aws

Guía para desarrolladores

Amazon Keyspaces (para Apache Cassandra)

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Keyspaces (para Apache Cassandra): Guía para desarrolladores

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Las marcas comerciales y la imagen comercial de Amazon no se pueden utilizar en relación con ningún producto o servicio que no sea de Amazon, de ninguna manera que pueda causar confusión entre los clientes y que menosprecie o desacredite a Amazon. Todas las demás marcas registradas que no son propiedad de Amazon son propiedad de sus respectivos propietarios, que pueden o no estar afiliados, conectados o patrocinados por Amazon.

Table of Contents

¿Qué es Amazon Keyspaces? 1
Cómo funciona 2
Arquitectura de alto nivel 2
Modelo de datos de Cassandra 5
Acceso a Amazon Keyspaces 6
Casos de uso 7
¿Qué es CQL? 7
Comparación de Amazon Keyspaces con Cassandra 9
Diferencias funcionales con Apache Cassandra 9
API, operaciones y tipos de datos de Apache Cassandra 10
Creación y eliminación asíncronas de espacios de claves y tablas 10
Autenticación y autorización 10
Lote 11
Configuración del clúster 11
Conexiones 11
Palabra clave `IN` 12
Ajuste de rendimiento de las consultas CQL 12
Colecciones `FROZEN` 13
Transacciones ligeras 13
Equilibrio de carga 13
Paginación 13
Particionadores 14
Instrucciones preparadas 14
Eliminación de rangos 14
Tablas del sistema 15
Marcas temporales 15
API, operaciones, funciones y tipos de datos de Cassandra admitidos 15
Compatibilidad con la API de Cassandra 16
Compatibilidad con la API de plano de control de Cassandra 17
Compatibilidad con la API de plano de datos de Cassandra 18
Compatibilidad con las funciones de Cassandra 18
Compatibilidad con tipos de datos de Cassandra 19
Niveles de coherencia de Cassandra admitidos 21
Niveles de coherencia de escritura 21

Niveles de coherencia de lectura 22
Niveles de coherencia no admitidos 23
Acceso a Amazon Keyspaces 24
Con AWS Identity and Access Management figuración 24
Inscríbase para obtener una Cuenta de AWS 24
Cómo crear un usuario administrativo 24
Configuración de Amazon Keyspaces 26
Mediante la consola 26
Usando AWS CloudShell 27
Obtener permisos de IAM para AWS CloudShell 27
Interactuar con Amazon Keyspaces mediante AWS CloudShell 28
Conexión mediante programación 30
Creación de credenciales 31
Puntos de conexión de servicio 42
Uso de `cqlsh` 47
Uso de la AWS CLI 54
Uso de la API 59
Trabajando con los SDK AWS 59
Utilización de un controlador de cliente de Cassandra 60
Conexión desde Amazon EKS 89
Conexión con puntos finales de VPC 110
Requisitos previos 111
Paso 1: Lanzar una instancia de Amazon EC2 111
Paso 2: configurar la instancia de Amazon EC2 113
Paso 3: Crear un punto de conexión de VPC para Amazon Keyspaces 116
Paso 4: Configurar los permisos para la conexión del punto de conexión de VPC 121
Paso 5: Configurar el monitoreo 125
Paso 6: (Opcional) Prácticas recomendadas para las conexiones 125
Paso 7: (Opcional) Sanear 128
Acceso entre cuentas 129
Acceso entre cuentas en una VPC compartida 130
Acceso entre cuentas sin una VPC compartida 134
Introducción 136
Requisitos previos 136
Paso 1: Crear un espacio de claves y una tabla 136
Creación de un espacio de claves 137

Creación de una tabla ... 139
Paso 2: Operaciones CRUD ... 143
Creación ... 144
Leer ... 145
Actualización ... 148
Eliminación ... 149
Paso 3: Saneamiento (opcional) ... 151
Eliminación de una tabla ... 151
Eliminación de un espacio de claves ... 152
Migración a Amazon Keyspaces ... 154
Carga de datos con cqlsh ... 155
Requisitos previos ... 155
Paso 1: Crear origen y destino ... 156
Paso 2: Preparar los datos ... 158
Paso 3: Establecer la capacidad de rendimiento de la tabla ... 159
Paso 4: Configurar los ajustes de `cqlsh COPY FROM` ... 161
Paso 5: Ejecutar el comando `cqlsh COPY FROM` ... 164
Solución de problemas ... 165
Carga de datos con DSBulk ... 167
Requisitos previos ... 168
Paso 1: Crear origen y destino ... 171
Paso 2: Preparar los datos ... 172
Paso 3: Establecer la capacidad de rendimiento de la tabla ... 174
Paso 4: Configurar los ajustes de `DSBulk` ... 176
Paso 5: Ejecutar el comando `load` de DSBulk ... 178
Ejemplos de código ... 180
Acciones ... 185
Crear un espacio de claves ... 186
Creación de una tabla ... 190
Eliminar un espacio de claves ... 197
Eliminación de una tabla ... 200
Obtener datos sobre un espacio de claves ... 204
Obtener datos sobre una tabla ... 208
Enumerar espacios clave ... 213
Enumerar tablas en un espacio de claves ... 218
Restaurar una tabla a un momento dado ... 222

Actualizar una tabla ... 226
Escenarios ... 231
Introducción a los espacios de claves y las tablas ... 231
Bibliotecas y herramientas ... 293
Bibliotecas y ejemplos ... 293
Kit de herramientas para desarrolladores de Amazon Keyspaces (para Apache Cassandra) ... 293
Ejemplos de Amazon Keyspaces (para Apache Cassandra) ... 293
Complementos de autenticación AWS Signature Version 4 (SigV4) ... 294
Repositorios destacados de ejemplos y herramientas para desarrolladores ... 294
Búferes de protocolo de Amazon Keyspaces ... 294
Plantilla de AWS CloudFormation para crear el panel de control de métricas de Amazon CloudWatch para Amazon Keyspaces (para Apache Cassandra) ... 295
Uso de Amazon Keyspaces (para Apache Cassandra) con AWS Lambda ... 295
Uso de Amazon Keyspaces (para Apache Cassandra) con Spring ... 295
Uso de Amazon Keyspaces (para Apache Cassandra) con Scala ... 295
Uso de Amazon Keyspaces (para Apache Cassandra) con AWS Glue ... 296
Conversor de lenguaje de consulta de Cassandra (CQL) a AWS CloudFormation de Amazon Keyspaces (para Apache Cassandra) ... 296
Asistentes de Amazon Keyspaces (para Apache Cassandra) para el controlador de Apache Cassandra para Java ... 296
Demostración de compresión rápida de Amazon Keyspaces (para Apache Cassandra) ... 296
Demostración del códec de Amazon S3 y Amazon Keyspaces (para Apache Cassandra) ... 296
Integración con Apache Spark ... 297
Requisitos previos ... 298
Paso 1: Configurar Amazon Keyspaces ... 298
Paso 2: Configurar Apache Cassandra Spark Connector ... 300
Paso 3: Crear el archivo de configuración de la aplicación ... 302
Conexión con autenticación SigV4 ... 302
Conexión con credenciales específicas del servicio ... 303
Conexión con una tasa fija ... 304
Paso 4: Preparar los datos de origen y la tabla de destino ... 305
Paso 5: Escribir y leer datos en Amazon Keyspaces ... 306
Solución de problemas ... 309
Errores y advertencias comunes ... 311
Solución de problemas ... 312

Conexiones ... 312
Errores al conectarse a un punto de conexión de Amazon Keyspaces ... 312
Administración de la capacidad ... 324
Errores de capacidad sin servidor ... 324
Lenguaje de definición de datos ... 330
Errores de lenguaje de definición de datos ... 330
Administración de recursos sin servidor ... 336
Almacenamiento ... 336
Modos de capacidad de lectura/escritura ... 337
Modo de capacidad bajo demanda ... 337
Modo de capacidad de rendimiento aprovisionada ... 340
Administración y visualización de los modos de capacidad ... 342
Consideraciones al cambiar los modos de capacidad ... 343
Administre la capacidad de rendimiento con el escalado automático ... 343
Funcionamiento del escalado automático de Amazon Keyspaces ... 344
Cómo funciona el escalado automático para tablas multirregionales ... 346
Notas de uso ... 347
Uso de la consola ... 348
Uso de CQL ... 352
Uso de la CLI ... 359
Capacidad de ampliación ... 365
Calcule el consumo de capacidad ... 365
Consultas de rango ... 366
Escaneos de tablas ... 367
Transacciones ligeras ... 368
Calcula el consumo de capacidad de lectura y escritura con Amazon CloudWatch ... 368
Uso con Amazon Keyspaces ... 369
Uso de los espacios de claves ... 369
Espacios de claves de sistema ... 369
Creación de espacios de claves ... 376
Uso de tablas ... 377
Creación de tablas ... 377
Tablas multirregionales ... 378
Columnas estáticas ... 379
Uso de las filas ... 384
Cálculo del tamaño de las filas ... 384

Uso de consultas ... 388
Instrucción IN SELECT ... 388
Ordenación de resultados ... 392
Paginación de resultados ... 393
Uso de particionadores ... 394
Trabajar con etiquetas ... 396
Restricciones de etiquetado ... 397
Operaciones de etiquetado ... 398
Informes de asignación de costos para Amazon Keyspaces ... 403
Prácticas recomendadas ... 405
Diseño NoSQL ... 406
NoSQL versus RDBMS ... 407
Dos conceptos clave ... 407
Enfoque general ... 408
Conexiones ... 409
Cómo funcionan ... 410
Configuración de conexiones ... 411
Conexiones de puntos de conexión de VPC ... 412
Monitoreo de conexiones ... 413
Gestión de errores de conexión ... 414
Modelado de datos ... 415
Diseño de claves de partición ... 415
Optimización de costos ... 418
Evaluar los costos en el nivel de tabla ... 418
Evaluación del modo de capacidad de una tabla ... 420
Evaluación de los ajustes de Application Auto Scaling de su tabla ... 425
Identificación de recursos no utilizados ... 432
Evaluar los patrones de uso de las tablas ... 438
Evaluar la capacidad aprovisionada para lograr un aprovisionamiento del tamaño adecuado ... 439
NoSQL Workbench ... 450
Descargar ... 451
Introducción ... 451
Modelador de datos ... 453
Creación de un modelo de datos ... 454
Edición de un modelo de datos ... 456

Visualizador de datos ... 458
Visualización de un modelo de datos ... 458
Vista agregada ... 459
Confirmación de un modelo de datos ... 461
Antes de empezar ... 462
Conexión con credenciales específicas del servicio ... 463
Conexión con credenciales de IAM ... 466
Uso de una conexión guardada ... 469
Apache Cassandra ... 469
Modelos de datos de ejemplo ... 472
Modelo de datos de empleados ... 472
Modelo de datos de transacciones de tarjetas de crédito ... 472
Modelo de datos de operaciones de aerolíneas ... 473
Historial de versiones ... 473
Réplica multirregión ... 475
Ventajas ... 475
Modos de capacidad y precios ... 476
Cómo funciona ... 477
Cómo funciona ... 477
Resolución de conflictos ... 478
Recuperación de desastres ... 478
Permisos de IAM ... 479
Integración con PITR ... 480
Integración con servicios de AWS ... 481
Notas de uso ... 481
Formas de uso de la réplica multirregión ... 483
Uso de la consola ... 484
Uso de CQL ... 490
Uso del AWS CLI ... 497
Recuperación a un momento dado ... 507
Cómo funciona ... 507
Habilitación de PITR ... 508
Permisos de restauración ... 511
Copias de seguridad continuas ... 513
Ajustes de restauración ... 514
PITR y tablas cifradas ... 516

PITR y tablas multirregión ... 516
Tiempo de restauración de la tabla ... 517
Integración con servicios de AWS ... 481
Restauración de una tabla a un momento específico ... 518
Antes de empezar ... 518
Restauración de una tabla a un momento dado (consola) ... 518
Restauración de una tabla a un momento dado con la AWS CLI ... 520
Restauración de una tabla a un momento dado con CQL ... 522
Restauración de una tabla eliminada con la AWS CLI ... 525
Restauración de una tabla eliminada con CQL ... 525
Caducidad de datos con Periodo de vida ... 527
Cómo funciona ... 528
Valor de TTL predeterminado ... 528
Valores de TTL personalizados ... 529
Habilitación de TTL ... 530
Integración con servicios de AWS ... 530
Cómo utilizar Periodo de vida ... 531
Para crear una nueva tabla con la configuración de periodo de vida (TTL) predeterminada habilitada (consola) ... 531
Para actualizar la configuración predeterminada de periodo de vida (TTL) en tablas existentes (consola) ... 532
Para deshabilitar la configuración predeterminada de periodo de vida (TTL) en tablas existentes (consola) ... 533
Para crear una nueva tabla con la configuración de periodo de vida (TTL) predeterminada habilitada mediante CQL ... 533
Para utilizar `ALTER TABLE` para editar la configuración predeterminada de periodo de vida (TTL) mediante CQL ... 534
Cómo habilitar el periodo de vida (TTL) en tablas nuevas mediante propiedades personalizadas ... 534
Cómo habilitar el periodo de vida (TTL) en tablas existentes mediante propiedades personalizadas ... 534
Para utilizar `INSERT` para editar la configuración personalizada de periodo de vida (TTL) mediante CQL ... 535
Para utilizar `UPDATE` para editar la configuración personalizada de periodo de vida (TTL) mediante CQL ... 535
Marcas de tiempo del lado del cliente ... 537

Cómo funciona ... 538
Marcas de tiempo del lado del cliente en Amazon Keyspaces ... 538
Integración con servicios de AWS ... 539
Uso de marcas de tiempo del lado del cliente ... 539
Creación de una nueva tabla con marcas de tiempo del lado del cliente activadas (consola) ... 540
Activación de las marcas de tiempo del lado del cliente en las tablas existentes (consola) ... 541
Creación de una nueva tabla con las marcas de tiempo del lado del cliente activadas (CQL) ... 541
Activación de las marcas de tiempo del lado del cliente para tablas existentes mediante ALTER TABLE (CQL) ... 542
Creación de una nueva tabla con las marcas de tiempo del lado del cliente activadas (CLI) ... 542
Activación de las marcas de tiempo del cliente en una tabla existente (CLI) ... 544
Uso de marcas de tiempo del lado del cliente en instrucciones de lenguaje de manipulación de datos (DML) ... 546
Recursos de AWS CloudFormation ... 547
Espacios de claves y plantillas de AWS CloudFormation de Amazon Keyspaces ... 547
Obtener más información sobre AWS CloudFormation ... 547
Monitoreo de Amazon Keyspaces ... 549
Monitorización con CloudWatch ... 550
Uso de las métricas ... 551
Métricas y dimensiones ... 552
Creación de alarmas ... 575
Registro con CloudTrail ... 576
Configuración de las entradas del archivo de registro en CloudTrail ... 577
Información de DDL en CloudTrail ... 578
Información de DML en CloudTrail ... 578
Descripción de las entradas de los archivos de registro de ... 579
Seguridad ... 591
Protección de datos ... 592
Cifrado en reposo ... 593
Cifrado en tránsito ... 614
Privacidad del tráfico entre redes ... 615
AWS Identity and Access Management ... 616
Público ... 617

Autenticación con identidades ... 618
Administración de acceso mediante políticas ... 621
Funcionamiento de Amazon Keyspaces con IAM ... 624
Ejemplos de políticas basadas en identidades ... 629
Políticas administradas por AWS ... 636
Solución de problemas ... 645
Uso de roles vinculados a servicios ... 648
Validación de conformidad ... 656
Resiliencia ... 657
Seguridad de infraestructuras ... 658
Uso de los puntos de conexión de VPC de interfaz ... 659
Configuración y análisis de vulnerabilidades para Amazon Keyspaces ... 666
Prácticas recomendadas de seguridad ... 667
Prácticas recomendadas de seguridad preventivas ... 667
Prácticas recomendadas de detección de seguridad ... 669
Referencia del lenguaje CQL ... 672
Elementos del lenguaje ... 672
Identificadores ... 672
Constantes ... 673
Términos ... 673
Tipos de datos ... 673
Codificación JSON de los tipos de datos de Amazon Keyspaces ... 677
Instrucciones DDL ... 681
Keyspaces ... 682
Tablas ... 684
instrucciones DML ... 698
SELECT ... 698
INSERT ... 701
UPDATE ... 703
DELETE ... 704
Funciones integradas ... 705
Funciones escalares ... 705
Cuotas ... 708
Cuotas de servicio de Amazon Keyspaces ... 708
Aumento o reducción del rendimiento (para las tablas aprovisionadas) ... 714
Aumento del rendimiento aprovisionado ... 714

Reducción de rendimiento aprovisionado 714
Cifrado en reposo de Amazon Keyspaces 714
Historial de documentos 716
..... dccxxvii

¿Qué es Amazon Keyspaces (para Apache Cassandra)?

Amazon Keyspaces (para Apache Cassandra) es un servicio de base de datos escalable, altamente disponible y administrado compatible con Apache Cassandra. Con Amazon Keyspaces, no tiene que aprovisionar, parchear ni administrar servidores, ni tampoco instalar, mantener ni utilizar software.

Amazon Keyspaces funciona sin servidor, por lo que solo paga por los recursos que utiliza, y el servicio escala automáticamente las tablas en alza o en baja en respuesta al tráfico de la aplicación. Puede crear aplicaciones que atiendan miles de solicitudes por segundo con un rendimiento y un almacenamiento prácticamente ilimitados.

> **Note**
>
> Apache Cassandra es un almacén de datos de código abierto y columnas anchas que está diseñado para gestionar grandes cantidades de datos. Para obtener más información, consulte Apache Cassandra.

Amazon Keyspaces facilita la migración, ejecución y escalado de las cargas de trabajo de Cassandra en la Nube de AWS. Con tan solo unos clics en la consola de administración de AWS o unas pocas líneas de código, puede crear espacios de claves y tablas en Amazon Keyspaces, sin necesidad de implementar ninguna infraestructura ni instalar software.

Con Amazon Keyspaces, puede ejecutar sus cargas de trabajo de Cassandra existentes en AWS utilizando el mismo código de aplicación y las mismas herramientas para desarrolladores de Cassandra que utiliza actualmente.

Para obtener una lista de Regiones de AWS y de puntos de conexión disponibles, consulte Puntos de conexión de servicios para Amazon Keyspaces.

Le recomendamos que, para comenzar, lea las siguientes secciones:

Temas

- Amazon Keyspaces: Cómo funciona
- Casos de uso de Amazon Keyspaces
- ¿Qué es Cassandra Query Language (CQL)?

Amazon Keyspaces: Cómo funciona

Amazon Keyspaces elimina la sobrecarga administrativa que supone administrar Cassandra. Para entender por qué, resulta útil comenzar con la arquitectura de Cassandra y luego compararla con la de Amazon Keyspaces.

Temas

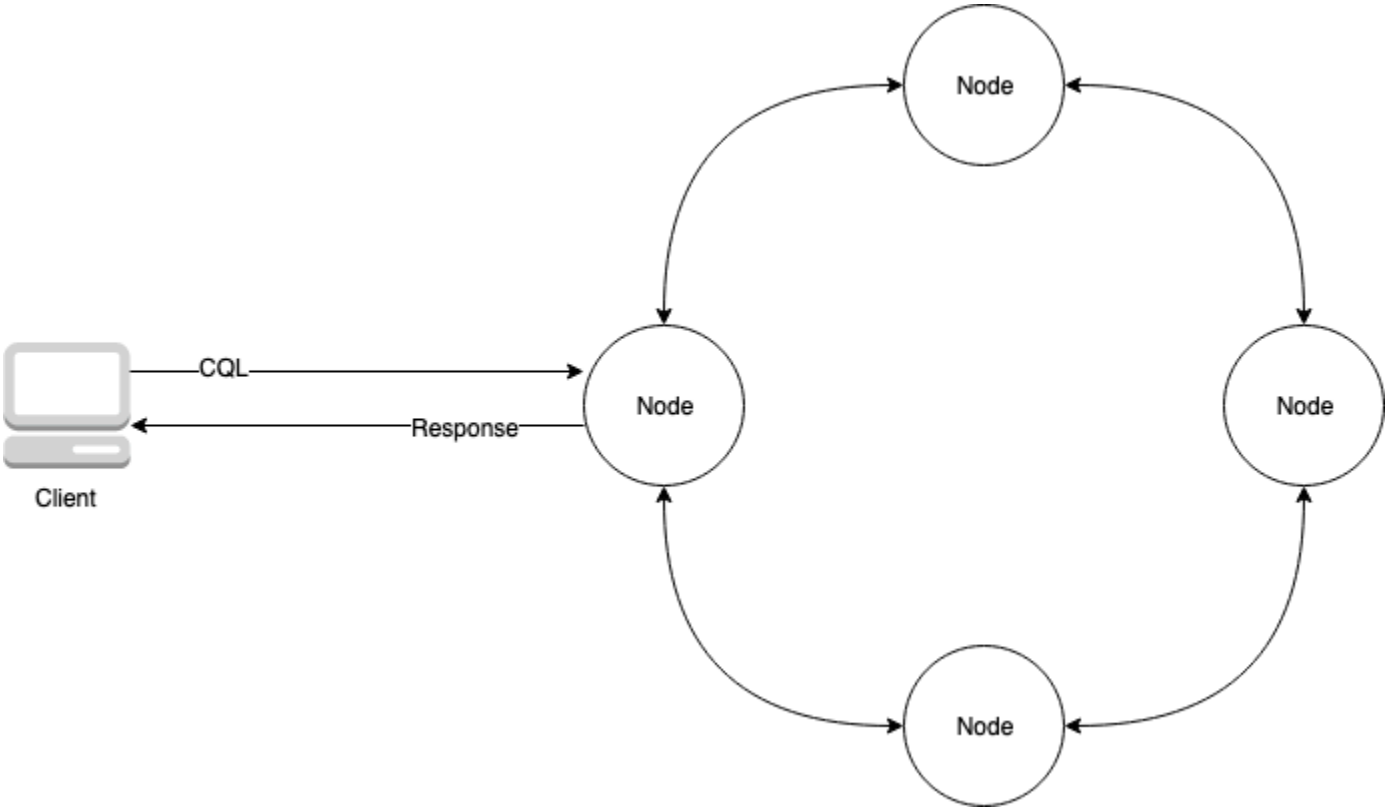
- Arquitectura de alto nivel: Apache Cassandra vs. Amazon Keyspaces
- Modelo de datos de Cassandra
- Acceso a Amazon Keyspaces desde una aplicación

Arquitectura de alto nivel: Apache Cassandra vs. Amazon Keyspaces

Apache Cassandra tradicional se implementa en un clúster formado por uno o más nodos. Usted es responsable de administrar cada nodo y de añadir y eliminar nodos a medida que su clúster escala.

Un programa cliente accede a Cassandra conectándose a uno de los nodos y emitiendo instrucciones en Cassandra Query Language (CQL). CQL es similar a SQL, el popular lenguaje utilizado en las bases de datos relacionales. Si bien Cassandra no es una base de datos relacional, CQL proporciona una interfaz familiar para consultar y manipular datos en Cassandra.

El siguiente diagrama muestra un clúster simple de Apache Cassandra formado por cuatro nodos.



Una implementación de Cassandra en producción puede constar de cientos de nodos ejecutándose en cientos de ordenadores físicos a través de uno o varios centros de datos físicos. Esto puede suponer una carga operativa para los desarrolladores de aplicaciones que necesitan aprovisionar, parchear y administrar los servidores, además de instalar, mantener y operar el software.

Con Amazon Keyspaces (para Apache Cassandra), no necesita aprovisionar, parchear ni administrar servidores, por lo que puede centrarse en crear mejores aplicaciones. Amazon Keyspaces ofrece dos modos de capacidad de rendimiento para lecturas y escrituras: bajo demanda y aprovisionada. Puede elegir el modo de capacidad de rendimiento de su tabla para optimizar el precio de las lecturas y escrituras en función de la previsibilidad y variabilidad de su carga de trabajo.

Con el modo bajo demanda, paga solo por las lecturas y escrituras que su aplicación realmente realiza. No necesita especificar de antemano la capacidad de rendimiento de su tabla. Amazon Keyspaces acomoda el tráfico de su aplicación casi al instante a medida que aumenta o disminuye, lo que lo convierte en una buena opción para aplicaciones con tráfico impredecible.

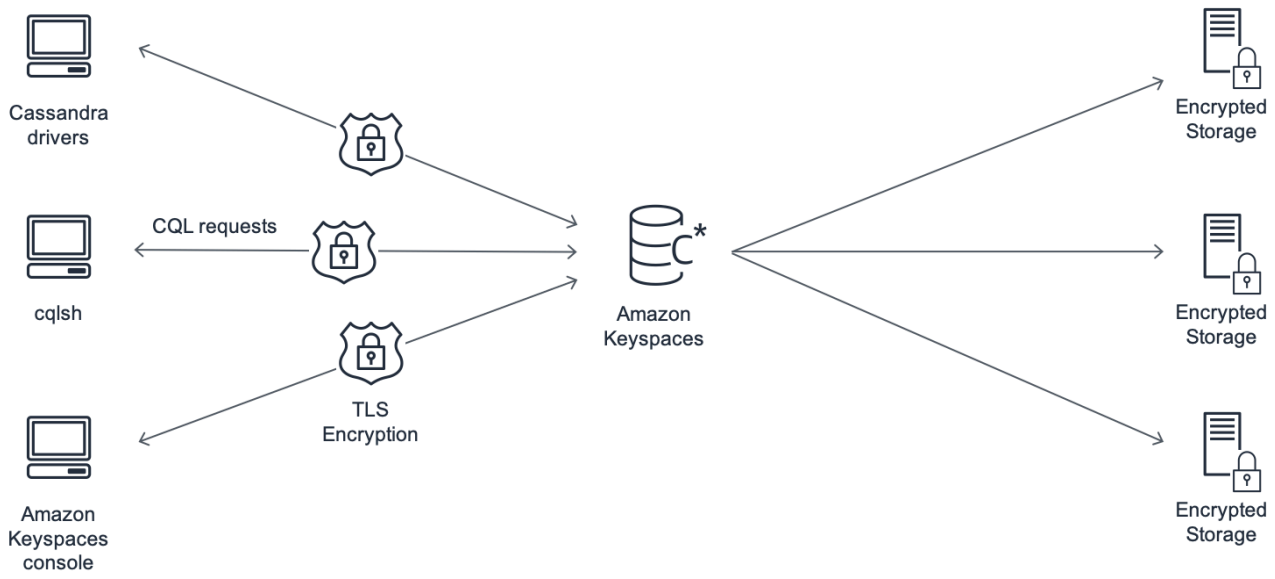
El modo de capacidad aprovisionada le ayuda a optimizar el precio del rendimiento si tiene un tráfico de aplicaciones predecible y puede prever con antelación los requisitos de capacidad de su tabla. Con el modo de capacidad aprovisionada, usted especifica el número de lecturas y escrituras por

segundo que espera que realice su aplicación. Al habilitar escalado automático, puede aumentar y disminuir automáticamente la capacidad aprovisionada de su tabla.

Puede cambiar el modo de capacidad de su tabla una vez al día a medida que obtenga más información sobre los patrones de tráfico de su carga de trabajo, o si espera tener un gran aumento de tráfico, por ejemplo, debido a un evento importante que prevea que generará mucho tráfico en la tabla. Para obtener más información sobre la capacidad de lectura y escritura aprovisionada, consulte the section called “Modos de capacidad de lectura/escritura”.

Amazon Keyspaces (para Apache Cassandra) almacena tres copias de sus datos en múltiples Zonas de Disponibilidad para garantizar durabilidad y alta disponibilidad. Además, se beneficia de un centro de datos y una arquitectura de red creados para satisfacer los requisitos de las organizaciones más sensibles a la seguridad. El cifrado en reposo se habilita de forma automática al crear una nueva tabla de Amazon Keyspaces y todas las conexiones de cliente requieren Seguridad de la capa de transporte (TLS). Otras características de seguridad de AWS incluyen monitoreo, AWS Identity and Access Management y puntos de conexión de nube privada virtual (VPC). Para obtener información general sobre todas las características de seguridad disponibles, consulte Seguridad.

En el siguiente diagrama se muestra la arquitectura de Amazon Keyspaces.



Un programa cliente accede a Amazon Keyspaces conectándose a un punto de conexión predeterminado (nombre de host y número de puerto) y emitiendo instrucciones CQL. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Modelo de datos de Cassandra

La forma en que modele los datos para su caso de negocio es fundamental a fin de lograr un rendimiento óptimo de Amazon Keyspaces. Un modelo de datos deficiente puede degradar significativamente el rendimiento.

Si bien CQL tiene un aspecto similar a SQL, los backends de Cassandra y las bases de datos relacionales son muy diferentes y deben abordarse de forma distinta. A continuación se exponen algunas de las cuestiones más significativas a tener en cuenta:

Almacenamiento

Puede visualizar sus datos de Cassandra en tablas, en las que cada fila representa un registro y cada columna un campo dentro de ese registro.

Diseño de tablas: Primero la consulta

En CQL no existen `JOIN`. Por lo tanto, debería diseñar sus tablas teniendo en cuenta la forma de sus datos y cómo necesita acceder a ellos para sus casos de uso empresarial. Esto podría dar lugar a una desnormalización con datos duplicados. Debería diseñar cada una de sus tablas de forma específica para un patrón de acceso concreto.

Particiones

Sus datos se almacenan en particiones en disco. El número de particiones en las que se almacenan sus datos y cómo se distribuyen entre las particiones viene determinado por su clave de partición. La forma en que defina su clave de partición puede tener un impacto significativo en el rendimiento de sus consultas.

clave principal

En Cassandra, los datos se almacenan como pares clave-valor. Para ello, cada tabla de Cassandra debe tener una clave principal, que es la clave de cada fila de la tabla. La clave principal es el compuesto de una clave de partición requerida y columnas de agrupación opcionales. Los datos que conformen la clave principal deben ser únicos en todos los registros de una tabla.

- Clave de partición: la parte clave de partición de la clave principal es obligatoria y determina en qué partición de su clúster se almacenan los datos. La clave de partición puede ser una única columna o un valor compuesto formado por dos o más columnas. Utilizaría una clave de partición compuesta si una clave de partición de una única columna diera como resultado

que una sola partición o muy pocas particiones tuvieran la mayoría de los datos y, por tanto, soportaran la mayoría de las operaciones de E/S de disco.

- Columna de agrupación: la parte opcional de la columna de agrupación de su clave principal determina cómo se agrupan y ordenan los datos dentro de cada partición. Si incluye una columna de agrupación en su clave principal, la columna de agrupación puede tener una o más columnas. Si hay múltiples columnas en la columna de agrupación, el orden de clasificación viene determinado por el orden en que aparecen las columnas en la columna de agrupación, de izquierda a derecha.

Acceso a Amazon Keyspaces desde una aplicación

Amazon Keyspaces (para Apache Cassandra) implementa la API Apache Cassandra Query Language (CQL), por lo que puede utilizar CQL y los controladores de Cassandra que ya utilice. Actualizar su aplicación es tan sencillo como actualizar su controlador de Cassandra o su configuración de `cqlsh` para que apunte al punto de conexión del servicio de Amazon Keyspaces.

> **Note**
>
> Como ayuda para empezar, puede encontrar ejemplos de códigos integrales de conexión a Amazon Keyspaces utilizando diversos controladores de cliente de Cassandra en el repositorio de ejemplos de código de Amazon Keyspaces en GitHub.

Considere el siguiente programa Python, que se conecta a un clúster de Cassandra y consulta una tabla.

```
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)
```

Para ejecutar el mismo programa en Amazon Keyspaces, necesita:

- Añadir el punto de conexión del clúster y el puerto: por ejemplo, el host se puede sustituir por un punto de conexión de servicio, como `cassandra.us-east-2.amazonaws.com`, y el número de puerto por: `9142`.
- Añadir la configuración TLS/SSL: para obtener más información sobre cómo añadir la configuración TLS/SSL para conectarse a Amazon Keyspaces con un controlador de cliente Python para Cassandra, consulte Uso de un controlador de cliente Python para Cassandra para acceder mediante programación a Amazon Keyspaces.

Casos de uso de Amazon Keyspaces

A continuación se describen algunas de las formas en que puede utilizar Amazon Keyspaces:

- Creación de aplicaciones que requieran baja latencia: procese datos a gran velocidad para aplicaciones que requieran una latencia de un dígito de milisegundo, como mantenimiento de equipos industriales, monitoreo empresarial, administración de flotas y optimización de rutas.
- Creación de aplicaciones utilizando tecnologías de código abierto: construya aplicaciones en AWS utilizando API y controladores de Cassandra de código abierto que están disponibles para una amplia gama de lenguajes de programación, como Java, Python, Ruby, Microsoft .NET, Node.js, PHP, C++, Perl y Go. Para ver ejemplos de código, consulte Bibliotecas y herramientas.
- Movimiento de sus cargas de trabajo de Cassandra a la nube: administrar usted mismo las tablas de Cassandra lleva mucho tiempo y resulta costoso. Con Amazon Keyspaces, puede configurar, asegurar y escalar tablas de Cassandra en la Nube de AWS sin administrar la infraestructura. Para obtener más información, consulte Administración de recursos sin servidor.

¿Qué es Cassandra Query Language (CQL)?

Cassandra Query Language (CQL) es el lenguaje principal para comunicar con Apache Cassandra. Amazon Keyspaces (para Apache Cassandra) es compatible con la API 3.x de CQL (retrocompatible con la versión 2.x).

Para ejecutar consultas CQL, puede realizar una de las siguientes acciones:

- Utilizar el editor CQL en la AWS Management Console.

- Ejecutarlas en el cliente `cqlsh`.
- Ejecutarlas mediante programación utilizando un controlador de cliente de Cassandra con licencia Apache 2.0.

Para obtener más información sobre el uso de estos métodos para acceder a Amazon Keyspaces, consulte Acceso a Amazon Keyspaces (para Apache Cassandra).

Para obtener más información sobre CQL, consulte Referencia del lenguaje CQL para Amazon Keyspaces (para Apache Cassandra).

¿Cómo se compara Amazon Keyspaces (para Apache Cassandra) con Apache Cassandra?

Amazon Keyspaces (para Apache Cassandra) aparece como un clúster de nueve nodos de Apache Cassandra 3.11.2 ante los clientes y admite controladores y clientes compatibles con Apache Cassandra 3.11.2. Amazon Keyspaces admite la API de Cassandra Query Language (CQL) 3.x y es retrocompatible con la versión 2.x. Con Amazon Keyspaces, puede ejecutar sus cargas de trabajo de Cassandra en AWS utilizando el mismo código de aplicación de Cassandra, los controladores con licencia Apache 2.0 y las herramientas que utiliza actualmente.

Amazon Keyspaces admite todas las operaciones del plano de datos de Cassandra de uso habitual, como la creación de espacios de claves y tablas, y la lectura y escritura de datos. Amazon Keyspaces funciona sin servidor, por lo que no tiene que aprovisionar, parchear ni administrar servidores. Tampoco tiene que instalar, mantener ni utilizar software. Como resultado, las operaciones de la API del plano de control de Cassandra para administrar la configuración de clústeres y nodos no son necesarias para utilizar Amazon Keyspaces.

Ajustes como el factor de réplica y el nivel de coherencia se configuran automáticamente a fin de ofrecerle alta disponibilidad, durabilidad y un dígito de milisegundo de rendimiento.

Temas

- Diferencias funcionales: Amazon Keyspaces vs. Apache Cassandra
- API, operaciones, funciones y tipos de datos de Cassandra admitidos en Amazon Keyspaces
- Niveles de coherencia de Apache Cassandra admitidos en Amazon Keyspaces

Diferencias funcionales: Amazon Keyspaces vs. Apache Cassandra

A continuación se indican las diferencias funcionales entre Amazon Keyspaces y Apache Cassandra.

Temas

- API, operaciones y tipos de datos de Apache Cassandra
- Creación y eliminación asíncronas de espacios de claves y tablas
- Autenticación y autorización

- Lote
- Configuración del clúster
- Conexiones
- Palabra clave IN
- Ajuste de rendimiento de las consultas CQL
- Colecciones FROZEN
- Transacciones ligeras
- Equilibrio de carga
- Paginación
- Particionadores
- Instrucciones preparadas
- Eliminación de rangos
- Tablas del sistema
- Marcas temporales

API, operaciones y tipos de datos de Apache Cassandra

Amazon Keyspaces admite todas las operaciones del plano de datos de Cassandra de uso habitual, como la creación de espacios de claves y tablas, y la lectura y escritura de datos. Para ver qué se admite en la actualidad, consulte API, operaciones, funciones y tipos de datos de Cassandra admitidos en Amazon Keyspaces.

Creación y eliminación asíncronas de espacios de claves y tablas

Amazon Keyspaces realiza operaciones de lenguaje de definición de datos (DDL), como la creación y eliminación de espacios de claves y tablas, de forma asíncrona. Para obtener información sobre cómo monitorear el estado de creación de recursos, consulte the section called “Creación de espacios de claves” y the section called “Creación de tablas”. Para obtener una lista de instrucciones DDL en la referencia del lenguaje CQL, consulte the section called “Instrucciones DDL”.

Autenticación y autorización

Amazon Keyspaces (para Apache Cassandra) usa AWS Identity and Access Management (IAM) para la autenticación y autorización de los usuarios y admite políticas de autorización equivalentes a

las de Apache Cassandra. Como tal, Amazon Keyspaces no admite los comandos de configuración de seguridad de Apache Cassandra.

Lote

Amazon Keyspaces admite comandos en lotes no registrados con un máximo de 30 comandos en el lote. Solo se permiten los comandos incondicionales INSERT, UPDATE o DELETE en un lote. No se admiten lotes registrados.

Configuración del clúster

Amazon Keyspaces funciona sin servidor, por lo que no hay clústeres, hosts ni máquinas virtuales Java (JVM) que configurar. La configuración de Cassandra para compactación, compresión, almacenamiento en caché, recopilación de elementos no utilizados y filtrado bloom no es aplicable a Amazon Keyspaces y se ignora si se especifica.

Conexiones

Puede utilizar los controladores de Cassandra existentes para comunicar con Amazon Keyspaces, pero debe configurarlos de forma diferente. Amazon Keyspaces admite hasta 3000 consultas CQL por conexión TCP por segundo, pero no hay límite en el número de conexiones que un controlador puede establecer.

La mayoría de los controladores de código abierto para Cassandra establecen un grupo de conexiones a Cassandra y equilibran la carga de consultas sobre ese grupo de conexiones. Amazon Keyspaces expone 9 direcciones IP pares a los controladores y el comportamiento predeterminado de la mayoría de los controladores es establecer una única conexión con cada dirección IP homóloga. Por lo tanto, el rendimiento máximo de consultas CQL de un controlador que utilice la configuración predeterminada es de 27 000 consultas CQL por segundo.

Para aumentar esta cifra, le recomendamos aumentar el número de conexiones por dirección IP que su controlador mantiene en su grupo de conexiones. Por ejemplo, si establece el máximo de conexiones por dirección IP en 2 dobles, duplicará el rendimiento máximo de su controlador a 54 000 consultas CQL por segundo.

Como práctica recomendada, recomendamos configurar los controladores para utilizar 500 consultas CQL por segundo por conexión para tener en cuenta la sobrecarga y mejorar la distribución. En este escenario, la planificación para 18 000 consultas CQL por segundo requiere 36 conexiones.

Configurar el controlador para 4 conexiones a través de 9 puntos de conexión proporciona 36 conexiones que realicen 500 consultas por segundo. Para obtener más información sobre las prácticas recomendadas para las conexiones, consulte the section called “Conexiones”.

Al conectarse con puntos de conexión de VPC, es posible que haya menos puntos de conexión disponibles. Esto significa que tiene que aumentar el número de conexiones en la configuración del controlador. Para obtener más información sobre las prácticas recomendadas para las conexiones VPC, consulte the section called “Conexiones de puntos de conexión de VPC”.

Palabra clave **`IN`**

Amazon Keyspaces admite la palabra clave `IN` en la instrucción `SELECT`. `IN` no se admite con `UPDATE` y `DELETE`. Al utilizar la palabra clave `IN` en la instrucción `SELECT`, los resultados de la consulta se devuelven en el orden en que se presentan las claves en la instrucción `SELECT`. En Cassandra, los resultados se ordenan lexicográficamente.

Al utilizar `ORDER BY`, no se admite la reordenación completa con la paginación deshabilitada y los resultados se ordenan dentro de una página. No se admiten las consultas segmentadas con la palabra clave `IN`. No se admiten `TOKENS` con la palabra clave `IN`. Amazon Keyspaces procesa las consultas con la palabra clave `IN` creando subconsultas. Cada subconsulta cuenta como una conexión dentro del límite de 3000 consultas CQL por conexión TCP por segundo. Para obtener más información, consulte the section called “Instrucción `IN SELECT`”.

Ajuste de rendimiento de las consultas CQL

Amazon Keyspaces admite hasta 3000 consultas CQL por conexión TCP por segundo, pero no hay límite en el número de conexiones que un controlador puede establecer.

La mayoría de los controladores de código abierto para Cassandra establecen un grupo de conexiones a Cassandra y equilibran la carga de consultas sobre ese grupo de conexiones. Amazon Keyspaces expone 9 direcciones IP pares a los controladores y el comportamiento predeterminado de la mayoría de los controladores es establecer una única conexión con cada dirección IP homóloga. Por lo tanto, el rendimiento máximo de consultas CQL de un controlador que utilice la configuración predeterminada será de 27 000 consultas CQL por segundo.

Para aumentar esta cifra, le recomendamos aumentar el número de conexiones por dirección IP que su controlador mantiene en su grupo de conexiones. Por ejemplo, si establece el máximo de conexiones por dirección IP en 2, duplicará el rendimiento máximo de su controlador a 54 000 consultas CQL por segundo.

Colecciones `FROZEN`

La palabra clave `FROZEN` en Cassandra serializa múltiples componentes de un tipo de datos de colección en un único valor inmutable que se trata como un `BLOB`. Las instrucciones `INSERT` y `UPDATE` sobrescriben toda la colección.

Amazon Keyspaces admite de forma predeterminada hasta cinco niveles de anidamiento para las colecciones congeladas. Para obtener más información, consulte the section called “Cuotas de servicio de Amazon Keyspaces”.

Amazon Keyspaces no admite comparaciones de desigualdades que utilicen toda la colección congelada en una instrucción condicional `UPDATE` o `SELECT`. El comportamiento de las colecciones y las colecciones congeladas es el mismo en Amazon Keyspaces.

Cuando se utilizan colecciones congeladas con marcas de tiempo del lado del cliente, en el caso de que la marca de tiempo de una operación de escritura coincida con la marca de tiempo de una columna existente que no esté caducada o lapidada, Amazon Keyspaces no realiza comparaciones. En su lugar, deja que el servidor determine el escritor más reciente y el escritor más reciente gana.

Para obtener más información sobre colecciones anidadas, consulte the section called “Tipos de colección”.

Transacciones ligeras

Amazon Keyspaces (para Apache Cassandra) admite íntegramente la funcionalidad compare y set en los comandos INSERT, UPDATE y DELETE, que se conocen como transacciones ligeras (LWT) en Apache Cassandra. Como oferta sin servidor, Amazon Keyspaces (para Apache Cassandra) proporciona un rendimiento coherente a cualquier escala, incluso para las transacciones ligeras. Con Amazon Keyspaces, el rendimiento no se ve afectado por el uso de transacciones ligeras.

Equilibrio de carga

Las entradas de la tabla `system.peers` corresponden a equilibradores de carga de Amazon Keyspaces. Para obtener los mejores resultados, recomendamos utilizar una política de equilibrador de carga round-robin y ajustar el número de conexiones por IP a las necesidades de su aplicación.

Paginación

Amazon Keyspaces pagina los resultados en función del número de filas que lee para procesar una solicitud, no del número de filas devueltas en el conjunto de resultados. Como resultado, algunas

páginas podrían contener menos filas de las que especifique en PAGE SIZE para las consultas filtradas. Además, Amazon Keyspaces pagina los resultados automáticamente después de leer 1 MB de datos para ofrecer a los clientes un rendimiento de lectura coherente de un dígito de milisegundo. Para obtener más información, consulte the section called “Paginación de resultados”.

Particionadores

El particionador predeterminado en Amazon Keyspaces es el `Murmur3Partitioner` compatible con Cassandra. Además, puede elegir entre utilizar el `DefaultPartitioner` de Amazon Keyspaces o el `RandomPartitioner` compatible con Cassandra.

Con Amazon Keyspaces, puede cambiar de forma segura el particionador de su cuenta sin tener que recargar los datos de Amazon Keyspaces. Una vez completado el cambio de configuración, que tarda unos 10 minutos, los clientes verán automáticamente la nueva configuración del particionador la próxima vez que se conecten. Para obtener más información, consulte the section called “Uso de particionadores”.

Instrucciones preparadas

Amazon Keyspaces admite el uso de instrucciones preparadas para operaciones de lenguaje de manipulación de datos (DML), como lectura y escritura de datos. En la actualidad, Amazon Keyspaces no admite el uso de instrucciones preparadas para operaciones de lenguaje de definición de datos (DDL), como la creación de tablas y espacios de claves. Las operaciones DDL deben ejecutarse fuera de las instrucciones preparadas.

Eliminación de rangos

Amazon Keyspaces admite la eliminación de filas dentro de un rango. Un rango es un conjunto contiguo de filas dentro de una partición. Puede especificar un rango en una operación DELETE utilizando una cláusula WHERE. Puede especificar que el rango sea una partición entera.

Además, puede especificar que un rango sea un subconjunto de filas contiguas dentro de una partición utilizando operadores relacionales (por ejemplo, “**>**”, “**<**”), o incluyendo la clave de partición y omitiendo una o más columnas de agrupación. Con Amazon Keyspaces, puede eliminar hasta 1000 filas dentro de un rango en una sola operación. Además, las eliminaciones de rangos son atómicas, pero no aisladas.

Tablas del sistema

Amazon Keyspaces rellena las tablas del sistema que son necesarias para los controladores de Cassandra de código abierto con licencia Apache 2.0. Las tablas del sistema visibles para un cliente contienen información exclusiva del usuario autenticado. Las tablas del sistema están totalmente controladas por Amazon Keyspaces y son de solo lectura.

El acceso de solo lectura a las tablas del sistema es obligatorio y puede controlarlo con las políticas de acceso de IAM. Para obtener más información, consulte the section called “Administración de acceso mediante políticas”. Debe definir las políticas de control de acceso basadas en etiquetas para las tablas del sistema de forma diferente en función de si utiliza el AWS SDK o las llamadas a la API Cassandra Query Language (CQL) a través de los controladores y las herramientas de desarrollo de Cassandra. Para obtener más información sobre el control de acceso basado en etiquetas para las tablas del sistema, consulte the section called “ Acceso a recursos de Amazon Keyspaces basado en etiquetas”.

Si accede a Amazon Keyspaces utilizando puntos de conexión de VPC de Amazon, verá entradas en la tabla `system.peers` para cada punto de conexión de VPC de Amazon que Amazon Keyspaces tenga permisos para ver. Como resultado, su controlador Cassandra podría emitir un mensaje de advertencia sobre el propio nodo de control en la tabla `system.peers`. Puede ignorar sin problemas esta advertencia.

Marcas temporales

En Amazon Keyspaces, las marcas de tiempo a nivel de celda que son compatibles con las marcas de tiempo predeterminadas en Apache Cassandra son una característica opcional.

La cláusula `USING TIMESTAMP` y la función `WRITETIME` solo están disponibles cuando las marcas de tiempo del lado del cliente están activadas para una tabla. Para obtener más información sobre las marcas de tiempo del lado del cliente en Amazon Keyspaces, consulte Marcas de tiempo del lado del cliente.

API, operaciones, funciones y tipos de datos de Cassandra admitidos en Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) es compatible con la API de Cassandra Query Language (CQL) 3.11 (retrocompatible con la versión 2.x).

Amazon Keyspaces admite todas las operaciones del plano de datos de Cassandra de uso habitual, como la creación de espacios de claves y tablas, y la lectura y escritura de datos.

En las secciones siguientes se enumeran las funcionalidades admitidas.

Temas

- Compatibilidad con la API de Cassandra
- Compatibilidad con la API de plano de control de Cassandra
- Compatibilidad con la API de plano de datos de Cassandra
- Compatibilidad con las funciones de Cassandra
- Compatibilidad con tipos de datos de Cassandra

Compatibilidad con la API de Cassandra

Operación de la API	Compatible
`CREATE KEYSPACE`	Sí
`ALTER KEYSPACE`	Sí
`DROP KEYSPACE`	Sí
`CREATE TABLE`	Sí
`ALTER TABLE`	Sí
`DROP TABLE`	Sí
`CREATE INDEX`	No
`DROP INDEX`	No
`UNLOGGED BATCH`	Sí
`LOGGED BATCH`	No
`SELECT`	Sí
`INSERT`	Sí

Operación de la API	Compatible
DELETE	Sí
UPDATE	Sí
USE	Sí
CREATE TYPE	No
ALTER TYPE	No
DROP TYPE	No
CREATE TRIGGER	No
DROP TRIGGER	No
CREATE FUNCTION	No
DROP FUNCTION	No
CREATE AGGREGATE	No
DROP AGGREGATE	No
CREATE MATERIALIZED VIEW	No
ALTER MATERIALIZED VIEW	No
DROP MATERIALIZED VIEW	No
TRUNCATE	No

Compatibilidad con la API de plano de control de Cassandra

Dado que Amazon Keyspaces está administrado, las operaciones de la API de plano de control de Cassandra para administrar la configuración de clústeres y nodos no son necesarias. En consecuencia, las siguientes características de Cassandra no son aplicables.

Funcionalidad	Motivo
Conmutador de escrituras duraderas	Todas las escrituras son duraderas
Ajustes de reparación de lectura	No aplicable
Segundos de gracia de GC	No aplicable
Configuración del filtro Bloom	No aplicable
Configuración de compactación	No aplicable
Configuración de compresión	No aplicable
Configuración de la caché	No aplicable
Configuración de seguridad	Sustituido por IAM

Compatibilidad con la API de plano de datos de Cassandra

Funcionalidad	Compatible
Compatibilidad con JSON para las instrucciones SELECT e INSERT	Sí
Columnas estáticas	Sí
Periodo de vida (TTL)	Sí

Compatibilidad con las funciones de Cassandra

Para obtener más información sobre las funciones admitidas, consulte the section called “Funciones integradas”.

Función	Compatible
Funciones `Aggregate`	No

Función	Compatible
Conversión de `Blob`	Sí
`Cast`	Sí
Funciones `Datetime`	Sí
Funciones de conversión de tiempo	Sí
Funciones `TimeUuid`	Sí
`Token`	Sí
`User defined functions (UDF)`	No
`Uuid`	Sí

Compatibilidad con tipos de datos de Cassandra

Tipo de datos	Compatible	Nota
`ascii`	Sí	
`bigint`	Sí	
`blob`	Sí	
`boolean`	Sí	
`counter`	Sí	
`date`	Sí	
`decimal`	Sí	
`double`	Sí	
`float`	Sí	

Tipo de datos	Compatible	Nota
`frozen`	Sí	
`inet`	Sí	
`int`	Sí	
`list`	Sí	
`map`	Sí	
`set`	Sí	
`smallint`	Sí	
`text`	Sí	
`time`	Sí	
`timestamp`	Sí	
`timeuuid`	Sí	
`tinyint`	Sí	
`tuple`	Sí	
`user-defined types` (UDT)	No	Para refactorizar los UDT con búferes de protocolo, consulte Búferes de protocolo de Amazon Keyspaces.
`uuid`	Sí	
`varchar`	Sí	
`varint`	Sí	

Niveles de coherencia de Apache Cassandra admitidos en Amazon Keyspaces

En los temas de esta sección se describen los niveles de coherencia de Apache Cassandra admitidos para las operaciones de lectura y escritura en Amazon Keyspaces (para Apache Cassandra).

Temas

- Niveles de coherencia de escritura
- Niveles de coherencia de lectura
- Niveles de coherencia no admitidos

Niveles de coherencia de escritura

Amazon Keyspaces replica todas las operaciones de escritura tres veces en múltiples zonas de disponibilidad para garantizar durabilidad y alta disponibilidad. Las escrituras se almacenan de forma duradera antes de que se reconozcan utilizando el nivel de coherencia `LOCAL_QUORUM`. Por cada escritura de 1 KB, se le factura 1 unidad de capacidad de escritura (WCU) para las tablas que utilicen el modo de capacidad aprovisionada o 1 unidad de solicitud de escritura (WRU) para las tablas que utilicen el modo bajo demanda.

Puede utilizar `cqlsh` para establecer la coherencia de todas las consultas de la sesión actual a `LOCAL_QUORUM` mediante el siguiente código.

```
CONSISTENCY LOCAL_QUORUM;
```

Para configurar el nivel de coherencia mediante programación, puede establecer la coherencia con los controladores de cliente de Cassandra apropiados. Por ejemplo, los controladores Java versión 4.x le permiten establecer el nivel de coherencia en el archivo `app config` como se muestra a continuación.

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

Si utiliza un controlador Java versión 3.x para Cassandra, puede especificar el nivel de coherencia para la sesión añadiendo `.withQueryOptions(new`

`QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)`, como se muestra en el siguiente ejemplo de código.

```
Session session = Cluster.builder()
                      .addContactPoint(endPoint)
                      .withPort(portNumber)
                      .withAuthProvider(new SigV4AuthProvider("us-east-2"))
                      .withSSL()
                      .withQueryOptions(new
 QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
                      .build()
                      .connect();
```

Para configurar el nivel de coherencia para operaciones de escritura específicas, puede definir la coherencia al llamar `QueryBuilder.insertInto` con un argumento `setConsistencyLevel` al utilizar el controlador Java.

Niveles de coherencia de lectura

Amazon Keyspaces admite tres niveles de coherencia de lectura: `ONE`, `LOCAL_ONE` y. `LOCAL_QUORUM`. Durante una lectura `LOCAL_QUORUM`, Amazon Keyspaces devuelve una respuesta que refleja las actualizaciones más recientes de todas las operaciones de escritura anteriores realizadas correctamente. El uso del nivel de coherencia `ONE` o `LOCAL_ONE` puede mejorar el rendimiento y la disponibilidad de sus solicitudes de lectura, pero es posible que la respuesta no refleje los resultados de una escritura realizada recientemente.

Por cada 4 KB de lectura utilizando la coherencia `ONE` o `LOCAL_ONE`, se le facturan 0,5 unidades de capacidad de lectura (RCU) para las tablas que utilicen el modo de capacidad aprovisionada o 0,5 unidades de solicitud de lectura (RRU) para las tablas que utilicen el modo bajo demanda. Por cada 4 KB de lectura utilizando la coherencia `LOCAL_QUORUM`, se le factura 1 unidad de capacidad de lectura (RCU) para las tablas que utilicen el modo de capacidad aprovisionada o 1 unidad de solicitud de lectura (RRU) para las tablas que utilicen el modo bajo demanda.

Facturación basada en la coherencia de lectura y en el modo de capacidad de rendimiento de lectura por tabla por cada 4 KB de lectura

Nivel de coherencia	Aprovisionado	Bajo demanda
`ONE`	0,5 CRU	0,5 RRU

Nivel de coherencia	Aprovisionado	Bajo demanda
LOCAL_ONE	0,5 CRU	0,5 RRU
LOCAL_QUORUM	1 RCU	1 RRU

Para especificar una coherencia diferente para las operaciones de lectura, llame `QueryBuilder.select` con un argumento `setConsistencyLevel` al utilizar el controlador Java.

Niveles de coherencia no admitidos

Los siguientes niveles de coherencia no son admitidos por Amazon Keyspaces y darán lugar a excepciones.

Niveles de coherencia no admitidos

Apache Cassandra	Amazon Keyspaces
EACH_QUORUM	No admitido
QUORUM	No admitido
ALL	No admitido
TWO	No admitido
THREE	No admitido
ANY	No admitido
SERIAL	No admitido
LOCAL_SERIAL	No admitido

Acceso a Amazon Keyspaces (para Apache Cassandra)

Puede acceder a Amazon Keyspaces mediante la consola AWS CloudShell, mediante programación mediante la ejecución de un `cqlsh` cliente, el AWS SDK o mediante un controlador Cassandra con licencia de Apache 2.0. Amazon Keyspaces admite controladores y clientes compatibles con Apache Cassandra 3.11.2. Antes de acceder a Amazon Keyspaces, debe completar la configuración AWS Identity and Access Management y, a continuación, conceder a una identidad de IAM los permisos de acceso a Amazon Keyspaces.

Con AWS Identity and Access Management figuración

Inscríbase para obtener una Cuenta de AWS

Si no tiene uno Cuenta de AWS, complete los siguientes pasos para crearlo.

Para suscribirse a una Cuenta de AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica e indicar un código de verificación en el teclado del teléfono.

   Cuando te registras en una Cuenta de AWS, Usuario raíz de la cuenta de AWSse crea una. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario administrativo y utilice únicamente el usuario raíz para ejecutar tareas que requieren acceso de usuario raíz.

AWS te envía un correo electrónico de confirmación una vez finalizado el proceso de registro. Puede ver la actividad de la cuenta y administrar la cuenta en cualquier momento entrando en https://aws.amazon.com/ y seleccionando Mi cuenta.

Cómo crear un usuario administrativo

Después de crear un usuario administrativo Cuenta de AWS, asegúrelo Usuario raíz de la cuenta de AWS AWS IAM Identity Center, habilite y cree un usuario administrativo para no usar el usuario root en las tareas diarias.

Proteja su Usuario raíz de la cuenta de AWS

1. Inicie sesión AWS Management Consolecomo propietario de la cuenta seleccionando el usuario root e introduciendo su dirección de Cuenta de AWS correo electrónico. En la siguiente página, escriba su contraseña.

   Para obtener ayuda sobre cómo iniciar sesión con el usuario raíz, consulte Iniciar sesión como usuario raíz en la Guía del usuario de AWS Sign-In .
2. Active la autenticación multifactor (MFA) para el usuario raíz.

   Para obtener instrucciones, consulte Habilitar un dispositivo MFA virtual para el usuario Cuenta de AWS raíz (consola) en la Guía del usuario de IAM.

Cómo crear un usuario administrativo

1. Activar IAM Identity Center

   Consulte las instrucciones en Enabling AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center .
2. En el Centro de identidades de IAM, conceda acceso administrativo a un usuario administrativo.

   Para ver un tutorial sobre su uso Directorio de IAM Identity Center como fuente de identidad, consulte Configurar el acceso de los usuarios con la configuración predeterminada Directorio de IAM Identity Center en la Guía del AWS IAM Identity Center usuario.

Inicio de sesión como usuario administrativo

- Para iniciar sesión con el usuario del IAM Identity Center, utilice la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario del Centro de identidades de IAM.

  Para obtener ayuda para iniciar sesión con un usuario del Centro de identidades de IAM, consulte Iniciar sesión en el portal de AWS acceso en la Guía del AWS Sign-In usuario.

Configuración de Amazon Keyspaces

El acceso a los recursos de Amazon Keyspaces se administra mediante IAM. Con IAM, puede adjuntar políticas a los usuarios, roles e identidades federadas de IAM que conceden permisos de lectura y escritura a recursos específicos en Amazon Keyspaces.

Para empezar a conceder permisos a una identidad de IAM, puedes usar una de las políticas AWS gestionadas para Amazon Keyspaces:

- AmazonKeyspacesFullAccess— esta política concede permisos para acceder a todos los recursos de Amazon Keyspaces con acceso total a todas las funciones.
- AmazonKeyspacesReadOnlyAccess_v2: esta política concede permisos de solo lectura a Amazon Keyspaces.

Para obtener una explicación detallada de las acciones definidas en las políticas administradas, consulte. the section called “Políticas administradas por AWS”

Para limitar el alcance de las acciones que puede realizar una identidad de IAM o limitar los recursos a los que puede acceder, puedes crear una política personalizada que utilice la política `AmazonKeyspacesFullAccess` gestionada como plantilla y eliminar todos los permisos que no necesites. También puedes limitar el acceso a espacios clave o tablas específicos. Para obtener más información sobre cómo restringir acciones o limitar el acceso a recursos específicos en Amazon Keyspaces, consulte. the section called “Funcionamiento de Amazon Keyspaces con IAM”

Para acceder a Amazon Keyspaces después de haber creado Cuenta de AWS y creado una política que conceda acceso a una identidad de IAM a Amazon Keyspaces, vaya a una de las siguientes secciones:

- Uso de la consola
- Utilizando AWS CloudShell
- Conexión mediante programación

Acceso a Amazon Keyspaces mediante la consola

Puede acceder a la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home. Para obtener más información sobre el AWS Management Console acceso, consulte Control del acceso de los usuarios de IAM AWS Management Console en la Guía del usuario de IAM.

Puede utilizar la consola para hacer lo siguiente en Amazon Keyspaces:

- Cree, elimine y gestione tablas y espacios clave.
- Supervisa las métricas importantes de las tablas en la pestaña Supervisar de una tabla:
  - Tamaño de la tabla facturable (bytes)
  - Métricas de capacidad
- Ejecute consultas con el editor CQL, por ejemplo, inserte, actualice y elimine datos.
- Cambie la configuración del particionador de la cuenta.
- Consulta las métricas de rendimiento y error de la cuenta en el panel de control.

Para obtener información sobre cómo crear un espacio de claves y una tabla de Amazon Keyspaces y configurarlos con datos de aplicación de ejemplo, consulte Introducción a Amazon Keyspaces (para Apache Cassandra).

Uso AWS CloudShell para acceder a Amazon Keyspaces

AWS CloudShell es un shell preautenticado y basado en un navegador que se puede iniciar directamente desde. AWS Management Console Puede ejecutar AWS CLI comandos en los AWS servicios mediante el shell que prefiera (Bash o Z shell). PowerShell Para trabajar con Amazon Keyspaces utilizando`cqlsh`, debe instalar el. `cqlsh-expansion` Para obtener instrucciones `cqlsh-expansion` de instalación, consultethe section called “Uso de la `cqlsh-expansion`”.

Se inicia AWS CloudShell desde la AWS Management Console consola y las AWS credenciales que utilizó para iniciar sesión en la consola están disponibles automáticamente en una nueva sesión de shell. Esta autenticación previa de AWS CloudShell los usuarios le permite omitir la configuración de las credenciales al interactuar con AWS servicios como Amazon Keyspaces `cqlsh` mediante AWS CLI la versión 2 (preinstalada en el entorno informático del shell).

Obtener permisos de IAM para AWS CloudShell

Con los recursos de administración de acceso que proporcionan AWS Identity and Access Management, los administradores pueden conceder permisos a los usuarios de IAM para que puedan acceder a las funciones del entorno AWS CloudShell y utilizarlas.

La forma más rápida de que un administrador conceda acceso a los usuarios es mediante una política AWS gestionada. Una política administrada de AWS es una política independiente creada y

administrada por AWS. La siguiente política AWS administrada para se CloudShell puede adjuntar a las identidades de IAM:

- `AWSCloudShellFullAccess`: Concede permiso de uso AWS CloudShell con acceso completo a todas las funciones.

Si desea limitar el alcance de las acciones con las que puede realizar un usuario de IAM AWS CloudShell, puede crear una política personalizada que utilice la política `AWSCloudShellFullAccess` gestionada como plantilla. Para obtener más información sobre cómo limitar las acciones que están disponibles para los usuarios CloudShell, consulte Administrar el AWS CloudShell acceso y el uso con políticas de IAM en la Guía del AWS CloudShell usuario.

> **Note**
>
> Su identidad de IAM también requiere una política que conceda permiso para realizar llamadas a Amazon Keyspaces.

Puedes usar una política AWS gestionada para permitir que tu identidad de IAM acceda a tus Amazon Keyspaces, o bien empezar con la política gestionada como plantilla y eliminar los permisos que no necesites. También puedes limitar el acceso a espacios clave y tablas específicos para crear una política personalizada. La siguiente política gestionada para Amazon Keyspaces se puede adjuntar a las identidades de IAM:

- AmazonKeyspacesFullAccess— Esta política otorga permiso para usar Amazon Keyspaces con acceso total a todas las funciones.

Para obtener una explicación detallada de las acciones definidas en la política gestionada, consultethe section called “Políticas administradas por AWS”.

Para obtener más información sobre cómo restringir acciones o limitar el acceso a recursos específicos en Amazon Keyspaces, consulte. the section called “Funcionamiento de Amazon Keyspaces con IAM”

Interactuar con Amazon Keyspaces mediante AWS CloudShell

Tras lanzarlo AWS CloudShell desde AWS Management Console, puede empezar inmediatamente a interactuar con Amazon Keyspaces mediante la interfaz de línea de comandos `cqlsh` o la interfaz de

línea de comandos. Si aún no lo ha instaladocqlsh-expansion, consulte the section called “Uso de la cqlsh-expansion” los pasos detallados.

> **Note**
>
> Al usar cqlsh-expansion la entrada AWS CloudShell, no necesitas configurar las credenciales antes de realizar llamadas, ya que ya estás autenticado en el shell.

Conéctate a Amazon Keyspaces y crea un nuevo espacio de claves. A continuación, lea una tabla del sistema para confirmar que el espacio de claves se creó con AWS CloudShell

1. Desde AWS Management Console, puede iniciar CloudShell seleccionando las siguientes opciones disponibles en la barra de navegación:
   - Selecciona el CloudShell icono.
   - Comience a escribir «cloudshell» en el cuadro de búsqueda y, a continuación, elija la CloudShell opción.
2. Puede establecer una conexión con Amazon Keyspaces mediante el siguiente comando. Asegúrese de sustituir *cassandra.us-east-1.amazonaws.com* por el punto de enlace correcto para su región.

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

   Si la conexión tiene éxito, debería ver una salida similar a la del siguiente ejemplo.

   ```
   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. Crea un nuevo espacio de claves con el nombremykeyspace. Para ello, puede utilizar el siguiente comando.

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

4. Para confirmar que se creó el espacio de claves, puede leer una tabla del sistema mediante el siguiente comando.

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

Si la llamada se realiza correctamente, la línea de comandos muestra una respuesta del servicio similar a la siguiente salida:

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

(1 rows)
```

Conexión mediante programación a Amazon Keyspaces

En este tema se describen los pasos necesarios para conectarse a Amazon Keyspaces mediante programación. Le guía a través de la creación de credenciales de IAM y enumera los puntos finales de AWS servicio disponibles. En la última sección se muestra cómo conectarse a Amazon Keyspaces mediante cqlsh. Para step-by-step ver tutoriales sobre cómo conectarse a Amazon Keyspaces mediante distintos controladores de Apache Cassandra, consulte. the section called “Utilización de un controlador de cliente de Cassandra” Para ver un step-by-step tutorial que muestra cómo conectarse a Amazon Keyspaces desde un punto de enlace de Amazon VPC, consulte. the section called “Conexión con puntos finales de VPC”

> **Note**
>
> Para ayudarle a empezar, puede encontrar ejemplos de end-to-end código sobre cómo conectarse a Amazon Keyspaces mediante varios controladores de cliente de Cassandra en el repositorio de ejemplos de código de Amazon Keyspaces en. GitHub

Amazon Keyspaces admite controladores y clientes compatibles con Apache Cassandra 3.11.2. Se supone que ya ha completado las instrucciones de AWS configuración incluidas en. Acceso a Amazon Keyspaces

Si ya tiene una Cuenta de AWS, consulte los siguientes temas para obtener información sobre cómo acceder a Amazon Keyspaces mediante cqlsh mediante programación:

Temas

- Creación de credenciales para acceder a Amazon Keyspaces mediante programación
- Puntos de conexión de servicio para Amazon Keyspaces
- Uso de cqlsh para conectarse a Amazon Keyspaces
- Uso de AWS CLI
- Uso de la API
- Uso de Amazon Keyspaces con un SDK AWS
- Uso de un controlador de cliente de Cassandra para acceder a Amazon Keyspaces mediante programación
- Tutorial: Conexión a Amazon Keyspaces desde Amazon Elastic Kubernetes Service

Creación de credenciales para acceder a Amazon Keyspaces mediante programación

Para proporcionar a los usuarios y a las aplicaciones credenciales de acceso programático a los recursos de Amazon Keyspaces, puede realizar cualquiera de las siguientes acciones:

- Cree credenciales específicas del servicio que sean similares al nombre de usuario y la contraseña tradicionales que Cassandra usa para la autenticación y la administración del acceso. AWS las credenciales específicas del servicio están asociadas a un usuario AWS Identity and Access Management (IAM) específico y solo se pueden usar para el servicio para el que se crearon. Para obtener más información, consulte Uso de IAM con Amazon Keyspaces (para Apache Cassandra) en la Guía del usuario de IAM.
- Para mejorar la seguridad, recomendamos crear claves de acceso de IAM para usuarios y roles de IAM que se utilicen en todos los servicios de AWS . El complemento de autenticación SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra habilita la autenticación de llamadas a Amazon Keyspaces utilizando claves de acceso de IAM en vez de nombre de usuario y contraseña. Para obtener más información sobre cómo el complemento SigV4 de Amazon Keyspaces habilita a los usuarios, roles e identidades federadas de IAM para autenticarse en las solicitudes de la API de Amazon Keyspaces, consulte Proceso Signature Version 4 (SigV4) de AWS.

  Puede descargar los complementos de SigV4 desde las siguientes ubicaciones.

  - 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.
- Go: https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Para ver ejemplos de código que muestran cómo establecer conexiones utilizando el complemento de autenticación SigV4, consulte the section called “Utilización de un controlador de cliente de Cassandra”.

Temas

- Generación de credenciales específicas del servicio
- Creación y configuración de credenciales de AWS para Amazon Keyspaces

Generación de credenciales específicas del servicio

Las credenciales específicas del servicio son similares al nombre de usuario y contraseña tradicionales que Cassandra utiliza para la autenticación y la administración del acceso. Las credenciales específicas del servicio permiten a los usuarios de IAM acceder a un servicio de AWS específico. Las credenciales no se pueden usar para acceder a otros servicios. AWS Están asociadas a un usuario de IAM específico y no se pueden utilizar por otros usuarios de IAM.

> **Important**
>
> Las credenciales específicas del servicio están asociadas a un usuario de IAM en concreto y solo se pueden utilizar para el servicio para el que se crearon. Para conceder permisos a las funciones de IAM o a las identidades federadas para acceder a todos sus AWS recursos, debe crear claves de acceso de IAM para la AWS autenticación y utilizar el complemento de autenticación SigV4.

Utilice uno de los siguientes procedimientos para generar una credencial específica del servicio.

Generación de credenciales específicas del servicio mediante la consola

Para generar credenciales específicas del servicio con la consola

1. Inicie sesión en AWS Management Console y abra la consola en. AWS Identity and Access Management https://console.aws.amazon.com/iam/home

2. En el panel de navegación, elija Usuarios y, a continuación, elija el usuario que creó anteriormente y que tiene permisos de Amazon Keyspaces (política vinculada).
3. Elija Security Credentials (Credenciales de seguridad). En Credenciales para Amazon Keyspaces, elija Generar credenciales a fin de generar las credenciales específicas del servicio.

   Las credenciales específicas del servicio ya están disponibles. Esta es la única vez que la contraseña se puede ver o descargar. No puede recuperarla más adelante. Sin embargo, puede restablecerla en cualquier momento. Guarde el usuario y la contraseña en una ubicación segura, ya que los necesitará más adelante.

Genere credenciales específicas del servicio mediante el AWS CLI

Para generar credenciales específicas de un servicio mediante el AWS CLI

Antes de generar las credenciales específicas del servicio, debe descargar, instalar y configurar (): AWS Command Line Interface AWS CLI

1. Descárguelo AWS CLI en http://aws.amazon.com/cli.

   > **Note**
   >
   > Se AWS CLI ejecuta en Windows, macOS o Linux.

2. Siga las instrucciones para instalar la AWS CLI y configurar la AWS CLI de la Guía del AWS Command Line Interface usuario.
3. Con el AWS CLI, ejecute el siguiente comando para generar credenciales específicas del servicio para el usuario`alice`, de modo que pueda acceder a Amazon Keyspaces.

```
aws iam create-service-specific-credential \
    --user-name alice \
    --service-name cassandra.amazonaws.com
```

El resultado es similar al siguiente.

```
{
    "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"
    }
}
```

En la salida, anote los valores de `ServiceUserName` y `ServicePassword`. Guarde estos valores en un lugar seguro, dado que los necesitará más adelante.

> **Important**
>
> Esta es la única vez que la `ServicePassword` estará disponible para usted.

Creación y configuración de credenciales de AWS para Amazon Keyspaces

Para acceder a Amazon Keyspaces mediante programación con la AWS CLI, el SDK de AWS o con controladores de cliente de Cassandra y el complemento SigV4, necesita un usuario o rol de IAM con claves de acceso. Al utilizar AWS mediante programación, usted proporciona sus claves de acceso de AWS para que AWS pueda verificar su identidad en las llamadas programáticas. Sus claves de acceso constan de un ID de clave de acceso (por ejemplo, AKIAIOSFODNN7EXAMPLE) y una clave de acceso secreta (por ejemplo, wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY). En este tema se explican los pasos necesarios en este proceso.

Temas

- Credenciales requeridas por la AWS CLI, el SDK de AWS o el complemento SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra
- Creación de un usuario de IAM para acceso programático a Amazon Keyspaces en su cuenta de AWS
- Creación de nuevas claves de acceso para un usuario de IAM
- Administración de claves de acceso para usuarios de IAM
- Uso de credenciales temporales para conectarse a Amazon Keyspaces mediante un rol de IAM y el complemento SigV4

Credenciales requeridas por la AWS CLI, el SDK de AWS o el complemento SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra

Para autenticar el usuario o rol de IAM se requieren las siguientes credenciales:

`AWS_ACCESS_KEY_ID`

Especifica una clave de acceso de AWS asociada a un usuario o un rol de IAM.

La clave de acceso `aws_access_key_id` es necesaria para conectarse a Amazon Keyspaces mediante programación.

`AWS_SECRET_ACCESS_KEY`

Especifica la clave secreta asociada a la clave de acceso. Se trata básicamente de la "contraseña" de la clave de acceso.

La `aws_secret_access_key` es necesaria para conectarse a Amazon Keyspaces mediante programación.

`AWS_SESSION_TOKEN`: opcional

Especifica el valor del token de sesión que se requiere si utiliza credenciales de seguridad temporales que ha recuperado directamente de las operaciones de AWS Security Token Service. Para obtener más información, consulte the section called “Uso de credenciales temporales para conectarse a Amazon Keyspaces”.

Si se conecta con un usuario de IAM, el `aws_session_token` no es necesario.

Creación de un usuario de IAM para acceso programático a Amazon Keyspaces en su cuenta de AWS

Para obtener credenciales para acceso programático a Amazon Keyspaces con la AWS CLI, el SDK de AWS o el complemento SigV4, primero debe crear un usuario o rol de IAM. El proceso de creación de un usuario de IAM y de configuración del mismo para tener acceso programático a Amazon Keyspaces se muestra en los pasos siguientes:

1. Cree el usuario en el AWS Management Console, el AWS CLI, Tools for Windows PowerShell o utilizar una Operación de la API AWS. Si crea el usuario en la AWS Management Console, las credenciales se crean automáticamente.
2. Si crea el usuario mediante programación, entonces debe crear una clave de acceso (ID de clave de acceso y clave de acceso secreta) para ese usuario en un paso adicional.

3. Otorgue al usuario permisos para acceder a Amazon Keyspaces.

Para obtener información sobre los permisos que necesita para crear un usuario, consulte Permisos necesarios para acceder a los recursos de IAM.

Creación de usuarios de IAM (consola)

Puede utilizar la AWS Management Console para crear usuarios de IAM.

Para crear un usuario de IAM con acceso programático (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Usuarios y, a continuación, elija Agregar usuarios.
3. Escriba el nombre de usuario del nuevo usuario. Este es el nombre de inicio de sesión para AWS.

   > **Note**
   >
   > Los nombres de usuario pueden ser una combinación de un máximo de 64 letras, dígitos y los siguientes caracteres: más (+), igual (=), coma (,), punto (.), arroba (@), guion bajo (_) y guion (-). Los nombres deben ser únicos dentro de una cuenta. No distinguen entre mayúsculas y minúsculas. Por ejemplo, no puede crear dos usuarios llamados TESTUSER y testuser.

4. Seleccione Clave de acceso - Acceso programático para crear una clave de acceso para el nuevo usuario. Puede ver o descargar la clave de acceso cuando llegue a la página Final.

   Elija Siguiente: Permisos.

5. En la página Establecer permisos, elija Añadir políticas existentes directamente para asignar permisos al nuevo usuario.

   Esta opción muestra la lista de políticas administradas de AWS y administradas por el cliente disponibles en su cuenta. Puede introducir `keyspaces` en el campo de búsqueda para mostrar solo las políticas relacionadas con Amazon Keyspaces.

   Para Amazon Keyspaces, las políticas administradas disponibles son `AmazonKeyspacesFullAccess` y `AmazonKeyspacesReadOnlyAccess`. Para obtener más información sobre cada política, consulte the section called “Políticas administradas por AWS”.

Con fines de prueba y para seguir los tutoriales de conexión, seleccione la política `AmazonKeyspacesReadOnlyAccess` para el nuevo usuario. Nota: Como práctica recomendada, le sugerimos que siga el principio de privilegio mínimo y cree políticas personalizadas que limiten el acceso a recursos específicos y solo permitan las acciones necesarias. Para obtener más información sobre las políticas de IAM y ver ejemplos de políticas para Amazon Keyspaces, consulte the section called “Políticas basadas en identidades de Amazon Keyspaces”. Una vez creadas las políticas personalizadas, vincule sus políticas a los grupos y, a continuación, convierta a los usuarios en miembros de los grupos apropiados.

Elija Next: Tags (Siguiente: etiquetas).

6. En la página Añadir etiquetas (opcional) puede añadir etiquetas para el usuario, o elegir Siguiente: Revisar.
7. En la página Revisar puede ver todas las opciones que ha elegido hasta este momento. Cuando esté listo para continuar, elija Crear usuario.
8. Para ver las claves de acceso del usuario (ID de clave de acceso y claves de acceso secretas), elija Show (Mostrar) junto a la contraseña y la clave de acceso. Para guardar las claves de acceso, elija Download .csv (Descargar archivo .csv) y, a continuación, guarde el archivo en un lugar seguro.

> **Important**
>
> Esta es su única oportunidad para ver o descargar las claves de acceso secretas, y necesita esta información antes de que puedan utilizar el complemento SigV4. Guarde el nuevo ID de clave de acceso del usuario y la clave de acceso secreta en un lugar seguro. No volverá a tener acceso a la clave de acceso secreta después de este paso.

Creación de usuarios de IAM (AWS CLI)

Puede utilizar la AWS CLI para crear un usuario de IAM.

Para crear un usuario de IAM con acceso programático (AWS CLI)

1. Cree un usuario con el siguiente código de la AWS CLI.
   - aws iam create-user

2. Dé al usuario acceso programático. Esto requiere claves de acceso, las cuales se pueden generar de las siguientes maneras.

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

   > **Important**
   >
   > Esta es su única oportunidad para ver o descargar las claves de acceso secretas, y necesita esta información antes de que puedan utilizar el complemento SigV4. Guarde el nuevo ID de clave de acceso del usuario y la clave de acceso secreta en un lugar seguro. No volverá a tener acceso a la clave de acceso secreta después de este paso.

3. Vincule al usuario la política `AmazonKeyspacesReadOnlyAccess` que define los permisos del usuario. Nota: Como práctica recomendada, le sugerimos que administre los permisos de usuario añadiendo el usuario a un grupo y adjuntando una política al grupo en vez de vincularla directamente a un usuario.

   - AWS CLI: aws iam attach-user-policy

Creación de nuevas claves de acceso para un usuario de IAM

Si ya tiene un usuario de IAM, puede crear nuevas claves de acceso en cualquier momento. Para obtener más información sobre administración de claves, por ejemplo, cómo rotar las claves de acceso, consulte Administración de las claves de acceso de los usuarios de IAM.

Para crear claves de acceso para un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuyas claves de acceso desea crear.
4. En la página Resumen del usuario, elija la pestaña Credenciales de seguridad.
5. En la sección Access keys (Claves de acceso), haga clic en Create access key (Crear clave de acceso).

Para ver el nuevo par de claves de acceso, elija Show (Mostrar). Sus credenciales tendrán el aspecto siguiente:

- ID de clave de acceso: AKIAIOSFODNN7EXAMPLE
- Clave de acceso secreta: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

> **Note**
>
> No podrá obtener acceso de nuevo a la clave de acceso secreta cuando este cuadro de diálogo se cierre.

6. Para descargar el par de claves, elija Download .csv file (Descargar archivo .csv). Almacene las claves en un lugar seguro.
7. Cuando descargue el archivo .csv, elija Close (Cerrar).

Cuando cree una clave de acceso, el par de claves se activa de forma predeterminada y puede utilizar el par de inmediato.

Administración de claves de acceso para usuarios de IAM

Como práctica recomendada, le sugerimos que no incruste claves de acceso directamente en el código. Los SDK de AWS y las herramientas de línea de comandos de AWS le permiten colocar las claves de acceso en ubicaciones conocidas para que no tenga que guardarlas en código. Ponga las claves de acceso en una de las siguientes ubicaciones:

- Variables de entorno: en un sistema multiusuario, elija variables de entorno de usuario, no variables de entorno de sistema.
- Archivo de credenciales de CLI: los archivos `credentials` y `config` se actualizan al ejecutar el comando `aws configure`. El archivo `credentials` se encuentra en `~/.aws/credentials` en Linux, macOS o Unix, o en `C:\Users\`*`USERNAME`*`\.aws\credentials` en Windows. Este archivo contiene las credenciales del perfil `default` y de todos los perfiles con nombre.
- Archivo de configuración de CLI: los archivos `credentials` y `config` se actualizan al ejecutar el comando `aws configure`. El archivo `config` se encuentra en `~/.aws/config` en Linux, macOS o Unix, o en `C:\Users\`*`USERNAME`*`\.aws\config` en Windows. Este archivo contiene las opciones de configuración del perfil predeterminado y de los perfiles con nombre.

Almacenar las claves de acceso como variables de entorno es un requisito previo para the section called “Complemento de autenticación para Java 4.x”. El cliente busca las credenciales utilizando la cadena de proveedores de credenciales predeterminada y las claves de acceso almacenadas como variables de entorno tienen prioridad sobre todas las demás ubicaciones, por ejemplo, los archivos de configuración. Para obtener más información, consulte Ajustes de configuración y precedencia.

En los siguientes ejemplos se muestra cómo se pueden configurar las variables de entorno para el usuario predeterminado.

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>
```

La configuración de la variable de entorno cambia el valor usado hasta el final de su sesión del intérprete de comandos o hasta que otorgue a la variable un valor diferente. Puede hacer que las variables persistan en sesiones futuras configurándolas en el script de startup del intérprete de comandos.

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>
```

El uso de `set` para configurar una variable de entorno cambia el valor usado hasta que finalice la sesión de símbolo de sistema actual o hasta que otorgue a la variable un valor diferente. El uso de `setx` para establecer una variable de entorno cambia el valor usado en la sesión de símbolo del sistema actual y en todas las sesiones de símbolo del sistema que cree después de ejecutar el comando. La operación no afecta a otros comandos del shell que ya se están ejecutando en el momento de ejecutar el comando.

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>"
```

Si establece una variable de entorno en el símbolo del sistema de PowerShell, tal y como se muestra en los ejemplos anteriores, el valor se guarda únicamente durante la sesión actual.

Para que el valor de la variable de entorno persista en todas las sesiones de PowerShell y del símbolo del sistema, guárdelo mediante la aplicación Sistema en el Panel de control. También puede definir la variable para todas las sesiones de PowerShell futuras añadiéndolo a su perfil de PowerShell. Consulte la documentación de PowerShell para obtener más información sobre el almacenamiento de variables de entorno o su persistencia entre sesiones.

Uso de credenciales temporales para conectarse a Amazon Keyspaces mediante un rol de IAM y el complemento SigV4

Para mejorar la seguridad, puede utilizar credenciales temporales para autenticarse con el complemento SigV4. En muchos casos, no necesita claves de acceso a largo plazo que nunca caducan (como sucede con un usuario de IAM). En su lugar, puede crear un rol de IAM y generar credenciales de seguridad temporales. Las credenciales de seguridad temporales se componen de un ID de clave de acceso y una clave de acceso secreta, pero, además, incluyen un token de seguridad que indica cuándo caducan las credenciales. Para obtener más información sobre cómo utilizar roles de IAM en vez de claves de acceso a largo plazo, consulte Cambio a un rol de IAM (API de AWS).

Para comenzar con las credenciales temporales, primero debe crear un rol de IAM.

Creación de un rol de IAM que conceda acceso de solo lectura a Amazon Keyspaces

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Roles y luego Crear rol.
3. En la página Crear rol, en Seleccionar tipo de entidad de confianza), elija Servicio de AWS. En Elegir un caso de uso, elija Amazon EC2 y luego Siguiente.
4. En la página Añadir permisos, en Políticas de permisos, elija Acceso de solo lectura a Amazon Keyspaces en la lista de políticas y luego Siguiente.
5. En la página Nombre, revisión y creación, introduzca un nombre para el rol y revise las secciones Seleccionar entidades de confianza y Añadir permisos. También puede añadir etiquetas opcionales para el rol en esta página. Cuando haya terminado, elija Crear rol. Recuerde este nombre, dado que lo necesitará cuando lance su instancia de Amazon EC2.

Para utilizar credenciales de seguridad temporales en código, debe llamar mediante programación a una API de AWS Security Token Service, como `AssumeRole`, y extraer las credenciales y el token de sesión resultantes de su rol de IAM que creó en el paso anterior. A continuación, utilice esos

valores como credenciales para las llamadas posteriores a AWS. En el siguiente ejemplo se muestra un pseudocódigo sobre cómo utilizar credenciales de seguridad temporales:

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

Para ver un ejemplo que implemente credenciales temporales utilizando el controlador Python para acceder a Amazon Keyspaces, consulte ???.

Para obtener más información sobre cómo llamar a `AssumeRole`, `GetFederationToken` y otras operaciones de API, consulte la Referencia de la API de AWS Security Token Service. Para obtener información sobre cómo obtener las credenciales de seguridad temporales y el token de sesión a partir del resultado, consulte la documentación para el SDK con el que está trabajando. Encontrará la documentación para todos los SDK de AWS en la página de documentación de AWS principal, en la sección SDK y conjuntos de herramientas.

Puntos de conexión de servicio para Amazon Keyspaces

Temas

- Puertos y protocolos
- Puntos de conexión globales
- Puntos de conexión FIPS AWS GovCloud (US) Region
- Puntos de conexión de regiones de China

Puertos y protocolos

Puede acceder a Amazon Keyspaces mediante programación ejecutando un cliente `cqlsh`, con un controlador de Cassandra con licencia Apache 2.0, o mediante la AWS CLI y el SDK de AWS.

En la tabla siguiente se muestran los puertos y protocolos para los distintos mecanismos de acceso.

Acceso programático	Port	Protocolo
CQLSH	9142	TLS

Acceso programático	Port	Protocolo
Controlador de Cassandra	9142	TLS
AWS CLI	443	HTTPS
SDK de AWS	443	HTTPS

Para las conexiones TLS, Amazon Keyspaces utiliza Starfield CA para autenticarse ante al servidor. Para obtener más información, consulte the section called “Configuración manual de conexiones de `cqlsh` para TLS” o la sección Antes de comenzar de su controlador en el capítulo the section called “Utilización de un controlador de cliente de Cassandra”.

Puntos de conexión globales

Amazon Keyspaces está disponible en las siguientes Regiones de AWS. En esta tabla se muestra el punto de conexión de servicio disponible para cada región.

Nombre de la región	Región	Punto de enlace	Protocolo
US East (Ohio)	us-east-2	cassandra.us-east-2.amazonaws.com	HTTPS y TLS
US East (N. Virginia)	us-east-1	cassandra.us-east-1.amazonaws.com cassandra-fips.us-east-1.amazonaws.com	HTTPS y TLS TLS
EE. UU Oeste (Norte de California)	us-west-1	cassandra.us-west-1.amazonaws.com	HTTPS y TLS
US West (Oregon)	us-west-2	cassandra.us-west-2.amazonaws.com cassandra-fips.us-west-2.amazonaws.com	HTTPS y TLS

Nombre de la región	Región	Punto de enlace	Protocolo
			TLS
Asia Pacífico (Hong Kong)	ap-east-1	cassandra.ap-east-1.amazonaws.com	HTTPS y TLS
Asia Pacific (Mumbai)	ap-south-1	cassandra.ap-south-1.amazonaws.com	HTTPS y TLS
Asia Pacific (Seoul)	ap-northeast-2	cassandra.ap-northeast-2.amazonaws.com	HTTPS y TLS
Asia Pacific (Singapore)	ap-southeast-1	cassandra.ap-southeast-1.amazonaws.com	HTTPS y TLS
Asia Pacific (Sydney)	ap-southeast-2	cassandra.ap-southeast-2.amazonaws.com	HTTPS y TLS
Asia Pacific (Tokyo)	ap-northeast-1	cassandra.ap-northeast-1.amazonaws.com	HTTPS y TLS
Canada (Central)	ca-central-1	cassandra.ca-central-1.amazonaws.com	HTTPS y TLS
Europe (Frankfurt)	eu-central-1	cassandra.eu-central-1.amazonaws.com	HTTPS y TLS

Nombre de la región	Región	Punto de enlace	Protocolo	
Europe (Ireland)	eu-west-1	cassandra.eu-west-1.amazonaws.com	HTTPS y TLS	
Europe (London)	eu-west-2	cassandra.eu-west-2.amazonaws.com	HTTPS y TLS	
Europe (Paris)	eu-west-3	cassandra.eu-west-3.amazonaws.com	HTTPS y TLS	
Europe (Stockholm)	eu-north-1	cassandra.eu-north-1.amazonaws.com	HTTPS y TLS	
Medio Oriente (Baréin)	me-south-1	cassandra.me-south-1.amazonaws.com	HTTPS y TLS	
South America (São Paulo)	sa-east-1	cassandra.sa-east-1.amazonaws.com	HTTPS y TLS	
AWS GovCloud (Este de EE. UU.)	us-gov-este-1	cassandra.us-gov-east-1.amazonaws.com	HTTPS y TLS	
AWS GovCloud (Oeste de EE. UU.)	us-gov-oeste-1	cassandra.us-gov-west-1.amazonaws.com	HTTPS y TLS	

Puntos de conexión FIPS AWS GovCloud (US) Region

Puntos de conexión FIPS disponibles en la AWS GovCloud (US) Region. Para obtener más información, consulte Amazon Keyspaces en la Guía del usuario de AWS GovCloud (US).

Nombre de la región	Región	Punto de conexión FIPS	Protocolo
AWS GovCloud (Este de EE. UU.)	us-gov-este-1	cassandra.us-gov-east-1.amazonaws.com	HTTPS y TLS
AWS GovCloud (Oeste de EE. UU.)	us-gov-oeste-1	cassandra.us-gov-west-1.amazonaws.com	HTTPS y TLS

Puntos de conexión de regiones de China

Los siguientes puntos de enlace de Amazon Keyspaces están disponibles en las regiones de China de AWS.

Para acceder a estos puntos de enlace, debe suscribirse a un conjunto independiente de credenciales de cuenta exclusivo para las regiones de China. Para obtener más información, consulte Registro, cuentas y credenciales en China.

Nombre de la región	Región	Punto de enlace	Protocolo
China (Pekín)	cn-north-1	cassandra.cn-north-1.amazonaws.com.cn	HTTPS y TLS
China (Ningxia)	cn-northwest-1	cassandra.cn-northwest-1.amazonaws.com.cn	HTTPS y TLS

Uso de **cqlsh** para conectarse a Amazon Keyspaces

Para conectarse a Amazon Keyspaces con `cqlsh`, puede utilizar la `cqlsh-expansion`. Se trata de un conjunto de herramientas que contiene herramientas comunes de Apache Cassandra como `cqlsh` y asistentes preconfigurados para Amazon Keyspaces que mantienen una compatibilidad total con Apache Cassandra. La `cqlsh-expansion` integra el complemento de autenticación SigV4 y le permite conectarse utilizando claves de acceso de IAM en lugar de nombre de usuario y contraseña. Solo necesita instalar los scripts de `cqlsh` para realizar una conexión y no la distribución completa de Apache Cassandra, dado que Amazon Keyspaces funciona sin servidor. Este paquete de instalación ligero incluye la `cqlsh-expansion` y los scripts clásicos de `cqlsh` que puede instalar en cualquier plataforma que admita Python.

Para obtener información general sobre `cqlsh`, consulte `cqlsh: el intérprete de comandos CQL`.

Temas

- Uso de la cqlsh-expansion para conectarse a Amazon Keyspaces
- Configuración manual de conexiones de cqlsh para TLS

Uso de la **cqlsh-expansion** para conectarse a Amazon Keyspaces

Instalación y configuración de la **cqlsh-expansion**

1. Para instalar el paquete `cqlsh-expansion` para Python, puede ejecutar un comando `pip`. Este instala los scripts de `cqlsh-expansion` en su máquina utilizando un pip install junto con un archivo que contiene una lista de dependencias. La `--user flag` le dice a `pip` que utilice el directorio de instalación del usuario de Python para su plataforma. En un sistema basado en Unix, ese debe ser el directorio `~/.local/`.

   Necesita Python 3 para instalar la `cqlsh-expansion`; a fin de averiguar su versión de Python, utilice `Python --version`. Para realizar la instalación, puede ejecutar uno de los siguientes comandos.

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

   ```
   pip3 install --user cqlsh-expansion
   ```

   La salida debería ser similar a la siguiente.

```
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
```

Si el directorio de instalación no está en elPATH, debe añadirlo siguiendo las instrucciones de su sistema operativo. A continuación se muestra un ejemplo de Ubuntu Linux.

```
export PATH DEFAULT=${PATH}:/home/ubuntu/.local/bin
```

Para confirmar que el paquete está instalado, puede ejecutar el siguiente comando.

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

La salida debería tener este aspecto.

```
cqlsh 6.1.0
```

2. Para configurar la `cqlsh-expansion`, puede ejecutar un script post-instalación para completar automáticamente los siguientes pasos:

   1. Cree el directorio `.cassandra` en el directorio de inicio del usuario si aún no existe.
   2. Copie un archivo de configuración `cqlshrc` preconfigurado en el directorio `.cassandra`.
   3. Copie el certificado digital Starfield en el directorio `.cassandra`. Amazon Keyspaces utiliza este certificado para configurar la conexión segura con Seguridad de la capa de transporte (TLS). El cifrado en tránsito proporciona una capa adicional de protección de datos al cifrar sus datos mientras viajan hacia y desde Amazon Keyspaces.

   Para revisar antes el script, puede acceder al mismo en el repositorio de Github en post_install.py.

   Para utilizar el script, puede ejecutar el siguiente comando.

   ```
   cqlsh-expansion.init
   ```

> **Note**
>
> El directorio y el archivo creados por el script post-instalación no se eliminan al desinstalar la `cqlsh-expansion` utilizando `pip uninstall` por lo que se tienen que eliminar manualmente.

Conexión a Amazon Keyspaces mediante la **`cqlsh-expansion`**

1. Configure su variable de entorno de usuario Región de AWS y agréguela como variable de entorno de usuario.

   Para añadir su región predeterminada como variable de entorno en un sistema basado en Unix, puede ejecutar el siguiente comando. Para este ejemplo, utilizaremos Este de EE. UU. (Norte de Virginia).

   ```
   export AWS_DEFAULT_REGION=us-east-1
   ```

   Para obtener más información sobre cómo establecer variables de entorno, incluso para otras plataformas, consulte Establecimiento de variables de entorno.

2. Encuentre el punto de conexión de servicio.

   Elija el punto de conexión de servicio apropiado para su región. Para revisar los puntos de conexión disponibles para Amazon Keyspaces, consulte the section called “Puntos de conexión de servicio”. En este ejemplo, utilizamos el punto de conexión `cassandra.us-east-1.amazonaws.com`.

3. Configure el método de autenticación.

   Para mejorar la seguridad, el método recomendado es conectarse con claves de acceso de IAM (usuarios, roles e identidades federadas de IAM).

   Para poder conectarse con claves de acceso de IAM, debe completar los siguientes pasos:

   a. Cree un usuario de IAM, u observe la práctica recomendada y cree un rol de IAM que los usuarios de IAM puedan asumir. Para obtener más información sobre cómo crear claves de acceso de IAM, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

b. Cree una política de IAM que conceda al rol (o usuario de IAM) al menos acceso de solo lectura a Amazon Keyspaces. Para obtener más información sobre los permisos necesarios para que el usuario o rol de IAM se conecten a Amazon Keyspaces, consulte the section called “Acceso a las tablas de Amazon Keyspaces”.

c. Añada las claves de acceso del usuario de IAM a las variables de entorno del usuario como se muestra en el siguiente ejemplo.

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Para obtener más información sobre cómo establecer variables de entorno, incluso para otras plataformas, consulte Establecimiento de variables de entorno.

> **Note**
>
> Si se conecta desde una instancia de Amazon EC2, también debe configurar una regla de salida en el grupo de seguridad que permita el tráfico desde la instancia a Amazon Keyspaces. Para obtener más información sobre cómo ver y editar las reglas de salida de EC2, consulte Adición de reglas a un grupo de seguridad en la Guía del usuario de Amazon EC2 para instancias Linux.

4. Conéctese a Amazon Keyspaces utilizando la `cqlsh-expansion` y la autenticación SigV4.

Para conectarse a Amazon Keyspaces con la `cqlsh-expansion`, puede utilizar el siguiente comando. Asegúrese de sustituir el punto de conexión de servicio por el punto de conexión correcto para su región.

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

Si la conexión tiene éxito, debería ver una salida similar a la del siguiente ejemplo.

