionen

Eine Skalarfunktion führt eine Berechnung für einen einzelnen Wert durch und gibt das Ergebnis als einzelnen Wert zurück. Amazon Keyspaces unterstützt die folgenden Skalarfunktionen.

Funktion	Description
`blobAsType`	Gibt einen Wert des angegebenen Datentyps zurück.
`cast`	Konvertiert einen systemeigenen Datentyp in einen anderen systemeigenen Datentyp.
`currentDate`	Gibt den aktuellen Wert date/time als Datum zurück.

Funktion	Description
`currentTime`	Gibt den aktuellen Wert date/time als Uhrzeit zurück.
`currentTimestamp`	Gibt den aktuellen Wert date/time als Zeitstempel zurück.
`currentTimeUUID`	Gibt den aktuellen Wert date/time als `timeuuid` zurück.
`fromJson`	Konvertiert die JSON-Zeichenfolge in den Datentyp der ausgewählten Spalte.
`maxTimeuuid`	Gibt den größtmöglichen Wert `timeuuid` für einen Zeitstempel oder eine Datumszeichenfolge zurück.
`minTimeuuid`	Gibt den kleinstmöglichen Wert `timeuuid` für einen Zeitstempel oder eine Datumszeichenfolge zurück.
`now`	Gibt ein neues Unikat `timeuuid` zurück.
`toDate`	Konvertiert entweder einen Zeitstempel `timeuuid` oder einen Zeitstempel in einen Datumstyp.
`toJson`	Gibt den Spaltenwert der ausgewählten Spalte im JSON-Format zurück.
`token`	Gibt den Hashwert des Partitionsschlüssels zurück.
`toTimestamp`	Konvertiert entweder ein `timeuuid` oder ein Datum in einen Zeitstempel.
`TTL`	Gibt die Ablaufzeit in Sekunden für eine Spalte zurück.

Funktion	Description
`typeAsBlob`	Konvertiert den angegebenen Datentyp in einen`blob`.
`toUnixTimestamp`	Konvertiert entweder einen Zeitstemp el `timeuuid` oder einen Zeitstempel in einen`bigInt`.
`uuid`	Gibt eine zufällige Version 4-UUID zurück.
`writetime`	Gibt den Zeitstempel des Werts der angegeben en Spalte zurück.
`dateOf`	(Veraltet) Extrahiert den Zeitstempel von a `timeuuid` und gibt den Wert als Datum zurück.
`unixTimestampOf`	(Veraltet) Extrahiert den Zeitstempel von a und gibt den Wert als `timeuuid` rohen 64-Bit-Ga nzzahl-Zeitstempel zurück.

Kontingente für Amazon Keyspaces (für Apache Cassandra)

In diesem Abschnitt werden aktuelle Kontingente und Standardwerte für Amazon Keyspaces (für Apache Cassandra) beschrieben.

Topics

- Amazon Keyspaces-Servicekontingente
- Erhöhen oder Verringern des Durchsatzes (für Tabellen im Modus bereitgestellter Kapazität)
- Amazon Keyspaces-Verschlüsselung im Ruhezustand
- Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces

Amazon Keyspaces-Servicekontingente

Die folgende Tabelle enthält Amazon Keyspaces-Kontingente (für Apache Cassandra) und die Standardwerte. Informationen darüber, welche Kontingente angepasst werden können, finden Sie in der Konsole Service Quotas, wo Sie auch Kontingenterhöhungen beantragen können. Weitere Informationen zu Kontingenten erhalten Sie von AWS Support.

Kontingent	Description	Amazon Keyspaces Standard
Max. Anzahl an Schlüssel räumen pro AWS-Region	Die maximale Anzahl von Schlüsselräumen für diesen Abonnenten pro Region. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	256
Max. Anzahl an Tabellen pro AWS-Region	Die maximale Anzahl von Tabellen in allen Schlüssel bereichen für diesen Abonnenten pro Region. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	256

Kontingent	Description	Amazon Keyspaces Standard
Maximale Datenmenge, die mithilfe von `ADD REGION` Vorgängen wiederhergestellt wurde	Die maximale Datengröße, die gleichzeitig durch `ADD REGION` Operationen wiederhergestellt werden kann. Um die Menge der gleichzeitig wiederherzustellenden Daten zu erhöhen, wenden Sie sich an. Support	10 TB
Max. Größe des Tabellenschemas	Die maximale Größe eines Tabellenschemas.	350 KB
Max. Anzahl gleichzeitiger DDL-Operationen	Die maximale Anzahl gleichzeitiger DDL-Operationen, die für diesen Abonnenten pro Region zulässig sind.	50
Max. Anzahl Abfragen pro Verbindung	Die maximale Anzahl von CQL-Abfragen, die von einer einzelnen Client-TCP-Verbindung pro Sekunde verarbeitet werden können.	3000
Maximale Zeilengröße	Die maximale Größe einer Zeile, ausgenommen statische Spaltendaten. Details hierzu finden Sie unter the section called “Schätzen Sie die Zeilengröße”.	1 MB