```
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>
```

Si se produce un error de conexión, consulte the section called “Errores de conexión con cqlsh” para obtener información sobre la solución de problemas.

- Conéctese a Amazon Keyspaces con credenciales específicas del servicio.

  Para conectarse con la combinación tradicional de nombre de usuario y contraseña que Cassandra utiliza para autenticación, primero debe crear credenciales específicas del servicio para Amazon Keyspaces como se describe en the section called “Credenciales específicas del servicio”. También tiene que dar a ese usuario permisos para acceder a Amazon Keyspaces. Para obtener más información, consulte the section called “Acceso a las tablas de Amazon Keyspaces”.

  Una vez creadas las credenciales específicas del servicio y los permisos para el usuario, debe actualizar el archivo `cqlshrc`, que suele encontrarse en la ruta del directorio de usuario `~/.cassandra/`. En el archivo `cqlshrc`, vaya a la sección `[authentication]` de Cassandra y comente el módulo SigV4 y la clase bajo `[auth_provider]` utilizando el carácter ";", como se muestra en el siguiente ejemplo.

  ```
  [auth_provider]

  ; module = cassandra_sigv4.auth

  ; classname = SigV4AuthProvider
  ```

  Una vez actualizado el archivo `cqlshrc`, puede conectarse a Amazon Keyspaces con las credenciales específicas del servicio mediante el siguiente comando.

  ```
  cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 -u myUserName -
  p myPassword --ssl
  ```

Limpieza

- Para eliminar el paquete de `cqlsh-expansion`, puede utilizar el comando `pip uninstall`.

  ```
  pip3 uninstall cqlsh-expansion
  ```

El comando `pip3 uninstall` no elimina el directorio ni los archivos relacionados creados por el script post-instalación. Para eliminar el directorio y los archivos creados por el script post-instalación, puede eliminar el directorio `.cassandra`.

Configuración manual de conexiones de **`cqlsh`** para TLS

Amazon Keyspaces solo acepta conexiones seguras utilizando Seguridad de la capa de transporte (TLS). Puede utilizar la utilidad `cqlsh-expansion` que descarga automáticamente el certificado por usted e instala un archivo de configuración `cqlshrc` preconfigurado. Para obtener más información, consulte the section called “Uso de la `cqlsh-expansion`” en esta página.

Si desea descargar el certificado y configurar la conexión manualmente, siga estos pasos.

1. Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces correctamente. El certificado de Starfield proporciona compatibilidad retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.

2. Abra el archivo de configuración `cqlshrc` en el directorio de inicio de Cassandra, por ejemplo `${HOME}/.cassandra/cqlshrc`, y añada las siguientes líneas.

```
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory

[ssl]
validate = true
certfile =  path_to_file/sf-class2-root.crt
```

Uso de AWS CLI

Puede usar la AWS Command Line Interface (AWS CLI) para controlar varios servicios de AWS desde la línea de comandos y automatizarlos mediante scripts. Con Amazon Keyspaces puede utilizar la AWS CLI para operaciones de lenguaje de definición de datos (DDL), como la de creación de una tabla. Además, puede utilizar herramientas y servicios de infraestructura como código (IaC), como AWS CloudFormation y Terraform.

Para poder utilizar la AWS CLI con Amazon Keyspaces, debe obtener un ID de clave de acceso y una clave de acceso secreta. Para obtener más información, consulte the section called "Credenciales de IAM para autenticación de AWS.".

Para obtener una lista completa de todos los comandos disponibles para Amazon Keyspaces en la AWS CLI, consulte la Referencia de comandos de AWS CLI.

Temas

- Descarga y configuración de la AWS CLI
- Uso de la AWS CLI con Amazon Keyspaces

Descarga y configuración de la AWS CLI

La AWS CLI está disponible en https://aws.amazon.com/cli. Se ejecuta en Windows, macOS o Linux. Después de descargar la AWS CLI, siga estos pasos para instalarla y configurarla:

1. Vaya a la Guía del usuario de AWS Command Line Interface.
2. Siga las instrucciones indicadas en Instalación de la AWS CLI y en Configuración de la AWS CLI

Uso de la AWS CLI con Amazon Keyspaces

El formato de la línea de comandos consiste en un nombre de operación de Amazon Keyspaces seguido de los parámetros para dicha operación. La AWS CLI admite la sintaxis abreviada de los valores de los parámetros, además de JSON. En los siguientes ejemplos de Amazon Keyspaces se utiliza la sintaxis abreviada AWS CLI. Para obtener más información, consulte Uso de la sintaxis abreviada con la CLI de AWS.

El siguiente comando crea un espacio de claves con el nombre catalog.

```
aws keyspaces create-keyspace --keyspace-name 'catalog'
```

El comando devuelve el nombre de recurso de Amazon (ARN) en la salida.

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/"
}
```

Para confirmar que el espacio de claves catalog existe, puede utilizar el siguiente comando.

```
aws keyspaces get-keyspace --keyspace-name 'catalog'
```

La salida del comando devuelve los siguientes valores.

```
{
    "keyspaceName": "catalog",
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/"
}
```

El siguiente comando crea una tabla con el nombre book_awards. La clave de partición de la tabla está formada por las columnas `year` y `award` y la clave de agrupación está formada por las columnas `category` y `rank`; ambas columnas de agrupación utilizan el orden de clasificación ascendente. Para facilitar la legibilidad, los comandos largos de esta sección se dividen en líneas separadas.

```
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}]'
```

Este comando da como resultado la siguiente salida.

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/catalog/table/
book_awards"
}
```

Para confirmar los metadatos y las propiedades de la tabla, puede utilizar el siguiente comando.

```
aws keyspaces get-table --keyspace-name 'catalog' --table-name 'book_awards'
```

Este comando devuelve la siguiente salida.

```
{
    "keyspaceName": "catalog",
    "tableName": "book_awards",
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/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": ""
    }
}
```

Al crear tablas con esquemas complejos, puede resultar útil cargar la definición del esquema de la tabla desde un archivo JSON. A continuación se muestra un ejemplo. Descargue el archivo JSON de ejemplo de definición de esquema de schema_definition.zip y extraiga `schema_definition.json`, tomando nota de la ruta de acceso al archivo. En este ejemplo, el archivo JSON de definición del

esquema se encuentra en el directorio actual. Para conocer las distintas opciones de ruta del archivo, consulte Carga de parámetros desde un archivo.

```
aws keyspaces create-table --keyspace-name 'catalog'
            --table-name 'book_awards' --schema-definition 'file://
schema_definition.json'
```

En los siguientes ejemplos se muestra cómo crear una tabla simple con el nombre myTable con opciones adicionales. Tenga en cuenta que los comandos están divididos en filas separadas para mejorar la legibilidad. Este comando muestra la forma de crear una tabla y:

- establecer el modo de capacidad de la tabla
- habilitar la recuperación en un momento dado para la tabla
- fijar el valor predeterminado de periodo de vida (TTL) de la tabla en un año
- añadir dos etiquetas para la tabla

```
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'
```

En este ejemplo se muestra cómo crear una nueva tabla que utilice una clave administrada por el cliente para el cifrado y tenga habilitado TTL a fin de permitirle establecer fechas de caducidad en columnas y filas. Para ejecutar este ejemplo, debe sustituir el ARN del recurso para la clave administrada por el cliente de AWS KMS por su propia clave y asegurarse de que Amazon Keyspaces tenga acceso a ella.

```
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-1:111222333444:key/11111111-2222-3333-4444-555555555555'
            --ttl 'status=ENABLED'
```

Uso de la API

Puede utilizar el SDK de AWS y la AWS Command Line Interface (AWS CLI) para trabajar de forma interactiva con Amazon Keyspaces. Puede utilizar la API para operaciones de lenguaje de definición de datos (DDL), como la de creación de un espacio de claves o una tabla. Además, puede utilizar herramientas y servicios de infraestructura como código (IaC), como AWS CloudFormation y Terraform.

Para poder utilizar la AWS CLI con Amazon Keyspaces, debe obtener un ID de clave de acceso y una clave de acceso secreta. Para obtener más información, consulte the section called "Credenciales de IAM para autenticación de AWS.".

Para obtener una lista completa de todas las operaciones disponibles para Amazon Keyspaces en la API, consulte Referencia de la API de Amazon Keyspaces.

Uso de Amazon Keyspaces con un SDK AWS

AWS Los kits de desarrollo de software (SDK) están disponibles para muchos lenguajes de programación populares. Cada SDK proporciona una API, ejemplos de código y documentación que facilitan a los desarrolladores la creación de aplicaciones en su lenguaje preferido.

Documentación de SDK	Ejemplos de código
AWS SDK for C++	AWS SDK for C++ ejemplos de código
AWS SDK for Go	AWS SDK for Go ejemplos de código
AWS SDK for Java	AWS SDK for Java ejemplos de código
AWS SDK for JavaScript	AWS SDK for JavaScript ejemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin ejemplos de código
AWS SDK for .NET	AWS SDK for .NET ejemplos de código
AWS SDK for PHP	AWS SDK for PHP ejemplos de código
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) ejemplos de código

Documentación de SDK	Ejemplos de código
AWS SDK for Ruby	AWS SDK for Ruby ejemplos de código
AWS SDK para Rust	AWS SDK para Rust ejemplos de código
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP ejemplos de código
AWS SDK para Swift	AWS SDK para Swift ejemplos de código

> **Ejemplo de disponibilidad**
>
> ¿No encuentra lo que necesita? Solicite un ejemplo de código a través del enlace de Provide feedback (Enviar comentarios) que se encuentra al final de esta página.

Uso de un controlador de cliente de Cassandra para acceder a Amazon Keyspaces mediante programación

Puede utilizar muchos controladores de Cassandra de terceros y de código abierto para conectarse a Amazon Keyspaces. Amazon Keyspaces es compatible con controladores de Cassandra que admiten Apache Cassandra versión 3.11.2. Para obtener más información sobre los controladores de Cassandra, consulte Controladores de cliente de Apache Cassandra.

> **Note**
>
> Para ayudarle a empezar, puede ver y descargar ejemplos de end-to-end código que establecen conexiones con Amazon Keyspaces con controladores populares. Consulte los ejemplos de Amazon Keyspaces en. GitHub

Los tutoriales de este capítulo incluyen una sencilla consulta CQL para confirmar que la conexión con Amazon Keyspaces se ha establecido correctamente. Para obtener información sobre cómo trabajar con espacios de claves y tablas después de conectarse a un punto de conexión de Amazon Keyspaces, consulte Referencia del lenguaje CQL. Para ver un step-by-step tutorial que muestra cómo conectarse a Amazon Keyspaces desde un punto de enlace de Amazon VPC, consulte. the section called “Conexión con puntos finales de VPC”

Temas

- Uso de un controlador de cliente Java de Cassandra para acceder mediante programación a Amazon Keyspaces
- Uso de un controlador de cliente Python para Cassandra para acceder mediante programación a Amazon Keyspaces
- Uso de un controlador de cliente Node.js para Cassandra para acceder mediante programación a Amazon Keyspaces
- Uso de un controlador de cliente .NET Core para Cassandra para acceder a Amazon Keyspaces mediante programación
- Uso de un controlador de cliente Go para Cassandra para acceder a Amazon Keyspaces mediante programación
- Uso de un controlador de cliente Perl para Cassandra para acceder a Amazon Keyspaces mediante programación

Uso de un controlador de cliente Java de Cassandra para acceder mediante programación a Amazon Keyspaces

En esta sección le mostramos cómo conectarse a Amazon Keyspaces mediante un controlador de cliente Java.

> **Note**
>
> Actualmente, Java 17 y el controlador DataStax Java 4.17 solo son compatibles con la versión beta. Para obtener más información, consulte https://docs.datastax.com/en/developer/java-driver/4.17/upgrade_guide/.

Para proporcionar a los usuarios y a las aplicaciones credenciales de acceso programático a los recursos de Amazon Keyspaces, puede realizar cualquiera de las siguientes acciones:

- Cree credenciales específicas del servicio que se vinculen a un usuario de AWS Identity and Access Management (IAM) específico.
- Para mejorar la seguridad, se recomienda crear claves de acceso de IAM para las identidades de IAM que se utilizan en todos los servicios. AWS El complemento de autenticación SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra habilita la autenticación de llamadas a Amazon Keyspaces utilizando claves de acceso de IAM en vez de nombre de usuario y

contraseña. Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

> **Note**
>
> Para ver un ejemplo de cómo utilizar Amazon Keyspaces con Spring Boot, consulte https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring.

Temas

- Antes de empezar
- Un tep-by-step tutorial para conectarse a Amazon Keyspaces mediante el controlador DataStax Java para Apache Cassandra con credenciales específicas del servicio
- Un tep-by-step tutorial para conectarse a Amazon Keyspaces mediante el controlador DataStax Java 4.x para Apache Cassandra y el complemento de autenticación SigV4
- Conéctese a Amazon Keyspaces mediante el controlador DataStax Java 3.x para Apache Cassandra y el complemento de autenticación SigV4

Antes de empezar

Para conectarse a Amazon Keyspaces, debe completar las siguientes tareas antes de poder comenzar.

1. Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes.

   a. Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

   ```
   curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
   ```

   > **Note**
   >
   > También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces correctamente. El certificado de Starfield proporciona compatibilidad

retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.

b. Convierta el certificado digital de Starfield en un archivo trustStore.

```
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
```

En este paso, debe crear una contraseña para el almacén de claves y dar confianza a este certificado. El comando interactivo tiene el siguiente aspecto.

```
Enter keystore password:
Re-enter new 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:
  MD5:  32:4A:4B:BB:C8:63:69:9B:BE:74:9A:C6:DD:1D:46:24
  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
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                                        ....
]
]
Trust this certificate? [no]:  y
```

2. Adjunte el archivo trustStore en los argumentos de la JVM:

```
-Djavax.net.ssl.trustStore=path_to_file/cassandra_truststore.jks
-Djavax.net.ssl.trustStorePassword=my_password
```

Un tep-by-step tutorial para conectarse a Amazon Keyspaces mediante el controlador DataStax Java para Apache Cassandra con credenciales específicas del servicio

En el siguiente step-by-step tutorial, se explica cómo conectarse a Amazon Keyspaces mediante un controlador Java para Cassandra con credenciales específicas del servicio. En concreto, utilizará la versión 4.0 del controlador de DataStax Java para Apache Cassandra.

Temas

- Paso 1: Requisitos previos
- Paso 2: Configurar el controlador
- Paso 3: Ejecutar el ejemplo de aplicación

Paso 1: Requisitos previos

Para seguir este tutorial, debe generar credenciales específicas del servicio y añadir el controlador DataStax Java para Apache Cassandra a su proyecto de Java.

- Genere credenciales específicas del servicio para su usuario de IAM de Amazon Keyspaces; para ello, siga los pasos que se indican en the section called “Credenciales específicas del servicio”. Si prefiere utilizar claves de acceso de IAM para la autenticación, consulte the section called “Complemento de autenticación para Java 4.x”.

- Añada el controlador de DataStax Java para Apache Cassandra a su proyecto de Java. Asegúrese de estar utilizando una versión de controlador compatible con Apache Cassandra 3.11.2. Para obtener más información, consulte la documentación del controlador DataStax Java para Apache Cassandra.

Paso 2: Configurar el controlador

Puede especificar los ajustes del controlador DataStax Java Cassandra creando un archivo de configuración para su aplicación. Este archivo de configuración anula los ajustes predeterminados e indica al controlador que se conecte al punto de conexión del servicio de Amazon Keyspaces a través del puerto 9142. Para obtener una lista de los puntos de conexión de servicio disponibles, consulte the section called “Puntos de conexión de servicio”.

Cree un archivo de configuración y guárdelo en la carpeta de recursos de la aplicación; por ejemplo, `src/main/resources/application.conf`. Abra `application.conf` y añada los siguientes ajustes de configuración.

1. Proveedor de autenticación: cree el proveedor de autenticación con la clase `PlainTextAuthProvider`. *`ServiceUserName`* y *`ServicePassword`* deben coincidir con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio siguiendo los pasos que se indican en. Generación de credenciales específicas del servicio

   > **Note**
   >
   > Puede usar credenciales de corta duración mediante el complemento de autenticación del controlador DataStax Java para Apache Cassandra en lugar de codificar las credenciales en el archivo de configuración del controlador. Para obtener más información, siga las instrucciones de the section called “Complemento de autenticación para Java 4.x”.

2. Centro de datos local: establezca el valor de `local-datacenter` a la región a la que se conecte. Por ejemplo, si la aplicación se conecta a `cassandra.us-east-2.amazonaws.com`, entonces establezca el centro de datos local en `us-east-2`. Para conocer todas las Regiones de AWS disponibles, consulte ???. Establezca `slow-replica-avoidance = false` para equilibrar la carga con menos nodos.
3. SSL/TLS: para inicializar el SSL, EngineFactory añada una sección en el archivo de configuración con una sola línea que especifique la clase con la que se especifica. `class =`

`DefaultSslEngineFactory` Proporcione la ruta al archivo trustStore y la contraseña que creó anteriormente. Amazon Keyspaces no admite `hostname-validation` de pares, por lo que debe establecer esta opción en 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**
>
> En vez de añadir la ruta al trustStore en el archivo de configuración, también puede añadir la ruta al trustStore directamente en el código de la aplicación o a sus argumentos de la JVM.

Paso 3: Ejecutar el ejemplo de aplicación

Este ejemplo de código muestra una sencilla aplicación de línea de comandos que crea un grupo de conexión a Amazon Keyspaces utilizando el archivo de configuración se ha creado anteriormente. Confirma que se ha establecido la conexión ejecutando una consulta sencilla.

```
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**
>
> Utilice un bloque `try` para establecer la conexión a fin de garantizar que siempre esté cerrada. Si no utilizas un bloque `try`, recuerde cerrar su conexión para evitar fugas de recursos.

Un tep-by-step tutorial para conectarse a Amazon Keyspaces mediante el controlador DataStax Java 4.x para Apache Cassandra y el complemento de autenticación SigV4

En la siguiente sección se describe cómo utilizar el complemento de autenticación SigV4 para el controlador DataStax Java 4.x de código abierto para Apache Cassandra a fin de acceder a Amazon Keyspaces (para Apache Cassandra). El complemento está disponible en el repositorio. GitHub

El complemento de autenticación SigV4 le permite utilizar credenciales de IAM para usuarios o roles al conectarse a Amazon Keyspaces. En vez de requerir un nombre de usuario y una contraseña, este complemento firma las solicitudes de la API utilizando claves de acceso. Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Paso 1: Requisitos previos

Para seguir este tutorial, necesita completar las siguientes tareas.

- Si aún no lo ha hecho, cree credenciales para su usuario o rol de IAM; para ello, siga los pasos que se indican en the section called “Credenciales de IAM para autenticación de AWS.”. En este tutorial se asume que las claves de acceso se almacenan como variables de entorno. Para obtener más información, consulte the section called “Administración de claves de acceso”.
- Añada el controlador DataStax Java para Apache Cassandra a su proyecto Java. Asegúrese de estar utilizando una versión de controlador compatible con Apache Cassandra 3.11.2. Para obtener más información, consulte la documentación del controlador DataStax Java para Apache Cassandra.
- Añada el complemento de autenticación a su aplicación. El complemento de autenticación es compatible con la versión 4.x del controlador DataStax Java para Apache Cassandra. Si utiliza Apache Maven, o un sistema de compilación que pueda utilizar dependencias de Maven, añada las siguientes dependencias a su archivo `pom.xml`.

> **Important**
>
> Sustituya la versión del complemento por la versión más reciente, tal y como se muestra en GitHub el repositorio.

```
<dependency>
        <groupId>software.aws.mcs</groupId>
        <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin</artifactId>
        <version>4.0.9</version>
</dependency>
```

Paso 2: Configurar el controlador

Puede especificar los ajustes del controlador DataStax Java Cassandra creando un archivo de configuración para su aplicación. Este archivo de configuración anula los ajustes predeterminados e indica al controlador que se conecte al punto de conexión del servicio de Amazon Keyspaces a través del puerto 9142. Para obtener una lista de los puntos de conexión de servicio disponibles, consulte the section called “Puntos de conexión de servicio”.

Cree un archivo de configuración y guárdelo en la carpeta de recursos de la aplicación; por ejemplo, `src/main/resources/application.conf`. Abra `application.conf` y añada los siguientes ajustes de configuración.

1. Proveedor de autenticación: establezca la `advanced.auth-provider.class` a una nueva instancia de `software.aws.mcs.auth.SigV4AuthProvider`. El SigV4 AuthProvider es el controlador de autenticación que proporciona el complemento para realizar la autenticación SigV4.
2. Centro de datos local: establezca el valor de `local-datacenter` a la región a la que se conecte. Por ejemplo, si la aplicación se conecta a `cassandra.us-east-2.amazonaws.com`, entonces establezca el centro de datos local en `us-east-2`. Para conocer todas las Regiones de AWS disponibles, consulte ???. Establezca `slow-replica-avoidance = false` para equilibrar la carga con menos nodos.
3. SSL/TLS: para inicializar el SSL, EngineFactory añada una sección en el archivo de configuración con una sola línea que especifique la clase con la que se especifica. `class = DefaultSslEngineFactory` Proporcione la ruta al archivo trustStore y la contraseña que creó anteriormente. Amazon Keyspaces no admite `hostname-validation` de pares, por lo que debe establecer esta opción en false.

```
datastax-java-driver {
        basic.contact-points = ["cassandra.us-east-2.amazonaws.com:9142"]
        basic.load-balancing-policy {
            class = DefaultLoadBalancingPolicy
            local-datacenter = us-east-2
            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
            }
        }
```

```
    }
```

> **Note**
>
> En vez de añadir la ruta al trustStore en el archivo de configuración, también puede añadir la ruta al trustStore directamente en el código de la aplicación o a sus argumentos de la JVM.

Paso 3: Ejecute la aplicación

Este ejemplo de código muestra una sencilla aplicación de línea de comandos que crea un grupo de conexión a Amazon Keyspaces utilizando el archivo de configuración se ha creado anteriormente. Confirma que se ha establecido la conexión ejecutando una consulta sencilla.

```
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**
>
> Utilice un bloque `try` para establecer la conexión a fin de garantizar que siempre esté cerrada. Si no utilizas un bloque `try`, recuerde cerrar su conexión para evitar fugas de recursos.

Conéctese a Amazon Keyspaces mediante el controlador DataStax Java 3.x para Apache Cassandra y el complemento de autenticación SigV4

En la siguiente sección, se describe cómo utilizar el complemento de autenticación SigV4 para el controlador DataStax Java 3.x de código abierto para que Apache Cassandra acceda a Amazon Keyspaces. El complemento está disponible en el repositorio. GitHub

El complemento de autenticación SigV4 le permite utilizar credenciales de IAM para usuarios o roles al conectarse a Amazon Keyspaces. En vez de requerir un nombre de usuario y una contraseña, este complemento firma las solicitudes de la API utilizando claves de acceso. Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Paso 1: Requisitos previos

Para ejecutar este código de ejemplo, primero debe completar las siguientes tareas.

- Cree credenciales para su usuario o rol de IAM; para ello, siga los pasos indicados en the section called “Credenciales de IAM para autenticación de AWS.”. En este tutorial se asume que las claves de acceso se almacenan como variables de entorno. Para obtener más información, consulte the section called “Administración de claves de acceso”.
- Siga los pasos indicados en the section called “Antes de empezar” para descargar el certificado digital de Starfield, convertirlo en un archivo trustStore y adjuntar el archivo trustStore en los argumentos de la JVM de su aplicación.
- Añada el controlador DataStax Java para Apache Cassandra a su proyecto Java. Asegúrese de estar utilizando una versión de controlador compatible con Apache Cassandra 3.11.2. Para obtener más información, consulte la documentación del controlador DataStax Java para Apache Cassandra.
- Añada el complemento de autenticación a su aplicación. El complemento de autenticación es compatible con la versión 3.x del controlador DataStax Java para Apache Cassandra. Si utiliza Apache Maven, o un sistema de compilación que pueda utilizar dependencias de Maven, añada

las siguientes dependencias a su archivo `pom.xml`. Sustituya la versión del complemento por la versión más reciente, tal y como se muestra en GitHub el repositorio.

```
<dependency>
        <groupId>software.aws.mcs</groupId>
        <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin_3</artifactId>
        <version>3.0.3</version>
</dependency>
```

Paso 2: Ejecute la aplicación

Este ejemplo de código muestra una sencilla aplicación de línea de comandos que crea un grupo de conexiones a Amazon Keyspaces. Confirma que se ha establecido la conexión ejecutando una consulta sencilla.

```
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"));
    }
}
```

Notas de uso:

Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Consulte el siguiente repositorio para obtener políticas, ejemplos y prácticas recomendadas útiles al utilizar el controlador Java con Amazon Keyspaces: https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers.

Uso de un controlador de cliente Python para Cassandra para acceder mediante programación a Amazon Keyspaces

En esta sección, le mostramos cómo conectarse a Amazon Keyspaces utilizando un controlador de cliente Python. Para proporcionar a los usuarios y a las aplicaciones credenciales de acceso programático a los recursos de Amazon Keyspaces, puede realizar cualquiera de las siguientes acciones:

- Cree credenciales específicas del servicio que se vinculen a un usuario de AWS Identity and Access Management (IAM) específico.
- Para mejorar la seguridad, recomendamos crear claves de acceso de IAM para los usuarios o roles de IAM que se utilicen en todos los AWS servicios. El complemento de autenticación SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra habilita la autenticación de llamadas a Amazon Keyspaces utilizando claves de acceso de IAM en vez de nombre de usuario y contraseña. Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Temas

- Antes de empezar
- Conexión a Amazon Keyspaces mediante el controlador Python para Apache Cassandra y credenciales específicas del servicio
- Conéctese a Amazon Keyspaces mediante el controlador de DataStax Python para Apache Cassandra y el complemento de autenticación SigV4

Antes de empezar

Debe completar la siguiente tarea antes de comenzar.

Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes. Para conectarse a Amazon Keyspaces mediante TLS, necesita descargar un certificado digital de Amazon y configurar el controlador Python para utilizar TLS.

Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -0
```

> **Note**
>
> También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces correctamente. El certificado de Starfield proporciona compatibilidad retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.


```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -0
```


Conexión a Amazon Keyspaces mediante el controlador Python para Apache Cassandra y credenciales específicas del servicio

En el siguiente ejemplo de código se muestra cómo conectarse a Amazon Keyspaces con un controlador de cliente Python y credenciales específicas del servicio.

```
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/sf-class2-root.crt')
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)
```

Notas de uso:

1. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.
2. Asegúrese de que *ServicePassword*coincidan con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio siguiendo los pasos que se indican a continuación. *ServiceUserName*Generación de credenciales específicas del servicio
3. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Conéctese a Amazon Keyspaces mediante el controlador de DataStax Python para Apache Cassandra y el complemento de autenticación SigV4

En la siguiente sección, se muestra cómo utilizar el complemento de autenticación SigV4 para que el controlador DataStax Python de código abierto para Apache Cassandra acceda a Amazon Keyspaces (para Apache Cassandra).

Si aún no lo ha hecho, empiece creando credenciales para su rol de IAM; para ello, siga los pasos que se indican en the section called “Credenciales de IAM para autenticación de AWS.”. En este tutorial se utilizan credenciales temporales que requieren un rol de IAM. Para obtener más información sobre credenciales temporales, consulte the section called “Uso de credenciales temporales para conectarse a Amazon Keyspaces”.

A continuación, añada el complemento de autenticación SigV4 de Python a su entorno desde el GitHub repositorio.

```
pip install cassandra-sigv4
```

El siguiente ejemplo de código muestra cómo conectarse a Amazon Keyspaces mediante el controlador DataStax Python de código abierto para Cassandra y el complemento de autenticación SigV4. El plugin depende del AWS SDK para Python (Boto3). Utiliza `boto3.session` para obtener credenciales temporales.

```
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/sf-class2-root.crt')
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)
```

Notas de uso:

1. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.
2. Asegúrese de que *aws_access_key_id*, *aws_secret_access_key* y *aws_session_token* coincidan con los `Access Key`, `Secret Access Key` y `Session Token` que obtuvo al utilizar `boto3.session`. Para obtener más información, consulte la sección sobre credenciales en la AWS SDK for Python (Boto3).
3. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Uso de un controlador de cliente Node.js para Cassandra para acceder mediante programación a Amazon Keyspaces

En esta sección le mostramos cómo conectarse a Amazon Keyspaces mediante un controlador de cliente Node.js. Para proporcionar a los usuarios y a las aplicaciones credenciales de acceso programático a los recursos de Amazon Keyspaces, puede realizar cualquiera de las siguientes acciones:

- Cree credenciales específicas del servicio que se vinculen a un usuario de AWS Identity and Access Management (IAM) específico.
- Para mejorar la seguridad, recomendamos crear claves de acceso de IAM para los usuarios o roles de IAM que se utilicen en todos los servicios. AWS El complemento de autenticación SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra habilita la autenticación de llamadas a Amazon Keyspaces utilizando claves de acceso de IAM en vez de nombre de usuario y contraseña. Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Temas

- Antes de empezar
- Conéctese a Amazon Keyspaces mediante el DataStax controlador Node.js para Apache Cassandra y credenciales específicas del servicio
- Conéctese a Amazon Keyspaces mediante el controlador DataStax Node.js para Apache Cassandra y el complemento de autenticación SigV4

Antes de empezar

Debe completar la siguiente tarea antes de comenzar.

Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes. Para conectarse a Amazon Keyspaces mediante TLS, necesita descargar un certificado digital de Amazon y configurar el controlador Python para utilizar TLS.

Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces correctamente. El certificado de Starfield proporciona compatibilidad retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Conéctese a Amazon Keyspaces mediante el DataStax controlador Node.js para Apache Cassandra y credenciales específicas del servicio

Configure su controlador para que utilice el certificado digital de Starfield para TLS y se autentique con credenciales específicas del servicio. Por ejemplo:

```
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/sf-class2-root.crt', '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}`));
```

Notas de uso:

1. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.
2. Asegúrese de que *ServicePassword*coincidan con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio siguiendo los pasos que se indican a continuación. *ServiceUserName*Generación de credenciales específicas del servicio
3. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Conéctese a Amazon Keyspaces mediante el controlador DataStax Node.js para Apache Cassandra y el complemento de autenticación SigV4

En la siguiente sección, se muestra cómo utilizar el complemento de autenticación SigV4 para que el controlador DataStax Node.js de código abierto de Apache Cassandra acceda a Amazon Keyspaces (para Apache Cassandra).

Si aún no lo ha hecho, cree credenciales para su usuario o rol de IAM; para ello, siga los pasos que se indican en the section called “Credenciales de IAM para autenticación de AWS.”.

Agregue el complemento de autenticación SigV4 de Node.js a su aplicación desde el repositorio. GitHub El complemento es compatible con la versión 4.x del controlador DataStax Node.js para Cassandra y depende del AWS SDK para Node.js. Utiliza `AWSCredentialsProvider` para obtener credenciales.

```
$ npm install aws-sigv4-auth-cassandra-plugin --save
```

En este ejemplo de código se muestra cómo configurar una instancia específica de una región de `SigV4AuthProvider` como proveedor de autenticación.

```
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_filecassandra/sf-class2-root.crt', '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}`));
```

Notas de uso:

1. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.
2. Asegúrese de que la clave de acceso *accessKeyId*y la clave de acceso secreta *secretAccessKey*coincidan con la clave de acceso secreta que utilizó. `AWSCredentialsProvider` Para obtener más información, consulte Configuración de credenciales en Node.js en el AWS SDK para JavaScript Node.js.
3. Para almacenar las claves de acceso fuera del código, consulta las prácticas recomendadas en the section called “Administración de claves de acceso”.
4. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Uso de un controlador de cliente .NET Core para Cassandra para acceder a Amazon Keyspaces mediante programación

En esta sección le mostramos cómo conectarse a Amazon Keyspaces mediante un controlador de cliente .NET Core. Los pasos de configuración varían en función de su entorno y del sistema

operativo, por lo que es posible que tenga que modificarlos en consecuencia. Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes. Para conectarse a Amazon Keyspaces mediante TLS, necesita descargar un certificado digital de Starfield y configurar su controlador para utilizar TLS.

1. Descargue el certificado de Starfield y guárdelo en un directorio local, tomando nota de la ruta. A continuación se muestra un ejemplo que utiliza PowerShell.

```
$client = new-object System.Net.WebClient
$client.DownloadFile("https://certs.secureserver.net/repository/sf-class2-
root.crt","path_to_file\sf-class2-root.crt")
```

2. Instale el CASSANDRAC SharpDriver a través de nuget, utilizando la consola nuget.

```
PM> Install-Package CassandraCSharpDriver
```

3. En el siguiente ejemplo, se utiliza un proyecto de consola C# de.NET Core para conectarse a Amazon Keyspaces y ejecutar una consulta.

```
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)
        {
            X509Certificate2Collection certCollection = new
 X509Certificate2Collection();
            X509Certificate2 amazoncert = new X509Certificate2(@"path_to_file\sf-
class2-root.crt");
            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);
            }
        }
    }
}
```

Notas de uso:

a. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.

b. Asegúrese de que *ServiceUserName*y *ServicePassword*coincidan con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio siguiendo los pasos que se indican a continuación. Generación de credenciales específicas del servicio

c. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Uso de un controlador de cliente Go para Cassandra para acceder a Amazon Keyspaces mediante programación

En esta sección le mostramos cómo conectarse a Amazon Keyspaces mediante un controlador de cliente Go. Para proporcionar a los usuarios y a las aplicaciones credenciales de acceso

programático a los recursos de Amazon Keyspaces, puede realizar cualquiera de las siguientes acciones:

- Cree credenciales específicas del servicio que se vinculen a un usuario de AWS Identity and Access Management (IAM) específico.
- Para mejorar la seguridad, recomendamos crear claves de acceso de IAM para los usuarios y roles de IAM que se utilicen en todos los servicios. AWS El complemento de autenticación SigV4 de Amazon Keyspaces para controladores de cliente de Cassandra habilita la autenticación de llamadas a Amazon Keyspaces utilizando claves de acceso de IAM en vez de nombre de usuario y contraseña. Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Temas

- Antes de empezar
- Conexión a Amazon Keyspaces mediante el controlador Gocql para Apache Cassandra y credenciales específicas del servicio
- Conexión a Amazon Keyspaces mediante el controlador Go para Apache Cassandra y el complemento de autenticación SigV4

Antes de empezar

Debe completar la siguiente tarea antes de comenzar.

Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes. Para conectarse a Amazon Keyspaces mediante TLS, necesita descargar un certificado digital de Amazon y configurar el controlador Python para utilizar TLS.

Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces

correctamente. El certificado de Starfield proporciona compatibilidad retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -0
```

Conexión a Amazon Keyspaces mediante el controlador Gocql para Apache Cassandra y credenciales específicas del servicio

1. Cree un directorio para la aplicación.

   ```
   mkdir ./gocqlexample
   ```

2. Vaya al nuevo directorio.

   ```
   cd gocqlexample
   ```

3. Cree un archivo para su aplicación.

   ```
   touch cqlapp.go
   ```

4. Descargue el controlador Go.

   ```
   go get github.com/gocql/gocql
   ```

5. Añada el siguiente código de ejemplo al nuevo archivo.

   ```
   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 sf-class2-root.crt
    cluster.SslOpts = &gocql.SslOptions{
            CaPath: "path_to_file/sf-class2-root.crt",
            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()
}
```

Notas de uso:

a. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.

b. Asegúrese de que *ServicePassword*coincidan con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio siguiendo los pasos que se indican a continuación. *ServiceUserName*Generación de credenciales específicas del servicio

c. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

6. Compile el programa.

```
go build cqlapp.go
```

7. Ejecute el programa.

```
./cqlapp
```

Conexión a Amazon Keyspaces mediante el controlador Go para Apache Cassandra y el complemento de autenticación SigV4

El siguiente ejemplo de código muestra cómo utilizar el complemento de autenticación SigV4 para el controlador Go de código abierto para acceder a Amazon Keyspaces (para Apache Cassandra).

Si aún no lo ha hecho, cree credenciales para su usuario o rol de IAM; para ello, siga los pasos que se indican en the section called “Credenciales de IAM para autenticación de AWS.”.

Añada el complemento de autenticación Go SigV4 a su aplicación desde el repositorio. GitHub
El complemento es compatible con la versión 1.2.x del controlador Go de código abierto para Cassandra y depende del SDK de Go AWS .

```
$ go mod init
$ go get github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin
```

En este ejemplo de código, el punto de conexión de Amazon Keyspaces está representado por la clase `Cluster`. Utiliza el `AwsAuthenticator` de la propiedad de autenticador del clúster para obtener las credenciales.

```
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
```

```
    var auth sigv4.AwsAuthenticator = sigv4.NewAwsAuthenticator()
    auth.Region = "us-west-2"
    auth.AccessKeyId = "AKIAIOSFODNN7EXAMPLE"
    auth.SecretAccessKey = "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"

    cluster.Authenticator = auth

    cluster.SslOpts = &gocql.SslOptions{
            CaPath: "path_to_file/sf-class2-root.crt",
            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)
    }
}
```

Notas de uso:

1. Sustituya "*path_to_file*/sf-class2-root.crt" por la ruta al certificado guardado en el primer paso.
2. Asegúrate de que *AccessKeyId* y *SecretAccessKey* coinciden con la clave de acceso y la clave de acceso secreta que obtuviste utilizando. `AwsAuthenticator` Para obtener más información, consulte Configuración del SDK de AWS para Go en la AWS SDK for Go.
3. Para almacenar las claves de acceso fuera del código, consulta las prácticas recomendadas en the section called “Administración de claves de acceso”.

4. Para obtener una lista de los puntos de conexión disponibles, consulte the section called “Puntos de conexión de servicio”.

Uso de un controlador de cliente Perl para Cassandra para acceder a Amazon Keyspaces mediante programación

En esta sección le mostramos cómo conectarse a Amazon Keyspaces mediante un controlador de cliente Perl. En este ejemplo de código, utilizamos Perl 5. Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes.

> **Important**
>
> Para crear una conexión segura, nuestros ejemplos de código utilizan el certificado digital de Starfield para autenticar el servidor antes de establecer la conexión TLS. El controlador Perl no valida el certificado SSL de Amazon del servidor, lo que significa que no puede confirmar que se esté conectando a Amazon Keyspaces. El segundo paso, configurar el controlador para que utilice TLS al conectarse a Amazon Keyspaces, sigue siendo necesario y garantiza que los datos transferidos entre el cliente y el servidor estén cifrados.

1. Descargue el controlador DBI de Cassandra desde https://metacpan.org/pod/DBD::Cassandra e instale el controlador en su entorno Perl. Los pasos exactos dependen del entorno. El siguiente es un ejemplo común.

```
cpanm DBD::Cassandra
```

2. Cree un archivo para su aplicación.

```
touch cqlapp.pl
```

3. Añada el siguiente código de ejemplo al archivo cqlapp.

```
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**
>
> Asegúrese de que el nombre de usuario *ServiceUserName*y la contraseña que obtuvo al generar las credenciales específicas del servicio *ServicePassword*coinciden. Para ello, siga los pasos que se indican a continuación. Generación de credenciales específicas del servicio

> **Note**
>
> Para obtener una lista de los puntos de conexión disponibles, consulte the section called "Puntos de conexión de servicio".

4. Ejecute la aplicación.

```
perl cqlapp.pl
```

Tutorial: Conexión a Amazon Keyspaces desde Amazon Elastic Kubernetes Service

En este tutorial, se explican los pasos necesarios para configurar un clúster de Amazon Elastic Kubernetes Service (Amazon EKS) para alojar una aplicación en contenedores que se conecte a Amazon Keyspaces mediante la autenticación SigV4.

Amazon EKS es un servicio gestionado que elimina la necesidad de instalar, operar y mantener su propio plano de control de Kubernetes. Kubernetes es un sistema de código abierto para automatizar la administración, el escalado y la implementación de las aplicaciones en contenedores.

El tutorial proporciona step-by-step instrucciones para configurar, crear e implementar una aplicación Java en contenedores en Amazon EKS. En el último paso, ejecuta la aplicación para escribir datos en una tabla de Amazon Keyspaces.

Temas

- Requisitos previos del tutorial
- Paso 1: Configurar el clúster Amazon EKS y configurar los permisos de IAM
- Paso 2: Configure la aplicación
- Paso 3: Cree la imagen de la aplicación y cargue el archivo Docker en su repositorio de Amazon ECR
- Paso 4: Implemente la aplicación en Amazon EKS y escriba los datos en la tabla de Amazon Keyspaces
- Paso 5: Limpieza (opcional)

Requisitos previos del tutorial

Cree los siguientes AWS recursos antes de empezar con el tutorial

1. Antes de empezar este tutorial, siga las instrucciones AWS de configuración incluidas enAcceso a Amazon Keyspaces (para Apache Cassandra). Estos pasos incluyen la inscripción AWS y la creación de un director AWS Identity and Access Management (IAM) con acceso a Amazon Keyspaces.
2. Cree un espacio de claves de Amazon Keyspaces con el nombre `aws` y una tabla con el nombre en la `user` que pueda escribir desde la aplicación contenerizada que se ejecuta en Amazon EKS más adelante en este tutorial. Puede hacerlo con o utilizando. AWS CLI `cqlsh`

   AWS CLI

   ```
   aws keyspaces create-keyspace --keyspace-name 'aws'
   ```

   Para confirmar que se creó el espacio de claves, puede utilizar el siguiente comando.

   ```
   aws keyspaces list-keyspaces
   ```

   Para crear la tabla, puede utilizar el siguiente comando.

```
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}]'
```

Para confirmar que se creó la tabla, puede usar el siguiente comando.

```
aws keyspaces list-tables --keyspace-name 'aws'
```

Para obtener más información, consulte crear un espacio de claves y crear una tabla en la Referencia de AWS CLI comandos.

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
);
```

Para comprobar que la tabla se ha creado, puede utilizar la siguiente declaración.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "aws";
```

La tabla debe aparecer en el resultado de esta declaración. Tenga en cuenta que puede haber un retraso hasta que se cree la tabla. Para obtener más información, consulte the section called “CREATE TABLE”.

3. Cree un clúster de Amazon EKS con un tipo de nodo Fargate: Linux. Fargate es un motor de procesamiento sin servidor que le permite implementar Kubernetes Pods sin administrar las instancias de Amazon Amazon EC2. Para seguir este tutorial sin tener que actualizar el nombre del clúster en todos los comandos de ejemplo, cree un clúster con ese nombre `my-eks-cluster` siguiendo las instrucciones de Introducción a Amazon EKS, `eksctl` en la Guía del usuario de Amazon EKS. Cuando se cree el clúster, compruebe que los nodos y los dos

pods predeterminados estén funcionando y en buen estado. Puedes hacerlo con el siguiente comando.

```
kubectl get pods -A -o wide
```

Debería ver algo similar a este resultado.

```
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. Instalar Docker. Para obtener instrucciones sobre cómo instalar Docker en una instancia de Amazon EC2, consulte Instalar Docker en la Guía del usuario de Amazon Elastic Container Registry.

   Docker está disponible para muchos sistemas operativos diferentes, incluidas las distribuciones de Linux más modernas, como Ubuntu, e incluso en macOS y Windows. Para obtener más información sobre cómo instalar Docker en su sistema operativo concreto, consulte la guía de instalación de Docker.

5. Cree un repositorio de Amazon ECR. Amazon ECR es un servicio AWS gestionado de registro de imágenes de contenedores que puede utilizar con la CLI que prefiera para insertar, extraer y gestionar imágenes de Docker. Para obtener más información sobre los repositorios de Amazon ECR, consulte la Guía del usuario de Amazon Elastic Container Registry. Puede usar el siguiente comando para crear un repositorio con ese nombre. `my-ecr-repository`

   ```
   aws ecr create-repository --repository-name my-ecr-repository
   ```

Tras completar los pasos de requisitos previos, continúe con the section called “Paso 1: Configurar el clúster de Amazon EKS”.

Paso 1: Configurar el clúster Amazon EKS y configurar los permisos de IAM

Configure el clúster de Amazon EKS y cree los recursos de IAM necesarios para permitir que una cuenta de servicio de Amazon EKS se conecte a su tabla de Amazon Keyspaces

1. Cree un proveedor de Open ID Connect (OIDC) para el clúster de Amazon EKS. Esto es necesario para utilizar las funciones de IAM en las cuentas de servicio. Para obtener más información sobre los proveedores de OIDC y cómo crearlos, consulte Creación de un proveedor de OIDC de IAM para su clúster en la Guía del usuario de Amazon EKS.

   a. Cree un proveedor de identidad de OIDC de IAM para su clúster con el siguiente comando. En este ejemplo se supone que el nombre de su clúster es. `my-eks-cluster` Si tiene un clúster con un nombre diferente, recuerde actualizar el nombre en todos los comandos futuros.

   ```
   eksctl utils associate-iam-oidc-provider --cluster my-eks-cluster --approve
   ```

   b. Confirme que el proveedor de identidad del OIDC se haya registrado en IAM con el siguiente comando.

   ```
   aws iam list-open-id-connect-providers --region aws-region
   ```

   La salida debería ser similar a la siguiente. Toma nota del Amazon Resource Name (ARN) de la OIDC. Lo necesitarás en el siguiente paso cuando crees una política de confianza para la cuenta de servicio.

   ```
   {
       "OpenIDConnectProviderList": [
           ..
           {
               "Arn": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.aws-
   region.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
           }
       ]
   }
   ```

2. Cree una cuenta de servicio para el clúster de Amazon EKS. Las cuentas de servicio proporcionan una identidad para los procesos que se ejecutan en un pod. Un pod es el objeto de Kubernetes más pequeño y sencillo que puedes usar para implementar una aplicación en contenedores. A continuación, crea una función de IAM que la cuenta de servicio pueda asumir

para obtener permisos para acceder a los recursos. Puedes acceder a cualquier AWS servicio desde un pod que se haya configurado para usar una cuenta de servicio que pueda asumir una función de IAM con permisos de acceso a ese servicio.

a. Crea un nuevo espacio de nombres para la cuenta de servicio. Un espacio de nombres ayuda a aislar los recursos del clúster creados para este tutorial. Puede crear un nuevo espacio de nombres mediante el siguiente comando.

```
kubectl create namespace my-eks-namespace
```

b. Para usar un espacio de nombres personalizado, debe asociarlo a un perfil de Fargate. El siguiente código es un ejemplo de ello.

```
eksctl create fargateprofile \
    --cluster my-eks-cluster \
    --name my-fargate-profile \
    --namespace my-eks-namespace \
    --labels *=*
```

c. Cree una cuenta de servicio con el nombre `my-eks-serviceaccount` del espacio de nombres de su clúster `my-eks-namespace` de Amazon EKS mediante el siguiente comando.

```
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. Ejecute el siguiente comando para crear un archivo de política de confianza que indique al rol de IAM que confíe en su cuenta de servicio. Esta relación de confianza es necesaria antes de que un principal pueda asumir un rol. Debe realizar las siguientes modificaciones en el archivo:

- Para el`Principal`, introduzca el ARN que IAM devolvió al comando. `list-open-id-connect-providers` El ARN contiene su número de cuenta y su región.

- En el `condition` extracto, sustituya el identificador OIDC Región de AWS y el identificador OIDC.
- Confirme que el nombre y el espacio de nombres de la cuenta de servicio son correctos.

Debe adjuntar el archivo de política de confianza en el siguiente paso al crear el rol de IAM.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.aws-region.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.aws-region.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount",
                    "oidc.eks.aws-region.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
EOF
```

Opcional: también puedes añadir varias entradas en las `StringLike` condiciones `StringEquals` o para permitir que varias cuentas de servicio o espacios de nombres asuman la función. Para permitir que su cuenta de servicio asuma una función de IAM en otra AWS cuenta, consulte Permisos de IAM entre cuentas en la Guía del usuario de Amazon EKS.

3. Cree una función de IAM con el nombre que vaya `my-iam-role` a asumir la cuenta de servicio de Amazon EKS. Adjunte al rol el archivo de política de confianza creado en el último paso. La

política de confianza especifica la cuenta de servicio y el proveedor de OIDC en los que puede confiar la función de IAM.

```
aws iam create-role --role-name my-iam-role --assume-role-policy-document file://
trust-relationship.json --description "EKS service account role"
```

4. Adjunte una política de acceso para asignar los permisos del rol de IAM a Amazon Keyspaces.

   a. Adjunta una política de acceso para definir las acciones que la función de IAM puede realizar en recursos específicos de Amazon Keyspaces. Para este tutorial utilizamos la política AWS gestionada`AmazonKeyspacesFullAccess`, ya que nuestra aplicación escribirá datos en tu tabla de Amazon Keyspaces. Sin embargo, como práctica recomendada, se recomienda crear políticas de acceso personalizadas que implementen el principio de privilegios mínimos. Para obtener más información, consulte the section called “Funcionamiento de Amazon Keyspaces con IAM”.

   ```
   aws iam attach-role-policy --role-name my-iam-role --policy-
   arn=arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess
   ```

   Confirme que la política se asoció correctamente a la función de IAM con la siguiente declaración.

   ```
   aws iam list-attached-role-policies --role-name my-iam-role
   ```

   La salida debería tener este aspecto.

   ```
   {
       "AttachedPolicies": [
           {
               "PolicyName": "AmazonKeyspacesFullAccess",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess"
           }
       ]
   }
   ```

   b. Anote la cuenta de servicio con el nombre de recurso de Amazon (ARN) de la función de IAM que puede asumir. Asegúrese de actualizar el ARN del rol con su ID de cuenta.

```
kubectl annotate serviceaccount -n my-eks-namespace my-eks-serviceaccount
 eks.amazonaws.com/role-arn=arn:aws:iam::111122223333:role/my-iam-role
```

5. Confirme que el rol de IAM y la cuenta de servicio estén configurados correctamente.

   a. Confirme que la política de confianza del rol de IAM está configurada correctamente con la siguiente declaración.

   ```
   aws iam get-role --role-name my-iam-role --query Role.AssumeRolePolicyDocument
   ```

   La salida debería ser similar a la siguiente.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Federated": "arn:aws:iam::111122223333:oidc-provider/
   oidc.eks.aws-region.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
               },
               "Action": "sts:AssumeRoleWithWebIdentity",
               "Condition": {
                   "StringEquals": {
                       "oidc.eks.aws-region/id/
   EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                       "oidc.eks.aws-region.amazonaws.com/id/
   EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
   namespace:my-eks-serviceaccount"
                   }
               }
           }
       ]
   }
   ```

   b. Confirme que la cuenta de servicio de Amazon EKS esté anotada con el rol de IAM.

   ```
   kubectl describe serviceaccount my-eks-serviceaccount -n my-eks-namespace
   ```

   La salida debería ser similar a la siguiente.

```
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>
[...]
```

Tras crear la cuenta de servicio Amazon EKS, el rol de IAM y configurar las relaciones y los permisos necesarios, continúe the section called “Paso 2: Configurar la aplicación” con.

Paso 2: Configure la aplicación

En este paso, crea la aplicación que se conecta a Amazon Keyspaces mediante el complemento SigV4. Puedes ver y descargar la aplicación Java de ejemplo desde el repositorio de código de ejemplo de Amazon Keyspaces en Github. O puede continuar con su propia aplicación, asegurándose de completar todos los pasos de configuración.

Configure su aplicación y añada las dependencias necesarias.

1. Puedes descargar la aplicación Java de ejemplo clonando el repositorio de Github con el siguiente comando.

   ```
   git clone https://github.com/aws-samples/amazon-keyspaces-examples.git
   ```

2. Después de descargar el repositorio de Github, descomprime el archivo descargado y navega hasta el `resources` directorio del archivo. `application.conf`

   a. Configuración de la aplicación

      En este paso, configura el complemento de autenticación SigV4. Puede usar el siguiente ejemplo en su aplicación. Si aún no lo ha hecho, debe generar sus claves de acceso a IAM (un ID de clave de acceso y una clave de acceso secreta) y guardarlas en su archivo de AWS configuración o como variables de entorno. Para obtener instrucciones detalladas, consulte the section called “Credenciales requeridas para la autenticación de AWS”. Actualice la AWS región y el punto de enlace del servicio para Amazon Keyspaces según sea necesario. Para obtener más puntos de enlace de servicio, consulte. the section called

"Puntos de conexión de servicio" Sustituya la ubicación de la tienda de confianza, el nombre de la tienda de confianza y la contraseña de la tienda de confianza por la suya propia.

```
datastax-java-driver {
  basic.contact-points = ["cassandra.aws-region.amazonaws.com:9142"]
  basic.load-balancing-policy.local-datacenter = "aws-region"
  advanced.auth-provider {
    class = software.aws.mcs.auth.SigV4AuthProvider
    aws-region = "aws-region"
  }
  advanced.ssl-engine-factory {
    class = DefaultSslEngineFactory
    truststore-path = "truststore_locationtruststore_name.jks"
    truststore-password = "truststore_password;"
  }
}
```

b. Agregue la dependencia del módulo STS.

Esto añade la posibilidad de utilizar una `WebIdentityTokenCredentialsProvider` que devuelva las AWS credenciales que la aplicación debe proporcionar para que la cuenta de servicio pueda asumir la función de IAM. Puede hacerlo basándose en el siguiente ejemplo.

```
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>aws-java-sdk-sts</artifactId>
            <version>1.11.717</version>
        </dependency>
```

c. Agregue la dependencia de SigV4.

Este paquete implementa el complemento de autenticación SigV4 que se necesita para autenticarse en Amazon Keyspaces.

```
        <dependency>
            <groupId>software.aws.mcs</groupId>
            <artifactId>aws-sigv4-auth-cassandra-java-driver-plugin</
artifactId>
            <version>4.0.3</version>
        </dependency>
```

3. Agregue una dependencia de registro.

Sin registros, es imposible solucionar los problemas de conexión. En este tutorial, lo utilizamos `slf4j` como marco de registro y lo utilizamos `logback.xml` para almacenar la salida del registro. Establecemos el nivel de registro `debug` en para establecer la conexión. Puede usar el siguiente ejemplo para agregar la dependencia.

```
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>2.0.5</version>
        </dependency>
```

Puede usar el siguiente fragmento de código para configurar el registro.

```
<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**
>
> El `debug` nivel es necesario para investigar los errores de conexión. Una vez que se haya conectado correctamente a Amazon Keyspaces desde su aplicación, podrá cambiar el nivel de registro a `info` o `warning` según sea necesario.

Paso 3: Cree la imagen de la aplicación y cargue el archivo Docker en su repositorio de Amazon ECR

En este paso, debe compilar la aplicación de ejemplo, crear una imagen de Docker y enviar la imagen a su repositorio de Amazon ECR.

Cree su aplicación, cree una imagen de Docker y envíela a Amazon Elastic Container Registry

1. Establezca las variables de entorno para la compilación que definan su. Región de AWS Sustituya las regiones de los ejemplos por las suyas propias.

```
export CASSANDRA_HOST=cassandra.aws-region.amazonaws.com:9142
export CASSANDRA_DC=aws-region
```

2. Compila tu aplicación con la versión 3.6.3 o superior de Apache Maven mediante el siguiente comando.

```
mvn clean install
```

   Esto crea un JAR archivo con todas las dependencias incluidas en el directorio. `target`

3. Recupera el URI del repositorio de ECR que necesitas para el siguiente paso con el siguiente comando. Asegúrese de actualizar la región a la que ha estado utilizando.

```
aws ecr describe-repositories --region aws-region
```

   El resultado debería tener el aspecto que se muestra en el siguiente ejemplo.

```
"repositories": [
 {
 "repositoryArn": "arn:aws:ecr:aws-region:111122223333:repository/my-ecr-
repository",
 "registryId": "111122223333",
 "repositoryName": "my-ecr-repository",
 "repositoryUri": "111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-
repository",
 "createdAt": "2023-11-02T03:46:34+00:00",
 "imageTagMutability": "MUTABLE",
 "imageScanningConfiguration": {
 "scanOnPush": false
 },
 "encryptionConfiguration": {
 "encryptionType": "AES256"
 }
 },
```

4. Desde el directorio raíz de la aplicación, cree la imagen de Docker utilizando el URI del repositorio del último paso. Modifique el archivo Docker según sea necesario. En el comando build, asegúrese de reemplazar su ID de cuenta y configurarlo en la región en la Región de AWS que se `my-ecr-repository` encuentra el repositorio de Amazon ECR.

```
docker build -t 111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-repository:latest .
```

5. Recupere un token de autenticación para enviar la imagen de Docker a Amazon ECR. Puede hacerlo con el siguiente comando.

```
aws ecr get-login-password --region aws-region | docker login --username AWS --password-stdin 111122223333.dkr.ecr.aws-region.amazonaws.com
```

6. En primer lugar, compruebe las imágenes existentes en su repositorio de Amazon ECR. Puede utilizar el siguiente comando.

```
aws ecr describe-images --repository-name my-ecr-repository --region aws-region
```

   A continuación, coloca la imagen de Docker en el repositorio. Puede utilizar el siguiente comando.

```
docker push 111122223333.dkr.ecr.aws-region.amazonaws.com/my-ecr-repository:latest
```

Paso 4: Implemente la aplicación en Amazon EKS y escriba los datos en la tabla de Amazon Keyspaces

En este paso del tutorial, configurará la implementación de Amazon EKS para su aplicación y confirmará que la aplicación se está ejecutando y que puede conectarse a Amazon Keyspaces.

Para implementar una aplicación en Amazon EKS, debe configurar todos los ajustes pertinentes en un archivo llamado`deployment.yaml`. A continuación, Amazon EKS utiliza este archivo para implementar la aplicación. Los metadatos del archivo deben contener la siguiente información:

- Nombre de la aplicación: el nombre de la aplicación. Para este tutorial, utilizamos`my-keyspaces-app`.
- Espacio de nombres de Kubernetes: el espacio de nombres del clúster de Amazon EKS. Para este tutorial, utilizamos. `my-eks-namespace`

- Nombre de la cuenta de servicio de Amazon EKS: el nombre de la cuenta de servicio de Amazon EKS. Para este tutorial, utilizamos`my-eks-serviceaccount`.
- nombre de imagen: el nombre de la imagen de la aplicación. Para este tutorial, utilizamos`my-keyspaces-app`.
- URI de imagen: el URI de imagen de Docker de Amazon ECR.
- AWS ID de cuenta: su ID de AWS cuenta.
- Rol de IAM ARN: el ARN del rol de IAM creado para que lo asuma la cuenta de servicio. Para este tutorial, utilizamos. `my-iam-role`
- Región de AWS del clúster de Amazon EKS en el Región de AWS que creó su clúster de Amazon EKS.

En este paso, debe implementar y ejecutar la aplicación que se conecta a Amazon Keyspaces y escribe los datos en la tabla.

1. Configure el archivo `deployment.yaml`. Debe reemplazar los siguientes valores:

   - `name`
   - `namespace`
   - `serviceAccountName`
   - `image`
   - `AWS_ROLE_ARN value`
   - La Región de AWS entrada `CASSANDRA_HOST`
   - `AWS_REGION`

   Puede usar el siguiente archivo como ejemplo.

```
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.aws-region.amazonaws.com/my-ecr-
repository:latest
        ports:
        - containerPort: 8080
        env:
        - name: CASSANDRA_HOST
          value: "cassandra.aws-region.amazonaws.com:9142"
        - name: CASSANDRA_DC
          value: "aws-region"
        - 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: "aws-region"
```

2. Implementar `deployment.yaml`.

```
kubectl apply -f deployment.yaml
```

La salida debería tener este aspecto.

```
deployment.apps/my-keyspaces-app created
```

3. Compruebe el estado del pod en el espacio de nombres del clúster de Amazon EKS.

```
kubectl get pods -n my-eks-namespace
```

La salida debe ser similar al siguiente ejemplo.

```
NAME                    READY STATUS RESTARTS AGE
my-keyspaces-app-123abcde4f-g5hij 1/1 Running 0 75s
```

Para obtener más información, puede usar el siguiente comando.

```
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.aws-region.amazonaws.com/
my_eks_repository:latest
    Image ID:       111122223333.dkr.ecr.aws-region.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.aws-region.amazonaws.com:9142
      CASSANDRA_DC:                  aws-region
      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:                   aws-region
      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.aws-region.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.aws-region.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.aws-region.amazonaws.com/
my_eks_repository:latest" in 222ms (222ms including waiting)
  Normal   Started          53s (x2 over 75s)  kubelet            Started container
 my-keyspaces-app
```

4. Comprueba los registros del pod para confirmar que la aplicación se está ejecutando y que se puede conectar a la tabla de Amazon Keyspaces. Puedes hacerlo con el siguiente comando. Asegúrese de reemplazar el nombre de la implementación.

```
kubectl logs -f my-keyspaces-app-123abcde4f-g5hij -n my-eks-namespace
```

Deberías poder ver las entradas del registro de la aplicación que confirman la conexión a Amazon Keyspaces, como en el ejemplo siguiente.

```
2:47:20.553 [s0-admin-0] DEBUG c.d.o.d.i.c.metadata.MetadataManager
 - [s0] Adding initial contact points [Node(endPoint=cassandra.aws-
region.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. Ejecute la siguiente consulta de CQL en la tabla de Amazon Keyspaces para confirmar que se ha escrito una fila de datos en la tabla:

```
SELECT * from aws.user;
```

Debería ver los siguientes datos de salida:

```
fname    | lname | username | last_update_date
----------+-------+----------+---------------------------
random    | k     | test     | 2023-12-07 13:58:31.57+0000
```

Paso 5: Limpieza (opcional)

Siga estos pasos para eliminar todos los recursos creados en este tutorial.

Elimine los recursos creados en este tutorial

1. Elimine la implementación. Para ello, puede utilizar el siguiente comando.

```
kubectl delete deployment my-keyspaces-app -n my-eks-namespace
```

2. Elimina el clúster de Amazon EKS y todos los pods que contiene. Esto también elimina los recursos relacionados, como la cuenta de servicio y el proveedor de identidad del OIDC. Para ello, puede utilizar el siguiente comando.

```
eksctl delete cluster --name my-eks-cluster --region aws-region
```

3. Elimine la función de IAM utilizada para la cuenta de servicio Amazon EKS con permisos de acceso a Amazon Keyspaces. En primer lugar, debe eliminar la política gestionada que está asociada a la función.

```
aws iam detach-role-policy --role-name my-iam-role --policy-arn
 arn:aws:iam::aws:policy/AmazonKeyspacesFullAccess
```

   A continuación, puede eliminar el rol mediante el siguiente comando.

```
aws iam delete-role --role-name my-iam-role
```

   Para obtener más información, consulte Eliminar un rol de IAM (AWS CLI) en la Guía del usuario de IAM.

4. Elimine el repositorio de Amazon ECR, incluidas todas las imágenes almacenadas en él. Puede hacerlo mediante el siguiente comando.

```
aws ecr delete-repository \
      --repository-name my-ecr-repository \
      --force \
      --region aws-region
```

   Tenga en cuenta que la `force` marca es necesaria para eliminar un repositorio que contiene imágenes. Para eliminar primero la imagen, puede hacerlo mediante el siguiente comando.

```
aws ecr batch-delete-image \
      --repository-name my-ecr-repository \
      --image-ids imageTag=latest \
      --region aws-region
```

Para obtener más información, consulte Eliminar una imagen en la Guía del usuario de Amazon Elastic Container Registry.

5. Elimina el espacio de claves y la tabla de Amazon Keyspaces. Al eliminar el espacio de claves, se eliminan automáticamente todas las tablas de ese espacio de claves. Para ello, puede utilizar una de las siguientes opciones.

   AWS CLI

   ```
   aws keyspaces delete-keyspace --keyspace-name 'aws'
   ```

   Para confirmar que se ha eliminado el espacio de claves, puede utilizar el siguiente comando.

   ```
   aws keyspaces list-keyspaces
   ```

   Para eliminar primero la tabla, puede usar el siguiente comando.

   ```
   aws keyspaces delete-table --keyspace-name 'aws' --table-name 'user'
   ```

   Para confirmar que se ha eliminado la tabla, puede utilizar el siguiente comando.

   ```
   aws keyspaces list-tables --keyspace-name 'aws'
   ```

   Para obtener más información, consulte eliminar el espacio de teclas y eliminar la tabla en la Referencia de AWS CLI comandos.

   cqlsh

   ```
   DROP KEYSPACE IF EXISTS "aws";
   ```

   Para comprobar que se han eliminado los espacios de teclas, puede utilizar la siguiente sentencia.

```
SELECT * FROM system_schema.keyspaces ;
```

Su espacio de claves no debe figurar en el resultado de esta declaración. Tenga en cuenta que puede haber un retraso hasta que se eliminen los espacios clave. Para obtener más información, consulte the section called “DROP KEYSPACE”.

Para eliminar primero la tabla, puede usar el siguiente comando.

```
DROP TABLE "aws.user"
```

Para confirmar que se ha eliminado la tabla, puede utilizar el siguiente comando.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = "aws";
```

La tabla no debe aparecer en el resultado de esta declaración. Tenga en cuenta que puede haber un retraso hasta que se elimine la tabla. Para obtener más información, consulte the section called “DROP TABLE”.

Tutorial: Conexión a Amazon Keyspaces mediante un punto de conexión de VPC de interfaz

En este tutorial se explica cómo configurar y utilizar un punto de conexión de VPC de interfaz para Amazon Keyspaces.

Los puntos de conexión de VPC de interfaz habilitan la comunicación privada entre su nube privada virtual (VPC) que se ejecuta en Amazon VPC y Amazon Keyspaces. Los puntos finales de la interfaz VPC funcionan con AWS PrivateLink, que es un AWS servicio que permite la comunicación privada entre las VPC y los servicios. AWS Para obtener más información, consulte the section called “Uso de los puntos de conexión de VPC de interfaz”.

Temas

- Requisitos y consideraciones previos del tutorial
- Paso 1: Lanzar una instancia de Amazon EC2
- Paso 2: configurar la instancia de Amazon EC2
- Paso 3: Crear un punto de conexión de VPC para Amazon Keyspaces

- Paso 4: Configurar los permisos para la conexión del punto de conexión de VPC
- Paso 5: Configure la supervisión con CloudWatch
- Paso 6: (Opcional) Prácticas recomendadas para configurar el tamaño del grupo de conexiones para su aplicación
- Paso 7: (Opcional) Sanear

Requisitos y consideraciones previos del tutorial

Antes de comenzar este tutorial, siga las instrucciones de AWS configuración incluidas en. Acceso a Amazon Keyspaces (para Apache Cassandra) Estos pasos incluyen la inscripción AWS y la creación de un director AWS Identity and Access Management (IAM) con acceso a Amazon Keyspaces. Tome nota del nombre de usuario de IAM y de las claves de acceso dado que las necesitará más adelante en este tutorial.

Cree un espacio de claves con el nombre `myKeyspace` y al menos una tabla para probar la conexión utilizando el punto de conexión de VPC más adelante en este tutorial. Puede encontrar instrucciones detalladas en Introducción.

Tras completar los pasos de requisitos previos, continúe con Paso 1: Lanzar una instancia de Amazon EC2.

Paso 1: Lanzar una instancia de Amazon EC2

En este paso, lanza una instancia de Amazon EC2 en su Amazon VPC predeterminada. A continuación, puede crear y utilizar un punto de conexión de VPC para Amazon Keyspaces.

Para lanzar una instancia Amazon EC2

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. Elija Launch Instance y proceda del modo siguiente:

   Desde el panel de control de la consola EC2, en el cuadro Lanzar instancia, elija Lanzar instancia y, a continuación, Lanzar instancia entre las opciones que aparezcan.

   En Nombre y etiquetas, en Nombre, introduzca un nombre descriptivo para su instancia.

   En Imágenes de aplicaciones y SO (Imagen de máquina de Amazon):

- Elija Inicio rápido y, a continuación, elija Ubuntu. Este es el sistema operativo (SO) de la instancia.
- En Imagen de máquina de Amazon (AMI), puede utilizar la imagen predeterminada que está marcada como Nivel libre eligible. Una imagen de máquina de Amazon (AMI) es una configuración básica que sirve de plantilla para su instancia.

En Tipo de instancia:

- En la lista Tipo de instancia, elija el tipo de instancia t2.micro, que se selecciona de forma predeterminada.

En Par de claves (inicio de sesión), en Nombre del par de claves, elija una de las siguientes opciones para este tutorial:

- Si no dispone de un par de claves de Amazon EC2, elija Create a new key pair (Crear un nuevo par de claves) y siga las instrucciones. Se le pedirá que descargue un archivo de clave privada (archivo .pem). Necesitará este archivo más adelante cuando inicie sesión en su instancia de Amazon EC2, así que tome nota de la ruta del archivo.
- Si ya dispone de un par de claves de Amazon EC2, vaya a Select a key pair y elíjalo en la lista. Debe tener el archivo de clave privada (archivo .pem) a su disposición para poder iniciar sesión en la instancia de Amazon EC2.

En Configuración de red:

- Elija Editar.
- Elija Seleccionar un grupo de seguridad existente.
- En la lista de grupos de seguridad, elija default. Se trata del grupo de seguridad predeterminado para la VPC.

Continúe en Resumen.

- Revise un resumen de la configuración de su instancia en el panel Resumen. Cuando esté preparado, elija Lanzar instancia.

3. En la pantalla de finalización de la nueva instancia de Amazon EC2, elija el mosaico Conectarse a la instancia. En la siguiente pantalla se muestra la información necesaria y los pasos requeridos para conectarse a su nueva instancia. Tome nota de la siguiente información:

   - El comando de ejemplo para proteger el archivo de claves
   - La cadena de conexión
   - El nombre de DNS IPv4 público

   Después de tomar nota de la información de esta página, puede continuar con el siguiente paso de este tutorial (Paso 2: configurar la instancia de Amazon EC2).

> **Note**
>
> La instancia de Amazon EC2 tardará unos minutos en estar disponible. Antes de continuar con el siguiente paso, asegúrese de que Instance State tenga el valor `running` y de que se hayan superado todas las comprobaciones de estado (Status Checks).

Paso 2: configurar la instancia de Amazon EC2

Cuando su instancia de Amazon EC2 esté disponible, puede iniciar sesión en ella y prepararla para su primer uso.

> **Note**
>
> En los pasos siguientes se asume que se conecta a su instancia de Amazon EC2 desde un equipo que ejecuta Linux. Para conocer otras formas de conectarse, consulte Conexión a su instancia Linux en la Guía del usuario de Amazon EC2 para instancias Linux.

Para configurar su instancia de Amazon EC2

1. Debe autorizar el tráfico SSH entrante a su instancia de Amazon EC2. Para ello, cree un nuevo grupo de seguridad EC2 y, a continuación, asigne el grupo de seguridad a su instancia EC2.

   a. En el panel de navegación, elija Grupos de seguridad.
   b. Elija Crear grupo de seguridad. En la ventana Crear grupo de seguridad, haga lo siguiente:

- Nombre del grupo de seguridad: introduzca un nombre para su grupo de seguridad. Por ejemplo: `my-ssh-access`
- Descripción: introduzca una breve descripción para el grupo de seguridad.
- VPC: elija su VPC predeterminada.
- En la sección Reglas de entrada, elija Añadir regla y haga lo siguiente:
  - Tipo: elija SSH.
  - Origen: elija Mi IP.
  - Seleccione Añadir regla.

En la parte inferior de la página, confirme los ajustes de configuración y elija Crear grupo de seguridad.

c. En el panel de navegación, seleccione Instancias.

d. Elija la instancia de Amazon EC2 que ha lanzado en Paso 1: Lanzar una instancia de Amazon EC2.

e. Elija Acciones, luego Seguridad y, a continuación, Cambiar grupos de seguridad.

f. En Cambiar grupos de seguridad, seleccione el grupo de seguridad que creó anteriormente en este procedimiento (por ejemplo, `my-ssh-access`). El grupo de seguridad `default` existente también debe estar seleccionado. Confirme los ajustes de configuración y elija Asignar grupos de seguridad.

2. Utilice el siguiente comando para proteger su archivo de clave privada contra el acceso. Si omite este paso, la conexión fallará.

```
chmod 400 path_to_file/my-keypair.pem
```

3. Utilice el comando `ssh` para iniciar sesión en su instancia de Amazon EC2, como se muestra en el siguiente ejemplo.

```
ssh -i path_to_file/my-keypair.pem ubuntu@public-dns-name
```

Debe especificar su archivo de clave privada (archivo .pem) y el nombre de DNS público de su instancia. (Consulte Paso 1: Lanzar una instancia de Amazon EC2).

El identificador de inicio de sesión es `ubuntu`. No se requiere contraseña.

Para obtener más información sobre cómo permitir conexiones a su instancia de Amazon EC2 y sobre instrucciones de la AWS CLI , consulte Autorización del tráfico entrante para instancias Linux en la Guía del usuario de Amazon EC2 para instancias Linux.

4. Descargue e instale la última versión de la AWS Command Line Interface.

   a. Instale `unzip`.

   ```
   sudo apt install unzip
   ```

   b. Descargue el archivo `zip` con la AWS CLI.

   ```
   curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
    "awscliv2.zip"
   ```

   c. Descomprima el archivo.

   ```
   unzip awscliv2.zip
   ```

   d. Instale el. AWS CLI

   ```
   sudo ./aws/install
   ```

   e. Confirme la versión de la AWS CLI instalación.

   ```
   aws --version
   ```

   La salida debe tener el siguiente aspecto:

   ```
   aws-cli/2.9.19 Python/3.9.11 Linux/5.15.0-1028-aws exe/x86_64.ubuntu.22 prompt/
   off
   ```

5. Configure sus AWS credenciales, como se muestra en el siguiente ejemplo. Introduzca el identificador de la clave de AWS acceso, la clave secreta y el nombre predeterminado de la región cuando se le solicite.

   ```
   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. Debe utilizar una conexión `cqlsh` a Amazon Keyspaces para confirmar que su punto de conexión de VPC se haya configurado correctamente. Si utiliza su entorno local o el editor CQL de Amazon Keyspaces en el AWS Management Console, la conexión pasa automáticamente por el punto de enlace público en lugar de por el punto de enlace de la VPC. Para utilizar `cqlsh` con el fin de probar su conexión de punto de conexión de VPC en este tutorial, complete las instrucciones de configuración indicadas en Uso de `cqlsh` para conectarse a Amazon Keyspaces.

Ya está listo para crear un punto de conexión de VPC para Amazon Keyspaces.

Paso 3: Crear un punto de conexión de VPC para Amazon Keyspaces

En este paso, usted crea un punto de conexión de VPC para Amazon Keyspaces utilizando la AWS CLI. Para crear el punto de conexión de VPC utilizando la consola de VPC, puede seguir las instrucciones indicadas en Creación de un punto de conexión de VPC en la Guía de AWS PrivateLink . Al filtrar para Nombre del servicio, introduzca **`Cassandra`**.

Para crear un punto final de VPC mediante el AWS CLI

1. Antes de comenzar, compruebe que puede comunicar con Amazon Keyspaces utilizando su punto de conexión público.

```
aws keyspaces list-tables --keyspace-name 'myKeyspace'
```

La salida muestra una lista de las tablas de Amazon Keyspaces contenidas en el espacio de claves especificado. Si no contiene ninguna tabla, la lista aparece vacía.

```
{
    "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. Compruebe que Amazon Keyspaces sea un servicio disponible para crear puntos de enlace de VPC en la región actual. AWS (El comando se muestra en negrita, seguido del resultado de ejemplo).

```
aws ec2 describe-vpc-endpoint-services

{
    "ServiceNames": [
        "com.amazonaws.us-east-1.cassandra",
        "com.amazonaws.us-east-1.cassandra-fips"
    ]
}
```

   En la salida del ejemplo, Amazon Keyspaces es uno de los servicios disponibles, por lo que puede proceder a crear un punto de conexión de VPC para el mismo.

3. Determine el identificador de su VPC.

```
aws ec2 describe-vpcs

{
    "Vpcs": [
        {
            "VpcId": "vpc-a1234bcd",
            "InstanceTenancy": "default",
            "State": "available",
            "DhcpOptionsId": "dopt-8454b7e1",
            "CidrBlock": "111.31.0.0/16",
            "IsDefault": true
        }
    ]
}
```

   En el resultado de ejemplo, el ID de la VPC es `vpc-a1234bcd`.

4. Utilice un filtro para recopilar detalles sobre las subredes de la VPC.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-a1234bcd"

{
     {
    "Subnets":[
       {
          "AvailabilityZone":"us-east-1a",
          "AvailabilityZoneId":"use2-az1",
          "AvailableIpAddressCount":4085,
          "CidrBlock":"111.31.0.0/20",
          "DefaultForAz":true,
          "MapPublicIpOnLaunch":true,
          "MapCustomerOwnedIpOnLaunch":false,
          "State":"available",
          "SubnetId":"subnet-920aacf9",
          "VpcId":"vpc-a1234bcd",
          "OwnerId":"111122223333",
          "AssignIpv6AddressOnCreation":false,
          "Ipv6CidrBlockAssociationSet":[

          ],
          "SubnetArn":"arn:aws:ec2:us-east-1:111122223333:subnet/subnet-920aacf9",
          "EnableDns64":false,
          "Ipv6Native":false,
          "PrivateDnsNameOptionsOnLaunch":{
             "HostnameType":"ip-name",
             "EnableResourceNameDnsARecord":false,
             "EnableResourceNameDnsAAAARecord":false
          }
       },
       {
          "AvailabilityZone":"us-east-1c",
          "AvailabilityZoneId":"use2-az3",
          "AvailableIpAddressCount":4085,
          "CidrBlock":"111.31.32.0/20",
          "DefaultForAz":true,
          "MapPublicIpOnLaunch":true,
          "MapCustomerOwnedIpOnLaunch":false,
          "State":"available",
          "SubnetId":"subnet-4c713600",
          "VpcId":"vpc-a1234bcd",
          "OwnerId":"111122223333",
          "AssignIpv6AddressOnCreation":false,
```

```
            "Ipv6CidrBlockAssociationSet":[

            ],
            "SubnetArn":"arn:aws:ec2:us-east-1:111122223333:subnet/subnet-4c713600",
            "EnableDns64":false,
            "Ipv6Native":false,
            "PrivateDnsNameOptionsOnLaunch":{
               "HostnameType":"ip-name",
               "EnableResourceNameDnsARecord":false,
               "EnableResourceNameDnsAAAARecord":false
            }
         },
         {
            "AvailabilityZone":"us-east-1b",
            "AvailabilityZoneId":"use2-az2",
            "AvailableIpAddressCount":4086,
            "CidrBlock":"111.31.16.0/20",
            "DefaultForAz":true,
            "MapPublicIpOnLaunch":true,


         }
      ]
}
```

En la salida del ejemplo, hay dos ID de subred disponibles: `subnet-920aacf9` y `subnet-4c713600`.

5. Crear el punto de enlace de la VPC. Para el parámetro `--vpc-id`, especifique el ID de la VPC del paso anterior. Para el parámetro `--subnet-id`, especifique los ID de subred del paso anterior. Utilice el parámetro `--vpc-endpoint-type` para definir el punto de conexión como interfaz. Para obtener más información sobre el comando, consulte `create-vpc-endpoint` en la Referencia de comandos de la AWS CLI .

```
aws ec2 create-vpc-endpoint --vpc-endpoint-type Interface --vpc-id vpc-a1234bcd
 --service-name com.amazonaws.us-east-1.cassandra --subnet-id subnet-920aacf9
 subnet-4c713600

{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-000ab1cdef23456789",
        "VpcEndpointType": "Interface",
        "VpcId": "vpc-a1234bcd",
```

```
        "ServiceName": "com.amazonaws.us-east-1.cassandra",
        "State": "pending",
        "RouteTableIds": [],
        "SubnetIds": [
            "subnet-920aacf9",
            "subnet-4c713600"
        ],
        "Groups": [
            {
                "GroupId": "sg-ac1b0e8d",
                "GroupName": "default"
            }
        ],
        "IpAddressType": "ipv4",
        "DnsOptions": {
            "DnsRecordIpType": "ipv4"
        },
        "PrivateDnsEnabled": true,
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-043c30c78196ad82e",
            "eni-06ce37e3fd878d9fa"
        ],
        "DnsEntries": [
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz.cassandra.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1a.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1c.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1b.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
```

```
            {
                "DnsName": "vpce-000ab1cdef23456789-m2b22rtz-us-
east-1d.cassandra.us-east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "cassandra.us-east-1.amazonaws.com",
                "HostedZoneId": "ZONEIDPENDING"
            }
        ],
        "CreationTimestamp": "2023-01-27T16:12:36.834000+00:00",
        "OwnerId": "111122223333"
    }
}

}
```

Paso 4: Configurar los permisos para la conexión del punto de conexión de VPC

Los procedimientos descritos en este paso muestran cómo configurar reglas y permisos para utilizar el punto de conexión de VPC con Amazon Keyspaces.

Para configurar una regla de entrada para que el nuevo punto de conexión permita el tráfico de entrada TCP

1. En la consola de Amazon VPC, en el panel izquierdo, elija Endpoints y luego el punto de conexión que creó en el paso anterior.
2. Elija Grupos de seguridad y luego el grupo de seguridad asociado a este punto de conexión.
3. Elija Reglas de entrada y, a continuación, Editar reglas de entrada.
4. Añada una regla de entrada con Tipo como TCP personalizado. En Rango de puertos, introduzca **9142**.
5. Para guardar la nueva regla de entrada, elija Guardar reglas.

Para configurar los permisos de usuario de IAM

1. Confirme que el usuario de IAM utilizado para conectarse a Amazon Keyspaces disponga de los permisos apropiados. En AWS Identity and Access Management (IAM), puedes usar la

política AWS gestionada `AmazonKeyspacesReadOnlyAccess` para conceder al usuario de IAM acceso de lectura a Amazon Keyspaces.

a. Inicie sesión en la consola de IAM AWS Management Console y ábrala en https://console.aws.amazon.com/iam/.

b. En el panel de la consola de IAM, elija Users y, a continuación, seleccione el usuario de IAM en la lista.

c. En la página Summary (Resumen), elija Añadir permisos.

d. Elija Attach existing policies directly (Adjuntar políticas existentes directamente).

e. En la lista de políticas, elija y AmazonKeyspacesReadOnlyAccess, a continuación, elija Siguiente: revisar.

f. Elija Añadir permisos.

2. Compruebe que pueda acceder a Amazon Keyspaces a través del punto de conexión de VPC.

```
aws keyspaces list-tables --keyspace-name 'my_Keyspace'
```

Si lo deseas, puedes probar otros AWS CLI comandos para Amazon Keyspaces. Para obtener más información, consulte Referencia de comandos de la AWS CLI.

> **Note**
>
> Los permisos mínimos necesarios para que un usuario o rol de IAM pueda acceder a Amazon Keyspaces son permisos de lectura en la tabla del sistema, como se muestra en la siguiente política. Para obtener más información sobre permisos basados en políticas, consulte the section called “Ejemplos de políticas basadas en identidades”.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:555555555555:/keyspace/system*"
```

```
            ]
        }
    ]
}
```

3. Conceda al usuario de IAM acceso de lectura a la instancia de Amazon EC2 con la VPC.

   Al utilizar Amazon Keyspaces con puntos de conexión de VPC, debe conceder al usuario o rol de IAM que acceda a Amazon Keyspaces permisos de solo lectura a su instancia de Amazon EC2 y a la VPC para recopilar datos de puntos de conexión e interfaces de red. Amazon Keyspaces almacena esta información en la tabla `system.peers` y la utiliza para administrar las conexiones.

   > **Note**
   >
   > Las políticas administradas `AmazonKeyspacesReadOnlyAccess_v2` y `AmazonKeyspacesFullAccess` incluyen los permisos necesarios para permitir que Amazon Keyspaces acceda a la instancia de Amazon EC2 para leer la información sobre los puntos de conexión de VPC de interfaz disponibles.

   a. Inicie sesión en la consola de IAM AWS Management Console y ábrala en https://console.aws.amazon.com/iam/.
   b. En el panel de control de la consola de IAM, elija Políticas.
   c. Elija Crear política y, a continuación, elija la pestaña JSON.
   d. Copie la siguiente política y elija Siguiente: Etiquetas.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource": "*"
      }
   ]
```

```
}
```

e. Elija Siguiente: Revisar, introduzca el nombre `keyspacesVPCendpoint` para la política y luego elija Crear política.

f. En el panel de la consola de IAM, elija Users y, a continuación, seleccione el usuario de IAM en la lista.

g. En la página Summary (Resumen), elija Añadir permisos.

h. Elija Attach existing policies directly (Adjuntar políticas existentes directamente).

i. En la lista de políticas, elija keyspacesVPCendpoint y, a continuación, Siguiente: Revisar.

j. Elija Añadir permisos.

4. Para comprobar que la tabla `system.peers` de Amazon Keyspaces se esté actualizando con información de la VPC, ejecute la siguiente consulta desde su instancia de Amazon EC2 mediante `cqlsh`. Si aún no ha instalado `cqlsh` en su instancia de Amazon EC2 según el paso 2, siga las instrucciones indicadas en the section called “Uso de la `cqlsh-expansion`”.

```
SELECT peer FROM system.peers;
```

El resultado devuelve nodos con direcciones IP privadas, según la configuración de VPC y subred en su región. AWS

```
peer
---------------
112.11.22.123
112.11.22.124
112.11.22.125
```

> **Note**
>
> Debe utilizar una conexión `cqlsh` a Amazon Keyspaces para confirmar que su punto de conexión de VPC se haya configurado correctamente. Si utiliza su entorno local o el editor CQL de Amazon Keyspaces en la AWS Management Console, la conexión pasa automáticamente por el punto de conexión público en lugar de por su punto de conexión de VPC. Si ve nueve direcciones IP, son las entradas que Amazon Keyspaces escribe automáticamente en la tabla `system.peers` para las conexiones de puntos de conexión públicos.

Paso 5: Configure la supervisión con CloudWatch

En este paso, se muestra cómo utilizar Amazon CloudWatch para supervisar la conexión del punto final de la VPC a Amazon Keyspaces.

AWS PrivateLink publica puntos de datos CloudWatch sobre los puntos finales de su interfaz. Puede utilizar estas métricas para comprobar si el sistema funciona de acuerdo con lo esperado. El espacio de `AWS/PrivateLinkEndpoints` nombres CloudWatch incluye las métricas de los puntos finales de la interfaz. Para obtener más información, consulte CloudWatch las métricas de la Guía AWS PrivateLink.AWS PrivateLink

Para crear un CloudWatch panel con métricas de puntos finales de VPC

1. Abre la CloudWatch consola en. https://console.aws.amazon.com/cloudwatch/
2. En el panel de navegación, seleccione Paneles. Elija Crear panel. Introduzca un nombre para el panel y elija Crear.
3. En Añadir widget, elija Número.
4. En Métricas, elija AWS/ PrivateLinkEndpoints.
5. Elija Tipo de punto de conexión, Nombre del servicio, ID del punto de conexión de VPC, ID de VPC.
6. Seleccione las métricas `ActiveConnections` y `NewConnections` y luego elija Crear widget.
7. Guarde el panel de control.

La métrica `ActiveConnections` se define como el número de conexiones activas concurrentes que el punto de conexión recibió durante el último periodo de un minuto. La métrica `NewConnections` se define como el número de nuevas conexiones que se establecieron a través del punto de conexión durante el último periodo de un minuto.

Para obtener más información sobre la creación de paneles, consulte Crear un panel en la Guía del CloudWatch usuario.

Paso 6: (Opcional) Prácticas recomendadas para configurar el tamaño del grupo de conexiones para su aplicación

En esta sección le explicamos la forma de determinar el tamaño ideal del grupo de conexiones en función de los requisitos de rendimiento de las consultas de su aplicación.

Amazon Keyspaces permite un máximo de 3000 consultas CQL por segundo por conexión TCP. Por tanto, no hay prácticamente ningún límite en el número de conexiones que un controlador puede establecer con Amazon Keyspaces. No obstante, le recomendamos que adapte el tamaño del grupo de conexiones a los requisitos de su aplicación y tenga en cuenta los puntos de conexión disponibles al utilizar Amazon Keyspaces con conexiones de punto de conexión de VPC.

El tamaño del grupo de conexiones se configura en el controlador de cliente. Por ejemplo, en base a un tamaño de grupo local de 2 y un punto de conexión de VPC creado a través de 3 zonas de disponibilidad, el controlador establece 6 conexiones para realizar consultas (7 en total, dado que incluye una conexión de control). A través de estas 6 conexiones, puede admitir un máximo de 18 000 consultas CQL por segundo.

Si su aplicación necesita admitir 40 000 consultas CQL por segundo, calcule hacia atrás a partir del número de consultas necesarias para determinar el tamaño del grupo de conexiones requerido. Para admitir 40 000 consultas CQL por segundo, debe configurar el tamaño del grupo local para que sea al menos de 5, lo que admite un mínimo de 45 000 consultas CQL por segundo.

Puedes monitorizar si superas la cuota del número máximo de operaciones por segundo y por conexión utilizando la `PerConnectionRequestRateExceeded` CloudWatch métrica del espacio de nombres. `AWS/Cassandra` La métrica `PerConnectionRequestRateExceeded` muestra el número de solicitudes efectuadas a Amazon Keyspaces que superan la cuota de tasa de solicitudes por conexión.

Los ejemplos de código en este paso muestran cómo calcular y configurar el grupo de conexiones cuando se utilizan puntos de conexión de VPC de interfaz.

Java

Puede configurar el número de conexiones por grupo en el controlador Java. Para ver un ejemplo completo de conexión de un controlador de cliente Java, consulte the section called “Uso de un controlador de cliente Java de Cassandra”.

Al iniciar el controlador de cliente, primero se establece la conexión de control para tareas administrativas, como los cambios de esquema y topología. A continuación se crean las conexiones adicionales.

En el siguiente ejemplo, la configuración del controlador de tamaño del grupo local se especifica como 2. Si el punto final de la VPC se crea en 3 subredes dentro de la VPC, se obtienen 7 entradas CloudWatch para el punto final de la interfaz, como se muestra `NewConnections` en la siguiente fórmula.

```
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
          }
}
```

Si el número de conexiones activas no coincide con el tamaño de su grupo configurado (agregación a través de subredes) + 1 conexión de control, algo impide que se creen las conexiones.

Node.js

Puede configurar el número de conexiones por grupo en el controlador Node.js. Para ver un ejemplo completo de una conexión del controlador de cliente Node.js, consulte the section called “Uso de un controlador de cliente Node.js para Cassandra”.

En el siguiente ejemplo de código, la configuración del controlador de tamaño del grupo local se especifica como 1. Si el punto final de la VPC se crea en 4 subredes dentro de la VPC, se obtienen 5 entradas CloudWatch para el punto final de la interfaz, como se muestra `NewConnections` en la siguiente fórmula.

```
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/sf-class2-root.crt', '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 }
        });
```

Paso 7: (Opcional) Sanear

Si desea eliminar los recursos que ha creado en este tutorial, siga estos procedimientos.

Para eliminar su punto de conexión de VPC para Amazon Keyspaces

1. Inicie sesión en la instancia de Amazon EC2.
2. Determine el ID del punto de conexión de VPC que se utiliza para Amazon Keyspaces. Si omite los parámetros `grep`, se muestra la información del punto de conexión de VPC para todos los servicios.

```
aws ec2 describe-vpc-endpoint-services | grep ServiceName | grep cassandra

{
```

```
    "VpcEndpoint": {
        "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*\"}]}",
        "VpcId": "vpc-0bbc736e",
        "State": "available",
        "ServiceName": "com.amazonaws.us-east-1.cassandra",
        "RouteTableIds": [],
        "VpcEndpointId": "vpce-9b15e2f2",
        "CreationTimestamp": "2017-07-26T22:00:14Z"
    }
}
```

En el resultado de ejemplo, el ID de punto de conexión de VPC es `vpce-9b15e2f2`.

3. Elimine el punto de conexión de VPC.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-9b15e2f2

{
    "Unsuccessful": []
}
```

La matriz vacía `[]` indica que la operación se ha realizado correctamente (no hay solicitudes con error).

Para terminar la instancia de Amazon EC2

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, seleccione Instancias.
3. Seleccione la instancia Amazon EC2.
4. Elija Acciones, luego Estado de la instancia y, a continuación, Terminar.
5. En la ventana de confirmación, elija Yes, Terminate.

Configuración del acceso entre cuentas para Amazon Keyspaces

Puede crear y utilizar Cuentas de AWS independientes para aislar recursos y para su uso en distintos entornos, por ejemplo, desarrollo y producción. En este tema se explica el acceso entre cuentas para Amazon Keyspaces mediante puntos de conexión de VPC de interfaz en una Amazon

Virtual Private Cloud. Para obtener más información sobre la configuración del acceso entre cuentas de IAM, consulte Escenario de ejemplo con cuentas de desarrollo y producción separadas en la Guía del usuario de IAM.

Para obtener más información sobre Amazon Keyspaces y los puntos de conexión de VPC privados, consulte the section called “Uso de los puntos de conexión de VPC de interfaz”.

Temas

- Configuración del acceso entre cuentas para Amazon Keyspaces en una VPC compartida
- Configuración del acceso entre cuentas para Amazon Keyspaces sin una VPC compartida

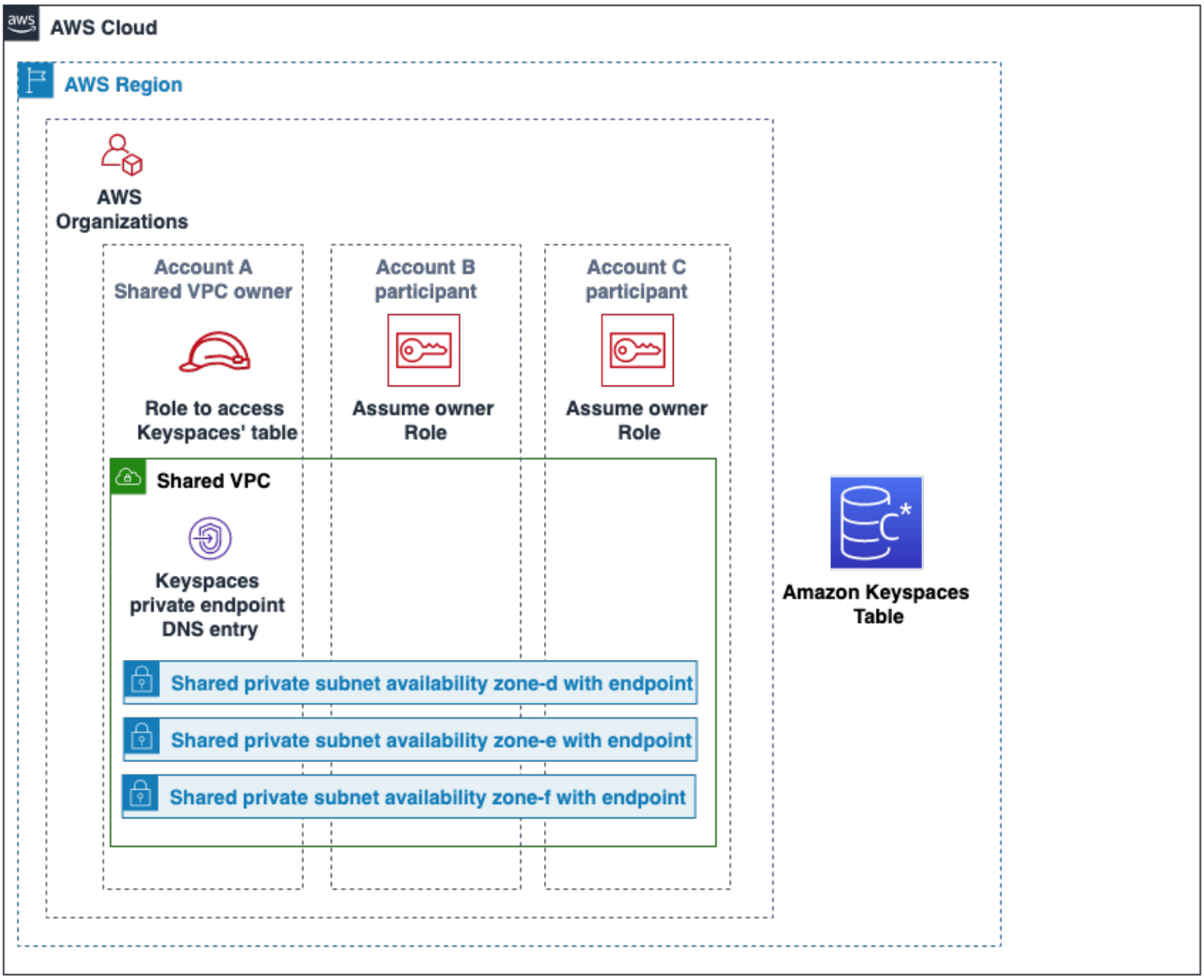
Configuración del acceso entre cuentas para Amazon Keyspaces en una VPC compartida

Puede crear diferentes Cuentas de AWS para separar los recursos de las aplicaciones. Por ejemplo, puede crear una cuenta para sus tablas de Amazon Keyspaces, una cuenta diferente para las aplicaciones en un entorno de desarrollo y otra cuenta para las aplicaciones en un entorno de producción. En este tema se describen los pasos de configuración necesarios para establecer el acceso entre cuentas para Amazon Keyspaces mediante puntos de conexión de VPC de interfaz en una VPC compartida.

Para obtener información detallada sobre cómo configurar un punto de conexión de VPC para Amazon Keyspaces, consulte the section called “Paso 3: Crear un punto de conexión de VPC para Amazon Keyspaces”.

En este ejemplo se utilizan las tres cuentas siguientes en una VPC compartida:

- `Account A`: esta cuenta contiene la infraestructura, incluyendo los puntos de conexión de VPC, las subredes de VPC y las tablas de Amazon Keyspaces.
- `Account B`: esta cuenta contiene una aplicación en un entorno de desarrollo que necesita conectarse a la tabla de Amazon Keyspaces en `Account A`.
- `Account C`: esta cuenta contiene una aplicación en un entorno de producción que necesita conectarse a la tabla de Amazon Keyspaces en `Account A`.



`Account A` es la cuenta que contiene los recursos a los que `Account B` y `Account C` necesitan acceder, por lo que `Account A` es la cuenta de confianza. y `Account B` y `Account C` son las cuentas con las entidades principales que necesitan acceder a los recursos de `Account A`, por lo que `Account B` y `Account C` son las cuentas confiadas. La cuenta de confianza concede los permisos a las cuentas confiadas compartiendo un rol de IAM. El siguiente procedimiento describe los pasos de configuración necesarios en `Account A`.

Configuración de **`Account A`**

1. Utilice AWS Resource Access Manager para crear un recurso compartido para la subred y comparta la subred privada con `Account B` y `Account C`.

`Account B` y `Account C` ya pueden ver y crear recursos en la subred que se ha compartido con ellos.

2. Cree un punto de conexión de VPC privado de Amazon Keyspaces de AWS PrivateLink. Esto crea varios puntos de conexión a través de subredes compartidas y entradas DNS para el punto de conexión del servicio de Amazon Keyspaces.
3. Cree un espacio de claves y una tabla de Amazon Keyspaces.
4. Cree un rol de IAM que tenga acceso completo a la tabla de Amazon Keyspaces, acceso de lectura a las tablas del sistema de Amazon Keyspaces y que pueda describir los recursos de VPC de Amazon EC2 como se muestra en el siguiente ejemplo de política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountAccess",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcEndpoints",
                "cassandra:*"
            ],
            "Resource": "*"
        }
    ]
}
```

5. Configure la política de confianza del rol de IAM que `Account B` y `Account C` puedan asumir como cuentas confiadas, como se muestra en el siguiente ejemplo.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111111111111:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {}
        }
```

```
    ]
}
```

Para obtener más información sobre las políticas de IAM entre cuentas, consulte Políticas entre cuentas en la Guía del usuario de IAM.

Configuración de **`Account B`** y **`Account C`**

1. En `Account B` y `Account C`, cree nuevos roles y vincule la siguiente política que permite a la entidad principal asumir el rol compartido creado en `Account A`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

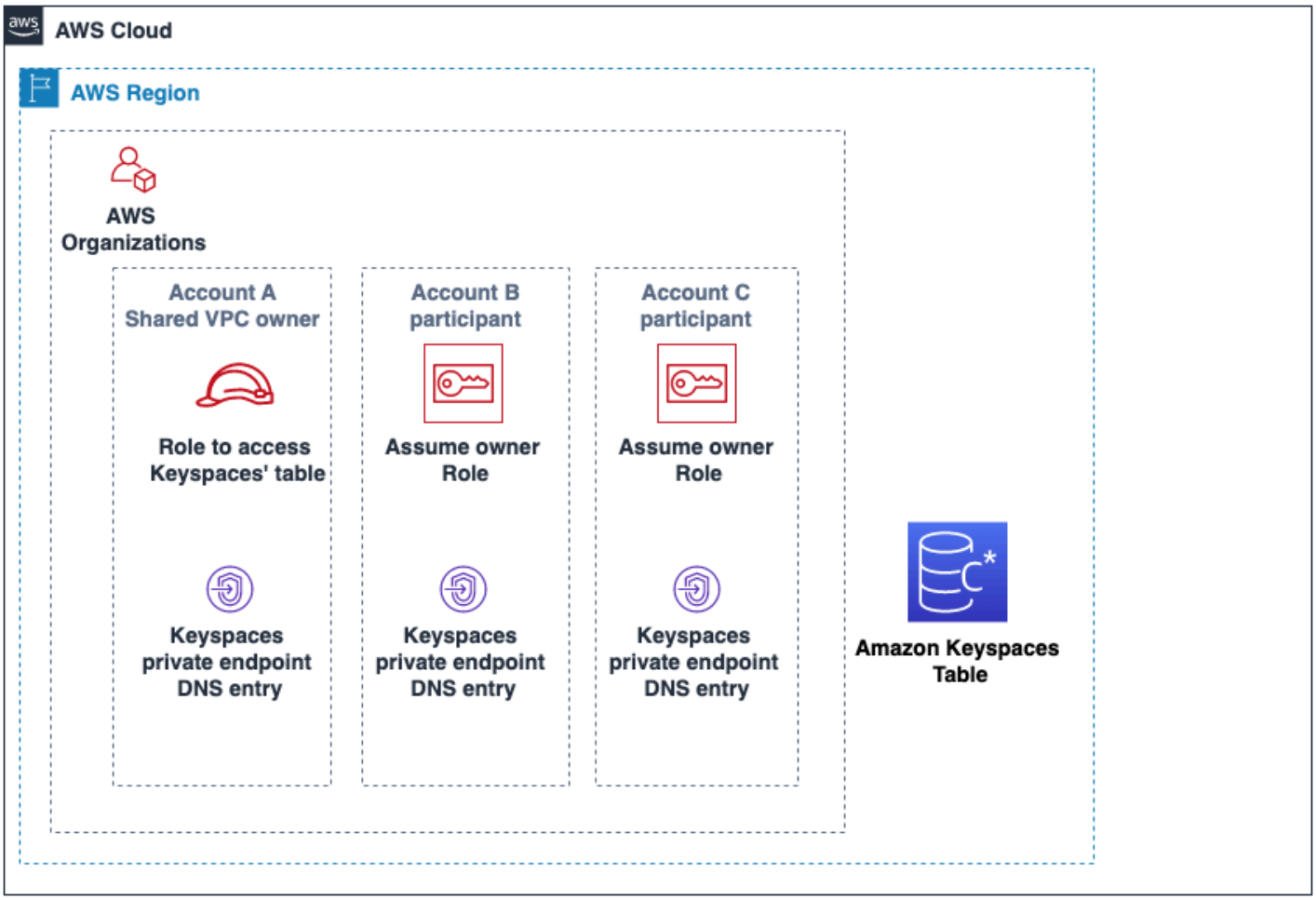
   El permiso para que la entidad principal asuma el rol compartido se implementa con la API `AssumeRole` de la AWS Security Token Service (AWS STS). Para obtener más información, consulte Concesión de acceso a un usuario de IAM en otra Cuenta de AWS de su propiedad en la Guía del usuario de IAM.

2. En `Account B` y `Account C`, puede crear aplicaciones que utilicen el complemento de autenticación SIGV4, que permite que una aplicación asuma el rol compartido para conectarse a la tabla de Amazon Keyspaces ubicada en `Account A` a través del punto de conexión de VPC en la VPC compartida. Para obtener más información sobre el complemento de autenticación SIGV4, consulte the section called “Creación de credenciales”.

Configuración del acceso entre cuentas para Amazon Keyspaces sin una VPC compartida

Si la tabla de Amazon Keyspaces y el punto de conexión de VPC privado pertenecen a cuentas diferentes pero no comparten una VPC, las aplicaciones podrán seguir conectándose entre cuentas mediante puntos de conexión de VPC. Dado que las cuentas no comparten los puntos de conexión de VPC, `Account A`, `Account B` y `Account C` requieren sus propios puntos de conexión de VPC. Para el controlador de cliente de Cassandra, Amazon Keyspaces aparece como un único nodo en lugar de un clúster multinodo. Al conectarse, el controlador de cliente llega al servidor DNS que devuelve uno de los puntos de conexión disponibles en la VPC de la cuenta.

También puede acceder a las tablas de Amazon Keyspaces a través de diferentes cuentas sin un punto de conexión de VPC compartido utilizando el punto de conexión público o implementando un punto de conexión de VPC privado en cada cuenta. Cuando no se utiliza una VPC compartida, cada cuenta requiere su propio punto de conexión de VPC. En este ejemplo, `Account A`, `Account B` y `Account C` requieren sus propios puntos de conexión de VPC para acceder a la tabla de `Account A`. Cuando se utilizan puntos de conexión de VPC en esta configuración, Amazon Keyspaces aparece como un clúster de nodo único para el controlador de cliente de Cassandra en lugar de un clúster multinodo. Al conectarse, el controlador de cliente llega al servidor DNS que devuelve uno de los puntos de conexión disponibles en la VPC de la cuenta. Sin embargo, el controlador de cliente no puede acceder a la tabla de `system.peers` para descubrir puntos de conexión adicionales. Dado que hay menos hosts disponibles, el controlador realiza menos conexiones. Para ajustar esto, aumente la configuración del grupo de conexiones del controlador en un factor de tres.


AWS Cloud
AWS Region
AWS Organizations
Account A
Shared VPC owner
Role to access Keyspaces' table
Keyspaces private endpoint DNS entry
Account B
participant
Assume owner Role
Keyspaces private endpoint DNS entry
Account C
participant
Assume owner Role
Keyspaces private endpoint DNS entry
Amazon Keyspaces Table

Introducción a Amazon Keyspaces (para Apache Cassandra)

Este tutorial es ideal para usted si es nuevo en Apache Cassandra y Amazon Keyspaces (para Apache Cassandra). En este tutorial, usted instala todos los programas y controladores que necesita para utilizar con éxito Amazon Keyspaces.

Para obtener tutoriales para conectarse mediante programación a Amazon Keyspaces utilizando diferentes controladores de cliente de Cassandra, consulte the section called “Utilización de un controlador de cliente de Cassandra”.

Temas

- Requisitos y consideraciones previos del tutorial
- Paso 1 del tutorial: Crear un espacio de claves y una tabla en Amazon Keyspaces
- Paso 2 del tutorial: Crear, leer, actualizar y eliminar datos (CRUD)
- Paso 1 del tutorial: Crear una tabla y un espacio de claves en Amazon Keyspaces

Requisitos y consideraciones previos del tutorial

Antes de empezar este tutorial, siga las instrucciones AWS de configuración incluidas enAcceso a Amazon Keyspaces (para Apache Cassandra). Estos pasos incluyen registrarse AWS y crear un usuario AWS Identity and Access Management (IAM) con acceso a Amazon Keyspaces.

Además, si va a completar el tutorial utilizando `cqlsh` o un controlador de cliente de Cassandra con licencia Apache 2.0, complete las instrucciones de configuración indicadas en Uso de `cqlsh` para conectarse a Amazon Keyspaces.

Tras completar los pasos de requisitos previos, continúe con Paso 1 del tutorial: Crear un espacio de claves y una tabla en Amazon Keyspaces.

Paso 1 del tutorial: Crear un espacio de claves y una tabla en Amazon Keyspaces

En esta sección, usted crea un espacio de claves y le añade una tabla utilizando la consola.

> **Note**
>
> Antes de comenzar, asegúrese de haber configurado todos los requisitos previos del tutorial.

Temas

- Creación de un espacio de claves
- Creación de una tabla

Creación de un espacio de claves

Un espacio de claves agrupa tablas relacionadas que son relevantes para una o más aplicaciones. Un espacio de claves contiene una o varias tablas y define la estrategia de réplica para todas las tablas que contiene. Para obtener más información sobre los espacios de claves, consulte los siguientes temas:

- Uso de los espacios de claves: the section called “Creación de espacios de claves”
- Instrucciones del lenguaje de definición de datos (DDL): Keyspaces
- Cuotas para Amazon Keyspaces (para Apache Cassandra)

Al crear un espacio de claves, debe especificar el nombre del espacio de claves.

> **Note**
>
> La estrategia de réplica del espacio de claves debe ser `SingleRegionStrategy`. `SingleRegionStrategy` replica los datos en tres zonas de disponibilidad en su Región de AWS.

Uso de la consola

Para crear un espacio de claves con la consola

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Espacios de claves.

3. Elija Crear espacio de claves.
4. En el cuadro Nombre del espacio de claves, introduzca **`myGSGKeyspace`** como nombre para su espacio de claves.

   Restricciones del nombre:

   - No puede estar vacío.
   - Caracteres permitidos: alfanuméricos y guion bajo (_).
   - La longitud máxima es de 48 caracteres.
5. Para crear el espacio de claves, elija Crear espacio de claves.
6. Para verificar que el espacio de claves `myGSGKeyspace` se haya creado, haga lo siguiente:

   a. En el panel de navegación, elija Espacios de claves.

   b. Localice su espacio de claves `myGSGKeyspace` en la lista de espacios de claves.

Uso de CQL

El siguiente procedimiento crea un espacio de claves utilizando CQL.

Para crear un espacio de claves con CQL

1. Abra un intérprete de comandos e introduzca lo siguiente:

   `cqlsh`
2. Cree su espacio de claves utilizando el siguiente comando CQL.

   ```
   CREATE KEYSPACE IF NOT EXISTS "myGSGKeyspace"
      WITH REPLICATION = {'class': 'SingleRegionStrategy'};
   ```

   `SingleRegionStrategy`utiliza un factor de replicación de tres y replica los datos en tres zonas de AWS disponibilidad de su región.

   > **Note**
   >
   > Amazon Keyspaces pone de forma predeterminada todas las entradas en minúsculas a menos que las encierre entre comillas. En este caso, introduzca `"myGSGKeyspace"`.
3. Compruebe que se haya creado su espacio de claves.

```
SELECT * from system_schema.keyspaces ;
```

Su espacio de claves debería aparecer en la lista.

Creación de una tabla

Una tabla es el lugar donde se organizan y almacenan sus datos. La clave principal de su tabla determina cómo se particionan los datos en su tabla. La clave principal se compone de una clave de partición obligatoria y una o varias columnas de agrupación opcionales. Los valores combinados que componen la clave principal deben ser únicos en todos los datos de la tabla. Para obtener más información sobre las tablas, consulte los siguientes temas:

- Uso de tablas: the section called “Creación de tablas”
- Instrucciones DDL: Tablas
- Administración de recursos de la tabla: Administración de recursos sin servidor
- Monitoreo de utilización de recursos de la tabla: the section called “Monitorización con CloudWatch”
- Cuotas para Amazon Keyspaces (para Apache Cassandra)

Al crear una tabla, especifica lo siguiente:

- El nombre de la tabla.
- El nombre y el tipo de datos de cada columna de la tabla.
- La clave principal de la tabla.
  - Clave de partición: obligatorio
  - Columnas de agrupación: opcional

Utilice el siguiente procedimiento para crear una tabla con las columnas, los tipos de datos, la clave de partición y la columna de agrupación especificados.

Uso de la consola

El siguiente procedimiento crea la tabla `employees_tbl` con estas columnas y tipos de datos.

```
ID           text
```

```
name          text
region        text
division      text
project       text
role          text
pay_scale     int
vacation_hrs  float
manager_id    text
```

Para crear una tabla con la consola

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Espacios de claves.
3. Elija `myGSGKeyspace` como el espacio de claves en el que desea crear esta tabla.
4. Elija Crear tabla.
5. En el cuadro Nombre de la tabla, introduzca **`employees_tbl`** como nombre para la tabla.

   Restricciones del nombre:

   - No puede estar vacío.
   - Caracteres permitidos: alfanuméricos y guion bajo (_).
   - La longitud máxima es de 48 caracteres.
6. En la sección Columnas, repita los siguientes pasos para cada columna que desee añadir a esta tabla.

   Añada las columnas y tipos de datos siguientes.

   ```
   id            text
   name          text
   region        text
   division      text
   project       text
   role          text
   pay_scale     int
   vacation_hrs  float
   manager_id    text
   ```

   a. Nombre: introduzca un nombre para la columna.

Restricciones del nombre:

- No puede estar vacío.
- Caracteres permitidos: alfanuméricos y guion bajo (_).
- La longitud máxima es de 48 caracteres.

b. Tipo: en la lista de tipos de datos, elija el tipo de datos para esta columna.

c. Si desea añadir otra columna, elija Añadir columna.

7. Elija `id` como clave de partición en Clave de partición. Se necesita una clave de partición para cada tabla. Una clave de partición puede estar formada por una o más columnas.
8. Añada `division` como columna de agrupación. Las columnas de agrupación son opcionales y determinan el orden de clasificación dentro de cada partición.

   a. Para añadir una columna de agrupación, elija Añadir columna de agrupación.

   b. En la lista Columna, elija division. En la lista Orden, elija ASC para clasificar en orden ascendente los valores de esta columna. (Elija DESC para orden descendente).
9. En la sección Configuración de la tabla, elija Configuración predeterminada.
10. Elija Crear tabla.
11. Verifique que se haya creado su tabla.

    a. En el panel de navegación, elija Tablas.

    b. Confirme que su tabla se encuentre en la lista de tablas.

    c. Elija el nombre de su tabla.

    d. Confirme que todas sus columnas y tipos de datos sean correctos.

> **Note**
>
> Es posible que las columnas no aparezcan en el mismo orden en que las añadió a la tabla.

    e. En la columna de agrupación, confirme que division se identifique con true. Todas las demás columnas de la tabla deben ser false.

Uso de CQL

El siguiente procedimiento crea una tabla con las siguientes columnas y tipos de datos utilizando CQL. La columna `id` va a ser la clave de partición.

```
id            text
name          text
region        text
division      text
project       text
role          text
pay_scale     int
vacation_hrs  float
manager_id    text
```

Para crear una tabla utilizando CQL

1. Abra un intérprete de comandos e introduzca lo siguiente:

   `cqlsh`

2. En la línea de comandos de `cqlsh` (`cqlsh>`), especifique un espacio de claves en el que crear su tabla.

   ```
   USE "myGSGKeyspace" ;
   ```

3. En la línea de comandos del espacio de claves (`cqlsh:keyspace_name>`), cree su tabla introduciendo el siguiente código en su ventana de comandos.

   ```
   CREATE TABLE IF NOT EXISTS "myGSGKeyspace".employees_tbl (
      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 CLUSTERING ORDER BY (division ASC) ;
   ```

> **Note**
>
> ASC es el orden de agrupación predeterminado. También puede especificar DESC para el orden descendente.

Tenga en cuenta que la columna `id` va a ser la clave de partición. Además, `division` es la columna de agrupación por orden ascendente (ASC).

4. Verifique que se haya creado su tabla.

```
SELECT * from system_schema.tables WHERE keyspace_name='myGSGKeyspace' ;
```

Su tabla debería aparecer en la lista.

5. Verifique la estructura de su tabla.

```
SELECT * FROM system_schema.columns WHERE keyspace_name = 'myGSGKeyspace' AND
 table_name = 'employees_tbl' ;
```

Confirme que todas las columnas y tipos de datos sean los deseados. El orden de las columnas podría diferir al de la instrucción CREATE.

Para realizar operaciones CRUD (crear, leer, actualizar y eliminar) en los datos de su tabla, vaya a the section called “Paso 2: Operaciones CRUD”.

Paso 2 del tutorial: Crear, leer, actualizar y eliminar datos (CRUD)

En esta sección, usted utiliza el editor CQL de la consola para realizar operaciones CRUD (crear, leer, actualizar y eliminar) en los datos de su tabla. También puede ejecutar los comandos con `cqlsh`.

Temas

- Tutorial: Inserción y carga de datos en una tabla de Amazon Keyspaces
- Tutorial: Lectura desde una tabla de Amazon Keyspaces
- Tutorial: Actualización de datos en una tabla de Amazon Keyspaces
- Tutorial: Eliminación de datos em una tabla de Amazon Keyspaces

Tutorial: Inserción y carga de datos en una tabla de Amazon Keyspaces

Para crear datos en su tabla `employees_tbl`, utilice la instrucción `INSERT` para añadir una única fila.

1. Para poder escribir datos en su tabla Amazon Keyspaces con cqlsh, antes debe establecer la coherencia de escritura de la sesión actual de cqlsh en `LOCAL_QUORUM`. Para obtener más información sobre los niveles de coherencia admitidos, consulte the section called “Niveles de coherencia de escritura”. Tenga en cuenta que este paso no es necesario si utiliza el editor CQL en la AWS Management Console.

```
CONSISTENCY LOCAL_QUORUM;
```

2. Para insertar un único registro, ejecute el siguiente comando en el editor CQL.

```
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') ;
```

3. Verifique que los datos se hayan añadido correctamente a su tabla; para ello, ejecute el siguiente comando.

```
SELECT * FROM "myGSGKeyspace".employees_tbl ;
```

Para insertar varios registros desde un archivo con cqlsh

1. Descargue el archivo de datos de ejemplo (employees.csv) contenido en el siguiente archivo: sampledata.zip. Este archivo CSV (valores separados por comas) contiene los siguientes datos. Recuerde la ruta en la que guarde el archivo.

ID	name	project	region	division	role	pay_scale	vacation_hrs	manager_id
123-45-6789	Bob	NightFlight	US	Engineering	Intern	1	0	234-56-7890
234-56-7890	Bob	NightFlight	US	Engineering	Manager	6	72	789-01-2345
345-67-8901	Sarah	Storm	US	Engineering	IC	4	108	234-56-7890
456-78-9012	Beth	NightFlight	US	Engineering	IC	7	100.5	234-56-7890
567-89-0123	Ahmed	NightFlight	US	Marketing	IC	4	88	678-90-1234
678-90-1234	Alan	Storm	US	Marketing	Manager	3	18.4	789-01-2345
789-01-2345	Roberta	All	US	Executive	CEO	15	184	None

2. Abra un intérprete de comandos e introduzca lo siguiente:

cqlsh

3. En la línea de comandos de `cqlsh` (`cqlsh>`), especifique un espacio de claves.

```
USE "myGSGKeyspace" ;
```

4. Establezca la coherencia de escritura en `LOCAL_QUORUM`. Para obtener más información sobre los niveles de coherencia admitidos, consulte the section called “Niveles de coherencia de escritura”.

```
CONSISTENCY LOCAL_QUORUM;
```

5. En la línea de comandos de espacio de claves (`cqlsh:keyspace_name>`), ejecute la siguiente consulta.

```
COPY employees_tbl
 (id,name,project,region,division,role,pay_scale,vacation_hrs,manager_id)
FROM 'path-to-the-csv-file/employees.csv' WITH delimiter=',' AND header=TRUE ;
```

6. Verifique que los datos se hayan añadido correctamente a su tabla; para ello, ejecute la siguiente consulta.

```
SELECT * FROM employees_tbl ;
```

Tutorial: Lectura desde una tabla de Amazon Keyspaces

En la sección Tutorial: Inserción y carga de datos en una tabla de Amazon Keyspaces, utilizó la instrucción `SELECT` para verificar que ha añadido correctamente los datos a su tabla. En esta sección, usted restringe el uso de `SELECT` para mostrar columnas específicas y solo las filas que cumplan criterios específicos.

La forma general de la instrucción `SELECT` es la siguiente.

```
SELECT column_list FROM table_name [WHERE condition [ALLOW FILTERING]] ;
```

Temas

- Selección de todos los datos de su tabla
- Selección de un subconjunto de columnas

- Selección de un subconjunto de filas

Selección de todos los datos de su tabla

La forma más sencilla de la instrucción `SELECT` devuelve todos los datos de su tabla.

> **Important**
>
> En un entorno de producción, no suele ser una práctica recomendada ejecutar este comando, dado que devuelve todos los datos de su tabla.

Para seleccionar todos los datos de su tabla

- Ejecute la siguiente consulta .

```
SELECT * FROM "myGSGKeyspace".employees_tbl ;
```

  El uso del carácter comodín (`*`) para `column_list` selecciona todas las columnas.

Selección de un subconjunto de columnas

Para consultar un subconjunto de columnas

- Para recuperar solo las columnas `id`, `name` y `manager_id`, ejecute la siguiente consulta.

```
SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

  La salida contendrá solo las columnas especificadas en el orden indicado en la instrucción `SELECT`.

Selección de un subconjunto de filas

Al consultar un gran conjunto de datos, es posible que solo desee registros que cumplan determinados criterios. Para ello, puede añadir una cláusula `WHERE` al final de nuestra instrucción `SELECT`.

Para consultar un subconjunto de filas

- Para recuperar solo el registro del empleado con el id `'234-56-7890'`, ejecute la siguiente consulta.

  ```
  SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='234-56-7890' ;
  ```

  La instrucción `SELECT` anterior devuelve solo las filas en las que `id` es `234-56-7890`.

Comprensión de la cláusula **`WHERE`**

La cláusula `WHERE` se utiliza para filtrar los datos y devolver solo los que cumplan los criterios especificados. Los criterios especificados pueden ser una condición simple o una condición compuesta.

Uso de condiciones en una cláusula **`WHERE`**

- Una condición simple: una única columna.

  ```
  WHERE column_name=value
  ```

  Puede utilizar una condición simple en una cláusula `WHERE` si se cumple alguna de las siguientes condiciones:

  - La columna es la única columna de la clave principal de la tabla.
  - Añada `ALLOW FILTERING` después de la condición en la cláusula `WHERE`.

    Tenga en cuenta que el uso de `ALLOW FILTERING` puede dar lugar a un rendimiento incoherente, sobre todo con tablas de gran tamaño y con varias particiones.
- Una condición compuesta: múltiples condiciones simples conectadas por `AND`.

  ```
  WHERE column_name1=value1 AND column_name2=value2 AND column_name3=value3...
  ```

  Puede utilizar condiciones compuestas en una cláusula `WHERE` si se cumple alguna de las siguientes condiciones:

  - Las columnas de la cláusula `WHERE` coinciden exactamente con las columnas de la clave principal de la tabla, ni más ni menos.

- Añada `ALLOW FILTERING` después de la condición compuesta en la cláusula `WHERE`, como en el siguiente ejemplo.

  ```
  SELECT * FROM my_table WHERE col1=5 AND col2='Bob' ALLOW FILTERING ;
  ```

  Tenga en cuenta que el uso de `ALLOW FILTERING` puede dar lugar a un rendimiento incoherente, sobre todo con tablas de gran tamaño y con varias particiones.

Pruébelo

Cree sus propias consultas CQL para obtener los siguientes resultados de su tabla `employees_tbl`:

- Encuentre los `name`, `project` y `id` de todos los empleados.
- Encuentre en qué proyecto está trabajando el becario `Bob` (incluya al menos su nombre, proyecto y función en el resultado).
- Avanzado: cree una aplicación para encontrar a todos los empleados que tengan el mismo administrador que el becario `Bob`. SUGERENCIA: Esto podría requerir más de una consulta.
- Avanzado: cree una aplicación para encontrar las columnas seleccionadas de todos los empleados que trabajan en el proyecto `NightFlight`. SUGERENCIA: Resolver esto podría requerir múltiples instrucciones.

Tutorial: Actualización de datos en una tabla de Amazon Keyspaces

Para actualizar los datos de su tabla `employees_tbl`, utilice la instrucción `UPDATE`.

La forma general de la instrucción `UPDATE` es la siguiente.

```
UPDATE table_name SET column_name=new_value WHERE primary_key=value ;
```

Tip

- Puede actualizar múltiples columnas utilizando una lista separada por comas de `column_names` y valores, como en el siguiente ejemplo.

  ```
  UPDATE my_table SET col1='new_value_1', col2='new_value2' WHERE id='12345' ;
  ```

- Si la clave principal se compone de múltiples columnas, todas las columnas de la clave principal y sus valores deben incluirse en la cláusula `WHERE`.
- No puede actualizar ninguna columna de la clave principal porque eso cambiaría la clave principal del registro.

Para actualizar una única celda

Utilice su tabla `employees_tbl` para dar al empleado con id `567-89-0123` un aumento de sueldo.

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale=5 WHERE id='567-89-0123' AND
 division='Marketing' ;
```

Compruebe que la escala salarial del empleado sea ahora 5.

```
SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='567-89-0123' ;
```

Pruébelo

Avanzado: su empresa ha contratado al becario Bob. Cambie su registro para que su función sea `'IC'` y su escala salarial sea 2.

Tutorial: Eliminación de datos em una tabla de Amazon Keyspaces

Para eliminar datos en su tabla `employees_tbl`, utilice la instrucción `DELETE`.

Puede eliminar datos de una fila o de una partición. Tenga cuidado al eliminar datos, dado que la eliminación es irreversible.

Al eliminar una o todas las filas de una tabla, no elimina la tabla. Por tanto, puede volver a llenarla con datos. Al eliminar una tabla, elimina la tabla y todos los datos que contiene. Para volver a utilizar la tabla, debe volver a crearla y añadirle datos. Al eliminar un espacio de claves, elimina el espacio de claves y todas las tablas que contiene. Para volver a utilizar el espacio de claves y las tablas, debe volver a crearlos y rellenarlos con datos.

Eliminación de celdas

Al eliminar una columna de una fila se eliminan los datos de la celda especificada. Al visualizar esa columna con una instrucción `SELECT`, los datos se muestran como *`null`*, si bien en esa ubicación no se almacena un valor null.

La sintaxis general para eliminar una o varias columnas en concreto es la siguiente.

```
DELETE column_name1[, column_name2...] FROM table_name WHERE condition ;
```

En su tabla `employees_tbl`, puede ver que el CEO tiene "None" para un administrador. En primer lugar, elimine esa celda para no tener ningún dato en ella.

Para eliminar una celda en concreto

1. Ejecute la siguiente consulta `DELETE`.

   ```
   DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
    division='Executive';
   ```

2. Compruebe que la eliminación se haya realizado como se esperaba.

   ```
   SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
    division='Executive';
   ```

Eliminación de filas

Podría darse el caso de que necesite eliminar una fila entera, como cuando un empleado se jubila. La sintaxis general para eliminar una fila es la siguiente.

```
DELETE FROM table_name WHERE condition ;
```

Para eliminar una fila

1. Ejecute la siguiente consulta `DELETE`.

   ```
   DELETE FROM "myGSGKeyspace".employees_tbl WHERE id='456-78-9012' AND
    division='Engineering';
   ```

2. Compruebe que la eliminación se haya realizado como se esperaba.

   ```
   SELECT * FROM "myGSGKeyspace".employees_tbl WHERE id='456-78-9012' AND
    division='Engineering';
   ```

Paso 1 del tutorial: Crear una tabla y un espacio de claves en Amazon Keyspaces

Para evitar que le cobren por tablas y datos que no necesite, elimine todas las tablas y espacios de claves que no utilice. Al eliminar una tabla, la tabla y sus datos se eliminan y deja de acumular cargos por ellos. Sin embargo, el espacio de claves permanece. Al eliminar un espacio de claves, el espacio de claves y todas sus tablas se eliminan y deja de acumular cargos por ellos.

Eliminación de una tabla

Puede eliminar una tabla utilizando la consola o CQL. Al eliminar una tabla, se eliminan la tabla y todos sus datos.

Uso de la consola

El siguiente procedimiento elimina una tabla y todos sus datos utilizando la AWS Management Console.

Para eliminar una tabla con la consola

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tablas.
3. Elija la casilla situada a la izquierda del nombre de cada tabla que desee eliminar.
4. Elija Eliminar.
5. En la pantalla Eliminar tabla, introduzca **`Delete`** en el cuadro. A continuación, elija Eliminar tabla.
6. Para verificar la eliminación de la tabla, elija Tablas en el panel de navegación y confirme que la tabla `employees_tbl` ya no aparezca en la lista.

Uso de CQL

El siguiente procedimiento elimina una tabla y todos sus datos utilizando CQL.

Elimine una tabla con CQL

1. Abra un intérprete de comandos e introduzca lo siguiente:

cqlsh

2. Para eliminar su tabla, introduzca el siguiente comando en la línea de comandos del espacio de claves (`cqlsh:keyspace_name>`).

```
DROP TABLE IF EXISTS "myGSGKeyspace".employees_tbl ;
```

3. Verifique que se haya eliminado su tabla.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = 'myGSGKeyspace' ;
```

Su tabla no debería aparecer en la lista.

Eliminación de un espacio de claves

Puede eliminar un espacio de claves mediante el CQL AWS Management Console o el CQL. Al eliminar un espacio de claves, se eliminan el espacio de claves y todas sus tablas y datos.

Uso de la AWS Management Console

El siguiente procedimiento elimina un espacio de claves y todas sus tablas y datos utilizando la AWS Management Console.

Para eliminar un espacio de claves con la consola

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Espacios de claves.
3. Elija la casilla situada a la izquierda del nombre de cada espacio de claves que desee eliminar.
4. Elija Eliminar.
5. En la pantalla Eliminar espacio de claves, introduzca **`Delete`** en el cuadro. A continuación, elija Eliminar espacio de claves.
6. Para verificar la eliminación del espacio de claves `myGSGKeyspace`, elija Espacios de claves en el panel de navegación y confirme que ya no aparezca en la lista. Dado que ha eliminado su espacio de claves, la tabla `employees_tbl` en Tablas tampoco debería aparecer en la lista.

Uso de CQL

El siguiente procedimiento elimina un espacio de claves y todas sus tablas y datos utilizando CQL.

Para eliminar un espacio de claves con CQL

1. Abra un intérprete de comandos e introduzca lo siguiente:

   `cqlsh`

2. Para eliminar su espacio de claves, introduzca el siguiente comando en la línea de comandos del espacio de claves (`cqlsh:`*`keyspace_name`*>).

   ```
   DROP KEYSPACE IF EXISTS "myGSGKeyspace" ;
   ```

3. Compruebe que su espacio de claves haya sido eliminado.

   ```
   SELECT * from system_schema.keyspaces ;
   ```

   Su espacio de claves no debería aparecer en la lista. Tenga en cuenta que, dado que se trata de una operación asíncrona, puede haber un retraso hasta que se elimine el espacio de claves.

Migración a Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) es un servicio de base de datos escalable, altamente disponible y administrado compatible con Apache Cassandra. Puede migrar sus datos a Amazon Keyspaces desde bases de datos de Cassandra que se ejecuten en las instalaciones o en Amazon Elastic Compute Cloud (Amazon EC2) realizando los pasos que se indican en esta sección.

Le recomendamos que siga estas prácticas recomendadas para garantizar el éxito de la migración:

- Divida la migración en componentes más pequeños.

  Considere las siguientes unidades de migración y su huella potencial en términos de tamaño de datos en bruto. Migrar cantidades más pequeñas de datos en una o varias fases puede ayudar a simplificar su migración.

  Por clúster: migre todos sus datos de Cassandra a la vez. Este enfoque podría ser adecuado para clústeres pequeños.

  Por espacio de claves o tabla: divida su migración en grupos de espacios de claves o tablas. Este enfoque puede ayudarle a migrar los datos por fases en función de sus necesidades para cada carga de trabajo.

  Por datos: considere la posibilidad de migrar los datos de un grupo específico de usuarios o productos, para reducir aún más el tamaño de los datos.
- Priorice qué datos migrar primero en función de la simplicidad.

  Considere si tiene datos que podrían migrarse primero con más facilidad; por ejemplo, datos que no cambien durante horas específicas, datos de trabajos por lotes nocturnos, datos que no se utilicen durante horas sin conexión o datos de aplicaciones internas.
- Utilice herramientas específicas.
  - Comience sin demora a cargar datos en Amazon Keyspaces con el comando `COPY FROM` de cqlsh. cqlsh se incluye con Apache Cassandra y es el más adecuado para cargar pequeños conjuntos de datos o datos de prueba. Para obtener instrucciones paso a paso, consulte the section called “Carga de datos con cqlsh”.
  - Para cargas de trabajo de producción con grandes conjuntos de datos, puede utilizar DataStax Bulk Loader para Apache Cassandra para cargar datos en Amazon Keyspaces mediante el comando `dsbulk`. DSBulk proporciona capacidades de importación más robustas y está

disponible en el repositorio GitHub. Para obtener instrucciones paso a paso, consulte the section called “Carga de datos con DSBulk”.

- Para obtener información sobre cómo utilizar el conector Apache Cassandra Spark para escribir datos en Amazon Keyspaces, consulte Integración con Apache Spark.
- Para migraciones complejas, considere la posibilidad de utilizar una herramienta de extracción, transformación y carga (ETL). Puede utilizar AWS Glue para realizar migraciones de transformación de datos de forma rápida y eficaz. Para obtener más información, consulte Migración de cargas de trabajo de Apache Cassandra a Amazon Keyspaces con AWS Glue.

Temas

- Tutorial: Carga de datos en Amazon Keyspaces utilizando cqlsh
- Tutorial: Carga de datos en Amazon Keyspaces utilizando DSBulk

Tutorial: Carga de datos en Amazon Keyspaces utilizando cqlsh

Este step-by-step tutorial le guía a través de la migración de datos de Apache Cassandra a Amazon Keyspaces mediante el comando. `cqlsh COPY` En este tutorial, aprenderá a hacer lo siguiente:

Temas

- Requisitos previos
- Paso 1: Crear el archivo CSV de origen y la tabla de destino
- Paso 2: Preparar los datos
- Paso 3: Establecer la capacidad de rendimiento de la tabla
- Paso 4: Configurar los ajustes de cqlsh COPY FROM
- Paso 5: Ejecutar el comando cqlsh COPY FROM
- Solución de problemas

Requisitos previos

Para poder comenzar el tutorial, antes debe completar las siguientes tareas.

1. Si aún no lo ha hecho, inscríbase en uno Cuenta de AWS siguiendo los pasos que se indican en. the section called “Con AWS Identity and Access Management figuración”

2. Cree las credenciales específicas del servicio; para ello, siga los pasos indicados en the section called “Generación de credenciales específicas del servicio mediante la consola”.
3. Configure la conexión del intérprete de comandos de Cassandra Query Language (cqlsh) y confirme que puede conectarse a Amazon Keyspaces siguiendo los pasos indicados en the section called “Uso de `cqlsh`”.

Paso 1: Crear el archivo CSV de origen y la tabla de destino

En este tutorial, utilizamos un archivo de valores separados por comas (CSV) llamado `keyspaces_sample_table.csv` como archivo de origen para la migración de datos. El archivo de ejemplo proporcionado contiene algunas filas de datos de una tabla llamada `book_awards`.

1. Cree el archivo de origen. Puede elegir una de las siguientes opciones:
   - Descargue el archivo CSV de ejemplo (`keyspaces_sample_table.csv`), contenido en el siguiente archivo comprimido: samplemigration.zip. Descomprima el archivo y tome nota de la ruta a `keyspaces_sample_table.csv`.
   - Para rellenar un archivo CSV de origen con sus propios datos almacenados en una base de datos de Apache Cassandra, utilice la instrucción `cqlsh COPY TO`, como se muestra en el siguiente ejemplo.

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

     Asegúrese de que el archivo CSV que cree satisfaga los siguientes requisitos:

     - La primera fila contiene los nombres de las columnas.
     - Los nombres de las columnas del archivo CSV de origen coinciden con los nombres de las columnas de la tabla de destino.
     - Los datos están delimitados con una coma.
     - Todos los valores de los datos son tipos de datos válidos de Amazon Keyspaces. Consulte the section called “Tipos de datos”.
2. Cree el espacio de claves y la tabla de destino en Amazon Keyspaces.
   a. Conéctese a Amazon Keyspaces con `cqlsh` y sustituya el punto de conexión del servicio, el nombre de usuario y la contraseña del siguiente ejemplo por sus propios valores.

```
cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

b. Cree un nuevo espacio de claves con el nombre `catalog` como se muestra en el siguiente ejemplo.

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

c. Una vez que el nuevo espacio de claves, utilice el siguiente código para crear la tabla de destino `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)
   );
```

Si Apache Cassandra es su origen de datos, una forma sencilla de crear la tabla de destino de Amazon Keyspaces con encabezados que coincidan es generar la instrucción `CREATE TABLE` a partir de la tabla de origen, como se muestra en la siguiente instrucción.

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

A continuación, cree la tabla de destino en Amazon Keyspaces con los nombres de las columnas y los tipos de datos que coincidan con la descripción de la tabla de origen de Cassandra.

Paso 2: Preparar los datos

La preparación de los datos de origen para una transferencia eficaz es un proceso de dos pasos. En primer lugar, usted aleatoriza los datos. En segundo lugar, usted analiza los datos para determinar los valores apropiados de los parámetros de `cqlsh` y los ajustes necesarios de la tabla.

Aleatorización de los datos

El comando `cqlsh COPY FROM` lee y escribe los datos en el mismo orden en que aparecen en el archivo CSV. Si utiliza el comando `cqlsh COPY TO` para crear el archivo de origen, los datos se escriben en el orden en que aparecen ordenados por claves en el CSV. Internamente, Amazon Keyspaces divide los datos utilizando claves de partición. Si bien Amazon Keyspaces tiene una lógica integrada para ayudar a equilibrar la carga de las solicitudes de la misma clave de partición, la carga de datos es más rápida y eficiente si aleatoriza el orden. Esto se debe a que puede aprovechar el equilibrio de carga integrado que se produce cuando Amazon Keyspaces escribe en diferentes particiones.

Para repartir las escrituras entre las particiones de manera uniforme, debe aleatorizar los datos en el archivo de origen. Puede escribir una aplicación para hacerlo o utilizar una herramienta de código abierto, como Shuf. Shuf está disponible de forma gratuita en distribuciones Linux, en macOS (instalando coreutils en homebrew) y en Windows (utilizando Windows Subsystem for Linux [WSL]). Se requiere un paso adicional para evitar que la fila del encabezado con los nombres de las columnas se baraje en este paso.

Para aleatorizar el archivo de origen conservando el encabezado, introduzca el siguiente código.

```
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 reescribe los datos en un nuevo archivo CSV llamado `keyspace.table.csv`. Ahora puede eliminar el archivo `keyspaces_sample_table.csv`, dado que ya no lo necesita.

Análisis de los datos

Determine el tamaño medio y máximo de las filas analizando los datos.

Debe hacerlo por las siguientes razones:

- El tamaño medio de fila ayuda a estimar la cantidad total de datos que se van a transferir.

- Necesita el tamaño medio de fila para aprovisionar la capacidad de escritura necesaria para la carga de datos.
- Puede asegurarse de que cada fila tenga un tamaño inferior a 1 MB, que es el tamaño máximo de fila en Amazon Keyspaces.

> **Note**
>
> Esta cuota se refiere al tamaño de fila, no al de la partición. A diferencia de las particiones de Apache Cassandra, las particiones de Amazon Keyspaces pueden tener un tamaño prácticamente ilimitado. Las claves de partición y las columnas de agrupación requieren almacenamiento adicional para los metadatos, que debe añadir al tamaño bruto de las filas. Para obtener más información, consulte the section called “Cálculo del tamaño de las filas”.

El siguiente código utiliza AWK para analizar un archivo CSV e imprimir el tamaño medio y máximo de las filas.

```
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
```

La ejecución de este código da como resultado la siguiente salida.

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

El tamaño medio de fila se utiliza en el siguiente paso de este tutorial para aprovisionar la capacidad de escritura de la tabla.

Paso 3: Establecer la capacidad de rendimiento de la tabla

En este tutorial se muestra cómo ajustar cqlsh para que cargue los datos dentro de un intervalo de tiempo establecido. Dado que sabe de antemano cuántas lecturas y escrituras realiza, utilice el modo de capacidad aprovisionada. Una vez finalizada la transferencia de datos, debe ajustar el modo de capacidad de la tabla para que se adapte a los patrones de tráfico de su aplicación. Para obtener más información sobre administración de capacidad, consulte Administración de recursos sin servidor.

Con el modo de capacidad aprovisionada, usted especifica con antelación cuánta capacidad de lectura y escritura desea aprovisionar a su tabla. La capacidad de escritura se factura por hora y se mide en unidades de capacidad de escritura (WCU). Cada WCU es capacidad de escritura suficiente para admitir la escritura de 1 KB de datos por segundo. Al cargar los datos, la tasa de escritura debe estar por debajo de las WCU máximas (parámetro: `write_capacity_units`) que se hayan establecido en la tabla de destino.

De forma predeterminada, puede aprovisionar hasta 40 000 WCU en una tabla y 80 000 WCU en todas las tablas de su cuenta. Si necesita capacidad adicional, puede solicitar un aumento de cuota en la consola de Service Quotas. Para obtener más información acerca de las cuotas, consulte Cuotas.

Cálculo del número medio de WCU necesarias para una inserción

Insertar 1 KB de datos por segundo requiere 1 WCU. Si su archivo CSV tiene 360 000 filas y quiere cargar todos los datos en 1 hora, debe escribir 100 filas por segundo (360 000 filas / 60 minutos / 60 segundos = 100 filas por segundo). Si cada fila tiene hasta 1 KB de datos, para insertar 100 filas por segundo, debe aprovisionar 100 WCU a su tabla. Si cada fila tiene 1,5 KB de datos, necesita 2 WCU para insertar una fila por segundo. Por tanto, para insertar 100 filas por segundo, debe aprovisionar 200 WCU.

Para determinar cuántas WCU necesita para insertar una fila por segundo, divida el tamaño medio de la fila en bytes por 1024 y redondee al número entero más próximo.

Por ejemplo, si el tamaño medio de las filas es de 3000 bytes, necesita 3 WCU para insertar una fila por segundo.

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

Cálculo de capacidad y tiempo de carga de datos

Ahora que conoce el tamaño medio y el número de filas de su archivo CSV, puede calcular cuántas WCU necesita para cargar los datos en un tiempo determinado y el tiempo aproximado que se tarda en cargar todos los datos de su archivo CSV utilizando diferentes ajustes de WCU.

Por ejemplo, si cada fila de su archivo es de 1 KB y tiene 1 000 000 de filas en su archivo CSV, para cargar los datos en 1 hora, necesita aprovisionar al menos 278 WCU a su tabla para esa hora.

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

Configuración de los ajustes de capacidad aprovisionada

Puede configurar los ajustes de capacidad de escritura de una tabla al crearla o mediante el comando CQL `ALTER TABLE`. A continuación se muestra la sintaxis para modificar los ajustes de capacidad aprovisionada de una tabla con la instrucción CQL `ALTER TABLE`.

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

Para ver la referencia completa del lenguaje, consulte the section called “ALTER TABLE”.

Paso 4: Configurar los ajustes de `cqlsh COPY FROM`

En esta sección se describe cómo determinar los valores de los parámetros para `cqlsh COPY FROM`. El comando `cqlsh COPY FROM` lee el archivo CSV que preparó anteriormente e inserta los datos en Amazon Keyspaces mediante CQL. El comando divide las filas y distribuye las operaciones `INSERT` entre un conjunto de trabajadores. Cada trabajador establece una conexión con Amazon Keyspaces y envía solicitudes `INSERT` por este canal.

El comando `cqlsh COPY` no dispone de lógica interna para distribuir el trabajo de manera uniforme entre sus trabajadores. Sin embargo, puede configurarlo manualmente para asegurarse de que el trabajo se distribuya uniformemente. Para comenzar, revise estos parámetros clave de cqlsh:

- DELIMITER: si ha utilizado un delimitador distinto de la coma, puede configurar este parámetro, que por defecto es la coma.
- INGESTRATE: el número objetivo de filas que `cqlsh COPY FROM` intenta procesar por segundo. Si no se establece, su valor predeterminado es 100 000.
- NUMPROCESSES: el número de procesos de trabajadores subordinados que cqlsh crea para las tareas `COPY FROM`. El valor máximo para este ajuste es 16, el valor predeterminado es `num_cores - 1`, donde `num_cores` es el número de núcleos de procesamiento en el host que ejecuta cqlsh.
- MAXBATCHSIZE: el tamaño del lote determina el número máximo de filas insertadas en la tabla de destino en un único lote. Si no se establece, cqlsh utiliza lotes de 20 filas insertadas.
- CHUNKSIZE: el tamaño de la unidad de trabajo que pasa al trabajador subordinado. De forma predeterminada, se establece en 5000.

- MAXATTEMPTS el número máximo de veces que se debe reintentar un bloque de trabajador fallido. Una vez alcanzado el máximo de intentos, los registros fallidos se escriben en un nuevo archivo CSV que puede volver a ejecutar más tarde tras investigar el fallo.

Establezca INGESTRATE en función del número de WCU que haya aprovisionado en la tabla de destino. El valor INGESTRATE del comando `cqlsh COPY FROM` no es un límite, es una media objetivo. Esto significa que puede (y a menudo lo hace) dispararse por encima del número que usted fije. Para tener en cuenta los picos y asegurarse de que exista capacidad suficiente para gestionar las solicitudes de carga de datos, establezca INGESTRATE en el 90 % de la capacidad de escritura de la tabla.

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

A continuación, fije el parámetro NUMPROCESSES en uno menos que el número de núcleos de su sistema. Para averiguar el número de núcleos de su sistema, puede ejecutar el siguiente código.

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

En este tutorial, utilizamos el siguiente valor.

```
NUMPROCESSES = 4
```

Cada proceso crea un trabajador, y cada trabajador establece una conexión con Amazon Keyspaces. Amazon Keyspaces puede admitir hasta 3000 solicitudes CQL por segundo en cada conexión. Esto significa que debe asegurarse de que cada trabajador procese menos de 3000 solicitudes por segundo.

Al igual que con INGESTRATE, los trabajadores suelen dispararse por encima del número que usted establezca y no están limitados por segundos de reloj. Por tanto, para tener en cuenta los picos, establezca sus parámetros de cqlsh para que cada trabajador procese 2500 solicitudes por segundo. Para calcular la cantidad de trabajo distribuida a un trabajador, utilice la siguiente pauta.

- Divida INGESTRATE por NUMPROCESSES.
- Si INGESTRATE / NUMPROCESSES > 2500, disminuya el valor INGESTRATE para que esta fórmula se cumpla.

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

Antes de configurar los ajustes para optimizar la carga de nuestros datos de ejemplo, revisemos los ajustes predeterminados de `cqlsh` y veamos cómo afecta su uso al proceso de carga de datos. Dado que `cqlsh COPY FROM` utiliza el valor `CHUNKSIZE` a fin de crear bloques de trabajo (instrucciones `INSERT`) para distribuir a los trabajadores, el trabajo no se distribuye automáticamente de manera uniforme. Algunos trabajadores podrían permanecer inactivos, en función del ajuste de `INGESTRATE`.

Para distribuir el trabajo de manera uniforme entre los trabajadores y mantener a cada trabajador en la tasa óptima de 2500 solicitudes por segundo, debe establecer `CHUNKSIZE`, `MAXBATCHSIZE` y `INGESTRATE` cambiando los parámetros de entrada. Para optimizar la utilización del tráfico de red durante la carga de datos, elija un valor para `MAXBATCHSIZE` que se acerque al valor máximo de 30. Al cambiar `CHUNKSIZE` a 100 y `MAXBATCHSIZE` a 25, las 10 000 filas se reparten uniformemente entre los cuatro trabajadores (10 000 / 2500 = 4).

El siguiente ejemplo de código lo ilustra.

```
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
```

En resumen, utilice las siguientes fórmulas al establecer los parámetros de `cqlsh COPY FROM`:

- INGESTRATE = write_capacity_units * .90
- NUMPROCESSES = num_cores -1 (predeterminado)
- INGESTRATE / NUMPROCESSES = 2500 (Esta debe ser una instrucción true)
- MAXBATCHSIZE = 30 (De forma predeterminada, 20. Amazon Keyspaces acepta lotes de hasta 30)
- CHUNKSIZE = (INGESTRATE / NUMPROCESSES) / MAXBATCHSIZE

Ahora que ha calculado NUMPROCESSES, INGESTRATE y CHUNKSIZE, está listo para cargar sus datos.

Paso 5: Ejecutar el comando **cqlsh COPY FROM**

Para ejecutar el comando cqlsh COPY FROM, complete los siguientes pasos.

1. Conéctese a Amazon Keyspaces con cqlsh.
2. Elija su espacio de claves con el siguiente código.

```
USE catalog;
```

3. Establezca la coherencia de escritura en LOCAL_QUORUM. Para garantizar la durabilidad de los datos, Amazon Keyspaces no permite otras configuraciones de coherencia de escritura. Consulte el código siguiente.

```
CONSISTENCY LOCAL_QUORUM;
```

4. Prepare su sintaxis de cqlsh COPY FROM utilizando el siguiente ejemplo de código.

```
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. Ejecute la instrucción preparada en el paso anterior. cqlsh le devuelve un eco con todos los ajustes que haya configurado.

   a. Asegúrese de que los ajustes coincidan con los que ha introducido. Consulte el siguiente ejemplo.

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

   b. Revise el número de filas transferidas y la tasa media actual, como se muestra en el siguiente ejemplo.

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

c. Cuando cqlsh haya terminado de cargar los datos, revise el resumen de las estadísticas de carga de datos (el número de archivos leídos, el tiempo de ejecución y las filas omitidas), como se muestra en el siguiente ejemplo.

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

En este último paso del tutorial, ha cargado los datos en Amazon Keyspaces.

> **Important**
>
> Ahora que ha transferido los datos, ajuste la configuración de modo de capacidad de su tabla de destino para que coincida con los patrones de tráfico habituales de su aplicación. Incurrirá en cargos según la tarifa horaria de su capacidad aprovisionada hasta que la modifique.

Solución de problemas

Una vez finalizada la carga de datos, compruebe si se han omitido filas. Para ello, navegue hasta el directorio de origen del archivo CSV de origen y busque un archivo con el siguiente nombre.

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

cqlsh escribe las filas de datos omitidas en un archivo con ese nombre. Si el archivo existe en su directorio de origen y contiene datos, estas filas no se cargaron en Amazon Keyspaces. Para reintentar estas filas, compruebe primero si se ha producido algún error durante la carga y ajuste los datos en consecuencia. Para reintentar estas filas, puede volver a ejecutar el proceso.

Errores comunes

Los motivos más comunes por los que no se cargan las filas son errores de capacidad y errores de análisis sintáctico.

Errores de solicitud no válida al cargar datos en Amazon Keyspaces

En el siguiente ejemplo, la tabla de origen contiene una columna de contador, lo que da lugar a llamadas por lotes registradas con el comando `COPY` de cqlsh. Amazon Keyspaces no admite llamadas por lotes registradas.

```
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
```

Para resolver este error, utilice DSBulk para migrar los datos. Para obtener más información, consulte the section called "Carga de datos con DSBulk".

Errores del analizador de sintaxis al cargar datos en Amazon Keyspaces

El siguiente ejemplo muestra una fila omitida debido a un `ParseError`.

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

Para resolver este error, debe asegurarse de que los datos que se vayan a importar coincidan con el esquema de la tabla en Amazon Keyspaces. Revise el archivo de importación en busca de errores de análisis sintáctico. Puede intentar utilizar una única fila de datos mediante una instrucción `INSERT` para aislar el error.

Errores de capacidad al cargar datos en 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 utiliza las excepciones `ReadTimeout` y `WriteTimeout` para indicar cuándo falla una solicitud de escritura debido a una capacidad de rendimiento insuficiente. Para ayudar a diagnosticar las excepciones de capacidad insuficiente, Amazon Keyspaces publica `WriteThrottleEvents` y elabora `ReadThrottledEvents` métricas en Amazon. CloudWatch Para obtener más información, consulte the section called "Monitorización con CloudWatch".

Errores de cqlsh al cargar datos en Amazon Keyspaces

Como ayuda para solucionar los errores de cqlsh, vuelva a ejecutar el comando que falla con la bandera `--debug`.

En caso de utilizar una versión incompatible de cqlsh, verá el siguiente error.

```
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
```

Confirme que la versión de cqlsh instalada sea correcta; para ello, ejecute el siguiente comando.

```
cqlsh --version
```

Debería ver algo similar a lo siguiente como salida.

```
cqlsh 5.0.1
```

Si utiliza Windows, sustituya todas las instancias de `cqlsh` con `cqlsh.bat`. Por ejemplo, para comprobar la versión de cqlsh en Windows, ejecute el siguiente comando.

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

La conexión a Amazon Keyspaces falla una vez que el cliente cqlsh recibe tres errores consecutivos de cualquier tipo procedentes del servidor. El cliente cqlsh falla con el siguiente mensaje.

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

Para resolver este error, debe asegurarse de que los datos que se vayan a importar coincidan con el esquema de la tabla en Amazon Keyspaces. Revise el archivo de importación en busca de errores de análisis sintáctico. Puede intentar utilizar una única fila de datos mediante una instrucción INSERT para aislar el error.

El cliente intentará restablecer la conexión de forma automática.

Tutorial: Carga de datos en Amazon Keyspaces utilizando DSBulk

Este step-by-step tutorial le guía a través de la migración de datos de Apache Cassandra a Amazon Keyspaces mediante el cargador DataStax masivo (DSBulk) disponible en. GitHub En este tutorial, debe completar los siguientes pasos:

Temas

- Requisitos previos
- Paso 1: Crear el archivo CSV de origen y la tabla de destino
- Paso 2: Preparar los datos

- Paso 3: Establecer la capacidad de rendimiento de la tabla
- Paso 4: Configurar los ajustes de DSBulk
- Paso 5: Ejecutar el comando load de DSBulk

Requisitos previos

Para poder comenzar el tutorial, antes debe completar las siguientes tareas.

1. Si aún no lo ha hecho, cree una AWS cuenta siguiendo los pasos que se indican en. the section called “Con AWS Identity and Access Management figuración”
2. Cree las credenciales; para ello, siga los pasos indicados en the section called “Credenciales de IAM para autenticación de AWS.”.
3. Cree un archivo de almacén de confianza de JKS.

   a. Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

   ```
   curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
   ```

   > **Note**
   >
   > También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces correctamente. El certificado de Starfield proporciona compatibilidad retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.

   b. Convierta el certificado digital de Starfield en un archivo trustStore.

   ```
   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
   ```

   En este paso, debe crear una contraseña para el almacén de claves y dar confianza a este certificado. El comando interactivo tiene el siguiente aspecto.

   ```
   Enter keystore password:
   ```

```
Re-enter new 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:
  MD5:  32:4A:4B:BB:C8:63:69:9B:BE:74:9A:C6:DD:1D:46:24
  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
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                                        ....
]
]
Trust this certificate? [no]:  y
```

4. Configure la conexión del intérprete de comandos de Cassandra Query Language (cqlsh) y confirme que puede conectarse a Amazon Keyspaces siguiendo los pasos indicados en the section called “Uso de cqlsh”.

5. Descargue e instale DSBulk.

   a. Para descargar DSBulk, puede usar el siguiente código.

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

   b. A continuación, descomprima el archivo tar y añada DSBulk a su PATH, como se muestra en el siguiente ejemplo.

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

   c. Cree un archivo `application.conf` para almacenar la configuración que DSBulk utilizará. Puede guardar el siguiente ejemplo como `./dsbulk_keyspaces.conf`. Sustituya `localhost` por el punto de contacto de su clúster local de Cassandra si no se encuentra en el nodo local, por ejemplo, el nombre DNS o la dirección IP. Tome nota del nombre del archivo y de la ruta, dado que deberá especificarlos más adelante en el comando `dsbulk load`.

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

   d. Para habilitar la compatibilidad con SiGv4, descargue el `jar` archivo sombreado GitHuby colóquelo en la `lib` carpeta DSBulk, como se muestra en el siguiente ejemplo.

   ```
   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
   ```

Paso 1: Crear el archivo CSV de origen y la tabla de destino

En este tutorial, utilizamos un archivo de valores separados por comas (CSV) llamado `keyspaces_sample_table.csv` como archivo de origen para la migración de datos. El archivo de ejemplo proporcionado contiene algunas filas de datos de una tabla llamada `book_awards`.

1. Cree el archivo de origen. Puede elegir una de las siguientes opciones:
   - Descargue el archivo CSV de ejemplo (`keyspaces_sample_table.csv`), contenido en el siguiente archivo comprimido: samplemigration.zip. Descomprima el archivo y tome nota de la ruta a `keyspaces_sample_table.csv`.
   - Para rellenar un archivo CSV de origen con sus propios datos almacenados en una base de datos de Apache Cassandra, utilice `dsbulk unload`, como se muestra en el siguiente ejemplo.

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

     Asegúrese de que el archivo CSV que cree satisfaga los siguientes requisitos:

     - La primera fila contiene los nombres de las columnas.
     - Los nombres de las columnas del archivo CSV de origen coinciden con los nombres de las columnas de la tabla de destino.
     - Los datos están delimitados con una coma.
     - Todos los valores de los datos son tipos de datos válidos de Amazon Keyspaces. Consulte the section called “Tipos de datos”.
2. Cree el espacio de claves y la tabla de destino en Amazon Keyspaces.
   a. Conéctese a Amazon Keyspaces con `cqlsh` y sustituya el punto de conexión del servicio, el nombre de usuario y la contraseña del siguiente ejemplo por sus propios valores.

      ```
      cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
      p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
      ```

   b. Cree un nuevo espacio de claves con el nombre `catalog` como se muestra en el siguiente ejemplo.

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

c. Una vez que el nuevo espacio de claves tenga el estado de disponible, utilice el siguiente código para crear la tabla de destino `book_awards`. Para obtener más información sobre la creación asíncrona de recursos y cómo comprobar si un recurso está disponible, consulte the section called “Creación de espacios de claves”.

```
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)
   );
```

Si Apache Cassandra es su origen de datos, una forma sencilla de crear la tabla de destino de Amazon Keyspaces con encabezados que coincidan es generar la instrucción `CREATE TABLE` a partir de la tabla de origen, como se muestra en la siguiente instrucción.

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

A continuación, cree la tabla de destino en Amazon Keyspaces con los nombres de las columnas y los tipos de datos que coincidan con la descripción de la tabla de origen de Cassandra.

Paso 2: Preparar los datos

La preparación de los datos de origen para una transferencia eficaz es un proceso de dos pasos. En primer lugar, usted aleatoriza los datos. En segundo lugar, usted analiza los datos para determinar los valores apropiados de los parámetros de `dsbulk` y los ajustes necesarios de la tabla.

Aleatorización de los datos

El comando `dsbulk` lee y escribe los datos en el mismo orden en que aparecen en el archivo CSV. Si utiliza el comando `dsbulk` para crear el archivo de origen, los datos se escriben en el orden en que aparecen ordenados por claves en el CSV. Internamente, Amazon Keyspaces divide los datos

utilizando claves de partición. Si bien Amazon Keyspaces tiene una lógica integrada para ayudar a equilibrar la carga de las solicitudes de la misma clave de partición, la carga de datos es más rápida y eficiente si aleatoriza el orden. Esto se debe a que puede aprovechar el equilibrio de carga integrado que se produce cuando Amazon Keyspaces escribe en diferentes particiones.

Para repartir las escrituras entre las particiones de manera uniforme, debe aleatorizar los datos en el archivo de origen. Puede escribir una aplicación para hacerlo o utilizar una herramienta de código abierto, como Shuf. Shuf está disponible de forma gratuita en distribuciones Linux, en macOS (instalando coreutils en homebrew) y en Windows (utilizando Windows Subsystem for Linux [WSL]). Se requiere un paso adicional para evitar que la fila del encabezado con los nombres de las columnas se baraje en este paso.

Para aleatorizar el archivo de origen conservando el encabezado, introduzca el siguiente código.

```
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 reescribe los datos en un nuevo archivo CSV llamado `keyspace.table.csv`. Ahora puede eliminar el archivo `keyspaces_sample_table.csv`, dado que ya no lo necesita.

Análisis de los datos

Determine el tamaño medio y máximo de las filas analizando los datos.

Debe hacerlo por las siguientes razones:

- El tamaño medio de fila ayuda a estimar la cantidad total de datos que se van a transferir.
- Necesita el tamaño medio de fila para aprovisionar la capacidad de escritura necesaria para la carga de datos.
- Puede asegurarse de que cada fila tenga un tamaño inferior a 1 MB, que es el tamaño máximo de fila en Amazon Keyspaces.

> **Note**
>
> Esta cuota se refiere al tamaño de fila, no al de la partición. A diferencia de las particiones de Apache Cassandra, las particiones de Amazon Keyspaces pueden tener un tamaño prácticamente ilimitado. Las claves de partición y las columnas de agrupación requieren

almacenamiento adicional para los metadatos, que debe añadir al tamaño bruto de las filas. Para obtener más información, consulte the section called “Cálculo del tamaño de las filas”.

El siguiente código utiliza AWK para analizar un archivo CSV e imprimir el tamaño medio y máximo de las filas.

```
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
```

La ejecución de este código da como resultado la siguiente salida.

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

Asegúrese de que el tamaño máximo de su fila no supere 1 MB. De ser así, debe dividir la fila o comprimir los datos para que el tamaño de la fila sea inferior a 1 MB. En el siguiente paso de este tutorial, usted utiliza el tamaño medio de fila para aprovisionar la capacidad de escritura de la tabla.

Paso 3: Establecer la capacidad de rendimiento de la tabla

En este tutorial se muestra cómo ajustar DSBulk para que cargue los datos dentro de un intervalo de tiempo establecido. Dado que sabe de antemano cuántas lecturas y escrituras realiza, utilice el modo de capacidad aprovisionada. Una vez finalizada la transferencia de datos, debe ajustar el modo de capacidad de la tabla para que se adapte a los patrones de tráfico de su aplicación. Para obtener más información sobre administración de capacidad, consulte Administración de recursos sin servidor.

Con el modo de capacidad aprovisionada, usted especifica con antelación cuánta capacidad de lectura y escritura desea aprovisionar a su tabla. La capacidad de escritura se factura por hora y se mide en unidades de capacidad de escritura (WCU). Cada WCU es capacidad de escritura suficiente para admitir la escritura de 1 KB de datos por segundo. Al cargar los datos, la tasa de escritura debe estar por debajo de las WCU máximas (parámetro: `write_capacity_units`) que se hayan establecido en la tabla de destino.

De forma predeterminada, puede aprovisionar hasta 40 000 WCU en una tabla y 80 000 WCU en todas las tablas de su cuenta. Si necesita capacidad adicional, puede solicitar un aumento de cuota

en la consola de Service Quotas. Para obtener más información acerca de las cuotas, consulte Cuotas.

Cálculo del número medio de WCU necesarias para una inserción

Insertar 1 KB de datos por segundo requiere 1 WCU. Si su archivo CSV tiene 360 000 filas y quiere cargar todos los datos en 1 hora, debe escribir 100 filas por segundo (360 000 filas / 60 minutos / 60 segundos = 100 filas por segundo). Si cada fila tiene hasta 1 KB de datos, para insertar 100 filas por segundo, debe aprovisionar 100 WCU a su tabla. Si cada fila tiene 1,5 KB de datos, necesita 2 WCU para insertar una fila por segundo. Por tanto, para insertar 100 filas por segundo, debe aprovisionar 200 WCU.

Para determinar cuántas WCU necesita para insertar una fila por segundo, divida el tamaño medio de la fila en bytes por 1024 y redondee al número entero más próximo.

Por ejemplo, si el tamaño medio de las filas es de 3000 bytes, necesita 3 WCU para insertar una fila por segundo.

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

Cálculo de capacidad y tiempo de carga de datos

Ahora que conoce el tamaño medio y el número de filas de su archivo CSV, puede calcular cuántas WCU necesita para cargar los datos en un tiempo determinado y el tiempo aproximado que se tarda en cargar todos los datos de su archivo CSV utilizando diferentes ajustes de WCU.

Por ejemplo, si cada fila de su archivo es de 1 KB y tiene 1 000 000 de filas en su archivo CSV, para cargar los datos en 1 hora, necesita aprovisionar al menos 278 WCU a su tabla para esa hora.

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

Configuración de los ajustes de capacidad aprovisionada

Puede configurar los ajustes de capacidad de escritura de una tabla al crearla o mediante el comando `ALTER TABLE`. A continuación se muestra la sintaxis para modificar los ajustes de capacidad aprovisionada de una tabla con el comando `ALTER TABLE`.

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

Para ver la referencia completa del lenguaje, consulte the section called “CREATE TABLE” y the section called “ALTER TABLE”.

Paso 4: Configurar los ajustes de `DSBulk`

En esta sección se describen los pasos necesarios para configurar DSBulk para la carga de datos en Amazon Keyspaces. Puede configurar DSBulk mediante un archivo de configuración. Puede especificar el archivo de configuración directamente desde la línea de comandos.

1. Cree un archivo de configuración DSBulk para la migración a Amazon Keyspaces; en este ejemplo utilizamos el nombre de archivo `dsbulk_keyspaces.conf`. Especifique los siguientes ajustes en el archivo de configuración de DSBulk.

   a. *`PlainTextAuthProvider`*: cree el proveedor de autenticación con la clase `PlainTextAuthProvider`. `ServiceUserName` y `ServicePassword` deben coincidir con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio siguiendo los pasos en the section called “Creación de credenciales”.

   b. *`local-datacenter`*— Defina el valor al valor `local-datacenter` al Región de AWS que se está conectando. Por ejemplo, si la aplicación se conecta a `cassandra.us-east-2.amazonaws.com`, entonces establezca el centro de datos local en `us-east-2`. Para ver todas las opciones disponibles Regiones de AWS, consultethe section called “Puntos de conexión de servicio”. Para evitar réplicas, establezca `slow-replica-avoidance` en `false`.

   c. *`SSLEngineFactory`*: para configurar SSL/TLS, inicialice la `SSLEngineFactory` añadiendo una sección en el archivo de configuración con una sola línea que especifique la clase con `class = DefaultSslEngineFactory`. Proporcione la ruta a `cassandra_truststore.jks` y la contraseña que creó anteriormente.

   d. *`consistency`*: establezca el nivel de coherencia en `LOCAL_QUORUM`. No se admiten otros niveles de coherencia de escritura. Para obtener más información consulte the section called “Niveles de coherencia de Cassandra admitidos”.

   e. El número de conexiones por grupo es configurable en el controlador Java. Para este ejemplo, establezca `advanced.connection.pool.local.size` en 3.

   A continuación se muestra el archivo de configuración de ejemplo completo.

```
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
}

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. Revise los parámetros del comando `load` de DSBulk.

   a. *`executor.maxPerSecond`*: el número máximo de filas que el comando de carga intenta procesar concurrentemente por segundo. Si no se establece, este parámetro se deshabilita con -1.

      Establezca `executor.maxPerSecond` en función del número de WCU que haya aprovisionado en la tabla de destino. El valor `executor.maxPerSecond` del comando `load` no es un límite, es una media objetivo. Esto significa que puede (y a menudo lo hace) dispararse por encima del número que usted fije. Para tener en cuenta los picos y asegurarse de que exista capacidad suficiente para gestionar las solicitudes de carga de datos, establezca `executor.maxPerSecond` en el 90 % de la capacidad de escritura de la tabla.

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

En este tutorial, establecemos `executor.maxPerSecond` en 5.

> **Note**
>
> Si utiliza DSBulk 1.6.0 o superior, puede usar `dsbulk.engine.maxConcurrentQueries` en su lugar.

b. Configure estos parámetros adicionales para el comando `load` de DSBulk.

- *`batch-mode`*: este parámetro indica al sistema que agrupe las operaciones por clave de partición. Dado que esto podría interferir con otros ajustes, recomendamos deshabilitar el modo por lotes.
- *`driver.advanced.retry-policy-max-retries`*: determina cuántas veces se debe reintentar una consulta fallida. Si no se establece, el valor predeterminado es 10. Puede ajustar este valor según sea necesario.
- *`driver.basic.request.timeout`*: el tiempo en minutos que el sistema espera el regreso de una consulta. Si no se establece, el valor predeterminado es “5 minutos”. Puede ajustar este valor según sea necesario.

Paso 5: Ejecutar el comando **`load`** de DSBulk

En el último paso de este tutorial, usted carga los datos en Amazon Keyspaces.

Para ejecutar el comando `load` de DSBulk, complete los siguientes pasos.

1. Ejecute el siguiente código para cargar los datos de su archivo csv en su tabla de Amazon Keyspaces. Asegúrese de actualizar la ruta al archivo de configuración de la aplicación que creó anteriormente.

```
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. La salida incluye la ubicación de un archivo de registro que detalla las operaciones exitosas y fallidas. El archivo se almacena en el siguiente directorio.

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

3. Las entradas del archivo de registro incluirán métricas, como se muestra en el siguiente ejemplo. Compruebe que el número de filas sea coherente con el número de filas de su archivo csv.

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

> **Important**
>
> Ahora que ha transferido los datos, ajuste la configuración de modo de capacidad de su tabla de destino para que coincida con los patrones de tráfico habituales de su aplicación. Incurrirá en cargos según la tarifa horaria de su capacidad aprovisionada hasta que la modifique. Para obtener más información, consulte the section called “Modos de capacidad de lectura/escritura”.

Ejemplos de código para Amazon Keyspaces que utilizan SDK AWS

Los siguientes ejemplos de código muestran cómo utilizar Amazon Keyspaces con un kit de desarrollo de AWS software (SDK).

Las acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Mientras las acciones muestran cómo llamar a las funciones de servicio individuales, es posible ver las acciones en contexto en los escenarios relacionados y en los ejemplos entre servicios.

Los escenarios son ejemplos de código que muestran cómo llevar a cabo una tarea específica llamando a varias funciones dentro del mismo servicio.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUso de Amazon Keyspaces con un SDK AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Introducción

Hello Amazon Keyspaces

En los siguientes ejemplos de código se muestra cómo empezar a utilizar Amazon Keyspaces.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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();
    }
}
```

- Para obtener más información sobre la API, consulta ListKeyspacesla Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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);
        }
    }
}
```

- Para obtener más información sobre la API, consulta ListKeyspacesla Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/**
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 { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.listKeyspaces(keyspacesRequest)
        response.keyspaces?.forEach { keyspace ->
```

```
            println("The name of the keyspace is ${keyspace.keyspaceName}")
        }
    }
}
```

- Para obtener más información sobre la API, consulta ListKeyspacesla referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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"))
```

- Para obtener más información sobre la API, consulta ListKeyspacesla AWS Referencia de API de SDK for Python (Boto3).

Ejemplos de código

- Acciones para Amazon Keyspaces mediante SDK AWS
  - Creación de un espacio de claves de Amazon Keyspaces mediante un SDK AWS
  - Creación de una tabla de Amazon Keyspaces mediante un SDK AWS
  - Eliminar un espacio de claves de Amazon Keyspaces mediante un SDK AWS
  - Eliminar una tabla de Amazon Keyspaces mediante un SDK AWS
  - Obtenga datos sobre un espacio de claves de Amazon Keyspaces mediante un SDK AWS
  - Obtenga datos sobre una tabla de Amazon Keyspaces mediante un SDK AWS
  - Listar los espacios clave de Amazon Keyspaces mediante un SDK AWS
  - Listar las tablas de Amazon Keyspaces en un espacio de claves mediante un SDK AWS
  - Restaurar una tabla de Amazon Keyspaces a un punto en el tiempo mediante un SDK AWS
  - Actualizar una tabla de Amazon Keyspaces mediante un SDK AWS
- Escenarios para Amazon Keyspaces que utilizan SDK AWS
  - Comience a utilizar Amazon Keyspaces, espacios clave y tablas mediante un SDK AWS

Acciones para Amazon Keyspaces mediante SDK AWS

Los siguientes ejemplos de código muestran cómo realizar acciones individuales de Amazon Keyspaces con AWS los SDK. Estos fragmentos llaman a la API de Amazon Keyspaces y son fragmentos de código de programas más grandes que deben ejecutarse en contexto. Cada ejemplo incluye un enlace a GitHub, donde puede encontrar instrucciones para configurar y ejecutar el código.

Los ejemplos siguientes incluyen solo las acciones que se utilizan con mayor frecuencia. Para obtener una lista completa, consulte la Referencia de la API de Amazon Keyspaces (para Apache Cassandra).

Ejemplos

- Creación de un espacio de claves de Amazon Keyspaces mediante un SDK AWS
- Creación de una tabla de Amazon Keyspaces mediante un SDK AWS
- Eliminar un espacio de claves de Amazon Keyspaces mediante un SDK AWS
- Eliminar una tabla de Amazon Keyspaces mediante un SDK AWS
- Obtenga datos sobre un espacio de claves de Amazon Keyspaces mediante un SDK AWS
- Obtenga datos sobre una tabla de Amazon Keyspaces mediante un SDK AWS
- Listar los espacios clave de Amazon Keyspaces mediante un SDK AWS
- Listar las tablas de Amazon Keyspaces en un espacio de claves mediante un SDK AWS
- Restaurar una tabla de Amazon Keyspaces a un punto en el tiempo mediante un SDK AWS
- Actualizar una tabla de Amazon Keyspaces mediante un SDK AWS

Creación de un espacio de claves de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo crear un espacio de claves de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <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;
    }
```

- Para obtener más información sobre la API, consulta CreateKeyspacela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
       }
```

```
    }
```

- Para obtener más información sobre la API, consulta CreateKeyspacela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest = CreateKeyspaceRequest {
        keyspaceName = keyspaceNameVal
    }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Para obtener más información sobre la API, consulta CreateKeyspacela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta CreateKeyspacela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de una tabla de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo crear una tabla de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /// <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;
    }
```

- Para obtener más información sobre la API, consulta CreateTablela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    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);
        }
    }
```

- Para obtener más información sobre la API, consulta CreateTablela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createTable(tableRequest)
        println("The table ARN is ${response.resourceArn}")
    }
}
```

- Para obtener más información sobre la API, consulta CreateTablela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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"]
```

- Para obtener más información sobre la API, consulta CreateTablela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar un espacio de claves de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo eliminar un espacio de claves de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <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;
}
```

- Para obtener más información sobre la API, consulta DeleteKeyspacela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
       }
   }
```

- Para obtener más información sobre la API, consulta DeleteKeyspacela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun deleteKeyspace(keyspaceNameVal: String?) {
    val deleteKeyspaceRequest = DeleteKeyspaceRequest {
        keyspaceName = keyspaceNameVal
    }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}
```

- Para obtener más información sobre la API, consulta DeleteKeyspacela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta DeleteKeyspacela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar una tabla de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo eliminar una tabla de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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;
    }
```

- Para obtener más información sobre la API, consulta DeleteTablela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    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);
        }
    }
```

- Para obtener más información sobre la API, consulta DeleteTablela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun deleteTable(keyspaceNameVal: String?, tableNameVal: String?) {
    val tableRequest = DeleteTableRequest {
        keyspaceName = keyspaceNameVal
        tableName = tableNameVal
    }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}
```

- Para obtener más información sobre la API, consulta DeleteTablela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta DeleteTablela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtenga datos sobre un espacio de claves de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo obtener datos sobre un espacio de claves de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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;
    }
```

- Para obtener más información sobre la API, consulta GetKeyspacela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
        }
    }
```

- Para obtener más información sobre la API, consulta GetKeyspacela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyspaceRequest = GetKeyspaceRequest {
        keyspaceName = keyspaceNameVal
    }
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        val response: GetKeyspaceResponse =
 keyClient.getKeyspace(keyspaceRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}
```

- Para obtener más información sobre la API, consulta GetKeyspacela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta GetKeyspacela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtenga datos sobre una tabla de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo obtener datos sobre una tabla de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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;
    }
```

- Para obtener más información sobre la API, consulta GetTablela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
       }
   }
```

- Para obtener más información sobre la API, consulta GetTablela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 { 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}")
            }
        }
    }
}
```

- Para obtener más información sobre la API, consulta GetTablela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta GetTablela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Listar los espacios clave de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo enumerar espacios de claves de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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}");
        }
    }
```

- Para obtener más información sobre la API, consulta ListKeyspacesla Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
       }
   }
```

- Para obtener más información sobre la API, consulta ListKeyspacesla Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun listKeyspacesPaginator() {
    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.listKeyspacesPaginated(ListKeyspacesRequest {})
            .transform { it.keyspaces?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println("Name: ${obj.keyspaceName}")
            }
    }
}
```

- Para obtener más información sobre la API, consulta ListKeyspaceslа referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta ListKeyspacesla AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Listar las tablas de Amazon Keyspaces en un espacio de claves mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo enumerar tablas en un espacio de claves de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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;
    }
```

- Para obtener más información sobre la API, consulta ListTablesla Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
       }
   }
```

- Para obtener más información sobre la API, consulta ListTablesla Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest = ListTablesRequest {
        keyspaceName = keyspaceNameVal
    }

    KeyspacesClient { region = "us-east-1" }.use { keyClient ->
        keyClient.listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println(
                    " ARN: " + obj.resourceArn.toString() +
                        " Table name: " + obj.tableName
                )
            }
    }
}
```

- Para obtener más información sobre la API, consulta ListTablesla referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta ListTablesla AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Restaurar una tabla de Amazon Keyspaces a un punto en el tiempo mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo restaurar una tabla de Amazon Keyspaces a un momento dado.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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;
    }
```

- Para obtener más información sobre la API, consulta RestoreTablela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
        }
    }
```

- Para obtener más información sobre la API, consulta RestoreTablela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.restoreTable(restoreTableRequest)
        println("The ARN of the restored table is ${response.restoredTableArn}")
    }
}
```

- Para obtener más información sobre la API, consulta RestoreTablela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta RestoreTablela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Actualizar una tabla de Amazon Keyspaces mediante un SDK AWS

En los siguientes ejemplos de código se muestra cómo actualizar una tabla de Amazon Keyspaces.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Introducción a los espacios de claves y las tablas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <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;
    }
```

- Para obtener más información sobre la API, consulta UpdateTablela Referencia AWS SDK for .NET de la API.

Java

SDK para Java 2.x

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   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);
       }
   }
```

- Para obtener más información sobre la API, consulta UpdateTablela Referencia AWS SDK for Java 2.x de la API.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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 { region = "us-east-1" }.use { keyClient ->
        keyClient.updateTable(tableRequest)
    }
}
```

- Para obtener más información sobre la API, consulta UpdateTablela referencia sobre el AWS SDK para la API de Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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
```

- Para obtener más información sobre la API, consulta UpdateTablela AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Uso de Amazon Keyspaces con un SDK AWS En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Escenarios para Amazon Keyspaces que utilizan SDK AWS

Los siguientes ejemplos de código muestran cómo implementar escenarios comunes en Amazon Keyspaces con AWS SDK. Estos escenarios le muestran cómo efectuar tareas específicas llamando a múltiples funciones dentro de Amazon Keyspaces. Cada escenario incluye un enlace a GitHub, donde puede encontrar instrucciones sobre cómo configurar y ejecutar el código.

Ejemplos

- Comience a utilizar Amazon Keyspaces, espacios clave y tablas mediante un SDK AWS

Comience a utilizar Amazon Keyspaces, espacios clave y tablas mediante un SDK AWS

En el siguiente ejemplo de código, se muestra cómo:

- Crear un espacio de claves y una tabla. El esquema de la tabla contiene los datos de las películas y tiene habilitada point-in-time la recuperación.
- Conectarse al espacio de claves mediante una conexión TLS segura con autenticación SigV4.
- Consultar la tabla. Agregue, recupere y actualice datos de películas.
- Actualizar la tabla. Agregue una columna para llevar un seguimiento de las películas vistas.
- Restaurar la tabla a su estado anterior y limpiar los recursos.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
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!;
        }
    }
```

```
}
```

- Para obtener detalles de la API, consulte los siguientes temas en la Referencia de la API de AWS SDK for .NET .
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Java

SDK para Java 2.x

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/**
 * 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);
        }
    }
}
```

- Para obtener detalles de la API, consulte los siguientes temas en la Referencia de la API de AWS SDK for Java 2.x .
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/**
 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 { 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 { 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 { 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 { 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 { 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 { 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 { region = "us-east-1" }.use { keyClient ->
        keyClient.listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println(
                    " ARN: " + obj.resourceArn.toString() +
                        " 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 { 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 { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createTable(tableRequest)
        println("The table ARN is ${response.resourceArn}")
    }
}

suspend fun listKeyspacesPaginator() {
    KeyspacesClient { 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 { 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 { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Para obtener información acerca de la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Kotlin.
  - CreateKeyspace
  - CreateTable
  - DeleteKeyspace
  - DeleteTable
  - GetKeyspace
  - GetTable
  - ListKeyspaces
  - ListTables
  - RestoreTable
  - UpdateTable

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecutar un escenario interactivo en un símbolo del sistema.

```
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.")
```

Defina una clase que contenga las acciones del espacio de claves y la tabla.

```
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
```

Defina una clase que cree una conexión TLS a un espacio de claves, se autentique con SigV4 y envíe las consultas CQL a una tabla del espacio de claves.

```
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],
        )
```

- Para obtener detalles de la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).

- CreateKeyspace
- CreateTable
- DeleteKeyspace
- DeleteTable
- GetKeyspace
- GetTable
- ListKeyspaces
- ListTables
- RestoreTable
- UpdateTable

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUso de Amazon Keyspaces con un SDK AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Bibliotecas y herramientas para Amazon Keyspaces (para Apache Cassandra)

En esta sección se ofrece información sobre las bibliotecas, ejemplos de código y herramientas de Amazon Keyspaces (para Apache Cassandra).

Temas

- Bibliotecas y ejemplos
- Repositorios destacados de ejemplos y herramientas para desarrolladores

Bibliotecas y ejemplos

Puede encontrar bibliotecas de código abierto y herramientas para desarrolladores de Amazon Keyspaces en GitHub, en los repositorios de AWS y de ejemplos de AWS.

Kit de herramientas para desarrolladores de Amazon Keyspaces (para Apache Cassandra)

Este repositorio proporciona una imagen de Docker con herramientas útiles para desarrolladores de Amazon Keyspaces. Por ejemplo, incluye un archivo CQLSHRC con prácticas recomendadas, una expansión opcional de autenticación de AWS para cqlsh y herramientas de ayuda para realizar tareas comunes. El conjunto de herramientas está optimizado para Amazon Keyspaces, pero también funciona con clústeres de Apache Cassandra.

https://github.com/aws-samples/amazon-keyspaces-toolkit.

Ejemplos de Amazon Keyspaces (para Apache Cassandra)

Este repositorio es nuestra lista oficial de ejemplos de código de Amazon Keyspaces. El repositorio está subdividido en secciones por lenguaje (consulte Ejemplos). Cada lenguaje tiene su propia subsección de ejemplos. En estos ejemplos se ilustran implementaciones de servicios y patrones comunes de Amazon Keyspaces que puede utilizar al crear aplicaciones.

https://github.com/aws-samples/amazon-keyspaces-examples/.

Complementos de autenticación AWS Signature Version 4 (SigV4)

Estos complementos le permiten administrar el acceso a Amazon Keyspaces mediante el uso de usuarios y roles de AWS Identity and Access Management (IAM).

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.

Go: https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Repositorios destacados de ejemplos y herramientas para desarrolladores

A continuación se incluye una selección de herramientas útiles de la comunidad para Amazon Keyspaces (para Apache Cassandra).

Búferes de protocolo de Amazon Keyspaces

Puede utilizar búferes de protocolo (Protobuf) con Amazon Keyspaces para proporcionar una alternativa a los tipos definidos por el usuario (UDT) de Apache Cassandra. Protobuf es un formato de datos multiplataforma gratuito y de código abierto que se utiliza para serializar datos estructurados. Puede almacenar datos Protobuf utilizando el tipo de datos `BL0B` de CQL y refactorizar los UDT conservando los datos estructurados a través de aplicaciones y lenguajes de programación.

Este repositorio proporciona un ejemplo de código que se conecta a Amazon Keyspaces, crea una nueva tabla e inserta una fila que contiene un mensaje Protobuf. A continuación, la fila se lee con una fuerte coherencia.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/protobuf-user-defined-types

Plantilla de AWS CloudFormation para crear el panel de control de métricas de Amazon CloudWatch para Amazon Keyspaces (para Apache Cassandra)

En este repositorio se proporcionan plantillas de AWS CloudFormation para configurar de forma rápida las métricas de CloudWatch para Amazon Keyspaces. El uso de esta plantilla le permite dar los primeros pasos de manera más sencilla al proporcionarle paneles de control de CloudWatch preconfigurados y desplegables con métricas de uso común.

https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates.

Uso de Amazon Keyspaces (para Apache Cassandra) con AWS Lambda

El repositorio contiene ejemplos que muestran cómo conectarse a Amazon Keyspaces desde Lambda. A continuación se muestran algunos ejemplos.

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.

Otro ejemplo de Lambda que muestra cómo implementar y utilizar Amazon Keyspaces desde una Lambda de Python está disponible en el siguiente repositorio.

https://github.com/aws-samples/aws-keyspaces-lambda-python

Uso de Amazon Keyspaces (para Apache Cassandra) con Spring

Este es un ejemplo que muestra cómo utilizar Amazon Keyspaces con Spring Boot.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring

Uso de Amazon Keyspaces (para Apache Cassandra) con Scala

Este es un ejemplo que muestra cómo conectarse a Amazon Keyspaces mediante el complemento de autenticación SigV4 con Scala.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/connection-sigv4

Uso de Amazon Keyspaces (para Apache Cassandra) con AWS Glue

Este es un ejemplo que muestra cómo utilizar Amazon Keyspaces con AWS Glue.

https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue

Conversor de lenguaje de consulta de Cassandra (CQL) a AWS CloudFormation de Amazon Keyspaces (para Apache Cassandra)

Este paquete implementa una herramienta de línea de comandos para convertir scripts de Cassandra Query Language (CQL) de Apache en plantillas de AWS CloudFormation (CloudFormation), lo que permite administrar con facilidad los esquemas de Amazon Keyspaces en pilas de CloudFormation.

https://github.com/aws/amazon-keyspaces-cql-to-cfn-converter.

Asistentes de Amazon Keyspaces (para Apache Cassandra) para el controlador de Apache Cassandra para Java

Este repositorio contiene políticas para el controlador, ejemplos y prácticas recomendadas al utilizar el controlador DataStax Java con Amazon Keyspaces (para Apache Cassandra).

https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers.

Demostración de compresión rápida de Amazon Keyspaces (para Apache Cassandra)

Este repositorio muestra cómo comprimir, almacenar y leer/escribir objetos de gran tamaño para obtener un desempeño más veloz y reducir el rendimiento y los costos de almacenamiento.

https://github.com/aws-samples/amazon-keyspaces-compression-example.

Demostración del códec de Amazon S3 y Amazon Keyspaces (para Apache Cassandra)

El códec personalizado de Amazon S3 admite asignación transparente y configurable por el usuario de punteros UUID a objetos de Amazon S3.

https://github.com/aws-samples/amazon-keyspaces-large-object-s3-demo.

Integración de Amazon Keyspaces con Apache Spark

Apache Spark es un motor de código abierto para el análisis de datos a gran escala. Apache Spark le permite realizar análisis de datos almacenados en Amazon Keyspaces de forma más eficiente. También puede utilizar Amazon Keyspaces para proporcionar a las aplicaciones acceso de lectura coherente en un dígito de milisegundo a los datos analíticos de Spark. El Spark Cassandra Connector de código abierto simplifica la lectura y escritura de datos entre Amazon Keyspaces y Spark.

La compatibilidad de Amazon Keyspaces con Spark Cassandra Connector agiliza la ejecución de cargas de trabajo de Cassandra en canalizaciones de análisis basadas en Spark mediante el uso de un servicio de base de datos totalmente administrado sin servidor. Con Amazon Keyspaces, no tiene que preocuparse de que Spark compita por los mismos recursos de infraestructura subyacentes que sus tablas. Las tablas de Amazon Keyspaces se amplían y reducen automáticamente en función del tráfico de su aplicación.

En el siguiente tutorial se describen los pasos y las prácticas recomendadas necesarios para leer y escribir datos en Amazon Keyspaces utilizando Spark Cassandra Connector. En el tutorial se explica cómo migrar datos a Amazon Keyspaces cargando datos desde un archivo con Spark Cassandra Connector y escribiéndolos en una tabla de Amazon Keyspaces. Además, en el tutorial se muestra cómo volver a leer los datos desde Amazon Keyspaces utilizando Spark Cassandra Connector. Esto se haría para ejecutar cargas de trabajo de Cassandra en canalizaciones de análisis basadas en Spark.

Temas

- Requisitos previos para establecer conexiones a Amazon Keyspaces con Spark Cassandra Connector
- Paso 1: Configurar Amazon Keyspaces para la integración con Apache Cassandra Spark Connector
- Paso 2: Configurar Apache Cassandra Spark Connector
- Paso 3: Crear el archivo de configuración de la aplicación
- Paso 4: Preparar los datos de origen y la tabla de destino en Amazon Keyspaces
- Paso 5: Escribir y leer datos en Amazon Keyspaces utilizando Apache Cassandra Spark Connector
- Solución de problemas comunes al utilizar Spark Cassandra Connector con Amazon Keyspaces

Requisitos previos para establecer conexiones a Amazon Keyspaces con Spark Cassandra Connector

Antes de conectarse a Amazon Keyspaces con Spark Cassandra Connector, debe asegurarse de haber instalado lo siguiente. La compatibilidad de Amazon Keyspaces con Spark Cassandra Connector se ha probado con las siguientes versiones recomendadas:

- Java versión 8
- Scala 2.12
- Spark 3.4
- Cassandra Connector 2.5 y superior
- Controlador Cassandra 4.12

1. Para instalar Scala, siga las instrucciones indicadas en https://www.scala-lang.org/download/scala2.html.
2. Para instalar Spark 3.4.1, siga este ejemplo.

```
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
```

Paso 1: Configurar Amazon Keyspaces para la integración con Apache Cassandra Spark Connector

En este paso, usted confirma que el particionador de su cuenta es compatible con Apache Spark Connector y configura los permisos de IAM necesarios. Las siguientes prácticas recomendadas le ayudarán a aprovisionar suficiente capacidad de lectura/escritura para la tabla.

1. Confirme que el particionador `Murmur3Partitioner` es el particionador predeterminado de su cuenta. Este particionador es compatible con Spark Cassandra Connector. Para obtener más información sobre los particionadores y cómo cambiarlos, consulte the section called “Uso de particionadores”.
2. Configure sus permisos de IAM para Amazon Keyspaces, utilizando puntos de conexión de VPC de interfaz, con Apache Spark.

   - Asigne acceso de lectura/escritura a la tabla de usuario y acceso de lectura a las tablas del sistema, como se muestra en el ejemplo de política de IAM que figura a continuación.
   - Es necesario rellenar la tabla system.peers con sus puntos de conexión de VPC de interfaz disponibles para los clientes que accedan a Amazon Keyspaces con Spark a través de puntos de conexión de VPC.

```
{
   "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. Tenga en cuenta las siguientes prácticas recomendadas a fin de configurar una capacidad de rendimiento de lectura/escritura suficiente para que su tabla de Amazon Keyspaces admita el tráfico de Spark Cassandra Connector.

   - Para comenzar, utilice capacidad bajo demanda como ayuda para probar el escenario.
   - Para optimizar el costo de rendimiento de la tabla para entornos de producción, utilice un limitador de velocidad para el tráfico procedente del conector y configure su tabla para utilizar capacidad aprovisionada con escalado automático. Para obtener más información, consulte the section called “Administre la capacidad de rendimiento con el escalado automático”.
   - Puede utilizar un limitador de tasa fijo que viene con el controlador Cassandra. Existen algunos limitadores de tasa adaptados para Amazon Keyspaces en el repositorio Ejemplos de AWS.
   - Para obtener más información sobre la administración de capacidad, consulte the section called “Modos de capacidad de lectura/escritura”.

Paso 2: Configurar Apache Cassandra Spark Connector

Apache Spark es una plataforma de cómputo de uso general que puede configurar de diferentes maneras. Para configurar Spark y Spark Cassandra Connector para su integración con Amazon Keyspaces, le recomendamos que comience con los ajustes de configuración mínimos que se describen en la siguiente sección y que los aumente posteriormente según resulte apropiado para su carga de trabajo.

- Cree particiones de Spark de tamaño inferior a 8 MB.

  En Spark, las particiones representan un bloque atómico de datos que puede ejecutarse en paralelo. Al escribir datos en Amazon Keyspaces con Spark Cassandra Connector, cuanto menor sea la partición de Spark, menor será la cantidad de registros que la tarea vaya a escribir. Si una tarea de Spark encuentra múltiples errores, falla una vez agotado el número designado de reintentos. Para evitar la repetición de tareas grandes y el reprocesamiento de muchos datos, mantenga pequeño el tamaño de la partición de Spark.
- Utilice un número bajo de escrituras concurrentes por ejecutor con un gran número de reintentos.

  Amazon Keyspaces devuelve errores de capacidad insuficiente a los controladores de Cassandra como tiempos de espera de operación. No puede abordar los tiempos de espera debidos a capacidad insuficiente cambiando la duración del tiempo de espera configurado porque Spark Cassandra Connector prueba el reintento de solicitudes de forma transparente utilizando

la `MultipleRetryPolicy`. Para asegurarse de que los reintentos no saturan el grupo de conexiones del controlador, utilice un número bajo de escrituras concurrentes por ejecutor con un gran número de reintentos. El siguiente fragmento de código es un ejemplo de ello.

```
spark.cassandra.query.retry.count = 500
spark.cassandra.output.concurrent.writes = 3
```

- Divida el rendimiento total y distribúyalo entre múltiples sesiones de Cassandra.
  - Spark Cassandra Connector crea una única sesión para cada ejecutor Spark. Piense en esta sesión como la unidad de escala para determinar el rendimiento requerido y el número de conexiones necesarias.
  - Al definir el número de núcleos por ejecutor y el número de núcleos por tarea, empiece por valores bajos y aumente según sea necesario.
  - Defina los fallos de las tareas de Spark para permitir el procesamiento en caso de errores transitorios. Una vez que se haya familiarizado con las características y los requisitos de tráfico de su aplicación, le recomendamos establecer `spark.task.maxFailures` en un valor acotado.
  - Por ejemplo, la siguiente configuración puede gestionar dos tareas concurrentes por ejecutor por sesión:

```
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
```

- Desactive el procesamiento por lotes.
  - Le recomendamos que desactive el procesamiento por lotes para mejorar los patrones de acceso aleatorio. El siguiente fragmento de código es un ejemplo de ello.

```
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)
```

- Establezca **`SPARK_LOCAL_DIRS`** en un disco local rápido con espacio suficiente.
  - De forma predeterminada, Spark guarda los archivos de salida de asignaciones y los conjuntos de datos distribuidos resistentes (RDD) en una carpeta `/tmp` . En función de la configuración de su host Spark, esto puede dar lugar a errores del estilo no hay espacio disponible en el dispositivo.

- Para establecer la variable de entorno `SPARK_LOCAL_DIRS` en un directorio llamado `/example/spark-dir`, puede utilizar el siguiente comando.