Kontingent	Description	Amazon Keyspaces Standard
Maximale Anzahl von Spalten in `INSERT` und `UPDATE` - Anweisungen	Die maximal zulässige Anzahl von Spalten in CQL `INSERT` - oder `UPDATE` Anweisungen. Eine `INSERT UPDATE` OR-Anweisung unterstützt bis zu 225 reguläre Spalten, wenn Time to Live (TTL) ausgeschaltet ist. Wenn TTL aktiviert ist, können bis zu 166 reguläre Spalten in einem einzigen Vorgang geändert werden.	225/166
Maximale statische Daten pro logischer Partition	Die maximale Aggregatgröße statischer Daten in einer logischen Partition. Details hierzu finden Sie unter the section called “Berechnen Sie die statische Spaltengröße pro logischer Partition”.	1 MB
Max. Anzahl von Unterabfragen pro Anweisung `IN SELECT`	Die maximale Anzahl von Unterabfragen, die Sie für das `IN` Schlüsselwort in einer `SELECT` Anweisung verwenden können. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	100

Kontingent	Description	Amazon Keyspaces Standard
Maximale Anzahl verschachtelter eingefrorener Sammlungen pro AWS-Region	Die maximale Anzahl verschachtelter Sammlungen, die unterstützt werden, wenn Sie das FROZEN Schlüsselwort für eine Spalte mit einem Sammlungsdatentyp verwenden. Weitere Informationen zu eingefrorenen Sammlungen finden Sie unterthe section called “Sammlungstypen”. Um die Verschachtelungsebene zu erhöhen, wenden Sie sich an Support.	8
Maximale Anzahl von Anweisungen in einem protokollierten Stapel	Die maximale Anzahl unterstützter CQL-Anweisungen, wenn Sie eine protokollierte Batch-Anweisung verwenden. Weitere Informationen zu Batch-Anweisungen finden Sie unter. the section called “Verwenden Sie Batch-Anweisungen”	100
Maximale Anzahl von Anweisungen in einem protokollierten Batch unter Verwendung statischer Spalten	Die maximale Anzahl unterstützter CQL-Anweisungen, wenn Sie statische Spalten in einer protokollierten Batch-Anweisung verwenden. Weitere Informationen zu Batch-Anweisungen finden Sie unter. the section called “Verwenden Sie Batch-Anweisungen”	50

Kontingent	Description	Amazon Keyspaces Standard
Maximale Größe der protokollierten Batch-Anweisung	Die maximale Größe einer protokollierten Batch-Anweisung. Weitere Hinweise zu Batch-Anweisungen finden Sie unterthe section called “Verwenden Sie Batch-Anweisungen”.	4 MB
Maximale Anzahl von Anweisungen in einem nicht protokollierten Stapel	Die maximale Anzahl unterstützter CQL-Anweisungen, wenn Sie eine Batch-Anweisung ohne Protokollierung verwenden. Weitere Informationen zu Batch-Anweisungen finden Sie unter. the section called “Verwenden Sie Batch-Anweisungen”	30
Maximaler Lesedurchsatz pro Sekunde	Der maximale Lesedurchsatz pro Sekunde — Read Request Units (RRUs) oder Read Capacity Units (RCUs) —, der einer Tabelle pro Region zugewiesen werden kann. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	40 000

Kontingent	Description	Amazon Keyspaces Standard
Maximaler Schreibdurchsatz pro Sekunde	Der maximale Schreibdurchsatz pro Sekunde — Schreibanforderungseinheiten (WRUs) oder Schreibkapazitätseinheiten (WCUs) —, der einer Tabelle pro Region zugewiesen werden kann. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	40 000
Maximaler Schreibdurchsatz für eine Tabelle mithilfe von Change-Data-Capture (CDC)	Die maximale Schreibkapazität für eine Amazon Keyspaces-Tabelle mit aktiviertem CDC. Dieses Standardkontingent gilt nur für Tabellen im Modus „Bereitgestellte Kapazität read/write “. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	40 000
Account-level Lesedurchsatz (bereitgestellt)	Die maximale Anzahl aggregierter Lesekapazitätseinheiten (RCUs), die dem Konto pro Region zugewiesen wurden. Dies gilt nur für Tabellen im Modus mit bereitgestellter read/write Kapazität. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	80 000