```
export SPARK_LOCAL_DIRS=/example/spark-dir
```

Paso 3: Crear el archivo de configuración de la aplicación

Para utilizar Spark Cassandra Connector de código abierto con Amazon Keyspaces, debe proporcionar un archivo de configuración de la aplicación que contenga los ajustes necesarios para conectarse con el controlador Java de DataStax. Para conectarse, puede utilizar credenciales específicas del servicio o el complemento SigV4.

Si aún no lo ha hecho, necesita convertir el certificado digital de Starfield en un archivo trustStore. Puede seguir los pasos detallados en the section called “Antes de empezar” del tutorial de conexión del controlador Java. Tome nota de la ruta y la contraseña del archivo trustStore porque necesitará esta información en el momento de crear el archivo de configuración de la aplicación.

Conexión con autenticación SigV4

En esta sección se muestra un archivo `application.conf` de ejemplo que puede utilizar cuando se conecte con credenciales de AWS y el complemento SigV4. Si aún no lo ha hecho, debe generar sus claves de acceso de IAM (un ID de clave de acceso y una clave de acceso secreta) y guardarlas en su archivo de configuración de AWS o como variables de entorno. Para obtener instrucciones detalladas, consulte the section called “Credenciales requeridas para la autenticación de AWS”.

En el siguiente ejemplo, sustituya la ruta de acceso a su archivo trustStore y la contraseña.

```
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
}
```

Actualice y guarde este archivo de configuración como `/home/user1/application.conf`. Los siguientes ejemplos utilizan esta ruta.

Conexión con credenciales específicas del servicio

En esta sección se muestra un archivo `application.conf` de ejemplo que puede utilizar cuando se conecte con credenciales específicas del servicio. Si aún no lo ha hecho, debe generar credenciales específicas del servicio para Amazon Keyspaces. Para obtener instrucciones detalladas, consulte the section called “Credenciales específicas del servicio”.

En el siguiente ejemplo, sustituya `username` y `password` por sus propias credenciales. Asimismo, sustituya la ruta de acceso a su archivo trustStore y la contraseña.

```
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
                }
            }
        }
}
```

Actualice y guarde este archivo de configuración como `/home/user1/application.conf` para utilizarlo con el código de ejemplo.

Conexión con una tasa fija

Para forzar una tasa fija por ejecutor Spark, puede definir un regulador de solicitudes. Este regulador de solicitudes limita la tasa de solicitudes por segundo. Spark Cassandra Connector implementa una sesión de Cassandra por ejecutor. El uso de la siguiente fórmula puede ayudarle a conseguir un rendimiento coherente en una tabla.

```
max-request-per-second * numberOfExecutors = total throughput against a table
```

Puede añadir este ejemplo al archivo de configuración de la aplicación que creó anteriormente.

```
datastax-java-driver {
  advanced.throttler {
    class = RateLimitingRequestThrottler

    max-requests-per-second = 3000
    max-queue-size = 30000
    drain-interval = 1 millisecond
  }
}
```

Paso 4: Preparar los datos de origen y la tabla de destino en Amazon Keyspaces

En este paso, usted crea un archivo de origen con datos de muestra y una tabla de Amazon Keyspaces.

1. Cree el archivo de origen. Puede elegir una de las siguientes opciones:
   - En este tutorial, usted utiliza un archivo de valores separados por comas (CSV) llamado `keyspaces_sample_table.csv` como archivo de origen para la migración de datos. El archivo de ejemplo proporcionado contiene algunas filas de datos de una tabla llamada `book_awards`.
     - Descargue el archivo CSV de ejemplo (`keyspaces_sample_table.csv`) que se encuentra en el siguiente archivo comprimido: samplemigration.zip. Descomprima el archivo y tome nota de la ruta a `keyspaces_sample_table.csv`.
   - Si desea seguir con su propio archivo CSV para escribir datos en Amazon Keyspaces, asegúrese de que los datos estén asignados al azar. Los datos que se leen directamente de una base de datos o se exportan a archivos planos suelen estar ordenados por la partición y la clave principal. Importar datos ordenados a Amazon Keyspaces puede hacer que se escriban en segmentos más pequeños de las particiones de Amazon Keyspaces, lo que da lugar a una distribución desigual del tráfico. Esto puede dar lugar a un rendimiento más lento y a tasas de error más elevadas.

     Por el contrario, la asignación al azar de los datos ayuda a aprovechar las capacidades de equilibrio de carga integradas de Amazon Keyspaces al distribuir el tráfico entre las particiones de forma más uniforme. Existen diversas herramientas que puede utilizar para aleatorizar los datos. Para ver un ejemplo que utiliza la herramienta de código abierto Shuf, consulte the section called “Paso 2: Preparar los datos” en el tutorial de migración de datos. El siguiente es un ejemplo que muestra cómo aleatorizar datos como un `DataFrame`.

     ```
     import org.apache.spark.sql.functions.randval
     shuffledDF = dataframe.orderBy(rand())
     ```

2. Cree el espacio de claves y la tabla de destino en Amazon Keyspaces.
   a. Conéctese a Amazon Keyspaces utilizando `cqlsh` y sustituya el punto de conexión del servicio, el nombre de usuario y la contraseña del siguiente ejemplo por sus propios valores.

```
cqlsh cassandra.us-east-2.amazonaws.com 9142 -u "111122223333" -
p "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

b. Cree un nuevo espacio de claves con el nombre `catalog` como se muestra en el siguiente ejemplo.

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

c. Una vez que el nuevo espacio de claves tenga el estado de disponible, utilice el siguiente código para crear la tabla de destino `book_awards`. Para obtener más información sobre la creación asíncrona de recursos y cómo comprobar si un recurso está disponible, consulte the section called “Creación de espacios de claves”.

```
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)
   );
```

Paso 5: Escribir y leer datos en Amazon Keyspaces utilizando Apache Cassandra Spark Connector

En este paso, usted comienza cargando los datos del archivo de ejemplo en un `DataFrame` con Spark Cassandra Connector. A continuación, escribe los datos del `DataFrame` en su tabla de Amazon Keyspaces. También puede utilizar esta parte de forma independiente, por ejemplo, para migrar datos a una tabla de Amazon Keyspaces. Por último, lea los datos de su tabla en un `DataFrame` con Spark Cassandra Connector. También puede utilizar esta parte de forma independiente, por ejemplo, para leer datos de una tabla de Amazon Keyspaces para realizar análisis de datos con Apache Spark.

1. Inicie el intérprete de comandos de Spark como se muestra en el siguiente ejemplo. Tenga en cuenta que en este ejemplo se utiliza la autenticación SigV4.

```
./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. Importe Spark Cassandra Connector con el siguiente código.

```
import org.apache.spark.sql.cassandra._
```

3. Para leer los datos del archivo CSV y almacenarlos en un `DataFrame`, puede utilizar el siguiente código de ejemplo.

```
var df =
 spark.read.option("header","true").option("inferSchema","true").csv("keyspaces_sample_tabl
```

Puede visualizar el resultado con el siguiente comando.

```
scala> df.show();
```

La salida debería ser similar a la siguiente.

```
+----------------+----+-----------+----+------------------+--------------------
+-------------+
|           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|
+----------------+----+-----------+----+------------------+--------------------
+-------------+
```

Puede confirmar el esquema de los datos en el `DataFrame` como se muestra en el siguiente ejemplo.

```
scala> df.printSchema
```

La salida debería tener este aspecto.

```
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. Utilice el siguiente comando para escribir los datos del `DataFrame` en la tabla de Amazon Keyspaces.

```
df.write.cassandraFormat("book_awards", "catalog").mode("APPEND").save()
```

5. Para confirmar que los datos se hayan guardado, puede volver a leerlos en un marco de datos, como se muestra en el siguiente ejemplo.

```
var newDf = spark.read.cassandraFormat("book_awards", "catalog").load()
```

A continuación, puede mostrar los datos que ahora contiene el marco de datos.

```
scala> newDf.show()
```

La salida de ese comando debería tener este aspecto.

```
+--------------------+-----------------+---------------+----------+------------
+----+----+
|          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|
+--------------------+-----------------+---------------+----------+------------
+----+----+
```

Solución de problemas comunes al utilizar Spark Cassandra Connector con Amazon Keyspaces

Si utiliza Amazon Virtual Private Cloud y se conecta a Amazon Keyspaces, los errores más comunes que se experimentan al utilizar el conector Spark se deben a los siguientes problemas de configuración.

- El usuario o rol de IAM utilizado en la VPC carece de los permisos necesarios para acceder a la tabla `system.peers` de Amazon Keyspaces. Para obtener más información, consulte the section called “Relleno de entradas de la tabla `system.peers` con información del punto de conexión de VPC de interfaz”.

- El usuario o rol de IAM carece de los permisos de lectura/escritura necesarios para la tabla del usuario y de acceso de lectura a las tablas del sistema en Amazon Keyspaces. Para obtener más información, consulte the section called “Paso 1: Configurar Amazon Keyspaces”.
- La configuración del controlador Java no deshabilita la verificación del nombre de host al crear la conexión SSL/TLS. Para ver ejemplos, consulte the section called “Paso 2: Configurar el controlador”.

Para conocer los pasos detallados para la solución de problemas de conexión, consulte the section called “Errores de conexión de punto de conexión de VPC”.

Además, puede utilizar las métricas de Amazon CloudWatch como ayuda para solucionar problemas con la configuración de Spark Cassandra Connector en Amazon Keyspaces. Para obtener más información sobre el uso de Amazon Keyspaces con CloudWatch, consulte the section called “Monitorización con CloudWatch”.

En la siguiente sección se describen las métricas de mayor utilidad que debe observar al utilizar Spark Cassandra Connector.

PerConnectionRequestRateExceeded

Amazon Keyspaces tiene una cuota de 3000 solicitudes por segundo por conexión. Cada ejecutor Spark establece una conexión con Amazon Keyspaces. Ejecutar múltiples reintentos podría agotar su cuota de tasa de solicitudes por conexión. Si supera esta cuota, Amazon Keyspaces emite una métrica `PerConnectionRequestRateExceeded` en CloudWatch.

Si observa eventos PerConnectionRequestRateExceeded presentes junto con otros errores del sistema o del usuario, es probable que Spark esté ejecutando múltiples reintentos por encima del número asignado de solicitudes por conexión.

Si observa eventos `PerConnectionRequestRateExceeded` sin otros errores, es posible que tenga que aumentar el número de conexiones en la configuración de su controlador para permitir un mayor rendimiento, o que tenga que aumentar el número de ejecutores en su trabajo de Spark.

StoragePartitionThroughputCapacityExceeded

Amazon Keyspaces tiene una cuota de 1000 WCU o WRU por segundo/3000 RCU o RRU por segundo por partición. Si observa eventos `StoragePartitionThroughputCapacityExceeded` de CloudWatch, esto podría indicar

que los datos no se aleatorizan en la carga. Para ver ejemplos de cómo se aleatorizan los datos, consulte. the section called “Paso 4: Preparar los datos de origen y la tabla de destino”

Errores y advertencias comunes

Si utiliza Amazon Virtual Private Cloud y se conecta a Amazon Keyspaces, el controlador de Cassandra podría emitir un mensaje de advertencia sobre el propio nodo de control en la tabla `system.peers`. Para obtener más información, consulte the section called “Errores y advertencias comunes”. Puede ignorar sin problemas esta advertencia.

Solución de problemas de Amazon Keyspaces (para Apache Cassandra)

En las siguientes secciones se proporciona información sobre cómo solucionar problemas de configuración comunes que puede encontrar al utilizar Amazon Keyspaces (para Apache Cassandra).

Para obtener una guía de solución de problemas específica para el acceso a IAM, consulte the section called “Solución de problemas”.

Para obtener más información sobre prácticas recomendadas de seguridad, consulte the section called “Prácticas recomendadas de seguridad”.

Temas

- Solución de problemas de conexión en Amazon Keyspaces
- Solución de problemas de administración de capacidad en Amazon Keyspaces
- Solución de problemas de lenguaje de definición de datos en Amazon Keyspaces

Solución de problemas de conexión en Amazon Keyspaces

¿Tiene problemas para conectarse? A continuación se indican algunos problemas comunes y la forma de resolverlos.

Errores al conectarse a un punto de conexión de Amazon Keyspaces

Las conexiones fallidas y los errores de conexión pueden dar lugar a diferentes mensajes de error. En la siguiente sección se describen los escenarios más comunes.

Temas

- No puedo conectarme a Amazon Keyspaces con cqlsh
- No puedo conectarme a Amazon Keyspaces con un controlador de cliente de Cassandra
- Mi punto de conexión de VPC no funciona correctamente
- No puedo conectarme utilizando cassandra-stress
- No puedo conectarme utilizando identidades de IAM

- Estoy intentando importar datos con cqlsh y se pierde la conexión con mi tabla de Amazon Keyspaces

No puedo conectarme a Amazon Keyspaces con cqlsh

Está intentando conectarse a un punto de conexión de Amazon Keyspaces con cqlsh y la conexión falla con un **`Connection error`**.

Si intenta conectarse a una tabla de Amazon Keyspaces y cqlsh no se ha configurado correctamente, la conexión falla. En la siguiente sección se ofrecen ejemplos de los problemas de configuración más comunes que causan errores de conexión al intentar establecer una conexión con cqlsh.

> **Note**
>
> Si intenta conectarse a Amazon Keyspaces desde una VPC, se requieren permisos adicionales. Para configurar correctamente una conexión utilizando puntos de conexión de VPC, siga los pasos indicados en el the section called “Conexión con puntos finales de VPC”.

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero obtiene un error de conexión **`timed out`**.

Esto podría deberse a que no ha suministrado el puerto correcto, lo que provoca el siguiente error.

```
#  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")})
```

Para resolver este problema, verifique que esté utilizando el puerto 9142 para la conexión.

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero obtiene un error **`Name or service not known`**.

Esto podría deberse a que ha utilizado un punto de conexión que está mal escrito o que no existe. En el siguiente ejemplo, el nombre del punto de conexión está mal escrito.

```
#  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
```

Para resolver este problema al utilizar puntos de conexión públicos para conectarse, seleccione un punto de conexión disponible en the section called “Puntos de conexión de servicio” y compruebe que el nombre del punto de conexión no tenga ningún error. Si utiliza puntos de conexión de VPC para conectarse, compruebe que la información del punto de conexión de VPC sea correcta en la configuración de cqlsh.

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`OperationTimedOut`**.

Amazon Keyspaces requiere que se habilite SSL para las conexiones a fin de garantizar una seguridad sólida. Es posible que falte el parámetro SSL si recibe el siguiente error.

```
# 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',)})
#
```

Para resolver este problema, añada el siguiente indicador al comando de conexión de cqlsh.

```
--ssl
```

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`SSL transport factory requires a valid certfile to be specified`**.

En este caso, falta la ruta al certificado SSL/TLS, lo que provoca el siguiente error.

```
# 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.
#
```

Para resolver este problema, añada la ruta al certfile de su ordenador.

```
certfile =  path_to_file/sf-class2-root.crt
```

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`No such file or directory`**.

Este podría deberse a que la ruta al archivo de certificado en su ordenador es incorrecta, lo que provoca el siguiente error.

```
# cat .cassandra/cqlshrc
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory

[ssl]
validate = true
certfile = /root/wrong_path/sf-class2-root.crt
#



# 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')})
#
```

Para resolver este problema, compruebe que la ruta al archivo de certificado en su ordenador sea correcta.

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`[X509] PEM lib`**.

Este podría deberse a que el archivo del certificado SSL/TLS `sf-class2-root.crt` no es válido, lo que provoca el siguiente error.

```
# 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)")})
#
```

Para resolver este problema, descargue el certificado digital de Starfield con el siguiente comando. Guarde `sf-class2-root.crt` localmente o en su directorio personal.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error de SSL **unknown**.

Este podría deberse a que el archivo del certificado SSL/TLS `sf-class2-root.crt` está vacío, lo que provoca el siguiente error.

```
# 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)")})
#
```

Para resolver este problema, descargue el certificado digital de Starfield mediante el siguiente comando. Guarde `sf-class2-root.crt` localmente o en su directorio personal.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`SSL: CERTIFICATE_VERIFY_FAILED`**.

Este podría deberse a que el archivo del certificado SSL/TLS no se pudo verificar, lo que provoca el siguiente error.

```
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)")})
```

Para resolver este problema, vuelva a descargar el archivo de certificado con el siguiente comando. Guarde `sf-class2-root.crt` localmente o en su directorio personal.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`Last error: timed out`**.

Esto podría deberse a que no configuró una regla de salida para Amazon Keyspaces en su grupo de seguridad de Amazon EC2, lo que provoca el siguiente error.

```
# 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")})
#
```

Para confirmar que este problema se debe a la configuración de la instancia Amazon EC2 y `nocqlsh`, puede intentar conectarse a su espacio de claves mediante AWS CLI, por ejemplo, el siguiente comando.

```
aws keyspaces list-tables --keyspace-name 'my_keyspace'
```

Si este comando también agota el tiempo de espera, la instancia de Amazon EC2 no está configurada correctamente.

Para confirmar que tienes permisos suficientes para acceder a Amazon Keyspaces, puedes usar AWS CloudShell para conectarte con. `cqlsh` Si se establecen esas conexiones, debe configurar la instancia de Amazon EC2.

Para resolver este problema, confirme que su instancia de Amazon EC2 tenga una regla de salida que permita el tráfico a Amazon Keyspaces. Si ese no es el caso, debe crear un nuevo grupo de seguridad para la instancia EC2 y añadir una regla que permita el tráfico saliente a los recursos de Amazon Keyspaces. Para actualizar la regla de salida para permitir el tráfico a Amazon Keyspaces, elija CQLSH/CASSANDRA en el menú desplegable Tipo.

Tras crear el nuevo grupo de seguridad con la regla de tráfico saliente, debe añadirlo a la instancia. Seleccione la instancia y, a continuación, elija Acciones, Seguridad y, por último, Cambiar grupos de seguridad. Agregue el nuevo grupo de seguridad con la regla de salida, pero asegúrese de que el grupo predeterminado también permanezca disponible.

Para obtener más información sobre cómo ver y editar las reglas de salida de EC2, consulte Adición de reglas a un grupo de seguridad en la Guía del usuario de Amazon EC2 para instancias Linux.

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`Unauthorized`**.

Este podría deberse a que le faltan permisos de Amazon Keyspaces en la política de usuario de IAM, lo que provoca el siguiente error.

```
# 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."',)})
#
```

Para resolver este problema, asegúrese de que el usuario de IAM `testuser-at-12345678910` tenga permisos para acceder a Amazon Keyspaces. Para ver ejemplos de políticas de IAM que concedan acceso a Amazon Keyspaces, consulte the section called “Ejemplos de políticas basadas en identidades”.

Para obtener una guía de solución de problemas específica para el acceso a IAM, consulte the section called “Solución de problemas”.

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **`Bad credentials`**.

Este podría deberse a que el nombre de usuario o la contraseña son incorrectos, lo que provoca el siguiente error.

```
# 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"',)})
#
```

Para resolver este problema, compruebe que *`USERNAME`* y *`PASSWORD`* en su código coincidan con el nombre de usuario y la contraseña que obtuvo al generar las credenciales específicas del servicio.

> **Important**
>
> Si sigue viendo errores al intentar conectarse con cqlsh, vuelva a ejecutar el comando con la opción `--debug` e incluya la salida detallada al contactar con AWS Support.

No puedo conectarme a Amazon Keyspaces con un controlador de cliente de Cassandra

En las siguientes secciones se muestran los errores más comunes al conectarse con un controlador de cliente de Cassandra.

Está intentando conectarse a una tabla de Amazon Keyspaces con un controlador y el complemento SigV4, pero recibe un error **`AttributeError`**.

Si las credenciales no están correctamente configuradas, se produce el siguiente error.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
 {'44.234.22.154:9142': AttributeError("'NoneType' object has no attribute
 'access_key'")})
```

Para resolver este problema, verifique que esté pasando las credenciales asociadas a su usuario o rol de IAM al utilizar el complemento SigV4. El complemento SigV4 requiere las siguientes credenciales.

- `AWS_ACCESS_KEY_ID`— Especifica una clave de AWS acceso asociada a un usuario o rol de IAM.
- `AWS_SECRET_ACCESS_KEY`: especifica la clave secreta asociada a la clave de acceso. Se trata básicamente de la "contraseña" de la clave de acceso.

Para obtener más información sobre claves de acceso y el complemento SigV4, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Está intentando conectarse a una tabla de Amazon Keyspaces con un controlador, pero recibe un error **`PartialCredentialsError`**.

Si falta la `AWS_SECRET_ACCESS_KEY`, puede producirse el siguiente error.

```
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')})
```

Para resolver este problema, compruebe que este pasando tanto el `AWS_ACCESS_KEY_ID` como la `AWS_SECRET_ACCESS_KEY` al utilizar el complemento SigV4. Para obtener más información sobre claves de acceso y el complemento SigV4, consulte the section called "Credenciales de IAM para autenticación de AWS.".

Está intentando conectarse a una tabla de Amazon Keyspaces con un controlador, pero recibe un error **`Invalid signature`**.

Esto podría deberse a que ha utilizado credenciales incorrectas, lo que da lugar al siguiente error.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
 {'44.234.22.134:9142':
 AuthenticationFailed('Failed to authenticate to 44.234.22.134:9142: Error from server:
 code=0100
 [Bad credentials] message="Authentication failure: Invalid signature"')})
```

Para resolver este problema, verifique que las credenciales que esté pasando estén asociadas con el usuario o rol de IAM que configuró para acceder a Amazon Keyspaces. Para obtener más información sobre claves de acceso y el complemento SigV4, consulte the section called "Credenciales de IAM para autenticación de AWS.".

Mi punto de conexión de VPC no funciona correctamente

Está intentando conectarse a Amazon Keyspaces con puntos de conexión de VPC, pero recibe errores de asignación de tokens o experimenta un bajo rendimiento.

Este podría deberse a que la conexión del punto de conexión de VPC no está configurada correctamente.

Para resolver estos problemas, verifique los siguientes detalles de configuración. Para seguir un step-by-step tutorial sobre cómo configurar una conexión a través de puntos de enlace de la interfaz de VPC para Amazon Keyspaces, consulte. the section called "Conexión con puntos finales de VPC"

1. Confirme que la entidad de IAM utilizada para conectarse a Amazon Keyspaces tenga acceso de lectura/escritura a la tabla del usuario y acceso de lectura a las tablas del sistema, como se muestra en el siguiente ejemplo.

```
{
   "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. Confirme que la entidad de IAM utilizada para conectarse a Amazon Keyspaces disponga de los permisos de lectura necesarios para acceder a la información del punto de conexión de VPC en su instancia de Amazon EC2, como se muestra en el siguiente ejemplo.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

> **Note**
>
> Las políticas administradas `AmazonKeyspacesReadOnlyAccess_v2` y `AmazonKeyspacesFullAccess` incluyen los permisos necesarios para permitir que

Amazon Keyspaces acceda a la instancia de Amazon EC2 para leer la información sobre los puntos de conexión de VPC de interfaz disponibles.

Para obtener más información sobre los puntos de conexión de VPC, consulte the section called "Uso de puntos de conexión de VPC de interfaz para Amazon Keyspaces".

3. Confirme que la configuración SSL del controlador Java establezca la validación del nombre de host en false, como se muestra en este ejemplo.

```
hostname-validation = false
```

Para obtener más información sobre la configuración del controlador, consulte the section called "Paso 2: Configurar el controlador".

4. Para confirmar que el punto de conexión de VPC se haya configurado correctamente, puede ejecutar la siguiente instrucción desde dentro de su VPC.

> **Note**
>
> No puede utilizar su entorno de desarrollo local o el editor CQL de Amazon Keyspaces para confirmar esta configuración, dado que utilizan un punto de conexión público.

```
SELECT peer FROM system.peers;
```

El resultado debería tener un aspecto similar al de este ejemplo y devolver entre 2 y 6 nodos con direcciones IP privadas, según la configuración de la VPC y AWS la región.

```
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)
```

No puedo conectarme utilizando **cassandra-stress**

Está intentando conectarse a Amazon Keyspaces con el comando **cassandra-stress**, pero recibe un error **SSL context**.

Esto ocurre al intentar conectarse a Amazon Keyspaces, pero no tiene correctamente configurado trustStore. Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes.

En este caso, aparece el siguiente error.

```
Error creating the initializing the SSL Context
```

Para resolver este problema, siga las instrucciones para configurar un trustStore como se muestra en este tema: the section called “Antes de empezar”.

Una vez configurado el trustStore, debería poder conectarse con el siguiente comando.

```
./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"
```

No puedo conectarme utilizando identidades de IAM

Está intentando conectarse a una tabla de Amazon Keyspaces utilizando una identidad de IAM, pero recibe un error **Unauthorized**.

Esto ocurre al intentar conectarse a una tabla de Amazon Keyspaces utilizando una identidad de IAM (por ejemplo, un usuario de IAM) sin antes implementar la política y conceder al usuario los permisos necesarios.

En este caso, aparece el siguiente error.

```
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."',)})
```

Para resolver este problema, verifique los permisos del usuario de IAM. Para conectarse con un controlador estándar, un usuario debe tener al menos acceso SELECT a las tablas del sistema, dado

que la mayoría de los controladores leen los espacios de claves/tablas del sistema al establecer la conexión.

Para ver ejemplos de políticas de IAM que concedan acceso a las tablas del sistema y del usuario de Amazon Keyspaces, consulte the section called “Acceso a las tablas de Amazon Keyspaces”.

Para revisar la sección de solución de problemas específicos de IAM, consulte the section called “Solución de problemas”.

Estoy intentando importar datos con cqlsh y se pierde la conexión con mi tabla de Amazon Keyspaces

Está intentando cargar datos en Amazon Keyspaces con cqlsh, pero recibe errores de conexión.

La conexión a Amazon Keyspaces falla una vez que el cliente cqlsh recibe tres errores consecutivos de cualquier tipo procedentes del servidor. El cliente cqlsh falla con el siguiente mensaje.

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

Para resolver este error, debe asegurarse de que los datos que se vayan a importar coincidan con el esquema de la tabla en Amazon Keyspaces. Revise el archivo de importación en busca de errores de análisis sintáctico. Puede intentar utilizar una única fila de datos mediante una instrucción INSERT para aislar el error.

El cliente intentará restablecer la conexión de forma automática.

Solución de problemas de administración de capacidad en Amazon Keyspaces

¿Tiene problemas con la capacidad sin servidor? A continuación se indican algunos problemas comunes y la forma de resolverlos.

Errores de capacidad sin servidor

En esta sección se describe la forma de reconocer los errores relacionados con la administración de capacidad sin servidor y cómo resolverlos. Por ejemplo, podría observar eventos de capacidad insuficiente cuando su aplicación excede su capacidad de rendimiento aprovisionada.

Dado que Apache Cassandra es un software basado en clústeres que está diseñado para ejecutarse en una flota de nodos, no dispone de mensajes de excepción relacionados con características sin

servidor como la capacidad de rendimiento. La mayoría de los controladores solo entienden los códigos de error que están disponibles en Apache Cassandra, por lo que Amazon Keyspaces utiliza ese mismo conjunto de códigos de error para mantener la compatibilidad.

Para asignar los errores de Cassandra a los eventos de capacidad subyacentes, puede utilizar Amazon CloudWatch para supervisar las métricas relevantes de Amazon Keyspaces. Los eventos de capacidad insuficiente que dan lugar a errores del lado del cliente se pueden clasificar en estos tres grupos en función del recurso que esté causando el evento:

- Tabla: si elige el modo de capacidad Aprovisionada para una tabla y su aplicación supera el rendimiento aprovisionado, es posible que observe errores de capacidad insuficiente. Para obtener más información, consulte the section called “Modos de capacidad de lectura/escritura”.
- Partición: es posible que experimente eventos de capacidad insuficiente si el tráfico en una partición determinada supera las 3000 RCU o las 1000 WCU. Como práctica recomendada, sugerimos distribuir el tráfico de forma uniforme entre las particiones. Para obtener más información, consulte the section called “Modelado de datos”.
- Conexión: es posible que experimente un rendimiento insuficiente si supera la cuota para el número máximo de operaciones por segundo, por conexión. Para aumentar el rendimiento, puede aumentar el número de conexiones predeterminadas al configurar la conexión con el controlador. Para obtener más información, consulte the section called “Ajuste de rendimiento de las consultas CQL” y the section called “Equilibrio de carga”.

Para determinar qué recurso está causando el evento de capacidad insuficiente que devuelve el error del lado del cliente, puede consultar el panel de control en la consola de Amazon Keyspaces. De forma predeterminada, la consola proporciona una vista agregada de las CloudWatch métricas más comunes relacionadas con la capacidad y el tráfico en la sección Capacidad y métricas relacionadas de la pestaña Capacidad de la tabla.

Para crear tu propio panel de control con Amazon CloudWatch, consulta las siguientes métricas de Amazon Keyspaces.

- `PerConnectionRequestRateExceeded`: solicitudes a Amazon Keyspaces que superan la cuota de tasa de solicitudes por conexión. Cada conexión de cliente a Amazon Keyspaces puede admitir hasta 3000 solicitudes CQL por segundo. Puede realizar más de 3000 solicitudes por segundo creando varias conexiones.
- `ReadThrottleEvents`: solicitudes a Amazon Keyspaces que superan la capacidad de lectura de una tabla.

- `StoragePartitionThroughputCapacityExceeded`: solicitudes a una partición de almacenamiento de Amazon Keyspaces que superan la capacidad de rendimiento de la partición. Las particiones de almacenamiento de Amazon Keyspaces pueden admitir hasta 1000 WCU/WRU por segundo y 3000 RCU/RRU por segundo. Para mitigar estas excepciones, le recomendamos que revise su modelo de datos para distribuir el tráfico de lectura/escritura entre más particiones.
- `WriteThrottleEvents`: solicitudes a Amazon Keyspaces que superan la capacidad de escritura de una tabla.

Para obtener más información CloudWatch, consultethe section called “Monitorización con CloudWatch”. Para obtener una lista de todas las CloudWatch métricas disponibles para Amazon Keyspaces, consulte. the section called “Métricas y dimensiones”

> **Note**
>
> Para empezar con un panel personalizado que muestre todas las métricas observadas habitualmente en Amazon Keyspaces, puede utilizar una CloudWatch plantilla prediseñada disponible GitHub en el AWS repositorio de muestras.

Temas

- Estoy recibiendo errores de capacidad insuficiente NoHostAvailable desde mi controlador de cliente
- Estoy recibiendo errores de tiempo de espera de escritura durante la importación de datos
- No puedo ver el tamaño real de almacenamiento de un espacio de claves o una tabla

Estoy recibiendo errores de capacidad insuficiente **`NoHostAvailable`** desde mi controlador de cliente

Está viendo excepciones **`Read_Timeout`** o **`Write_Timeout`** para una tabla.

Intentar escribir o leer repetidamente en una tabla de Amazon Keyspaces con capacidad insuficiente puede dar lugar a errores del lado del cliente específicos del controlador.

Úselo CloudWatch para monitorear las métricas de rendimiento aprovisionadas y reales, así como los eventos de capacidad insuficiente para la tabla. Por ejemplo, una solicitud de lectura que no tenga suficiente capacidad de rendimiento falla con una excepción `Read_Timeout` y se

contabiliza en la métrica `ReadThrottleEvents`. Una solicitud de escritura que no tenga suficiente capacidad de rendimiento falla con una excepción `Write_Timeout` y se contabiliza en la métrica `WriteThrottleEvents`. Para obtener más información sobre estas métricas, consulte the section called “Métricas y dimensiones”.

Para resolver estos problemas, considere una de las siguientes opciones.

- Aumente el rendimiento aprovisionado para la tabla, que es la cantidad máxima de capacidad de rendimiento que puede consumir una aplicación. Para obtener más información, consulte the section called “Unidades de capacidad de lectura y de escritura”.
- Deje que el servicio administre la capacidad de rendimiento en su nombre con escalado automático. Para obtener más información, consulte the section called “Administre la capacidad de rendimiento con el escalado automático”.
- Elija el modo de capacidad Bajo demanda para la tabla. Para obtener más información, consulte the section called “Modo de capacidad bajo demanda”.

Si necesita aumentar la cuota de capacidad predeterminada para su cuenta, consulte Cuotas.

Está viendo errores relacionados con exceso de capacidad de la partición.

La limitación de partición puede producirse cuando se supera temporalmente la capacidad de la partición (que podría gestionarse automáticamente mediante capacidad adaptativa o capacidad bajo demanda). Este error también podría apuntar a un problema con su modelo de datos. Las particiones de almacenamiento de Amazon Keyspaces pueden admitir hasta 1000 WCU/WRU por segundo y 3000 RCU/RRU por segundo. Para obtener más información sobre cómo mejorar su modelo de datos para distribuir el tráfico de lectura/escritura entre más particiones, consulte the section called “Modelado de datos”.

Las excepciones `Write_Timeout` también podrían deberse a una tasa elevada de operaciones de escritura concurrentes que incluyan datos estáticos y no estáticos en la misma partición lógica. Si se espera que el tráfico ejecute varias operaciones de escritura concurrentes que incluyan datos estáticos y no estáticos dentro de la misma partición lógica, recomendamos escribir los datos estáticos y no estáticos por separado. Escribir los datos por separado también ayuda a optimizar los costos de rendimiento.

Está viendo errores relacionados con una tasa de solicitudes de conexión excedida.

La limitación de conexión podría deberse a una de las siguientes causas.

- Es posible que no tenga suficientes conexiones configuradas por sesión.
- Es posible que esté recibiendo menos conexiones que pares disponibles, dado que no tiene configurados correctamente los permisos del punto de conexión de VPC. Para obtener más información sobre las políticas de punto de conexión de VPC, consulte the section called “Uso de puntos de conexión de VPC de interfaz para Amazon Keyspaces”.
- Si utiliza un controlador 4.x, compruebe si la validación del nombre de host está habilitada. El controlador habilita la verificación del nombre de host TLS de forma predeterminada. Esta configuración hace que Amazon Keyspaces aparezca como un clúster de nodo único ante el controlador. Le recomendamos que desactive la verificación de nombre de host.

Le recomendamos que siga estas prácticas recomendadas para garantizar la optimización de sus conexiones y su rendimiento:

- Configure el ajuste de rendimiento de las consultas CQL.

  Amazon Keyspaces admite hasta 3000 consultas CQL por conexión TCP por segundo, pero no hay límite en el número de conexiones que un controlador puede establecer.

  La mayoría de los controladores de código abierto para Cassandra establecen un grupo de conexiones a Cassandra y equilibran la carga de consultas sobre ese grupo de conexiones. Amazon Keyspaces expone 9 direcciones IP homólogas a los conductores. El comportamiento predeterminado de la mayoría de los controladores es establecer una única conexión con cada dirección IP homóloga. Por lo tanto, el rendimiento máximo de consultas CQL de un controlador que utilice la configuración predeterminada será de 27 000 consultas CQL por segundo.

  Para aumentar esta cifra, le recomendamos que aumente el número de conexiones por dirección IP que su controlador mantiene en su grupo de conexiones. Por ejemplo, si establece el máximo de conexiones por dirección IP en 2, duplicará el rendimiento máximo de su controlador a 54 000 consultas CQL por segundo.
- Optimice sus conexiones de nodo único.

  De forma predeterminada, la mayoría de los controladores de código abierto para Cassandra establecen una o más conexiones con cada dirección IP anunciada en la tabla `system.peers` al establecerse una sesión. Sin embargo, determinadas configuraciones pueden hacer que un controlador se conecte a una única dirección IP de Amazon Keyspaces. Esto puede suceder si el controlador intenta validar el nombre de host SSL de los nodos homólogos (por ejemplo, los controladores DataStax Java) o cuando se conecta a través de un punto final de VPC.

Para obtener la misma disponibilidad y rendimiento que un controlador con conexiones a varias direcciones IP, le recomendamos que haga lo siguiente:

- Aumente el número de conexiones por IP a 9 o más en función del rendimiento deseado del cliente.
- Cree una política de reintentos personalizada que garantice que los reintentos se ejecuten en el mismo nodo.
- Si utiliza puntos de conexión de VPC, conceda a la entidad de IAM que se utilice para conectarse a Amazon Keyspaces permisos de acceso para consultar en su VPC la información sobre el punto de conexión y la interfaz de red. Esto mejora el equilibrio de la carga y aumenta el rendimiento de lectura/escritura. Para obtener más información, consulte ???.

Estoy recibiendo errores de tiempo de espera de escritura durante la importación de datos

Está recibiendo un error de tiempo de espera al cargar datos con el comando de **`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 utiliza las excepciones `ReadTimeout` y `WriteTimeout` para indicar cuándo falla una solicitud de escritura debido a una capacidad de rendimiento insuficiente. Para ayudar a diagnosticar las excepciones de capacidad insuficiente, Amazon Keyspaces publica las siguientes métricas en Amazon. CloudWatch

- `WriteThrottleEvents`
- `ReadThrottledEvents`
- `StoragePartitionThroughputCapacityExceeded`

Para resolver los errores de capacidad insuficiente durante una carga de datos, reduzca la tasa de escritura por trabajador o la tasa de ingesta total y, a continuación, vuelva a intentar cargar las filas. Para obtener más información, consulte the section called “Paso 4: Configurar los ajustes de `cqlsh COPY FROM`”. Si busca una opción de carga de datos más sólida, considere la posibilidad de

utilizar DSBulk, que está disponible en el GitHub repositorio. Para obtener step-by-step instrucciones, consultethe section called “Carga de datos con DSBulk”.

No puedo ver el tamaño real de almacenamiento de un espacio de claves o una tabla

No puede ver el tamaño real de almacenamiento del espacio de claves o de la tabla.

Puede estimar el tamaño de almacenamiento a partir del cálculo de tamaño de fila de una tabla. Puede encontrar instrucciones detalladas para calcular el tamaño de fila en the section called “Cálculo del tamaño de las filas”.

Solución de problemas de lenguaje de definición de datos en Amazon Keyspaces

¿Tiene problemas para crear recursos? A continuación se indican algunos problemas comunes y la forma de resolverlos.

Errores de lenguaje de definición de datos

Amazon Keyspaces realiza operaciones de lenguaje de definición de datos (DDL) de forma asíncrona, por ejemplo, creando y eliminando espacios de claves y tablas. Si una aplicación intenta utilizar el recurso antes de que esté listo, la operación falla.

Puede supervisar el estado de creación de los nuevos espacios clave y tablas en el AWS Management Console, lo que indica cuándo un espacio clave o una tabla están pendientes o activos. También puede monitorear el estado de creación de un nuevo espacio de claves o tabla mediante programación, consultando la tabla de esquemas del sistema. Un espacio de claves o tabla se hace visible en el esquema del sistema cuando está listo para su uso.

> **Note**
>
> Para optimizar la creación de espacios de claves mediante esta utilidad AWS CloudFormation, puede convertir los scripts de CQL en plantillas. CloudFormation La herramienta está disponible en el GitHub repositorio.

Temas

- He creado un nuevo espacio de claves, pero no puedo verlo ni acceder al mismo

- He creado una nueva tabla, pero no puedo verla ni acceder a la misma
- Estoy intentando restaurar una tabla mediante la point-in-time recuperación de Amazon Keyspaces (PITR), pero la restauración falla
- Estoy intentando utilizar INSERT/UPDATE para editar ajustes personalizados de periodo de vida (TTL), pero la operación falla
- Estoy intentando cargar datos en mi tabla de Amazon Keyspaces y obtengo un error de exceso del número de columnas
- Estoy intentando eliminar datos en mi tabla Amazon Keyspaces y la eliminación falla para el rango

He creado un nuevo espacio de claves, pero no puedo verlo ni acceder al mismo

Está recibiendo errores de su aplicación que intenta acceder a un nuevo espacio de claves.

Si intenta acceder a un espacio de claves de Amazon Keyspaces recién creado que aún se esté creando de forma asíncrona, recibirá un error. A continuación, se muestra un ejemplo de error.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 keyspace mykeyspace"
```

El patrón de diseño recomendado para comprobar cuándo un nuevo espacio de claves está listo para su uso es sondear las tablas de esquema del sistema de Amazon Keyspaces (system_schema_mcs.*).

Para obtener más información, consulte the section called “Creación de espacios de claves”.

He creado una nueva tabla, pero no puedo verla ni acceder a la misma

Está recibiendo errores de su aplicación que intenta acceder a una nueva tabla.

Si intenta acceder a una tabla de Amazon Keyspaces recién creada que aún se esté creando de forma asíncrona, obtendrá un error. Por ejemplo, al intentar consultar una tabla que aún no esté disponible se produce un error `unconfigured table`.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured
 table mykeyspace.mytable"
```

Al intentar visualizar la tabla con `sync_table()` se produce un error `KeyError`.

```
KeyError: 'mytable'
```

El patrón de diseño recomendado para comprobar cuándo una nueva tabla está lista para su uso es sondear las tablas de esquema del sistema de Amazon Keyspaces (system_schema_mcs.*).

Este es un ejemplo de salida para una tabla que se está creando.

```
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)
```

Este es un ejemplo de salida para una tabla que está activa.

```
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)
```

Para obtener más información, consulte the section called “Creación de tablas”.

Estoy intentando restaurar una tabla mediante la point-in-time recuperación de Amazon Keyspaces (PITR), pero la restauración falla

Si está intentando restaurar una tabla de Amazon Keyspaces con point-in-time recuperación (PITR) y observa que el proceso de restauración comienza pero no se completa correctamente, es posible

que no haya configurado todos los permisos necesarios para el proceso de restauración para esta tabla concreta.

Además de los permisos de usuario, Amazon Keyspaces podría requerir permisos para realizar acciones durante el proceso de restauración en nombre de su entidad principal. Este es el caso si la tabla está cifrada con una clave administrada por el cliente, o si está utilizando políticas de IAM que restrinjan el tráfico entrante.

Por ejemplo, si utiliza claves de condición en su política de IAM para restringir el tráfico de origen a puntos de conexión o rangos de IP específicos, la operación de restauración falla. Para permitir que Amazon Keyspaces realice la operación de restauración de tablas en nombre de su entidad principal, debe añadir una clave de condición global `aws:ViaAWSService` en la política de IAM.

Para obtener más información sobre los permisos para restaurar tablas, consulte the section called “Permisos de restauración”.

Estoy intentando utilizar INSERT/UPDATE para editar ajustes personalizados de periodo de vida (TTL), pero la operación falla

Al intentar insertar o actualizar un valor TTL personalizado, la operación podría fallar con el siguiente error.

```
TTL is not yet supported.
```

Para especificar valores TTL personalizados para filas o columnas mediante las operaciones `INSERT` o `UPDATE`, primero debe habilitar TTL para la tabla. Puede habilitar TTL para una tabla mediante la propiedad personalizada `ttl`.

Para obtener más información sobre cómo habilitar los ajustes de TTL personalizados para tablas, consulte the section called “Cómo habilitar el periodo de vida (TTL) en tablas existentes mediante propiedades personalizadas”.

Estoy intentando cargar datos en mi tabla de Amazon Keyspaces y obtengo un error de exceso del número de columnas

Está cargando datos y ha superado el número de columnas que se pueden actualizar simultáneamente.

Este error se produce cuando el esquema de su tabla supera el tamaño máximo de 350 KB. Para obtener más información, consulte Cuotas.

Estoy intentando eliminar datos en mi tabla Amazon Keyspaces y la eliminación falla para el rango

Está intentando eliminar datos por clave de partición y recibe un error de eliminación de rango.

Este error se produce al intentar eliminar más de 1000 filas en una sola operación de eliminación.

```
Range delete requests are limited by the amount of items that can be deleted in a
 single range.
```

Para obtener más información, consulte the section called “Eliminación de rangos”.

Para eliminar más de 1000 filas en una única partición, considere las siguientes opciones.

- Eliminar por partición: si la mayoría de las particiones tienen menos de 1000 filas, puede intentar eliminar los datos por partición. Si las particiones contienen más de 1000 filas, intente en cambio eliminar por la columna de agrupación.
- Eliminar por columna de agrupación: si su modelo contiene varias columnas de agrupación, puede utilizar la jerarquía de columnas para eliminar múltiples filas. Las columnas de agrupación son una estructura anidada y puede eliminar muchas filas operando respecto a la columna de nivel superior.
- Eliminar por fila individual: puede iterar por las filas y eliminar cada fila por su clave principal completa (columnas de partición y columnas de agrupación).
- Como práctica recomendada, considere la posibilidad de dividir las filas en particiones. En Amazon Keyspaces, le recomendamos que distribuya el rendimiento entre las particiones de las tablas. De este modo, los datos y el acceso se distribuyen uniformemente entre los recursos físicos, lo que proporciona el mejor rendimiento. Para obtener más información, consulte the section called “Modelado de datos”.

Considere también las siguientes recomendaciones al planificar operaciones de eliminación para cargas de trabajo pesadas.

- Con Amazon Keyspaces, las particiones pueden contener un número prácticamente ilimitado de filas. Esto le permite escalar particiones “más amplias” que la orientación tradicional de Cassandra de 100 MB. No es raro que las series temporales o los registros de transacciones crezcan más de un gigabyte de datos en el tiempo.

- Con Amazon Keyspaces, no hay estrategias de compactación ni lápidas que tener en cuenta cuando deba realizar operaciones de eliminación para cargas de trabajo pesadas. Puede eliminar todos los datos que desee sin que ello afecte al rendimiento de la lectura.

Administración de recursos sin servidor en Amazon Keyspaces (para Apache Cassandra)

Amazon Keyspaces (para Apache Cassandra) opera sin servidor. En vez de implementar, administrar y mantener recursos de almacenamiento y cómputo para su carga de trabajo a través de nodos en un clúster, Amazon Keyspaces asigna recursos de almacenamiento y rendimiento de lectura/escritura directamente a las tablas.

En este capítulo se ofrecen detalles sobre la administración de recursos sin servidor en Amazon Keyspaces. Para obtener información sobre cómo supervisar los recursos sin servidor con Amazon CloudWatch, consultethe section called “Monitorización con CloudWatch”.

Temas

- Almacenamiento en Amazon Keyspaces
- Modos de capacidad de lectura/escritura en Amazon Keyspaces
- Administre la capacidad de rendimiento automáticamente con el escalado automático de Amazon Keyspaces
- Uso eficaz de la capacidad de ráfaga en Amazon Keyspaces
- Cómo estimar el consumo de capacidad en Amazon Keyspaces

Almacenamiento en Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) aprovisiona almacenamiento a las tablas de forma automática en función de los datos reales almacenados en su tabla. No es necesario aprovisionar almacenamiento a las tablas por adelantado. Amazon Keyspaces escala automáticamente el almacenamiento de su tabla al alza o a la baja a medida que su aplicación escribe, actualiza y elimina datos. A diferencia de los clústeres tradicionales de Apache Cassandra, Amazon Keyspaces no requiere almacenamiento adicional para admitir operaciones de sistema de bajo nivel, como la compactación. Solo paga por el almacenamiento que utiliza.

Amazon Keyspaces configura de forma predeterminada los espacios de claves con un factor de réplica de tres. No puede modificar el factor de réplica. Amazon Keyspaces replica los datos de la tabla tres veces de forma automática en varias zonas de disponibilidad para lograr una alta AWS disponibilidad. El precio por GB de almacenamiento de Amazon Keyspaces ya incluye la réplica. Para obtener más información, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Amazon Keyspaces monitorea continuamente el tamaño de sus tablas para determinar sus cargas de almacenamiento. Para obtener más información sobre cómo calcula Amazon Keyspaces el tamaño facturable de los datos, consulte the section called “Cálculo del tamaño de las filas”.

Modos de capacidad de lectura/escritura en Amazon Keyspaces

Amazon Keyspaces dispone de dos modos de capacidad de lectura/escritura para procesar las lecturas y escrituras en sus tablas:

- Bajo demanda (predeterminado)
- Aprovisionado

El modo de capacidad de lectura/escritura que elija controla cómo se le cobra por el rendimiento de lectura y escritura y cómo se administra la capacidad de rendimiento de las tablas.

Temas

- Modo de capacidad bajo demanda
- Modo de capacidad de rendimiento aprovisionada
- Administración y visualización de los modos de capacidad
- Consideraciones al cambiar los modos de capacidad

Modo de capacidad bajo demanda

El modo de capacidad bajo demanda de Amazon Keyspaces (para Apache Cassandra) es una opción de facturación flexible capaz de servir miles de solicitudes por segundo sin necesidad de planificar la capacidad. Esta opción ofrece pay-per-request precios para las solicitudes de lectura y escritura, de modo que solo pague por lo que utilice.

Al elegir el modo bajo demanda, Amazon Keyspaces puede escalar la capacidad de rendimiento de su tabla hasta cualquier nivel de tráfico alcanzado previamente de forma instantánea y, a continuación, volver a reducirla al disminuir el tráfico de solicitudes. Si el nivel de tráfico de una carga de trabajo alcanza un nuevo pico, el servicio se adapta rápidamente para aumentar la capacidad de rendimiento de su tabla. Puede habilitar el modo de capacidad bajo demanda tanto para las tablas nuevas como para las ya existentes.

El modo bajo demanda es una buena opción si se cumple alguna de las siguientes condiciones:

- Crea nuevas tablas con cargas de trabajo desconocidas.
- El tráfico de la aplicación es impredecible.
- Prefiere disfrutar de la comodidad de pagar solo por lo que usa.

Para empezar con el modo bajo demanda, puede crear una nueva tabla o actualizar una tabla existente para utilizar el modo de capacidad bajo demanda mediante la consola o con unas pocas líneas de código Cassandra Query Language (CQL). Para obtener más información, consulte the section called “Tablas”.

Temas

- Unidades de solicitud de lectura y de escritura
- Picos de tráfico y propiedades de escalado
- Rendimiento inicial del modo de capacidad bajo demanda

Unidades de solicitud de lectura y de escritura

Con las tablas en modo de capacidad bajo demanda, no necesita especificar de antemano cuánto rendimiento de lectura y escritura espera que utilice su aplicación. Amazon Keyspaces le cobra por las lecturas y escrituras que realice en sus tablas en términos de unidades de solicitud de lectura (RRU) y unidades de solicitud de escritura (WRU).

- Una RRU representa una solicitud de lectura `LOCAL_QUORUM` o dos solicitudes de lectura `LOCAL_ONE` para una fila de hasta 4 KB de tamaño. Si necesita leer una fila mayor que 4 KB, la operación de lectura utiliza RRU adicionales. El número total de RRU necesarias depende del tamaño de la fila y de si desea utilizar la coherencia de lectura `LOCAL_QUORUM` o `LOCAL_ONE`. Por ejemplo, la lectura de una fila de 8 KB requiere 2 RRU con coherencia de lectura `LOCAL_QUORUM` y 1 RRU si elige coherencia de lectura `LOCAL_ONE`.
- Una WRU representa una escritura para una fila de hasta 1 KB de tamaño. Todas las escrituras utilizan coherencia `LOCAL_QUORUM` y no hay cargos adicionales por utilizar transacciones ligeras (LWT). Si necesita escribir una fila de más de 1 KB, la operación de escritura utiliza WRU adicionales. El número total de WRU necesarias depende del tamaño de la fila. Por ejemplo, si el tamaño de su fila es de 2 KB, necesitará 2 WRU para realizar una solicitud de escritura.

Para obtener información sobre los niveles de coherencia admitidos, consulte the section called “Niveles de coherencia de Cassandra admitidos”.

Picos de tráfico y propiedades de escalado

Las tablas de Amazon Keyspaces que utilizan el modo de capacidad bajo demanda se adaptan automáticamente al volumen de tráfico de su aplicación. El modo de capacidad bajo demanda acomoda al instante hasta el doble del tráfico máximo alcanzado previamente en una tabla. Por ejemplo, el patrón de tráfico de su aplicación puede variar entre 5000 y 10 000 lecturas `LOCAL_QUORUM` por segundo, donde 10 000 lecturas por segundo es el pico de tráfico anterior.

Con este patrón, el modo de capacidad bajo demanda se adapta instantáneamente a un tráfico sostenido de hasta 20 000 lecturas por segundo. Si su aplicación sostiene un tráfico de 20 000 lecturas por segundo, ese pico se convierte en su nuevo pico anterior, habilitando el tráfico posterior para alcanzar hasta 40 000 lecturas por segundo.

Si necesita más del doble de su pico anterior en una tabla, Amazon Keyspaces asigna automáticamente más capacidad a medida que aumenta su volumen de tráfico. Esto permite garantizar que su tabla tenga suficiente capacidad de rendimiento para procesar las solicitudes adicionales. Sin embargo, es posible que observe errores de capacidad de rendimiento insuficiente si supera el doble de su pico anterior en un periodo de 30 minutos.

Por ejemplo, suponga que el patrón de tráfico de su aplicación varía entre 5000 y 10 000 lecturas por segundo de gran coherencia, siendo 20 000 lecturas por segundo el pico de tráfico alcanzado anteriormente. En este caso, el servicio le recomienda que espacie el crecimiento de su tráfico durante al menos 30 minutos antes de subir a 40 000 lecturas por segundo.

Para obtener información sobre cómo estimar el consumo de capacidad de lectura y escritura de una tabla, consultethe section called “Calcule el consumo de capacidad”.

Para obtener más información sobre las cuotas predeterminadas de su cuenta y cómo aumentarlas, consulte Cuotas.

Rendimiento inicial del modo de capacidad bajo demanda

Si crea una nueva tabla con el modo de capacidad bajo demanda habilitado o cambia una tabla existente al modo de capacidad bajo demanda por primera vez, la tabla tendrá los siguientes ajustes de pico anteriores, aunque no haya servido tráfico anteriormente utilizando el modo de capacidad bajo demanda:

- Tabla recién creada con el modo de capacidad bajo demanda: el pico anterior es de 2000 WRU y 6000 RRU. Puede manejar hasta el doble del pico anterior de forma inmediata. Esto habilita las tablas bajo demanda recién creadas para servir hasta 4000 WRU y 12 000 RRU.

- Tabla existente cambiada al modo de capacidad bajo demanda: el pico anterior es la mitad de las WCU y RCU anteriores aprovisionadas para la tabla o la configuración de una tabla recién creada con el modo de capacidad bajo demanda, la que sea mayor.

Modo de capacidad de rendimiento aprovisionada

Si elige el modo de capacidad de rendimiento aprovisionado, especifica el número de lecturas y escrituras por segundo que se requieren para su aplicación. Esto le permite administrar el uso de Amazon Keyspaces para que pueda mantenerse en una tasa de solicitudes definida, o por debajo de ella, a fin de optimizar el precio y mantener la previsibilidad. Para obtener más información sobre el escalado automático para el rendimiento aprovisionado, consulte the section called “Administre la capacidad de rendimiento con el escalado automático”.

El modo de capacidad de rendimiento aprovisionada es una buena opción si se cumple alguna de las siguientes condiciones:

- El tráfico de la aplicación es predecible.
- Ejecuta aplicaciones cuyo tráfico es coherente o aumenta gradualmente.
- Puede prever las necesidades de capacidad para optimizar el precio.

Unidades de capacidad de lectura y de escritura

En las tablas de modo de capacidad de rendimiento aprovisionada, se especifica la capacidad de rendimiento en términos de unidades de capacidad de lectura (RCU) y unidades de capacidad de escritura (WCU):

- Una RCU representa una lectura `LOCAL_QUORUM` por segundo o dos lecturas `LOCAL_ONE` por segundo, para una fila de hasta 4 KB de tamaño. Si necesita leer una fila de más de 4 KB, la operación de lectura utiliza RCU adicionales.

  El número total de RCU necesarias depende del tamaño de la fila y de si desea utilizar la lectura `LOCAL_QUORUM` o `LOCAL_ONE`. Por ejemplo, si el tamaño de su fila es de 8 KB, necesitará 2 RCU para mantener una lectura `LOCAL_QUORUM` por segundo y 1 RCU si elige lecturas `LOCAL_ONE`.
- Una WCU representa una escritura por segundo para una fila de hasta 1 KB de tamaño. Todas las escrituras utilizan coherencia `LOCAL_QUORUM` y no hay cargos adicionales por utilizar transacciones ligeras (LWT). Si necesita escribir una fila de más de 1 KB, la operación de escritura utiliza WCU adicionales.

El número total de WCU necesarias depende del tamaño de la fila. Por ejemplo, si el tamaño de su fila es de 2 KB, necesitará 2 WCU para realizar una solicitud de escritura por segundo. Para obtener más información sobre cómo estimar el consumo de capacidad de lectura y escritura de una tabla, consultethe section called “Calcule el consumo de capacidad”.

Si su aplicación lee o escribe filas más grandes (hasta el tamaño máximo de fila de Amazon Keyspaces de 1 MB), consumirá más unidades de capacidad. Para obtener más información sobre cómo calcular el tamaño de las filas, consulte the section called “Cálculo del tamaño de las filas”. Por ejemplo, suponga que crea una tabla aprovisionada con 6 RCU y 6 WCU. Con estos ajustes, la aplicación podría hacer lo siguiente:

- Realizar lecturas `LOCAL_QUORUM` de hasta 24 KB por segundo (4 KB × 6 RCU).
- Realizar lecturas `LOCAL_ONE` de hasta 48 KB por segundo (el doble de rendimiento de lectura).
- Escribir hasta 6 KB por segundo (1 KB × 6 WCU).

El rendimiento aprovisionado es la cantidad máxima de capacidad de rendimiento que una aplicación puede consumir de una tabla. Si su aplicación supera la capacidad de rendimiento aprovisionada, es posible que observe errores de capacidad insuficiente.

Por ejemplo, una solicitud de lectura que no tenga suficiente capacidad de rendimiento falla con una excepción `Read_Timeout` y se contabiliza en la métrica `ReadThrottleEvents`. Una solicitud de escritura que no tenga suficiente capacidad de rendimiento falla con una excepción `Write_Timeout` y se contabiliza en la métrica `WriteThrottleEvents`.

Puedes usar Amazon CloudWatch para monitorear las métricas de rendimiento aprovisionadas y reales y los eventos de capacidad insuficiente. Para obtener más información sobre estas métricas, consulte the section called “Métricas y dimensiones”.

> **Note**
>
> Los errores repetidos debidos a una capacidad insuficiente pueden dar lugar a excepciones específicas del controlador del lado del cliente, por ejemplo, el controlador DataStax Java falla con un. `NoHostAvailableException`

Para cambiar la configuración de capacidad de rendimiento de las tablas, puede utilizar la AWS Management Console o la instrucción ALTER TABLE con CQL. Para obtener más información consulte the section called “ALTER TABLE”.

Para obtener más información sobre las cuotas predeterminadas de su cuenta y cómo aumentarlas, consulte Cuotas.

Administración y visualización de los modos de capacidad

Puede consultar la tabla del sistema en el espacio de claves del sistema de Amazon Keyspaces para revisar la información sobre el modo de capacidad de una tabla. También puede ver si una tabla utiliza el modo de capacidad de rendimiento bajo demanda o aprovisionada. Si la tabla está configurada con el modo de capacidad de rendimiento aprovisionada, puede ver la capacidad de rendimiento aprovisionada de la tabla.

Ejemplo

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'mykeyspace' and
 table_name = 'mytable';
```

Una tabla configurada con el modo de capacidad bajo demanda devuelve lo siguiente.

```
                    {
                        'capacity_mode': {
                            'last_update_to_pay_per_request_timestamp':
'1579551547603',
                            'throughput_mode': 'PAY_PER_REQUEST'
                         }
                    }
```

Una tabla configurada con el modo de capacidad de rendimiento aprovisionada devuelve lo siguiente.

```
                    {
                        'capacity_mode': {
                            'last_update_to_pay_per_request_timestamp':
'1579048006000',
```

```
                    'read_capacity_units': '5000',
                    'throughput_mode': 'PROVISIONED',
                    'write_capacity_units': '6000'
                }
            }
```

El valor `last_update_to_pay_per_request_timestamp` se mide en milisegundos.

Para cambiar la capacidad de rendimiento aprovisionada de una tabla, utilice the section called "ALTER TABLE".

Consideraciones al cambiar los modos de capacidad

Al cambiar una tabla del modo de capacidad aprovisionada al modo de capacidad bajo demanda, Amazon Keyspaces realiza varios cambios en la estructura de su tabla y sus particiones. Este proceso puede tardar varios minutos. Durante el periodo de cambio, su tabla ofrece un rendimiento coherente con las cantidades de WCU y RCU previamente aprovisionadas.

Al volver a cambiar del modo de capacidad bajo demanda al modo de capacidad aprovisionada, su tabla ofrece un rendimiento coherente con el pico alcanzado anteriormente cuando la tabla estaba en modo de capacidad bajo demanda.

> **Note**
>
> Puede cambiar entre modos de capacidad de lectura o escritura dos veces en un periodo de 24 horas.

Administre la capacidad de rendimiento automáticamente con el escalado automático de Amazon Keyspaces

Muchas cargas de trabajo de base de datos son cíclicas o difíciles de predecir con antelación. Por ejemplo, tomemos una aplicación de redes sociales en la que la mayoría de los usuarios están activos en el horario diurno. La base de datos debe satisfacer los requisitos de la actividad diurna, pero no se requieren los mismos niveles de rendimiento por la noche.

Otro ejemplo podría ser una nueva aplicación de juegos para móviles cuya adopción está siendo muy rápida. Si el juego se vuelve muy popular, podría superar los recursos de base de datos

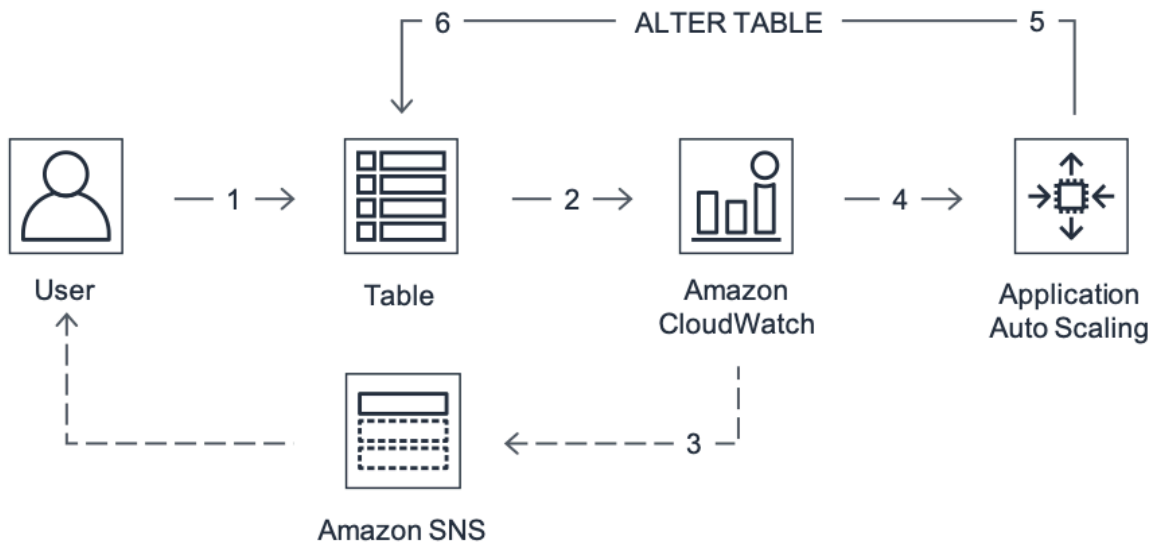
disponibles, lo que provocaría un rendimiento lento y clientes descontentos. Estos tipos de cargas de trabajo suelen requerir intervención manual para escalar los recursos de la base de datos en sentido ascendente o descendente en respuesta a las variaciones en los niveles de uso.

Amazon Keyspaces (para Apache Cassandra) le ayuda a aprovisionar capacidad de rendimiento de forma eficiente para cargas de trabajo variables ajustando de forma automática la capacidad de rendimiento en respuesta al tráfico real de la aplicación. Amazon Keyspaces utiliza el servicio Application Auto Scaling para aumentar y reducir la capacidad de lectura y escritura de una tabla en su nombre. Para obtener más información sobre Application Auto Scaling, consulte la Guía del usuario de Application Auto Scaling.

> **Note**
>
> Para una introducción rápida al escalado automático de Amazon Keyspaces, consulte the section called “Uso de la consola”.. Para gestionar las políticas de escalado de Amazon Keyspaces con Cassandra Query Language (CQL), consulte. the section called “Uso de CQL” Para obtener información sobre cómo gestionar las políticas de escalado de Amazon Keyspaces mediante la CLI, consulte. the section called “Uso de la CLI”

Funcionamiento del escalado automático de Amazon Keyspaces

En el siguiente diagrama se proporciona información general de alto nivel sobre el modo en que el escalado automático de Amazon Keyspaces administra la capacidad de rendimiento de una tabla.



Para habilitar el escalado automático de una tabla, cree una política de escalado. La política de escalado especifica si desea escalar la capacidad de lectura, la capacidad de escritura o ambas, así como la configuración mínima y máxima de la unidad de capacidad aprovisionada para la tabla.

La política de escalado también define una utilización objetivo. La utilización objetivo es la relación entre las unidades de capacidad consumidas y las unidades de capacidad aprovisionadas en un momento dado, expresada como porcentaje. El escalado automático utiliza un algoritmo de seguimiento de objetivos para ajustar el rendimiento aprovisionado de la tabla al alza o a la baja en respuesta a las cargas de trabajo reales. Lo hace para que la utilización real de la capacidad se mantenga en su utilización objetivo o cerca de ella.

Puede establecer los valores de utilización objetivo de escalado automático entre el 20 y el 90 por ciento de su capacidad de lectura y escritura. La tasa de utilización objetivo predeterminada es del 70 por ciento. Puede fijar la utilización objetivo en un porcentaje inferior si su tráfico cambia rápidamente y desea que la capacidad empiece a escalarse antes. También puede fijar la tasa de utilización objetivo en una tasa más alta si el tráfico de su aplicación cambia con lentitud y desea reducir el costo del rendimiento.

Para obtener más información sobre las políticas de escalado, consulte Políticas de escalado de seguimiento de objetivos para Application Auto Scaling en la Guía del usuario de Application Auto Scaling.

Cuando crea una política de escalado, Amazon Keyspaces crea dos pares de CloudWatch alarmas de Amazon en su nombre. Cada par representa sus límites superior e inferior para las configuraciones de rendimiento aprovisionadas y consumidas. Estas CloudWatch alarmas se activan cuando la utilización real de la tabla se desvía de la utilización objetivo durante un período prolongado. Para obtener más información sobre Amazon CloudWatch, consulta la Guía del CloudWatch usuario de Amazon.

Cuando se activa una de las CloudWatch alarmas, Amazon Simple Notification Service (Amazon SNS) te envía una notificación (si la has activado). A continuación, la CloudWatch alarma invoca Application Auto Scaling para evaluar su política de escalado. Esto, a su vez, emite una solicitud Alterar tabla a Amazon Keyspaces para ajustar la capacidad aprovisionada de la tabla al alza o a la baja según corresponda. Para obtener más información sobre las notificaciones de Amazon SNS, consulte Configuración de las notificaciones de Amazon SNS.

Amazon Keyspaces procesa la solicitud Alterar tabla con el aumento (o la disminución) de la capacidad de rendimiento aprovisionada de la tabla para que se aproxime a su utilización objetivo.

> **Note**
>
> El escalado automático de Amazon Keyspaces modifica la configuración de rendimiento aprovisionada solo cuando la carga de trabajo real permanece elevada (o reducida) durante un período prolongado de varios minutos. El algoritmo de seguimiento de objetivo de intenta mantener el objetivo de utilización en el valor elegido o en valores próximos a él a largo plazo. Los picos de actividad repentinos y breves se atienden gracias a la capacidad de ampliación incorporada de la tabla.

Cómo funciona el escalado automático para tablas multirregionales

Para garantizar que siempre haya suficiente capacidad de lectura y escritura para todas las réplicas de tablas Regiones de AWS de una tabla multirregional en el modo de capacidad aprovisionada, le recomendamos que configure el escalado automático de Amazon Keyspaces.

Cuando utilizas una tabla de varias regiones en modo aprovisionado con escalado automático, no puedes deshabilitar el escalado automático para una réplica de una sola tabla. Sin embargo, puede ajustar la configuración de escalado automático de lectura de la tabla para diferentes regiones. Por ejemplo, puede especificar diferentes configuraciones de capacidad de lectura y escalado automático de lectura para cada región en la que se replica la tabla.

Los ajustes de autoescalado de lectura que se configuran para una réplica de tabla en una región específica sobrescriben los ajustes generales de escalado automático de la tabla. Sin embargo, la capacidad de escritura debe permanecer sincronizada en todas las réplicas de la tabla para garantizar que haya suficiente capacidad para replicar las escrituras en todas las regiones.

El escalado automático de Amazon Keyspaces actualiza de forma independiente la capacidad aprovisionada de la tabla en cada una de ellas en Región de AWS función del uso en esa región. Como resultado, la capacidad aprovisionada en cada región para una tabla de varias regiones puede ser diferente cuando el autoescalado está activo.

Puede configurar los ajustes de escalado automático de una tabla multirregión y sus réplicas mediante la consola, la API o el CQL de Amazon Keyspaces. AWS CLI Para obtener más información sobre cómo crear y actualizar la configuración de escalado automático para tablas de varias regiones, consultethe section called “Formas de uso de la réplica multirregión”.

> **Note**
>
> Si usa el escalado automático para tablas de varias regiones, siempre debe usar las operaciones de la API Amazon Keyspaces para configurar los ajustes de escalado automático. Si utiliza las operaciones de la API Application Auto Scaling directamente para configurar los ajustes de autoescalado, no podrá especificar los Regiones de AWS de la tabla multirregión. Esto puede provocar que las configuraciones no sean compatibles.

Notas de uso

Antes de comenzar a utilizar el escalado automático de Amazon Keyspaces, debe tener en cuenta lo siguiente:

- El escalado automático de Amazon Keyspaces puede aumentar la capacidad de lectura o de escritura tantas veces como sea necesario, de acuerdo con su política de escalado. Todas las cuotas de Amazon Keyspaces permanecen en vigor, tal y como se describe en Cuotas.
- El escalado automático de Amazon Keyspaces no le impide modificar manualmente los ajustes de capacidad aprovisionada. Estos ajustes manuales no afectan a las CloudWatch alarmas existentes asociadas a la política de escalado.
- Si utiliza la consola para crear una tabla con capacidad de rendimiento aprovisionada, el escalado automático de Amazon Keyspaces se habilita de forma predeterminada. Puede modificar la configuración del escalado automático en cualquier momento. Para obtener más información, consulte the section called “Uso de la consola”.
- Si utiliza políticas de escalado AWS CloudFormation para crear políticas de escalado, debe administrar las políticas de escalado AWS CloudFormation para que la pila esté sincronizada con la plantilla de pila. Si cambias las políticas de escalado de Amazon Keyspaces, se sobrescribirán con los valores originales de la plantilla de AWS CloudFormation pila cuando se restablezca la pila.
- Si lo utiliza CloudTrail para supervisar el escalado automático de Amazon Keyspaces, es posible que vea alertas de las llamadas realizadas por Application Auto Scaling como parte de su proceso de validación de la configuración. Puede filtrar estas alertas utilizando el campo `invokedBy`, que contiene `application-autoscaling.amazonaws.com` para estas comprobaciones de validación.

Administración de las políticas de escalado automático de Amazon Keyspaces con la consola

Puede utilizar la consola para habilitar el escalado automático de Amazon Keyspaces para tablas nuevas y existentes. También puede utilizar la consola para modificar la configuración del escalado automático o deshabilitarlo.

> **Note**
>
> Para funciones más avanzadas, como establecer tiempos de enfriamiento de escalado vertical y escalable, utilice CQL o the () AWS Command Line Interface para AWS CLI gestionar las políticas de escalado de Amazon Keyspaces de forma programática. Para obtener más información, consulte Administración del escalado automático de Amazon Keyspaces con Cassandra Query Language (CQL) o Administración de las políticas de escalado de Amazon Keyspaces con la CLI.

Temas

- Antes de comenzar: Concesión de permisos a los usuarios para el escalado automático de Amazon Keyspaces
- Creación de una nueva tabla con escalado automático de Amazon Keyspaces habilitado
- Habilitación del escalado automático de Amazon Keyspaces en tablas existentes
- Modificación o deshabilitación de los ajustes de escalado automático de Amazon Keyspaces
- Visualización de las actividades de escalado automático de Amazon Keyspaces en la consola

Antes de comenzar: Concesión de permisos a los usuarios para el escalado automático de Amazon Keyspaces

Para empezar, confirme que el usuario tenga los permisos apropiados para crear y administrar la configuración del escalado automático. En AWS Identity and Access Management (IAM), la política AWS gestionada `AmazonKeyspacesFullAccess` es necesaria para gestionar las políticas de escalado de Amazon Keyspaces.

> **Important**
>
> Se requieren permisos `application-autoscaling:*` para deshabilitar el escalado automático en una tabla. Debe desactivar el escalado automático de una tabla antes de poder eliminarla.

Para configurar un usuario de IAM para el acceso a la consola de Amazon Keyspaces y el escalado automático de Amazon Keyspaces, añada la siguiente política.

Para vincular la política **`AmazonKeyspacesFullAccess`**

1. Inicie sesión en la consola de IAM AWS Management Console y ábrala en https://console.aws.amazon.com/iam/.
2. En el panel de la consola de IAM, elija Users y, a continuación, seleccione el usuario de IAM en la lista.
3. En la página Summary (Resumen), elija Añadir permisos.
4. Elija Attach existing policies directly (Adjuntar políticas existentes directamente).
5. En la lista de políticas, elija y AmazonKeyspacesFullAccess, a continuación, elija Siguiente: revisar.
6. Elija Añadir permisos.

Creación de una nueva tabla con escalado automático de Amazon Keyspaces habilitado

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a servicios (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realice acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

Para crear una nueva tabla con el escalado automático habilitado

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tablas y, a continuación, seleccione Crear tabla.
3. En la página Crear tabla, en la sección Detalles de la tabla, seleccione un espacio de claves y proporcione un nombre para la nueva tabla.
4. En la sección Columnas, cree el esquema de la tabla.
5. En la sección Clave principal, defina la clave principal de la tabla y seleccione las columnas de agrupamiento opcionales.
6. En la sección Configuración de la tabla, elija Personalizar configuración.
7. Continúe en Configuración de capacidad de lectura/escritura.
8. En Modo de capacidad, elija Aprovisionada.
9. En la sección Capacidad de lectura, confirme que esté seleccionada la opción Escalado automático.

   En este paso, usted selecciona las unidades de capacidad de lectura mínima y máxima para la tabla, así como la utilización objetivo.

   - Unidades de capacidad mínima: introduzca el valor de nivel mínimo de rendimiento para el que la tabla debería estar siempre preparada para admitir. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
   - Unidades de capacidad máxima: introduzca la cantidad máxima de rendimiento que desee aprovisionar para la tabla. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
   - Utilización objetivo: introduzca una tasa de utilización objetivo entre el 20 % y el 90 %. Cuando el tráfico supera la tasa de utilización objetivo definida, la capacidad aumenta automáticamente. Cuando el tráfico cae por debajo del objetivo definido, se vuelve a reducir automáticamente.

> **Note**
>
> Para obtener más información sobre las cuotas predeterminadas de su cuenta y cómo aumentarlas, consulte Cuotas.

10. En la sección Capacidad de escritura, elija los mismos ajustes que se definieron en el paso anterior para la capacidad de lectura o configure los valores de capacidad manualmente.
11. Elija Crear tabla. Su tabla se crea con los parámetros de escalado automático especificados.

Habilitación del escalado automático de Amazon Keyspaces en tablas existentes

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a servicios (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realice acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

Para habilitar el escalado automático de Amazon Keyspaces para una tabla existente.

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. Elija la tabla con la que desee trabajar y vaya a la pestaña Capacidad.
3. En la sección Configuración de capacidad, selecciona Editar.
4. En el modo de capacidad, asegúrese de que la tabla utilice el modo de capacidad aprovisionada.
5. Seleccione Escalado automático y, a continuación, consulte el paso 6 en Creación de una nueva tabla con escalado automático de Amazon Keyspaces habilitado para editar la capacidad de lectura y escritura.
6. Cuando se hayan definido los ajustes de escalado automático, elija Guardar.

Modificación o deshabilitación de los ajustes de escalado automático de Amazon Keyspaces

Puede usarlo AWS Management Console para modificar la configuración de escalado automático de Amazon Keyspaces. Para ello, elija la tabla que desee editar y vaya a la pestaña Capacidad. En la sección Configuración de capacidad, selecciona Editar. Ahora puede modificar la configuración en las secciones Capacidad de lectura o Capacidad de escritura. Para obtener más información sobre estas opciones, consulte Creación de una nueva tabla con escalado automático de Amazon Keyspaces habilitado.

Para deshabilitar el escalado automático de Amazon Keyspaces, desactive la casilla Escalar automáticamente. Al deshabilitar el escalado automático, se anula el registro de la tabla como objetivo escalable con Application Auto Scaling. Para eliminar el rol vinculado a servicios utilizado por Application Auto Scaling para acceder a su tabla Amazon Keyspaces, siga los pasos indicados en the section called “Eliminación de un rol vinculado a servicios para Amazon Keyspaces”.

> **Note**
>
> Para eliminar el rol vinculado al servicio que utiliza Application Auto Scaling, debe deshabilitar el escalado automático en todas las tablas de la cuenta. Regiones de AWS

Visualización de las actividades de escalado automático de Amazon Keyspaces en la consola

Puede supervisar la forma en que el escalado automático de Amazon Keyspaces utiliza los recursos mediante Amazon CloudWatch, que genera métricas sobre su uso y rendimiento. Siga los pasos de la Guía del Application Auto Scaling usuario para crear un CloudWatch panel.

Administración del escalado automático de Amazon Keyspaces con Cassandra Query Language (CQL)

Para crear y administrar la configuración de escalado automático para las tablas de Amazon Keyspaces con Cassandra Query Language (CQL), puede utilizar. `cqlsh` En este tema se ofrece una descripción general de las tareas de autoescalado que puede gestionar mediante programación mediante CQL.

Para obtener más información sobre las instrucciones de CQL descritas en este tema, consulte. the section called “Instrucciones DDL”

Temas

- Antes de empezar
- Cree una tabla nueva con escalado automático mediante CQL
- Habilite el escalado automático en una tabla existente mediante CQL
- Vea la configuración de escalado automático de Amazon Keyspaces de su mesa mediante CQL
- Desactive el escalado automático de Amazon Keyspaces para una tabla mediante CQL

Antes de empezar

Debe completar las siguientes tareas antes de empezar.

Configuración de permisos de

Si aún no lo ha hecho, debe configurar los permisos apropiados para que el usuario pueda crear y administrar los ajustes de escalado automático. En AWS Identity and Access Management (IAM), la política AWS gestionada `AmazonKeyspacesFullAccess` es necesaria para gestionar las políticas de escalado de Amazon Keyspaces. Para conocer los pasos en detalle, consulte the section called "Antes de comenzar: Concesión de permisos a los usuarios para el escalado automático de Amazon Keyspaces".

Configurar **`cqlsh`**

Si aún no lo ha hecho, debe instalarlo y configurarlo. `cqlsh` Para ello, siga las instrucciones que aparecen enthe section called "Uso de la `cqlsh-expansion`". A continuación, puede utilizar el AWS CloudShell para ejecutar los comandos de las siguientes secciones.

Cree una tabla nueva con escalado automático mediante CQL

Al crear una nueva tabla de Amazon Keyspaces, puede activar automáticamente el escalado automático para la capacidad de escritura o lectura de la tabla en la `CREATE TABLE` declaración. Esto permite a Amazon Keyspaces ponerse en contacto con Application Auto Scaling en su nombre para registrar la tabla como un objetivo escalable y ajustar la capacidad de escritura o lectura aprovisionada.

Para obtener más información sobre cómo crear una tabla de varias regiones y configurar diferentes ajustes de escalado automático para las réplicas de tablas, consulte. the section called "Crear una tabla multirregional con la configuración predeterminada (CQL)"

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a un servicio (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) para realizar acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

Para configurar los ajustes de escalado automático de una tabla mediante programación, utilice la `AUTOSCALING_SETTINGS` instrucción que contiene los parámetros del escalado automático de Amazon Keyspaces. Los parámetros definen las condiciones que obligan a Amazon Keyspaces a ajustar el rendimiento aprovisionado de la tabla y qué acciones opcionales adicionales se deben realizar. En este ejemplo, se definen los ajustes de escalado automático para mytable.

La política contiene los componentes siguientes:

- `AUTOSCALING_SETTINGS`— Especifica si Amazon Keyspaces puede ajustar la capacidad de procesamiento en su nombre. Se requieren los siguientes valores:
  - `provisioned_write_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `provisioned_read_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
- `scaling_policy`— Amazon Keyspaces es compatible con la política de seguimiento de objetivos. Para definir la política de seguimiento objetivo, debe configurar los siguientes parámetros.
  - `target_value`— El escalado automático de Amazon Keyspaces garantiza que la relación entre la capacidad consumida y la capacidad aprovisionada se mantenga en este valor o cerca de él. `target_value` se define como un porcentaje.
  - `disableScaleIn`: (Opcional) A `boolean` que especifica si `scale-in` está deshabilitada o habilitada para la tabla. Este parámetro está desactivado de forma predeterminada. Para activarlo`scale-in`, defina el `boolean` valor en`FALSE`. Esto significa que la capacidad de una mesa se reduce automáticamente en tu nombre.

- `scale_out_cooldown`: una actividad de escalado horizontal aumenta el rendimiento aprovisionado de su tabla. Para añadir un periodo de recuperación para las actividades de escalado horizontal, especifique un valor para `scale_out_cooldown` en segundos. Si no especificas ningún valor, el valor predeterminado es 0. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.
- `scale_in_cooldown`: una actividad de reducción horizontal disminuye el rendimiento aprovisionado de su tabla. Para añadir un periodo de recuperación para las actividades de reducción horizontal, especifique un valor para `scale_in_cooldown` en segundos. Si no especifica ningún valor, el valor predeterminado es 0. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.

> **Note**
>
> Para entender cómo funciona `target_value`, imagine que tiene una tabla con una configuración de rendimiento aprovisionado de 200 unidades de capacidad de escritura. Decide crear una política de escalado para esta tabla, con un valor de `target_value` del 70 %.
>
> Ahora, supongamos que comienza a dirigir el tráfico de escritura a la tabla, de tal forma que el rendimiento de escritura real es de 150 unidades de capacidad. La consumed-to-provisioned proporción es ahora (150/200), o sea, del 75 por ciento. Esta proporción supera su objetivo, por lo que el escalado automático aumenta la capacidad de escritura aprovisionada a 215, de modo que la proporción es (150/215), o un 69,77 por ciento, lo más cerca `target_value` posible de la suya, pero sin superarla.

En mytable, has establecido la capacidad de lectura y escritura en un 50 `TargetValue` por ciento. El escalado automático de Amazon Keyspaces ajusta el rendimiento aprovisionado de la tabla dentro del rango de 5 a 10 unidades de capacidad, de modo que la consumed-to-provisioned proporción se mantenga en el 50 por ciento o cerca de él. En cuanto a la capacidad de lectura, debe establecer los valores para y en 60 segundos. `ScaleOutCooldown` `ScaleInCooldown`

Puedes usar la siguiente declaración para crear una nueva tabla de Amazon Keyspaces con el escalado automático activado.

```
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
            }
        }
    }
};
```

Habilite el escalado automático en una tabla existente mediante CQL

En el caso de una tabla de Amazon Keyspaces existente, puede activar el escalado automático para la capacidad de escritura o lectura de la tabla mediante la `ALTER TABLE` instrucción. Si va a actualizar una tabla que actualmente está en modo de capacidad bajo demanda, `capacity_mode` es obligatorio. Si la tabla ya está en el modo de capacidad aprovisionada, puedes omitir este campo.

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a servicios (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realice acciones de escalado automático en su nombre. Este rol se crea automáticamente

para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

En el siguiente ejemplo, la instrucción actualiza la tabla mytable, que está en modo de capacidad bajo demanda. La declaración cambia el modo de capacidad de la tabla al modo aprovisionado con el escalado automático habilitado.

La capacidad de escritura se configura dentro del rango de 5 a 10 unidades de capacidad con un valor objetivo del 50%. La capacidad de lectura también se configura dentro del rango de 5 a 10 unidades de capacidad con un valor objetivo del 50%. En cuanto a la capacidad de lectura, se establecen los valores `scale_in_cooldown` para `scale_out_cooldown` y en 60 segundos.

```
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
            }
        }
    }
```

```
};
```

Vea la configuración de escalado automático de Amazon Keyspaces de su mesa mediante CQL

Para ver los detalles de la configuración de escalado automático de una tabla, utilice el siguiente comando.

```
SELECT * FROM system_schema_mcs.autoscaling WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

La salida de este comando tiene el siguiente aspecto.

```
 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}}}
```

Desactive el escalado automático de Amazon Keyspaces para una tabla mediante CQL

Puedes desactivar el escalado automático de Amazon Keyspaces para tu mesa en cualquier momento. Si ya no necesita escalar la capacidad de lectura o escritura de la tabla, debería considerar la posibilidad de desactivar el escalado automático para que Amazon Keyspaces no siga modificando la configuración de la capacidad de lectura o escritura de la tabla. Puede actualizar la tabla con una `ALTER TABLE` declaración.

La siguiente instrucción desactiva el escalado automático para la capacidad de escritura de la tabla mytable. También elimina las CloudWatch alarmas que se crearon en su nombre.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

> **Note**
>
> Para eliminar el rol vinculado al servicio que utiliza Application Auto Scaling, debe deshabilitar el escalado automático en todas las tablas de la cuenta. Regiones de AWS

Administración de las políticas de escalado de Amazon Keyspaces con la CLI

Para actualizar y administrar la configuración de escalado automático de Amazon Keyspaces mediante programación, puede usar AWS Command Line Interface (AWS CLI) o la API. AWS Para gestionar las políticas de escalado automático de Amazon Keyspaces mediante Cassandra Query Language (CQL), consulte. the section called “Uso de CQL” En este tema se ofrece una descripción general de las tareas de autoescalado que puede gestionar mediante programación mediante el. AWS CLI

Para obtener más información sobre los AWS CLI comandos de Amazon Keyspaces descritos en este tema, consulte la Referencia de AWS CLI comandos.

Temas

- Antes de empezar
- Cree una tabla nueva con escalado automático mediante el AWS CLI
- Habilite el escalado automático en una tabla existente mediante el AWS CLI
- Consulta la configuración de escalado automático de Amazon Keyspaces de tu mesa mediante el AWS CLI
- Desactive el escalado automático de Amazon Keyspaces para una tabla mediante el AWS CLI

Antes de empezar

Debe completar las siguientes tareas antes de empezar.

Configuración de permisos de

Si aún no lo ha hecho, debe configurar los permisos apropiados para que el usuario pueda crear y administrar los ajustes de escalado automático. En AWS Identity and Access Management (IAM), la política AWS gestionada `AmazonKeyspacesFullAccess` es necesaria para gestionar las políticas de escalado de Amazon Keyspaces. Para conocer los pasos en detalle, consulte the section called "Antes de comenzar: Concesión de permisos a los usuarios para el escalado automático de Amazon Keyspaces".

Instalar la AWS CLI

Si aún no lo ha hecho, debe instalar y configurar la AWS CLI. Para ello, consulte la Guía del AWS Command Line Interface usuario y siga estas instrucciones:

- Instalación de la AWS CLI
- Configuración de la AWS CLI

Cree una tabla nueva con escalado automático mediante el AWS CLI

Al crear una nueva tabla de Amazon Keyspaces, puede activar automáticamente el escalado automático para la capacidad de escritura o lectura de la tabla en la `CreateTable` operación. Esto permite a Amazon Keyspaces ponerse en contacto con Application Auto Scaling en su nombre para registrar la tabla que especifique como objetivo escalable y ajustar la capacidad de escritura o lectura aprovisionada.

Para obtener más información sobre cómo crear una tabla de varias regiones con una configuración de escalado automático, consultethe section called "Crear una nueva tabla multirregional en modo aprovisionado con escalado automático (CLI)".

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a un servicio (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) para realizar acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called "Uso de roles vinculados a servicios".

Para configurar los ajustes de escalado automático de una tabla mediante programación, utilice la `autoScalingSpecification` acción que define los parámetros del escalado automático de Amazon Keyspaces. Los parámetros definen las condiciones que obligan a Amazon Keyspaces a ajustar el rendimiento aprovisionado de la tabla y qué acciones opcionales adicionales se deben realizar. En este ejemplo, se definen los ajustes de escalado automático para mytable.

La política contiene los componentes siguientes:

- `autoScalingSpecification`— Especifica si Amazon Keyspaces puede ajustar la capacidad de procesamiento en su nombre. Puede activar el escalado automático para la capacidad de lectura y escritura por separado. A continuación, debe especificar los siguientes parámetros para`autoScalingSpecification`:
  - `writeCapacityAutoScaling`— Las unidades de capacidad de escritura máxima y mínima.
  - `readCapacityAutoScaling`— Las unidades de capacidad de lectura máxima y mínima.
  - `scalingPolicy`— Amazon Keyspaces es compatible con la política de seguimiento de objetivos. Para definir la política de seguimiento objetivo, debe configurar los siguientes parámetros.
    - `targetValue`— El escalado automático de Amazon Keyspaces garantiza que la relación entre la capacidad consumida y la capacidad aprovisionada se mantenga en este valor o cerca de él. `targetValue` se define como un porcentaje.
    - `disableScaleIn`: (Opcional) A `boolean` que especifica si `scale-in` está deshabilitada o habilitada para la tabla. Este parámetro está desactivado de forma predeterminada. Para activarlo`scale-in`, defina el `boolean` valor en`FALSE`. Esto significa que la capacidad de una mesa se reduce automáticamente en tu nombre.
    - `scaleOutCooldown`: una actividad de escalado horizontal aumenta el rendimiento aprovisionado de su tabla. Para añadir un periodo de recuperación para las actividades de escalado horizontal, especifique un valor para `ScaleOutCooldown` en segundos. El valor predeterminado es 0. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.
    - `scaleInCooldown`: una actividad de reducción horizontal disminuye el rendimiento aprovisionado de su tabla. Para añadir un periodo de recuperación para las actividades de reducción horizontal, especifique un valor para `ScaleInCooldown` en segundos. El valor predeterminado es 0. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.

> **Note**
>
> Para entender cómo funciona `TargetValue`, imagine que tiene una tabla con una configuración de rendimiento aprovisionado de 200 unidades de capacidad de escritura. Decide crear una política de escalado para esta tabla, con un valor de `TargetValue` del 70 %.
> Ahora, supongamos que comienza a dirigir el tráfico de escritura a la tabla, de tal forma que el rendimiento de escritura real es de 150 unidades de capacidad. La consumed-to-provisioned proporción es ahora (150/200), o sea, del 75 por ciento. Esta proporción supera su objetivo, por lo que el escalado automático aumenta la capacidad de escritura aprovisionada a 215, de modo que la proporción es (150/215), o un 69,77 por ciento, lo más cerca `TargetValue` posible de la suya, pero sin superarla.

En mytable, has establecido la capacidad de lectura y escritura en un 50 `TargetValue` por ciento. El escalado automático de Amazon Keyspaces ajusta el rendimiento aprovisionado de la tabla dentro del rango de 5 a 10 unidades de capacidad, de modo que la consumed-to-provisioned proporción se mantenga en el 50 por ciento o cerca de él. En cuanto a la capacidad de lectura, debe establecer los valores para y en 60 segundos. `ScaleOutCooldown` `ScaleInCooldown`

Al crear tablas con ajustes de autoescalado complejos, resulta útil cargar los ajustes de autoescalado desde un archivo JSON. Para el siguiente ejemplo, puede descargar el archivo JSON de ejemplo desde auto-scaling.zip y extraerlo`auto-scaling.json`, tomando nota de la ruta al archivo. En este ejemplo, el archivo JSON se encuentra en el directorio actual. Para conocer las distintas opciones de ruta del archivo, consulte Carga de parámetros desde un archivo.

```
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
```

Habilite el escalado automático en una tabla existente mediante el AWS CLI

Para una tabla Amazon Keyspaces existente, puede activar el escalado automático para la capacidad de escritura o lectura de la tabla mediante la `UpdateTable` operación. Para obtener más información sobre cómo actualizar la configuración de escalado automático de una tabla de varias

regiones, consultethe section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CLI)”.

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a servicios (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realice acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

Puedes usar el siguiente comando para activar el escalado automático de Amazon Keyspaces en una tabla existente. La configuración de escalado automático de la tabla se carga desde un archivo JSON. Para el siguiente ejemplo, puede descargar el archivo JSON de ejemplo desde auto-scaling.zip y extraerlo`auto-scaling.json`, tomando nota de la ruta al archivo. En este ejemplo, el archivo JSON se encuentra en el directorio actual. Para conocer las distintas opciones de ruta del archivo, consulte Carga de parámetros desde un archivo.

Para obtener más información sobre la configuración de escalado automático utilizada en el siguiente ejemplo, consultethe section called “Cree una tabla nueva con escalado automático mediante el AWS CLI”.

```
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
```

Consulta la configuración de escalado automático de Amazon Keyspaces de tu mesa mediante el AWS CLI

Para ver la configuración de escalado automático de una tabla, puede utilizar la `get-table-auto-scaling-settings` operación. El siguiente comando CLI es un ejemplo de ello.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name
 mytable
```

La salida de este comando tiene el siguiente aspecto.

```
{
    "keyspaceName": "mykeyspace",
    "tableName": "mytable",
    "resourceArn": "arn:aws:cassandra:us-east-1:5555-5555-5555:/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
                }
            }
        }
    }
}
```

Desactive el escalado automático de Amazon Keyspaces para una tabla mediante el AWS CLI

Puedes desactivar el escalado automático de Amazon Keyspaces para tu mesa en cualquier momento. Si ya no necesita escalar la capacidad de lectura o escritura de la tabla, debería considerar la posibilidad de desactivar el escalado automático para que Amazon Keyspaces no siga

modificando la configuración de la capacidad de lectura o escritura de la tabla. Puede actualizar la tabla con una `UpdateTable` operación.

El siguiente comando desactiva el escalado automático de la capacidad de lectura de la tabla. También elimina las CloudWatch alarmas que se crearon en su nombre.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
            \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

> **Note**
>
> Para eliminar el rol vinculado al servicio que utiliza Application Auto Scaling, debe deshabilitar el escalado automático en todas las tablas de la cuenta. Regiones de AWS

Uso eficaz de la capacidad de ráfaga en Amazon Keyspaces

Amazon Keyspaces proporciona cierta flexibilidad en su aprovisionamiento de rendimiento por partición al ofrecer capacidad de ampliación. Toda vez que no utilice por completo el rendimiento de una partición, Amazon Keyspaces reserva una parte de esa capacidad no utilizada para posteriores ampliaciones de rendimiento con el fin de gestionar los picos de uso.

En la actualidad, Amazon Keyspaces retiene hasta 5 minutos (300 segundos) de capacidad de lectura y escritura no utilizada. Durante un pico ocasional de actividad de lectura o escritura, estas unidades de capacidad adicionales pueden consumirse rápidamente, incluso más rápido que la capacidad de rendimiento aprovisionada por segundo que haya definido para su tabla.

Amazon Keyspaces también puede consumir capacidad de ampliación para mantenimiento en segundo plano y otras tareas sin previo aviso.

Tenga en cuenta que estos detalles sobre la capacidad de ampliación podrían cambiar en el futuro.

Cómo estimar el consumo de capacidad en Amazon Keyspaces

Al leer o escribir datos en Amazon Keyspaces, la cantidad de unidades de solicitud de lectura/escritura (RRUS/WRU) o unidades de capacidad de lectura/escritura (RCU/WCU) que consume la

consulta depende de la cantidad total de datos que Amazon Keyspaces debe procesar para ejecutar la consulta. En algunos casos, los datos devueltos al cliente pueden ser un subconjunto de los datos que Amazon Keyspaces tuvo que leer para procesar la consulta. En el caso de las escrituras condicionales, Amazon Keyspaces consume capacidad de escritura incluso si se produce un error en la comprobación condicional.

Para estimar la cantidad total de datos que se están procesando para una solicitud, debe tener en cuenta el tamaño codificado de una fila y el número total de filas. En este tema se describen algunos ejemplos de escenarios y patrones de acceso comunes para mostrar cómo Amazon Keyspaces procesa las consultas y cómo esto afecta al consumo de capacidad. Puedes seguir los ejemplos para estimar los requisitos de capacidad de tus tablas y utilizar Amazon CloudWatch para observar el consumo de capacidad de lectura y escritura en estos casos de uso.

Para obtener información sobre cómo calcular el tamaño codificado de las filas en Amazon Keyspaces, consulte. the section called “Cálculo del tamaño de las filas”

Temas

- Consultas de rango
- Escaneos de tablas
- Transacciones ligeras
- Calcula el consumo de capacidad de lectura y escritura con Amazon CloudWatch

Consultas de rango

Para analizar el consumo de capacidad de lectura de una consulta de rango, utilizamos la siguiente tabla de ejemplo, que utiliza el modo de capacidad bajo demanda.

```
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>
```

Ahora ejecute la siguiente consulta en esta tabla.

```
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;
```

Recibes el siguiente conjunto de resultados de la consulta y la operación de lectura realizada por Amazon Keyspaces consume 2 RRU en `LOCAL_QUORUM` modo de coherencia.

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-------
a | b | 1 | a | b | 60 | value_1
```

Amazon Keyspaces consume 2 RRU para evaluar las filas con los valores `ck3=60` y `ck3=70` procesar la consulta. Sin embargo, Amazon Keyspaces solo devuelve la fila en la que la `WHERE` condición especificada en la consulta es verdadera, que es la fila con valor. `ck3=60` Para evaluar el rango especificado en la consulta, Amazon Keyspaces lee la fila que coincide con el límite superior del rango, en este caso`ck3 = 70`, pero no devuelve esa fila en el resultado. El consumo de capacidad de lectura se basa en los datos leídos al procesar la consulta, no en los datos devueltos.

Escaneos de tablas

Las consultas que dan como resultado un escaneo completo de la tabla, por ejemplo, las consultas que utilizan la `ALLOW FILTERING` opción, son otro ejemplo de consultas que procesan más lecturas de las que devuelven como resultados. Además, el consumo de capacidad de lectura se basa en los datos leídos, no en los datos devueltos.

Para el ejemplo de escaneo de tablas, utilizamos la siguiente tabla de ejemplo en el modo de capacidad bajo demanda.