Kontingent	Description	Amazon Keyspaces Standard
Account-level Schreibdurchsatz (bereitgestellt)	Die maximale Anzahl von aggregierten Schreibkapazitätseinheiten (WCU), die dem Konto pro Region zugewiesen wurden. Dies gilt nur für Tabellen im Modus mit bereitgestellter read/write Kapazität. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	80 000
Maximale Anzahl skalierbarer Ziele pro Region und Konto	Die maximale Anzahl skalierbarer Ziele für das Konto pro Region. Eine Amazon Keyspaces-Tabelle gilt als ein skalierbares Ziel, wenn Auto Scaling für Lesekapazität aktiviert ist, und als weiteres skalierbares Ziel, wenn Auto Scaling für Schreibkapazität aktiviert ist. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole für Application Auto Scaling anpassen, indem Sie Skalierbare Ziele für Amazon Keyspaces auswählen.	1.500

Kontingent	Description	Amazon Keyspaces Standard
Maximale Partionsschlüssel-Größe	Die maximale Größe des zusammengesetzten Partitionsschlüssels. Bis zu 3 Byte zusätzlicher Speicher werden für Metadaten zur Rohgröße jeder Cluster-Spalte hinzugefügt, die im Partitionsschlüssel enthalten sind.	2048 Bytes
Maximale Clustering-Schlüssel-Größe	Die maximale kombinierte Größe aller Cluster-Spalten. Bis zu 4 Byte zusätzlicher Speicher werden zur Rohgröße jeder Cluster-Spalte für Metadaten hinzugefügt.	850 Byte
Max. Anzahl gleichzeitiger Tabellenwiederherstellungen mithilfe von Point-in-time Recovery (PITR)	Die maximale Anzahl gleichzeitiger Tabellenwiederherstellungen mit PITR pro Abonnent beträgt 4. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	4
Maximale Datenmenge, die mithilfe von Point-in-Time Recovery (PITR) wiederhergestellt wurde	Die maximale Größe der Daten, die mit PITR innerhalb von 24 Stunden wiederhergestellt werden können. Sie können diesen Standardwert in der Service Quotas Quotas-Konsole anpassen.	5 TB

Erhöhen oder Verringern des Durchsatzes (für Tabellen im Modus bereitgestellter Kapazität)

Erhöhen des bereitgestellten Durchsatzes

Sie können den Wert erhöhen `ReadCapacityUnits` oder `WriteCapacityUnits` so oft wie nötig, indem Sie die Konsole oder die `ALTER TABLE` Anweisung verwenden. Die neuen Einstellungen werden erst wirksam, wenn die `ALTER TABLE`-Operation abgeschlossen ist.

Sie können Ihre Kontingente pro Konto nicht überschreiten, wenn Sie die bereitgestellte Kapazität hinzufügen. Und Sie können die bereitgestellte Kapazität für Ihre Tabellen beliebig erhöhen. Weitere Informationen zu diesen Kontingenten auf Kontoebene finden Sie im vorhergehenden Abschnitt, the section called “Amazon Keyspaces-Servicekontingente”.

Senken des bereitgestellten Durchsatzes

Für jede Tabelle in einer `ALTER TABLE` Abrechnung können Sie den Wert `ReadCapacityUnits` oder `WriteCapacityUnits` (oder beides) verringern. Die neuen Einstellungen werden erst wirksam, wenn die `ALTER TABLE`-Operation abgeschlossen ist.

Eine Reduzierung ist jederzeit und bis zu viermal täglich möglich. Tage sind entsprechend der Universal Coordinated Time (UTC) definiert. Außerdem ist eine zusätzliche Verringerung zulässig, wenn in der vorangehenden Stunde keine Verringerung erfolgt ist. Dadurch wird die maximale Anzahl an Rückgängen an einem Tag auf 27 erhöht (4 Abnahmen in der ersten Stunde und 1 Rückgang für jedes der folgenden einstündigen Zeitfenster an einem Tag).

Amazon Keyspaces-Verschlüsselung im Ruhezustand

Sie können die Verschlüsselungsoptionen zwischen einem AWS eigenen AWS KMS Schlüssel und einem vom Kunden verwalteten AWS KMS Schlüssel innerhalb eines 24-Stunden-Fensters bis zu viermal pro Tabelle ändern, und zwar ab dem Zeitpunkt, an dem die Tabelle erstellt wurde. Wenn in den vorhergehenden 6 Stunden kein Wechsel erfolgt ist, ist ein zusätzlicher Wechsel erlaubt. Dadurch wird die maximale Anzahl von Änderungen an einem Tag auf acht erhöht (vier Änderungen in den ersten sechs Stunden und eine Änderung für jedes der nachfolgenden Sechsstundenfenster an einem Tag).