```
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 crea una tabla en el modo de capacidad bajo demanda con cuatro particiones de forma predeterminada. En esta tabla de ejemplo, todos los datos se almacenan en una partición y las tres particiones restantes están vacías.

Ahora ejecute la siguiente consulta en la tabla.

```
SELECT * from amazon_keyspaces.example_table_2;
```

Esta consulta da como resultado una operación de escaneo de tablas en la que Amazon Keyspaces escanea las cuatro particiones de la tabla y consume 6 RRU en `LOCAL_QUORUM` modo de

coherencia. En primer lugar, Amazon Keyspaces consume 3 RRU para leer las tres filas. `pk='pk'` A continuación, Amazon Keyspaces consume las 3 RRU adicionales para escanear las tres particiones vacías de la tabla. Como esta consulta da como resultado un escaneo de la tabla, Amazon Keyspaces escanea todas las particiones de la tabla, incluidas las particiones sin datos.

Transacciones ligeras

Las transacciones ligeras (LWT) te permiten realizar operaciones de escritura condicional con los datos de tu tabla. Las operaciones de actualización condicional son útiles al insertar, actualizar y eliminar registros en función de las condiciones que evalúan el estado actual.

En Amazon Keyspaces, todas las operaciones de escritura requieren coherencia con LOCAL_QUORUM y el uso de los LWT no conlleva ningún cargo adicional. La diferencia con los LWT es que cuando una comprobación del estado del LWT da como resultado FALSE, consume unidades de capacidad de escritura. El número de unidades de capacidad de escritura consumidas depende del tamaño de la fila. Si el tamaño de la fila es de 2 KB, la escritura condicional fallida consume dos unidades de capacidad de escritura. Si la fila no existe actualmente en la tabla, la operación consume una unidad de capacidad de escritura. Al monitorear la `ConditionalCheckFailed` métrica, CloudWatch puede determinar la capacidad consumida por los errores de verificación de estado del LWT.

Calcula el consumo de capacidad de lectura y escritura con Amazon CloudWatch

Para estimar y supervisar el consumo de capacidad de lectura y escritura, puede utilizar un CloudWatch panel de control. Para obtener más información sobre las métricas disponibles para Amazon Keyspaces, consulte. the section called “Métricas y dimensiones”

Para monitorear las unidades de capacidad de lectura y escritura consumidas por una declaración específica CloudWatch, puede seguir estos pasos.

1. Cree una tabla nueva con datos de ejemplo
2. Configure un CloudWatch panel de Amazon Keyspaces para la tabla. Para empezar, puedes usar una plantilla de panel disponible en Github.
3. Ejecute la instrucción CQL, por ejemplo, utilizando la `ALLOW FILTERING` opción, y compruebe en el panel de control las unidades de capacidad de lectura consumidas durante el escaneo completo de la tabla.

Uso de los espacios de claves, tablas y filas en Amazon Keyspaces (para Apache Cassandra)

En este capítulo se ofrecen detalles sobre el uso de espacios de claves, tablas, filas y mucho más en Amazon Keyspaces (para Apache Cassandra). Para obtener información sobre cómo supervisar los espacios clave y las tablas con Amazon CloudWatch, consultethe section called “Monitorización con CloudWatch”.

Temas

- Uso de los espacios de claves en Amazon Keyspaces
- Uso de las tablas en Amazon Keyspaces
- Uso de las filas en Amazon Keyspaces
- Uso de consultas en Amazon Keyspaces
- Uso de particionadores en Amazon Keyspaces
- Trabajar con etiquetas y rótulos para los recursos de Amazon Keyspaces

Uso de los espacios de claves en Amazon Keyspaces

En esta sección se proporcionan detalles sobre cómo trabajar con espacios de claves en Amazon Keyspaces (para Apache Cassandra).

Temas

- Uso de los espacios de claves de sistema en Amazon Keyspaces
- Creación de espacios de claves en Amazon Keyspaces

Uso de los espacios de claves de sistema en Amazon Keyspaces

Amazon Keyspaces utiliza cuatro espacios de claves de sistema:

- `system`
- `system_schema`
- `system_schema_mcs`
- `system_multiregion_info`

En las siguientes secciones se proporcionan detalles sobre los espacios de claves del sistema y las tablas del sistema que se admiten en Amazon Keyspaces.

`system`

Este es un espacio de claves de Cassandra. Amazon Keyspaces utiliza las siguientes tablas.

Nombres de tablas	Nombres de columnas	Comentarios
`local`	`key, bootstrapped, broadcast_address, cluster_name, cql_version, data_center, gossip_generation, host_id, listen_address, native_protocol_version, partitioner, rack, release_version, rpc_address, schema_version, thrift_version, tokens, truncated_at`	Información sobre el espacio de claves local.
`peers`	`peer, data_center, host_id, preferred_ip, rack, release_version, rpc_address, schema_version, tokens`	Consulte esta tabla para ver los puntos de conexión disponibles. Por ejemplo, si se conecta a través de un punto final público, verá una lista de nueve direcciones IP disponibles. Si te conectas a través de un punto final FIPS, verás una lista de tres direcciones IP. Si te conectas a través de un punto de conexión de AWS PrivateLink VPC, verás la lista de direcciones

Nombres de tablas	Nombres de columnas	Comentarios
		IP que has configurado. Para obtener más información, consulte the section called “Relleno de entradas de la tabla `system.peers` con información del punto de conexión de VPC de interfaz”.
`size_estimates`	`keyspace_name, table_name, range_start, range_end, mean_partition_size, partitions_count`	En esta tabla se definen el tamaño total y el número de particiones de cada rango de tokens para cada tabla. Esto es necesario para el conector Spark de Apache Cassandra, que utiliza el tamaño estimado de las particiones para distribuir el trabajo.
`prepared_statements`	`prepared_id, logged_keyspace, query_string`	Esta tabla contiene información sobre las consultas guardadas.

`system_schema`

Este es un espacio de claves de Cassandra. Amazon Keyspaces utiliza las siguientes tablas.

Nombres de tablas	Nombres de columnas	Comentarios
`keyspaces`	`keyspace_name, durable_writes, replication`	Información sobre un espacio de claves específico.
`tables`	`keyspace_name, table_name, bloom_fil`	Información sobre una tabla específica.

Nombres de tablas	Nombres de columnas	Comentarios
	`ter_fp_chance, caching, 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`	
`columns`	`keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type`	Información sobre una columna específica.

`system_schema_mcs`

Se trata de un espacio de claves de Amazon Keyspaces que almacena información o ajustes específicos de AWS Amazon Keyspaces.

Nombres de tablas	Nombres de columnas	Comentarios
`keyspaces`	`keyspace_name, durable_writes, replication`	Consulte esta tabla para averiguar mediante programación si se ha creado un espacio de claves. Para obtener más

Nombres de tablas	Nombres de columnas	Comentarios
		información, consulte the section called “Creación de espacios de claves”.
`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, status`	Consulte esta tabla para averiguar el estado de una tabla específica. Para obtener más información, consulte the section called “Creación de tablas”. También puedes consultar esta tabla para ver los ajustes que son específicos de Amazon Keyspaces y que se almacenan como. `custom_properties` Por ejemplo: • `capacity_mode` • `client_side_timestamps` • `encryption_specification` • `point_in_time_recovery` • `ttl`
`tables_history`	`keyspace_name, table_name, event_time, creation_time, custom_properties, event`	Consulte esta tabla para obtener información sobre los cambios de esquema para una tabla específica.

Nombres de tablas	Nombres de columnas	Comentarios
`columns`	`keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type`	Esta tabla es idéntica a la tabla de Cassandra en el espacio de claves `system_schema` .
`tags`	`resource_id, keyspace_name, resource_name, resource_type, tags`	Consulte esta tabla para averiguar si un espacio de claves tiene etiquetas. Para obtener más información, consulte the section called “Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante CQL”.
`autoscaling`	`keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update`	Consulte esta tabla para obtener la configuración de escalado automático de una tabla aprovisionada. Tenga en cuenta que estos ajustes no estarán disponibles hasta que la tabla esté activa. Para consultar esta tabla, debe especificar `keyspace_name` y `table_name` en la cláusula `WHERE`. Para obtener más información, consulte the section called “Uso de CQL”.

`system_multiregion_info`

Este es un espacio de claves de Amazon Keyspaces que almacena información sobre la réplica multirregión.

Nombres de tablas	Nombres de columnas	Comentarios
`tables`	`keyspace_name, table_name, region, status`	Esta tabla contiene información sobre las tablas de varias regiones, por ejemplo, en qué se replica la tabla y su estado. Regiones de AWS También puedes consultar esta tabla para ver una lista de los ajustes específicos de Amazon Keyspaces como los que están almacenados. `custom_properties` Por ejemplo: • `capacity_mode` Para consultar esta tabla, debe especificar `keyspace_name` y `table_name` en la cláusula `WHERE`. Para obtener más información, consulte the section called “Creación de un espacio de claves multirregión (CQL)”.
`autoscaling`	`keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update, region`	Consulte esta tabla para obtener la configuración de escalado automático de una tabla aprovisionada en varias regiones. Tenga en cuenta que estos ajustes no estarán disponibles hasta que la tabla esté activa. Para consultar esta tabla, debe especificar `keyspace_name` y

Nombres de tablas	Nombres de columnas	Comentarios
		`table_name` en la cláusula `WHERE`. Para obtener más información, consulte the section called “Uso de CQL”.

Creación de espacios de claves en Amazon Keyspaces

Amazon Keyspaces realiza las operaciones de lenguaje de definición de datos (DDL), como crear y eliminar espacios de claves, de forma asíncrona.

Puede supervisar el estado de creación de los nuevos espacios clave en la AWS Management Console, lo que indica si un espacio clave está pendiente o activo. También puede monitorear el estado de creación de un nuevo espacio de claves mediante programación utilizando el espacio de claves `system_schema_mcs`. Un espacio de teclas se hace visible en la `system_schema_mcs keyspaces` tabla cuando está listo para su uso.

El patrón de diseño recomendado para comprobar cuándo un nuevo espacio de claves está listo para su uso es sondear las tablas `system_schema_mcs keyspaces` de Amazon Keyspaces (system_schema_mcs.*). Para obtener una lista de las instrucciones DDL para espacios de claves, consulte la sección the section called “Keyspaces” en la referencia del lenguaje CQL.

La siguiente consulta muestra si un espacio de claves se ha creado correctamente.

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

En el caso de un espacio de claves que se haya creado correctamente, el resultado de la consulta es el siguiente.

```
keyspace_name | durable_writes  | replication
--------------+-----------------+--------------
   mykeyspace | true            |{...} 1 item
```

Uso de las tablas en Amazon Keyspaces

En esta sección se ofrecen detalles sobre el uso de las tablas en Amazon Keyspaces (para Apache Cassandra).

Temas

- Creación de tablas en Amazon Keyspaces
- Trabajar con tablas de varias regiones en Amazon Keyspaces
- Columnas estáticas en Amazon Keyspaces

Creación de tablas en Amazon Keyspaces

Amazon Keyspaces realiza operaciones de lenguaje de definición de datos (DDL), como crear y eliminar tablas, de forma asíncrona. Puede supervisar el estado de creación de nuevas tablas en AWS Management Console, lo que indica cuándo una tabla está pendiente o activa. También puede monitorear el estado de creación de una nueva tabla mediante programación utilizando la tabla de esquema del sistema.

Una tabla aparece como activa en el esquema del sistema cuando está lista para su uso. El patrón de diseño recomendado para comprobar cuándo una nueva tabla está lista para su uso es sondear las tablas de esquema del sistema de Amazon Keyspaces (`system_schema_mcs.*`). Para obtener una lista de instrucciones DDL para tablas, consulte la sección the section called “Tablas” en la referencia del lenguaje CQL.

La siguiente consulta muestra el estado de una tabla.

```
SELECT keyspace_name, table_name, status FROM system_schema_mcs.tables WHERE
 keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Para una tabla que todavía se esté creando y esté pendiente, la salida de la consulta tiene este aspecto.

```
keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | CREATING
```

Para una tabla que se haya creado correctamente y esté activa, la salida de la consulta tiene el siguiente aspecto.

```
keyspace_name | table_name | status
--------------+------------+--------
   mykeyspace |    mytable | ACTIVE
```

Trabajar con tablas de varias regiones en Amazon Keyspaces

Una tabla multirregional debe tener la capacidad de rendimiento de escritura configurada de dos maneras:

- Modo de capacidad bajo demanda, medido en unidades de solicitud de escritura (WRU)
- Modo de capacidad aprovisionada con escalado automático, medido en unidades de capacidad de escritura (WCU)

Puede usar el modo de capacidad aprovisionada con escalado automático o el modo de capacidad bajo demanda para garantizar que una tabla multirregional tenga la capacidad suficiente para realizar escrituras replicadas en todas ellas. Regiones de AWS

> **Note**
>
> Al cambiar el modo de capacidad de la tabla en una de las regiones, se cambia el modo de capacidad de todas las réplicas.

De forma predeterminada, Amazon Keyspaces utiliza el modo bajo demanda para las tablas de varias regiones. Con el modo bajo demanda, no necesita especificar el rendimiento de lectura y escritura que espera que alcance su aplicación. Amazon Keyspaces se adapta instantáneamente a sus cargas de trabajo a medida que aumentan o disminuyen hasta alcanzar cualquier nivel de tráfico alcanzado anteriormente. Si el nivel de tráfico de una carga de trabajo alcanza un nuevo pico, Amazon Keyspaces se adapta rápidamente para adaptarse a la carga de trabajo.

Si elige el modo de capacidad aprovisionada para una tabla, debe configurar el número de unidades de capacidad de lectura (RCU) y unidades de capacidad de escritura (WCU) por segundo que necesita su aplicación.

Para planificar las necesidades de capacidad de rendimiento de una tabla multirregional, primero debe estimar la cantidad de WCU por segundo necesaria para cada región. A continuación, añada las escrituras de todas las regiones en las que está replicada la tabla y utilice la suma para aprovisionar la capacidad de cada región. Esto es necesario porque cada escritura que se realiza en una región también debe repetirse en cada región de réplica.

Si la tabla no tiene la capacidad suficiente para gestionar las escrituras de todas las regiones, se producirán excepciones de capacidad. Además, aumentarán los tiempos de espera para la replicación interregional.

Por ejemplo, si tiene una tabla multirregional en la que espera 5 escrituras por segundo en EE.UU. Este (Norte de Virginia), 10 escrituras por segundo en EE.UU. Este (Ohio) y 5 escrituras por segundo en Europa (Irlanda), debería esperar que la tabla consuma 20 WCU en cada región: EE.UU. Este (Norte de Virginia), EE.UU. Este (Ohio) y Europa (Irlanda). Esto significa que, en este ejemplo, debe aprovisionar 20 WCU para cada una de las réplicas de la tabla. Puedes monitorizar el consumo de capacidad de tu mesa a través de Amazon CloudWatch. Para obtener más información, consulte the section called “Monitorización con CloudWatch”.

Como cada escritura multirregional se factura 1,25 veces las WCU, en este ejemplo se facturarán un total de 75 WCU. Para obtener más información sobre precios, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Para obtener más información sobre la capacidad aprovisionada con el the section called “Administre la capacidad de rendimiento con el escalado automático” escalado automático de Amazon Keyspaces, consulte.

> **Note**
>
> Si una tabla se ejecuta en modo de capacidad aprovisionada con escalado automático, se permite que la capacidad de escritura aprovisionada flote dentro de esa configuración de escalado automático para cada región.

Columnas estáticas en Amazon Keyspaces

En una tabla de Amazon Keyspaces con columnas de agrupación, puede utilizar la palabra clave `STATIC` para crear una columna estática. El valor almacenado en una columna estática se comparte entre todas las filas de una partición lógica. Al actualizar el valor de esta columna, Amazon Keyspaces aplica el cambio de forma automática a todas las filas de la partición.

Esta sección describe cómo calcular el tamaño codificado de datos al escribir en columnas estáticas. Este proceso se gestiona por separado del proceso que escribe datos en las columnas no estáticas de una fila. Además de las cuotas de tamaño para datos estáticos, las operaciones de lectura y escritura en columnas estáticas también afectan a la medición y a la capacidad de rendimiento de las tablas de forma independiente.

Cálculo de tamaño de columna estática por partición lógica en Amazon Keyspaces

En esta sección se proporciona información detallada sobre cómo estimar el tamaño codificado de columnas estáticas en Amazon Keyspaces. El tamaño codificado se utiliza para el cálculo de su factura y uso de cuotas. También debe utilizar el tamaño codificado al calcular los requisitos de capacidad de rendimiento aprovisionada para las tablas. Para calcular el tamaño codificado de columnas estáticas en Amazon Keyspaces, puede utilizar las siguientes pautas.

- Las claves de partición pueden contener hasta 2048 bytes de datos. Cada columna de clave en la clave de partición requiere hasta 3 bytes de metadatos. Estos bytes de metadatos cuentan para su cuota de tamaño de datos estáticos de 1 MB por partición. Al calcular el tamaño de sus datos estáticos, debe suponer que cada columna de clave de partición utiliza los 3 bytes completos de metadatos.
- Utilice el tamaño bruto de los valores de los datos de las columnas estáticas en función del tipo de datos. Para obtener más información sobre los tipos de datos, consulte the section called “Tipos de datos”.
- Añada 104 bytes al tamaño de los datos estáticos para los metadatos.
- Las columnas de agrupación y las columnas de claves normales no primarias no cuentan para el tamaño de los datos estáticos. Para obtener información sobre cómo calcular el tamaño de los datos no estáticos dentro de las filas, consulte the section called “Cálculo del tamaño de las filas”.

El tamaño codificado total de una columna estática se basa en la siguiente fórmula:

```
partition key columns + static columns + metadata = total encoded size of static data
```

Considere el siguiente ejemplo de una tabla en la que todas las columnas son de tipo entero. La tabla tiene dos columnas de clave de partición, dos columnas de agrupación, una columna normal y una columna estática.

```
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));
```

En este ejemplo, calculamos el tamaño de los datos estáticos con la siguiente instrucción:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, static_col1) values(1,2,6);
```

Para estimar el total de bytes requeridos por esta operación de escritura, puede seguir los siguientes pasos.

1. Calcule el tamaño de una columna de clave de partición sumando los bytes correspondientes al tipo de datos almacenados en la columna y los bytes de metadatos. Repita este procedimiento para todas las columnas de clave de partición.

   a. Calcule el tamaño de la primera columna de clave de partición (pk_col1):

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   b. Calcule el tamaño de la segunda columna de clave de partición (pk_col2):

   ```
   4 bytes for the integer data type + 3 bytes for partition key metadata = 7
    bytes
   ```

   c. Sume ambas columnas para obtener el tamaño total estimado de las columnas de clave de partición:

   ```
   7 bytes + 7 bytes = 14 bytes for the partition key columns
   ```

2. Añada el tamaño de las columnas estáticas. En este ejemplo, solo tenemos una columna estática que almacena un número entero (que requiere 4 bytes).
3. Por último, para obtener el tamaño codificado total de datos de la columna estática, sume los bytes de las columnas de clave principal y las columnas estáticas, y añada los 104 bytes adicionales de metadatos:

   ```
   14 bytes for the partition key columns + 4 bytes for the static column + 104 bytes
    for metadata = 122 bytes.
   ```

También puede actualizar datos estáticos y no estáticos con la misma instrucción. Para estimar el tamaño total de la operación de escritura, primero debe calcular el tamaño de la actualización de datos no estáticos. A continuación, calcule el tamaño de la actualización de filas, como se muestra en el ejemplo en the section called “Cálculo del tamaño de las filas”, y sume los resultados.

En este caso, puede escribir un total de 2 MB (1 MB es la cuota máxima de tamaño de fila y 1 MB es la cuota de tamaño máximo de datos estáticos por partición lógica).

Para calcular el tamaño total de una actualización de datos estáticos y no estáticos en la misma instrucción, puede utilizar la siguiente fórmula:

```
(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
```

Considere el siguiente ejemplo de una tabla en la que todas las columnas son de tipo entero. La tabla tiene dos columnas de clave de partición, dos columnas de agrupación, una columna normal y una columna estática.

```
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));
```

En este ejemplo, calculamos el tamaño de los datos al escribir una fila en la tabla, como se muestra en la siguiente instrucción:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1,
 static_col1) values(2,3,4,5,6,7);
```

Para estimar el total de bytes requeridos por esta operación de escritura, puede seguir los siguientes pasos.

1. Calcule el tamaño codificado total de datos estáticos como se ha indicado anteriormente. En este ejemplo, es de 122 bytes.
2. Añada el tamaño del tamaño codificado total de la fila en función de la actualización de los datos no estáticos, siguiendo los pasos indicados en the section called “Cálculo del tamaño de las filas”. En este ejemplo, el tamaño total de la actualización de la fila es de 134 bytes.

```
122 bytes for static data + 134 bytes for nonstatic data = 256 bytes.
```

Medición de las operaciones de lectura/escritura de datos estáticos en Amazon Keyspaces

En Cassandra, los datos estáticos se asocian a particiones lógicas, no a filas individuales. Las particiones lógicas en Amazon Keyspaces pueden tener un tamaño prácticamente ilimitado al abarcar varias particiones de almacenamiento físico. Como resultado, Amazon Keyspaces mide las operaciones de escritura en datos estáticos y no estáticos por separado. Además, las escrituras que incluyan datos estáticos y no estáticos requieren operaciones subyacentes adicionales para proporcionar coherencia de datos.

Al realizar una operación de escritura mixta de datos estáticos y no estáticos, se producen dos operaciones de escritura separadas: una para datos no estáticos y otra para datos estáticos. Esto se aplica a ambos modos de capacidad de lectura/escritura, bajo demanda y aprovisionada.

En el siguiente ejemplo se proporciona información detallada sobre cómo estimar las unidades de capacidad de lectura (RCU) y las unidades de capacidad de escritura (WCU) necesarias al calcular los requisitos de capacidad de rendimiento aprovisionada para tablas en Amazon Keyspaces que tengan columnas estáticas. Puede calcular la capacidad que necesita su tabla para procesar escrituras que incluyan datos estáticos y no estáticos mediante la siguiente fórmula:

```
2 x WCUs required for nonstatic data + 2 x WCUs required for static data
```

Por ejemplo, si su aplicación escribe 27 KB de datos por segundo y cada escritura incluye 25,5 KB de datos no estáticos y 1,5 KB de datos estáticos, su tabla necesita 56 WCU (2 x 26 WCU + 2 x 2 WCU).

Amazon Keyspaces mide las lecturas de datos estáticos y no estáticos igual que las lecturas de múltiples filas. Como resultado, el precio de la lectura de datos estáticos y no estáticos en la misma operación se basa en el tamaño agregado de los datos procesados para realizar la lectura.

Para obtener información sobre cómo supervisar los recursos sin servidor con Amazon CloudWatch, consultethe section called “Monitorización con CloudWatch”.

Uso de las filas en Amazon Keyspaces

En esta sección se ofrecen detalles sobre el uso de las filas en Amazon Keyspaces (para Apache Cassandra). Las tablas son las estructuras de datos principales en Amazon Keyspaces y los datos de las tablas se organizan en columnas y filas.

Temas

- Cálculo del tamaño de las filas en Amazon Keyspaces

Cálculo del tamaño de las filas en Amazon Keyspaces

Amazon Keyspaces proporciona un almacenamiento totalmente administrado que ofrece un rendimiento de lectura y escritura de un dígito de milisegundo y almacena datos de forma duradera en múltiples zonas de disponibilidad de AWS. Amazon Keyspaces vincula metadatos a todas las filas y columnas de clave principal para admitir un acceso eficaz a los datos y una alta disponibilidad.

En esta sección se ofrece información detallada sobre cómo calcular el tamaño codificado de fila en Amazon Keyspaces. El tamaño codificado de fila se utiliza al calcular su factura y el uso de cuotas. También debe utilizar el tamaño codificado de fila al calcular los requisitos de capacidad de rendimiento aprovisionada para las tablas. Para calcular el tamaño codificado de fila en Amazon Keyspaces, puede utilizar las siguientes pautas.

- Para las columnas normales, que son aquellas que no son claves principales, columnas de agrupación ni columnas `STATIC`, utilice el tamaño bruto de los datos de las celdas en función del tipo de datos y añada los metadatos necesarios. Para obtener más información sobre los tipos de datos admitidos en Amazon Keyspaces, consulte the section called “Tipos de datos”. A continuación se enumeran algunas diferencias clave en la forma en que Amazon Keyspaces almacena los valores de los tipos de datos y los metadatos.
- El espacio necesario para cada nombre de columna se almacena utilizando un identificador de columna y se añade a cada valor de datos almacenado en la columna. El valor de almacenamiento del identificador de columna depende del número total de columnas de su tabla:
  - De 1 a 62 columnas: 1 byte
  - De 63 a 124 columnas: 2 bytes
  - De 125 a 186 columnas: 3 bytes

  Por cada 62 columnas adicionales, añada 1 byte. Tenga en cuenta que en Amazon Keyspaces se pueden modificar hasta 225 columnas regulares con una única instrucción `INSERT` o `UPDATE`.

Para obtener más información, consulte the section called “Cuotas de servicio de Amazon Keyspaces”.

- Las claves de partición pueden contener hasta 2048 bytes de datos. Cada columna de clave en la clave de partición requiere hasta 3 bytes de metadatos. Al calcular el tamaño de su fila, debe asumir que cada columna de clave de partición utiliza los 3 bytes completos de metadatos.
- Las columnas de agrupación pueden almacenar hasta 850 bytes de datos. Además del tamaño del valor de los datos, cada columna de agrupación requiere hasta un 20 % del tamaño del valor de los datos para metadatos. Al calcular el tamaño de su fila, debe añadir 1 byte de metadatos por cada 5 bytes de valor de datos de la columna de agrupación.
- Amazon Keyspaces almacena dos veces el valor de los datos de cada columna de clave de partición y clave de agrupación. La sobrecarga adicional se utiliza para realizar consultas eficientes y para la indexación integrada.
- Los tipos de datos de cadena `ASCII`, `TEXT` y `VARCHAR` de Cassandra se almacenan todos en Amazon Keyspaces utilizando Unicode con codificación binaria UTF-8. El tamaño de una cadena en Amazon Keyspaces es igual al número de bytes codificados en UTF-8.
- Los tipos de datos `INT`, `BIGINT`, `SMALLINT` y `TINYINT` de Cassandra se almacenan en Amazon Keyspaces como valores de datos de longitud variable, con hasta 38 dígitos significativos. Los ceros iniciales y finales se recortan. El tamaño de cualquiera de estos tipos de datos es de aproximadamente 1 byte por cada dos dígitos significativos + 1 byte.
- Un valor `BLOB` en Amazon Keyspaces se almacena con la longitud bruta en bytes del valor.
- El tamaño de un valor `Null` o un valor `Boolean` es de 1 byte.
- Una columna que almacena tipos de datos de colección como `LIST` o `MAP` requiere 3 bytes de metadatos, independientemente de su contenido. El tamaño de una `LIST` o `MAP` es (id de columna) + suma (tamaño de los elementos anidados) + (3 bytes). El tamaño de una `LIST` o `MAP` vacíos es (id de columna) + (3 bytes). Cada elemento `LIST` o `MAP` individual también requiere 1 byte de metadatos.
- Los datos de las columnas `STATIC` no cuentan para el tamaño máximo de fila de 1 MB. Para calcular el tamaño de los datos de las columnas estáticas, consulte the section called “Cálculo de tamaño de columna estática por partición lógica”.
- Las marcas de tiempo del lado del cliente se almacenan para cada columna de cada fila cuando la característica está activada. Estas marcas de tiempo ocupan aproximadamente entre 20 y 40 bytes (dependiendo de sus datos) y contribuyen al costo de almacenamiento y rendimiento de la fila. Para obtener más información, consulte the section called “Marcas de tiempo del lado del cliente en Amazon Keyspaces”.

- Añada 100 bytes al tamaño de cada fila para metadatos de la fila.

El tamaño total de una fila de datos codificada se basa en la siguiente fórmula:

```
partition key columns + clustering columns + regular columns + row metadata = total
 encoded size of row
```

> **Important**
>
> Todos los metadatos de columna, por ejemplo los id de columna, los metadatos de clave de partición, los metadatos de columna de agrupación, así como las marcas de tiempo del lado del cliente y los metadatos de fila cuentan para el tamaño máximo de fila de 1 MB.

Considere el siguiente ejemplo de una tabla en la que todas las columnas son de tipo entero. La tabla tiene dos columnas de clave de partición, dos columnas de agrupación y una columna normal. Dado que esta tabla tiene cinco columnas, el espacio necesario para el identificador de nombre de columna es de 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));
```

En este ejemplo, calculamos el tamaño de los datos cuando escribimos una fila en la tabla como se muestra en la siguiente instrucción:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1)
 values(1,2,3,4,5);
```

Para estimar el total de bytes requeridos por esta operación de escritura, puede seguir los siguientes pasos.

1. Calcule el tamaño de una columna de clave de partición sumando los bytes correspondientes al tipo de datos almacenados en la columna y los bytes de metadatos. Repita este procedimiento para todas las columnas de clave de partición.

   a. Calcule el tamaño de la primera columna de clave de partición (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. Calcule el tamaño de la segunda columna de clave de partición (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. Sume ambas columnas para obtener el tamaño total estimado de las columnas de clave de partición:

```
8 bytes + 8 bytes = 16 bytes for the partition key columns
```

2. Calcule el tamaño de la columna de agrupación sumando los bytes correspondientes al tipo de datos almacenados en la columna y los bytes de metadatos. Repita este procedimiento para todas las columnas de agrupación.

a. Calcule el tamaño de la primera columna de la columna de agrupación (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. Calcule el tamaño de la segunda columna de la columna de agrupación (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. Sume ambas columnas para obtener el tamaño total estimado de las columnas de agrupación:

```
6 bytes + 6 bytes = 12 bytes for the clustering columns
```

3. Añada el tamaño de las columnas normales. En este ejemplo solo tenemos una columna que almacena un número entero de un dígito, lo que requiere 2 bytes, con 1 byte para el id de columna.

4. Por último, para obtener el tamaño total de la fila codificada, sume los bytes de todas las columnas y añada los 100 bytes adicionales para los metadatos de la fila:

```
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.
```

Para obtener información sobre cómo monitorear los recursos sin servidor con Amazon CloudWatch, consulte the section called “Monitorización con CloudWatch”.

Uso de consultas en Amazon Keyspaces

En esta sección se ofrece una introducción al uso de las consultas en Amazon Keyspaces (para Apache Cassandra). Las instrucciones CQL disponibles para consultar, transformar y administrar datos son `SELECT`, `INSERT`, `UPDATE` y `DELETE`. En los siguientes temas se describen algunas de las opciones más complejas disponibles al utilizar consultas. Para ver la sintaxis completa del lenguaje con ejemplos, consulte the section called “instrucciones DML”.

Temas

- Uso del operador IN con la instrucción SELECT en Amazon Keyspaces
- Ordenación de resultados en Amazon Keyspaces
- Paginación de resultados en Amazon Keyspaces

Uso del operador **`IN`** con la instrucción **`SELECT`** en Amazon Keyspaces

SELECT IN

Puede consultar datos de tablas mediante la instrucción `SELECT`, que lee una o varias columnas de una o varias filas de una tabla y devuelve un conjunto de resultados que contiene las filas que coinciden con la solicitud. Una instrucción `SELECT` contiene una cláusula `select_clause` que determina qué columnas leer y devolver en el conjunto de resultados. La cláusula puede contener instrucciones para transformar los datos antes de devolverlos. La cláusula `WHERE` opcional especifica qué filas deben consultarse y se compone de relaciones sobre las columnas que forman parte de la clave principal. Amazon Keyspaces admite la palabra clave `IN` en la cláusula `WHERE`. En esta sección se utilizan ejemplos para mostrar cómo procesa Amazon Keyspaces las instrucciones `SELECT` con la palabra clave `IN`.

Estos ejemplos demuestran cómo Amazon Keyspaces descompone la instrucción `SELECT` con la palabra clave `IN` en subconsultas. En este ejemplo, utilizamos una tabla con el nombre `my_keyspace.customers`. La tabla tiene una columna de clave principal `department_id`, dos

columnas de agrupación `sales_region_id` y `sales_representative_id`, y una columna que contiene el nombre del cliente en la columna `customer_name`.

```
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
```

Con esta tabla, puede ejecutar la instrucción `SELECT` siguiente para encontrar los clientes en los departamentos y regiones de ventas que le interesen con la palabra clave `IN` en la cláusula `WHERE`. La siguiente instrucción es un ejemplo de ello.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1) AND sales_region_id
 IN (0, 1);
```

Amazon Keyspaces divide esta instrucción en cuatro subconsultas como se muestra en el siguiente resultado.

```
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
```

Al utilizar la palabra clave `IN`, Amazon Keyspaces pagina automáticamente los resultados en cualquiera de los siguientes casos:

- Después de procesarse cada 10.ª subconsulta.
- Después de procesar 1 MB de E/S lógica.
- Si configuró un `PAGE SIZE`, Amazon Keyspaces pagina después de leer el número de consultas por procesar en función del ajuste `PAGE SIZE`.
- Al utilizar la palabra clave `LIMIT` para reducir el número de filas devueltas, Amazon Keyspaces pagina después de leer el número de consultas por procesar en función del ajuste `LIMIT`.

La siguiente tabla sirve para ilustrar esto con un ejemplo.

Para obtener más información sobre paginación, consulte the section called “Paginación de resultados”.

```
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
```

Puede ejecutar la siguiente instrucción en esta tabla para ver cómo funciona la paginación.

```
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 procesa esta instrucción como 24 subconsultas, porque la cardinalidad del producto cartesiano de todos los términos `IN` contenidos en esta consulta es 24.

```
 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
```

En este ejemplo se muestra cómo puede utilizar la cláusula `ORDER BY` en una instrucción `SELECT` con la palabra clave `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
```

Las subconsultas se procesan en el orden en que las columnas de clave de partición y clave de agrupación se presentan en la consulta. En el ejemplo siguiente, las subconsultas para el valor de clave de partición “2” se procesan primero, seguidas de las subconsultas para el valor de clave de partición “3” y “1”. Los resultados de una subconsulta dada se ordenan según la cláusula de ordenación de la consulta, si está presente, o el orden de agrupación de la tabla definido durante la creación de la tabla.

```
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
```

Ordenación de resultados en Amazon Keyspaces

La cláusula `ORDER BY` especifica el orden de clasificación de los resultados devueltos en una instrucción `SELECT`. La instrucción toma una lista de nombres de columnas como argumentos y para cada columna puede especificar el orden de clasificación de los datos. Solo puede especificar columnas de agrupación en las cláusulas de ordenación, las columnas de no agrupación no están permitidas.

Las dos opciones de ordenación disponibles para los resultados devueltos son `ASC` para ordenación ascendente y `DESC` para ordenación descendente.

```
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
```

Si no especifica el orden de clasificación en la instrucción de consulta, se utiliza el orden predeterminado de la columna de agrupación.

Los posibles órdenes de clasificación que puede utilizar en una cláusula de ordenación dependen del orden de clasificación asignado a cada columna de agrupación en la creación de la tabla. Los resultados de la consulta solo pueden ordenarse en el orden definido para todas las columnas de agrupación en la creación de la tabla o a la inversa del orden de ordenación definido. No se permiten otras combinaciones posibles.

Por ejemplo, si el `CLUSTERING ORDER` de la tabla es (col1 ASC, col2 DESC, col3 ASC), los parámetros válidos para `ORDER BY` son (col1 ASC, col2 DESC, col3 ASC) o (col1 DESC, col2 ASC, col3 DESC). Para obtener más información sobre `CLUSTERING ORDER`, consulte `table_options` en the section called “CREATE TABLE”.

Paginación de resultados en Amazon Keyspaces

Amazon Keyspaces pagina automáticamente los resultados de las instrucciones `SELECT` cuando los datos leídos para procesar la instrucción `SELECT` superan 1 MB. Con la paginación, los resultados de las instrucciones `SELECT` se dividen en “páginas” de datos que tienen un tamaño de 1 MB (o

menos). Una aplicación puede procesar la primera página de resultados, luego la segunda página, y así sucesivamente. Los clientes siempre deben comprobar si hay tokens de paginación al procesar consultas `SELECT` que devuelvan varias filas.

Si un cliente suministra un `PAGE SIZE` que requiere la lectura de más de 1 MB de datos, Amazon Keyspaces divide los resultados automáticamente en varias páginas en función de los incrementos de lectura de datos de 1 MB.

Por ejemplo, si el tamaño medio de una fila es de 100 KB y se especifica un `PAGE SIZE` de 20, Amazon Keyspaces paginará los datos automáticamente después de leer 10 filas (1000 KB de datos leídos).

Dado que Amazon Keyspaces pagina los resultados en función del número de filas que lee para procesar una solicitud y no del número de filas devueltas en el conjunto de resultados, es posible que algunas páginas no contengan ninguna fila si está ejecutando consultas filtradas.

Por ejemplo, si establece `PAGE SIZE` en 10 y Keyspaces evalúa 30 filas para procesar su consulta `SELECT`, Amazon Keyspaces devolverá tres páginas. Si solo un subconjunto de las filas coincide con su consulta, algunas páginas podrían tener menos de 10 filas.

Uso de particionadores en Amazon Keyspaces

En Apache Cassandra, los particionadores controlan en qué nodos se almacenan los datos en el clúster. Los particionadores crean un token numérico mediante un valor hash de la clave de partición. Cassandra utiliza este token para distribuir los datos entre los nodos. Los clientes también pueden utilizar estos tokens en operaciones `SELECT` y cláusulas `WHERE` para optimizar las operaciones de lectura y escritura. Por ejemplo, los clientes pueden realizar eficientemente consultas paralelas en tablas grandes especificando distintos rangos de tokens para consultar en cada trabajo paralelo.

Amazon Keyspaces proporciona tres particionadores diferentes.

Murmur3Partitioner (predeterminado)

`Murmur3Partitioner` compatible con Apache Cassandra. `Murmur3Partitioner` es el particionador predeterminado de Cassandra en Amazon Keyspaces y en Cassandra 1.2 y versiones posteriores.

RandomPartitioner

`RandomPartitioner` compatible con Apache Cassandra. `RandomPartitioner` es el particionador predeterminado de Cassandra para las versiones anteriores a Cassandra 1.2.

Particionador predeterminado de Keyspaces

El `DefaultPartitioner` devuelve los mismos resultados de la función `token` que el `RandomPartitioner`.

El ajuste del particionador se aplica por región a nivel de cuenta. Por ejemplo, si cambia el particionador en Este de EE. UU. (Norte de Virginia), el cambio se aplica a todas las tablas de la misma cuenta en esta región. Puede cambiar el particionador de forma segura en cualquier momento. Tenga en cuenta que el cambio de configuración tarda aproximadamente 10 minutos en completarse. No es necesario que vuelva a cargar los datos de Amazon Keyspaces al cambiar la configuración del particionador. Los clientes utilizan automáticamente la nueva configuración del particionador la próxima vez que se conecten.

Puede cambiar el particionador con la AWS Management Console o con Cassandra Query Language (CQL).

AWS Management Console

Para cambiar el particionador con la consola de Amazon Keyspaces

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Configuration (Configuración).
3. En la página Configuración, vaya a Editar particionador.
4. Selecciona el particionador compatible con su versión de Cassandra. El cambio de particionador tarda aproximadamente 10 minutos en aplicarse.

> **Note**
>
> Una vez completado el cambio de configuración, debe desconectarse y volver a conectarse a Amazon Keyspaces para que las solicitudes utilicen el nuevo particionador.

Cassandra Query Language (CQL)

1. Para ver qué particionador está configurado para la cuenta, puede utilizar la siguiente consulta.

```
SELECT partitioner from system.local;
```

Si no se ha cambiado el particionador, la consulta tiene la siguiente salida.

```
partitioner
-----------------------------------------
com.amazonaws.cassandra.DefaultPartitioner
```

2. Para actualizar el particionador al particionador `Murmur3`, puede utilizar la siguiente instrucción.

```
UPDATE system.local set
 partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

3. Tenga en cuenta que este cambio de configuración tarda aproximadamente 10 minutos en completarse. Para confirmar que se haya configurado el particionador, puede ejecutar de nuevo la consulta `SELECT`. Tenga en cuenta que, debido a la posible coherencia de lectura, es posible que la respuesta aún no refleje los resultados del cambio de particionador recientemente completado. Si repite la operación `SELECT` tras un breve lapso de tiempo, la respuesta debería devolver los datos más recientes.

```
SELECT partitioner from system.local;
```

> **Note**
>
> Debe desconectarse y volver a conectarse a Amazon Keyspaces para que las solicitudes utilicen el nuevo particionador.

Trabajar con etiquetas y rótulos para los recursos de Amazon Keyspaces

Puede rotular los recursos de Amazon Keyspaces (para Apache Cassandra) mediante etiquetas. Las etiquetas le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario, entorno u otros criterios. Las etiquetas pueden ayudar a hacer lo siguiente:

- Identificar rápidamente un recurso según las etiquetas que le haya asignado.

- Consulte AWS las facturas desglosadas por etiquetas.
- Controle el acceso a los recursos de Amazon Keyspaces en función de etiquetas. Para ver ejemplos de políticas de IAM que utilicen etiquetas, consulte the section called “Autorización basada en etiquetas de Amazon Keyspaces”.

El etiquetado es compatible con AWS servicios como Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Keyspaces y más. Un etiquetado eficiente puede ofrecerle información detallada sobre los costos permitiendo crear informes sobre los servicios que llevan una etiqueta determinada.

Para empezar a usar etiquetas, haga lo siguiente:

1. Comprender Restricciones de etiquetado para Amazon Keyspaces.
2. Crear etiquetas mediante Operaciones de etiquetado para Amazon Keyspaces.
3. Úselo Informes de asignación de costos para Amazon Keyspaces para realizar un seguimiento de sus AWS costes por etiqueta activa.

Por último, es conveniente seguir estrategias de etiquetado óptimas. Para obtener más información, consulte Estrategias de etiquetado de AWS.

Restricciones de etiquetado para Amazon Keyspaces

Cada etiqueta consta de una clave y un valor, ambos definidos por el usuario. Se aplican las siguientes restricciones:

- Cada espacio de claves o tabla de Amazon Keyspaces solo puede tener una única etiqueta con la misma clave. Si intenta agregar una etiqueta existente (con la misma clave), el valor de la etiqueta existente se actualiza con el valor nuevo.
- Las etiquetas aplicadas a un espacio de claves no se aplican automáticamente a las tablas dentro de ese espacio de claves. Para aplicar la misma etiqueta a un espacio de claves y a todas sus tablas, cada recurso debe etiquetarse individualmente.
- Al crear un espacio de claves o una tabla multirregión, las etiquetas que defina durante el proceso de creación se aplicarán automáticamente a todos los espacios de claves y tablas de todas las regiones. Al modificar las etiquetas existentes utilizando `ALTER KEYSPACE` o `ALTER TABLE`, la actualización solo se aplica al espacio de claves o a la tabla de la región en la que esté realizando el cambio.

- Un valor actúa como un descriptor en una categoría de etiquetas (clave). En Amazon Keyspaces, el valor no puede estar vacío ni ser nulo.
- Las claves y los valores de las etiquetas distinguen entre mayúsculas y minúsculas.
- La longitud máxima de la clave es de 128 caracteres Unicode.
- La longitud máxima del valor es de 256 caracteres Unicode.
- Los caracteres permitidos son letras, espacios en blanco y números, además de los caracteres especiales siguientes: `+ - = . _ : /`
- El número máximo de etiquetas por recurso es 50.
- A los nombres y valores de etiquetas asignados por AWS se les asigna automáticamente el prefijo `aws:`, que usted no puede asignar. Los nombres de etiquetas asignados por AWS no cuentan para el límite de etiquetas de 50. Los nombres de etiquetas asignados por el usuario presentan el prefijo `user:` en el informe de asignación de costos.
- No es posible antedatar la aplicación de una etiqueta.

Operaciones de etiquetado para Amazon Keyspaces

Puede añadir, enumerar, editar o eliminar etiquetas de espacios de claves y tablas mediante la consola de Amazon Keyspaces (para Apache Cassandra), la CLI de AWS o Cassandra Query Language (CQL). A continuación, puede activar estas etiquetas definidas por el usuario de modo que aparezcan en la consola de AWS Billing and Cost Management y así poder usarlas para el seguimiento de asignación de costos. Para obtener más información, consulte Informes de asignación de costos para Amazon Keyspaces.

Para la edición en bloque, también puede utilizar el Editor de etiquetas en la consola. Para obtener más información, consulte Uso de Tag Editor en la Guía del usuario de AWS Resource Groups.

Temas

- Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante la consola
- Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante la CLI de AWS
- Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante CQL

Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante la consola

Puede utilizar la consola de Amazon Keyspaces para añadir etiquetas a nuevos espacios de claves y tablas al crearlos. También puede añadir, editar o eliminar etiquetas de tablas existentes.

Para etiquetar espacios de claves al crearlos (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Espacios de claves y, a continuación, Crear espacio de claves.
3. En la página Crear espacio de claves, proporcione un nombre para el espacio de claves. Introduzca una clave y un valor para la etiqueta y, a continuación, elija Añadir nueva etiqueta.
4. Elija Crear espacio de claves.

Para etiquetar tablas al crearlas (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tables (Tablas) y, a continuación, seleccione Create table (Crear tabla).
3. En la página Crear tabla, en la sección Detalles de la tabla, seleccione un espacio de claves y proporcione un nombre para la tabla.
4. En la sección Esquema, cree el esquema para su tabla.
5. En la sección Configuración de la tabla, elija Personalizar configuración.
6. Continúe en la sección Etiquetas de la tabla - opcional y elija Añadir nueva etiqueta para crear nuevas etiquetas.
7. Elija Crear tabla.

Para etiquetar recursos existentes (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Espacios de claves o Tablas.

3. Elija un espacio de claves o una tabla en la lista. A continuación, elija Administrar etiquetas para añadir, editar o eliminar sus etiquetas.

Para obtener más información sobre la estructura de las etiquetas, consulte Restricciones de etiquetado para Amazon Keyspaces.

Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante la CLI de AWS

En los ejemplos de esta sección se muestra cómo utilizar la CLI de AWS para especificar etiquetas al crear espacios de claves y tablas, cómo añadir o eliminar etiquetas de recursos existentes y cómo enumerar etiquetas.

En el siguiente ejemplo se muestra cómo crear una nueva tabla con etiquetas. El comando crea una tabla myTable en un espacio de claves ya existente myKeyspace. Tenga en cuenta que el comando se ha dividido en diferentes líneas para facilitar la lectura.

```
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'
```

En el siguiente ejemplo se muestra cómo añadir nuevas etiquetas a una tabla existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111222333444:/
keyspace/myKeyspace/table/myTable' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

En el siguiente ejemplo se muestra cómo enumerar las etiquetas del recurso especificado.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-
east-1:111222333444:/keyspace/myKeyspace/table/myTable'
```

La salida del último comando tiene este aspecto.

```
{
    "tags": [
        {
            "key": "key1",
            "value": "val1"
```

```
        },
        {
            "key": "key2",
            "value": "val2"
        },
        {
            "key": "key3",
            "value": "val3"
        },
        {
            "key": "key4",
            "value": "val4"
        }
    ]
}
```

Adición de etiquetas a espacios de claves y tablas nuevos o existentes mediante CQL

En los siguientes ejemplos se muestra cómo utilizar CQL para especificar etiquetas al crear espacios de claves y tablas, cómo etiquetar recursos existentes y cómo leer etiquetas.

En el siguiente ejemplo se crea un nuevo espacio de claves con etiquetas.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'} ;
```

En el siguiente ejemplo se crea una nueva tabla con etiquetas.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

Para etiquetar recursos en una instrucción con otros comandos.

```
CREATE KEYSPACE mykeyspace WITH REPLICATION = {'class': 'Simple Strategy'} AND TAGS
 = {'key1':'val1', 'key2':'val2'};
```

En el siguiente ejemplo se muestra cómo añadir o eliminar etiquetas en espacios de claves y tablas existentes.

```
ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

```
ALTER TABLE mytable DROP TAGS {'key1':'val1', 'key2':'val2'};
```

Para leer las etiquetas adjuntas a un recurso, utilice la siguiente instrucción CQL.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

La cláusula WHERE es obligatoria y debe tener uno de los siguientes formatos:

- keyspace_name = **'mykeyspace'** AND resource_type = 'keyspace'
- keyspace_name = **'mykeyspace'** AND resource_name = **'mytable'**
- resource_id = *arn*

Ejemplos:

La siguiente consulta muestra si un espacio de claves tiene etiquetas.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_type = 'keyspace';
```

La salida de la consulta tiene el siguiente aspecto.

```
resource_id                                                    | keyspace_name |
 resource_name | resource_type | tags
---------------------------------------------------------------+---------------
+---------------+---------------+------
arn:aws:cassandra:us-east-1:123456789:/keyspace/mykeyspace/      | mykeyspace    |
 mykeyspace    | keyspace      | {'key1': 'val1', 'key2': 'val2'}
```

La siguiente consulta muestra las etiquetas de una tabla.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
 resource_name = 'mytable';
```

La salida de esa consulta tiene el siguiente aspecto.

```
resource_id                                                                  |
 keyspace_name | resource_name | resource_type | tags
------------------------------------------------------------------------------
+---------------+---------------+---------------+------
arn:aws:cassandra:us-east-1:123456789:/keyspace/mykeyspace/table/mytable    |
 mykeyspace    | mytable       | table         | {'key1': 'val1', 'key2': 'val2'}
```

Informes de asignación de costos para Amazon Keyspaces

AWS utiliza etiquetas para organizar los costos de los recursos en el informe de asignación de costos. AWS proporciona dos tipos de etiquetas de asignación de costos:

- Una etiqueta generada por AWS. AWS define, crea y aplica esta etiqueta por usted.
- Etiquetas definidas por el usuario. Usted puede definir, crear y aplicar estas etiquetas.

Debe activar ambos tipos de etiquetas por separado para que puedan aparecer en Cost Explorer o en un informe de asignación de costos.

Para activar las etiquetas generadas por AWS:

1. Inicie sesión en la AWS Management Console abra la consola de Billing and Cost Management en https://console.aws.amazon.com/billing/home#/.
2. En el panel de navegación, seleccione Cost Allocation Tags.
3. En Etiquetas de asignación de costes generada por AWS, elija Activate.

Para activar las etiquetas definidas por el usuario:

1. Inicie sesión en la AWS Management Console abra la consola de Billing and Cost Management en https://console.aws.amazon.com/billing/home#/.
2. En el panel de navegación, seleccione Cost Allocation Tags.
3. En User-Defined Cost Allocation Tags, elija Activate.

Después de crear y activar las etiquetas, AWS genera un informe de asignación de costos con el uso y los costos agrupados según las etiquetas activas. El informe de asignación de costes incluye todos

los costes de AWS para cada periodo de facturación. El informe incluye tanto recursos etiquetados como sin etiquetar, para que pueda organizar con claridad los cargos de los recursos.

> **Note**
>
> En la actualidad, los datos transferidos desde Amazon Keyspaces no se desglosan por etiquetas en los informes de asignación de costos.

Para obtener más información, consulte Uso de etiquetas de asignación de costos.

Prácticas recomendadas de diseño y arquitectura con Amazon Keyspaces

Utilice esta sección para encontrar con rapidez recomendaciones para maximizar el rendimiento y minimizar los costos de rendimiento al trabajar con Amazon Keyspaces.

Contenido

- Diseño NoSQL de Amazon Keyspaces
  - Diferencias entre el diseño de datos relacionales y NoSQL
  - Dos conceptos clave del diseño NoSQL
  - Aproximación al diseño NoSQL
- Conexiones del controlador de cliente a Amazon Keyspaces (para Apache Cassandra)
  - Funcionamiento de las conexiones en Amazon Keyspaces
  - Configuración de conexiones en Amazon Keyspaces
  - Configuración de conexiones sobre puntos de conexión de VPC en Amazon Keyspaces
  - Monitoreo de conexiones en Amazon Keyspaces
  - Gestión de errores de conexión en Amazon Keyspaces
- Modelado de datos en Amazon Keyspaces (para Apache Cassandra)
  - Uso eficaz de las claves de partición en Amazon Keyspaces
    - Uso de la partición de escritura para distribuir de forma uniforme las cargas de trabajo en Amazon Keyspaces
      - Fragmentación mediante claves de partición compuestas y valores aleatorios
      - Partición mediante claves de partición compuestas y valores aleatorios
- Optimización de costos de las tablas de Amazon Keyspaces
  - Evaluar los costos en el nivel de tabla
    - Visualización de costos de una única tabla de Amazon Keyspaces
    - Vista predeterminada del Explorador de costos
    - Cómo utilizar y aplicar las etiquetas de tabla en el Explorador de costos
  - Evaluación del modo de capacidad de una tabla
    - Qué modos de capacidad de tabla hay disponibles
    - Cuándo seleccionar el modo de capacidad bajo demanda

    - Cuándo seleccionar el modo de capacidad aprovisionada
    - Factores adicionales que se deben tener en cuenta al elegir un modo de capacidad de tabla
- Evaluación de los ajustes de Application Auto Scaling de su tabla
    - Comprensión de los ajustes de Application Auto Scaling
    - Cómo identificar tablas con un uso objetivo bajo (<=50 %)
    - Cómo abordar las cargas de trabajo con variaciones estacionales
    - Cómo abordar cargas de trabajo con picos con patrones desconocidos
    - Cómo abordar las cargas de trabajo con aplicaciones enlazadas
- Identificación de recursos no utilizados
    - Cómo identificar los recursos sin utilizar
    - Identificación de recursos de tabla sin utilizar
    - Limpieza de los recursos de tabla no utilizados
    - Limpiar las copias de seguridad point-in-time de recuperación no utilizadas (PITR)
- Evaluar los patrones de uso de las tablas
    - Realizar menos operaciones de lectura altamente coherente
    - Habilitar el Periodo de vida (TTL)
- Evaluar la capacidad aprovisionada para lograr un aprovisionamiento del tamaño adecuado
    - Recuperación de métricas de consumo de sus tablas de Amazon Keyspaces
    - Identificación de tablas de DynamoDB subaprovisionadas
    - Identificación de tablas de DynamoDB sobreaprovisionadas

Diseño NoSQL de Amazon Keyspaces

Los sistemas de bases de datos NoSQL como Amazon Keyspaces utilizan modelos alternativos para la administración de datos, como pares clave-valor o almacenamiento de documentos. Al pasar de un sistema de administración de bases de datos relacionales a un sistema de bases de datos NoSQL como Amazon Keyspaces, es importante comprender las diferencias clave y los enfoques de diseño específicos.

Temas

- Diferencias entre el diseño de datos relacionales y NoSQL
- Dos conceptos clave del diseño NoSQL

- Aproximación al diseño NoSQL

Diferencias entre el diseño de datos relacionales y NoSQL

Los sistemas de bases de datos relacionales (RDBMS) y las bases de datos NoSQL tienen diferentes ventajas y desventajas:

- En RDBMS, los datos se pueden consultar de manera flexible, pero las consultas son relativamente costosas y no escalan bien cuando hay mucho tráfico (consulte the section called "Modelado de datos").
- En una base de datos NoSQL como Amazon Keyspaces, los datos se pueden consultar de manera eficiente en un número limitado de formas, fuera de las cuales las consultas podrían resultar caras y lentas.

Estas diferencias hacen que el diseño de las bases de datos sea muy distinto entre los dos sistemas:

- En RDBMS, el diseño busca la flexibilidad sin preocuparse por los detalles o el rendimiento de la implementación. Por lo general, la optimización de consultas no afecta al diseño del esquema, pero la normalización es importante.
- En Amazon Keyspaces, usted diseña específicamente su esquema para que las consultas más comunes e importantes sean lo más rápidas y económicas posible. Las estructuras de datos se ajustan a los requisitos específicos de los casos de uso de la organización.

Dos conceptos clave del diseño NoSQL

El diseño NoSQL requiere un modo de pensar distinto al diseño de RDBMS. En un sistema RDBMS, puede empezar a crear un modelo de datos normalizados sin pensar en los patrones de acceso. Posteriormente, podrá ampliar este modelo cuando surjan nuevos requisitos sobre preguntas y consultas. Puede organizar cada tipo de datos en su propia tabla.

Cómo es diferente el diseño NoSQL:

- Por el contrario, no debería comenzar a diseñar su esquema para Amazon Keyspaces hasta que conozca las preguntas a las que debe responder. Es esencial conocer de antemano los problemas del negocio y los casos de uso de la aplicación.
- En una aplicación de Amazon Keyspaces debería mantener el menor número posible de tablas. Tener menos tablas permite que las cosas sean más escalables, requiere menos administración

de permisos y reduce la sobrecarga de su aplicación de Amazon Keyspaces. También puede ayudar a mantener los costos de las copias de seguridad generalmente más bajos.

Aproximación al diseño NoSQL

El primer paso en el diseño de su aplicación de Amazon Keyspaces es identificar los patrones de consulta específicos que debe satisfacer el sistema.

En particular, antes de empezar, es importante entender tres propiedades fundamentales de los patrones de acceso de la aplicación:

- Tamaño de los datos: saber cuántos datos se almacenarán y solicitarán a la vez permite determinar la forma más eficaz de particionar los datos.
- Forma de los datos: en lugar de dar forma a los datos al procesar las consultas (como ocurre en los sistemas RDBMS), las bases de datos NoSQL organizan los datos de modo que la forma que tienen en la base de datos se corresponde con la que se va a consultar. Este es un factor crucial para aumentar la velocidad y la escalabilidad.
- Velocidad de los datos: Amazon Keyspaces escala mediante el aumento del número de particiones físicas disponibles para procesar las consultas y la distribución eficiente de los datos entre dichas particiones. Conocer de antemano cuáles serán las cargas de consulta máximas puede ayudar a determinar cómo deben particionarse los datos para hacer un uso óptimo de la capacidad de E/S.

Después de identificar los requisitos de consulta específicos, puede organizar los datos con arreglo a los principios generales que rigen el rendimiento:

- Agrupar los datos relacionales. Cuando hace 20 años se investigaba cómo optimizar las tablas de direccionamiento, se descubrió que la "cercanía de referencias" era el factor más importante para acelerar el tiempo de respuesta y consistía en mantener los datos relacionados reunidos en el mismo lugar. Esto sigue siendo aplicable a los sistemas NoSQL actuales, donde mantener los datos relacionales cerca unos de otros tiene un impacto determinante en los costos y el rendimiento. En lugar de distribuir los elementos de datos relacionados entre diferentes tablas, en los sistemas NoSQL deben mantenerse lo más juntos posible.

  Como regla general, en una aplicación de Amazon Keyspaces debería mantener el menor número posible de tablas.

Las excepciones son aquellos casos en los que hay implicados datos de serie temporal de gran volumen o conjuntos de datos que tienen patrones de acceso muy diferentes. Normalmente, basta una sola tabla con índices invertidos para permitir que, a través de consultas simples, se creen y recuperen las estructuras de datos jerárquicas y complejas que necesita la aplicación.

- Utilizar un orden de clasificación. Los elementos relacionados pueden agruparse y consultarse de forma eficaz si su diseño de claves hace que se ordenen juntos. Esta es una estrategia importante en el diseño NoSQL.
- Distribuir consultas. También es importante que no haya una gran cantidad de consultas concentradas en la misma parte de la base de datos, donde podría sobrepasarse la capacidad de E/S. En su lugar, debe diseñar claves de datos que distribuyan el tráfico lo más uniformemente posible entre las particiones y eviten la creación de "puntos calientes".

Estos principios generales se traducen en unos patrones de diseño comunes que puede utilizar para modelar los datos de forma eficaz en Amazon Keyspaces.

Conexiones del controlador de cliente a Amazon Keyspaces (para Apache Cassandra)

Para comunicar con Amazon Keyspaces, puede utilizar cualquiera de los controladores de cliente de Apache Cassandra existentes de su elección. Dado que Amazon Keyspaces es un servicio sin servidor, le recomendamos que optimice la configuración de conexión de su controlador de cliente para las necesidades de rendimiento de su aplicación. En este tema se presentan prácticas recomendadas, incluso de cómo calcular cuántas conexiones necesita su aplicación, así como la monitorización y gestión de errores de las conexiones.

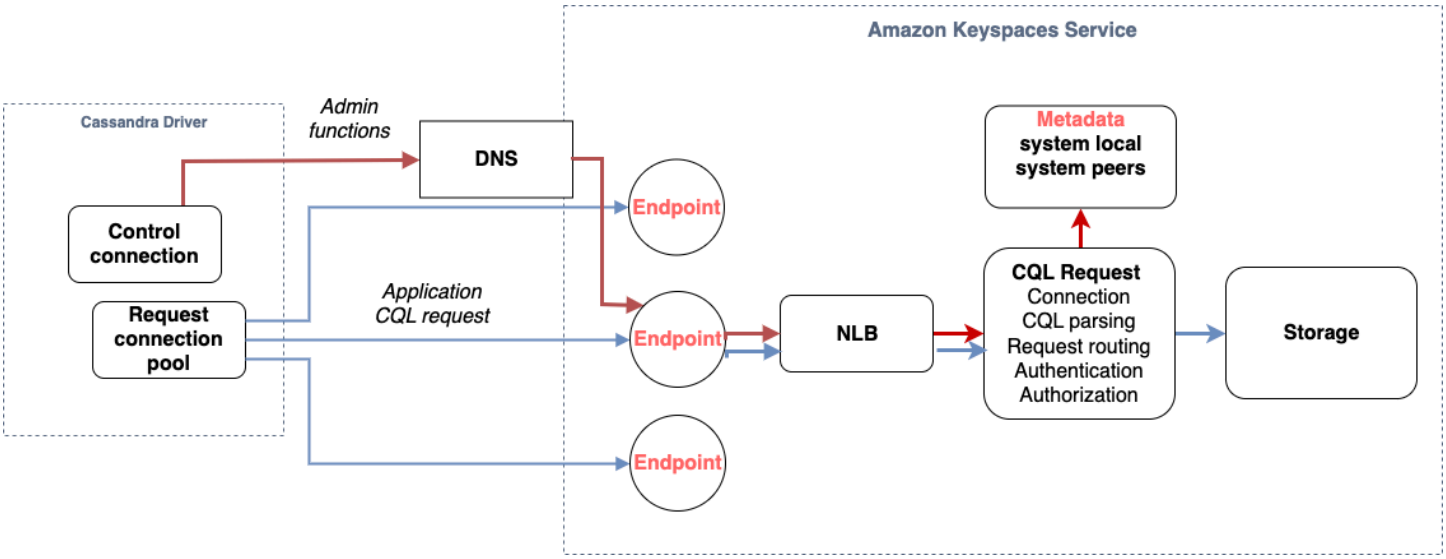
Temas

- Funcionamiento de las conexiones en Amazon Keyspaces
- Configuración de conexiones en Amazon Keyspaces
- Configuración de conexiones sobre puntos de conexión de VPC en Amazon Keyspaces
- Monitoreo de conexiones en Amazon Keyspaces
- Gestión de errores de conexión en Amazon Keyspaces

Funcionamiento de las conexiones en Amazon Keyspaces

En esta sección se ofrece información general del funcionamiento de las conexiones del controlador de cliente en Amazon Keyspaces. Dado que una configuración incorrecta del controlador de cliente de Cassandra podría causar eventos `PerConnectionRequestExceeded` en Amazon Keyspaces, es necesario configurar la cantidad correcta de conexiones en la configuración del controlador de cliente para evitar estos errores de conexión y otros similares.

Al conectarse a Amazon Keyspaces, el controlador requiere un punto de conexión inicial para establecer una conexión inicial. Amazon Keyspaces utiliza DNS para dirigir la conexión inicial a uno de los muchos puntos de conexión disponibles. Los puntos de conexión están vinculados a equilibradores de carga de red que, a su vez, establecen una conexión con uno de los controladores de solicitudes de la flota. Una vez establecida la conexión inicial, el controlador de cliente recopila información sobre todos los puntos de conexión disponibles en la tabla `system.peers`. Con esta información, el controlador de cliente puede crear conexiones adicionales a los puntos de conexión de la lista. El número de conexiones que puede crear el controlador de cliente está limitado por el número de conexiones locales especificado en la configuración del controlador cliente. De forma predeterminada, la mayoría de los controladores de cliente establecen una conexión por punto de conexión y crean un grupo de conexiones a Cassandra y equilibran la carga de consultas sobre ese grupo de conexiones. Si bien se pueden establecer múltiples conexiones al mismo punto de conexión, detrás del equilibrador de carga de red podrían conectarse a muchos controladores de solicitudes diferentes. Al conectarse a través del punto de conexión público, el establecimiento de una conexión a cada uno de los nueve puntos de conexión enumerados en la tabla `system.peers` da como resultado nueve conexiones a diferentes controladores de solicitudes.



Configuración de conexiones en Amazon Keyspaces

Amazon Keyspaces admite hasta 3000 consultas CQL por conexión TCP por segundo. Dado que no hay límite en el número de conexiones que un controlador puede establecer, recomendamos apuntar solo a 500 consultas CQL por segundo por conexión para tener en cuenta la sobrecarga, los picos de tráfico y un mejor equilibrio de la carga. Siga estos pasos para asegurarse de que la conexión de su controlador esté correctamente configurada para las necesidades de su aplicación.

Aumente el número de conexiones por dirección IP que su controlador mantiene en su grupo de conexiones.

- La mayoría de los controladores de Cassandra establecen un grupo de conexiones a Cassandra y equilibran la carga de consultas sobre ese grupo de conexiones. El comportamiento predeterminado de la mayoría de los controladores es establecer una única conexión con cada punto de conexión. Amazon Keyspaces expone nueve direcciones IP homólogas a los controladores, por lo que, en función del comportamiento predeterminado de la mayoría de los controladores, esto se traduce en 9 conexiones. Amazon Keyspaces admite hasta 3000 consultas CQL por conexión TCP por segundo, por lo que el rendimiento máximo de consulta CQL de un controlador que utilice la configuración predeterminada es de 27 000 consultas CQL por segundo. Si utiliza la configuración predeterminada del controlador, es posible que una sola conexión tenga que procesar más del rendimiento máximo de consulta CQL de 3000 consultas CQL por segundo. Esto podría dar lugar a eventos `PerConnectionRequestExceeded`.
- Para evitar los eventos `PerConnectionRequestExceeded`, debe configurar el controlador para crear conexiones adicionales por punto de conexión para distribuir el rendimiento.
- Como práctica recomendada en Amazon Keyspaces, suponga que cada conexión pueda admitir 500 consultas CQL por segundo.
- Esto significa que, para una aplicación de producción que deba admitir unas 27 000 consultas CQL por segundo distribuidas entre los nueve puntos de conexión disponibles, deberá configurar seis conexiones por punto de conexión. Esto garantiza que cada conexión no procese más de 500 solicitudes por segundo.

Calcule el número de conexiones por dirección IP que necesita configurar para su controlador en función de las necesidades de su aplicación.

Para determinar el número de conexiones que necesita configurar por punto de conexión para su aplicación, considere el siguiente ejemplo. Usted tiene una aplicación que debe admitir 20 000 consultas CQL por segundo consistentes en 10 000 operaciones `INSERT`, 5000 operaciones

SELECT y 5000 operaciones DELETE. La aplicación Java se ejecuta en tres instancias en Amazon Elastic Container Service (Amazon ECS) donde cada instancia establece una única sesión en Amazon Keyspaces. El cálculo que puede utilizar para estimar cuántas conexiones necesita configurar para su controlador utiliza la siguiente entrada.

1. El número de solicitudes por segundo que debe admitir su aplicación.
2. El número de instancias disponibles, menos una a fin de tener en cuenta mantenimiento o fallos.
3. El número de puntos de conexión disponibles. Si se conecta a través de puntos de conexión públicos, cuenta con nueve puntos de conexión disponibles. Si utiliza puntos de conexión de VPC, tiene entre dos y cinco puntos de conexión disponibles, según la región.
4. Utilice 500 consultas CQL por segundo y por conexión como práctica recomendada para Amazon Keyspaces.
5. Redondee el resultado.

En este ejemplo, la fórmula tiene este aspecto.

```
20,000 CQL queries / (3 instances - 1 failure) / 9 public endpoints / 500 CQL queries
 per second = ROUND(2.22) = 3
```

En función de este cálculo, debe especificar tres conexiones locales por punto de conexión en la configuración del controlador. Para las conexiones remotas, configure solo una conexión por punto de conexión.

Configuración de conexiones sobre puntos de conexión de VPC en Amazon Keyspaces

Al conectarse a través de puntos de conexión de VPC privados, lo más probable es que disponga de menos de 9 puntos de conexión. Además, el número de puntos de conexión de VPC puede ser diferente por región, en función del número de zonas de disponibilidad y del número de subredes de la VPC asignada. La región Este de EE. UU. (Norte de Virginia) tiene cinco zonas de disponibilidad y puede tener hasta cinco puntos de conexión de Amazon Keyspaces. La región Oeste de EE. UU. (Norte de California) tiene dos zonas de disponibilidad y puede tener hasta dos puntos de conexión de Amazon Keyspaces. El número de puntos de conexión no afecta a la escala, pero sí aumenta el número de conexiones que necesita establecer en la configuración del controlador. Considere el siguiente ejemplo. Su aplicación debe admitir 20 000 consultas CQL y se ejecuta en tres instancias

en Amazon ECS, donde cada instancia establece una única sesión con Amazon Keyspaces. La única diferencia es el número de puntos de conexión disponibles en las distintas Regiones de AWS.

Conexiones necesarias en la región Este de EE. UU. (Norte de Virginia):

```
20,000 CQL queries / (3 instances - 1 failure) / 5 private VPC endpoints / 500 CQL
 queries per second = 4 local connections
```

Conexiones necesarias en la región Oeste de EE. UU. (Norte de California):

```
20,000 CQL queries / (3 instances - 1 failure) / 2 private VPC endpoints / 500 CQL
 queries per second = 10 local connections
```

> **Important**
>
> Cuando se utilizan puntos de conexión de VPC privados, se requieren permisos adicionales para que Amazon Keyspaces descubra los puntos de conexión de VPC disponibles de forma dinámica y rellene la tabla `system.peers`. Para obtener más información, consulte the section called “Relleno de entradas de la tabla `system.peers` con información del punto de conexión de VPC de interfaz”.

Al acceder a Amazon Keyspaces a través de un punto de enlace de VPC privado utilizando otro Cuenta de AWS, es probable que solo vea un punto de enlace de Amazon Keyspaces. Una vez más, esto no repercute en la escala del posible rendimiento de Amazon Keyspaces, pero es posible que tenga que aumentar el número de conexiones en la configuración de su controlador. En este ejemplo se muestra el mismo cálculo para un único punto de conexión disponible.

```
20,000 CQL queries / (3 instances - 1 failure) / 1 private VPC endpoints / 500 CQL
 queries per second = 20 local connections
```

Para obtener más información sobre el acceso entre cuentas a Amazon Keyspaces mediante una VPC compartida, consulte the section called “Acceso entre cuentas en una VPC compartida”.

Monitoreo de conexiones en Amazon Keyspaces

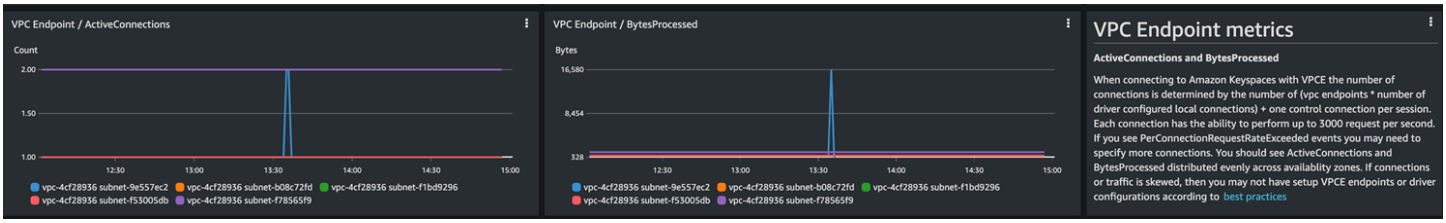
Con el fin de identificar el número de puntos de conexión a los que está conectada su aplicación, puede registrar el número de pares descubiertos en la tabla `system.peers`. En el siguiente ejemplo, un ejemplo de código Java imprime el número de pares una vez establecida la conexión.

```
ResultSet result = session.execute(new SimpleStatement("SELECT * FROM system.peers"));

logger.info("number of Amazon Keyspaces endpoints:" + result.all().stream().count());
```

> **Note**
>
> La consola o AWS consola CQL no se implementan en una VPC y, por lo tanto, utilizan el punto final público. Como resultado, la ejecución de la consulta `system.peers` desde aplicaciones ubicadas fuera de la VPCE suele dar como resultado 9 pares. También podría ser útil imprimir las direcciones IP de cada par.

También puedes observar el número de pares cuando utilizas un punto final de VPC configurando las métricas de Amazon CloudWatch de VPCE. En CloudWatch, puede ver la cantidad de conexiones establecidas con el punto final de la VPC. Los controladores de Cassandra establecen una conexión para cada punto de conexión para enviar consultas CQL y una conexión de control para recopilar información de la tabla del sistema. La siguiente imagen muestra las CloudWatch métricas de los puntos de conexión de la VPC tras conectarse a Amazon Keyspaces con 1 conexión configurada en la configuración del controlador. La métrica muestra seis conexiones activas consistentes en una conexión de control y cinco conexiones (1 por punto de conexión en todas las zonas de disponibilidad).



Para empezar a monitorizar el número de conexiones mediante un CloudWatch gráfico, puede implementar esta AWS CloudFormation plantilla disponible GitHub en el repositorio de plantillas de Amazon Keyspaces.

Gestión de errores de conexión en Amazon Keyspaces

Cuando se supera la cuota de 3000 solicitudes por conexión, Amazon Keyspaces devuelve un evento `PerConnectionRequestExceeded` y el controlador de Cassandra recibe una excepción `WriteTimeout` o `ReadTimeout`. Debería reintentar esta excepción con un retroceso exponencial en su política de reintentos de Cassandra o en su aplicación. Debería proporcionar un retroceso exponencial para evitar el envío de solicitudes adicionales.

La política de reintento predeterminada intenta `try next host` en el plan de consultas. Dado que Amazon Keyspaces podría tener de uno a tres puntos de conexión disponibles al conectarse al punto de conexión de VPC, es posible que también vea la `NoHostAvailableException` además de las excepciones `WriteTimeout` y `ReadTimeout` en los registros de su aplicación. Puede utilizar las políticas de reintento proporcionadas por Amazon Keyspaces, que reintentan en el mismo punto de conexión pero a través de conexiones diferentes.

Puede encontrar ejemplos de políticas de reintentos exponenciales para Java GitHub en el repositorio de ejemplos de código Java de Amazon Keyspaces. Puede encontrar ejemplos adicionales de lenguaje en Github en el repositorio Ejemplos de código de Amazon Keyspaces.

Modelado de datos en Amazon Keyspaces (para Apache Cassandra)

En este tema se presentan conceptos de modelado de datos en Amazon Keyspaces (para Apache Cassandra). Utilice esta sección para encontrar recomendaciones a fin de diseñar modelos de datos que se alineen con los patrones de acceso a datos de su aplicación. La implementación de las prácticas recomendadas de modelado de datos mejora el rendimiento y minimiza los costos de procesamiento al trabajar con Amazon Keyspaces.

Para visualizar y diseñar modelos de datos con mayor facilidad, puede utilizar NoSQL Workbench.

Temas

- Uso eficaz de las claves de partición en Amazon Keyspaces

Uso eficaz de las claves de partición en Amazon Keyspaces

La clave principal que identifica de forma exclusiva cada fila de una tabla de Amazon Keyspaces puede constar de una o varias columnas de clave de partición que determinan en qué particiones se almacenan los datos y de una o varias columnas de agrupación opcionales que definen cómo se agrupan y ordenan los datos dentro de una partición.

Dado que la clave de partición establece el número de particiones en las que se almacenan sus datos y cómo se distribuyen los datos entre estas particiones, la forma en que elija su clave de partición puede tener un impacto significativo en el rendimiento de sus consultas. En general, debe diseñar su aplicación para que la actividad sea uniforme en todas las particiones del disco.

Distribuir la actividad de lectura y escritura de su aplicación de forma uniforme por todas las particiones ayuda a minimizar los costos de rendimiento y esto se aplica a los modos de capacidad de lectura/escritura tanto bajo demanda como aprovisionada. Por ejemplo, si utiliza el modo de capacidad aprovisionada, puede determinar los patrones de acceso que necesita su aplicación y estimar el total de unidades de capacidad de lectura (RCU) y de unidades de capacidad de escritura (WCU) que requiere cada tabla. Amazon Keyspaces admite sus patrones de acceso utilizando el rendimiento que haya aprovisionado siempre que el tráfico que afecte a una partición determinada no supere las 3000 RCU y las 1000 WCU.

Amazon Keyspaces ofrece flexibilidad adicional en su aprovisionamiento de rendimiento por partición proporcionando capacidad de ampliación. Para obtener más información, consulte the section called “Capacidad de ampliación”.

Temas

- Uso de la partición de escritura para distribuir de forma uniforme las cargas de trabajo en Amazon Keyspaces

Uso de la partición de escritura para distribuir de forma uniforme las cargas de trabajo en Amazon Keyspaces

Una forma de distribuir mejor las escrituras en una partición en Amazon Keyspaces es ampliar el espacio. Esto puede hacerse de diferentes maneras. Puede añadir una columna de clave de partición adicional en la que escriba números aleatorios para distribuir las filas entre las particiones. O puede usar un número que se calcula en función de algo que esté consultando.

Fragmentación mediante claves de partición compuestas y valores aleatorios

Una estrategia para distribuir cargas de forma más uniforme en una partición consiste en añadir una columna de clave de partición adicional en la que se escriben números aleatorios. De ese modo, las escrituras se distribuyen aleatoriamente por un espacio mayor.

Por ejemplo, considere la siguiente tabla que tiene una única clave de partición que representa una fecha.

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   title text,
   description int,
   PRIMARY KEY (publish_date));
```

Para distribuir de forma más uniforme esta tabla entre las particiones, podría incluir una columna de clave de partición adicional `shard` que almacene números aleatorios. Por ejemplo:

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
   publish_date date,
   shard int,
   title text,
   description int,
   PRIMARY KEY ((publish_date, shard)));
```

Al insertar datos podría elegir un número aleatorio entre `1` y `200` para la columna `shard`. Esto produce valores de clave de partición compuestos como `(2020-07-09, 1)`, `(2020-07-09, 2)` y así sucesivamente hasta `(2020-07-09, 200)`. Al aplicar un número aleatorio a la clave de partición, las escrituras que se producen en la tabla de cada día se distribuyen uniformemente por varias particiones. Como resultado, se mejora el paralelismo y el rendimiento general.

Sin embargo, para leer todas las filas de un día determinado, tendría que consultar las filas de todas las particiones y luego fusionar los resultados. Por ejemplo, primero emitiría una instrucción `SELECT` para el valor de la clave de partición `(2020-07-09, 1)`. A continuación, emitiría otra instrucción `SELECT` para `(2020-07-09, 2)` y así sucesivamente hasta `(2020-07-09, 200)`. Por último, su aplicación tendría que fusionar los resultados de todas esas instrucciones `SELECT`.

Partición mediante claves de partición compuestas y valores aleatorios

Aplicar una estrategia de aleatorización puede mejorar considerablemente el rendimiento de la escritura. Pero resulta difícil leer una fila en concreto porque no sabe qué valor se escribió en la columna `shard` al escribirse la fila. Para facilitar la lectura de filas individuales, puede utilizar una estrategia diferente. En vez de utilizar un número aleatorio para distribuir las filas entre las particiones, utilice un número que pueda calcular basándose en algo que desee consultar.

Consideremos el ejemplo anterior, en el que una tabla utiliza la fecha de hoy en la clave de partición. Supongamos ahora que cada fila tiene una columna `title` accesible y que lo más frecuente es que necesite encontrar filas por título además de por fecha. Antes de que su aplicación escriba la fila en la tabla, podría calcular un valor hash basado en el título y utilizarlo para rellenar la columna `shard`. El cálculo podría generar un número comprendido entre 1 y 200 que esté bastante bien distribuido, igual que el que se genera con la estrategia aleatoria.

Probablemente bastaría con un cálculo sencillo, como el producto de los valores de punto de código UTF-8 de los caracteres del título, módulo 200, + 1. El valor de la clave de partición compuesta sería entonces la combinación de la fecha y el resultado del cálculo.

Con esta estrategia, las escrituras se distribuyen de manera uniforme entre los valores de clave de partición y, por lo tanto, entre las particiones físicas. Puede realizar fácilmente una instrucción `SELECT` para una fila y una fecha en concreto porque puede calcular el valor de clave de partición para un valor `title` específico.

Para leer todas las filas de un día determinado, aún debe `SELECT` cada una de las claves `(2020-07-09, N)` (donde `N` es de 1 a 200), y su aplicación tiene entonces que fusionar todos los resultados. El beneficio es que evitaría que un único valor de clave de partición "caliente" acaparase toda la carga de trabajo.

Optimización de costos de las tablas de Amazon Keyspaces

En esta sección se describen las prácticas recomendadas para optimizar los costos de sus tablas de Amazon Keyspaces existentes. Debería examinar las siguientes estrategias para ver qué estrategia de optimización de costos se adapta mejor a sus necesidades y abordarlas de forma iterativa. Cada estrategia proporciona información general sobre lo que puede estar afectando a sus costos, cómo buscar oportunidades para optimizar costos y orientación prescriptiva sobre cómo implementar estas prácticas recomendadas para ayudarle a ahorrar.

Temas

- Evaluar los costos en el nivel de tabla
- Evaluación del modo de capacidad de una tabla
- Evaluación de los ajustes de Application Auto Scaling de su tabla
- Identificación de recursos no utilizados
- Evaluar los patrones de uso de las tablas
- Evaluar la capacidad aprovisionada para lograr un aprovisionamiento del tamaño adecuado

Evaluar los costos en el nivel de tabla

La herramienta Cost Explorer que se encuentra en el AWS Management Console le permite ver los costos desglosados por tipo, como los cargos de lectura, escritura, almacenamiento y respaldo. También puede ver estos costos resumidos por periodo, por ejemplo, por mes o por día.

Un reto al que pueden enfrentarse los administradores es cuando hay que revisar los costos de solo una tabla concreta. Algunos de estos datos están disponibles a través de la consola de Amazon Keyspaces o mediante llamadas a la API de `GetTable`; sin embargo, el Explorador de costos no

permite, de forma predeterminada, filtrar ni agrupar por costos asociados a una tabla específica. En esta sección se mostrará cómo utilizar el etiquetado para realizar un análisis de costos de tablas individuales en el Explorador de costos.

Temas

- Visualización de costos de una única tabla de Amazon Keyspaces
- Vista predeterminada del Explorador de costos
- Cómo utilizar y aplicar las etiquetas de tabla en el Explorador de costos

Visualización de costos de una única tabla de Amazon Keyspaces

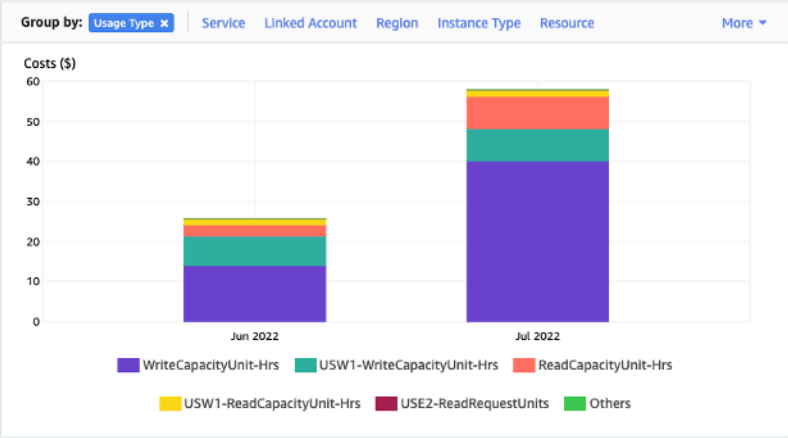
Tanto la consola de Amazon Keyspaces como la API de `GetTable` le muestran información sobre una única tabla, incluyendo el esquema de clave principal y el tamaño y el recuento de filas de la tabla. El tamaño de la tabla se puede utilizar para calcular el costo mensual de almacenamiento de su tabla. Por ejemplo, 0,25$ por GB en el Región de AWS US-east-1.

Si la tabla utiliza el modo de capacidad aprovisionada, también se devuelven los ajustes actuales de la unidad de capacidad de lectura (RCU) y de la unidad de capacidad de escritura (WCU). Estos se pueden utilizar para calcular los costos actuales de lectura y escritura de la tabla, pero estos costos podrían cambiar, en particular si la tabla se ha configurado con Application Auto Scaling.

> **Note**
>
> Si la tabla utiliza el modo de capacidad bajo demanda, entonces `GetTable` no ayuda en el cálculo de costos de rendimiento, dado que estos se facturan en función del uso real, no aprovisionado, en un periodo determinado.

Vista predeterminada del Explorador de costos

La vista predeterminada del Explorador de costos ofrece gráficos que muestran el costo de los recursos consumidos, por ejemplo, de rendimiento y almacenamiento. Puede optar por agrupar estos costos por periodos, como totales por mes o por día. Además, los costos de almacenamiento, lecturas, escrituras y otras categorías se pueden desglosar y comparar.



Cómo utilizar y aplicar las etiquetas de tabla en el Explorador de costos

De forma predeterminada, el Explorador de costos no proporciona un resumen de costos de ninguna tabla específica, porque combina los costos de múltiples tablas en un total. No obstante, puede utilizar el etiquetado de recursos de AWS para identificar cada tabla con una etiqueta de metadatos. Las etiquetas son pares clave-valor que puede utilizar para diversos fines, por ejemplo, para identificar todos los recursos pertenecientes a un proyecto o departamento. Para este ejemplo, utilizamos una tabla con el nombre. MyTable

1. Establece una etiqueta con la clave table_name y el valor de. MyTable
2. Active la etiqueta en el Explorador de costos y, a continuación, filtre el valor de la etiqueta para obtener una mayor visibilidad de los costos de cada tabla.

> **Note**
>
> La etiqueta puede tardar uno o dos días en comenzar a aparecer en el Explorador de costos

Puede configurar las etiquetas de metadatos usted mismo en la consola o mediante programación con el o el SDK. AWS CLI AWS Considere la posibilidad de exigir que se establezca una etiqueta table_name como parte del proceso de creación de una nueva tabla en su organización.

Evaluación del modo de capacidad de una tabla

En esta sección se ofrece información general sobre cómo seleccionar el modo de capacidad apropiado para su tabla de Amazon Keyspaces. Cada modo está ajustado para satisfacer las necesidades de una carga de trabajo diferente en cuanto a la capacidad de respuesta a los cambios

en el rendimiento, así como a la forma de facturar ese uso. Debe sopesar estos factores al tomar su decisión.

Temas

- Qué modos de capacidad de tabla hay disponibles
- Cuándo seleccionar el modo de capacidad bajo demanda
- Cuándo seleccionar el modo de capacidad aprovisionada
- Factores adicionales que se deben tener en cuenta al elegir un modo de capacidad de tabla

Qué modos de capacidad de tabla hay disponibles

Al crear una tabla de Amazon Keyspaces, debe seleccionar el modo de capacidad bajo demanda o aprovisionada. Para obtener más información, consulte the section called “Modos de capacidad de lectura/escritura”.

Modo de capacidad bajo demanda

El modo de capacidad bajo demanda está diseñado para eliminar la necesidad de planificar o aprovisionar la capacidad de su tabla de Amazon Keyspaces. En este modo, su tabla se adapta instantáneamente a las solicitudes sin necesidad de ampliar o reducir ningún recurso (hasta el doble del rendimiento máximo anterior de la tabla).

Las tablas bajo demanda se facturan contando el número de solicitudes reales que recibe la tabla, por lo que solo paga por lo que utiliza y no por lo que ha sido aprovisionada.

Modo de capacidad aprovisionada

El modo de capacidad aprovisionada es un modelo más tradicional en el que puede definir cuánta capacidad tiene disponible la tabla para solicitudes, ya sea de forma directa o con la ayuda de Application Auto Scaling. Dado que se aprovisiona una capacidad específica para la tabla en un momento dado, la facturación se basa en la capacidad aprovisionada y no en el número de solicitudes. Superar la capacidad asignada también puede hacer que la tabla rechace solicitudes y reducir la experiencia de los usuarios de su aplicación.

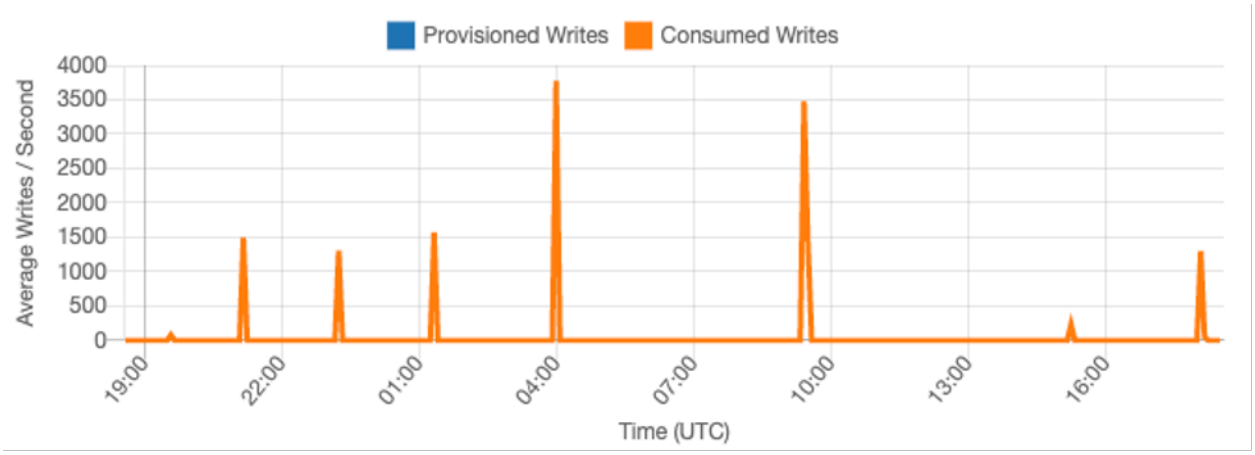
El modo de capacidad aprovisionada requiere un equilibrio entre no sobreaprovisionar ni subaprovisionar la tabla a fin de lograr tanto una baja incidencia de errores de capacidad de rendimiento insuficiente como costos optimizados.

Cuándo seleccionar el modo de capacidad bajo demanda

A la hora de optimizar costos, el modo bajo demanda es su mejor opción cuando tiene una carga de trabajo impredecible similar a la que se muestra en el siguiente gráfico.

Estos factores contribuyen a este tipo de carga de trabajo:

- Tiempo de solicitud imprevisible (lo que provoca picos de tráfico)
- Volumen variable de solicitudes (resultante de las cargas de trabajo por lotes)
- Cae a cero o por debajo del 18 % del pico para una hora determinada (que resulta de entornos de desarrollo o prueba)



En el caso de cargas de trabajo con las características anteriores, el uso de Application Auto Scaling para mantener una capacidad suficiente para que la tabla responda a los picos de tráfico podría dar lugar a resultados no deseados. O bien la tabla podría sobreaprovisionarse y costar más de lo necesario, o subaprovisionarse y las solicitudes darían lugar a errores innecesarios de baja capacidad de rendimiento. En casos como este, las tablas bajo demanda son la mejor opción.

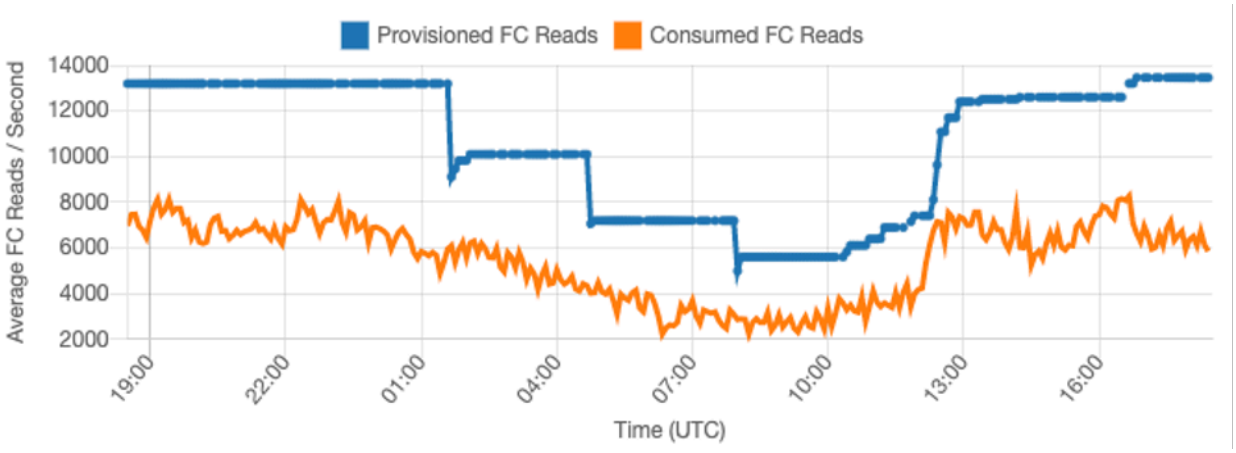
Dado que las tablas bajo demanda se facturan por solicitud, no necesita hacer nada más a nivel de tabla para optimizar costos. Debe evaluar periódicamente sus tablas bajo demanda para comprobar que la carga de trabajo siga teniendo las características mencionadas. Si la carga de trabajo se ha estabilizado, considere la posibilidad de cambiar al modo aprovisionado para mantener la optimización de costos.

Cuándo seleccionar el modo de capacidad aprovisionada

Una carga de trabajo ideal para el modo de capacidad aprovisionada es aquella con un patrón de uso más predecible como el que se muestra en el gráfico siguiente.

Los siguientes factores contribuyen a una carga de trabajo predecible:

- Tráfico predecible o cíclico para una hora o un día determinado
- Ampliaciones limitadas de corta duración



Dado que los volúmenes de tráfico en un momento o día determinados son más estables, puede fijar la capacidad aprovisionada relativamente cerca de la capacidad consumida real de la tabla. Optimizar los costos de una tabla de capacidad aprovisionada es, en última instancia, el ejercicio de conseguir que la capacidad aprovisionada (línea azul) se acerque tanto como sea posible a la capacidad consumida (línea naranja) sin aumentar los eventos `ThrottledRequests` de la tabla. El espacio entre las dos líneas es tanto capacidad desaprovechada como un seguro contra una mala experiencia del usuario debida a errores de capacidad de rendimiento insuficiente.

Amazon Keyspaces proporciona Application Auto Scaling para las tablas de capacidad aprovisionada, que equilibra de forma automática esta situación en su nombre. Puede hacer un seguimiento de la capacidad consumida a lo largo del día y configurar la capacidad aprovisionada de la tabla basándose en un puñado de variables.

Unidades de capacidad mínima

Puede fijar la capacidad mínima de una tabla para limitar la aparición de errores de capacidad de rendimiento insuficiente, pero esto no reduce el costo de la tabla. Si su tabla tiene periodos de baja utilización seguidos de un repentino pico de alta utilización, fijar el mínimo puede evitar que Application Auto Scaling fije la capacidad de la tabla demasiado baja.

Unidades de capacidad máxima

Puede establecer la capacidad máxima de una tabla para limitar el escalado de una tabla por encima de lo previsto. Considere la posibilidad de aplicar un máximo para las tablas de desarrollo o prueba, en las que no se desea realizar pruebas de carga a gran escala. Puede establecer un máximo para cualquier tabla, pero asegúrese de evaluar periódicamente este ajuste con respecto a la línea de base de la tabla cuando la utilice en producción a fin de evitar errores accidentales de capacidad de rendimiento insuficiente.

Objetivo de utilización

El establecimiento de la utilización objetivo de la tabla es el principal medio de optimización de costos para una tabla de capacidad aprovisionada. Establecer aquí un valor porcentual más bajo aumenta la medida en que la tabla se sobreaprovisiona, lo que incrementa el costo, pero reduce el riesgo de errores de capacidad de rendimiento insuficiente. Establecer aquí un valor porcentual más alto reduce la medida en que la tabla se sobreaprovisiona, lo que disminuye el costo, pero aumenta el riesgo de errores de capacidad de rendimiento insuficiente.

Factores adicionales que se deben tener en cuenta al elegir un modo de capacidad de tabla

A la hora de decidir entre los dos modos de capacidad, hay algunos factores adicionales que vale la pena considerar.

Al decidir entre los dos modos de tabla, tenga en cuenta en qué medida este descuento adicional afecta al costo de la tabla. En muchos casos, incluso una carga de trabajo relativamente impredecible puede ser más rentable ejecutarla en una tabla de capacidad aprovisionada sobreaprovisioanda con capacidad reservada.

Mejorar la previsibilidad de la carga de trabajo

En algunas situaciones, una carga de trabajo podría tener aparentemente tanto un patrón predecible como uno impredecible. Si bien esto se puede admitir con facilidad con una tabla bajo demanda, es probable que los costos sean menores si se pueden mejorar los patrones impredecibles de la carga de trabajo.

Una de las causas más comunes de estos patrones son las importaciones por lotes. Este tipo de tráfico puede superar a menudo la capacidad de base de la tabla hasta tal punto que, si se ejecutara, se producirían errores de capacidad de rendimiento insuficiente. Para mantener una carga de trabajo como esta en una tabla de capacidad aprovisionada, considere las siguientes opciones:

- Si el lote se procesa en horas programadas, puede programar un aumento de la capacidad de escalado automático de su aplicación antes de que se ejecute.
- Si el lote se procesa de forma aleatoria, considere la posibilidad de ampliar el tiempo que tarda en ejecutarse en vez de hacerlo lo más rápido posible.
- Añada un periodo de aceleración a la importación, en el que la velocidad de la importación empiece siendo pequeña pero aumente lentamente a lo largo de unos minutos hasta que Application Auto Scaling haya tenido la oportunidad de empezar a ajustar la capacidad de la tabla.