Sie können die Verschlüsselungsoption so ändern, dass ein AWS eigener AWS KMS Schlüssel so oft wie nötig verwendet wird, auch wenn das vorherige Kontingent ausgeschöpft ist.

Es gelten die folgenden Kontingente, sofern Sie keine höheren Werte beantragen. Informationen zur Beantragung einer Erhöhung des Servicekontingents finden Sie unter Support.

Kontingente und Standardwerte für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces

Amazon Keyspaces UDT-Kontingente und Standardwerte

Die folgende Tabelle enthält Kontingente und Standardwerte für UDTs in Amazon Keyspaces. Informationen darüber, welche Kontingente angepasst werden können, finden Sie in der Konsole Service Quotas, wo Sie auch Kontingenterhöhungen beantragen können. Weitere Informationen zu diesen Kontingenten erhalten Sie von AWS Support.

Kontingent	Description	Amazon Keyspaces Standard
Maximale Anzahl von UDTs pro AWS-Region	Die maximale Anzahl von UDTs in allen Schlüssel bereichen für diesen Abonnenten pro Region.	256
Maximale Anzahl von Tabellen pro UDT	Die maximale Anzahl von Tabellen, die auf dieselbe UDT verweisen können.	100
Maximale Anzahl von UDTs pro Tabelle	Die maximale Anzahl von UDTs, auf die eine Tabelle verweisen kann.	50
Maximaler Grad der Verschachtelung von UDTs	Die maximale Verschach telungstiefe, die für UDTs unterstützt wird.	8
Maximale Anzahl direkt untergeordneter UDTs pro UDT	Die maximale Anzahl untergeordneter UDTs, die für eine UDT unterstützt werden.	10

Kontingent	Description	Amazon Keyspaces Standard
Maximale Anzahl direkt übergeordneter UDTs pro UDT	Die maximale Anzahl übergeordneter UDTs, die für eine UDT unterstützt werden.	10
Max. UDT-Schemagröße	Die maximale Größe des Schemas für eine UDT.	25 KB
Maximale Länge des UDT-Namens	Die maximale Anzahl von Zeichen im UDT-Namen.	48
Max. Länge des UDT-Feldnamens	Die maximale Anzahl von Zeichen in einem UDT-Feldnamen.	128

Dokumentenverlauf für Amazon Keyspaces (für Apache Cassandra)

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation seit der letzten Version von Amazon Keyspaces (für Apache Cassandra) beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- Letzte Aktualisierung der Dokumentation: 28. Januar 2026

Änderung	Beschreibung	Datum
GetRecords Die API gibt jetzt eine Beschreibung des Iterators zurück	Die `GetRecords` API-Antwort enthält jetzt ein `iteratorDescription` Feld, das einen `iteratorPosition` Wert enthält, der die Position des aktuellen Iterators innerhalb des Shards angibt, sodass Sie die Abfragehäufigkeit optimieren können.	2. Juni 2026
Sie können jetzt neue und bestehende Tabellen in Amazon Keyspaces vorwärmen	Sie können jetzt `warm-throughput` für neue und bestehende Tabellen in Amazon Keyspaces konfigurieren, um sie für erwartete Spitzenkapazitätsspitzen vorzuwärmen.	28. Januar 2026
Amazon Keyspaces unterstützt jetzt ein integriertes Erlebnis AWS CloudShell	Sie können jetzt eine vorauthentifizierte Shell direkt von der Amazon Keyspaces-Konsole aus starten.	26. Januar 2026
Amazon Keyspaces unterstützt jetzt protokollierte Batches	Sie können jetzt mehrere Kontoauszüge zu protokoll	11. November 2025

	ierten Batches in Amazon Keyspaces zusammenfassen.	
Änderungen an Amazon Keyspaces TLS-Zertifikaten	Die Anweisungen wurden aktualisiert, um Zertifikate aufzunehmen, die im Rahmen der Amazon Trust Services (ATS) -Kette ausgestellt wurden.	25. Oktober 2025
Amazon Keyspaces unterstützt jetzt IPv4- und IPv6-Anfragen	Die Dual-Stack-Endgeräte von Amazon Keyspaces unterstützen IPv4- und IPv6-Konfigurationen.	30. September 2025
`NOW`und `UUID` werden jetzt in allen CQL-Anweisungen in Amazon Keyspaces unterstützt	Sie können jetzt die `UUID` Funktionen `NOW` und in allen CQL-Anweisungen in Amazon Keyspaces verwenden.	8. September 2025
Der Amazon Keyspaces Streams Kinesis Adapter ist jetzt verfügbar	Sie können jetzt Amazon Keyspaces CDC-Streams mit der Kinesis Client Library (KCL) verarbeiten.	2. Juli 2025

Neue verwaltete Richtlinie von Amazon Keyspaces	Amazon Keyspaces hat eine neue verwaltete Richtlinie hinzugefügt `KeyspacesCDCServiceRolePolicy` , die der serviceverknüpften Rolle die erforderlichen Berechtigungen erteilt, `AWSServiceRoleForAmazonKeyspacesCDC` um Amazon Keyspaces CDC-Stream-Metrikdaten in Ihrem Namen zu CloudWatch veröffentlichen. Weitere Informationen finden Sie unter Rollen für Amazon Keyspaces CDC-Streams verwenden.	2. Juli 2025

Aktualisierung der verwalteten Richtlinien von Amazon Keyspaces	Amazon Keyspaces hat den `AmazonKeyspacesReadOnlyAccess_v2` und den `AmazonKeyspacesReadOnlyAccess` verwalteten Richtlinien neue Berechtigungen hinzugefügt, um es IAM-Prinzipalen zu ermöglichen, Streams anzusehen. Amazon Keyspaces `AmazonKeyspacesFullAccess` hat die verwaltete Richtlinie mit Berechtigungen zum Erstellen einer serviceverknüpften Rolle aktualisiert `AWSServiceRoleForAmazonKeyspacesCDC`, die CloudWatch Metriken im Namen der Principals veröffentlicht.	2. Juli 2025
Amazon Keyspaces Change Data Capture (CDC) -Streams sind jetzt verfügbar	Aktivieren Sie CDC-Streams, um Änderungen in Ihren Amazon Keyspaces-Tabellen aufzuzeichnen.	2. Juli 2025
Die multiregionale Replikation von Amazon Keyspaces unterstützt jetzt die Region Afrika (Kapstadt).	Sie können jetzt die Region Afrika (Kapstadt) für Schlüsselräume mit mehreren Regionen definieren.	31. März 2025
Amazon Keyspaces Multi-Region-Replikation unterstützt jetzt unbegrenzt. AWS-Regionen	Sie können jetzt beliebig viele Regionen für Schlüsselräume mit mehreren Regionen definieren.	25. März 2025

Amazon Keyspaces-Unterstützung für benutzerdefinierte Typen (UDTs) in Schlüsselräumen mit mehreren Regionen	Sie können jetzt UDTs in Schlüsselräumen mit mehreren Regionen erstellen und verwenden.	24. März 2025
Aktualisierung der verwalteten Richtlinien von Amazon Keyspaces	Amazon Keyspaces hat der `AmazonKeyspacesFullAccess` verwalteten Richtlinie neue Berechtigungen hinzugefügt, damit IAM-Prinzipale einem vorhandenen Schlüsselraum neue Regionen hinzufügen können. Dies beinhaltet eine Aktualisierung der serviceverknüpften Rolle. AWSServiceRoleForAmazonKeyspacesReplication	19. November 2024
Fügen Sie Replikate zu Tabellen mit mehreren Regionen in Amazon Keyspaces hinzu	Sie können jetzt neue AWS-Regionen zu bestehenden Schlüsselräumen mit einer oder mehreren Regionen hinzufügen.	19. November 2024
Support für benutzerdefinierte Typen (UDTs) in Amazon Keyspaces	Mit der nativen Unterstützung für UDTs in Amazon Keyspaces können Sie jetzt Datenstrukturen in Ihren Anwendungen definieren, die reale Datenhierarchien repräsentieren.	30. Oktober 2024
`ADD COLUMN`Unterstützung für Amazon Keyspaces-Replikation in mehreren Regionen	Amazon Keyspaces unterstützt jetzt Schemaänderungen für Tabellen mit mehreren Regionen.	17. September 2024

Aktualisierte Migrationsanleitung für Amazon Keyspaces	In den aktualisierten Migrationsleitlinien wird beschrieben, wie Sie einen Migrationsplan für eine erfolgreiche Migration von Apache Cassandra zu Amazon Keyspaces erstellen, einschließlich verschiedener Strategien für Offline- und Online-Migrationen.	28. Juni 2024
Stellen Sie über Amazon Elastic Kubernetes Service eine Connect zu Amazon Keyspaces her	Sie können jetzt einer schrittweisen Anleitung folgen, um von Amazon EKS aus eine Verbindung zu Amazon Keyspaces herzustellen.	7. Februar 2024
Amazon Keyspaces Unterstützung für die Replikation in mehreren Regionen für bereitgestellte Tabellen	Amazon Keyspaces unterstützt jetzt den Modus „Bereitgestellte Kapazität“ für Tabellen mit mehreren Regionen.	23. Januar 2024
auto Skalierungs-APIs von Amazon Keyspaces für bereitgestellte Tabellen	Amazon Keyspaces bietet CQL jetzt AWS API-Unterstützung für die Einrichtung von Auto Scaling mit bereitgestelltem Kapazitätsmodus.	23. Januar 2024
Amazon Keyspaces DML-Aktivität in Protokollen enthalten CloudTrail	Sie können jetzt API-Aufrufe der Amazon Keyspaces Data Manipulation Language (DML) prüfen. AWS CloudTrail	20. Dezember 2023
Amazon Keyspaces-Unterstützung für das FROZEN Keyword	Amazon Keyspaces unterstützt jetzt das FROZEN Schlüsselwort für Sammlungsdatentypen.	15. November 2023