Evaluación de los ajustes de Application Auto Scaling de su tabla

En esta sección se proporciona información general sobre cómo evaluar la configuración de Application Auto Scaling en las tablas de Amazon Keyspaces. Amazon Keyspaces Application Auto Scaling es una característica que administra el rendimiento de las tablas en función del tráfico de su aplicación y de su métrica de utilización objetivo. Esto garantiza que sus tablas tengan la capacidad necesaria para sus patrones de aplicación.

El servicio Application Auto Scaling monitorea el uso actual de sus tablas y lo compara con el valor de utilización objetivo: `TargetValue`. Le notifica si ha llegado el momento de aumentar o reducir la capacidad asignada.

Temas

- Comprensión de los ajustes de Application Auto Scaling
- Cómo identificar tablas con un uso objetivo bajo (<=50 %)
- Cómo abordar las cargas de trabajo con variaciones estacionales
- Cómo abordar cargas de trabajo con picos con patrones desconocidos
- Cómo abordar las cargas de trabajo con aplicaciones enlazadas

Comprensión de los ajustes de Application Auto Scaling

Definir el valor correcto para el uso objetivo, el paso inicial y los valores finales es una actividad que requiere la participación del equipo de operaciones. Esto le permite definir adecuadamente los valores basados en el uso histórico de la aplicación, que se utilizan para activar las políticas de Application Auto Scaling. El objetivo de utilización es el porcentaje de su capacidad total que debe alcanzarse durante un periodo de tiempo antes de que se apliquen las reglas de Application Auto Scaling.

Establecer un objetivo de utilización alta (un objetivo en torno al 90 %) significa que su tráfico necesita ser superior al 90 % durante un cierto periodo antes de que se active Application Auto Scaling. No debe usar un objetivo de alta utilización a menos que la aplicación sea muy constante y no reciba picos de tráfico.

Establecer una utilización muy baja (un objetivo inferior al 50 %) significa que su aplicación necesitaría alcanzar el 50 % de la capacidad aprovisionada antes de que se active una política de Application Auto Scaling. A menos que el tráfico de la aplicación crezca a un ritmo muy agresivo, esto normalmente se traduce en capacidad no utilizada y en recursos desperdiciados.

Cómo identificar tablas con un uso objetivo bajo (<=50 %)

Puede usar AWS CLI o AWS Management Console para monitorear e identificar las `TargetValues` políticas de Auto Scaling de su aplicación en sus recursos de Amazon Keyspaces:

AWS CLI

1. Devuelva la lista completa de recursos mediante la ejecución del siguiente comando:

   ```
   aws application-autoscaling describe-scaling-policies --service-namespace
    cassandra
   ```

   Este comando devuelve la lista completa de políticas de Application Auto Scaling emitidas para cualquier recurso de Amazon Keyspaces. Si solo desea recuperar los recursos de una tabla en particular, puede añadir `–resource-id parameter`. Por ejemplo:

   ```
   aws application-autoscaling describe-scaling-policies --service-namespace
    cassandra --resource-id "keyspace/keyspace-name/table/table-name”
   ```

2. Devuelva solo las políticas de escalado automático de una tabla en particular ejecutando el siguiente comando

   ```
   aws application-autoscaling describe-scaling-policies --service-namespace
    cassandra --resource-id "keyspace/keyspace-name/table/table-name”
   ```

   A continuación se destacan los valores de las políticas de Application Auto Scaling. Debe asegurarse de que el valor objetivo sea superior al 50 % para evitar el sobreaprovisionamiento. Debería obtener un resultado similar al siguiente:

   ```
   {
   ```

```
    "ScalingPolicies": [
        {
            "PolicyARN": "arn:aws:autoscaling:<region>:<account-
id>:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-scaling-policy",
            "PolicyName": $<full-gsi-name>”,
            "ServiceNamespace": "cassandra",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:index:WriteCapacityUnits",
            "PolicyType": "TargetTrackingScaling",
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "KeyspacesWriteCapacityUtilization"
                }
            },
            "Alarms": [
                ...
            ],
            "CreationTime": "2022-03-04T16:23:48.641000+10:00"
        },
        {
            "PolicyARN": "arn:aws:autoscaling:<region>:<account-
id>:scalingPolicy:<uuid>:resource/keyspaces/table/table-name/index/<index-
name>:policyName/$<full-gsi-name>-scaling-policy",
            "PolicyName":$<full-table-name>”,
            "ServiceNamespace": "cassandra",
            "ResourceId": "keyspace/keyspace-name/table/table-name",
            "ScalableDimension": "cassandra:index:ReadCapacityUnits",
            "PolicyType": "TargetTrackingScaling",
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "CassandraReadCapacityUtilization"
                }
            },
            "Alarms": [
                ...
            ],
            "CreationTime": "2022-03-04T16:23:47.820000+10:00"
        }
    ]
}
```

AWS Management Console

1. Inicie sesión AWS Management Console y diríjase a la página del CloudWatch servicio en Cómo empezar con. AWS Management Console Seleccione lo apropiado Región de AWS si es necesario.
2. En la barra de navegación izquierda, seleccione Tables (Tablas). En la página Tables (Tablas), seleccione el Name (Nombre) de la tabla.
3. En la página Detalles de la tabla, en la pestaña Capacidad, revise la configuración de Application Auto Scaling de su tabla.

Si los valores de utilización objetivo son inferiores o iguales al 50 %, debe analizar las métricas de uso de la tabla para ver si están subaprovisionadas o sobreaprovisionadas.

Cómo abordar las cargas de trabajo con variaciones estacionales

Considere el siguiente escenario: la aplicación funciona por debajo de un valor medio mínimo la mayor parte del tiempo, pero el objetivo de uso es bajo, por lo que la aplicación puede reaccionar rápidamente ante los eventos que se producen a determinadas horas del día y usted tiene suficiente capacidad y evita que se limite. Este escenario es habitual cuando una aplicación está muy ocupada durante el horario de oficina normal (de 9:00 a 17:00), pero que luego funciona a un nivel básico fuera del horario laboral. Dado que algunos usuarios comienzan a conectarse antes de las 09:00, la aplicación utiliza este umbral bajo para aumentar rápidamente su capacidad hasta alcanzar la capacidad requerida durante las horas punta.

Este escenario podría ser así:

- Entre las 17:00 y las 9:00, las unidades de `ConsumedWriteCapacityUnits` permanecen entre 90 y 100
- Los usuarios comienzan a conectarse a la aplicación antes de las 9:00 y la capacidad de las unidades aumenta considerablemente (el valor máximo que ha visto es de 1500 WCU)
- Por término medio, el uso de la aplicación varía entre 800 y 1200 durante las horas de trabajo

Si el escenario anterior se aplica a su aplicación, considere el uso de escalado automático de aplicaciones programado, en el que su tabla podría seguir teniendo configurada una regla de Application Auto Scaling, pero con una utilización objetivo menos agresiva que solo aprovisione la capacidad extra en los intervalos específicos que usted requiera.

Puede usar el AWS CLI para ejecutar los siguientes pasos a fin de crear una regla de autoescalado programada que se ejecute en función de la hora del día y el día de la semana.

1. Registre su tabla de Amazon Keyspaces como objetivo de escalado con Application Auto Scaling. Un destino escalable es un recurso que Application Auto Scaling puede escalar horizontalmente o escalar verticalmente.

```
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. Configure acciones programadas según los requisitos.

   Necesita dos reglas para cubrir el escenario: una para escalar verticalmente y otra para reducir verticalmente. La primera regla para escalar verticalmente la acción programada se muestra en el siguiente ejemplo.

```
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"
```

   La segunda regla para reducir verticalmente la acción programada se muestra en este ejemplo.

```
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"
```

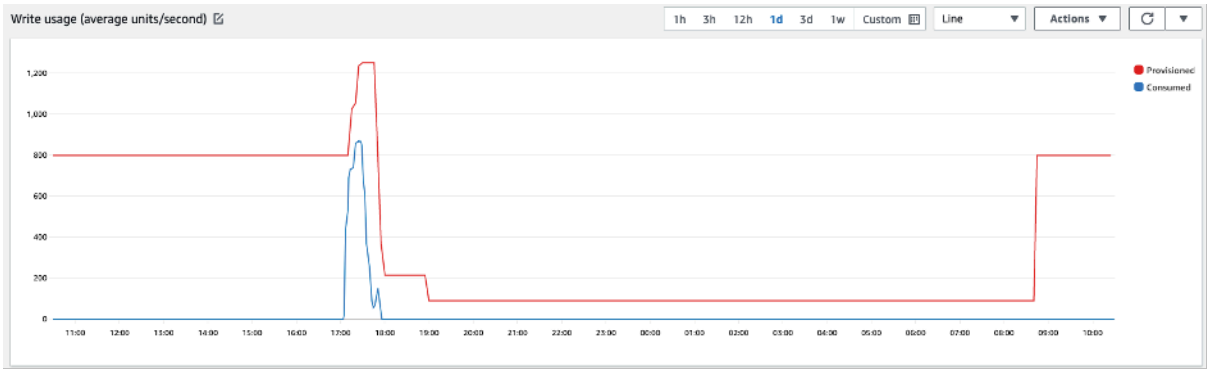
3. Ejecute el siguiente comando para validar que ambas reglas se han activado:

```
aws application-autoscaling describe-scheduled-actions --service-namespace
 cassandra
```

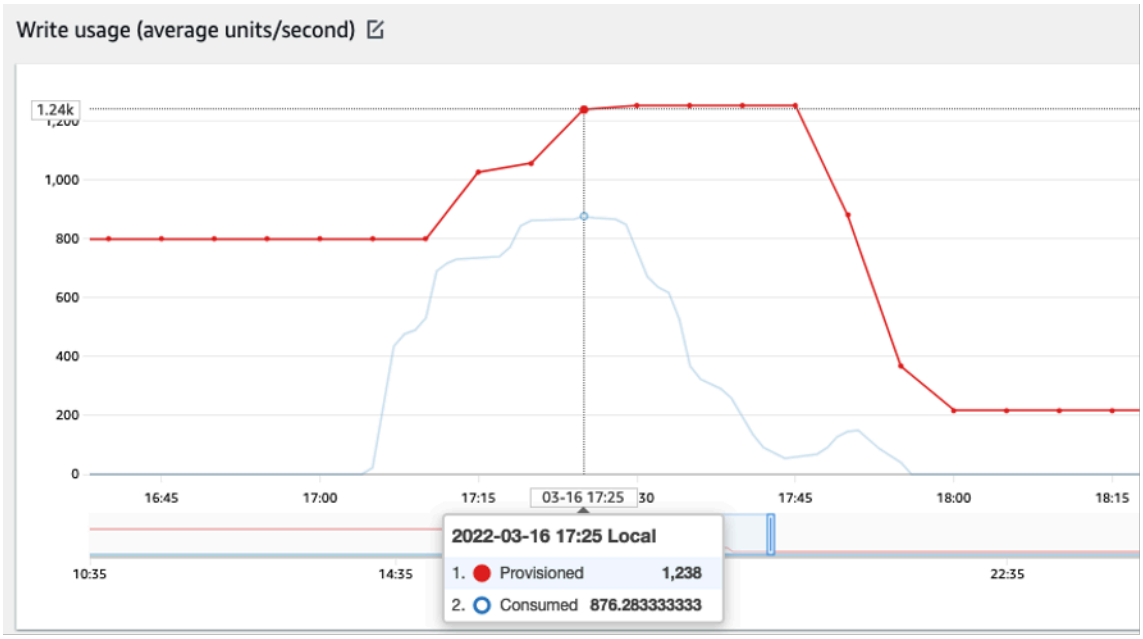
Debería obtener un resultado como este:

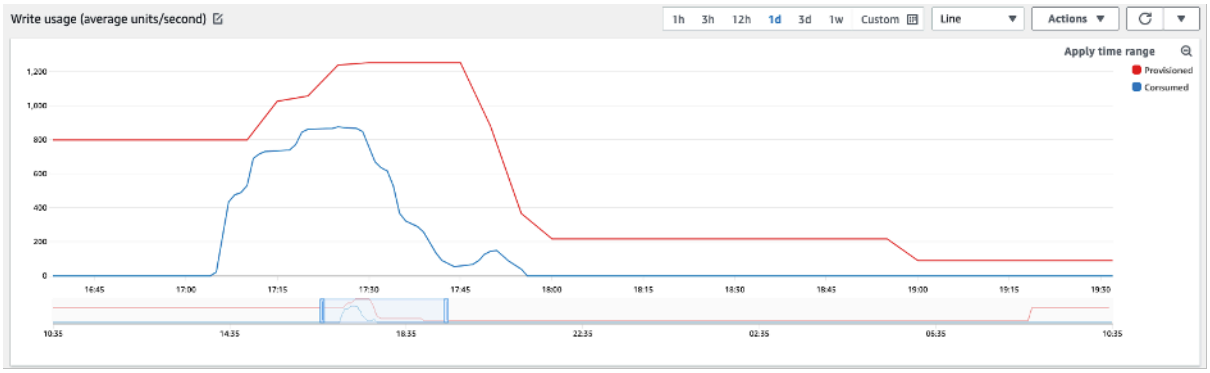
```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-5-8-scheduled-down-action",
            "ScheduledActionARN":
 "arn:aws:autoscaling:<region>:<account>:scheduledAction:<uuid>:resource/keyspaces/
table/table-name:scheduledActionName/my-5-8-scheduled-down-action",
            "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:<region>:<account>: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"
        }
    ]
}
```

La siguiente imagen muestra un ejemplo de carga de trabajo que siempre mantiene el objetivo de utilización del 70 %. Observe cómo las reglas de escalado automático siguen aplicándose y el rendimiento no se reduce.



Al ampliar el zoom, podemos ver que hubo un aumento en la aplicación que desencadenó el umbral de escalado automático del 70 %, lo que obligó a que el escalado automático entrara en vigor y proporcionara la capacidad adicional necesaria para la tabla. La acción de escalado automático programada afectará a los valores máximo y mínimo y es su responsabilidad configurarlos.





Cómo abordar cargas de trabajo con picos con patrones desconocidos

En este escenario, la aplicación utiliza un objetivo de utilización muy bajo, porque aún no conoce los patrones de la aplicación y quiere asegurarse de que su carga de trabajo no experimenta errores de rendimiento de baja capacidad.

Considere utilizar el modo de capacidad bajo demanda en su lugar. Las tablas bajo demanda son perfectas para cargas de trabajo con picos en los que no se conocen los patrones de tráfico. Con el modo de capacidad bajo demanda, usted paga por solicitud por las lecturas y escrituras de datos que la aplicación realiza en las tablas. No necesita especificar el rendimiento de lectura y escritura que espera de su aplicación, dado que Amazon Keyspaces se adapta instantáneamente a sus cargas de trabajo a medida que aumentan o disminuyen.

Cómo abordar las cargas de trabajo con aplicaciones enlazadas

En este escenario, la aplicación depende de otros sistemas, como los escenarios de procesamiento por lotes, en los que se pueden producir grandes picos de tráfico en función de los eventos de la lógica de la aplicación.

Considere la posibilidad de desarrollar una lógica de escalado automático de la aplicación personalizada que reaccione a esos eventos en los que puede aumentar la capacidad de la tabla y `TargetValues` en función de sus necesidades específicas. Podría beneficiarse de una combinación de AWS servicios como λ Amazon EventBridge y Step Functions y utilizarlos para responder a las necesidades específicas de sus aplicaciones.

Identificación de recursos no utilizados

En esta sección se ofrece una visión general de cómo evaluar periódicamente sus recursos sin utilizar. A medida que evolucionan los requisitos de su aplicación, debe asegurarse de que no quedan recursos sin utilizar que contribuyan a generar costos innecesarios de Amazon Keyspaces.

Los procedimientos que se describen a continuación utilizan CloudWatch las métricas de Amazon para identificar los recursos no utilizados y tomar medidas para reducir los costes.

Puede monitorizar Amazon Keyspaces utilizando CloudWatch, que recopila y procesa datos sin procesar de Amazon Keyspaces para convertirlos en métricas legibles y prácticamente en tiempo real. Estas estadísticas se conservan durante un periodo de tiempo, de modo que pueda acceder a la información histórica para comprender mejor su utilización. De forma predeterminada, los datos de las métricas de Amazon Keyspaces se envían automáticamente a CloudWatch . Para obtener más información, consulta ¿Qué es Amazon CloudWatch? y retención de métricas en la Guía del CloudWatch usuario de Amazon.

Temas

- Cómo identificar los recursos sin utilizar
- Identificación de recursos de tabla sin utilizar
- Limpieza de los recursos de tabla no utilizados
- Limpiar las copias de seguridad point-in-time de recuperación no utilizadas (PITR)

Cómo identificar los recursos sin utilizar

Para identificar las tablas no utilizadas, puedes echar un vistazo a las siguientes CloudWatch métricas durante un período de 30 días para saber si hay alguna lectura o escritura activa en una tabla específica:

`ConsumedReadCapacityUnits`

Número de unidades de capacidad de lectura consumidas durante el periodo de tiempo especificado, para que pueda hacer un seguimiento de la capacidad consumida. Puede recuperar la capacidad de lectura total consumida de una tabla.

`ConsumedWriteCapacityUnits`

Número de unidades de capacidad de escritura consumidas durante el periodo de tiempo especificado, para que pueda hacer un seguimiento de la capacidad consumida. Puede recuperar la capacidad de escritura total consumida para una tabla.

Identificación de recursos de tabla sin utilizar

Amazon CloudWatch es un servicio de supervisión y observabilidad que proporciona las métricas de la tabla Amazon Keyspaces que puede utilizar para identificar los recursos no utilizados. CloudWatch

las métricas se pueden ver tanto a través de AWS Management Console como a través de. AWS Command Line Interface

AWS Command Line Interface

Para ver las métricas de sus tablas a través de AWS Command Line Interface, puede utilizar los siguientes comandos.

1. En primer lugar, evalúe las lecturas de su tabla:

> **Note**
>
> Si el nombre de la tabla no es único dentro de su cuenta, deberá especificar también el nombre del espacio de claves.

```
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>
```

Para evitar la identificación falsa de una tabla como no utilizada, evalúe las métricas durante un periodo más largo. Elija un intervalo de tiempo de inicio y fin apropiado, como 30 días, y un periodo apropiado, como 86400.

En los datos devueltos, cualquier suma superior a 0 indica que la tabla que está evaluando recibió tráfico de lectura durante ese periodo.

En el siguiente resultado se muestra una tabla que recibe tráfico de lectura en el periodo evaluado:

```
        {
            "Timestamp": "2022-08-25T19:40:00Z",
            "Sum": 36023355.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-12T19:40:00Z",
            "Sum": 38025777.5,
            "Unit": "Count"
```

```
        },
```

En el siguiente resultado se muestra una tabla que no recibe tráfico de lectura en el periodo evaluado:

```
        {
            "Timestamp": "2022-08-01T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-20T19:50:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

2. A continuación, evalúe las escrituras de su tabla:

```
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>
```

Para evitar la identificación falsa de una tabla como no utilizada, es recomendable evaluar las métricas durante un periodo más largo. Elija un intervalo de tiempo de inicio y de finalización apropiado, como 30 días, y un periodo apropiado, como 86 400.

En los datos devueltos, cualquier suma superior a 0 indica que la tabla que está evaluando recibió tráfico de lectura durante ese periodo.

En el siguiente resultado se muestra una tabla que recibe el tráfico de escritura en el periodo evaluado:

```
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 41014457.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-18T20:15:00Z",
            "Sum": 40048531.0,
```

```
            "Unit": "Count"
        },
```

En el siguiente resultado se muestra una tabla que no recibe tráfico de escritura en el periodo evaluado:

```
        {
            "Timestamp": "2022-07-31T20:15:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2022-08-19T20:15:00Z",
            "Sum": 0.0,
            "Unit": "Count"
        },
```

AWS Management Console

Los pasos siguientes le permiten evaluar la utilización de sus recursos a través de la AWS Management Console.

1. Inicie sesión AWS Management Console y navegue hasta la página CloudWatch de servicio en https://console.aws.amazon.com/cloudwatch/. Si es necesario, selecciona la opción correspondiente Región de AWS en la parte superior derecha de la consola.
2. En la barra de navegación izquierda, localice la sección Métricas y elija Todas las métricas.
3. La acción anterior abre un panel de control con dos paneles. En el panel superior, puede ver las métricas graficadas actualmente. En la parte inferior puede seleccionar las métricas disponibles para graficar. Elija Amazon Keyspaces en el panel inferior.
4. En el panel de selección de métricas de Amazon Keyspaces, elija la categoría Métricas de tabla para mostrar las métricas de sus tablas en la región actual.
5. Identifique el nombre de su tabla desplazándose por el menú y, a continuación, elija las métricas `ConsumedReadCapacityUnits` y `ConsumedWriteCapacityUnits` de su tabla.
6. Elija la pestaña Métricas gráficas (2) y ajuste la columna Estadística a Suma.
7. Para evitar una falsa identificación de una tabla como no utilizada, evalúe las métricas de la tabla durante un periodo más largo. En la parte superior del panel de gráficos, elija un

periodo de tiempo apropiado, como 1 mes, para evaluar su tabla. Elija Personalizar, luego 1 Mes en el menú desplegable y finalmente Aplicar.

8. Evalúe la métrica diagramada de su tabla para determinar si se está utilizando. Las métricas que han subido por encima de 0 indican que se ha utilizado una tabla durante el periodo de tiempo evaluado. Un gráfico plano en 0 tanto para lectura como para escritura indica que la tabla no se utiliza.

Limpieza de los recursos de tabla no utilizados

Si ha identificado los recursos de tabla no utilizados, puede reducir sus costos continuos de las siguientes maneras.

> **Note**
>
> Si ha identificado una tabla que no se utiliza, pero quiere mantenerla disponible por si es necesario acceder a ella en el futuro, considere la posibilidad de cambiarla al modo bajo demanda. Caso contrario, puede plantearse la posibilidad de eliminar la tabla.

Modos de capacidad

Amazon Keyspaces cobra por leer, escribir y almacenar datos en sus tablas de Amazon Keyspaces.

Amazon Keyspaces dispone de dos modos de capacidad, que tienen opciones de facturación específicas para el procesamiento de lecturas y escrituras en sus tablas: bajo demanda y aprovisionada. El modo de capacidad de lectura/escritura controla cómo se le cobrará el rendimiento de lectura y escritura y cómo se administra la capacidad.

Para tablas en modo en diferido, no necesita especificar el rendimiento de lectura y escritura que espera de su aplicación. Amazon Keyspaces le cobra por las lecturas y escrituras que su aplicación realiza en sus tablas en términos de unidades de solicitud de lectura y unidades de solicitud de escritura. Si no hay actividad en su tabla, no paga por rendimiento, pero sigue incurriendo en un cargo por almacenamiento.

Eliminación de tablas

Si ha descubierto una tabla sin utilizar y desea eliminarla, considere la posibilidad de hacer primero una copia de seguridad o exportar los datos.

Las copias de seguridad realizadas AWS Backup pueden aprovechar la organización en niveles del almacenamiento en frío, lo que reduce aún más los costos. Consulte la documentación de Administración de planes de copias de seguridad para obtener información sobre cómo utilizar un ciclo de vida para trasladar la copia de seguridad al almacenamiento en frío.

Una vez realizada la copia de seguridad de la tabla, puede eliminarla a través de la AWS Management Console o la AWS Command Line Interface.

Limpiar las copias de seguridad point-in-time de recuperación no utilizadas (PITR)

Amazon Keyspaces ofrece la oint-in-time recuperación de P, que proporciona copias de seguridad continuas durante 35 días para ayudarlo a protegerse contra escrituras o eliminaciones accidentales. Las copias de seguridad de PITR tienen costos asociados.

Consulte la documentación en Recuperación a un momento dado para determinar si sus tablas tienen habilitadas copias de seguridad que tal vez ya no sean necesarias.

Evaluar los patrones de uso de las tablas

En esta sección se ofrece información general sobre cómo evaluar si sus tablas de Amazon Keyspaces se utilizan de forma eficiente. Existen determinados patrones de uso que no son óptimos para Amazon Keyspaces y que permiten un margen de optimización tanto desde el punto de vista de rendimiento como de costo.

Temas

- Realizar menos operaciones de lectura altamente coherente
- Habilitar el Periodo de vida (TTL)

Realizar menos operaciones de lectura altamente coherente

Amazon Keyspaces le permite configurar la coherencia de lectura en función de cada solicitud. Las solicitudes de lectura tienen coherencia final de forma predeterminada. Las lecturas coherentes posteriores se cobran a 0,5 RCU para hasta 4 KB de datos.

La mayoría de las partes de las cargas de trabajo distribuidas son flexibles y pueden tolerar una coherencia eventual. Sin embargo, puede haber patrones de acceso que requieran lecturas altamente coherentes. Las lecturas altamente coherentes se cobran a 1 RCU para hasta 4 KB de datos, lo que básicamente duplica sus costos de lectura. Amazon Keyspaces le ofrece la flexibilidad de utilizar ambos modelos de coherencia en la misma tabla.

Puede evaluar la carga de trabajo y el código de la aplicación para confirmar si solo se utilizan lecturas altamente coherentes cuando sea necesario.

Habilitar el Periodo de vida (TTL)

Periodo de vida (TTL) le ayuda a simplificar la lógica de su aplicación y a optimizar el precio de almacenamiento al expirar automáticamente los datos de las tablas. Los datos que ya no necesite se eliminan automáticamente de su tabla en función del valor de tiempo de vida que establezca.

Evaluar la capacidad aprovisionada para lograr un aprovisionamiento del tamaño adecuado

En esta sección se ofrece información general sobre cómo evaluar si dispone de un aprovisionamiento de tamaño adecuado en sus tablas de Amazon Keyspaces. A medida que evolucione la carga de trabajo, debe modificar los procedimientos operativos de manera adecuada, en particular si la tabla de Amazon Keyspaces está configurada en modo aprovisionado y corre el riesgo de sobreaprovisionar o subaprovisionar sus tablas.

Los procedimientos que se describen en esta sección requieren información estadística que se debería capturar de tablas de Amazon Keyspaces que sean compatibles con la aplicación de producción. Para entender el comportamiento de la aplicación, debe definir un periodo de tiempo que sea lo suficientemente significativo como para captar la estacionalidad de los datos de su aplicación. Por ejemplo, si la aplicación muestra patrones semanales, utilizar un periodo de tres semanas debería darle suficiente espacio para analizar las necesidades de rendimiento de la aplicación.

Si no sabe por dónde empezar, utilice al menos un mes de uso de datos para los cálculos que se indican a continuación.

Al evaluar la capacidad, puede configurar para las tablas de Amazon Keyspaces Unidades de capacidad de lectura (RCU) y Unidades de capacidad de escritura (WCU) de forma independiente.

Temas

- Recuperación de métricas de consumo de sus tablas de Amazon Keyspaces
- Identificación de tablas de DynamoDB subaprovisionadas
- Identificación de tablas de DynamoDB sobreaprovisionadas

Recuperación de métricas de consumo de sus tablas de Amazon Keyspaces

Para evaluar la capacidad de la tabla, supervise las siguientes CloudWatch métricas y seleccione la dimensión adecuada para recuperar la información de la tabla:

Unidades de capacidad de lectura	Unidades de capacidad de escritura
`ConsumedReadCapacityUnits`	`ConsumedWriteCapacityUnits`
`ProvisionedReadCapacityUnits`	`ProvisionedWriteCapacityUnits`
`ReadThrottleEvents`	`WriteThrottleEvents`

Puede hacerlo mediante el AWS CLI o el AWS Management Console.

AWS CLI

Antes de recuperar las métricas de consumo de la tabla, debes empezar por capturar algunos puntos de datos históricos mediante la CloudWatch API.

Comience por crear dos archivos: `write-calc.json` y `read-calc.json`. Estos archivos representan los cálculos de la tabla. Debe actualizar algunos de los campos, como se indica en la tabla siguiente, para que coincidan con su entorno.

> **Note**
>
> Si el nombre de la tabla no es único dentro de su cuenta, deberá especificar también el nombre del espacio de claves.

Nombre del campo	Definición	Ejemplo
`<table-name>`	El nombre de la tabla que vaya a analizar	SampleTable
`<period>`	El periodo que utilizará para evaluar el objetivo de utilización, en segundos	Para un periodo de 1 hora, debe especificar: 3600

Nombre del campo	Definición	Ejemplo
`<start-time>`	El comienzo del intervalo de evaluación, especificado en formato ISO8601	2022-02-21T23:00:00
`<end-time>`	El final del intervalo de evaluación, especificado en formato ISO8601	2022-02-22T06:00:00

El archivo de cálculos de escritura recupera el número de WCU aprovisionadas y consumidas en el periodo de tiempo para el intervalo de fechas especificado. También genera un porcentaje de utilización que puede utilizarse para el análisis. El contenido completo del archivo `write-calc.json` debería asemejarse al del siguiente ejemplo.

```
{
  "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
}
```

El archivo de cálculos de lectura utiliza una métrica similar. Este archivo recupera cuántas RCU fueron aprovisionadas y consumidas durante el periodo de tiempo para el intervalo de fechas especificado. El contenido del archivo `read-calc.json` debería asemejarse al del siguiente ejemplo.

```
{
  "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
}
```

Una vez creados los archivos, puede empezar a recuperar los datos de utilización.

1. Para recuperar los datos de utilización de escritura, ejecute el siguiente comando:

```
aws cloudwatch get-metric-data --cli-input-json file://write-calc.json
```

2. Para recuperar los datos de utilización de lectura, ejecute el siguiente comando:

```
aws cloudwatch get-metric-data --cli-input-json file://read-calc.json
```

El resultado de ambas consultas es una serie de puntos de datos en formato JSON que pueden utilizarse para el análisis. Sus resultados dependen del número de puntos de datos que haya especificado, del periodo y de sus propios datos específicos de carga de trabajo. Podría asemejarse al ejemplo siguiente.

```
{
    "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**
>
> Si especifica un periodo corto y un intervalo de tiempo largo, es posible que tenga que modificar el valor `MaxDatapoints`, que de forma predeterminada está fijado en 24 en el script. Este representa un punto de datos por hora y 24 por día.

AWS Management Console

1. Inicie sesión AWS Management Console y navegue hasta la página del CloudWatch servicio en Cómo empezar con AWS Management Console. Seleccione lo apropiado Región de AWS si es necesario.
2. Localice la sección Métricas en la barra de navegación izquierda y elija Todas las métricas.
3. Esto abre un panel de control con dos paneles. El panel superior le muestra el gráfico y el panel inferior tiene las métricas que desea graficar. Elija el panel Amazon Keyspaces.
4. Elija la categoría Métricas de tabla en los subpaneles. Esto le mostrará las tablas actuales Región de AWS.
5. Identifique el nombre de la tabla desplazándose hacia abajo en el menú y seleccionando las métricas de operaciones de escritura: `ConsumedWriteCapacityUnits` y `ProvisionedWriteCapacityUnits`

> **Note**
>
> En este ejemplo se habla de las métricas de las operaciones de escritura, pero también puede utilizar estos pasos para representar gráficamente las métricas de las operaciones de lectura.

6. Seleccione la pestaña Graphed metrics (2) (Métricas gráficas [2]) para modificar las fórmulas. De forma predeterminada, CloudWatch elige la función estadística Promedio para los gráficos.
7. Con ambas métricas gráficas seleccionadas (la casilla de verificación de la izquierda), seleccione el menú Add math (Agregar matemáticas), seguido de Common (Común) y, a continuación, seleccione la función Percentage (Porcentaje). Repita el procedimiento dos veces.

   Primera vez al seleccionar la función Porcentaje.

   Segunda vez al seleccionar la función Porcentaje.
8. En este punto, debería tener cuatro métricas en el menú inferior. Vamos a trabajar en el cálculo de `ConsumedWriteCapacityUnits`. Para ser coherente, debe hacer coincidir los nombres con los que utilizó en la AWS CLI sección. Haga clic en ID de m1 y cambie este valor a consumedWCU.
9. Cambie la estadística de Average (Media) a Sum (Suma). Esta acción crea automáticamente otra métrica denominada ANOMALY_DETECTION_BAND. En el ámbito de este procedimiento, puede ignorar esto desmarcando la casilla de verificación en la Métrica ad1 recién generada.
10. Repita el paso 8 para cambiar el nombre de ID de m2 a provisionedWCU. Deje la estadística establecida en Average (Media).
11. Elija la etiqueta Expression1 y actualice el valor a m1 y la etiqueta a WCU consumidas.

> **Note**
>
> Asegúrese de haber seleccionado solo m1 (casilla de verificación de la izquierda) y provisionedWCU para visualizar correctamente los datos. Actualice la fórmula haciendo clic en Details (Detalles) y cambiándola a consumedWCU/PERIOD(consumedWCU). Es posible que este paso también genere otra métrica

ANOMALY_DETECTION_BAND, pero para el ámbito de este procedimiento puede ignorarla.

12. Ahora debería tener dos gráficos: uno que indique sus WCU aprovisionadas en la tabla y otro que indique las WCU consumidas.
13. Actualice la fórmula porcentual seleccionando el gráfico Expression2 (e2). Cambie el nombre de las etiquetas e ID a utilizationPercentage. Cambie el nombre de la fórmula para que coincida con 100*(m1/provisionedWCU).
14. Desmarque la casilla de verificación de todas las métricas excepto utilizationPercentage para visualizar sus patrones de utilización. El intervalo predeterminado es de 1 minuto, pero siéntase libre de modificarlo según sus necesidades.

Los resultados que obtiene dependen de los datos reales de su carga de trabajo. Los intervalos con una utilización superior al 100 % son propensos a sufrir eventos de error de baja capacidad de rendimiento. Amazon Keyspaces ofrece capacidad de ampliación, pero en cuanto esta se agote, todo lo que supere el 100 % experimenta eventos de error de baja capacidad de rendimiento.

Identificación de tablas de DynamoDB subaprovisionadas

Para la mayoría de las cargas de trabajo, una tabla se considera subaprovisionada cuando consume de forma continua más del 80 % de la capacidad aprovisionada.

La capacidad de ampliación es una característica de Amazon Keyspaces que permite a los clientes consumir temporalmente más RCU/WCU de las aprovisionadas originalmente (más que el rendimiento aprovisionado por segundo que se definió en la tabla). La capacidad de ampliación se creó para absorber los aumentos repentinos del tráfico debido a eventos especiales o picos de uso. Esta capacidad de ampliación es limitada. Para obtener más información, consulte the section called “Capacidad de ampliación”. En cuanto se agoten las RCU y WCU no utilizadas, puede experimentar eventos de error de rendimiento de baja capacidad si intenta consumir más capacidad de la aprovisionada. Cuando el tráfico de su aplicación se acerque a la tasa de utilización del 80 %, el riesgo de experimentar eventos de error de baja capacidad de rendimiento será significativamente mayor.

La regla de la tasa de utilización del 80 % varía según la estacionalidad de los datos y el crecimiento del tráfico. Considere los siguientes escenarios:

- Si el tráfico se ha mantenido estable a una tasa de utilización de aproximadamente el 90 % durante los últimos 12 meses, la tabla tiene la capacidad adecuada
- Si el tráfico de la aplicación crece a un ritmo del 8 % mensual en menos de 3 meses, llegará al 100 %
- Si el tráfico de la aplicación crece a un ritmo del 5 % mensual en un poco más de 4 meses, llegará al 100 %

Los resultados de las consultas anteriores proporcionan una imagen de la tasa de utilización. Úselos como guía para evaluar con más detalle otras métricas que pueden ayudarle a aumentar la capacidad de la tabla según sea necesario (por ejemplo, una tasa de crecimiento mensual o semanal). Trabaje con el equipo de operaciones para definir cuál es un buen porcentaje para la carga de trabajo y las tablas.

Existen escenarios especiales en los que los datos se sesgan al analizarlos a diario o semanalmente. Por ejemplo, con aplicaciones estacionales que tienen picos de utilización durante las horas de trabajo (pero luego caen casi a cero fuera de las horas de trabajo), podría beneficiarse de programar el escalado automático de la aplicación, donde usted especifica las horas del día (y los días de la semana) para aumentar la capacidad aprovisionada, así como cuándo reducirla. En vez de optar por una mayor capacidad para poder cubrir las horas de mayor actividad, también podría beneficiarse de las configuraciones de escalado automático de tablas de Amazon Keyspaces si su estacionalidad es menos pronunciada.

Identificación de tablas de DynamoDB sobreaprovisionadas

Los resultados de la consulta obtenidos de los scripts anteriores proporcionan los puntos de datos necesarios para realizar algunos análisis iniciales. Si el conjunto de datos presenta valores de utilización inferiores al 20 % durante varios intervalos, es posible que la tabla tenga sobreaprovisionamiento. Para definir con más detalle si necesita reducir el número de WCU y RCU, debe revisitar las demás lecturas en los intervalos.

Si su tabla contiene varios intervalos de baja utilización, puede beneficiarse del uso de políticas de Application Auto Scaling, ya sea programando Application Auto Scaling o simplemente configurando las políticas de Application Auto Scaling predeterminadas para la tabla que se basen en la utilización.

Si tiene una carga de trabajo con un índice de utilización bajo y una relación de aceleración alta (máx. (ThrottleEvents) /minuto (ThrottleEvents) en el intervalo), esto podría ocurrir si tiene una carga de trabajo muy intensa, en la que el tráfico aumenta considerablemente en días (o horas del día)

específicos, pero por lo demás es constantemente bajo. En estos escenarios, podría ser beneficioso utilizar Application Auto Scaling programado.

Uso de NoSQL Workbench con Amazon Keyspaces (para Apache Cassandra)

NoSQL Workbench es una aplicación del lado del cliente que le ayuda a diseñar y visualizar modelos de datos no relacionales para Amazon Keyspaces con mayor facilidad. Los clientes de NoSQL Workbench están disponibles para Windows, macOS y Linux.

Diseño de modelos de datos y creación de recursos de forma automática

NoSQL Workbench le proporciona una interfaz de apuntar y hacer clic para diseñar y crear modelos de datos de Amazon Keyspaces. Puede crear fácilmente nuevos modelos de datos desde cero definiendo espacios de claves, tablas y columnas. También puede importar modelos de datos existentes y realizar modificaciones (como añadir, editar o eliminar columnas) para adaptar los modelos de datos a nuevas aplicaciones. NoSQL Workbench le permite luego enviar los modelos de datos a Amazon Keyspaces o Apache Cassandra, y crear los espacios de claves y las tablas automáticamente. Para obtener información sobre cómo crear modelos de datos, consulte the section called “Modelador de datos”.

Visualización de modelos de datos

Con NoSQL Workbench, puede visualizar sus modelos de datos para asegurarse de que estos admitan las consultas y los patrones de acceso de su aplicación. También puede guardar y exportar sus modelos de datos en diversos formatos para colaboración, documentación y presentaciones. Para obtener más información, consulte the section called “Visualizador de datos”.

Temas

- Descargar NoSQL Workbench
- Introducción a NoSQL Workbench
- Creación de modelos de datos
- Visualización de modelos de datos
- Confirmación de modelos de datos en Amazon Keyspaces y Apache Cassandra
- Modelos de datos de ejemplo en NoSQL Workbench
- Historial de versiones de NoSQL Workbench

Descargar NoSQL Workbench

Siga estas instrucciones para descargar e instalar NoSQL Workbench.

Para descargar e instalar NoSQL Workbench

1. Utilice uno de los siguientes enlaces para descargar NoSQL Workbench de forma gratuita.

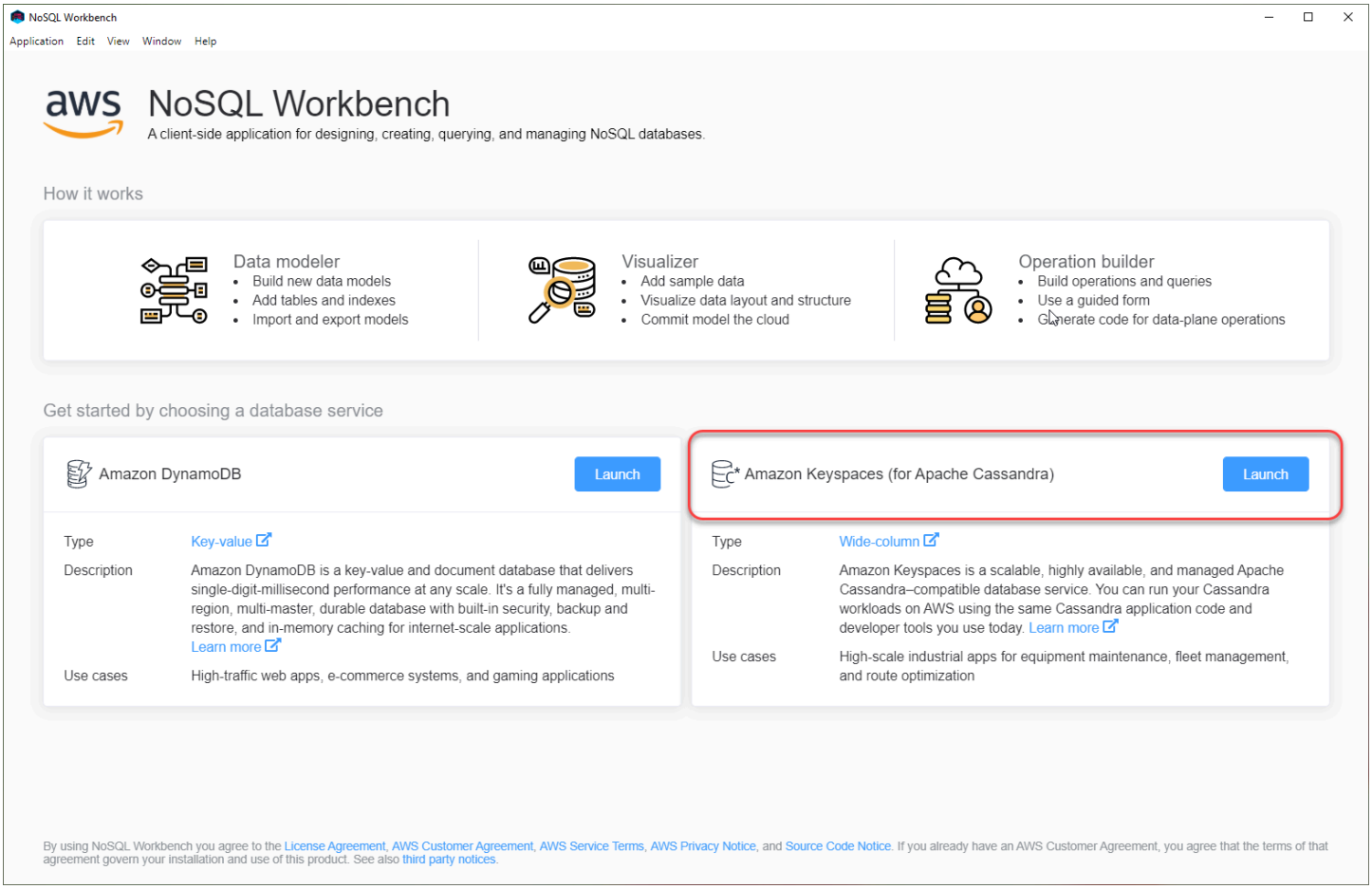
Sistema operativo	Enlace de descarga
macOS	Descargar para macOS
Linux*	Descargar para Linux
Windows	Descargar para Windows

   * NoSQL Workbench es compatible con Ubuntu 12.04, Fedora 21 y Debian 8 o cualquier versión más reciente de estas distribuciones de Linux.

2. Una vez finalizada la descarga, inicie la aplicación y siga las instrucciones en pantalla para completar la instalación.

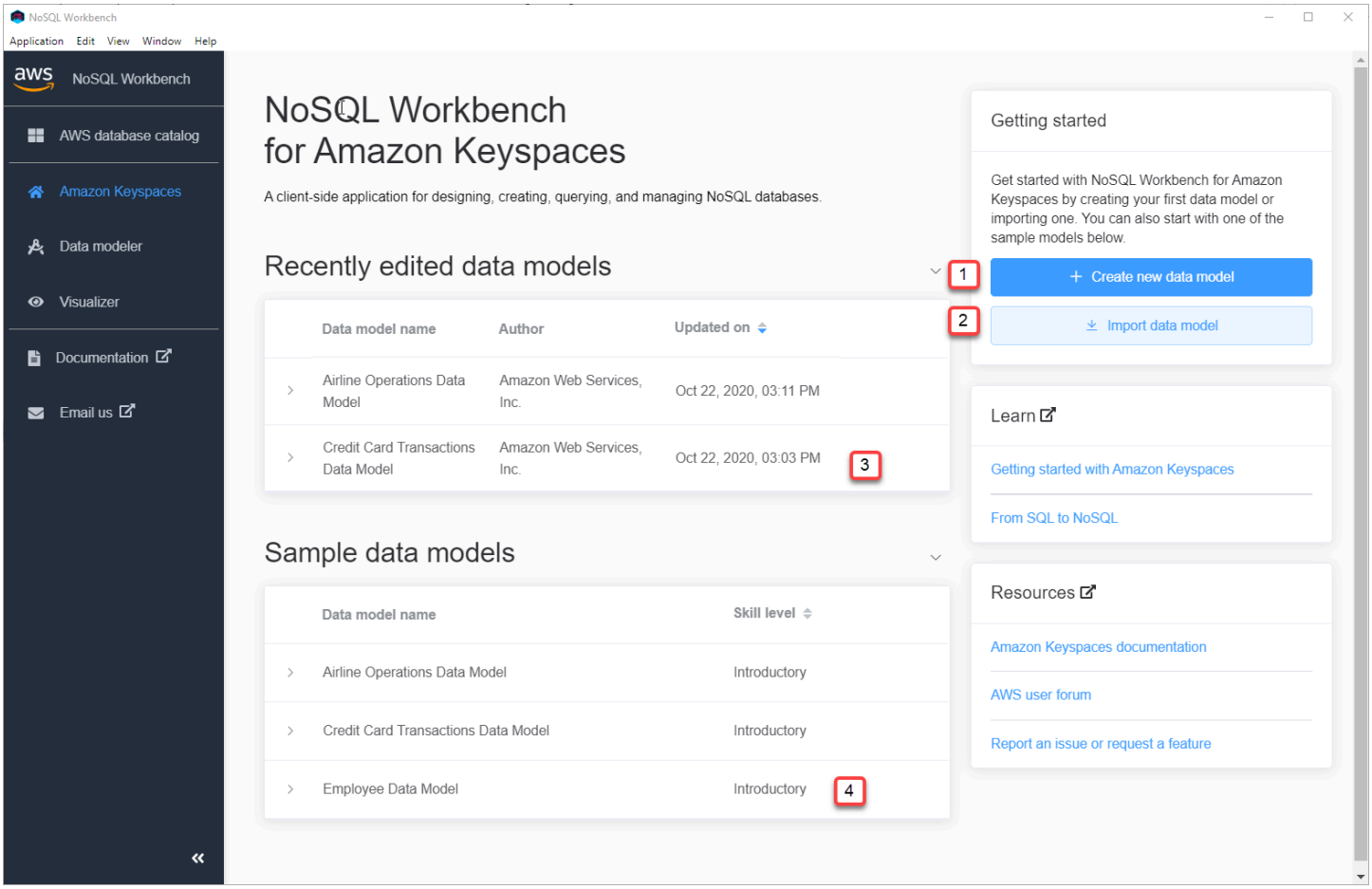
Introducción a NoSQL Workbench

Para comenzar a utilizar NoSQL Workbench, en la página Catálogo de bases de datos de NoSQL Workbench, elija Amazon Keyspaces y, a continuación, Lanzar.



Esto abre la página de inicio de NoSQL Workbench para Amazon Keyspaces, donde dispone de las siguientes opciones para comenzar:

1. Crear un nuevo modelo de datos.
2. Importar un modelo de datos existente en formato JSON.
3. Abrir un modelo de datos editado recientemente.
4. Abrir uno de los modelos de ejemplo disponibles.



Cada una de las opciones abre el modelador de datos de NoSQL Workbench. Para continuar creando un nuevo modelo de datos, consulte the section called “Creación de un modelo de datos”. Para editar un modelo de datos existente, consulte the section called “Edición de un modelo de datos”.

Creación de modelos de datos

Puede utilizar el modelador de datos de NoSQL Workbench para diseñar nuevos modelos de datos basados en los patrones de acceso a datos de su aplicación. Puede utilizar el modelador de datos para diseñar nuevos modelos de datos o importar y modificar modelos de datos existentes creados con NoSQL Workbench. El modelador de datos también incluye algunos modelos de datos de muestra que le ayudarán a iniciarse en el modelado de datos.

Temas

- Creación de nuevos modelos de datos con NoSQL Workbench
- Edición de modelos de datos existentes con NoSQL Workbench

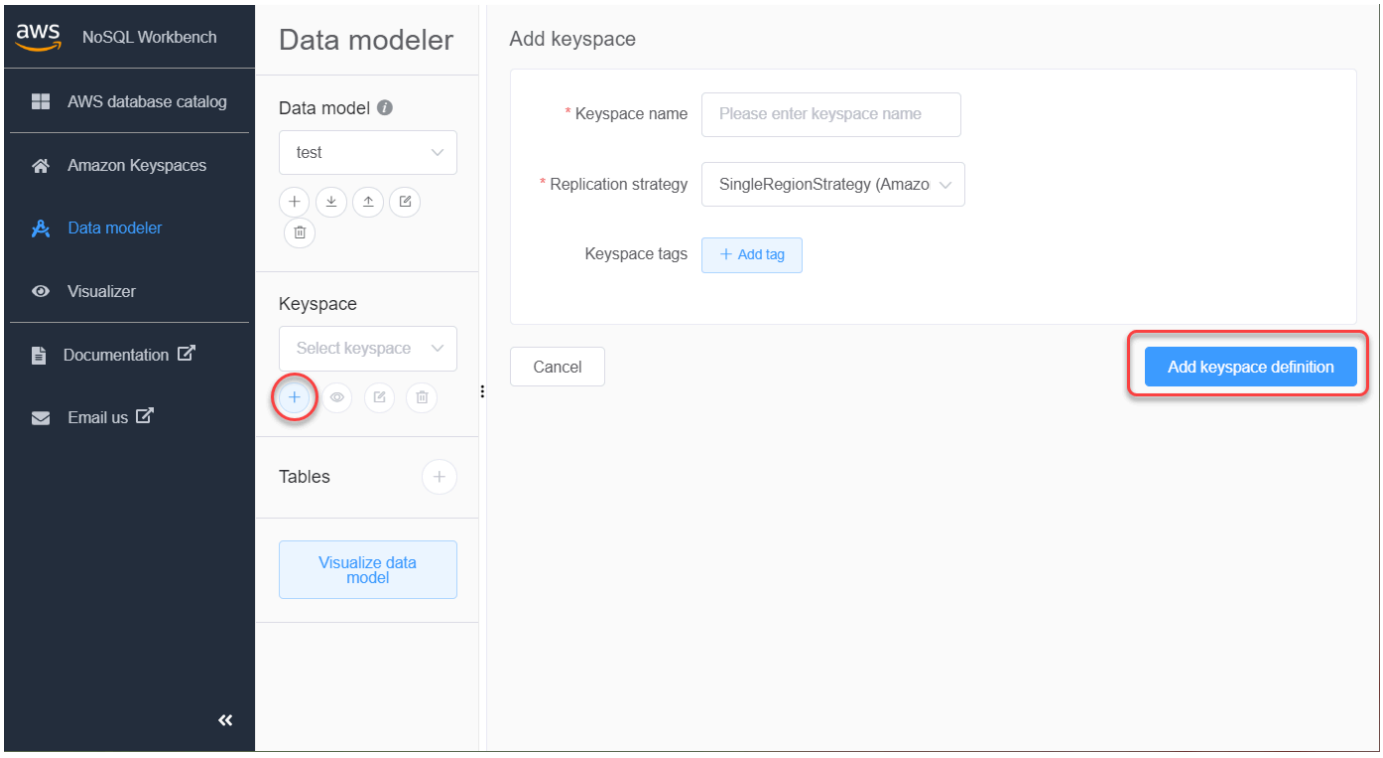
Creación de nuevos modelos de datos con NoSQL Workbench

Para crear un nuevo modelo de datos para Amazon Keyspaces, puede utilizar el modelador de datos de NoSQL Workbench para crear espacios de claves, tablas y columnas. Siga estos pasos para crear un nuevo modelo de datos.

1. Para crear un nuevo espacio de claves, elija el signo más en Espacio de claves.

   En este paso, elija las siguientes propiedades y configuraciones.

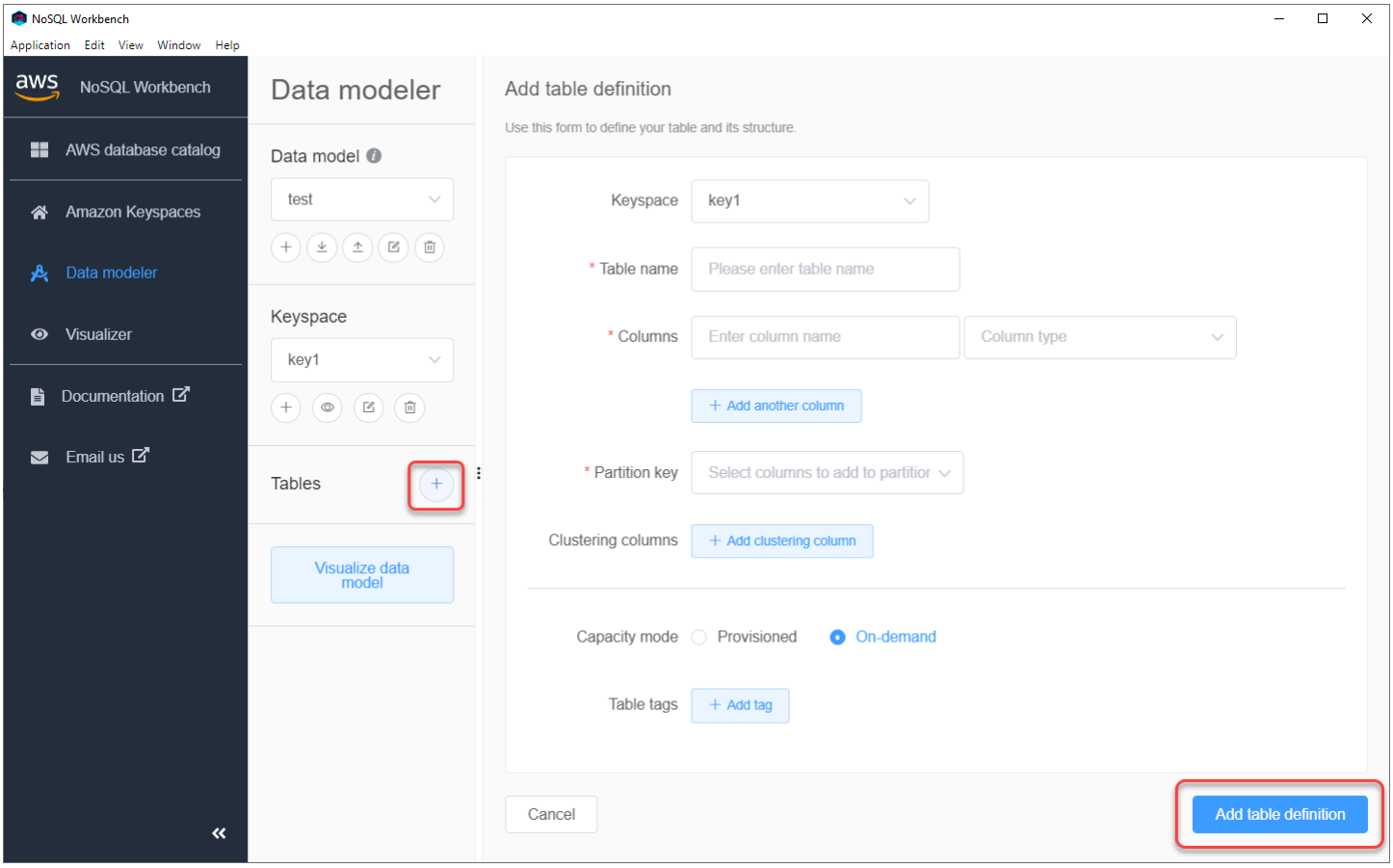
   - Nombre del espacio de claves: introduzca el nombre del nuevo espacio de claves.
   - Estrategia de réplica: elija la estrategia de réplica para el espacio de claves. Amazon Keyspaces utiliza la SingleRegionStrategy para replicar los datos tres veces automáticamente en múltiples zonas de disponibilidad de AWS. Si tiene previsto confirmar el modelo de datos en un clúster de Apache Cassandra, puede elegir SimpleStrategy o NetworkTopologyStrategy.
   - Etiquetas de espacios de claves: las etiquetas de recursos son opcionales y le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario, entorno u otros criterios. Para obtener más información sobre etiquetas para los recursos de Amazon Keyspaces, consulte the section called “Trabajar con etiquetas”.
2. Elija Añadir definición de espacio de claves para crear el espacio de claves.



3. Para crear una nueva tabla, elija el signo más junto a Tablas. En este paso, usted define las siguientes propiedades y configuraciones.

   - Nombre de la tabla: el nombre de la nueva tabla.
   - Columnas: añada un nombre de columna y elija el tipo de datos. Repita estos pasos para cada columna de su esquema.
   - Clave de partición: elija las columnas para la clave de partición.
   - Columnas de agrupación: elija columnas de agrupación (opcional).
   - Modo de capacidad: elija el modo de capacidad de lectura/escritura para la tabla. Puede elegir capacidad aprovisionada o bajo demanda. Para obtener más información sobre los modos de capacidad, consulte the section called “Modos de capacidad de lectura/escritura”.
   - Etiquetas de tabla: las etiquetas de recursos son opcionales y le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario, entorno u otros criterios. Para obtener más información sobre etiquetas para los recursos de Amazon Keyspaces, consulte the section called “Trabajar con etiquetas”.

4. Elija Añadir definición de tabla para crear la nueva tabla.
5. Para crear tablas adicionales, repita estos pasos.
6. Vaya a the section called “Visualización de un modelo de datos” para visualizar el modelo de datos que ha creado.



Edición de modelos de datos existentes con NoSQL Workbench

Con el modelador de datos de NoSQL Workbench, puede editar los modelos de datos existentes en Amazon Keyspaces. Podrían ser modelos de datos importados de un archivo, modelos de datos de muestra proporcionados o modelos de datos que haya creado previamente.

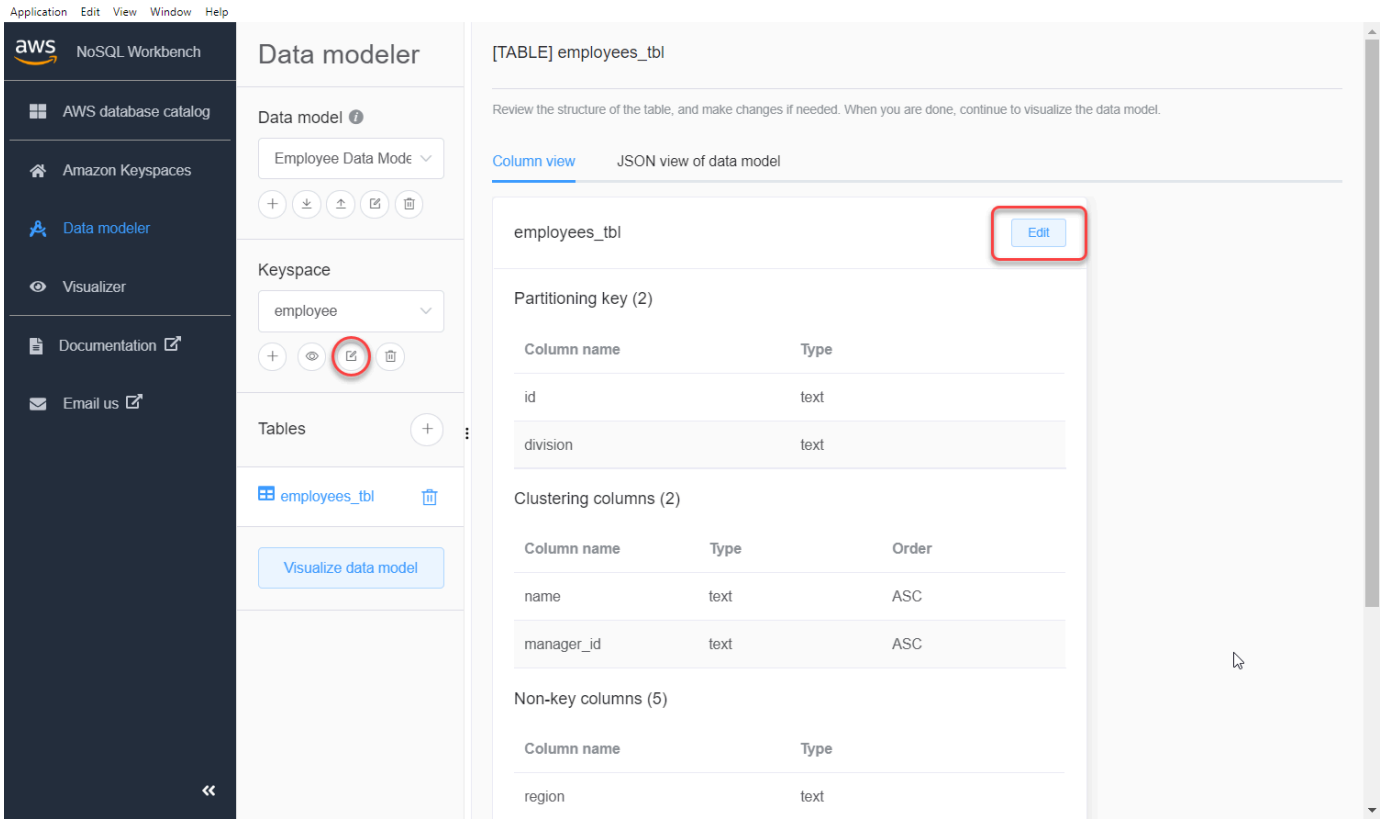
1. Para editar un espacio de claves, elija el símbolo de edición en Espacio de claves.

   En este paso, puede editar las siguientes propiedades y configuraciones.

   - Nombre del espacio de claves: introduzca el nombre del nuevo espacio de claves.
   - Estrategia de réplica: elija la estrategia de réplica para el espacio de claves. Amazon Keyspaces utiliza la SingleRegionStrategy para replicar los datos tres veces automáticamente en múltiples zonas de disponibilidad de AWS. Si tiene previsto confirmar el modelo de datos en un clúster de Apache Cassandra, puede elegir SimpleStrategy o NetworkTopologyStrategy.
   - Etiquetas de espacios de claves: las etiquetas de recursos son opcionales y le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario,

entorno u otros criterios. Para obtener más información sobre etiquetas para los recursos de Amazon Keyspaces, consulte the section called “Trabajar con etiquetas”.

2. Elija Guardar ediciones para actualizar el espacio de claves.



3. Para editar una tabla, elija Editar junto al nombre de la tabla. En este paso, puede actualizar las siguientes propiedades y configuraciones.

   - Nombre de la tabla: el nombre de la nueva tabla.
   - Columnas: añada un nombre de columna y elija el tipo de datos. Repita estos pasos para cada columna de su esquema.
   - Clave de partición: elija las columnas para la clave de partición.
   - Columnas de agrupación: elija columnas de agrupación (opcional).
   - Modo de capacidad: elija el modo de capacidad de lectura/escritura para la tabla. Puede elegir capacidad aprovisionada o bajo demanda. Para obtener más información sobre los modos de capacidad, consulte the section called “Modos de capacidad de lectura/escritura”.
   - Etiquetas de tabla: las etiquetas de recursos son opcionales y le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario, entorno u otros criterios. Para obtener más información sobre etiquetas para los recursos de Amazon Keyspaces, consulte the section called “Trabajar con etiquetas”.

4. Elija Guardar ediciones para actualizar la tabla.
5. Vaya a the section called “Visualización de un modelo de datos” para visualizar el modelo de datos que ha actualizado.

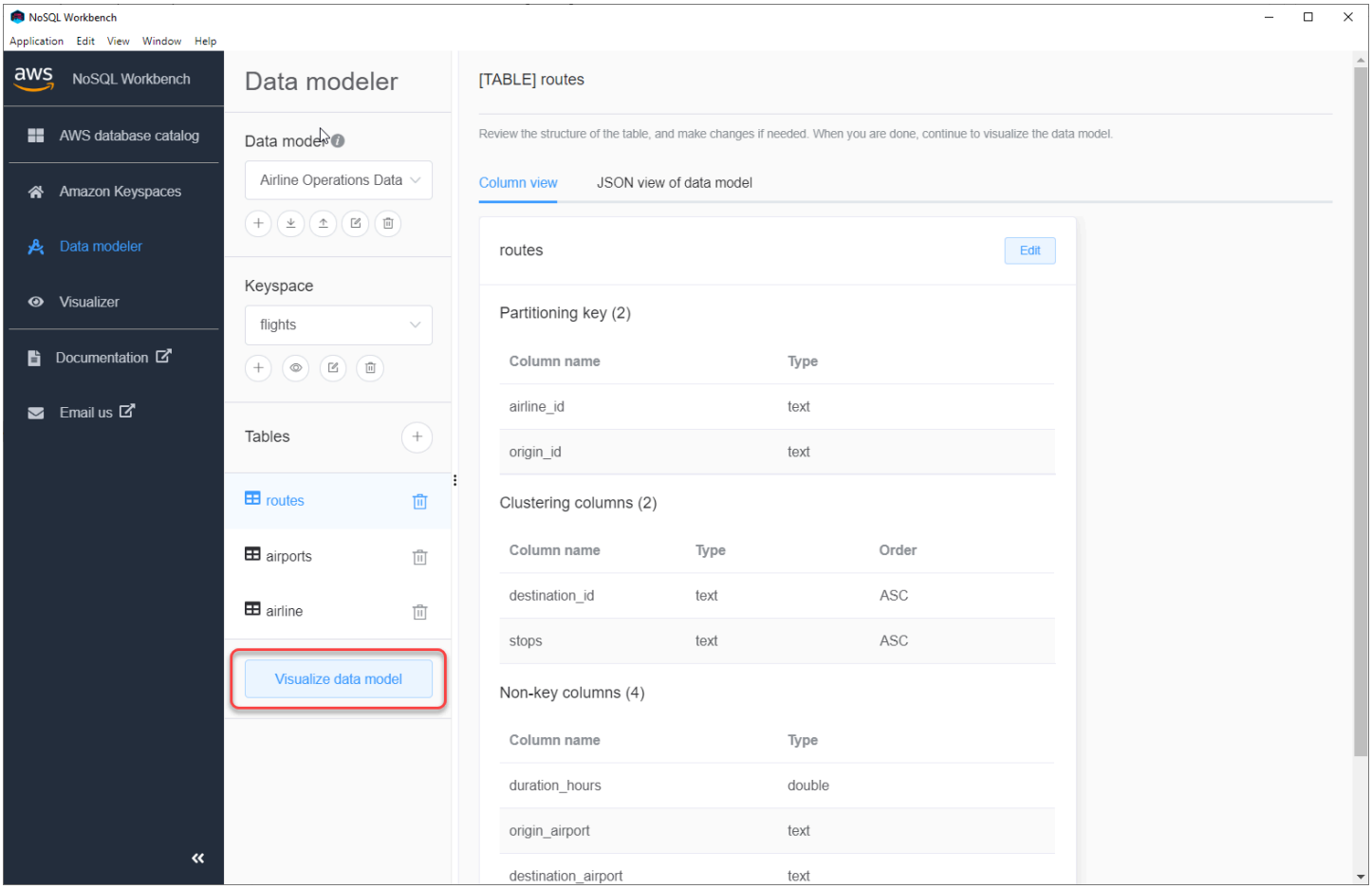
Visualización de modelos de datos

Con NoSQL Workbench, puede visualizar sus modelos de datos para asegurarse de que estos admitan las consultas y los patrones de acceso de su aplicación. También puede guardar y exportar sus modelos de datos en diversos formatos para colaboración, documentación y presentaciones.

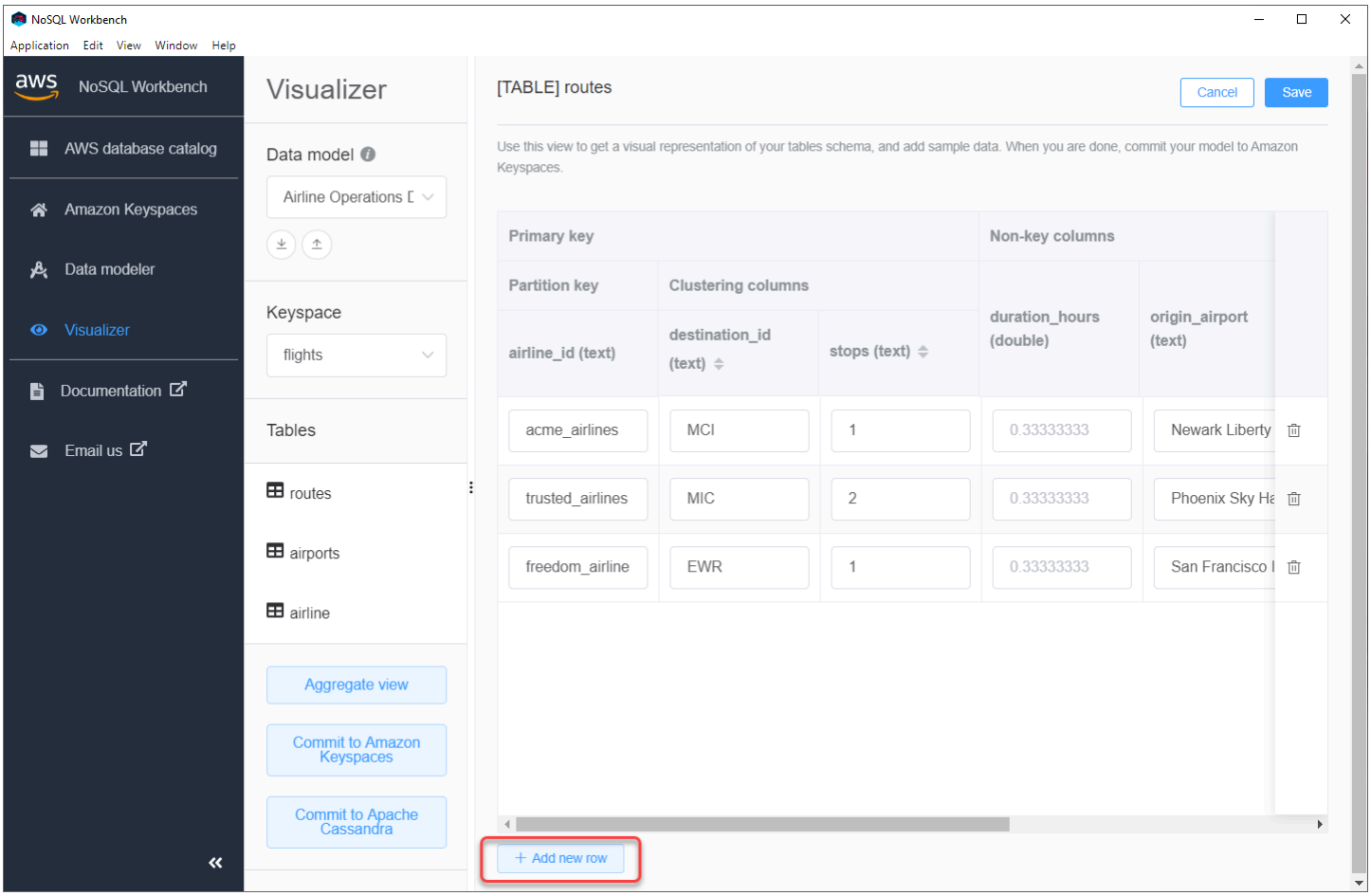
Después de haber creado un nuevo modelo de datos o editado un modelo de datos existente, puede visualizar el modelo.

Visualización de modelos de datos con NoSQL Workbench

Cuando haya completado el modelo de datos en el modelador de datos, elija Visualizar modelo de datos.

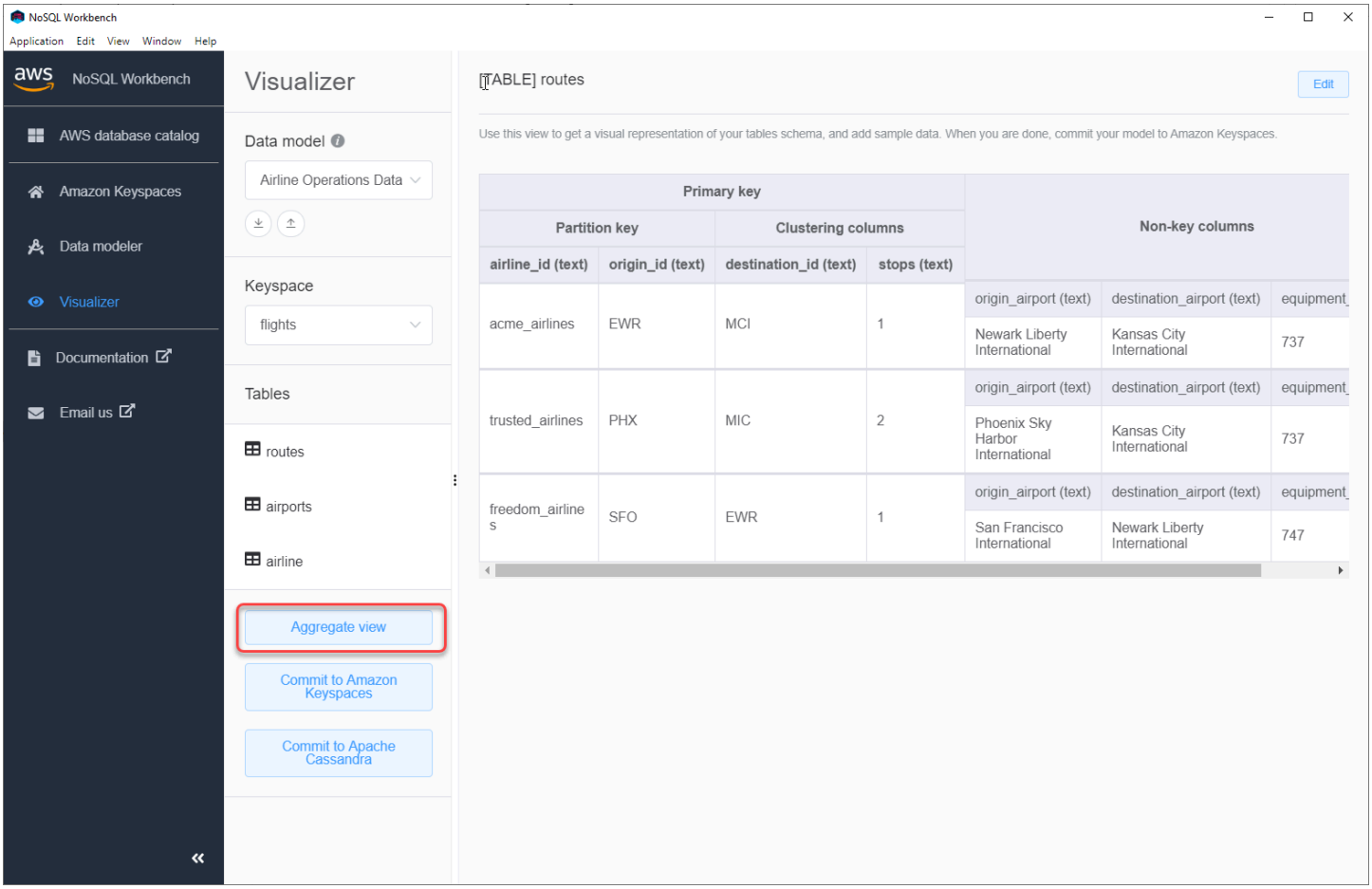


Esto lo lleva al visualizador de datos en NoSQL Workbench. El visualizador de datos proporciona una representación visual del esquema de la tabla y le permite añadir datos de muestra. Para añadir datos de muestra a una tabla, elija una tabla del modelo y, a continuación, Editar. Para añadir una nueva fila de datos, elija Añadir nueva fila en la parte inferior de la pantalla. Elija Save cuando haya terminado.



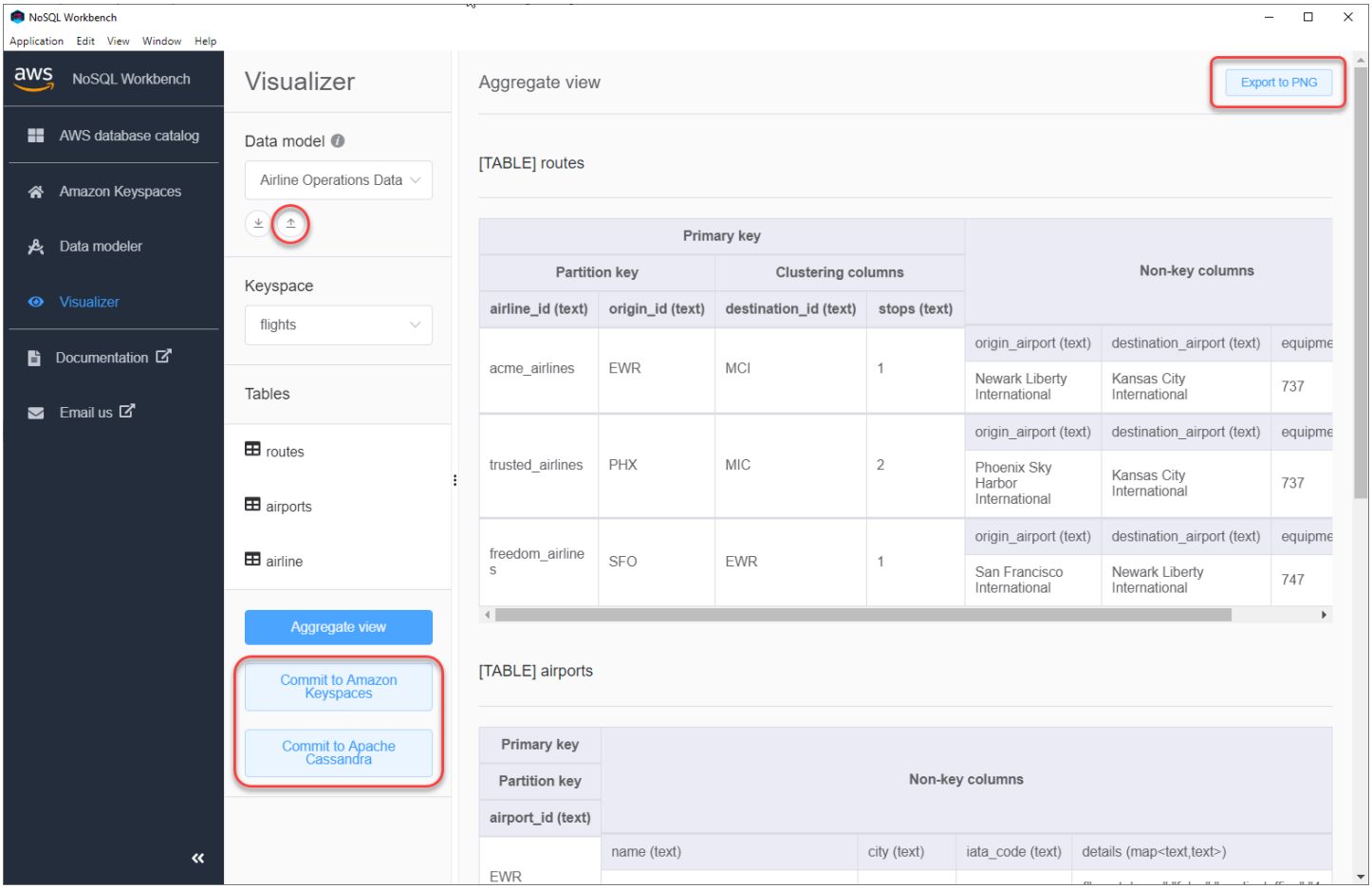
Vista agregada

Una vez confirmado el esquema de la tabla, puede añadir visualizaciones del modelo de datos.



Después de haber agregado la vista del modelo de datos, puede exportar la vista a un archivo PNG. Para exportar el modelo de datos a un archivo JSON, seleccione el signo de carga bajo el nombre del modelo de datos.

> **Note**
>
> Puede exportar el modelo de datos en formato JSON en cualquier momento del proceso de diseño.



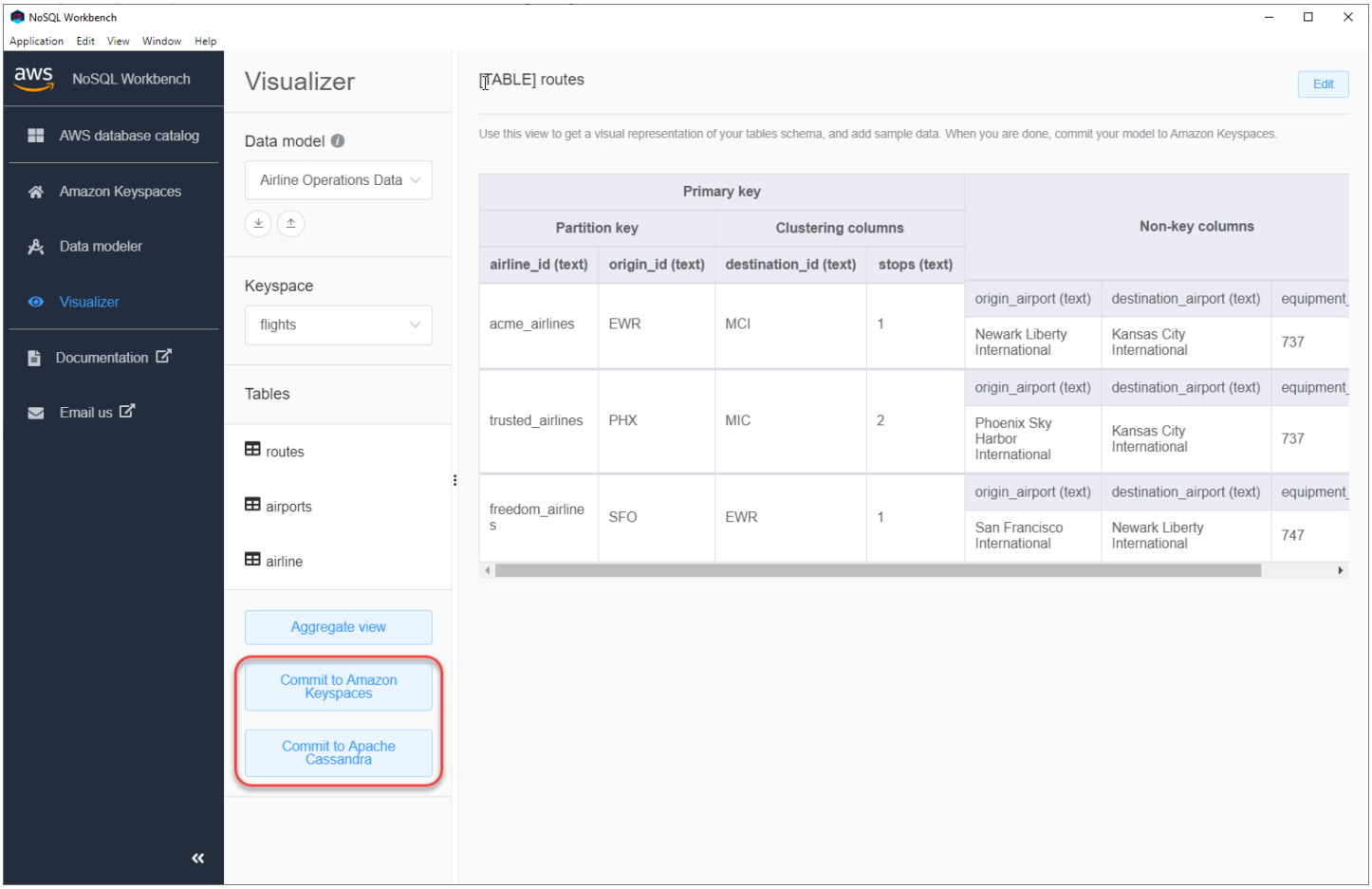
Dispone de las siguientes opciones para confirmar los cambios:

- Confirmar en Amazon Keyspaces
- Confirmar en un clúster de Apache Cassandra

Para obtener más información sobre cómo confirmar los cambios, consulte the section called "Confirmación de un modelo de datos".

Confirmación de modelos de datos en Amazon Keyspaces y Apache Cassandra

En esta sección se muestra cómo confirmar modelos de datos completados en clústeres de Amazon Keyspaces y Apache Cassandra. Este proceso crea automáticamente los recursos del lado del servidor para los espacios de claves y las tablas en función de la configuración que haya definido en el modelo de datos.



Temas

- Antes de empezar
- Conexión a Amazon Keyspaces con credenciales específicas del servicio
- Conexión a Amazon Keyspaces con credenciales de AWS Identity and Access Management (IAM)
- Uso de una conexión guardada
- Confirmación en Apache Cassandra

Antes de empezar

Amazon Keyspaces requiere el uso de Seguridad de la capa de transporte (TLS) para ayudar a proteger las conexiones con los clientes. Para conectarse a Amazon Keyspaces con TLS, debe completar antes las siguientes tareas para poder comenzar.

- Descargue el certificado digital de Starfield utilizando el siguiente comando y guarde `sf-class2-root.crt` localmente o en su directorio de inicio.

```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```

> **Note**
>
> También puede utilizar el certificado digital de Amazon para conectarse a Amazon Keyspaces, y puede seguir haciéndolo si su cliente se conecta a Amazon Keyspaces correctamente. El certificado de Starfield proporciona compatibilidad retroactiva adicional para los clientes que utilicen autoridades de certificación más antiguas.


```
curl https://certs.secureserver.net/repository/sf-class2-root.crt -O
```


Una vez guardado el archivo de certificado, ya puede conectarse a Amazon Keyspaces. Una opción es conectarse utilizando credenciales específicas del servicio. Las credenciales específicas del servicio son un nombre de usuario y una contraseña que están asociados a un usuario de IAM en concreto y que solo pueden utilizarse con el servicio especificado. La segunda opción es conectarse con credenciales de IAM que utilicen el proceso Signature Version 4 (SigV4) de AWS. Para obtener más información sobre estas dos opciones, consulte the section called “Creación de credenciales”.

Para conectarse con credenciales específicas del servicio, consulte the section called “Conexión con credenciales específicas del servicio”.

Para conectarse con credenciales de IAM, consulte the section called “Conexión con credenciales de IAM”.

Conexión a Amazon Keyspaces con credenciales específicas del servicio

En esta sección se muestra cómo utilizar credenciales específicas del servicio para confirmar el modelo de datos que ha creado o editado con NoSQL Workbench.

1. Para crear una nueva conexión con credenciales específicas del servicio, elija la pestaña Conectar con nombre de usuario y contraseña.

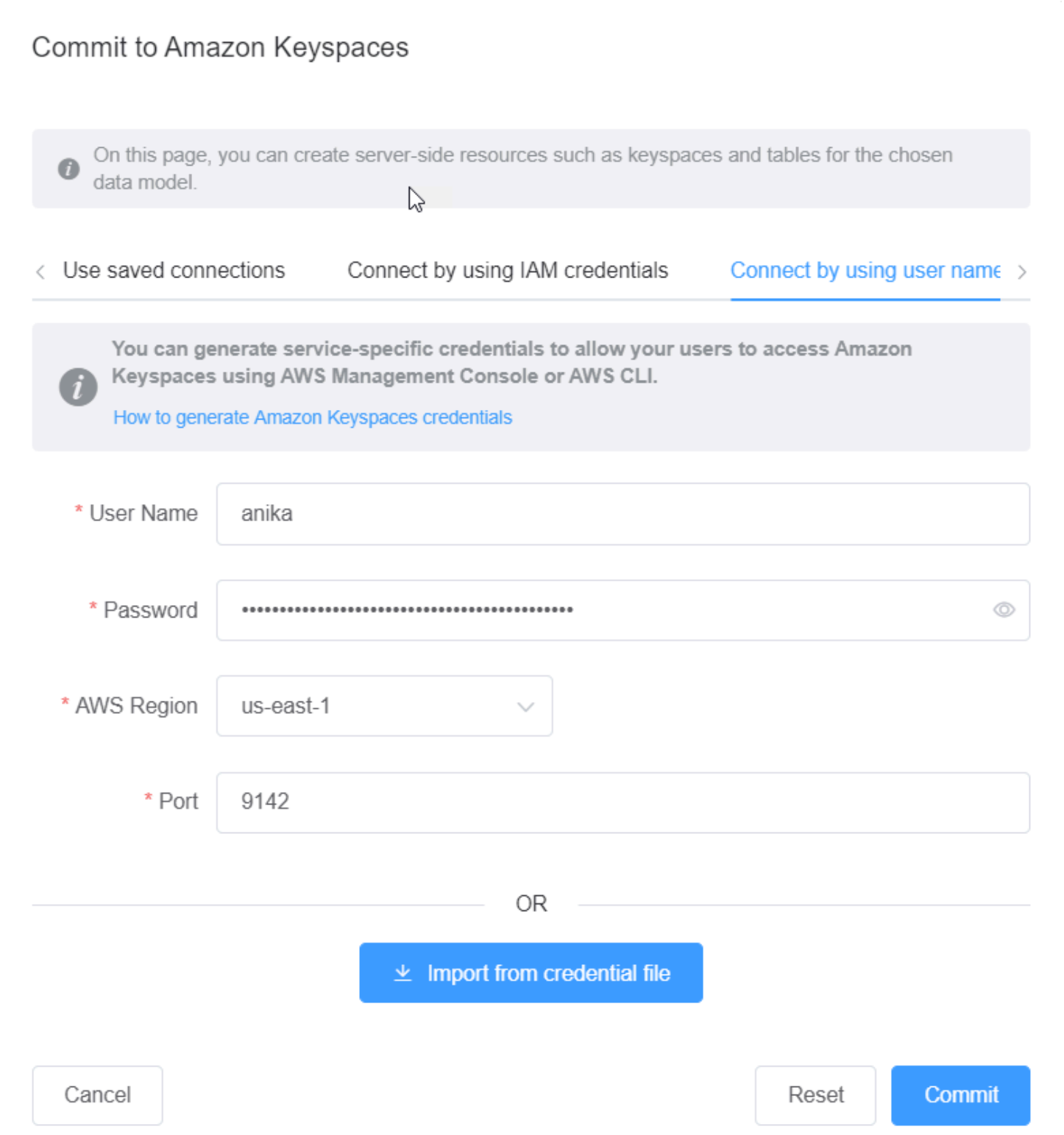
   - Antes de comenzar, debe crear las credenciales específicas del servicio mediante el proceso documentado en the section called “Credenciales específicas del servicio”.

Una vez obtenidas las credenciales específicas del servicio, puede seguir configurando la conexión. Continúe con una de las siguientes opciones:

- Nombre de usuario: introduzca el nombre de usuario.
- Contraseña: introduzca la contraseña.
- Región de AWS: para conocer las regiones disponibles, consulte the section called “Puntos de conexión de servicio”.
- Puerto: Amazon Keyspaces utiliza el puerto 9142.

Como alternativa, puede importar credenciales guardadas desde un archivo.

2. Elija Confirmar para actualizar Amazon Keyspaces con el modelo de datos.


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

Conexión a Amazon Keyspaces con credenciales de AWS Identity and Access Management (IAM)

En esta sección se muestra cómo utilizar credenciales de IAM para confirmar el modelo de datos que ha creado o editado con NoSQL Workbench.

1. Para crear una nueva conexión utilizando credenciales IAM, elija la pestaña Conectar con credenciales de IAM.

   - Antes de comenzar, debe crear las credenciales de IAM utilizando uno de los siguientes métodos.

     - Para acceder a la consola, utilice su nombre de usuario y contraseña de IAM para iniciar sesión en la AWS Management Console desde la página de inicio de sesión de IAM. Para más información sobre las credenciales de seguridad de AWS, incluido el acceso programático y las alternativas a las credenciales a largo plazo, consulte Credenciales de seguridad de AWS en la Guía del usuario de IAM. Para información sobre cómo iniciar sesión en su cuenta de Cuenta de AWS, consulte Cómo iniciar sesión en AWS en la Guía del usuario de AWS Sign-In.
     - Para acceder a la CLI, necesita un ID de clave de acceso y una clave de acceso secreta. Cuando sea posible, utilice credenciales temporales en lugar de claves de acceso. Las credenciales temporales incluyen un ID de clave de acceso y una clave de acceso secreta, pero, además, incluyen un token de seguridad que indica cuándo caducan las credenciales. Para obtener más información, consulte Uso de credenciales temporales con AWS en la Guía del usuario de IAM.
     - Para acceder a la API, necesita un ID de clave de acceso y una clave de acceso secreta. Utilice las claves de acceso de usuario de IAM en lugar de las claves de acceso de Usuario raíz de la cuenta de AWS. Para obtener más información sobre la creación de claves de acceso, consulte Administración de las claves de acceso de los usuarios de IAM en la Guía del usuario de IAM

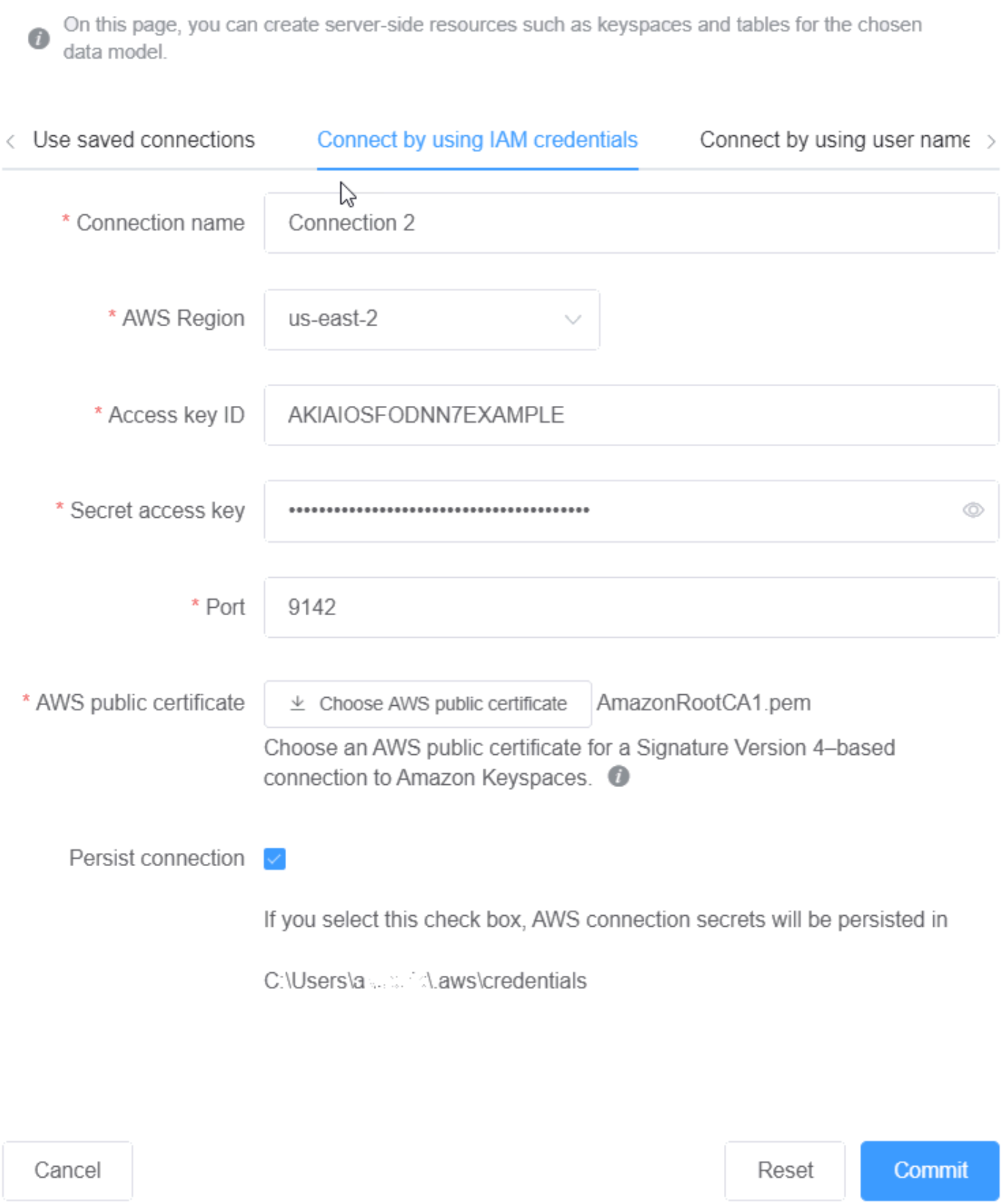
     Para obtener más información, consulte Administración de las claves de acceso de los usuarios de IAM.

   Una vez obtenidas las credenciales de IAM, puede seguir configurando la conexión.

   - Nombre de conexión: el nombre de la conexión.