Aktualisierung der verwalteten Richtlinien von Amazon Keyspaces	Amazon Keyspaces hat der `AmazonKeyspacesFullAccess` verwalteten Richtlinie neue Berechtigungen hinzugefügt, um Clients, die über VPC-Schnittstellen-Endpunkte eine Verbindung zu Amazon Keyspaces herstellen, Zugriff auf die Amazon EC2 EC2-Instance zu ermöglichen, um die Amazon `system.peers` Keyspaces-Tabelle mit Netzwerkinformationen aus der VPC zu aktualisieren.	3. Oktober 2023
Aktualisierung der verwalteten Richtlinien von Amazon Keyspaces	Amazon Keyspaces hat eine neue `AmazonKeyspacesReadOnlyAccess_v2` verwaltete Richtlinie erstellt, um Clients, die über VPC-Schnittstellen-Endpunkte eine Verbindung zu Amazon Keyspaces herstellen, den Zugriff auf die Amazon EC2 EC2-Instance zu ermöglichen, um die Amazon `system.peers` Keyspaces-Tabelle mit Netzwerkinformationen aus der VPC zu aktualisieren.	12. September 2023
Bewährte Methoden für das Erstellen von Verbindungen in Amazon Keyspaces	Erfahren Sie, wie Sie die Client-Treiberkonfigurationen in Amazon Keyspaces verbessern und optimieren können.	30. Juni 2023

Systemschlüsselräume sind jetzt für Amazon Keyspaces dokumentiert	Erfahren Sie, was in Systemschlüsselräumen gespeichert ist und wie Sie sie nach nützlichen Informationen in Amazon Keyspaces abfragen können.	21. Juni 2023
Amazon Keyspaces unterstützt jetzt die Replikation in mehreren Regionen	Die multiregionale Replikation von Amazon Keyspaces unterstützt Sie bei der Wartung global verteilter Anwendungen, indem sie Ihnen eine verbesserte Fehlertoleranz, Stabilität und Belastbarkeit bietet.	5. Juni 2023
Aktualisierung der verwalteten Richtlinien von Amazon Keyspaces	Amazon Keyspaces hat der `AmazonKeyspacesFullAccess` verwalteten Richtlinie neue Berechtigungen hinzugefügt, damit Amazon Keyspaces eine servicebezogene Rolle erstellen kann, wenn ein Administrator einen Schlüsselraum für mehrere Regionen erstellt.	5. Juni 2023
Amazon Keyspaces-Unterstützung für das `IN` Keyword	Amazon Keyspaces unterstützt jetzt das `IN` Schlüsselwort in `SELECT` Kontoauszügen.	25. April 2023
Cross-account Zugriff für Amazon Keyspaces und Schnittstellen-VPC-Endpunkte	Erfahren Sie, wie Sie den kontoübergreifenden Zugriff für Amazon Keyspaces mit VPC-Endpunkten implementieren.	20. April 2023

Amazon Keyspaces-Unterstützung für clientseitige Zeitstempel	Clientseitige Zeitstempel von Amazon Keyspace sind Zeitstempel auf Cassandra -compatible Zellenebene, die verteilten Anwendungen helfen, die Reihenfolge der Schreibvorgänge zu bestimmen, wenn verschiedene Clients Änderungen an denselben Daten vornehmen.	14. März 2023
Erste Schritte mit Amazon Keyspaces und Schnittstellen-VPC-Endpunkten	In dieser schrittweisen Anleitung erfahren Sie, wie Sie von einer VPC aus eine Verbindung zu Amazon Keyspaces herstellen.	1. März 2023
Optimierung der Kosten von Amazon Keyspaces-Tabellen	Es stehen bewährte Methoden und Anleitungen zur Verfügung, mit denen Sie Strategien zur Kostenoptimierung Ihrer vorhandenen Amazon Keyspaces-Tabellen ermitteln können.	17. Februar 2023
Das `Murmur3Partitioner` ist jetzt die Standardeinstellung	Das `Murmur3Partitioner` ist jetzt die Standardpartitionierung in Amazon Keyspaces.	17. November 2022
Amazon Keyspaces unterstützt jetzt `Murmur3Partitioner`	Das `Murmur3Partitioner` ist jetzt in Amazon Keyspaces verfügbar.	9. November 2022
Unterstützungsupdate für leere Zeichenketten und Blob-Werte	Amazon Keyspaces unterstützt jetzt auch leere Zeichenketten und Blob-Werte als Cluster-Spaltenwerte.	19. Oktober 2022

Amazon Keyspaces ist jetzt verfügbar in AWS GovCloud (US)	Amazon Keyspaces ist jetzt in der verfügbar AWS GovCloud (US) Region und befindet sich im Geltungsbereich der FedRAMP-High Konformität. Informationen zu verfügbaren Endpunkten finden Sie unter AWS GovCloud (US) Region FIPS-Endgeräte.	04. August 2022
Überwachen Sie die Speicherkosten für Amazon Keyspaces -Tabellen mit Amazon CloudWatch	Amazon Keyspaces hilft Ihnen jetzt dabei, die Speicherkosten für Tabellen mithilfe der `BillableTableSizeInBytes` CloudWatch Metrik im Laufe der Zeit zu überwachen und nachzuverfolgen.	14. Juni 2022
Amazon Keyspaces unterstützt jetzt Terraform	Sie können jetzt Terraform verwenden, um DDL-Operationen (Data Definition Language) in Amazon Keyspaces durchzuführen.	9. Juni 2022
Unterstützung für Amazon `token` Keyspaces-Funktionen	Amazon Keyspaces hilft Ihnen jetzt, Anwendungsabfragen mithilfe der `token` Funktion zu optimieren.	19. April 2022
Amazon Keyspaces-Integration mit Apache Spark	Amazon Keyspaces hilft Ihnen jetzt, Daten in Apache Spark einfacher zu lesen und zu schreiben, indem Sie den Open-Source-Spark Cassandra Connector verwenden.	19. April 2022

Amazon Keyspaces API-Referenz	Amazon Keyspaces unterstützt Operationen auf Steuerungsebene zur Verwaltung von Schlüsselräumen und Tabellen mithilfe des AWS SDK und. AWS CLI Das API-Referenzhandbuch beschreibt die unterstützten Operationen auf der Kontrollebene ausführlich.	2. März 2022
So beheben Sie häufig auftretende Konfigurationsprobleme bei der Verwendung von Amazon Keyspaces.	Erfahren Sie mehr darüber, wie Sie häufig auftretende Konfigurationsprobleme lösen können, die bei der Verwendung von Amazon Keyspaces auftreten können.	22. November 2021
Amazon Keyspaces-Unterstützung für Time to Live (TTL).	Amazon Keyspaces Time to Live (TTL) hilft Ihnen dabei, Ihre Anwendungslogik zu vereinfachen und den Speicherpreis zu optimieren, indem Daten aus Tabellen automatisch ablaufen.	18. Oktober 2021
Migrieren Sie Daten mit DSBulk zu Amazon Keyspaces.	Step-by-step Tutorial für die Migration von Daten von Apache Cassandra zu Amazon Keyspaces mit dem DataStax Bulk Loader (DSBulk).	9. August 2021

Amazon Keyspaces-Unterstützung für VPC-Endpunkteinträge in der `system.peers` Tabelle.	Amazon Keyspaces ermöglicht es Ihnen, die `system.peers` Tabelle mit verfügbaren VPC-Endpunktinformationen der Schnittstelle zu füllen, um den Lastenausgleich zu verbessern und den Durchsatz zu erhöhen. read/write	29. Juli 2021
Aktualisierung der von IAM verwalteten Richtlinien zur Unterstützung von kundenverwalteten Schlüsseln. AWS KMS	IAM-verwaltete Richtlinien für Amazon Keyspaces beinhalten jetzt Berechtigungen zum Auflisten und Anzeigen verfügbarer, von Kunden verwalteter AWS KMS Schlüssel, die in gespeichert sind. AWS KMS	1. Juni 2021
Amazon Keyspaces-Unterstützung für vom Kunden verwaltete AWS KMS Schlüssel.	Mit Amazon Keyspaces können Sie die Kontrolle über vom Kunden verwaltete AWS KMS Schlüssel übernehmen, die AWS KMS zur Verschlüsselung im Ruhezustand gespeichert sind.	1. Juni 2021
Amazon Keyspaces-Unterstützung für JSON-Syntax	Amazon Keyspaces hilft Ihnen dabei, JSON-Dokumente einfacher zu lesen und zu schreiben, indem es die JSON-Syntax für INSERT- und SELECT-Operationen unterstützt.	21. Januar 2021