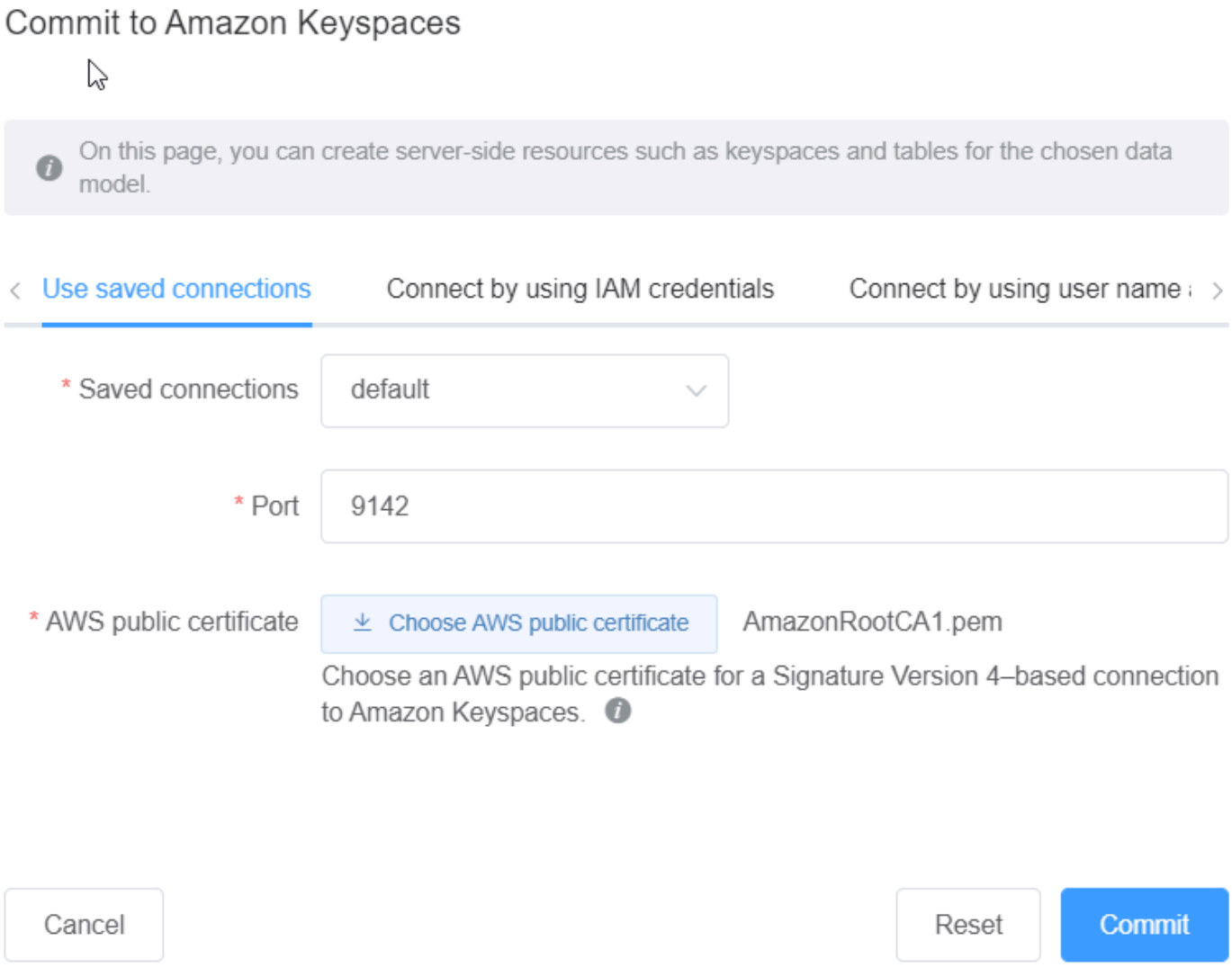
- Región de AWS: para conocer las regiones disponibles, consulte the section called “Puntos de conexión de servicio”.
- ID de clave de acceso: introduzca el ID de clave de acceso.
- Clave de acceso secreta: introduzca la clave de acceso secreta.
- Puerto: Amazon Keyspaces utiliza el puerto 9142.
- Certificado público de AWS: señale el certificado de AWS que se descargó en el primer paso.
- Mantener conexión: seleccione esta casilla de verificación si desea guardar los secretos de conexión de AWS localmente.

2. Elija Confirmar para actualizar Amazon Keyspaces con el modelo de datos.


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

Uso de una conexión guardada

Si ha configurado previamente una conexión a Amazon Keyspaces, puede utilizarla como conexión predeterminada para confirmar los cambios del modelo de datos. Elija la pestaña Utilizar conexiones guardadas y continúe confirmando las actualizaciones.

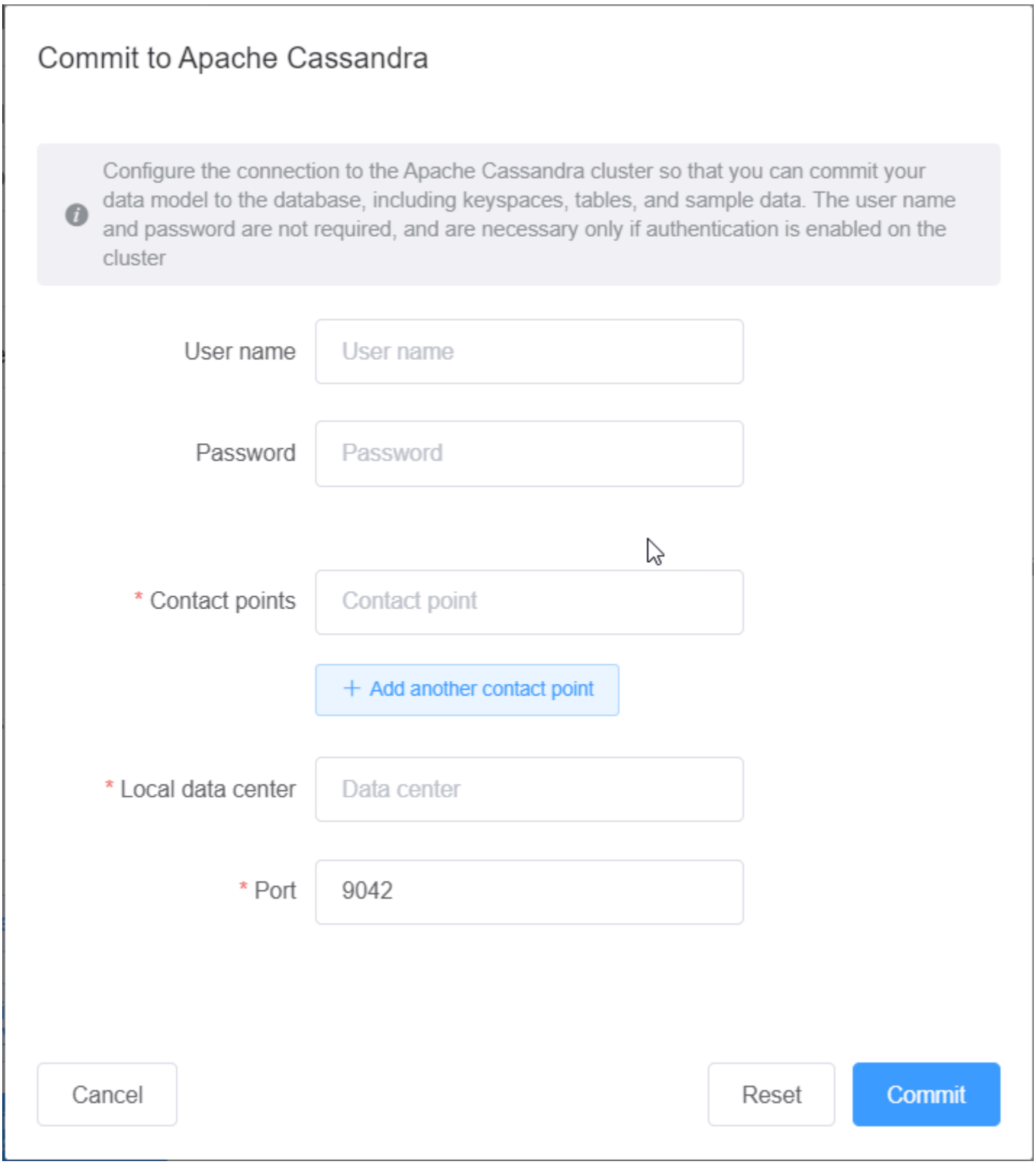


Confirmación en Apache Cassandra

En esta sección se explica cómo establecer las conexiones con un clúster de Apache Cassandra para confirmar el modelo de datos creado o editado con NoSQL Workbench.

> **Note**
>
> Solo los modelos de datos que se hayan creado con `SimpleStrategy` o `NetworkTopologyStrategy` pueden confirmarse en clústeres de Apache Cassandra. Para cambiar la estrategia de réplica, edite el espacio de claves en el modelador de datos.

1.
   - Nombre de usuario: introduzca el nombre de usuario si la autenticación está habilitada en el clúster.
   - Contraseña: introduzca la contraseña si la autenticación está habilitada en el clúster.
   - Puntos de contacto: introduzca los puntos de contacto.
   - Centro de datos local: introduzca el nombre del centro de datos local.
   - Puerto: la conexión utiliza el puerto 9042.
2. Elija Confirmar para actualizar el clúster de Apache Cassandra con el modelo de datos.


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

Modelos de datos de ejemplo en NoSQL Workbench

La página de inicio del modelador y el visualizador muestra una serie de modelos de ejemplo que se incluyen con NoSQL Workbench. En esta sección se describen estos modelos y sus usos potenciales.

Temas

- Modelo de datos de empleados
- Modelo de datos de transacciones de tarjetas de crédito
- Modelo de datos de operaciones de aerolíneas

Modelo de datos de empleados

Este modelo de datos representa un esquema de Amazon Keyspaces para una aplicación de base de datos de empleados.

Las aplicaciones que acceden a la información de los empleados de una empresa determinada pueden utilizar este modelo de datos.

Los patrones de acceso que este modelo de datos admite son:

- Recuperación de un registro de empleado con un ID dado.
- Recuperación de un registro de empleado con un ID y una división dados.
- Recuperación de un registro de empleado con un ID y un nombre dados.

Modelo de datos de transacciones de tarjetas de crédito

Este modelo de datos representa un esquema de Amazon Keyspaces para transacciones de tarjetas de crédito en tiendas minoristas.

El almacenamiento de transacciones de tarjetas de crédito no solo ayuda a las tiendas con la contabilidad, sino que también ayuda a los administradores de las tiendas a analizar las tendencias de compra, lo que puede ayudarles con las previsiones y la planificación.

Los patrones de acceso que este modelo de datos admite son:

- Recuperación de transacciones por número de tarjeta de crédito, mes y año, y fecha.

- Recuperación de transacciones por número de tarjeta de crédito, categoría y fecha.
- Recuperación de transacciones por categoría, ubicación y número de tarjeta de crédito.
- Recuperación de transacciones por número de tarjeta de crédito y estado de disputa.

Modelo de datos de operaciones de aerolíneas

Este modelo de datos muestra datos de vuelos de avión, incluyendo aeropuertos, aerolíneas y rutas de vuelo.

Los componentes clave del modelado de Amazon Keyspaces que se demuestran son los pares clave-valor, los almacenes de datos de columnas anchas, las claves compuestas y los tipos de datos complejos como las asignaciones con el fin de demostrar los patrones comunes de acceso a datos NoSQL.

Los patrones de acceso que este modelo de datos admite son:

- Recuperación de rutas con origen en una aerolínea dada en un aeropuerto dado.
- Recuperación de rutas con un aeropuerto de destino determinado.
- Recuperación de aeropuertos con vuelos directos.
- Recuperación de detalles del aeropuerto y de la compañía aérea.

Historial de versiones de NoSQL Workbench

En la siguiente tabla se describen los cambios importantes de cada versión de la aplicación del lado del cliente NoSQL Workbench..

Cambio	Descripción	Fecha
NoSQL Workbench para Amazon Keyspaces - Disponibilidad general.	NoSQL Workbench para Amazon Keyspaces está disponible de forma general.	28 de octubre de 2020
Publicación de la vista previa de NoSQL Workbench.	NoSQL Workbench es una aplicación del lado del cliente que le ayuda a diseñar y visualizar modelos de	5 de octubre de 2020

Cambio	Descripción	Fecha
	datos no relacionales para Amazon Keyspaces con mayor facilidad. Los clientes de NoSQL Workbench están disponibles para Windows, macOS y Linux. Para obtener más información, consulte NoSQL Workbench para Amazon Keyspaces.	

Réplica multirregión para Amazon Keyspaces (para Apache Cassandra)

Puede utilizar la replicación multirregional de Amazon Keyspaces para replicar sus datos con una replicación automática, totalmente gestionada y activa-activa en todos los lugares de su elección. Regiones de AWS Con la réplica activa-activa, cada región puede realizar lecturas y escrituras de forma aislada. Puede mejorar tanto la disponibilidad como la resiliencia ante la degradación regional y, al mismo tiempo, beneficiarse de las lecturas y escrituras locales de baja latencia para las aplicaciones globales.

Con la réplica multirregión, Amazon Keyspaces replica los datos de forma asíncrona entre las regiones y los datos suelen propagarse por las regiones en menos de un segundo. Además, con la réplica multirregión, se deshace del difícil trabajo de resolver conflictos y corregir problemas de divergencia de datos, lo que le permite centrarse en su aplicación.

De forma predeterminada, Amazon Keyspaces replica los datos en tres zonas de disponibilidad dentro de la misma Región de AWS para garantizar una mayor durabilidad y alta disponibilidad. Con la replicación multirregional, puede crear espacios de claves multirregionales que repliquen sus tablas en hasta seis zonas geográficas diferentes de su elección. Regiones de AWS

Temas

- Ventajas de utilizar la replicación multirregional
- Capacidad, modos y precios
- Cómo funciona la réplica multirregión en Amazon Keyspaces
- Notas de uso de la réplica multirregión de Amazon Keyspaces
- Formas de uso de la réplica multirregión

Ventajas de utilizar la replicación multirregional

La replicación multirregional ofrece los siguientes beneficios.

- Lecturas y escrituras globales con una latencia de milisegundos de un solo dígito: en Amazon Keyspaces, la replicación es activa-activa. Puede efectuar lecturas y escrituras localmente desde las regiones más cercanas a sus clientes con una latencia de un dígito de milisegundo a cualquier

escala. Puede utilizar tablas multirregión de Amazon Keyspaces para aplicaciones globales que necesiten un tiempo de respuesta rápido en cualquier parte del mundo.

- Continuidad empresarial mejorada y protección contra la degradación en una sola región: con la replicación multirregional, puede recuperarse de la degradación en una sola región al redirigir la aplicación a una región Región de AWS diferente de su espacio de claves multirregional. Dado que Amazon Keyspaces ofrece una réplica activa-activa, sus lecturas y escrituras no se ven afectadas.

  Amazon Keyspaces realiza un seguimiento de cualquier escritura que se haya realizado en su espacio de claves multirregión pero que no se haya propagado a todas las regiones de réplica. Cuando la región vuelva a estar en línea, Amazon Keyspaces sincronizará automáticamente los cambios que falten para que pueda recuperarlos sin que ello afecte a la aplicación.
- Replicación de alta velocidad entre regiones: la replicación multirregional utiliza una replicación física de datos rápida y basada en el almacenamiento entre regiones, con un retraso de replicación que suele ser inferior a 1 segundo.

  La réplica en Amazon Keyspaces tiene un impacto mínimo o nulo en las consultas de su base de datos, dado que no comparte recursos informáticos con su aplicación. Esto significa que puede abordar casos de uso de alto rendimiento de escritura o casos de uso con picos o ráfagas repentinos en el rendimiento sin que ello afecte a las aplicaciones.
- Coherencia y resolución de conflictos: cualquier cambio realizado en los datos de cualquier región se replica en las demás regiones en un espacio de claves multirregional. Si las aplicaciones actualizaran los mismos datos en distintas regiones al mismo tiempo, podrían surgir conflictos .

  Con el fin de proporcionar coherencia eventual, Amazon Keyspaces utiliza marcas de tiempo a nivel de celda y una conciliación el último escritor gana entre actualizaciones concurrentes. La resolución de conflictos está totalmente administrada y se produce en segundo plano sin ningún impacto en la aplicación.

Para obtener más información sobre configuraciones y características compatibles, consulte the section called “Notas de uso”.

Capacidad, modos y precios

Para un espacio de claves multirregional, puede utilizar el modo de capacidad bajo demanda o el modo de capacidad aprovisionada. Para obtener más información, consulte the section called “Modos de capacidad de lectura/escritura”.

En el modo bajo demanda, se facturan 1,25 unidades de solicitud de escritura (WRU) para escribir hasta 1 KB de datos por fila. Se te facturan las escrituras en cada región de tu espacio de claves multirregional. Por ejemplo, escribir una fila de 3 KB de datos en un espacio de claves multirregión con dos regiones requiere 7,5 WRU: 3 * 1,25 * 2 = 7,5 WRU. Además, las escrituras que incluyan datos estáticos y no estáticos requieren operaciones de escritura adicionales.

En el modo aprovisionado, se facturan 1,25 unidades de capacidad de escritura (WCU) para escribir hasta 1 KB de datos por fila. Se le facturarán las escrituras en cada región de su espacio de claves multirregional. Por ejemplo, escribir una fila de 3 KB de datos por segundo en un espacio de claves multirregional con dos regiones requiere 7,5 WCU: 3 * 1,25 * 2 = 7,5 WCU. Además, las escrituras que incluyan datos estáticos y no estáticos requieren operaciones de escritura adicionales.

Para obtener más información sobre precios, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Cómo funciona la réplica multirregión en Amazon Keyspaces

Esta sección ofrece una descripción general del funcionamiento de la réplica multirregión de Amazon Keyspaces. Para obtener más información sobre precios, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Temas

- Cómo funciona la réplica multirregión en Amazon Keyspaces
- Resolución de conflictos de réplica multirregión
- Recuperación de desastres de la réplica multirregión
- Permisos de IAM necesarios para crear espacios de claves y tablas multirregión
- Réplica multirregión e integración con la recuperación en un momento dado (PITR)
- Réplica multirregión e integración con los servicios de AWS.

Cómo funciona la réplica multirregión en Amazon Keyspaces

La réplica multirregión de Amazon Keyspaces implementa una arquitectura de resistencia de datos que distribuye sus datos a través de Regiones de AWS independientes geográficamente distribuidas. Utiliza la réplica activa-activa, que ofrece una baja latencia local, dado que cada región puede realizar lecturas y escrituras de forma aislada.

Al crear un espacio de claves multirregión de Amazon Keyspaces, puede seleccionar hasta cinco regiones adicionales a las que se replicarán los datos. Cada tabla que cree en un espacio de claves multirregión consta de varias tablas de réplica (una por región) que Amazon Keyspaces considera como una sola unidad.

Cada réplica tiene el mismo nombre de tabla y el mismo esquema de clave principal. Cuando una aplicación escribe datos en una tabla local en una región, los datos se escriben de forma duradera con el nivel de coherencia `LOCAL_QUORUM`. Amazon Keyspaces replica automáticamente los datos de forma asíncrona a las demás regiones de réplica. El retardo de réplica entre regiones suele ser inferior a un segundo y no afecta al rendimiento o la capacidad de procesamiento de su aplicación.

Una vez escritos los datos, puede leerlos de la tabla multirregión en otra región de réplica con los niveles de coherencia `LOCAL_ONE/LOCAL_QUORUM`. Para obtener más información sobre configuraciones y características compatibles, consulte the section called “Notas de uso”.

Resolución de conflictos de réplica multirregión

La réplica multirregión de Amazon Keyspaces está totalmente administrada, lo que significa que no tiene que realizar tareas de réplica como ejecutar regularmente operaciones de reparación para sanear los problemas de sincronización de datos. Amazon Keyspaces monitorea la coherencia de datos entre tablas de diferentes Regiones de AWS para detectar y reparar conflictos, y sincroniza las réplicas de forma automática.

Amazon Keyspaces utiliza el método el último escritor gana de conciliación de datos. Con este mecanismo de resolución de conflictos, todas las regiones en un espacio de claves multirregión se ponen de acuerdo sobre la última actualización y convergen hacia un estado en el que todas tienen datos idénticos. El proceso de reconciliación no tiene ningún impacto en el rendimiento de la aplicación. Para soportar la resolución de conflictos, las marcas de tiempo del lado del cliente se activan automáticamente para las tablas multirregión y no se pueden desactivar. Para obtener más información, consulte Marcas de tiempo del lado del cliente.

Recuperación de desastres de la réplica multirregión

Con la réplica multirregión de Amazon Keyspaces, tanto las lecturas como las escrituras se replican de forma asíncrona en cada región. En el raro caso de que se produzca una degradación o un fallo en una única región, la réplica multirregión le ayuda a recuperarse del desastre con un impacto mínimo o nulo en su aplicación. La recuperación tras un desastre suele medirse utilizando valores de objetivo de tiempo de recuperación (RTO) y objetivo de punto de recuperación (RPO).

Objetivo de tiempo de recuperación: el tiempo que tarda un sistema en volver a un estado de funcionamiento tras un desastre. El RTO mide el tiempo de inactividad que su carga de trabajo puede tolerar, medido en tiempo. En los planes de recuperación de desastres que utilizan la réplica multirregión para conmutar por error a una región no afectada, el RTO puede ser casi cero. El RTO está limitado por la rapidez con la que su aplicación puede detectar la condición de fallo y redirigir el tráfico a otra región.

Objetivo de punto de recuperación: la cantidad de datos que pueden perderse (medido en tiempo). En los planes de recuperación de desastres que utilicen la réplica multirregión para conmutar por error a una región no afectada, el RPO suele ser de un dígito de segundo. El RPO está limitado por la latencia de la réplica a la réplica de destino de la conmutación por error.

En caso de fallo o degradación regional, no necesita promover una región secundaria ni realizar procedimientos de conmutación por error de la base de datos porque la réplica en Amazon Keyspaces es activo-activo. En cambio, puede utilizar Amazon Route 53 para dirigir su aplicación a la región sana más cercana. Para obtener más información sobre Route 53, consulte ¿Qué es Amazon Route 53?.

Si una única Región de AWS se aísla o degrada, su aplicación puede redirigir el tráfico a una región diferente mediante Route 53 para realizar lecturas y escrituras ante una tabla de réplica diferente. También puede aplicar lógica empresarial personalizada a fin de determinar cuándo redirigir las solicitudes a otras regiones. Un ejemplo de ello es hacer que su aplicación sea consciente de los múltiples puntos de conexión disponibles.

Cuando la región vuelve a estar en línea, Amazon Keyspaces reanuda la propagación de las escrituras pendientes de esa región a las tablas de réplica en otras regiones.. Asimismo, reanudará la propagación de las operaciones de escritura de otras réplicas de tabla a la región que ahora está en línea.

Permisos de IAM necesarios para crear espacios de claves y tablas multirregión

Para crear con éxito espacios de claves y tablas multirregión, la entidad principal de IAM necesita poder crear un rol vinculado a servicios. Este rol vinculado a servicios es un tipo único de rol de IAM predefinido por Amazon Keyspaces. Incluye todos los permisos que Amazon Keyspaces necesita para realizar acciones en su nombre. Para obtener más información sobre el rol vinculado a servicios, consulte the section called “Réplica multirregión”.

Para crear el rol vinculado a servicios requerido por la réplica multirregión, la política de la entidad principal de IAM requiere los siguientes elementos:

- `iam:CreateServiceLinkedRole`: la acción que la entidad principal puede realizar.
- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`: el recurso sobre el que se puede realizar la acción.
- `iam:AWSServiceName": "replication.cassandra.amazonaws.com`: el único servicio de AWS al que se puede vincular este rol es Amazon Keyspaces.

A continuación se muestra un ejemplo de política que concede los permisos mínimos necesarios a una entidad principal para crear espacios de claves y tablas multirregión.

```
{
            "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"}}
}
```

Para obtener permisos de IAM adicionales para espacios de claves y tablas multirregión, consulte Claves de acción, recursos y condiciones para Amazon Keyspaces (para Apache Cassandra) en la Referencia de autorización de servicios.

Réplica multirregión e integración con la recuperación en un momento dado (PITR)

La recuperación en un momento dado es compatible con las tablas multirregión. Para restaurar correctamente una tabla multirregión con PITR, se deben satisfacer las siguientes condiciones.

- La tabla de origen y la tabla de destino deben estar configuradas como tablas multirregión.
- Las regiones de réplica para el espacio de claves de la tabla de origen y para el espacio de claves de la tabla de destino deben ser las mismas.

Puede ejecutar la sentencia de restauración desde cualquiera de las regiones en las que esté disponible la tabla de origen. Amazon Keyspaces restaura automáticamente la tabla de destino

en cada región. Para obtener más información acerca de PITR, consulte the section called “Cómo funciona”.

Réplica multirregión e integración con los servicios de AWS.

Puede monitorear el rendimiento de la réplica entre tablas en diferentes Regiones de AWS mediante las métricas de Amazon CloudWatch. La siguiente métrica proporciona un monitoreo continuo de los espacios de claves multirregión.

- `ReplicationLatency`: Esta métrica mide el tiempo que se tardó en replicar `updates`, `inserts` o `deletes` de una tabla réplica a otra tabla réplica en un espacio de claves multirregión.

Para obtener más información sobre cómo monitorear las métricas de CloudWatch, consulte the section called “Monitorización con CloudWatch”.

Notas de uso de la réplica multirregión de Amazon Keyspaces

Tenga en cuenta lo siguiente al utilizar la réplica multirregión con Amazon Keyspaces.

- Puede seleccionar hasta seis de los públicos disponibles Regiones de AWS. AWS GovCloud (US) Regions, las regiones de China y las Regiones de AWS que están deshabilitadas de forma predeterminada no son compatibles.
- Elija con cuidado las regiones de réplica para el espacio de claves dado que no podrá añadirlas o eliminarlas a posteriori.
- Finalice el esquema de la tabla antes de crear una tabla multirregión dado que no podrá añadir nuevas columnas a posteriori.
- Para el cifrado en reposo, utilice una AWS clave propia. Las claves administradas por el cliente no son compatibles con las tablas multirregión. Para obtener más información, consulte the section called “Cómo funciona”.
- Cuando utilice la administración de capacidad aprovisionada con el escalado automático de Amazon Keyspaces, asegúrese de utilizar las operaciones de la API de Amazon Keyspaces para crear y configurar sus tablas multirregionales. Las operaciones subyacentes de la API Application Auto Scaling a las que Amazon Keyspaces llama en su nombre no tienen capacidades multirregionales.

Para obtener más información, consulte the section called “Formas de uso de la réplica multirregión”. Para obtener más información sobre cómo estimar el rendimiento de la capacidad de escritura de las tablas multirregionales aprovisionadas, consulte. the section called “Tablas multirregionales”

- Decida si la tabla necesita periodo de vida (TTL). No podrá activarlo a posteriori. Para obtener más información, consulte Caducidad de datos con Periodo de vida.
- Si bien los datos se replican automáticamente en las regiones seleccionadas de una tabla multirregión, cuando un cliente se conecta a un punto de conexión en una región y consulta la tabla `system.peers`, la consulta devuelve solo información local. El resultado de la consulta aparece como un único clúster de centro de datos para el cliente.
- La replicación multirregional de Amazon Keyspaces es asíncrona y admite la coherencia de las escrituras. `LOCAL_QUORUM` `LOCAL_QUORUM`La coherencia requiere que la actualización de una fila se mantenga de forma duradera en dos réplicas de la región local antes de que el cliente se realice correctamente. A continuación, la propagación de las escrituras a la región (o regiones) replicadas se realiza de forma asíncrona.

  La replicación multirregional de Amazon Keyspaces no admite la replicación sincrónica ni la coherencia. `QUORUM`
- Al crear un espacio de claves o una tabla multirregión, las etiquetas que defina durante el proceso de creación se aplicarán automáticamente a todos los espacios de claves y tablas de todas las regiones. Si cambia las etiquetas existentes mediante `ALTER KEYSPACE` o`ALTER TABLE`, la actualización solo se aplica al espacio de claves o a la tabla de la región en la que va a realizar el cambio.
- Amazon CloudWatch proporciona una `ReplicationLatency` métrica para cada región replicada. Calcula esta métrica rastreando las filas que llegan, comparando su hora de llegada con su tiempo de escritura inicial y calculando una media. Los tiempos se almacenan CloudWatch en la región de origen. Para obtener más información, consulte the section called “Monitorización con CloudWatch”.

  Puede resultar útil ver los tiempos promedio y máximo para determinar el retraso de replicación promedio y el peor de los casos. No hay SLA por esta latencia.

Formas de uso de la réplica multirregión

Puede crear y administrar tablas y espacios de claves de varias regiones mediante la consola Amazon Keyspaces (para Apache Cassandra), Cassandra Query Language (CQL), el AWS SDK y (). AWS Command Line Interface AWS CLI

En esta sección se proporcionan ejemplos de cómo crear tablas y espacios de claves multirregionales con la consola, con CQL y con el modo de capacidad aprovisionada y bajo demanda. AWS CLI Todas las tablas creadas en un espacio de claves multirregional heredan automáticamente la configuración multirregional del espacio de claves.

En esta sección también se incluyen ejemplos de cómo usar la consola, el CQL y la administración de la AWS CLI configuración de escalado automático de Amazon Keyspaces de las tablas multirregiones aprovisionadas. Para obtener más información sobre las opciones generales de configuración del autoescalado y su funcionamiento, consultethe section called “Administre la capacidad de rendimiento con el escalado automático”.

Tenga en cuenta que si utiliza el modo de capacidad aprovisionada para tablas de varias regiones, siempre debe utilizar las llamadas a la API de Amazon Keyspaces para configurar el escalado automático. Esto se debe a que las operaciones subyacentes de la API Application Auto Scaling no tienen en cuenta las regiones.

Para obtener más información sobre cómo estimar el rendimiento de la capacidad de escritura de las tablas multirregionales aprovisionadas, consulte. the section called “Tablas multirregionales”

Para obtener más información sobre la API de Amazon Keyspaces, consulte la referencia de la API de Amazon Keyspaces.

Para obtener más información sobre las configuraciones compatibles y las funciones de replicación multirregional, consulte. the section called “Notas de uso”

Temas

- Uso de la consola para crear y administrar tablas multirregionales
- Uso de CQL para crear y administrar tablas multirregionales
- Uso del AWS CLI para crear y administrar tablas multirregionales

Uso de la consola para crear y administrar tablas multirregionales

En esta sección se proporcionan ejemplos de cómo crear tablas y espacios de claves de varias regiones en modo de capacidad aprovisionada y bajo demanda mediante la consola Amazon Keyspaces (para Apache Cassandra). Todas las tablas que cree en un espacio de claves multirregional heredan automáticamente la configuración multirregional del espacio de claves.

Para ver ejemplos de CQL, consulte. the section called “Uso de CQL” Para ver AWS CLI ejemplos, consultethe section called “Uso del AWS CLI”.

Temas

- Creación de un espacio de claves multirregión (consola)
- Crear una tabla multirregional con la configuración predeterminada (consola)
- Creación de una tabla multirregional en modo aprovisionado con el escalado automático activado (consola)
- Habilitar el escalado automático para una tabla multirregión existente (consola)
- Desactivar el escalado automático para una tabla multirregional (consola)
- Visualización de las actividades de escalado automático de Amazon Keyspaces en la consola

Creación de un espacio de claves multirregión (consola)

Siga estos pasos para crear un nuevo espacio de claves multirregión mediante la consola Amazon Keyspaces.

Para crear un espacio de claves multirregión (consola)

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Espacios de claves y, a continuación, Crear espacio de claves.
3. En Nombre del espacio de claves, introduzca el nombre del espacio de claves.
4. En la sección Réplica multirregión, puede añadir hasta cinco regiones adicionales que estén disponibles en la lista.
5. Para terminar, elija Crear espacio de claves.

> **Note**
>
> Al crear un espacio de claves multirregión, Amazon Keyspaces crea un rol vinculado a servicios con el nombre `AWSServiceRoleForAmazonKeyspacesReplication` en su cuenta. Este rol permite a Amazon Keyspaces replicar las escrituras en todas las réplicas de una tabla multirregión en su nombre. Para obtener más información, consulte the section called “Réplica multirregión”.

Crear una tabla multirregional con la configuración predeterminada (consola)

Para crear una tabla multirregión mediante la consola de Amazon Keyspaces, siga estos pasos.

Para crear una tabla multirregión (consola)

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. Elija un espacio de claves multirregión.
3. En la pestaña Tablas, elija Crear tabla.
4. En Nombre de la tabla, introduzca el nombre de la tabla. Las Regiones de AWS en que se replica esta tabla se muestran en el cuadro de información.
5. Continúe con el esquema de la tabla.
6. En Configuración de la tabla, continúe con la opción Configuración predeterminada. Tenga en cuenta los siguientes ajustes predeterminados para las tablas multirregión.

   - Modo de capacidad: el modo de capacidad predeterminado es Bajo demanda. Para obtener más información sobre la configuración del modo aprovisionado, consultethe section called “Creación de una tabla multirregional en modo aprovisionado con el escalado automático activado (consola)”.
   - Administración de claves de cifrado: solo se admite la opción Clave propiedad de AWS.
   - Marcas de tiempo del lado del cliente: esta característica es necesaria para las tablas multirregión.

   - Elija Personalizar configuración si necesita activar el periodo de vida (TTL) para la tabla y todas sus réplicas.

> **Note**
>
> No puede cambiar los ajustes de TTL en una tabla multirregión existente.

7. Para terminar, elija Crear tabla.

Creación de una tabla multirregional en modo aprovisionado con el escalado automático activado (consola)

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a servicios (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realice acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

Para crear una nueva tabla multirregional con el escalado automático activado

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. Elija un espacio de claves multirregión.
3. En la pestaña Tablas, elija Crear tabla.
4. En la página Crear tabla, en la sección Detalles de la tabla, seleccione un espacio de claves y proporcione un nombre para la nueva tabla.
5. En la sección Columnas, cree el esquema de la tabla.
6. En la sección Clave principal, defina la clave principal de la tabla y seleccione las columnas de agrupamiento opcionales.
7. En la sección Configuración de la tabla, elija Personalizar configuración.
8. Continúe en Configuración de capacidad de lectura/escritura.
9. En Modo de capacidad, elija Aprovisionada.
10. En la sección Capacidad de lectura, confirme que esté seleccionada la opción Escalado automático.

Puede optar por configurar las mismas unidades de capacidad de lectura para todas las unidades en las Regiones de AWS que está replicada la tabla. Como alternativa, puede desactivar la casilla de verificación y configurar la capacidad de lectura de cada región de forma diferente.

Si decide configurar cada región de forma diferente, debe seleccionar las unidades de capacidad de lectura mínima y máxima para cada réplica de tabla, así como el uso objetivo.

- Unidades de capacidad mínima: introduzca el valor de nivel mínimo de rendimiento para el que la tabla debería estar siempre preparada para admitir. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
- Unidades de capacidad máxima: introduzca la cantidad máxima de rendimiento que desea aprovisionar para la tabla. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
- Utilización objetivo: introduzca una tasa de utilización objetivo entre el 20 % y el 90 %. Cuando el tráfico supera la tasa de utilización objetivo definida, la capacidad aumenta automáticamente. Cuando el tráfico cae por debajo del objetivo definido, se vuelve a reducir automáticamente.

- Desactive la casilla Escalar automáticamente si desea aprovisionar la capacidad de lectura de la tabla manualmente. Esta configuración se aplica a todas las réplicas de la tabla.

> **Note**
>
> Para garantizar que haya suficiente capacidad de lectura para todas las réplicas, recomendamos el escalado automático de Amazon Keyspaces para las tablas multirregiones aprovisionadas.

> **Note**
>
> Para obtener más información sobre las cuotas predeterminadas de su cuenta y cómo aumentarlas, consulte Cuotas.

11. En la sección Capacidad de escritura, confirme que esté seleccionada la opción Escalar automáticamente. A continuación, configure las unidades de capacidad de la tabla. Las unidades

de capacidad de escritura permanecen sincronizadas en todas las regiones Regiones de AWS para garantizar que haya suficiente capacidad para replicar los eventos de escritura en todas las regiones.

- Desactive la opción Escalar automáticamente si desea aprovisionar la capacidad de escritura de la tabla de forma manual. Esta configuración se aplica a todas las réplicas de la tabla.

> **Note**
>
> Para garantizar que haya suficiente capacidad de escritura para todas las réplicas, recomendamos el escalado automático de Amazon Keyspaces para las tablas multirregiones aprovisionadas.

12. Elija Crear tabla. Su tabla se crea con los parámetros de escalado automático especificados.

Habilitar el escalado automático para una tabla multirregión existente (consola)

Siga estos pasos para habilitar el escalado automático de una tabla de varias regiones en modo aprovisionado con la consola Amazon Keyspaces.

> **Note**
>
> El escalado automático de Amazon Keyspaces requiere la presencia de un rol vinculado a servicios (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) que realice acciones de escalado automático en su nombre. Este rol se crea automáticamente para usted. Para obtener más información, consulte the section called “Uso de roles vinculados a servicios”.

Para habilitar el escalado automático de Amazon Keyspaces para una tabla multirregión existente

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. Elija la tabla con la que desee trabajar y vaya a la pestaña Capacidad.
3. En la sección Configuración de capacidad, selecciona Editar.
4. En el modo de capacidad, asegúrese de que la tabla utilice el modo de capacidad aprovisionada.

5. Seleccione Escalar automáticamente y consulte el paso 9 Creación de una tabla multirregional en modo aprovisionado con el escalado automático activado (consola) para editar la capacidad de lectura y escritura.
6. Cuando se hayan definido los ajustes de escalado automático, elija Guardar.

Desactivar el escalado automático para una tabla multirregional (consola)

Siga estos pasos para desactivar el escalado automático de una tabla de varias regiones en modo aprovisionado con la consola Amazon Keyspaces.

Para desactivar el escalado automático de Amazon Keyspaces para una tabla multirregión existente

1. Inicie sesión en la AWS Management Console consola de Amazon Keyspaces y ábrala en https://console.aws.amazon.com/keyspaces/home.
2. Elija la tabla con la que desee trabajar y seleccione la pestaña Capacidad.
3. En la sección Configuración de capacidad, selecciona Editar.
4. Para deshabilitar el escalado automático de Amazon Keyspaces, desactive la casilla Escalar automáticamente. Al deshabilitar el escalado automático, se anula el registro de la tabla como objetivo escalable con Application Auto Scaling. Para eliminar el rol vinculado al servicio que Application Auto Scaling utiliza para acceder a la tabla de Amazon Keyspaces, siga los pasos que se indican. the section called “Eliminación de un rol vinculado a servicios para Amazon Keyspaces”

   > **Note**
   >
   > Para eliminar el rol vinculado al servicio que utiliza Application Auto Scaling, debe deshabilitar el escalado automático en todas las tablas de la cuenta. Regiones de AWS

5. Cuando se hayan definido los ajustes de escalado automático, elija Guardar.

Visualización de las actividades de escalado automático de Amazon Keyspaces en la consola

Puede supervisar la forma en que el escalado automático de Amazon Keyspaces utiliza los recursos mediante Amazon CloudWatch, que genera métricas sobre su uso y rendimiento. Siga los pasos de la Guía del Application Auto Scaling usuario para crear un CloudWatch panel de control.

Uso de CQL para crear y administrar tablas multirregionales

Puede usar Cassandra Query Language (CQL) para crear y administrar tablas y espacios clave multirregionales en Amazon Keyspaces.

En esta sección se proporcionan ejemplos de cómo crear y gestionar tablas de varias regiones con CQL. Todas las tablas que cree en un espacio de claves multirregional heredan automáticamente la configuración multirregional del espacio de claves. Para obtener más información sobre el CQL, consulte la referencia del lenguaje CQL de Amazon Keyspaces.

Para obtener más información sobre configuraciones y características compatibles, consulte the section called “Notas de uso”.

Temas

- Creación de un espacio de claves multirregión (CQL)
- Crear una tabla multirregional con la configuración predeterminada (CQL)
- Creación de una tabla multirregional con modo de capacidad aprovisionada y escalado automático (CQL)
- Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CQL)
- Visualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CQL)
- Desactivar el escalado automático para una tabla multirregión (CQL)
- Configurar manualmente la capacidad aprovisionada de una tabla multirregional (CQL)

Creación de un espacio de claves multirregión (CQL)

Para crear un espacio de claves multirregional, utilice esta opción `NetworkTopologyStrategy` para especificar en qué espacio de claves Regiones de AWS se va a replicar. Debe incluir su región actual y al menos una región adicional. La siguiente instrucción CQL es un ejemplo de ello.

```
CREATE KEYSPACE mykeyspace
WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3' };
```

Todas las tablas del espacio de claves utilizan la misma estrategia de réplica que el espacio de claves. No puede cambiar la estrategia de réplica a nivel de tabla.

`NetworkTopologyStrategy`— El factor de replicación para cada región es tres porque Amazon Keyspaces replica los datos en tres zonas de disponibilidad dentro de la misma Región de AWS, de forma predeterminada.

> **Note**
>
> Al crear un espacio de claves multirregión, Amazon Keyspaces crea un rol vinculado a servicios con el nombre `AWSServiceRoleForAmazonKeyspacesReplication` en su cuenta. Este rol permite a Amazon Keyspaces replicar las escrituras en todas las réplicas de una tabla multirregión en su nombre. Para obtener más información, consulte the section called “Réplica multirregión”.

Puede utilizar una instrucción CQL para consultar la `tables` tabla del `system_multiregion_info` espacio de claves y enumerar mediante programación las regiones y el estado de la tabla multirregional que especifique. El siguiente código es un ejemplo de ello.

```
SELECT * from system_multiregion_info.tables WHERE keyspace_name = 'mykeyspace' AND
 table_name = 'mytable';
```

El resultado de la declaración tiene el siguiente aspecto:

```
 keyspace_name  | table_name     | region         | status
----------------+----------------+----------------+--------
 mykeyspace     | mytable        | us-east-1      | ACTIVE
 mykeyspace     | mytable        | ap-southeast-1 | ACTIVE
 mykeyspace     | mytable        | eu-west-1      | ACTIVE
```

Crear una tabla multirregional con la configuración predeterminada (CQL)

Para crear una tabla de varias regiones con la configuración predeterminada, puede utilizar el siguiente ejemplo.

```
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'
 }
};
```

Creación de una tabla multirregional con modo de capacidad aprovisionada y escalado automático (CQL)

Para crear una tabla de varias regiones en modo aprovisionado con escalado automático, primero debe especificar el modo de capacidad definiéndolo `CUSTOM_PROPERTIES` para la tabla. Después de especificar el modo de capacidad aprovisionada, puede configurar los ajustes de escalado automático de la tabla mediante`AUTOSCALING_SETTINGS`.

Para obtener información detallada sobre la configuración del escalado automático, la política de seguimiento de objetivos, el valor objetivo y la configuración opcional, consultethe section called “Cree una tabla nueva con escalado automático mediante CQL”.

Al crear una tabla de varias regiones, también puede especificar diferentes configuraciones de capacidad de lectura y escalado automático de lectura para cada réplica de la tabla. La configuración que especifique sobrescribe la configuración general de la tabla para la especificada. Región de AWS Sin embargo, la capacidad de escritura permanece sincronizada entre todas las réplicas para garantizar que haya suficiente capacidad para replicar las escrituras en todas las regiones.

Para definir la capacidad de lectura de una réplica de tabla en una región específica, puedes configurar los siguientes parámetros como parte de la tabla: `replica_updates`

- La región
- Las unidades de capacidad de lectura aprovisionadas (opcionales)
- Configuración de escalado automático para la capacidad de lectura (opcional)

En el siguiente ejemplo, se muestra una `CREATE TABLE` sentencia para una tabla multirregional en modo aprovisionado. Los ajustes generales de escalado automático de la capacidad de escritura y lectura son los mismos. Sin embargo, la configuración de escalado automático de lectura especifica

períodos de enfriamiento adicionales de 60 segundos antes de aumentar o reducir la capacidad de lectura de la mesa. Además, la configuración de escalado automático de la capacidad de lectura para la región EE.UU. Este (Virginia del Norte) es superior a la de otras réplicas. Además, el valor objetivo se establece en el 70% en lugar del 50%.

```
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
                    }
                }
```

```
            }
        }
    }
};
```

Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CQL)

Se puede utilizar `ALTER TABLE` para actualizar el modo de capacidad y la configuración de escalado automático de una tabla existente. Si vas a actualizar una tabla que actualmente está en modo de capacidad bajo demanda, `capacity_mode` es obligatorio. Si la tabla ya está en el modo de capacidad aprovisionada, este campo se puede omitir.

Para obtener información detallada sobre la configuración del escalado automático, la política de seguimiento de objetivos, el valor objetivo y la configuración opcional, consultethe section called “Cree una tabla nueva con escalado automático mediante CQL”.

En la misma declaración, también puede actualizar la capacidad de lectura y la configuración de escalado automático de las réplicas de tablas en regiones específicas actualizando la `replica_updates` propiedad de la tabla. La siguiente instrucción es un ejemplo de ello.

```
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
                    }
                }
            }
        }
    }
};
```

Visualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CQL)

Para ver la configuración de escalado automático de una tabla multirregión, utilice el siguiente comando.

```
SELECT * FROM system_multiregion_info.autoscaling WHERE keyspace_name = 'mykeyspace'
 AND table_name = 'mytable';
```

El resultado de este comando tiene el siguiente aspecto:

```
 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}}}
```

Desactivar el escalado automático para una tabla multirregión (CQL)

Se puede utilizar `ALTER TABLE` para desactivar el escalado automático de una tabla existente. Tenga en cuenta que no puede desactivar el escalado automático para una réplica de tabla individual.

En el siguiente ejemplo, el escalado automático está desactivado para la capacidad de lectura de la tabla.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_read_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

> **Note**
>
> Para eliminar el rol vinculado a servicios utilizado por Application Auto Scaling, debe deshabilitar el escalado automático en todas las tablas de la cuenta en todas las Regiones de AWS.

Configurar manualmente la capacidad aprovisionada de una tabla multirregional (CQL)

Si tiene que desactivar el escalado automático para una tabla de varias regiones, puede utilizarlo `ALTER TABLE` para aprovisionar manualmente la capacidad de lectura de la tabla para una tabla de réplica.

```
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
         }
    }
};
```

> **Note**
>
> Recomendamos usar el escalado automático para las tablas de varias regiones que utilizan la capacidad aprovisionada. Para obtener más información, consulte the section called “Tablas multirregionales”.

Uso del AWS CLI para crear y administrar tablas multirregionales

Puede usar AWS Command Line Interface (AWS CLI) para crear y administrar tablas y espacios clave multirregionales en Amazon Keyspaces.

En esta sección se proporcionan ejemplos de cómo crear y gestionar tablas multirregionales con. AWS CLI Todas las tablas que cree en un espacio de claves multirregional heredan automáticamente la configuración multirregional del espacio de claves.

Para obtener más información sobre los AWS CLI comandos de Amazon Keyspaces descritos en este tema, consulte la Referencia de AWS CLI comandos para Amazon Keyspaces.

Temas

- Creación de un nuevo espacio de claves multirregión (CLI)
- Creación de una nueva tabla multirregión con la configuración predeterminada (CLI)
- Crear una nueva tabla multirregional en modo aprovisionado con escalado automático (CLI)
- Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CLI)
- Visualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CLI)
- Desactivar el escalado automático para una tabla multirregión (CLI)
- Configuración manual de la capacidad aprovisionada de una tabla multirregional (CLI)

Creación de un nuevo espacio de claves multirregión (CLI)

Para crear un espacio de claves multirregión, puede utilizar la siguiente sentencia de la CLI. Especifique su región actual y al menos una región adicional en la `regionList`.

```
aws keyspaces create-keyspace --keyspace-name mykeyspace
             \ --replication-specification
 replicationStrategy=MULTI_REGION,regionList=us-east-1,eu-west-1
```

> **Note**
>
> Al crear un espacio de claves multirregión, Amazon Keyspaces crea un rol vinculado a servicios con el nombre `AWSServiceRoleForAmazonKeyspacesReplication` en su cuenta. Este rol permite a Amazon Keyspaces replicar las escrituras en todas las réplicas de una tabla multirregión en su nombre. Para obtener más información, consulte the section called “Réplica multirregión”.

Creación de una nueva tabla multirregión con la configuración predeterminada (CLI)

Para crear una tabla multirregional con la configuración predeterminada, solo necesita especificar el esquema. Puede utilizar el siguiente ejemplo.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable
           \ --schema-definition 'allColumns=[{name=pk,type=int}],partitionKeys={name=
 pk}'
```

El resultado del comando es:

```
{
    "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable"
}
```

Para confirmar la configuración de la tabla, puede utilizar la siguiente sentencia.

```
aws keyspaces get-table --keyspace-name mykeyspace --table-name mytable
```

El resultado muestra todos los ajustes predeterminados de una tabla multirregional.

```
{
    "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
            }
        }
    ]
}
```

Crear una nueva tabla multirregional en modo aprovisionado con escalado automático (CLI)

Para crear una tabla de varias regiones en modo aprovisionado con una configuración de escalado automático, puede usar. AWS CLI Tenga en cuenta que debe usar el `create-table` comando CLI de Amazon Keyspaces para configurar los ajustes de escalado automático multirregión. Esto se debe a que Application Auto Scaling, el servicio que Amazon Keyspaces utiliza para realizar el escalado automático en su nombre, no admite varias regiones.

Para obtener más información sobre la configuración del escalado automático, la política de seguimiento de objetivos, el valor objetivo y la configuración opcional, consultethe section called “Cree una tabla nueva con escalado automático mediante el AWS CLI”.

Al crear una nueva tabla multirregión en modo aprovisionado con una configuración de escalado automático, puede especificar la configuración general de la tabla que es válida para todos los lugares en los Regiones de AWS que se replica la tabla. A continuación, puede sobrescribir la configuración de capacidad de lectura y leer la configuración de autoescalado de cada réplica. Sin embargo, la capacidad de escritura permanece sincronizada entre todas las réplicas para garantizar que haya suficiente capacidad para replicar las escrituras en todas las regiones.

Para definir la capacidad de lectura de una réplica de tabla en una región específica, puedes configurar los siguientes parámetros como parte de la tabla: `replicaSpecifications`

- La región
- Las unidades de capacidad de lectura aprovisionadas (opcionales)
- Configuración de escalado automático para la capacidad de lectura (opcional)

Al crear tablas aprovisionadas de varias regiones con ajustes de autoescalado complejos y diferentes configuraciones para las réplicas de tablas, resulta útil cargar los ajustes de escalado automático y las configuraciones de réplica de la tabla desde archivos JSON.

Para usar el siguiente ejemplo de código, puede descargar los archivos JSON de ejemplo desde auto-scaling.zip y extraer `auto-scaling.json` `yreplication.json`. Toma nota de la ruta de acceso a los archivos.

En este ejemplo, los archivos JSON se encuentran en el directorio actual. Para conocer las distintas opciones de ruta del archivo, consulte Carga de parámetros desde un archivo.

```
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
```

Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CLI)

Para actualizar el modo aprovisionado y la configuración de escalado automático de una tabla existente, puede usar el AWS CLI `update-table` comando.

Tenga en cuenta que debe usar los comandos CLI de Amazon Keyspaces para crear o modificar la configuración de escalado automático multirregional. Esto se debe a que Application Auto Scaling, el servicio que Amazon Keyspaces utiliza para realizar el escalado automático de la capacidad de las tablas en su nombre, no admite varios. Regiones de AWS

Al actualizar el modo aprovisionado o la configuración de escalado automático de una tabla de varias regiones, puede actualizar la configuración de capacidad de lectura y la configuración de escalado automático de lectura para cada réplica de la tabla.

Sin embargo, la capacidad de escritura permanece sincronizada entre todas las réplicas para garantizar que haya suficiente capacidad para replicar las escrituras en todas las regiones. Para actualizar la capacidad de lectura de una réplica de tabla en una región específica, puedes cambiar uno de los siguientes parámetros opcionales de la tabla: `replicaSpecifications`

- Las unidades de capacidad de lectura aprovisionadas (opcionales)
- Configuración de escalado automático para la capacidad de lectura (opcional)

Al actualizar tablas de varias regiones con ajustes de autoescalado complejos y diferentes configuraciones para réplicas de tablas, resulta útil cargar los ajustes de escalado automático y las configuraciones de réplica de la tabla desde archivos JSON.

Para usar el siguiente ejemplo de código, puede descargar los archivos JSON de ejemplo desde auto-scaling.zip y extraer `auto-scaling.json` y`replication.json`. Toma nota de la ruta de acceso a los archivos.

En este ejemplo, los archivos JSON se encuentran en el directorio actual. Para conocer las distintas opciones de ruta del archivo, consulte Carga de parámetros desde un archivo.

```
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
```

Visualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CLI)

Para ver la configuración de escalado automático de una tabla de varias regiones, puede utilizar la `get-table-auto-scaling-settings` operación. El siguiente comando CLI es un ejemplo de ello.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name
 mytable
```

Debería ver la siguiente salida.

```
{
    "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
                        }
                    }
                }
            }
        }
    ]
}
```

Desactivar el escalado automático para una tabla multirregión (CLI)

Puede usar el AWS CLI `update-table` comando para desactivar el escalado automático de una tabla existente. Tenga en cuenta que no puede desactivar el escalado automático para una réplica de tabla individual.

En el siguiente ejemplo, el escalado automático está desactivado para la capacidad de lectura de la tabla.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
           \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

> **Note**
>
> Para eliminar el rol vinculado al servicio que utiliza Application Auto Scaling, debe deshabilitar el escalado automático en todas las tablas de la cuenta. Regiones de AWS

Configuración manual de la capacidad aprovisionada de una tabla multirregional (CLI)

Si tiene que desactivar el escalado automático para una tabla de varias regiones, puede utilizarlo `update-table` para aprovisionar manualmente la capacidad de lectura de la tabla para una tabla de réplica.

```
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
```

> **Note**
>
> Recomendamos usar el escalado automático para las tablas de varias regiones que utilizan la capacidad aprovisionada. Para obtener más información, consulte the section called “Tablas multirregionales”.

Recuperación en un momento dado para Amazon Keyspaces (para Apache Cassandra)

La recuperación en un momento dado (PITR) le permite proteger sus tablas de Amazon Keyspaces de operaciones accidentales de escritura o eliminación al ofrecerle copias de seguridad continuas de los datos de sus tablas.

Por ejemplo, suponga que un script de prueba escriba accidentalmente en una tabla de Amazon Keyspaces de producción. Con la recuperación en un momento dado, puede restaurar los datos de esa tabla a cualquier segundo en el tiempo en los últimos 35 días desde que PITR fue habilitado. Si elimina una tabla con la recuperación en un momento dado habilitada, puede consultar los datos de la tabla eliminada durante 35 días (sin costo adicional) y restaurarla al estado en que se encontraba justo antes del momento dado de la eliminación.

Puede restaurar una tabla de Amazon Keyspaces a un punto en el tiempo mediante la consola, el SDK de AWS y la AWS Command Line Interface (AWS CLI), o Cassandra Query Language (CQL). Para obtener más información, consulte Restauración de una tabla de Amazon Keyspaces a un momento dado.

Las operaciones puntuales no tienen ningún impacto sobre el rendimiento o la disponibilidad de la tabla base, y la restauración de una tabla no consume rendimiento adicional.

Para obtener información sobre las cuotas de PITR, consulte Cuotas.

Para obtener información sobre precios, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Temas

- Cómo funciona point-in-time la recuperación en Amazon Keyspaces
- Restauración de una tabla de Amazon Keyspaces a un momento dado

Cómo funciona point-in-time la recuperación en Amazon Keyspaces

En esta sección se proporciona información general sobre cómo funciona la point-in-time recuperación de Amazon Keyspaces (PITR). Para obtener más información sobre precios, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Temas

- Habilitar la point-in-time recuperación (PITR)
- Permisos necesarios para restaurar una tabla
- Periodo de copias de seguridad continuas con PITR
- Ajustes de restauración de PITR
- Restauración de tablas cifradas con PITR
- Restauración de tablas multirregión con PITR
- Tiempo de restauración de la tabla con PITR
- PITR e integración de Amazon Keyspaces con servicios de AWS.

Habilitar la point-in-time recuperación (PITR)

Puede habilitar PITR mediante la consola o mediante programación.

Habilitación de PITR con la consola

La configuración de PITR para las tablas nuevas se puede administrar en la opción Configuración personalizada. De forma predeterminada, PITR está habilitada en las tablas nuevas creadas a través de la consola.

Para habilitar PITR para una tabla existente, realice los pasos siguientes.

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tablas y seleccione la tabla que desee editar.
3. En la pestaña Copias de seguridad, elija Editar.
4. En la sección Editar la configuración point-in-time de recuperación, selecciona Habilitar la recuperación de P. oint-in-time

Puede deshabilitar PITR en una tabla en cualquier momento; para ello, realice los siguientes pasos.

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tablas y seleccione la tabla que desee editar.
3. En la pestaña Copias de seguridad, elija Editar.

4. En la sección Editar la configuración de point-in-time recuperación, desactive la casilla Habilitar oint-in-time la recuperación de P.

> **Important**
>
> Al deshabilitar PITR se elimina de inmediato el historial de copias de seguridad, sin importar si vuelve a habilitar PITR en la tabla en el plazo de 35 días.

Para obtener información sobre cómo restaurar una tabla mediante la consola, consulte the section called “Restauración de una tabla a un momento dado (consola)”.

Habilitación de PITR con la AWS CLI

Puede administrar la configuración de PITR para las tablas con la API `UpdateTable`.

Al crear una nueva tabla con la AWS CLI, debe habilitar explícitamente PITR en la misma.

Para habilitar PITR al momento de crear una nueva tabla, puede utilizar el siguiente comando de la AWS CLI como ejemplo. El comando se ha dividido en líneas separadas para mejorar la legibilidad.

```
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**
>
> Si no se especifica ningún point-in-time valor de point-in-time recuperación, la recuperación está deshabilitada de forma predeterminada.

Para confirmar la configuración de point-in-time recuperación de una tabla, puede usar el siguiente AWS CLI comando.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Para habilitar PITR en una tabla existente mediante la AWS CLI, ejecute el siguiente comando.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=ENABLED'
```

Para deshabilitar PITR en una tabla existente, ejecute el siguiente comando de la AWS CLI.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=DISABLED'
```

Important

Al deshabilitar PITR se elimina de inmediato el historial de copias de seguridad, sin importar si vuelve a habilitar PITR en la tabla en un plazo de 35 días.

Habilitación de PITR con CQL

Puede administrar la configuración de PITR para las tablas mediante la propiedad personalizada `point_in_time_recovery`.

Al crear una nueva tabla con CQL, debe habilitar explícitamente PITR en la misma.

Para habilitar PITR al momento de crear una nueva tabla, puede utilizar el siguiente comando CQL como ejemplo.

```
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

Si no se especifica ninguna point-in-time propiedad personalizada de point-in-time recuperación, la recuperación está deshabilitada de forma predeterminada.

Para habilitar PITR en una tabla existente mediante CQL, ejecute el siguiente comando CQL.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'enabled'}}
```

Para deshabilitar PITR en una tabla existente, ejecute el siguiente comando CQL.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

> **Important**
>
> Al deshabilitar PITR se elimina de inmediato el historial de copias de seguridad, sin importar si vuelve a habilitar PITR en la tabla en un plazo de 35 días.

Para obtener más información, consulte the section called “CREATE TABLE” y the section called “ALTER TABLE” en la Referencia del lenguaje CQL. Para obtener información sobre cómo restaurar una tabla mediante CQL, consulte the section called “Restauración de una tabla a un momento dado con CQL”.

Permisos necesarios para restaurar una tabla

Para restaurar correctamente una tabla, el usuario o rol de IAM necesita los siguientes permisos mínimos:

- `cassandra:Restore`: la acción restaurar es necesaria para restaurar la tabla de destino.
- `cassandra:Select`: la acción seleccionar es necesaria para leer desde la tabla de origen.
- `cassandra:TagResource`: la acción etiquetar es opcional y solo se requiere si la operación de restauración añade etiquetas.

El siguiente es un ejemplo de una política que concede los permisos mínimos necesarios a un usuario para restaurar tablas en el espacio de claves `mykeyspace`.

```
{
   "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*"
         ]
      }
   ]
}
```

Es posible que se requieran permisos adicionales para restaurar una tabla en función de otras características seleccionadas. Por ejemplo, si la tabla de origen está cifrada en reposo con una clave administrada por el cliente, Amazon Keyspaces debe tener permisos para acceder a la clave administrada por el cliente de la tabla de origen a fin de restaurar correctamente la tabla. Para obtener más información, consulte the section called “PITR y tablas cifradas”.

Si utiliza políticas de IAM con claves de condición para restringir el tráfico entrante a orígenes específicos, debe asegurarse de que Amazon Keyspaces tenga permiso para realizar una operación de restauración en nombre de su entidad principal. Debe añadir una clave de condición `aws:ViaAWSService` a su política de IAM si su política restringe el tráfico entrante a cualquiera de los siguientes:

- Puntos de conexión de VPC con `aws:SourceVpce`
- Rangos IP con `aws:SourceIp`
- VPC con `aws:SourceVpc`

La clave de condición `aws:ViaAWSService` permite el acceso cuando cualquier servicio de AWS realiza una solicitud empleando las credenciales de la entidad principal. Para obtener más información, consulte Elementos JSON de la política de IAM: Clave de condición en la Guía del usuario de IAM.

A continuación se muestra un ejemplo de política que restringe el tráfico de origen a una dirección IP específica y permite a Amazon Keyspaces restaurar una tabla en nombre de la entidad principal.

```
{
```

```
   "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"
            }
         }
      }
   ]
}
```

Para ver un ejemplo de política que utilice la clave de condición global de `aws:ViaAWSService`, consulte the section called “Políticas de puntos de conexión de VPC y recuperación de Amazon Keyspaces point-in-time (PITR)”.

Periodo de copias de seguridad continuas con PITR

La PITR de Amazon Keyspaces utiliza dos marcas de tiempo para mantener el periodo de disponibilidad de copias de seguridad restaurables de una tabla.

- Hora para restauración más antigua: marca la hora de la copia de seguridad restaurable más antigua. La copia de seguridad restaurable más antigua se remonta hasta 35 días o hasta el

momento en que se habilitó PITR, lo que sea más reciente. El periodo máximo de 35 días de copias de seguridad no se puede modificar.

- Hora actual: la marca de tiempo de la última copia de seguridad restaurable es la hora actual. Si no se proporciona ninguna marca de tiempo durante una restauración, se utiliza la hora actual.

Cuando PITR está habilitada, puede restaurar a cualquier punto en el tiempo entre `EarliestRestorableDateTime` y `CurrentTime`. Solo puede restaurar los datos de una tabla a un momento en que la PITR estuviera habilitada.

Si deshabilita la PITR y más tarde la vuelve a habilitar, restablece la hora inicial de la primera copia de seguridad disponible a la hora en que se volvió a habilitar la PITR. Esto significa que al deshabilitar la PITR se elimina el historial de copias de seguridad.

> **Note**
>
> Las operaciones de lenguaje de definición de datos (DDL) en las tablas, al igual que los cambios de esquema, se realizan de forma asíncrona. Solo puede ver las operaciones completadas en los datos de su tabla restaurada, pero podría ver acciones adicionales en su tabla de origen si estaban en curso en el momento de la restauración. Para obtener una lista de las sentencias DDL, consulte the section called “Instrucciones DDL”.

Una tabla no necesita estar activa para poder restaurarla. También puede restaurar tablas eliminadas si PITR estaba habilitado en la tabla eliminada y la eliminación se produjo dentro del periodo de copia de seguridad (o en los últimos 35 días).

> **Note**
>
> Si se crea una nueva tabla con el mismo nombre cualificado (por ejemplo, mykeyspace.mytable) que una tabla previamente eliminada, la tabla eliminada ya no se puede restaurar. Si intenta hacerlo desde la consola, aparece una advertencia.

Ajustes de restauración de PITR

Al restaurar una tabla mediante la PITR, Amazon Keyspaces restaura el esquema y los datos de su tabla de origen al estado basado en la marca de tiempo seleccionada (`day:hour:minute:second`) en una nueva tabla. La PITR no sobrescribe las tablas existentes.

Además del esquema y los datos de la tabla, la PITR restaura las `custom_properties` de la tabla de origen. A diferencia de los datos de la tabla, que se restauran en función de la marca de tiempo seleccionada entre la hora de restauración más antigua y la hora actual, las propiedades personalizadas siempre se restauran en función de la configuración de la tabla según la hora actual.

La configuración de la tabla restaurada coincide con la configuración de la tabla de origen con la marca de tiempo de cuando se inició la restauración. Si desea sobrescribir estos ajustes durante la restauración, puede hacerlo utilizando `WITH custom_properties`. Las propiedades personalizadas incluyen los siguientes ajustes.

- Modo de capacidad de lectura/escritura
- Ajustes de capacidad de rendimiento aprovisionada
- Ajustes de PITR

Si la tabla está en modo de capacidad aprovisionada con el escalado automático activado, la operación de restauración también restaura la configuración de autoescalado de la tabla. Puede sobrescribirlos mediante el `autoscaling_settings` parámetro en CQL o `autoScalingSpecification` con la CLI. Para obtener más información sobre la configuración de escalado automático, consultethe section called “Administre la capacidad de rendimiento con el escalado automático”.

Cuando realiza una restauración de tabla completa, todos los valores de la tabla restaurada provienen de la configuración actual de la tabla de origen en el momento de la restauración.

Por ejemplo, suponga que el rendimiento aprovisionado de una tabla se acaba de reducir a 50 unidades de capacidad de lectura y 50 unidades de capacidad de escritura. A continuación, restaura el estado de la tabla a hace tres semanas. En ese momento, su rendimiento aprovisionado estaba establecido en 100 unidades de capacidad de lectura y 100 unidades de capacidad de escritura. En este caso, Amazon Keyspaces restaura los datos de su tabla a ese momento, pero utiliza la configuración de rendimiento aprovisionada actual (50 unidades de capacidad de lectura y 50 unidades de capacidad de escritura).

Los siguientes ajustes no se restauran y debe configurarlos manualmente para la nueva tabla.

- Políticas de AWS Identity and Access Management (IAM)
- CloudWatch Métricas y alarmas de Amazon
- Etiquetas (se pueden añadir a la sentencia CQL `RESTORE` utilizando `WITH TAGS`)

Restauración de tablas cifradas con PITR

Al restaurar una tabla mediante la PITR, Amazon Keyspaces restaura la configuración de cifrado de su tabla de origen. Si la tabla estaba cifrada con una Clave propiedad de AWS (predeterminada), la tabla se restaura de manera automática con la misma configuración. Si la tabla que desea restaurar se cifró con una clave administrada por el cliente, es necesario que Amazon Keyspaces pueda acceder a la misma clave administrada por el cliente para restaurar los datos de la tabla.

Puede cambiar la configuración de cifrado de la tabla en el momento de la restauración. Para cambiar de una Clave propiedad de AWS a una clave administrada por el cliente, debe proporcionar una clave administrada por el cliente válida y accesible en el momento de la restauración.

Si desea cambiar de una clave administrada por el cliente a una Clave propiedad de AWS, confirme que Amazon Keyspaces tenga acceso a la clave administrada por el cliente de la tabla de origen para restaurar la tabla con una Clave propiedad de AWS. Para obtener más información sobre la configuración del cifrado en reposo de tablas, consulte the section called “Cómo funciona”.

> **Note**
>
> Si la tabla se eliminó porque Amazon Keyspaces perdió el acceso a su clave administrada por el cliente, debe asegurarse de que Amazon Keyspaces tenga acceso a la clave administrada por el cliente antes de intentar restaurar la tabla. Una tabla que se cifró con una clave administrada por el cliente no puede restaurarse si Amazon Keyspaces no tiene acceso a dicha clave. Para obtener más información, consulte Solución de problemas de acceso a claves en la Guía para desarrolladores de AWS Key Management Service.

Restauración de tablas multirregión con PITR

Puede restaurar una tabla multirregión con PITR. Para que la operación de restauración tenga éxito, tanto la tabla de origen como la de destino tienen que estar replicadas en las mismas Regiones de AWS.

Amazon Keyspaces restaura la configuración de la tabla de origen en cada una de las regiones replicadas que forman parte del espacio de claves. También puede anular los ajustes durante la operación de restauración. Para obtener más información sobre los ajustes que se pueden modificar durante la restauración, consulte the section called “Ajustes de restauración”.

Para obtener más información sobre la réplica multirregión, consulte the section called “Cómo funciona”.

Tiempo de restauración de la tabla con PITR

El tiempo necesario para restaurar una tabla depende de múltiples factores y no siempre se correlaciona directamente con el tamaño de la tabla.

A continuación se exponen algunas consideraciones sobre tiempos de restauración.

- Las copias de seguridad se restauran en una tabla nueva. Es posible que la demora para realizar todas las acciones destinadas a crear una tabla nueva e iniciar el proceso de restauración sea de hasta 20 minutos (aunque la tabla esté vacía).
- El tiempo de restauración para tablas grandes con modelos de datos bien distribuidos puede ser de varias horas o más.
- Si su tabla de origen contiene datos muy sesgados, el tiempo de restauración podría ser mayor. Por ejemplo, si la clave principal de su tabla utiliza el mes del año como clave de partición y todos sus datos son del mes de diciembre, tiene datos sesgados.

Una práctica recomendada a la hora de planificar la recuperación de desastres es documentar regularmente los tiempos promedio de finalización de la restauración y establecer cómo estos tiempos afectan a su objetivo general de tiempo de recuperación.

PITR e integración de Amazon Keyspaces con servicios de AWS.

Las siguientes operaciones de PITR se registran con AWS CloudTrail a fin de habilitar el monitoreo y la auditoría continuos.

- Crear una nueva tabla con PITR habilitado o deshabilitado.
- Habilitar o deshabilitar PITR en una tabla existente.
- Restaurar una tabla activa o eliminada.

Para obtener más información, consulte Registro de llamadas a la API de Amazon Keyspaces con AWS CloudTrail.

Puede realizar las siguientes acciones de PITR con AWS CloudFormation.

- Crear una nueva tabla con PITR habilitado o deshabilitado.

- Habilitar o deshabilitar PITR en una tabla existente.

Para obtener más información, consulte Referencia de tipos de recursos de Cassandra en la Guía del usuario de AWS CloudFormation.

Restauración de una tabla de Amazon Keyspaces a un momento dado

La recuperación en un momento dado (PITR) de Amazon Keyspaces (para Apache Cassandra) le permite restaurar los datos de una tabla de Amazon Keyspaces a cualquier momento dado dentro de los últimos 35 días. En la primera parte de este tutorial le mostramos cómo restaurar una tabla a un momento dado con la consola de Amazon Keyspaces, la AWS Command Line Interface (AWS CLI) y Cassandra Query Language (CQL). En la segunda parte le mostramos cómo restaurar una tabla eliminada con la AWS CLI y CQL.

Temas

- Antes de empezar
- Restauración de una tabla a un momento dado (consola)
- Restauración de una tabla a un momento dado con la AWS CLI
- Restauración de una tabla a un momento dado con CQL
- Restauración de una tabla eliminada con la AWS CLI
- Restauración de una tabla eliminada con CQL

Antes de empezar

Si aún no lo ha hecho, debe configurar los permisos apropiados para que el usuario restaure las tablas de Amazon Keyspaces. En AWS Identity and Access Management (IAM), la política administrada de AWS `AmazonKeyspacesFullAccess` incluye los permisos para restaurar tablas de Amazon Keyspaces. Para conocer los pasos detallados a fin de implementar una política con los permisos mínimos necesarios, consulte the section called “Permisos de restauración”.

Restauración de una tabla a un momento dado (consola)

En el siguiente ejemplo se muestra cómo utilizar la consola de Amazon Keyspaces para restaurar una tabla existente llamada `mytable` a un momento dado.

> **Note**
>
> En este procedimiento se presupone que ha habilitado la recuperación a un momento dado. Para habilitar PITR para la tabla `mytable`, siga los pasos que se indican en the section called “Uso de la consola”.

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación del lado izquierdo de la consola, elija Tables (Tablas).
3. En la lista de tablas, elija la tabla `mytable`.
4. En la pestaña Copias de seguridad de la tabla `mytable`, en la sección Recuperación en un momento dado, elija Restaurar.
5. Para el nombre de tabla nuevo, escriba **`mytable_restored`**.
6. Para definir el momento dado para la operación de restauración, puede elegir entre dos opciones:
   - Seleccione la hora Más antigua preconfigurada.
   - Seleccione Especificar fecha y hora e introduzca la fecha y hora a la que desea restaurar la nueva tabla.

   > **Note**
   >
   > Puede restaurar a cualquier momento dado entre la hora Más antigua y la hora actual. Amazon Keyspaces restaura los datos de su tabla al estado basado en la fecha y hora seleccionadas (day:hour:minute:second).
7. Elija Restaurar para iniciar el proceso de restauración.

   La tabla que se va a restaurar se muestra con el estado Restoring (Restaurándose). Una vez finalizado el proceso de restauración, el estado de la tabla `mytable_restored` cambia a Active (Activa).

   > **Important**
   >
   > Mientras haya una restauración en curso, no modifique ni elimine las políticas de AWS Identity and Access Management (IAM) que le conceden a la entidad de (usuario,

grupo o rol) permiso para restaurar. De lo contrario, puede ocurrir un comportamiento inesperado. Por ejemplo, suponga que ha eliminado los permisos de escritura de una tabla mientras se estaba restaurando. En ese caso, la operación subyacente `RestoreTableToPointInTime` no puede escribir ninguno de los datos restaurados en la tabla.
Puede modificar o eliminar los permisos solo después de que finalice la operación de restauración.

Restauración de una tabla a un momento dado con la AWS CLI

En el siguiente procedimiento, se muestra cómo utilizar la AWS CLI para restaurar una tabla llamada `myTable` a un momento dado.

1. En el primer paso, crea una tabla sencilla llamada `myTable` con la PITR habilitada. El comando se ha dividido en líneas separadas para mejorar la legibilidad.

```
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. Confirme las propiedades de la nueva tabla y revise la `earliestRestorableTimestamp` de la PITR.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Este comando devuelve la siguiente salida.

```
{
    "keyspaceName": "myKeyspace",
    "tableName": "myTable",
    "resourceArn": "arn:aws:cassandra:us-east-1:111222333444:/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": ""
    }
}
```

Puede restaurar una tabla activa a cualquier momento dado entre la `earliestRestorableTimestamp` y la hora actual en intervalos de un segundo. El valor predeterminado es la hora actual.

3. Para restaurar una tabla a un momento dado, especifique una `restore_timestamp` en formato ISO 8601. Puede elegir cualquier momento dado durante los últimos 35 días en

intervalos de un segundo. Por ejemplo, el siguiente comando restaura la tabla al a fecha y hora del valor `EarliestRestorableDateTime`.

```
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"
```

La salida de este comando devuelve el ARN de la tabla restaurada.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/
myKeyspace/table/myTable_restored"
}
```

Para restaurar la tabla a la hora actual, puede omitir la `restore-timestamp`.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored1'"
```

Important

Mientras haya una restauración en curso, no modifique ni elimine las políticas de AWS Identity and Access Management (IAM) que le conceden a la entidad de (usuario, grupo o rol) permiso para restaurar. De lo contrario, puede ocurrir un comportamiento inesperado. Por ejemplo, suponga que ha eliminado los permisos de escritura de una tabla mientras se estaba restaurando. En ese caso, la operación subyacente `RestoreTableToPointInTime` no puede escribir ninguno de los datos restaurados en la tabla.
Puede modificar o eliminar los permisos solo después de que finalice la operación de restauración.

Restauración de una tabla a un momento dado con CQL

En el siguiente procedimiento, se muestra cómo utilizar CQL para restaurar una tabla llamada `mytable` a un momento dado.

> **Note**
>
> En este procedimiento se presupone que ha habilitado la recuperación a un momento dado. Para habilitar PITR para la tabla , siga los pasos que se indican en the section called “CQL”.

1. Puede restaurar una tabla activa a un momento dado entre la `earliest_restorable_timestamp` y la hora actual. El valor predeterminado es la hora actual.

   Para confirmar que la recuperación en un momento dado está habilitada para la tabla `mytable`, consulte `system_schema_mcs.tables` como se indica a continuación.

   ```
   SELECT custom_properties
   FROM system_schema_mcs.tables
   WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
   ```

   La recuperación en un momento dado está habilitada, como se muestra en el siguiente ejemplo de salida.

   ```
   custom_properties
   -----------------
   {
     ...,
       "point_in_time_recovery": {
       "earliest_restorable_timestamp":"2020-06-30T19:19:21.175Z"
       "status":"enabled"
     }
   }
   ```

2. Restaure la tabla a un momento dado, especificado por una `restore_timestamp` en formato ISO 8601. En este caso, la tabla `mytable` se restaura a la hora actual. Puede omitir la cláusula `WITH restore_timestamp = ...`. Sin la cláusula, se utiliza la marca de tiempo actual.

   ```
   RESTORE TABLE mykeyspace.mytable_restored
   FROM TABLE mykeyspace.mytable;
   ```

También puede restaurarla a un momento dado específico. Puede especificar cualquier momento de los últimos 35 días. Por ejemplo, el siguiente comando restaura la tabla al a fecha y hora del valor `EarliestRestorableDateTime`.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable
WITH restore_timestamp = '2020-06-30T19:19:21.175Z';
```

Para obtener una descripción completa de la sintaxis, consulte the section called “RESTORE TABLE” en la referencia del lenguaje.

Para comprobar que la restauración de la tabla se haya realizado correctamente, consulte `system_schema_mcs.tables` para confirmar el estado de la tabla.

```
SELECT status
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable_restored'
```

La consulta muestra la siguiente salida.

```
status
------
RESTORING
```

La tabla que se va a restaurar se muestra con el estado Restoring (Restaurándose). Una vez finalizado el proceso de restauración, el estado de la tabla `mytable_restored` cambia a Active (Activa).

Important

Mientras haya una restauración en curso, no modifique ni elimine las políticas de AWS Identity and Access Management (IAM) que le conceden a la entidad de (usuario, grupo o rol) permiso para restaurar. De lo contrario, puede ocurrir un comportamiento inesperado. Por ejemplo, suponga que ha eliminado los permisos de escritura de una tabla mientras se estaba restaurando. En ese caso, la operación subyacente `RestoreTableToPointInTime` no puede escribir ninguno de los datos restaurados en la tabla.

Puede modificar o eliminar los permisos solo después de que finalice la operación de restauración.

Restauración de una tabla eliminada con la AWS CLI

En el siguiente procedimiento se muestra cómo utilizar la AWS CLI para restaurar una tabla eliminada llamada `myTable` al momento de su eliminación.

> **Note**
>
> En este procedimiento se asume que la PITR estaba habilitado en la tabla eliminada.

1. Elimine la tabla que creó en el tutorial anterior.

```
aws keyspaces delete-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

2. Restaure la tabla eliminada al momento de su eliminación con el siguiente comando.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
 'myTable_restored2'
```

La salida de este comando devuelve el ARN de la tabla restaurada.

```
{
    "restoredTableARN": "arn:aws:cassandra:us-east-1:111222333444:/keyspace/
myKeyspace/table/myTable_restored2"
}
```

Restauración de una tabla eliminada con CQL

En el siguiente procedimiento se muestra cómo utilizar CQL para restaurar una tabla eliminada llamada `mytable` al momento de su eliminación.

> **Note**
>
> En este procedimiento se asume que la PITR estaba habilitado en la tabla eliminada.

1. Para confirmar si la recuperación en un momento dado está habilitada para una tabla eliminada, consulte la tabla del sistema. Solo se muestran las tablas con recuperación en un momento dado habilitada.

```
SELECT custom_properties
FROM system_schema_mcs.tables_history
WHERE keyspace_name = 'mykeyspace' AND table_name = 'my_table';
```

   La consulta muestra la siguiente salida.

```
custom_properties
------------------
{
    ...,
   "point_in_time_recovery":{
      "restorable_until_time":"2020-08-04T00:48:58.381Z",
      "status":"enabled"
   }
}
```

2. Restaure la tabla al momento de su eliminación con el siguiente ejemplo de comando.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

Caducidad de datos con Periodo de vida (TTL) de Amazon Keyspaces

Periodo de vida (TTL) de Amazon Keyspaces (para Apache Cassandra) le ayuda a simplificar la lógica de su aplicación y a optimizar el precio de almacenamiento al expirar automáticamente los datos de las tablas. Los datos que ya no necesite se eliminan automáticamente de su tabla en función del valor de tiempo de vida que establezca. Esto facilita el cumplimiento de las políticas de retención de datos basadas en requisitos empresariales, industriales o normativos que definen cuánto tiempo deben conservarse los datos o especifican cuándo deben eliminarse.

Por ejemplo, puede utilizar TTL en una aplicación AdTech para programar cuándo caducan los datos de anuncios específicos y dejan de ser visibles para los clientes. También puede utilizar TTL para retirar automáticamente datos antiguos y ahorrar en costos de almacenamiento. Puede establecer un valor de TTL predeterminado para toda la tabla y sobrescribir ese valor para filas y columnas individuales. Las operaciones de TTL no afectan al rendimiento de su aplicación. Además, el número de filas y columnas marcadas para caducar con TTL no afecta a la disponibilidad de su tabla.

Amazon Keyspaces filtra automáticamente los datos caducados para que estos no se devuelvan en resultados de consultas o estén disponibles para su uso en instrucciones de lenguaje de manipulación de datos (DML). Amazon Keyspaces suele eliminar los datos caducados del almacenamiento en un plazo de 10 días a partir de la fecha de caducidad. En raras ocasiones, es posible que Amazon Keyspaces no pueda eliminar los datos en el plazo de 10 días si existe una actividad sostenida en la partición de almacenamiento subyacente para proteger la disponibilidad. En estos casos, Amazon Keyspaces sigue intentando eliminar los datos caducados una vez que disminuya el tráfico en la partición. Una vez eliminados de forma permanente los datos del almacenamiento, deja de incurrir en gastos de almacenamiento. Para obtener más información, consulte the section called “Cómo funciona”.

Puede establecer, modificar o deshabilitar la configuración de TTL predeterminada para tablas nuevas y existentes mediante la consola o Cassandra Query Language (CQL). En las tablas con TTL predeterminado configurado, puede utilizar Cassandra Query Language (CQL) para anular la configuración TTL predeterminada y aplicar valores de TTL personalizados a filas y columnas. Para obtener más información, consulte the section called “Cómo utilizar Periodo de vida”.

Los precios de TTL se basan en el tamaño de las filas que se eliminen o actualicen al utilizar periodo de vida. Las operaciones de TTL se miden en unidades de `TTL deletes`. Se consume una eliminación TTL por KB de datos por fila que se elimine o actualice. Por ejemplo, para actualizar

una fila que almacena 2,5 KB de datos y eliminar una o más columnas dentro de la fila al mismo tiempo se requieren tres eliminaciones TTL. O bien, para eliminar una fila entera que contiene 3,5 KB de datos se requieren cuatro eliminaciones TTL. Se consume una eliminación TTL por cada KB de datos eliminados por fila. Para obtener más información sobre precios, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Temas

- Cómo funciona: Periodo de vida (TTL) de Amazon Keyspaces
- Cómo utilizar Periodo de vida (TTL)

Cómo funciona: Periodo de vida (TTL) de Amazon Keyspaces

Periodo de vida (TTL) de Amazon Keyspaces está totalmente administrado. No tiene que administrar configuraciones de sistema de bajo nivel, como las estrategias de compactación. Los datos caducan en el momento que especifique y Amazon Keyspaces elimina los datos caducados automáticamente (normalmente en un plazo de 10 días) sin afectar al rendimiento o la disponibilidad de su aplicación.

Los datos caducados se marcan para su eliminación y no están disponibles para las instrucciones de lenguaje de manipulación de datos (DML). Mientras siga realizando lecturas y escrituras en filas que contengan datos caducados, estos seguirán contando para las unidades de capacidad de lectura (RCU) y las unidades de capacidad de escritura (WCU) hasta que se eliminen del almacenamiento.

Temas

- Establecimiento del valor de TTL predeterminado para una tabla
- Establecimiento de valores de TTL personalizados para filas y columnas
- Habilitación de TTL en tablas
- Tiempo de vida de Amazon Keyspaces e integración con servicios de AWS.

Establecimiento del valor de TTL predeterminado para una tabla

En Amazon Keyspaces, puede establecer un valor de TTL predeterminado para todas las filas de una tabla al crearla. También puede editar una tabla existente para establecer o cambiar el valor de TTL predeterminado para las nuevas filas insertadas en la tabla. Cambiar el valor de TTL predeterminado de una tabla no modifica el valor de TTL de ningún dato existente en la tabla. El valor de TTL predeterminado de una tabla es cero, lo que significa que los datos no caducan

automáticamente. Si el valor de TTL predeterminado de una tabla es superior a cero, se añade una marca de tiempo de caducidad a cada fila.

Amazon Keyspaces calcula una nueva marca de tiempo de TTL cada vez que se actualizan los datos. Los valores de TTL se establecen en segundos y el valor máximo configurable es de 630 720 000 segundos, lo que equivale a 20 años. Para obtener más información sobre cómo establecer, modificar y deshabilitar el valor de TTL predeterminado de las tablas con la AWS Management Console o CQL, consulte the section called “Cómo utilizar Periodo de vida”.

Establecimiento de valores de TTL personalizados para filas y columnas

> **Note**
>
> Antes de establecer valores de TTL personalizados para filas y columnas, primero se debe habilitar TTL en la tabla. Para obtener más información, consulte the section called “Cómo habilitar el periodo de vida (TTL) en tablas existentes mediante propiedades personalizadas”.

Para sobrescribir el valor de TTL predeterminado de una tabla o establecer fechas de caducidad para filas individuales, puede utilizar las siguientes instrucciones de lenguaje de manipulación de datos (DML) CQL:

- `INSERT`: se utiliza para insertar una nueva fila de datos con un valor de TTL establecido.
- `UPDATE`: se utiliza para modificar una fila de datos existente con un nuevo valor de TTL.

El establecimiento de valores de TTL para las filas tiene prioridad sobre los ajustes de TTL predeterminados para la tabla.

Para ver la sintaxis CQL y ejemplos, consulte the section called “Para utilizar `INSERT` para editar la configuración personalizada de periodo de vida (TTL) mediante CQL”.

Para sobrescribir o establecer valores de TTL para columnas individuales, puede actualizar los ajustes de TTL para un subconjunto de columnas dentro de las filas existentes mediante la siguiente instrucción DML de CQL:

- `UPDATE`: se utiliza para actualizar una columna de datos.

El establecimiento de valores de TTL para las columnas tiene prioridad sobre los ajustes de TTL predeterminados para la tabla y sobre cualquier ajuste de TTL personalizado para la fila. Para ver la sintaxis CQL y ejemplos, consulte the section called “Para utilizar UPDATE para editar la configuración personalizada de periodo de vida (TTL) mediante CQL”.

Habilitación de TTL en tablas

TTL se habilita automáticamente para las tablas al especificar un valor `default_time_to_live` mayor que 0 en las instrucciones `CREATE TABLE` o `ALTER TABLE`. Si no especifica un `default_time_to_live` para la tabla, pero desea especificar valores de TTL personalizados para filas o columnas con las operaciones `INSERT` o `UPDATE`, primero debe habilitar TTL para la tabla. Puede habilitar TTL para una tabla mediante la propiedad personalizada `ttl`.

Al habilitar TTL en una tabla, Amazon Keyspaces comienza a almacenar metadatos adicionales relacionados con TTL para cada fila. Además, TTL utiliza marcas de tiempo de caducidad para hacer un seguimiento de cuándo caducan las filas o columnas. Las marcas de tiempo se almacenan como metadatos de fila y contribuyen al costo de almacenamiento de la fila.

Una vez habilitada la característica TTL, no puede deshabilitarla para una tabla. Establecer el `default_time_to_live` de la tabla en 0 deshabilita los tiempos de caducidad predeterminados para los datos nuevos, pero no desactiva la característica TTL ni revierte la tabla a los metadatos de almacenamiento originales de Amazon Keyspaces o al comportamiento de escritura.

Tiempo de vida de Amazon Keyspaces e integración con servicios de AWS.

La siguiente métrica TTL está disponible en Amazon CloudWatch para habilitar la monitorización continua.

- `TTLDeletes`: las unidades consumidas para eliminar o actualizar datos en una fila utilizando Tiempo de vida (TTL).

Para obtener más información sobre cómo monitorear las métricas de CloudWatch, consulte the section called “Monitorización con CloudWatch”.

Al utilizar AWS CloudFormation, puede activar TTL al crear una tabla de Amazon Keyspaces. Para obtener más información, consulte la Guía del usuario de AWS CloudFormation.

Cómo utilizar Periodo de vida (TTL)

Puede utilizar la consola de Amazon Keyspaces (para Apache Cassandra) o CQL para habilitar, actualizar y deshabilitar la configuración de periodo de vida.

Temas

- Para crear una nueva tabla con la configuración de periodo de vida (TTL) predeterminada habilitada (consola)
- Para actualizar la configuración predeterminada de periodo de vida (TTL) en tablas existentes (consola)
- Para deshabilitar la configuración predeterminada de periodo de vida (TTL) en tablas existentes (consola)
- Para crear una nueva tabla con la configuración de periodo de vida (TTL) predeterminada habilitada mediante CQL
- Para utilizar ALTER TABLE para editar la configuración predeterminada de periodo de vida (TTL) mediante CQL
- Cómo habilitar el periodo de vida (TTL) en tablas nuevas mediante propiedades personalizadas
- Cómo habilitar el periodo de vida (TTL) en tablas existentes mediante propiedades personalizadas
- Para utilizar INSERT para editar la configuración personalizada de periodo de vida (TTL) mediante CQL
- Para utilizar UPDATE para editar la configuración personalizada de periodo de vida (TTL) mediante CQL

Para crear una nueva tabla con la configuración de periodo de vida (TTL) predeterminada habilitada (consola)

Siga estos pasos para crear una nueva tabla con la configuración de periodo de vida habilitada con la consola de Amazon Keyspaces.

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tables (Tablas) y, a continuación, seleccione Create table (Crear tabla).

3. En la página Crear tabla, en la sección Detalles de la tabla, seleccione un espacio de claves y proporcione un nombre para la nueva tabla.
4. En la sección Esquema, cree el esquema para su tabla.
5. En la sección Configuración de la tabla, elija Personalizar configuración.
6. Continúe en Periodo de vida (TTL).

   En este paso, seleccione la configuración TTL predeterminada para la tabla.

   En Periodo TTL predeterminado, introduzca el tiempo de caducidad y elija la unidad de tiempo introducida, por ejemplo segundos, días o años. Amazon Keyspaces almacena el valor en segundos.
7. Elija Crear tabla. Su tabla se crea con el valor TTL predeterminado especificado.

> **Note**
>
> Puede sobrescribir el valor TTL predeterminado de la tabla para filas o columnas específicas utilizando el lenguaje de manipulación de datos (DML) en el editor CQL.

Para actualizar la configuración predeterminada de periodo de vida (TTL) en tablas existentes (consola)

Siga estos pasos para actualizar la configuración de periodo de vida (TTL) para tablas existentes con la consola de Amazon Keyspaces.

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. Elija la tabla que desee actualizar y luego la pestaña Configuración adicional.
3. Vaya a Periodo de vida (TTL) y elija Editar.
4. En Periodo TTL predeterminado, introduzca el tiempo de caducidad y elija la unidad de tiempo introducida, por ejemplo segundos, días o años. Amazon Keyspaces almacena el valor en segundos. Esto no cambia el valor de TTL de las filas existentes.
5. Una vez definida la configuración de TTL, elija Guardar cambios.

Para deshabilitar la configuración predeterminada de periodo de vida (TTL) en tablas existentes (consola)

Siga estos pasos para deshabilitar la configuración de periodo de vida (TTL) para tablas existentes con la AWS Management Console de Amazon Keyspaces.

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. Elija la tabla que desee actualizar y luego la pestaña Configuración adicional.
3. Vaya a Periodo de vida (TTL) y elija Editar.
4. Seleccione Período TTL predeterminado y establezca el valor en cero. Esto deshabilita TTL en la tabla de forma predeterminada para datos futuros. Esto no cambia el valor de TTL de las filas existentes.
5. Una vez definida la configuración de TTL, elija Guardar cambios.

Para crear una nueva tabla con la configuración de periodo de vida (TTL) predeterminada habilitada mediante CQL

Active TTL al crear una tabla nueva con el valor de TTL predeterminado establecido en 3 024 000 segundos, lo que representa 35 días.

```
CREATE TABLE my_table (
                userid uuid,
                time timeuuid,
                subject text,
                body text,
                user inet,
                PRIMARY KEY (userid, time)
                ) WITH default_time_to_live = 3024000;
```

Para confirmar la configuración de TTL para la nueva tabla, utilice la instrucción `cqlsh describe` como se muestra en el siguiente ejemplo. El resultado muestra la configuración de TTL predeterminada para la tabla como `default_time_to_live`.

```
describe my_table;
```

Para utilizar **ALTER TABLE** para editar la configuración predeterminada de periodo de vida (TTL) mediante CQL

Actualice la configuración de TTL para la tabla existente a 2 592 000 segundos, lo que representa 30 días.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Para confirmar la configuración de TTL para la tabla actualizada, utilice la instrucción `cqlsh describe` como se muestra en el siguiente ejemplo. El resultado muestra la configuración de TTL predeterminada para la tabla como `default_time_to_live`.

```
describe my_table;
```

Cómo habilitar el periodo de vida (TTL) en tablas nuevas mediante propiedades personalizadas

Para habilitar la configuración personalizada de periodo de vida que se puede aplicar a filas y columnas sin habilitar la configuración predeterminada de TTL para toda la tabla, puede utilizar la siguiente instrucción CQL.

```
CREATE TABLE my_keyspace.my_table (id int primary key) WITH CUSTOM_PROPERTIES={'ttl':
{'status': 'enabled'}};
```

Una vez habilitado, `ttl` no se puede deshabilitar para la tabla.

Cómo habilitar el periodo de vida (TTL) en tablas existentes mediante propiedades personalizadas

Para habilitar la configuración personalizada de periodo de vida que se puede aplicar a filas y columnas sin habilitar la configuración predeterminada de TTL para toda la tabla, puede utilizar la siguiente instrucción CQL.

```
ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

Una vez habilitado, `ttl` no se puede deshabilitar para la tabla.

Para utilizar **INSERT** para editar la configuración personalizada de periodo de vida (TTL) mediante CQL

La siguiente instrucción CQL inserta una fila de datos en la tabla y cambia la configuración predeterminada de TTL a 259 200 segundos (lo que equivale a 3 días).

```
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;
```

Para confirmar la configuración de TTL para la fila insertada, utilice la siguiente instrucción.

```
SELECT TTL (subject) from my_table;
```

Para utilizar **UPDATE** para editar la configuración personalizada de periodo de vida (TTL) mediante CQL

Para cambiar la configuración de TTL de la columna “subject” insertada anteriormente de 259 200 segundos (3 días) a 86 400 segundos (un día), utilice la siguiente instrucción.

```
UPDATE my_table USING TTL 86400 set subject = 'Updated Message' WHERE userid =
 B79CB3BA-745E-5D9A-8903-4A02327A7E09 and time = 96a29100-5e25-11ec-90d7-b5d91eceda0a;
```

Puede ejecutar una consulta select simple para ver el registro actualizado antes del tiempo de caducidad.

```
SELECT * from my_table;
```

La consulta muestra la siguiente salida.

```
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
```

Para confirmar que la caducidad tuvo éxito, vuelva a ejecutar la misma consulta después del tiempo de caducidad configurado.

```
SELECT * from my_table;
```

La consulta muestra la siguiente salida después de que la columna “subject” haya caducado.

```
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
```

Funcionamiento de las marcas de tiempo del lado del cliente en Amazon Keyspaces

En Amazon Keyspaces, las marcas de tiempo del lado del cliente son marcas de tiempo compatibles con Cassandra que se mantienen para cada celda de su tabla. Puede utilizar las marcas de tiempo del lado del cliente para resolver conflictos al permitir que sus aplicaciones cliente determinen el orden de las escrituras. Por ejemplo, cuando los clientes de una aplicación distribuida globalmente realizan actualizaciones de los mismos datos, las marcas de tiempo del lado del cliente conservan el orden en que se realizaron las actualizaciones en los clientes. Amazon Keyspaces utiliza estas marcas de tiempo para procesar las escrituras. Para obtener más información, consulte the section called “Cómo funciona”.

Una vez activadas las marcas de tiempo del lado del cliente para una tabla, puede especificar una marca de tiempo con la cláusula `USING TIMESTAMP` en su consulta CQL en lenguaje de manipulación de datos (DML). Si no especifica una marca de tiempo en su consulta CQL, Amazon Keyspaces utiliza la marca de tiempo transferida por su controlador de cliente. Si el controlador de cliente no suministra marcas de tiempo, Amazon Keyspaces asigna automáticamente una marca de tiempo a nivel de celda. Para consultar las marcas de tiempo, puede utilizar la función `WRITETIME` en su instrucción DML. Para obtener más información, consulte the section called “Uso de marcas de tiempo del lado del cliente”.

Amazon Keyspaces no cobra cargos adicionales por activar las marcas de tiempo del lado del cliente. Sin embargo, con las marcas de tiempo del lado del cliente se almacenan y escriben datos adicionales para cada valor de la fila. Esto puede conllevar un uso adicional de almacenamiento y, en algunos casos, de rendimiento. Para obtener más información en cuanto a estimación del impacto en el tamaño de las filas, consulte the section called “Cómo funciona”. Para obtener más información sobre precios de Amazon Keyspaces, consulte Precios de Amazon Keyspaces (para Apache Cassandra).

Temas

- Funcionamiento de las marcas de tiempo del lado del cliente en Amazon Keyspaces
- Uso de marcas de tiempo del lado del cliente en Amazon Keyspaces

Funcionamiento de las marcas de tiempo del lado del cliente en Amazon Keyspaces

Las marcas de tiempo del lado del cliente de Amazon Keyspaces están totalmente administradas. No tiene que administrar configuraciones de sistema de bajo nivel, como las estrategias de saneamiento y compactación.