Amazon Keyspaces-Unterstützung für statische Spalten	Amazon Keyspaces hilft Ihnen jetzt dabei, gemeinsame Daten zwischen mehreren Zeilen effizient zu aktualisieren und zu speichern, indem statische Spalten verwendet werden.	9. November 2020
GA-Version der NoSQL Workbench-Unterstützung für Amazon Keyspaces	NoSQL Workbench ist eine clientseitige Anwendung, mit der Sie nichtrelationale Datenmodelle für Amazon Keyspaces einfacher entwerfen und visualisieren können. NoSQL Workbench-Clients sind für Windows, macOS und Linux verfügbar.	28. Oktober 2020
Vorabversion der NoSQL Workbench-Unterstützung für Amazon Keyspaces	NoSQL Workbench ist eine clientseitige Anwendung, mit der Sie nichtrelationale Datenmodelle für Amazon Keyspaces einfacher entwerfen und visualisieren können. NoSQL Workbench-Clients sind für Windows, macOS und Linux verfügbar.	5. Oktober 2020
Neue Codebeispiele für den programmatischen Zugriff auf Amazon Keyspaces	Wir fügen weiterhin Codebeispiele für den programmatischen Zugriff auf Amazon Keyspaces hinzu. Beispiele sind jetzt für Java-, Python-, Go-, C#- und Perl-Cassandra-Treiber verfügbar, die Apache Cassandra Version 3.11.2 unterstützen.	17. Juli 2020

Amazon Keyspaces Point-in-Time-Wiederherstellung (PITR)	Amazon Keyspaces bietet jetzt Point-in-Time-Recovery (PITR), um Ihre Tabellen vor versehentlichen Schreib- oder Löschvorgängen zu schützen, indem es Ihnen kontinuierliche Backups Ihrer Tabellendaten bietet.	9. Juli 2020
Allgemeine Verfügbarkeit von Amazon Keyspaces	Mit Amazon Keyspaces, das in der Vorschauversion als Amazon Managed Apache Cassandra Service (MCS) bekannt war, können Sie den Cassandra Query Language (CQL) -Code, Apache 2.0-lizenzierte Cassandra-Treiber und Entwicklertools verwenden, die Sie bereits heute verwenden.	23. April 2020
Automatische Skalierung von Amazon Keyspaces	Amazon Keyspaces (für Apache Cassandra) ist in Application Auto Scaling integriert, sodass Sie Durchsatzkapazität für variable Workloads als Reaktion auf den tatsächlichen Anwendungsverkehr effizient bereitstellen können, indem die Durchsatzkapazität automatisch angepasst wird.	23. April 2020

Schnittstelle für Virtual Private Cloud (VPC) -Endpunkte für Amazon Keyspaces	Amazon Keyspaces bietet private Kommunikation zwischen dem Service und Ihrer VPC, sodass der Netzwerkverkehr das Amazon-Netzwerk nicht verlässt.	16. April 2020
Tag-based Zugriffsrichtlinien	Sie können jetzt Ressource n-Tags in IAM-Richtlinien verwenden, um den Zugriff auf Amazon Keyspaces zu verwalten.	8. April 2020
Datentyp des Zählers	Amazon Keyspaces hilft Ihnen jetzt dabei, Inkremente und Dekremente von Spaltenwe rten mithilfe von Zählern zu koordinieren.	7. April 2020
Taggen von Ressourcen	Amazon Keyspaces ermöglich t es Ihnen jetzt, Ressourcen mithilfe von Tags zu kennzeich nen und zu kategorisieren.	31. März 2020
CloudFormation Unterstützung	Amazon Keyspaces hilft Ihnen jetzt dabei, die Erstellung und Verwaltung von Ressourcen zu automatisieren, indem Sie CloudFormation	25. März 2020

Support für IAM-Rollen und -Richtlinien sowie SigV4-Authentifizierung	Es wurden Informationen hinzugefügt, wie Sie AWS Identity and Access Management (IAM) zur Verwaltung von Zugriffsberechtigungen und zur Implementierung von Sicherheitsrichtlinien für Amazon Keyspaces verwenden können und wie Sie das Authentifizierungs-Plugin für den DataStax Java-Treiber für Cassandra verwenden können, um mithilfe von IAM-Rollen und föderierten Identitäten programmgesteuert auf Amazon Keyspaces zuzugreifen.	17. März 2020
Read/write Kapazitätsmodus	Amazon Keyspaces unterstützt jetzt zwei read/write Durchsatzkapazitätsmodi. Der read/write Kapazitätsmodus steuert, wie Ihnen der Lese- und Schreibdurchsatz in Rechnung gestellt wird und wie die Tabellendurchsatzkapazität verwaltet wird.	20. Februar 2020
Erstversion	Diese Dokumentation behandelt die erste Version von Amazon Keyspaces (für Apache Cassandra).	3. Dezember 2019

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.