Al eliminar datos, las filas se marcan para su eliminación con una lápida. Amazon Keyspaces elimina los datos marcados automáticamente (normalmente en un plazo de 10 días) sin afectar al rendimiento o la disponibilidad de su aplicación. Los datos lapidados no están disponibles para las instrucciones de lenguaje de manipulación de datos (DML). Si sigue realizando lecturas y escrituras en filas que contengan datos lapidados, estos siguen contando para el almacenamiento, las unidades de capacidad de lectura (RCU) y las unidades de capacidad de escritura (WCU) hasta que se eliminen del almacenamiento.

Temas

- Funcionamiento de las marcas de tiempo del lado del cliente en Amazon Keyspaces
- Marcas de tiempo del lado del cliente de Amazon Keyspaces e integración con servicios de AWS.

Funcionamiento de las marcas de tiempo del lado del cliente en Amazon Keyspaces

Cuando las marcas de tiempo del lado del cliente están activadas en Amazon Keyspaces, cada columna de cada fila almacena una marca de tiempo. Estas marcas de tiempo ocupan aproximadamente entre 20 y 40 bytes (dependiendo de sus datos) y contribuyen al costo de almacenamiento y rendimiento de la fila. Estos bytes de metadatos también cuentan para su cuota de tamaño de fila de 1 MB. Para determinar el aumento global de espacio de almacenamiento (a fin de garantizar que el tamaño de la fila se mantenga por debajo de 1 MB), tenga en cuenta el número de columnas de su tabla y el número de elementos de la colección en cada fila. Por ejemplo, si una tabla tiene 20 columnas y cada columna almacena 40 bytes de datos, el tamaño de la fila aumenta de 800 bytes a 1200 bytes. Para obtener más información sobre cómo calcular el tamaño de una fila, consulte the section called “Cálculo del tamaño de las filas”. Además de los 400 bytes adicionales para el almacenamiento, en este ejemplo, el número de unidades de capacidad de escritura (WCU) consumidas por escritura aumenta de 1 WCU a 2 WCU. Para obtener más información sobre cómo calcular la capacidad de lectura y escritura, consulte the section called “Modos de capacidad de lectura/escritura”.

Una vez activadas las marcas de tiempo del lado del cliente para una tabla, no puede desactivarlas. Además, las marcas de tiempo no pueden ser NULL, por lo que si la instrucción CQL o el controlador del cliente no proporcionan marcas de tiempo del lado del cliente, se añadirá automáticamente una marca de tiempo generada por Amazon Keyspaces.

Marcas de tiempo del lado del cliente de Amazon Keyspaces e integración con servicios de AWS.

La siguiente métrica de marcas de tiempo del lado del cliente está disponible en Amazon CloudWatch para habilitar la monitorización continua.

- `SystemReconciliationDeletes`: el número de operaciones de eliminación necesarias para eliminar los datos lapidados.

Para obtener más información sobre cómo monitorear las métricas de CloudWatch, consulte the section called “Monitorización con CloudWatch”.

Uso de marcas de tiempo del lado del cliente en Amazon Keyspaces

Puede utilizar la consola de Amazon Keyspaces (para Apache Cassandra), Cassandra Query Language (CQL), el SDK de AWS y la AWS Command Line Interface (AWS CLI) para activar las marcas de tiempo del lado del cliente. En esta sección se ofrecen ejemplos de cómo activar las marcas de tiempo del lado del cliente en tablas nuevas y existentes y cómo utilizar las marcas de tiempo del lado del cliente en las consultas. Para obtener más información sobre la API, consulte Referencia de la API de Amazon Keyspaces.

> **Important**
>
> Las marcas de tiempo del lado del cliente no se pueden desactivar. La activación de las marcas de tiempo del lado del cliente es un cambio que se realiza una única vez. Amazon Keyspaces no proporciona una opción de desactivación sin eliminar la tabla.

Temas

- Creación de una nueva tabla con marcas de tiempo del lado del cliente activadas (consola)

- Activación de las marcas de tiempo del lado del cliente en las tablas existentes (consola)
- Creación de una nueva tabla con las marcas de tiempo del lado del cliente activadas (CQL)
- Activación de las marcas de tiempo del lado del cliente para tablas existentes mediante ALTER TABLE (CQL)
- Creación de una nueva tabla con las marcas de tiempo del lado del cliente activadas (CLI)
- Activación de las marcas de tiempo del cliente en una tabla existente (CLI)
- Uso de marcas de tiempo del lado del cliente en instrucciones de lenguaje de manipulación de datos (DML)

Creación de una nueva tabla con marcas de tiempo del lado del cliente activadas (consola)

Siga estos pasos para crear una nueva tabla con las marcas de tiempo del lado del cliente activadas con la consola de Amazon Keyspaces.

Para crear una nueva tabla con marcas de tiempo del lado del cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tables (Tablas) y, a continuación, seleccione Create table (Crear tabla).
3. En la página Crear tabla, en la sección Detalles de la tabla, seleccione un espacio de claves y proporcione un nombre para la nueva tabla.
4. En la sección Esquema, cree el esquema para su tabla.
5. En la sección Configuración de la tabla, elija Personalizar configuración.
6. Continúe hasta Marcas de tiempo del lado del cliente.

   Elija Activar marcas de tiempo del lado del cliente para activar las marcas de tiempo del lado del cliente para la tabla.
7. Elija Crear tabla. Su tabla se crea con las marcas de tiempo del lado del cliente activadas.

Activación de las marcas de tiempo del lado del cliente en las tablas existentes (consola)

Siga estos pasos para activar las marcas de tiempo del lado del cliente en las tablas existentes mediante Amazon Keyspaces AWS Management Console.

Para activar las marcas de tiempo del lado del cliente para una tabla existente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. Elija la tabla que desee actualizar y elija la pestaña Configuración adicional.
3. En la pestaña Configuración adicional, vaya a Modificar marcas de tiempo del lado del cliente y seleccione Activar marcas de tiempo del lado del cliente.
4. Elija Guardar cambios para cambiar la configuración de la tabla.

Creación de una nueva tabla con las marcas de tiempo del lado del cliente activadas (CQL)

Para activar las marcas de tiempo del lado del cliente al crear una nueva tabla, puede utilizar la siguiente instrucción CQL.

```
CREATE TABLE my_table (
   userid uuid,
   time timeuuid,
   subject text,
   body text,
   user inet,
   PRIMARY KEY (userid, time)
) WITH CUSTOM_PROPERTIES = {'client_side_timestamps': {'status': 'enabled'}};
```

Para confirmar la configuración de las marcas de tiempo del lado del cliente para la nueva tabla, utilice la instrucción `SELECT` para revisar `custom_properties` como se muestra en el siguiente ejemplo.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

La salida de esta instrucción muestra el estado de las marcas de tiempo del lado del cliente.

```
'client_side_timestamps': {'status': 'enabled'}
```

Activación de las marcas de tiempo del lado del cliente para tablas existentes mediante **ALTER TABLE** (CQL)

Para activar las marcas de tiempo del lado del cliente de una tabla existente, puede utilizar la siguiente instrucción CQL.

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps': {'status':
 'enabled'}};;
```

Para confirmar la configuración de las marcas de tiempo del lado del cliente para la nueva tabla, utilice la instrucción `SELECT` para revisar `custom_properties` como se muestra en el siguiente ejemplo.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
 'my_keyspace' and table_name = 'my_table';
```

La salida de esta instrucción muestra el estado de las marcas de tiempo del lado del cliente.

```
'client_side_timestamps': {'status': 'enabled'}
```

Creación de una nueva tabla con las marcas de tiempo del lado del cliente activadas (CLI)

Para activar las marcas de tiempo del lado del cliente al crear una nueva tabla, puede utilizar la siguiente instrucción CLI.

```
./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}]'
```

Para confirmar que las marcas de tiempo del lado del cliente estén activadas para la nueva tabla, ejecute el siguiente código.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

La salida debe ser similar al siguiente ejemplo.

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    "resourceArn": "arn:aws:cassandra:us-east-2:555555555555:/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": ""
    }
}
```

Activación de las marcas de tiempo del cliente en una tabla existente (CLI)

Para activar las marcas de tiempo del lado del cliente en una tabla existente con la CLI, puede utilizar el siguiente código.

```
./aws keyspaces update-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED'
```

Para confirmar que las marcas de tiempo del lado del cliente estén activadas para la tabla, ejecute el siguiente código.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

La salida debe ser similar al siguiente ejemplo.

```
{
    "keyspaceName": "my_keyspace",
    "tableName": "my_table",
    "resourceArn": "arn:aws:cassandra:us-east-2:555555555555:/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": ""
    }
}
```

Uso de marcas de tiempo del lado del cliente en instrucciones de lenguaje de manipulación de datos (DML)

Una vez activadas las marcas de tiempo del lado del cliente, puede pasar la marca de tiempo en sus instrucciones `INSERT`, `UPDATE` y `DELETE` con la cláusula `USING TIMESTAMP`. El valor de la marca de tiempo es un `bigint` que representa un número de microsegundos desde la hora base estándar conocida como epoch: 1 de enero de 1970 a las 00:00:00 GMT. Una marca de tiempo proporcionada por el cliente tiene que caer en el rango de 2 días en el pasado a 5 minutos en el futuro desde la hora actual del reloj de pared. Amazon Keyspaces conserva los metadatos de marca de tiempo durante toda la vida de los datos. Puede utilizar la función `WRITETIME` para buscar marcas de tiempo que ocurrieron años atrás. Para obtener más información sobre la sintaxis CQL, consulte the section called “instrucciones DML”.

La siguiente instrucción CQL es un ejemplo de cómo utilizar una marca de tiempo como `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;
```

Si no especifica una marca de tiempo en su consulta CQL, Amazon Keyspaces utiliza la marca de tiempo transferida por su controlador de cliente. Si el controlador de cliente no proporciona ninguna marca de tiempo, Amazon Keyspaces asigna una marca de tiempo del lado del servidor para su operación de escritura.

Para ver el valor de marca de tiempo que se almacena para una columna específica, puede utilizar la función `WRITETIME` en una instrucción `SELECT` como se muestra en el siguiente ejemplo.

```
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;
```

Creación de recursos de Amazon Keyspaces con AWS CloudFormation

Amazon Keyspaces (para Apache Cassandra) está integrado con AWS CloudFormation, un servicio que le ayuda a modelar y configurar sus recursos de AWS para que pueda dedicar menos tiempo a la creación y administración de sus recursos e infraestructura. Usted crea una plantilla que describe todos los recursos de AWS que desea (como espacios de claves y tablas) y AWS CloudFormation se encarga de aprovisionar y configurar esos recursos por usted.

Al utilizar AWS CloudFormation, puede reutilizar su plantilla para configurar sus recursos de Amazon Keyspaces de forma coherente y repetida. Solo tiene que describir los recursos una vez y, luego, proporcionar los mismos recursos una y otra vez en varias cuentas y regiones de las Cuentas de AWS.

Espacios de claves y plantillas de AWS CloudFormation de Amazon Keyspaces

Para aprovisionar y configurar recursos para Amazon Keyspaces, debe comprender las plantillas de AWS CloudFormation. Las plantillas son archivos de texto con formato de tipo JSON o YAML. Estas plantillas describen los recursos que desea aprovisionar en sus pilas de AWS CloudFormation. Si no está familiarizado con JSON o YAML, puede utilizar Designer de AWS CloudFormation como ayuda para comenzar a utilizar las plantillas de AWS CloudFormation. Para obtener más información, consulte ¿Qué es el diseñador AWS CloudFormation? en la Guía del usuario de AWS CloudFormation.

Amazon Keyspaces permite crear espacios de claves y tablas en AWS CloudFormation. Para las tablas que cree con plantillas de AWS CloudFormation, puede especificar el esquema, el modo de lectura/escritura y la configuración de rendimiento aprovisionada. Para obtener más información, incluyendo ejemplos de plantillas JSON y YAML para espacios de claves y tablas, consulte Referencia de tipos de recursos de Cassandra en la Guía del usuario de AWS CloudFormation.

Obtener más información sobre AWS CloudFormation

Para obtener más información acerca de AWS CloudFormation, consulte los siguientes recursos:

- AWS CloudFormation

- Guía del usuario de AWS CloudFormation
- Guía del usuario de la interfaz de línea de comandos de AWS CloudFormation

Monitoreo de Amazon Keyspaces (para Apache Cassandra)

El monitoreo es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el rendimiento de Amazon Keyspaces y de sus otras soluciones de AWS. AWS ofrece las siguientes herramientas de monitoreo para controlar Amazon Keyspaces, informar cuando algo está mal y tomar medidas automáticas cuando sea necesario:

- Amazon Keyspaces ofrece un panel preconfigurado en la AWS Management Console que muestra la latencia y los errores agregados en todas las tablas de la cuenta.
- Amazon CloudWatch monitorea los recursos de AWS y las aplicaciones que ejecuta en AWS en tiempo real. Puede recopilar las métricas y hacer un seguimiento de ellas con paneles de control personalizados. Por ejemplo, puede crear una línea de base de rendimiento normal de Amazon Keyspaces en su entorno midiendo el rendimiento en varios momentos y en diferentes condiciones de carga. Al monitorear Amazon Keyspaces, puede almacenar los datos de monitoreo históricos con el fin de compararlos con los datos de rendimiento actuales, identificar patrones de rendimiento normales y anomalías en el rendimiento, e idear métodos para resolver los problemas. Para establecer una línea de base, debería, como mínimo, monitorear los errores del sistema. Para obtener más información, consulte la Guía del usuario de Amazon CloudWatch.
- Las alarmas de Amazon CloudWatch monitorean una única métrica a lo largo de un periodo de tiempo que usted especifique y realizan una o más acciones basadas en el valor de la métrica en relación con un umbral dado a lo largo de una serie de periodos de tiempo. Por ejemplo, si utiliza Amazon Keyspaces en modo aprovisionado con escalado automático de aplicaciones, la acción es una notificación enviada por Amazon Simple Notification Service (Amazon SNS) para evaluar una política de Application Auto Scaling.

  Las alarmas de CloudWatch no invocan acciones simplemente porque se encuentren en un estado determinado. El estado debe haber cambiado y debe mantenerse durante el número de periodos especificado. Para obtener más información, consulte Supervisión de Amazon Keyspaces con Amazon CloudWatch.
- Los Registros de Amazon CloudWatch le permiten monitorear, almacenar y acceder a sus archivos de registro desde las tablas de Amazon Keyspaces, CloudTrail y otros orígenes. CloudWatch Logs puede monitorear información en los registros y enviarle una notificación cuando se llega a determinados umbrales. También se pueden archivar los datos de los registros en un almacenamiento de larga duración. Para obtener más información, consulte la Guía del usuario de Amazon CloudWatch Logs.

- AWS CloudTrail captura las llamadas a la API y otros eventos relacionados que realiza la Cuenta de AWS o se realizan en nombre de esta. Además, entrega los archivos de registros a un bucket de Amazon S3 especificado. También pueden identificar qué usuarios y cuentas llamaron a AWS, la dirección IP de origen de las llamadas y el momento en que se hicieron. Para obtener más información, consulte la Guía del usuario de AWS CloudTrail.

Amazon EventBridge: es un bus de eventos sin servidor que facilita la conexión de sus aplicaciones con datos de varios orígenes. EventBridge proporciona un flujo de datos en tiempo real desde sus propias aplicaciones, aplicaciones de software como servicio (SaaS) y servicios de AWS y, luego, dirige dichos datos a destinos como Lambda. Esto le permite monitorear los eventos que ocurren en los servicios y crear arquitecturas basadas en eventos. Para obtener más información, consulte la Guía del usuario de Amazon EventBridge.

Temas

- Supervisión de Amazon Keyspaces con Amazon CloudWatch
- Registro de llamadas a la API de Amazon Keyspaces con AWS CloudTrail

Supervisión de Amazon Keyspaces con Amazon CloudWatch

Puedes monitorizar Amazon Keyspaces con Amazon CloudWatch, que recopila datos sin procesar y los procesa para convertirlos en métricas legibles prácticamente en tiempo real. Estas estadísticas se mantienen durante 15 meses, de forma que pueda obtener acceso a información histórica y disponer de una mejor perspectiva sobre el desempeño de su aplicación web o servicio.

También puede establecer alarmas que vigilen determinados umbrales y enviar notificaciones o realizar acciones cuando se cumplan dichos umbrales. Para obtener más información, consulta la Guía del CloudWatch usuario de Amazon.

> **Note**
>
> Para empezar rápidamente con un CloudWatch panel preconfigurado que muestre las métricas comunes de Amazon Keyspaces, puede utilizar AWS CloudFormation una plantilla disponible en. https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates

Temas

- ¿Cómo utilizo las métricas de Amazon Keyspaces?
- Métricas y dimensiones de Amazon Keyspaces
- Creación de CloudWatch alarmas para monitorear Amazon Keyspaces

¿Cómo utilizo las métricas de Amazon Keyspaces?

Las métricas reportadas por Amazon Keyspaces proporcionan información que puede analizar de diferentes maneras. En la siguiente lista se indican algunos usos frecuentes de las métricas. Se trata de sugerencias que puede usar como punto de partida y no de una lista completa. Para obtener más información sobre métricas y retención, consulte Métricas.

¿Cómo...?	Métricas relevantes
¿Cómo puedo determinar si se ha producido algún error en el sistema?	Puede monitorear `SystemErrors` para determinar si alguna solicitud resultó en un código de error del servidor. Normalmente, esta métrica debe ser igual a cero. De no ser así, es posible que desee investigarlo.
¿Cómo puedo comparar la capacidad de lectura aprovisionada promedio frente a la capacidad de lectura consumida?	Para monitorear la capacidad de lectura aprovisionada promedio y la capacidad de lectura consumida 1. Establezca el Periodo para `ConsumedReadCapacityUnits` y `ProvisionedReadCapacityUnits` en el intervalo que desee monitorear. 2. Cambie la Estadística para `ConsumedReadCapacityUnits` de `Average` a `Sum`. 3. Cree una nueva Expresión matemática vacía. 4. En la sección Detalles de la nueva expresión matemática, introduzca el identificador de **`ConsumedReadCapacityUnits`** y divida la métrica por la función CloudWatch PERIOD de la métrica (metric_id/ (PERIOD (metric_id)). 5. Deseleccione `ConsumedReadCapacityUnits` . Ahora puede comparar su capacidad de lectura consumida promedio frente a su capacidad aprovisionada. Para obtener

¿Cómo...?	Métricas relevantes
	más información sobre las funciones aritméticas básicas y cómo crear una serie temporal, consulte Uso de matemáticas de métricas.
¿Cómo puedo comparar la capacidad de escritura aprovisionada promedio frente a la capacidad de escritura consumida?	Para monitorear la capacidad de escritura aprovisionada promedio frente a la capacidad de escritura consumida 1. Establezca el Periodo para `ConsumedWriteCapacityUnits` y `ProvisionedWriteCapacityUnits` en el intervalo que desee monitorear. 2. Cambie la Estadística para `ConsumedWriteCapacityUnits` de `Average` a `Sum`. 3. Cree una nueva Expresión matemática vacía. 4. En la sección Detalles de la nueva expresión matemática, introduzca el identificador **`ConsumedWriteCapacityUnits`** y divida la métrica entre la función CloudWatch PERÍODO de la métrica (metric_id/ (PERIOD (metric_id)). 5. Deseleccione `ConsumedWriteCapacityUnits` . Ahora puede comparar su capacidad de escritura consumida promedio frente a su capacidad aprovisionada. Para obtener más información sobre las funciones aritméticas básicas y cómo crear una serie temporal, consulte Uso de matemáticas de métricas.

Métricas y dimensiones de Amazon Keyspaces

Cuando interactúas con Amazon Keyspaces, envía las siguientes métricas y dimensiones a Amazon. CloudWatch Todas las métricas se agregan y notifican cada minuto. Puede utilizar los siguientes procedimientos para ver las métricas de Amazon Keyspaces.

Para ver las métricas mediante la consola CloudWatch

Las métricas se agrupan en primer lugar por el espacio de nombres de servicio y, a continuación, por las diversas combinaciones de dimensiones dentro de cada espacio de nombres.

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. Si es necesario, cambie la región. En la barra de navegación, seleccione la región donde residen sus recursos de AWS . Para obtener más información, consulte puntos de conexión de servicio de AWS.
3. En el panel de navegación, seleccione Métricas.
4. En la pestaña Todas las métricas, elija `AWS/Cassandra`.

Para ver las métricas mediante la AWS CLI

- En el símbolo del sistema, ejecute el siguiente comando.

```
aws cloudwatch list-metrics --namespace "AWS/Cassandra"
```

Métricas y dimensiones de Amazon Keyspaces

Aquí se muestran las métricas y dimensiones que Amazon Keyspaces envía a Amazon CloudWatch .

Métricas de Amazon Keyspaces

Amazon CloudWatch agrega las métricas de Amazon Keyspaces en intervalos de un minuto.

No todas las estadísticas, tales como `Average` o `Sum`, son aplicables a todas las métricas. Sin embargo, todos estos valores están disponibles a través de la consola de Amazon Keyspaces o mediante la CloudWatch consola o los AWS SDK para todas las métricas. AWS CLI En la siguiente tabla, cada métrica tiene una lista de estadísticas válidas y aplicables.

Métrica	Descripción
`AccountMaxTableLevelReads`	El número máximo de unidades de capacidad de lectura que puede utilizar una tabla de la cuenta. Para las tablas bajo demanda, este límite establece el máximo de unidades de solicitud de lectura que puede utilizar una tabla.

Métrica	Descripción
	Unidades: `Count` Estadísticas válidas: • `Maximum`: el número máximo de unidades de capacidad de lectura que puede utilizar una tabla de la cuenta.
`AccountMaxTableLevelWrites`	El número máximo de unidades de capacidad de escritura que puede utilizar una tabla de la cuenta. Para las tablas bajo demanda, este límite establece el máximo de unidades de solicitud de escritura que puede utilizar una tabla. Unidades: `Count` Estadísticas válidas: • `Maximum`: el número máximo de unidades de capacidad de escritura que puede utilizar una tabla de la cuenta.

Métrica	Descripción
`AccountProvisionedReadCapacityUtilization`	El porcentaje de unidades de capacidad de lectura aprovisionadas utilizadas por su cuenta. Unidades: `Percent` Estadísticas válidas: • `Maximum`: el porcentaje máximo de unidades de capacidad de lectura aprovisionadas utilizadas por la cuenta. • `Minimum`: el porcentaje mínimo de unidades de capacidad de lectura aprovisionadas utilizadas por la cuenta. • `Average`: el porcentaje promedio de unidades de capacidad de lectura aprovisionadas utilizadas por la cuenta. La métrica se publica a intervalos de cinco minutos. Por lo tanto, si ajusta rápidamente las unidades de capacidad de lectura aprovisionadas, es posible que esta estadística no refleje el promedio real.

Métrica	Descripción
`AccountProvisionedWriteCapacityUtilization`	El porcentaje de unidades de capacidad de escritura aprovisionadas utilizadas por su cuenta. Unidades: `Percent` Estadísticas válidas: • `Maximum`: el porcentaje máximo de unidades de capacidad de escritura aprovisionadas utilizadas por la cuenta. • `Minimum`: el porcentaje mínimo de unidades de capacidad de escritura aprovisionadas utilizadas por la cuenta. • `Average`: el porcentaje promedio de unidades de capacidad de escritura aprovisionadas utilizadas por la cuenta. La métrica se publica a intervalos de cinco minutos. Por lo tanto, si ajusta rápidamente las unidades de capacidad de escritura aprovisionadas, es posible que esta estadística no refleje el promedio real.

Métrica	Descripción
`BillableTableSizeInBytes`	El tamaño facturable de la tabla en bytes. Es la suma del tamaño codificado de todas las filas de la tabla. Esta métrica le ayuda a hacer un seguimiento de los costos de almacenamiento de su tabla en el tiempo. Unidades: `Bytes` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Maximum`: el tamaño máximo de almacenamiento de la tabla. • `Minimum`: el tamaño mínimo de almacenamiento de la tabla. • `Average`: el tamaño promedio de almacenamiento de la tabla. Esta métrica se calcula en intervalos de 4 a 6 horas.

Métrica	Descripción
`ConditionalCheckFailedRequests`	El número de solicitudes de escritura de transacciones ligeras (LWT) fallidas. La operaciones `INSERT`, `UPDATE`, y `DELETE` le permiten proporcionar una condición lógica que debe evaluarse como true (verdadera) antes de que la operación pueda continuar. Si esta condición se evalúa como false, `ConditionalCheckFailedRequests` se incrementa en uno. Las comprobaciones de estado que dan como resultado falsas consumen unidades de capacidad de escritura en función del tamaño de la fila. Para obtener más información, consulte the section called “Transacciones ligeras”. Unidades: `Count` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `Sum`

Métrica	Descripción
`ConsumedReadCapacityUnits`	El número de unidades de capacidad de lectura consumidas durante el periodo especificado. Para obtener más información, consulte Modo de capacidad de lectura/escritura. **Note** Para conocer el promedio de utilización de rendimiento por segundo, utilice la estadística `Sum` para calcular el rendimiento consumido durante el periodo de un minuto. A continuación, divida la suma por el número de segundos de un minuto (60) para calcular el promedio de `ConsumedReadCapacityUnits` por segundo (reconociendo que este promedio no resalta los picos grandes pero breves de actividad de lectura que se hayan producido durante ese minuto). Para obtener más información sobre comparación de la capacidad de lectura promedio consumida con la capacidad de lectura aprovisionada, consulte the section called “Uso de las métricas”. Unidades: `Count` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Minimum`: el número mínimo de unidades de capacidad de lectura consumidas por cualquier solicitud individual a la tabla. • `Maximum`: el número máximo de unidades de capacidad de lectura consumidas por cualquier solicitud individual a la tabla.

Métrica	Descripción
	• `Average`: la capacidad de lectura promedio por solicitud consumida. **Note** El valor `Average` está influenciado por periodos de inactividad donde el valor de la muestra será cero. • `Sum`: el total de unidades de capacidad de lectura consumidas. Esta es la estadística más útil para la métrica `ConsumedReadCapacityUnits` . • `SampleCount` : el número de solicitudes a Amazon Keyspaces, aunque no se haya consumido capacidad de lectura. **Note** El valor `SampleCount` está influenciado por periodos de inactividad donde el valor de la muestra será cero.

Métrica	Descripción
`ConsumedWriteCapacityUnits`	El número de unidades de capacidad de escritura consumidas durante el periodo especificado. Puede recuperar la capacidad de escritura total consumida para una tabla. Para obtener más información, consulte Modo de capacidad de lectura/escritura. **Note** Para conocer el promedio de utilización de rendimiento por segundo, utilice la estadística `Sum` para calcular el rendimiento consumido durante el periodo de un minuto. A continuación, divida la suma por el número de segundos de un minuto (60) para calcular el promedio de `ConsumedWriteCapacityUnits` por segundo (reconociendo que este promedio no resalta los picos grandes pero breves de actividad de escritura que se hayan producido durante ese minuto). Para obtener más información sobre comparación de la capacidad de escritura promedio consumida con la capacidad de escritura aprovisionada, consulte the section called “Uso de las métricas”. Unidades: `Count` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Minimum`: el número mínimo de unidades de capacidad de escritura consumidas por cualquier solicitud individual a la tabla.

Métrica	Descripción
	• `Maximum`: el número máximo de unidades de capacidad de escritura consumidas por cualquier solicitud individual a la tabla. • `Average`: la capacidad de escritura promedio por solicitud consumida. **Note** El valor `Average` está influenciado por periodos de inactividad donde el valor de la muestra será cero. • `Sum`: el total de unidades de capacidad de escritura consumidas. Esta es la estadística más útil para la métrica `ConsumedWriteCapacityUnits` . • `SampleCount` : el número de solicitudes a Amazon Keyspaces, aunque no se haya consumido capacidad de escritura. **Note** El valor `SampleCount` está influenciado por periodos de inactividad donde el valor de la muestra será cero.

Métrica	Descripción
`MaxProvisionedTableReadCapacityUtilization`	El porcentaje de unidades de capacidad de lectura aprovisionadas utilizadas por la tabla de lectura aprovisionada más alta de una cuenta. Unidades: `Percent` Estadísticas válidas: • `Maximum`: el porcentaje máximo de unidades de capacidad de lectura aprovisionadas utilizadas por la tabla de lectura aprovisionada más alta de la cuenta. • `Minimum`: el porcentaje mínimo de unidades de capacidad de lectura aprovisionadas utilizadas por la tabla de lectura aprovisionada más alta de la cuenta. • `Average`: el porcentaje promedio de unidades de capacidad de lectura aprovisionadas utilizadas por la tabla de lectura aprovisionada más alta de la cuenta. La métrica se publica a intervalos de cinco minutos. Por lo tanto, si ajusta rápidamente las unidades de capacidad de lectura aprovisionadas, es posible que esta estadística no refleje el promedio real.

Métrica	Descripción
`MaxProvisionedTableWriteCapacityUtilization`	El porcentaje de capacidad de escritura aprovisionada utilizada por la tabla de escritura aprovisionada más alta de una cuenta. Unidades: `Percent` Estadísticas válidas: • `Maximum`: el porcentaje máximo de unidades de capacidad de escritura aprovisionadas utilizadas por la tabla de escritura aprovisionada más alta de la cuenta. • `Minimum`: el porcentaje mínimo de unidades de capacidad de escritura aprovisionadas utilizadas por la tabla de escritura aprovisionada más alta de la cuenta. • `Average`: el porcentaje promedio de unidades de capacidad de escritura aprovisionadas utilizadas por la tabla de escritura aprovisionada más alta de la cuenta. La métrica se publica a intervalos de cinco minutos. Por lo tanto, si ajusta rápidamente las unidades de capacidad de escritura aprovisionadas, es posible que esta estadística no refleje el promedio real.

Métrica	Descripción
`PerConnectionRequestRateExceeded`	Solicitudes a Amazon Keyspaces que superan la cuota de tasa de solicitudes por conexión. Cada conexión de cliente a Amazon Keyspaces puede admitir hasta 3000 solicitudes CQL por segundo. Los clientes pueden crear varias conexiones para aumentar el rendimiento. Si utiliza la réplica multirregión, cada escritura replicada también contribuye a esta cuota. Como práctica recomendada, le sugerimos que aumente el número de conexiones a sus tablas para evitar errores `PerConnectionRequestRateExceeded`. No existe límite para el número de conexiones que pueda tener en Amazon Keyspaces. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `SampleCount` • `Sum`

Métrica	Descripción
`ProvisionedReadCapacityUnits`	El número de unidades de capacidad de lectura aprovisionadas para una tabla. La dimensión `TableName` devuelve las `ProvisionedReadCapacityUnits` de la tabla. Unidades: `Count` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Minimum`: la configuración más baja para la capacidad de lectura aprovisionada. Si utiliza `ALTER TABLE` para aumentar la capacidad de lectura, esta métrica muestra el valor más bajo de `ReadCapacityUnits` aprovisionado durante este periodo de tiempo. • `Maximum`: la configuración más alta para la capacidad de lectura aprovisionada. Si utiliza `ALTER TABLE` para reducir la capacidad de lectura, esta métrica muestra el valor más alto de `ReadCapacityUnits` aprovisionado durante este período de tiempo. • `Average`: la capacidad de lectura aprovisionada promedio. La métrica `ProvisionedReadCapacityUnits` se publica a intervalos de cinco minutos. Por lo tanto, si ajusta rápidamente las unidades de capacidad de lectura aprovisionadas, es posible que esta estadística no refleje el promedio real.

Métrica	Descripción
`ProvisionedWriteCapacityUnits`	El número de unidades de capacidad de escritura aprovisionadas para una tabla. La dimensión `TableName` devuelve las `ProvisionedWriteCapacityUnits` de la tabla. Unidades: `Count` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Minimum`: la configuración más baja para la capacidad de escritura aprovisionada. Si utiliza `ALTER TABLE` para aumentar la capacidad de escritura, esta métrica muestra el valor más bajo de `WriteCapacityUnits` aprovisionado durante este período de tiempo. • `Maximum`: la configuración más alta para la capacidad de escritura aprovisionada. Si utiliza `ALTER TABLE` para reducir la capacidad de escritura, esta métrica muestra el valor más alto de `WriteCapacityUnits` aprovisionado durante este período de tiempo. • `Average`: la capacidad de escritura aprovisionada promedio. La métrica `ProvisionedWriteCapacityUnits` se publica a intervalos de cinco minutos. Por lo tanto, si ajusta rápidamente las unidades de capacidad de escritura aprovisionadas, es posible que esta estadística no refleje el promedio real.

Métrica	Descripción
`ReadThrottleEvents`	Las solicitudes a Amazon Keyspaces que superan la capacidad de lectura aprovisionada de una tabla, o las cuotas a nivel de cuenta, las cuotas de solicitudes por conexión o las cuotas a nivel de partición. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `SampleCount` • `Sum`
`ReplicationLatency`	Esta métrica se aplica solo a los espacios de claves multirregión y mide el tiempo que se tardó en replicar `updates`, `inserts` o `deletes` de una tabla réplica a otra tabla réplica en un espacio de claves multirregión. Unidades: `Millisecond` Dimensiones: `TableName, ReceivingRegion` Estadísticas válidas: • `Average` • `Maximum` • `Minimum`

Métrica	Descripción
`ReturnedItemCountBySelect`	Número de filas devueltas por consultas `SELECT` multifila durante el periodo especificado. Las consultas `SELECT` multifila son consultas que no contienen la clave principal totalmente cualificada, como los escaneos de tabla completa y las consultas de rango. El número de filas devueltas no es necesariamente el mismo que el número de filas que se evaluaron. Por ejemplo, suponga que solicitó una `SELECT * con ALLOW FILTERING` en una tabla que tiene 100 filas, pero especificó una cláusula `WHERE` que redujo los resultados de forma que solo se devuelven 15 filas. En este caso, la respuesta de `SELECT` contendría un `ScanCount` de 100 y un `Count` de 15 filas devueltas. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `Sum`

Métrica	Descripción
`StoragePartitionThroughputCapacityExceeded`	Solicitudes a una partición de almacenamiento de Amazon Keyspaces que superan la capacidad de rendimiento de la partición. Las particiones de almacenamiento de Amazon Keyspaces pueden admitir hasta 1000 WCU/WRU por segundo y 3000 RCU/RRU por segundo. Le recomendamos que revise su modelo de datos para distribuir el tráfico de lectura/escritura entre más particiones a fin de mitigar estas excepciones. **Note** Las particiones lógicas de Amazon Keyspaces pueden abarcar múltiples particiones de almacenamiento y su tamaño es prácticamente ilimitado. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `SampleCount` • `Sum`
`SuccessfulRequestCount`	El número de solicitudes procesadas con éxito durante el periodo de tiempo especificado. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `SampleCount`

Métrica	Descripción
SuccessfulRequestLatency	Las solicitudes exitosas a Amazon Keyspaces durante el periodo de tiempo especificado. SuccessfulRequestLatency puede proporcionar dos tipos diferentes de información: • El tiempo transcurrido para las solicitudes correctas (Minimum, Maximum, Sum o Average). • El número de solicitudes realizadas correctamente (SampleCount). SuccessfulRequestLatency refleja la actividad solo dentro de Amazon Keyspaces y no tiene en cuenta la latencia en red o la actividad del lado del cliente. Unidades: Milliseconds Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: • Minimum • Maximum • Average • SampleCount

Métrica	Descripción
SystemErrors	Las solicitudes a Amazon Keyspaces que generan un ServerError durante el periodo especificado. Un ServerError suele indicar un error interno del servicio. Unidades: Count Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: • Sum • SampleCount
SystemReconciliationDeletes	Las unidades consumidas para eliminar datos lapidados al estar habilitadas las marcas de tiempo del lado del cliente. Cada SystemReconciliationDelete proporciona capacidad suficiente para eliminar o actualizar hasta 1 KB de datos por fila. Por ejemplo, para actualizar una fila que almacena 2,5 KB de datos y eliminar al mismo tiempo una o más columnas dentro de la fila se requieren 3 SystemReconciliationDeletes . O bien, para eliminar una fila completa que contenga 3,5 KB de datos se requieren 4 SystemReconciliationDeletes . Unidades: Count Dimensiones: Keyspace, TableName Estadísticas válidas: • Sum: el número total de SystemReconciliationDeletes consumidos en un periodo de tiempo.

Métrica	Descripción
`TTLDeletes`	Las unidades consumidas para eliminar o actualizar datos en una fila utilizando Tiempo de vida (TTL). Cada `TTLDelete` proporciona capacidad suficiente para eliminar o actualizar hasta 1 KB de datos por fila. Por ejemplo, para actualizar una fila que almacena 2,5 KB de datos y eliminar una o más columnas dentro de la fila al mismo tiempo se requieren 3 eliminaciones TTL. O bien, para eliminar una fila entera que contiene 3,5 KB de datos se requieren 4 eliminaciones TTL. Unidades: `Count` Dimensiones: `Keyspace, TableName` Estadísticas válidas: • `Sum`: el número total de `TTLDeletes` consumidos en un periodo de tiempo.

Métrica	Descripción
`UserErrors`	Las solicitudes a Amazon Keyspaces que generan un error `InvalidRequest` durante el periodo especific ado. Una `InvalidRequest` suele indicar un error del lado del cliente, como una combinación no válida de parámetros, un intento de actualizar una tabla inexistente o una firma de solicitud incorrecta. `UserErrors` representa la suma de las solicitudes no válidas de la actual Región de AWS y la actual Cuenta de AWS. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `Sum` • `SampleCount`
`WriteThrottleEvents`	Las solicitudes a Amazon Keyspaces que superan la capacidad de escritura aprovisionada de una tabla, o las cuotas a nivel de cuenta, las cuotas de solicitudes por conexión o las cuotas a nivel de partición. Unidades: `Count` Dimensiones: `Keyspace, TableName, Operation` Estadísticas válidas: • `SampleCount` • `Sum`

Dimensiones de las métricas de Amazon Keyspaces

Las métricas de Amazon Keyspaces se identifican por los valores de la cuenta, el nombre de la tabla o la operación. Puede utilizar la CloudWatch consola para recuperar los datos de Amazon Keyspaces en cualquiera de las dimensiones de la siguiente tabla.

Dimensión	Descripción
`Keyspace`	Esta dimensión limita los datos a un espacio de claves específico. Este valor puede ser cualquier espacio de claves en la región y la Cuenta de AWS actuales.
`Operation`	Esta dimensión limita los datos a una de las operaciones CQL de Amazon Keyspaces, como las operaciones `INSERT` o `SELECT`.
`TableName`	Esta dimensión limita los datos a una tabla específica. Este valor puede ser cualquier nombre de tabla en la región y la Cuenta de AWS actuales. Si el nombre de la tabla no es único dentro de la cuenta, también debe especificar el `Keyspace`.

Creación de CloudWatch alarmas para monitorear Amazon Keyspaces

Puede crear una CloudWatch alarma de Amazon para Amazon Keyspaces que envíe un mensaje del Amazon Simple Notification Service (Amazon SNS) cuando la alarma cambie de estado. Una alarma vigila una métrica determinada durante el periodo especificado. Realiza una o varias acciones según el valor de la métrica con respecto a un umbral dado durante varios períodos de tiempo. La acción es una notificación enviada a un tema de Amazon SNS o a una política de Application Auto Scaling.

Cuando utiliza Amazon Keyspaces en modo aprovisionado con Application Auto Scaling, el servicio crea dos pares de CloudWatch alarmas en su nombre. Cada par representa sus límites superior e inferior para las configuraciones de rendimiento aprovisionadas y consumidas. Estas CloudWatch alarmas se activan cuando la utilización real de la tabla se desvía de la utilización objetivo durante un período prolongado. Para obtener más información sobre CloudWatch las alarmas creadas por Application Auto Scaling, consultethe section called “Funcionamiento del escalado automático de Amazon Keyspaces”.

Las alarmas invocan acciones únicamente en caso de cambios de estado sostenidos. CloudWatch las alarmas no invocan acciones simplemente porque se encuentran en un estado determinado. El estado debe haber cambiado y debe mantenerse durante el número de periodos especificado.

Para obtener más información sobre la creación de CloudWatch alarmas, consulta Uso de CloudWatch alarmas de Amazon en la Guía del CloudWatch usuario de Amazon.

Registro de llamadas a la API de Amazon Keyspaces con AWS CloudTrail

Amazon Keyspaces se integra con AWS CloudTrail, un servicio que proporciona un registro de las acciones realizadas por un usuario, rol o servicio de AWS en Amazon Keyspaces. CloudTrail captura las llamadas a la API del lenguaje de definición de datos (DDL) y las llamadas a la API del lenguaje de manipulación de datos para Amazon Keyspaces como eventos. Las llamadas que se capturan incluyen llamadas desde la consola de Amazon Keyspaces y llamadas mediante programación a las operaciones de la API de Amazon Keyspaces.

Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail a un bucket de Amazon Simple Storage Service (Amazon S3), incluyendo los eventos de Amazon Keyspaces.

Si no configura un registro de seguimiento, puede ver los eventos admitidos más recientes en la consola de CloudTrail en el Historial de eventos. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon Keyspaces, la dirección IP desde la que se realizó la solicitud, quién realizó la solicitud, cuándo se realizó y detalles adicionales.

Para obtener más información acerca de CloudTrail, consulte la AWS CloudTrailGuía del usuario de .

Temas

- Configuración de las entradas del archivo de registro de Amazon Keyspaces en CloudTrail
- Información sobre el lenguaje de definición de datos (DDL) de Amazon Keyspaces en CloudTrail
- Información sobre el lenguaje de manipulación de datos (DDL) de Amazon Keyspaces en CloudTrail
- Comprensión de las entradas del archivo de registro de Amazon Keyspaces

Configuración de las entradas del archivo de registro de Amazon Keyspaces en CloudTrail

Cada acción de la API de Amazon Keyspaces registrada en CloudTrail incluye parámetros de solicitud que se expresan en lenguaje de consulta CQL. Para obtener más información, consulte Referencia del lenguaje CQL.

Puede ver, buscar y descargar eventos recientes en su Cuenta de AWS. Para obtener más información, consulte Ver eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de los eventos en su Cuenta de AWS, incluyendo los eventos de Amazon Keyspaces, cree un registro de seguimiento. Un registro de seguimiento permite a CloudTrail enviar archivos de registro a un bucket de Amazon S3. De manera predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de AWS. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail.

Para obtener más información, consulte los siguientes temas en la Guía del usuario deAWS CloudTrail:

- Introducción a la creación de registros de seguimiento
- Servicios e integraciones compatibles con CloudTrail
- Configuración de notificaciones de Amazon SNS para CloudTrail
- Recepción de archivos de registro de CloudTrail de varias regiones
- Recepción de archivos de registro de CloudTrail de varias cuentas

Cada entrada de registro o evento contiene información acerca de quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario AWS Identity and Access Management (IAM) o credenciales de usuario raíz.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro servicio de AWS.

Para más información, consulte el elemento userIdentity de CloudTrail.

Información sobre el lenguaje de definición de datos (DDL) de Amazon Keyspaces en CloudTrail

CloudTrail se habilita en su Cuenta de AWS cuando se crea la cuenta. Cuando se produce una actividad DDL en Amazon Keyspaces, dicha actividad se registra automáticamente como un evento de CloudTrail junto con otros eventos de servicios de AWS en el Historial de eventos. En la tabla siguiente se muestran las instrucciones DDL que se registran para Amazon Keyspaces.

`eventName` de CloudTrail	Instrucción	Acción de CQL	Acción de SDK de AWS
CreateKeyspace	DDL	`CREATE KEYSPACE`	`CreateKeyspace`
DropKeyspace	DDL	`DROP KEYSPACE`	`DeleteKeyspace`
CreateTable	DDL	`CREATE TABLE`	`CreateTable`
DropTable	DDL	`DROP TABLE`	`DeleteTable`
AlterTable	DDL	`ALTER TABLE`	`UpdateTable` , `TagResource` , `UntagResource`

Información sobre el lenguaje de manipulación de datos (DDL) de Amazon Keyspaces en CloudTrail

Para habilitar el registro de las instrucciones DML de Amazon Keyspaces con CloudTrail, debe habilitar primero el registro de la actividad de la API del plano de datos en CloudTrail. Puede empezar a registrar los eventos DML de Amazon Keyspaces en rutas nuevas o existentes si elige registrar la actividad para el tipo de evento de datos de la tabla Cassandra mediante la consola de CloudTrail o configurando el valor `resources.type` para que `AWS::Cassandra::Table` utilice la CLI de AWS o las operaciones de la API de CloudTrail. Para obtener más información, consulte Registro de eventos de datos.

En la siguiente tabla, se muestran los eventos de datos que registra CloudTrail para `Cassandra table`.

eventName de CloudTrail	Instrucción	Acción de CQL	Acción de SDK de AWS
Select	DML	`SELECT`	`GetKeyspace` , `GetTable`, `ListKeyspaces` , `ListTables` `ListTagsForResource`
Inserte	DML	`INSERT`	no hay acciones del AWS SDK disponibles
Actualizar	DML	`UPDATE`	no hay acciones del AWS SDK disponibles
Eliminación	DML	`DELETE`	no hay acciones del AWS SDK disponibles

Comprensión de las entradas del archivo de registro de Amazon Keyspaces

Los archivos de registro de CloudTrail pueden contener una o varias entradas de registro. Un evento representa una solicitud específica realizada desde un código fuente y contiene información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. Los archivos de registro de CloudTrail no rastrean el orden en la pila de las llamadas públicas a la API, por lo que estas no aparecen en ningún orden específico.

En el siguiente ejemplo se muestra una entrada de registro de CloudTrail que ilustra las acciones `CreateKeyspace`, `DropKeyspace`, `CreateTable` y `DropTable`:

```
{
  "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"
    }
  ]
}
```

En el siguiente archivo de registro se muestra un ejemplo de una instrucción SELECT.

```
{
    "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"
    }
}
```

En el siguiente archivo de registro se muestra un ejemplo de una instrucción `INSERT`.

```
{
    "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"
    }
}
```

En el siguiente archivo de registro se muestra un ejemplo de una instrucción UPDATE.

```
{
    "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"
        }
}
```

En el siguiente archivo de registro se muestra un ejemplo de una instrucción DELETE.

```
{
    "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"
```

```
    }
}
```

Seguridad en Amazon Keyspaces (para Apache Cassandra)

La seguridad en la nube de AWS es la mayor prioridad. Como cliente de AWS, se beneficia de una arquitectura de red y un centro de datos que se han diseñado para satisfacer los requisitos de seguridad de las organizaciones más exigentes.

La seguridad es una responsabilidad compartida entre AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta servicios de AWS en Nube de AWS. AWS también le proporciona servicios que puede utilizar de forma segura. Auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad en el marco de los programas de conformidad de AWS. Para obtener información sobre los programas de conformidad que se aplican a Amazon Keyspaces, consulte Servicios en el ámbito del programa de conformidad de AWS.
- Seguridad en la nube: su responsabilidad viene determinada por el servicio de AWS que utilice. Usted también es responsable de otros factores incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos aplicables.

Esta documentación le ayudará a comprender cómo aplicar el modelo de responsabilidad compartida al utilizar Amazon Keyspaces. En los siguientes temas le mostramos cómo configurar Amazon Keyspaces para cumplir sus objetivos de seguridad y conformidad. También obtendrá información sobre cómo utilizar otros servicios de AWS que pueden ayudarle a monitorear y asegurar sus recursos de Amazon Keyspaces.

Temas

- Protección de datos en Amazon Keyspaces
- AWS Identity and Access Management para Amazon Keyspaces
- Validación de conformidad para Amazon Keyspaces (para Apache Cassandra)
- Resiliencia y recuperación de desastres en Amazon Keyspaces
- Seguridad de la infraestructura en Amazon Keyspaces
- Configuración y análisis de vulnerabilidades para Amazon Keyspaces
- Prácticas recomendadas de seguridad para Amazon Keyspaces

Protección de datos en Amazon Keyspaces

El modelo de responsabilidad compartida de AWS se aplica a la protección de datos en Amazon Keyspaces (para Apache Cassandra). Como se describe en este modelo, AWS es responsable de proteger la infraestructura global que ejecuta la totalidad de Nube de AWS. Usted es responsable de mantener el control sobre el contenido alojado en esta infraestructura. Usted también es responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte las Preguntas frecuentes sobre la privacidad de datos. Para obtener información sobre la protección de datos en Europa, consulte la publicación de blog Modelo de responsabilidad compartida y GDPR de AWS en el Blog de seguridad de AWS.

Con fines de protección de datos, recomendamos proteger las credenciales de la cuenta de Cuenta de AWS y configurar cuentas de usuario individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se conceden a cada usuario los permisos necesarios para cumplir con sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utilice autenticación multifactor (MFA) en cada cuenta.
- Utilice SSL/TLS para comunicarse con los recursos de AWS. reSe recomienda el uso de TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad del usuario con AWS CloudTrail.
- Utilice las soluciones de cifrado de AWS, junto con todos los controles de seguridad predeterminados dentro de los Servicios de AWS.
- Utilice servicios de seguridad gestionados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita módulos criptográficos validados FIPS 140-2 al acceder a AWS a través de una interfaz de la línea de comandos o una API, utilice un punto de conexión de FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-2.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto es igualmente válido al trabajar con Amazon Keyspaces u otros Servicios de AWS con la consola, la API, la AWS CLI, o los SDK de AWS. Cualquier dato que introduzca en etiquetas o campos de formato libre utilizados para nombres se pueden emplear

para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Temas

- Cifrado en reposo en Amazon Keyspaces
- Cifrado en tránsito en Amazon Keyspaces
- Privacidad del tráfico entre redes en Amazon Keyspaces

Cifrado en reposo en Amazon Keyspaces

El cifrado en reposo de Amazon Keyspaces (para Apache Cassandra) proporciona mayor seguridad al cifrar todos sus datos en reposo mediante claves de cifrado almacenadas en AWS Key Management Service (AWS KMS). Esta funcionalidad ayuda a reducir la carga y la complejidad operativas que conlleva la protección de información confidencial. Con el cifrado en reposo, puede crear aplicaciones sensibles a la seguridad que cumplan estrictos requisitos normativos y de conformidad para la protección de datos.

El cifrado en reposo de Amazon Keyspaces cifra sus datos utilizando el estándar de cifrado avanzado de 256 bits (AES-256). Esto ayuda a proteger los datos del acceso no autorizado al almacenamiento subyacente.

Amazon Keyspaces cifra y descifra los datos de la tabla de forma transparente. Amazon Keyspaces utiliza el cifrado de sobre y una jerarquía de claves para proteger las claves de cifrado de datos. Se integra con AWS KMS para almacenar y administrar la clave de cifrado raíz. Para obtener más información sobre jerarquía de claves de cifrado, consulte the section called “Cómo funciona”. Para obtener más información sobre conceptos de AWS KMS como el cifrado de sobre, consulte Conceptos del servicio de administración de AWS KMS en la Guía para desarrolladores de AWS Key Management Service.

Al crear una nueva tabla, puede elegir una de las siguientes claves AWS KMS (claves KMS):

- Clave propiedad de AWS: este es el tipo de cifrado predeterminado. La clave es propiedad de Amazon Keyspaces (sin cargo adicional).
- Clave administrada por el cliente): esta clave se almacena en la cuenta y usted la crea, posee y administra. Usted controla plenamente la clave administrada por el cliente (se aplican cargos de AWS KMS).

Puede alternar entre una Clave propiedad de AWS y una clave administrada por el cliente en cualquier momento. Puede especificar una clave administrada por el cliente al crear una nueva tabla o cambiar la clave KMS de una tabla existente ya sea con la consola o mediante programación utilizando instrucciones CQL. Para saber cómo hacerlo, consulte Cifrado en reposo: Cómo utilizar claves administradas por el cliente para cifrar tablas en Amazon Keyspacess..

El cifrado en reposo mediante la opción predeterminada de Claves propiedad de AWS se ofrece sin cargo adicional. Sin embargo, se aplican cargos de AWS KMS para las claves administradas por el cliente. Para obtener más información acerca de los precios, consulte Precios de AWS KMS.

El cifrado en reposo de Amazon Keyspaces está disponible en todas las Regiones de AWS, incluyendo las regiones China (Pekín) de AWS y China (Ningxia) de AWS. Para obtener más información, consulte Cifrado en reposo: Cómo funciona en Amazon Keyspaces.

Temas

- Cifrado en reposo: Cómo funciona en Amazon Keyspaces
- Cifrado en reposo: Cómo utilizar claves administradas por el cliente para cifrar tablas en Amazon Keyspacess.

Cifrado en reposo: Cómo funciona en Amazon Keyspaces

El cifrado en reposo de Amazon Keyspaces (para Apache Cassandra) cifra sus datos utilizando el estándar de cifrado avanzado de 256 bits (AES-256). Esto ayuda a proteger los datos del acceso no autorizado al almacenamiento subyacente. Todos los datos de clientes en las tablas de Amazon Keyspaces se cifran en reposo de forma predeterminada y el cifrado del lado del servidor es transparente, lo que significa que no es necesario realizar cambios en las aplicaciones.

El cifrado en reposo se integra de forma automática con AWS Key Management Service (AWS KMS) para administrar la clave de cifrado que se utiliza para cifrar las tablas. Al crear una tabla nueva o actualizar una existente, puede elegir una de las siguientes opciones de clave de AWS KMS:

- Clave propiedad de AWS: este es el tipo de cifrado predeterminado. La clave es propiedad de Amazon Keyspaces (sin cargo adicional).
- Clave administrada por el cliente): esta clave se almacena en la cuenta y usted la crea, posee y administra. Usted controla plenamente la clave administrada por el cliente (se aplican cargos de AWS KMS).

Clave AWS KMS (clave KMS)

El cifrado en reposo protege todos sus datos de Amazon Keyspaces con una clave AWS KMS. De forma predeterminada, Amazon Keyspaces utiliza una Clave propiedad de AWS, una clave de cifrado multiusuario que se crea y administra en una cuenta de servicio de Amazon Keyspaces.

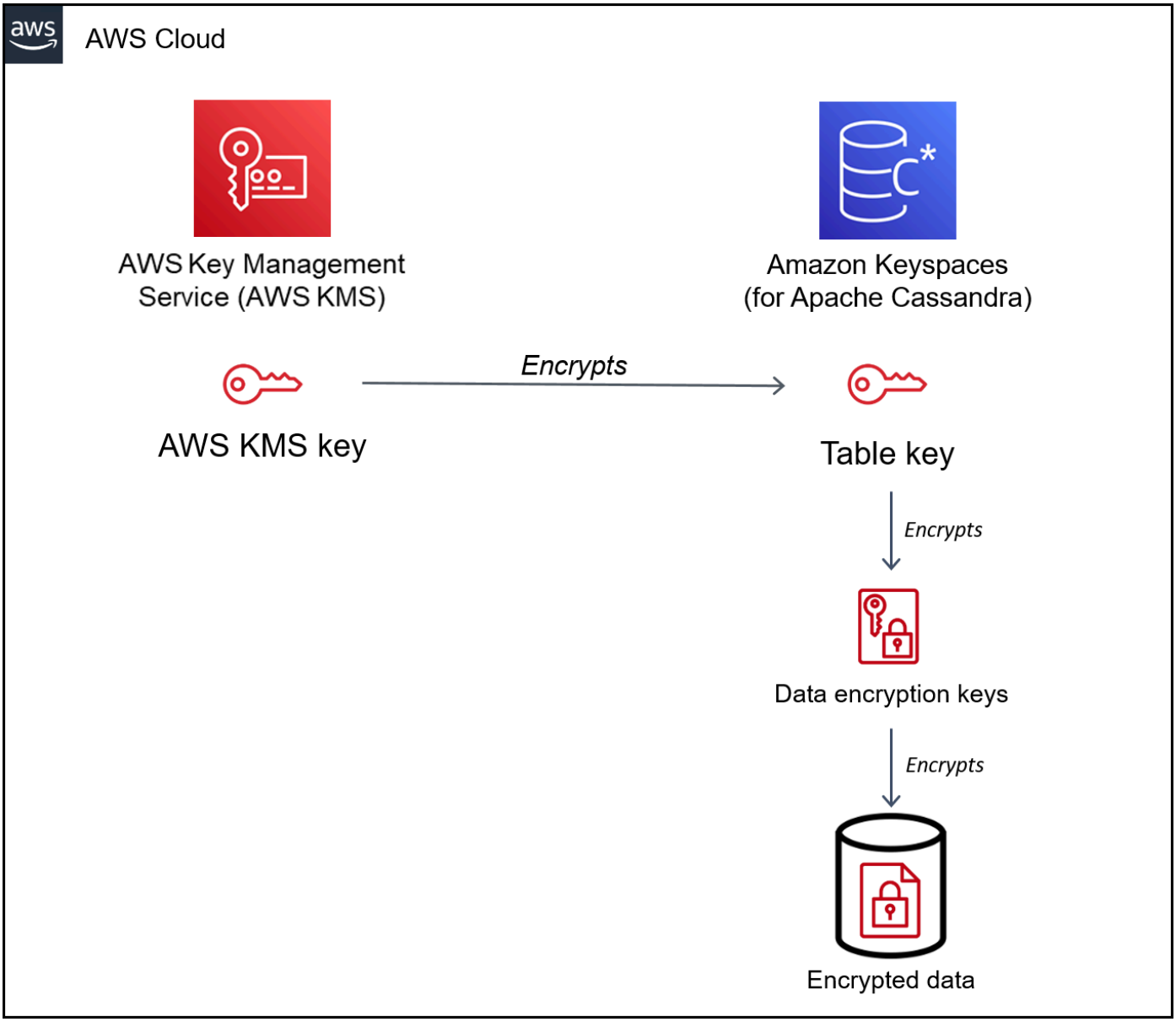
Sin embargo, puede cifrar sus tablas de Amazon Keyspaces utilizando una clave administrada por el cliente en su Cuenta de AWS. Puede seleccionar una clave KMS diferente para cada tabla de un espacio de claves. La clave KMS que seleccione para una tabla también se utiliza para cifrar todos sus metadatos y copias de seguridad restaurables.

Al crear o actualizar la tabla, seleccione la clave KMS para una tabla. Puede cambiar la clave KMS de una tabla en cualquier momento, ya sea en la consola de Amazon Keyspaces o mediante la instrucción ALTER TABLE. El proceso de cambio de claves KMS es fluido y no requiere tiempo de inactividad ni causa una degradación del servicio.

Jerarquía de claves

Amazon Keyspaces utiliza una jerarquía de claves para cifrar los datos. En esta jerarquía de claves, la clave KMS es la clave raíz. Se utiliza para cifrar y descifrar la clave de cifrado de la tabla de Amazon Keyspaces. La clave de cifrado de la tabla se utiliza para cifrar las claves de cifrado utilizadas internamente por Amazon Keyspaces para cifrar y descifrar datos al realizar operaciones de lectura y escritura.

Con la jerarquía de claves de cifrado, puede realizar cambios en la clave KMS sin tener que volver a cifrar los datos ni afectar a las aplicaciones y a las operaciones de datos en curso.



Clave de tabla

La clave de tabla de Amazon Keyspaces se utiliza como clave de cifrado de datos. Amazon Keyspaces utiliza la clave de tabla para proteger las claves internas de cifrado de datos que se utilizan para cifrar los datos almacenados en tablas, archivos de registro y copias de seguridad restaurables. Amazon Keyspaces genera una clave de cifrado de datos única para cada estructura subyacente de una tabla. Sin embargo, es posible que varias filas de una tabla estén protegidas por la misma clave de cifrado de datos.

Al establecer por primera vez la clave KMS en una clave administrada por el cliente, AWS KMS genera una clave de datos. La clave de datos de AWS KMS se refiere a la clave de tabla en Amazon Keyspaces.

Al acceder a una tabla cifrada, Amazon Keyspaces envía una solicitud a AWS KMS para que utilice la clave KMS a fin de descifrar la clave de tabla. A continuación, utiliza la clave de tabla de texto sin formato para descifrar las claves de cifrado de datos de Amazon Keyspaces y utiliza las claves de cifrado de datos de texto sin formato para descifrar los datos de la tabla.

Amazon Keyspaces utiliza y almacena la clave de tabla y las claves de cifrado de datos fuera de AWS KMS. Protege todas las claves con cifrado Advanced Encryption Standard (AES) y claves de cifrado de 256 bits. A continuación, almacena las claves cifradas con los datos cifrados para que estén disponibles para descifrar los datos de la tabla bajo demanda.

Almacenamiento en caché de las claves de tabla

Para evitar llamar a AWS KMS en cada operación de Amazon Keyspaces, Amazon Keyspaces almacena en caché las claves de la tabla en texto sin formato de cada conexión en memoria. Si Amazon Keyspaces recibe una solicitud de la clave de tabla almacenada en caché tras cinco minutos de inactividad, envía una nueva solicitud a AWS KMS para descifrar la clave de tabla. Esta llamada captura cualquier cambio realizado en las políticas de acceso de la clave KMS en AWS KMS o AWS Identity and Access Management (IAM) desde la última solicitud de descifrado de la clave de tabla.

Cifrado de sobre

Si cambia la clave administrada por el cliente para su tabla, Amazon Keyspaces genera una nueva clave de tabla. A continuación, utiliza la nueva clave de tabla para volver a cifrar las claves de cifrado de datos. También utiliza la nueva clave de tabla para cifrar las claves de tabla anteriores que se utilizan para proteger las copias de seguridad restaurables. Este proceso se denomina cifrado de sobre. Esto garantiza que pueda acceder a las copias de seguridad restaurables aunque rote la clave administrada por el cliente. Para obtener más información sobre el cifrado de sobre, consulte Cifrado de sobre en la Guía para desarrolladores de AWS Key Management Service.

Temas

- Claves propiedad de AWS
- Claves administradas por el cliente
- Notas de uso del cifrado en reposo

Claves propiedad de AWS

Las Claves propiedad de AWS no están almacenadas en su Cuenta de AWS. Forman parte de una colección de claves KMS que AWS posee y administra para su uso en múltiples Cuentas de AWS. Los servicios de AWS pueden utilizar Claves propiedad de AWS para proteger sus datos.

No puede ver, administrar o usar las Claves propiedad de AWS, ni auditar su uso. Sin embargo, no necesita hacer ninguna tarea ni cambiar ningún programa para proteger las claves que encriptan sus datos.

No se le cobra una cuota mensual ni de uso por el uso de Claves propiedad de AWS y no cuentan para las cuotas de AWS KMS de su cuenta.

Claves administradas por el cliente

Las claves administradas por el cliente son claves en su Cuenta de AWS que usted crea, posee y administra. Usted tiene el control total sobre estas claves de KMS.

Utilice una clave administrada por el cliente para obtener las siguientes características:

- Usted crea y administra la clave gestionada por el cliente, lo que incluye establecer y mantener las políticas de claves, políticas de IAM y concesiones para controlar el acceso a la clave gestionada por el cliente. Puede habilitar y deshabilitar la clave administrada por el cliente, habilitar y deshabilitar la rotación automática de claves y programar la clave administrada por el cliente para eliminarla cuando ya no esté en uso. Puede crear etiquetas y alias para las claves administradas por el cliente que administre.
- Puede utilizar una clave administrada por el cliente con material de claves importado o una clave administrada por el cliente en un almacén de claves personalizado que tenga y administre.
- Puede utilizar AWS CloudTrail y los Registros de Amazon CloudWatch para realizar un seguimiento de las solicitudes que Amazon Keyspaces envía a AWS KMS en su nombre. Para obtener más información, consulte the section called “Paso 6: Configurar el monitoreo con AWS CloudTrail”.

Las claves administradas por el cliente generan cargos por cada llamada a la API y se aplican cuotas AWS KMS a estas claves KMS. Para obtener más información, consulte cuotas de solicitudes o de recursos de AWS KMS.

Al especificar una clave administrada por el cliente como clave de cifrado raíz para una tabla, las copias de seguridad restaurables se cifran con la misma clave de cifrado que se especifica para la

tabla en el momento de crear la copia de seguridad. Si se rota la clave KMS para la tabla, el sobre de claves garantiza que la clave KMS más reciente tenga acceso a todas las copias de seguridad restaurables.

Amazon Keyspaces debe tener acceso a su clave administrada por el cliente para proporcionarle acceso a los datos de su tabla. Si el estado de la clave de cifrado es deshabilitado o su eliminación está programada, Amazon Keyspaces no puede cifrar ni descifrar datos. Como resultado, no puede realizar operaciones de lectura ni de escritura en la tabla. En cuanto el servicio detecte que su clave de cifrado es inaccesible, Amazon Keyspaces le enviará una notificación por correo electrónico para avisarle.

Debe restablecer el acceso a su clave de cifrado en un plazo de siete días o Amazon Keyspaces eliminará automáticamente su tabla. Como medida de precaución, Amazon Keyspaces crea una copia de seguridad restaurable de los datos de su tabla antes de eliminarla. Amazon Keyspaces mantiene la copia de seguridad restaurable durante 35 días. Transcurridos 35 días, ya no podrá restaurar los datos de su tabla. No se le factura por la copia de seguridad restaurable, pero se aplican los cargos de restauración estándar.

Puede utilizar esta copia de seguridad restaurable para restaurar sus datos en una nueva tabla. Para iniciar la restauración, la última clave administrada por el cliente utilizada para la tabla debe estar habilitada y Amazon Keyspaces debe tener acceso a ella.

> **Note**
>
> Al crear una tabla cifrada con una clave administrada por el cliente inaccesible o programada para eliminación antes de que finalice el proceso de creación, se produce un error. La operación de creación de la tabla falla y se le envía una notificación por correo electrónico.

Notas de uso del cifrado en reposo

Tenga en cuenta lo siguiente cuando utilice el cifrado en reposo en Amazon Keyspaces.

- El cifrado en reposo del lado del servidor está habilitado en todas las tablas de Amazon Keyspaces y no se puede deshabilitar. Toda la tabla se cifra en reposo, no puede seleccionar columnas ni filas específicas para su cifrado.
- De forma predeterminada, Amazon Keyspaces utiliza una clave predeterminada de servicio único (Clave propiedad de AWS) para cifrar todas sus tablas. Si esta clave no existe, se crea una para usted. Las claves predeterminadas de servicio no se pueden deshabilitar.

- El cifrado en reposo solo cifra los datos mientras estén estáticos (en reposo) en un medio de almacenamiento persistente. Si le preocupa la seguridad de los datos en tránsito o en uso, debe adoptar medidas adicionales:
  - Datos en tránsito: todos los datos de Amazon Keyspaces se cifran en tránsito. De forma predeterminada, las comunicaciones hacia y desde Amazon Keyspaces están protegidas mediante el cifrado Capa de conexión segura (SSL)/Seguridad de la capa de transporte (TLS).
  - Datos en uso: proteja sus datos antes de enviarlos a Amazon Keyspaces utilizando el cifrado del cliente.
  - Claves administradas por el cliente: los datos en reposo de sus tablas se cifran siempre utilizando sus claves administradas por el cliente. Sin embargo, las operaciones que realicen actualizaciones atómicas de varias filas cifran los datos temporalmente utilizando Claves propiedad de AWS durante el procesamiento. Esto incluye las operaciones de eliminación de rangos y las operaciones que accedan simultáneamente a datos estáticos y no estáticos.
- Una única clave administrada por el cliente puede tener hasta 50 000 concesiones. Cada tabla de Amazon Keyspaces asociada a una clave administrada por el cliente consume 2 concesiones. Una concesión se libera cuando se elimina la tabla. La segunda concesión se utiliza para crear una instantánea automática de la tabla para protegerla de la pérdida de datos en caso de que Amazon Keyspaces perdiera el acceso a la clave administrada por el cliente de forma involuntaria. Esta concesión se otorga 42 días después de la eliminación de la tabla.

Cifrado en reposo: Cómo utilizar claves administradas por el cliente para cifrar tablas en Amazon Keyspacess.

Puede utilizar la consola o instrucciones CQL para especificar la AWS KMS key para las tablas nuevas y actualizar las claves de cifrado de las tablas existentes en Amazon Keyspaces. En el siguiente tema se describe cómo implementar claves administradas por el cliente para tablas nuevas y existentes.

Temas

- Requisitos previos: Crear una clave administrada por el cliente utilizando AWS KMS y conceder permisos a Amazon Keyspaces
- Paso 3: Especificar una clave administrada por el cliente para una tabla nueva
- Paso 4: Actualizar la clave de cifrado de una tabla existente
- Paso 5: Utilizar el contexto de cifrado de Amazon Keyspaces en los registros
- Paso 6: Configurar el monitoreo con AWS CloudTrail

Requisitos previos: Crear una clave administrada por el cliente utilizando AWS KMS y conceder permisos a Amazon Keyspaces

Para poder proteger una tabla de Amazon Keyspaces con una clave administrada por el cliente, primero debe crear la clave en AWS Key Management Service (AWS KMS) y, a continuación, autorizar a Amazon Keyspaces para que utilice dicha clave.

Paso 1: Crear una clave administrada por el cliente utilizando AWS KMS

Para crear una clave administrada por el cliente a fin de utilizarla para proteger una tabla de Amazon Keyspaces, puede seguir los pasos indicados en Creación de claves KMS de cifrado simétrico con la consola o la API de AWS.

Paso 2: Autorizar el uso de su clave administrada por el cliente

Para que pueda elegir una clave administrada por el cliente para proteger una tabla de Amazon Keyspaces, las políticas de dicha clave administrada por el cliente deben conceder a Amazon Keyspaces permiso para utilizarla en su nombre. Usted tiene el control total sobre las políticas y concesiones de la clave administrada por el cliente. Puede proporcionar estos permisos en una política de claves, una política de IAM o una concesión.

Amazon Keyspaces no necesita autorización adicional para utilizar la Clave propiedad de AWS predeterminada para proteger las tablas de Amazon Keyspaces en su cuenta de AWS.

En los siguientes temas se muestra cómo configurar los permisos necesarios mediante las políticas de IAM y las concesiones que permiten que las tablas de Amazon Keyspaces utilicen una clave administrada por el cliente.

Temas

- Política de claves para claves administradas por el cliente
- Política de claves de ejemplo
- Uso de concesiones para autorizar a Amazon Keyspaces

Política de claves para claves administradas por el cliente

Al seleccionar una clave administrada por el cliente para proteger una tabla de Amazon Keyspaces, Amazon Keyspaces obtiene permiso para utilizar la clave administrada por el cliente en nombre de la entidad principal que realiza la selección. Esa entidad principal, un usuario o rol debe tener en la clave administrada por el cliente los permisos que Amazon Keyspaces requiere.

Como mínimo, Amazon Keyspaces requiere los siguientes permisos en una clave administrada por el cliente:

- kms:Encrypt
- kms:Decrypt
- kms:ReEncrypt* (para kms:ReEncryptFrom y kms:ReEncryptTo)
- kms:GenerateDataKey* (para kms:GenerateDataKey y kms:GenerateDataKeyWithoutPlaintext)
- kms:DescribeKey
- kms:CreateGrant

Política de claves de ejemplo

Por ejemplo, la política de claves de ejemplo siguiente proporciona solo los permisos necesarios. La política tiene las siguientes consecuencias:

- Permite a Amazon Keyspaces utilizar la clave administrada por el cliente en operaciones de cifrado y crear concesiones, pero solo cuando actúa en nombre de entidades principales de la cuenta que tienen permiso para utilizar Amazon Keyspaces. Si las entidades principales especificadas en la declaración de la política no tienen permiso para utilizar Amazon Keyspaces, la llamada falla, aunque proceda del servicio Amazon Keyspaces.
- La clave de condición kms:ViaService solo permite los permisos cuando la solicitud procede de Amazon Keyspaces en nombre de las entidades principales indicadas en la instrucción de la política. Estas entidades principales no pueden llamar a estas operaciones directamente. Tenga en cuenta que el valor `kms:ViaService`, `cassandra.*.amazonaws.com`, tiene un asterisco (*) en la posición Región. Amazon Keyspaces exige que el permiso sea independiente de cualquier Región de AWS en particular.
- Otorga a los administradores de la clave administrada por el cliente (usuarios que pueden asumir la función `db-team`) acceso de solo lectura a la clave administrada por el cliente y permiso para revocar concesiones, incluyendo las concesiones que Amazon Keyspaces requiere para proteger la tabla.
- Otorga a Amazon Keyspaces acceso de solo lectura a la clave administrada por el cliente. En este caso, Amazon Keyspaces puede llamar directamente a estas operaciones. No tiene que actuar en nombre de una entidad principal de la cuenta.

Antes de utilizar una política de claves de ejemplo, sustituya las entidades principales de ejemplo por las entidades principales reales de su cuenta de Cuenta de AWS.

```
{
  "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": "*"
    }
  ]
```

```
}
```

Uso de concesiones para autorizar a Amazon Keyspaces

Además de las políticas de claves, Amazon Keyspaces utiliza concesiones para establecer permisos en una clave administrada por el cliente. Para ver las concesiones de una clave administrada por el cliente en su cuenta, utilice la operación ListGrants. Amazon Keyspaces no necesita concesiones, ni ningún permiso adicional, para utilizar la Clave propiedad de AWS para proteger su tabla.

Amazon Keyspaces utiliza los permisos de concesión cuando realiza tareas de mantenimiento del sistema en segundo plano y de protección continua de datos. También utiliza las concesiones para generar las claves de la tabla.

Cada concesión es específica de una tabla. Si la cuenta incluye múltiples tablas cifradas bajo la misma clave administrada por el cliente, existe una concesión de cada tipo para cada tabla. La concesión está limitada por el contexto de cifrado de Amazon Keyspaces, que incluye el nombre de la tabla y el ID de la Cuenta de AWS. La concesión incluye el permiso para retirar la concesión si ya no fuera necesaria.

Para crear las concesiones, Amazon Keyspaces debe tener permiso para llamar a `CreateGrant` en nombre del usuario que creó la tabla cifrada.

La política de claves también puede permitir a la cuenta revocar la concesión en la clave administrada por el cliente. Sin embargo, si revoca la concesión en una tabla cifrada activa, Amazon Keyspaces no podrá proteger ni mantener la tabla.

Paso 3: Especificar una clave administrada por el cliente para una tabla nueva

Siga estos pasos para especificar la clave administrada por el cliente en una tabla nueva mediante la consola de Amazon Keyspaces o CQL.

Creación de una tabla cifrada utilizando una clave administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tables (Tablas) y, a continuación, seleccione Create table (Crear tabla).
3. En la página Crear tabla, en la sección Detalles de la tabla, seleccione un espacio de claves y proporcione un nombre para la nueva tabla.

4. En la sección Esquema, cree el esquema para su tabla.
5. En la sección Configuración de la tabla, elija Personalizar configuración.
6. Continúe en Configuración del cifrado.

   En este paso, usted selecciona la configuración del cifrado para la tabla.

   En la sección Cifrado en reposo, en Elegir una AWS KMS key, elija la opción Elegir una clave KMS diferente (avanzada) y, en el campo de búsqueda, elija una AWS KMS key o introduzca un nombre de recurso de Amazon (ARN).

   > **Note**
   >
   > Si la clave que ha seleccionado no es accesible o carece de los permisos necesarios, consulte Solución de problemas de acceso a la clave en la Guía para desarrolladores de AWS Key Management Service.

7. Elija Crear para crear la tabla cifrada.

Creación de una tabla nueva mediante una clave administrada por el cliente para el cifrado en reposo (CQL)

Para crear una nueva tabla que utilice una clave administrada por el cliente para el cifrado en reposo, puede utilizar la instrucción `CREATE TABLE` como se muestra en el siguiente ejemplo. Asegúrese de sustituir el ARN de la clave por el ARN de una clave válida con permisos concedidos a Amazon Keyspaces.

```
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'
            }
    };
```

Si recibe una `Invalid Request Exception`, debe confirmar que la clave administrada por el cliente sea válida y que Amazon Keyspaces disponga de los permisos necesarios. Para confirmar que la clave se haya configurado correctamente, consulte Solución de problemas de acceso a la clave en la Guía para desarrolladores de AWS Key Management Service.

Paso 4: Actualizar la clave de cifrado de una tabla existente

También puede utilizar la consola de Amazon Keyspaces o CQL para cambiar las claves de cifrado de una tabla existente entre una Clave propiedad de AWS y una clave KMS administrada por el cliente en cualquier momento.

Actualización de una tabla existente con la nueva clave administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. En el panel de navegación, elija Tables (Tablas).
3. Elija la tabla que desee actualizar y luego la pestaña Configuración adicional.
4. En la sección Cifrado en reposo, elija Administrar cifrado para editar la configuración del cifrado de la tabla.

   En Elegir una AWS KMS key, elija la opción Elegir una clave KMS diferente (avanzada) y, en el campo de búsqueda, elija una AWS KMS key o introduzca un nombre de recurso de Amazon (ARN).

   > **Note**
   >
   > Si la clave que ha seleccionado no es válida, consulte Solución de problemas de acceso a la clave en la Guía para desarrolladores de AWS Key Management Service.

   Como alternativa, también puede elegir una Clave propiedad de AWS para una tabla que esté cifrada con una clave administrada por el cliente.
5. Elija Guardar para guardar los cambios en la tabla.

Actualización de la clave de cifrado utilizada en una tabla existente

Para cambiar la clave de cifrado de una tabla existente, utilice la instrucción `ALTER TABLE` para especificar una clave administrada por el cliente para el cifrado en reposo. Asegúrese de sustituir el ARN de la clave por el ARN de una clave válida con permisos concedidos a Amazon Keyspaces.

```
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'
                }
        };
```

Si recibe una `Invalid Request Exception`, debe confirmar que la clave administrada por el cliente sea válida y que Amazon Keyspaces disponga de los permisos necesarios. Para confirmar que la clave se haya configurado correctamente, consulte Solución de problemas de acceso a la clave en la Guía para desarrolladores de AWS Key Management Service.

Para volver a cambiar la clave de cifrado a la opción predeterminada de cifrado en reposo con las Claves propiedad de AWS, puede utilizar la instrucción `ALTER TABLE` como se muestra en el siguiente ejemplo.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
               'encryption_specification':{
                    'encryption_type' : 'AWS_OWNED_KMS_KEY'
                }
        };
```

Paso 5: Utilizar el contexto de cifrado de Amazon Keyspaces en los registros

Un contexto de cifrado es un conjunto de pares de clave-valor que contienen datos no secretos arbitrarios. Cuando se incluye un contexto de cifrado en una solicitud para cifrar datos, AWS KMS vincula criptográficamente el contexto de cifrado a los datos cifrados. Para descifrar los datos, es necesario pasar el mismo contexto de cifrado.

Amazon Keyspaces utiliza el mismo contexto de cifrado en todas las operaciones de cifrado de AWS KMS. Si utiliza una clave administrada por el cliente para proteger su tabla de Amazon Keyspaces, puede utilizar el contexto de cifrado para identificar el uso de la clave administrada por el cliente en los registros de auditoría y en los registros. También aparece en texto sin formato en los registros, como en los registros de AWS CloudTrail y en los Registros de Amazon CloudWatch.

En su solicitudes a AWS KMS, Amazon Keyspaces utiliza un contexto de cifrado con tres pares clave-valor.

```
"encryptionContextSubset": {
    "aws:cassandra:keyspaceName": "my_keyspace",
    "aws:cassandra:tableName": "mytable"
    "aws:cassandra:subscriberId": "111122223333"
```

```
}
```

- Espacio de claves: el primer par clave-valor identifica el espacio de claves que incluye la tabla que Amazon Keyspaces está cifrando. La clave es `aws:cassandra:keyspaceName`. El valor es el nombre del espacio de claves.

  ```
  "aws:cassandra:keyspaceName": "<keyspace-name>"
  ```

  Por ejemplo:

  ```
  "aws:cassandra:keyspaceName": "my_keyspace"
  ```

- Tabla: el segundo par clave-valor identifica la tabla que Amazon Keyspaces está cifrando. La clave es `aws:cassandra:tableName`. El valor es el nombre de la tabla.

  ```
  "aws:cassandra:tableName": "<table-name>"
  ```

  Por ejemplo:

  ```
  "aws:cassandra:tableName": "my_table"
  ```

- Cuenta: el tercer par clave-valor identifica la Cuenta de AWS. La clave es `aws:cassandra:subscriberId`. El valor es el ID de la cuenta.

  ```
  "aws:cassandra:subscriberId": "<account-id>"
  ```

  Por ejemplo:

  ```
  "aws:cassandra:subscriberId": "111122223333"
  ```

Paso 6: Configurar el monitoreo con AWS CloudTrail

Si utiliza una clave administrada por el cliente para proteger sus tablas de Amazon Keyspaces, puede utilizar los registros de AWS CloudTrail para realizar un seguimiento de las solicitudes que Amazon Keyspaces envía a AWS KMS en su nombre.

Las solicitudes `GenerateDataKey`, `DescribeKey`, `Decrypt` y `CreateGrant` se tratan en esta sección. Además, Amazon Keyspaces utiliza una operación RetireGrant para eliminar una concesión cuando usted elimina una tabla.

GenerateDataKey

Amazon Keyspaces crea una clave de tabla única para cifrar los datos en reposo. Envía una solicitud GenerateDataKey a AWS KMS que especifica la clave KMS de la tabla.

El evento que registra la operación `GenerateDataKey` es similar al siguiente evento de ejemplo. El usuario es la cuenta de servicio de Amazon Keyspaces. Los parámetros incluyen el Nombre de recurso de Amazon (ARN) de la clave administrada por el cliente, un especificador de clave que requiere una clave de 256 bits y el contexto de cifrado que identifica la tabla y la Cuenta de AWS.

```
{
    "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

Además, Amazon Keyspaces utiliza una operación DescribeKey para determinar si la clave KMS que ha seleccionado existe en la cuenta y región.

El evento que registra la operación `DescribeKey` es similar al siguiente evento de ejemplo. El usuario es la cuenta de servicio de Amazon Keyspaces. Los parámetros incluyen el ARN de la clave administrada por el cliente y un especificador de clave que requiere una clave de 256 bits.

```
{
    "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

Al acceder a una tabla de Amazon Keyspaces cifrada, Amazon Keyspaces necesita descifrar la clave de la tabla de manera que pueda descifrar las claves de menor nivel en la jerarquía. A continuación, descifra los datos de la tabla. Para descifrar la clave de la tabla, Amazon Keyspaces envía una solicitud Decrypt a AWS KMS que especifica la clave KMS de la tabla.

El evento que registra la operación `Decrypt` es similar al siguiente evento de ejemplo. El usuario es la entidad principal en su cuenta de Cuenta de AWS que está accediendo a la tabla. Los parámetros incluyen la clave cifrada de la tabla (como un blob de texto cifrado) y el contexto de cifrado que identifica la tabla y la Cuenta de AWS. AWS KMS deduce el ID de la clave administrada por el cliente a partir del texto cifrado.

```
{
    "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

Al utilizar una clave administrada por el cliente para proteger su tabla de Amazon Keyspaces, Amazon Keyspaces utiliza concesiones para permitir que el servicio realice tareas continuas de protección, mantenimiento y durabilidad de los datos. Estas concesiones no son necesarias en las Claves propiedad de AWS.

Las concesiones que Amazon Keyspaces crea son específicas de una tabla. El principal en la solicitud CreateGrant es el usuario que creó la tabla.

El evento que registra la operación `CreateGrant` es similar al siguiente evento de ejemplo. Los parámetros incluyen el ARN de la clave administrada por el cliente para la tabla, la entidad principal que recibe la concesión y la entidad principal que la retira (el servicio Amazon Keyspaces), y las operaciones que cubre la concesión. También incluye una restricción que exige que todas las operaciones de cifrado utilicen el contexto de cifrado especificado.

```
{
    "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"
}
```

Cifrado en tránsito en Amazon Keyspaces

Amazon Keyspaces solo acepta conexiones seguras utilizando Seguridad de la capa de transporte (TLS). El cifrado en tránsito proporciona una capa adicional de protección de datos al cifrar sus datos mientras viajan hacia y desde Amazon Keyspaces. Las políticas de la organización, las normativas industriales o gubernamentales y los requisitos de conformidad suelen requerir el uso del cifrado en

tránsito para aumentar la seguridad de los datos de las aplicaciones cuando transmiten datos a la red.

Para obtener información sobre cómo cifrar las conexiones de `cqlsh` a Amazon Keyspaces utilizando TLS, consulte the section called “Configuración manual de conexiones de `cqlsh` para TLS”. Para obtener información sobre cómo utilizar el cifrado TLS con controladores de cliente, consulte the section called “Utilización de un controlador de cliente de Cassandra”.

Privacidad del tráfico entre redes en Amazon Keyspaces

En este tema se describe cómo Amazon Keyspaces (para Apache Cassandra) asegura las conexiones desde las aplicaciones en las instalaciones a Amazon Keyspaces y entre Amazon Keyspaces y otros recursos de AWS dentro de la misma Región de AWS.

Tráfico entre el servicio y las aplicaciones y clientes locales

Tiene dos opciones de conectividad entre su red privada y AWS:

- Una conexión de AWS Site-to-Site VPN. Para obtener más información, consulte ¿Qué es AWS Site-to-Site VPN? en la Guía del usuario de AWS Site-to-Site VPN.
- Una conexión de AWS Direct Connect. Para obtener más información, consulte ¿Qué es AWS Direct Connect? en la Guía del usuario de AWS Direct Connect.

Como se trata de un servicio administrado, Amazon Keyspaces (para Apache Cassandra) está protegido por la seguridad de red global de AWS. Para obtener información sobre los servicios de seguridad de AWS y cómo AWS protege la infraestructura, consulte Seguridad en la nube de AWS. Para diseñar su entorno de AWS con las prácticas recomendadas de seguridad de infraestructura, consulte Protección de la infraestructura en Portal de seguridad de AWS Well-Architected Framework.

Puede utilizar llamadas a la API publicadas de AWS para acceder a Amazon Keyspaces a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Nosotros exigimos TLS 1.2 y recomendamos TLS 1.3.
- Conjuntos de cifrado con confidencialidad directa total (PFS) como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad de seguridad de IAM. También puede utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Amazon Keyspaces admite dos métodos de autenticación de solicitudes de clientes. El primer método utiliza credenciales específicas del servicio, que son credenciales basadas en contraseña generadas para un usuario de IAM específico. Puede crear y administrar la contraseña mediante la consola de IAM, la API AWS CLI o la API AWS. Para obtener más información, consulte Uso de IAM con Amazon Keyspaces.

El segundo método utiliza un complemento de autenticación para el controlador Java DataStax de código abierto para Cassandra. Este complemento permite a los usuarios, roles e identidades federadas de IAM añadir información de autenticación a las solicitudes de la API de Amazon Keyspaces (para Apache Cassandra) mediante el proceso AWS Signature Version 4 (SigV4). Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Tráfico entre recursos de AWS en la misma región

Los puntos de conexión de VPC de interfaz habilitan la comunicación privada entre su nube privada virtual (VPC), que se ejecuta en Amazon VPC, y Amazon Keyspaces. Los puntos de conexión de VPC de interfaz funcionan con AWS PrivateLink, que es un servicio de AWS que habilita la comunicación privada entre las VPC y los servicios de AWS. AWS PrivateLink lo permite mediante el uso de una interfaz de red elástica con IP privadas en su VPC para que el tráfico de red no salga de la red de Amazon. Los puntos de conexión de VPC de interfaz no requieren una puerta de enlace de Internet, un dispositivo NAT, una conexión de VPN ni una conexión de AWS Direct Connect. Para obtener más información, consulte Nube privada virtual de Amazon e Interfaz de puntos de conexión de VPC (AWS PrivateLink). Para ver ejemplos de políticas, consulte the section called “Uso de puntos de conexión de VPC de interfaz para Amazon Keyspaces”.

AWS Identity and Access Management para Amazon Keyspaces

AWS Identity and Access Management (IAM) es una herramienta Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a los AWS recursos. Los administradores de IAM controlan quién se puede autenticar (iniciar sesión) y autorizar (tener permisos) para utilizar los recursos de Amazon Keyspaces. La IAM es una Servicio de AWS herramienta que puede utilizar sin coste adicional.

Temas

- Público
- Autenticación con identidades
- Administración de acceso mediante políticas
- Funcionamiento de Amazon Keyspaces con IAM
- Ejemplos de políticas basadas en identidades de Amazon Keyspaces
- Políticas administradas de AWS para Amazon Keyspaces
- Solución de problemas de identidad y acceso a Amazon Keyspaces
- Uso de roles vinculados a servicios para Amazon Keyspaces

Público

La forma de usar AWS Identity and Access Management (IAM) varía en función del trabajo que se realice en Amazon Keyspaces.

Usuario de servicio: si utiliza el servicio de Amazon Keyspaces para realizar su trabajo, entonces su administrador le proporciona las credenciales y los permisos que necesita. A medida que utilice más características de Amazon Keyspaces para realizar su trabajo, es posible que necesite permisos adicionales. Entender cómo se administra el acceso puede ayudarlo a solicitar los permisos correctos al administrador. Si no puede acceder a una característica de Amazon Keyspaces, consulte Solución de problemas de identidad y acceso a Amazon Keyspaces.

Administrador de servicio: si está a cargo de los recursos de Amazon Keyspaces en su empresa, es probable que tenga acceso total a Amazon Keyspaces. Su trabajo consiste en determinar a qué características y recursos de Amazon Keyspaces deben tener acceso los usuarios de su servicio. Luego, debe enviar solicitudes a su administrador de IAM para cambiar los permisos de los usuarios de su servicio. Revise la información de esta página para conocer los conceptos básicos de IAM. Para obtener más información sobre cómo su empresa puede utilizar IAM con Amazon Keyspaces, consulte Funcionamiento de Amazon Keyspaces con IAM.

Administrador de IAM: si es administrador de IAM, es posible que desee obtener información detallada sobre cómo puede redactar políticas para administrar el acceso a Amazon Keyspaces. Para consultar ejemplos de políticas basadas en identidades de Amazon Keyspaces que puede utilizar en IAM, consulte Ejemplos de políticas basadas en identidades de Amazon Keyspaces.

Autenticación con identidades

La autenticación es la forma de iniciar sesión AWS con sus credenciales de identidad. Debe estar autenticado (con quien haya iniciado sesión AWS) como usuario de IAM o asumiendo una función de IAM. Usuario raíz de la cuenta de AWS

Puede iniciar sesión AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. AWS IAM Identity Center Los usuarios (Centro de identidades de IAM), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, su administrador habrá configurado previamente la federación de identidades mediante roles de IAM. Cuando accedes AWS mediante la federación, estás asumiendo un rol de forma indirecta.

Según el tipo de usuario que sea, puede iniciar sesión en el portal AWS Management Console o en el de AWS acceso. Para obtener más información sobre cómo iniciar sesión AWS, consulte Cómo iniciar sesión Cuenta de AWS en su Guía del AWS Sign-In usuario.

Si accede AWS mediante programación, AWS proporciona un kit de desarrollo de software (SDK) y una interfaz de línea de comandos (CLI) para firmar criptográficamente sus solicitudes con sus credenciales. Si no utilizas AWS herramientas, debes firmar las solicitudes tú mismo. Para obtener más información sobre cómo usar el método recomendado para firmar las solicitudes usted mismo, consulte Firmar las solicitudes de la AWS API en la Guía del usuario de IAM.

Independientemente del método de autenticación que use, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, le AWS recomienda que utilice la autenticación multifactor (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactor en la Guía del usuario de AWS Single Sign-On y Uso de la autenticación multifactor (MFA) en AWS en la Guía del usuario de IAM.

Cuenta de AWS usuario root

Al crear una Cuenta de AWS, comienza con una identidad de inicio de sesión que tiene acceso completo a todos Servicios de AWS los recursos de la cuenta. Esta identidad se denomina usuario Cuenta de AWS raíz y se accede a ella iniciando sesión con la dirección de correo electrónico y la contraseña que utilizaste para crear la cuenta. Recomendamos encarecidamente que no utilice el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para ver la lista completa de las tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Usuarios y grupos de IAM

Un usuario de IAM es una identidad propia Cuenta de AWS que tiene permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos emplear credenciales temporales, en lugar de crear usuarios de IAM que tengan credenciales de larga duración como contraseñas y claves de acceso. No obstante, si tiene casos de uso específicos que requieran credenciales de larga duración con usuarios de IAM, recomendamos rotar las claves de acceso. Para más información, consulte Rotar las claves de acceso periódicamente para casos de uso que requieran credenciales de larga duración en la Guía del usuario de IAM.

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puede iniciar sesión como grupo. Puede usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos de grandes conjuntos de usuarios. Por ejemplo, podría tener un grupo cuyo nombre fuese IAMAdmins y conceder permisos a dicho grupo para administrar los recursos de IAM.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales permanentes a largo plazo y los roles proporcionan credenciales temporales. Para más información, consulte Cuándo crear un usuario de IAM (en lugar de un rol) en la Guía del usuario de IAM.

Roles de IAM

Un rol de IAM es una identidad dentro de usted Cuenta de AWS que tiene permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una determinada persona. Puede asumir temporalmente una función de IAM en el AWS Management Console cambiando de función. Puede asumir un rol llamando a una operación de AWS API AWS CLI o utilizando una URL personalizada. Para más información sobre los métodos para el uso de roles, consulte Uso de roles de IAM en la Guía del usuario de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información acerca de roles para federación, consulte Creación de un rol para un proveedor de identidades de terceros en la Guía del usuario de IAM. Si utiliza el IAM Identity Center, debe configurar un conjunto de permisos. El IAM Identity Center correlaciona el conjunto de permisos con un rol en IAM para controlar a qué

pueden acceder las identidades después de autenticarse. Para obtener información acerca de los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS Single Sign-On.

- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.
- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, con algunas Servicios de AWS, puedes adjuntar una política directamente a un recurso (en lugar de usar un rol como proxy). Para obtener información acerca de la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulte Cómo los roles de IAM difieren de las políticas basadas en recursos en la Guía del usuario de IAM.
- Acceso entre servicios: algunos Servicios de AWS utilizan funciones en otros Servicios de AWS. Por ejemplo, cuando realiza una llamada en un servicio, es común que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado al servicio.
  - Sesiones de acceso directo (FAS): cuando utilizas un usuario o un rol de IAM para realizar acciones en ellas AWS, se te considera director. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. El FAS utiliza los permisos del principal que llama Servicio de AWS y los solicita Servicio de AWS para realizar solicitudes a los servicios descendentes. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.
  - Rol de servicio: un rol de servicio es un rol de IAM que adopta un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.
  - Función vinculada al servicio: una función vinculada a un servicio es un tipo de función de servicio que está vinculada a un. Servicio de AWS El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

- Aplicaciones que se ejecutan en Amazon EC2: puede usar un rol de IAM para administrar las credenciales temporales de las aplicaciones que se ejecutan en una instancia EC2 y realizan AWS CLI solicitudes a la API. AWS Es preferible hacerlo de este modo a almacenar claves de acceso en la instancia EC2. Para asignar una AWS función a una instancia EC2 y ponerla a disposición de todas sus aplicaciones, debe crear un perfil de instancia adjunto a la instancia. Un perfil de instancia contiene el rol y permite a los programas que se ejecutan en la instancia EC2 obtener credenciales temporales. Para más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2 en la Guía del usuario de IAM.

Para obtener información sobre el uso de los roles de IAM, consulte Cuándo crear un rol de IAM (en lugar de un usuario) en la Guía del usuario de IAM.

Administración de acceso mediante políticas

El acceso se controla AWS creando políticas y adjuntándolas a AWS identidades o recursos. Una política es un objeto AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando un director (usuario, usuario raíz o sesión de rol) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan AWS como documentos JSON. Para obtener más información sobre la estructura y el contenido de los documentos de política JSON, consulte Información general de políticas JSON en la Guía del usuario de IAM.

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Para conceder permiso a los usuarios para realizar acciones en los recursos que necesiten, un administrador de IAM puede crear políticas de IAM. A continuación, el administrador puede añadir las políticas de IAM a roles y los usuarios pueden asumirlos.

Las políticas de IAM definen permisos para una acción independientemente del método que se utilice para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con esa política puede obtener información sobre el rol de la API AWS Management Console AWS CLI, la o la AWS API.

Políticas basadas en identidades

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo o rol. Las políticas administradas son políticas independientes que puede adjuntar a varios usuarios, grupos y roles de su Cuenta de AWS empresa. Las políticas administradas incluyen políticas AWS administradas y políticas administradas por el cliente. Para más información sobre cómo elegir una política administrada o una política insertada, consulte Elegir entre políticas administradas y políticas insertadas en la Guía del usuario de IAM.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No puedes usar políticas AWS gestionadas de IAM en una política basada en recursos.

Listas de control de acceso (ACL)

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Amazon S3 y Amazon VPC son ejemplos de servicios que admiten las ACL. AWS WAF Para obtener más información sobre las ACL, consulte Información general de Lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite tipos de políticas adicionales y menos comunes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una característica avanzada que le permite establecer los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM (usuario o rol de IAM). Puede establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad y los límites de permisos. Las políticas basadas en recursos que especifique el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de los permisos, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.
- Políticas de control de servicios (SCP): las SCP son políticas de JSON que especifican los permisos máximos para una organización o unidad organizativa (OU). AWS Organizations AWS Organizations es un servicio para agrupar y gestionar de forma centralizada varios de los Cuentas de AWS que son propiedad de su empresa. Si habilita todas las características en una organización, entonces podrá aplicar políticas de control de servicio (SCP) a una o a todas sus cuentas. El SCP limita los permisos de las entidades en las cuentas de los miembros, incluidas las de cada una. Usuario raíz de la cuenta de AWS Para obtener más información acerca de Organizations y las SCP, consulte Funcionamiento de las SCP en la Guía del usuario de AWS Organizations.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Los permisos también pueden proceder de una política en función de recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para más información, consulte Políticas de sesión en la Guía del usuario de IAM.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para saber cómo AWS determinar si se debe permitir una solicitud cuando se trata de varios tipos de políticas, consulte la lógica de evaluación de políticas en la Guía del usuario de IAM.

Funcionamiento de Amazon Keyspaces con IAM

Antes de utilizar IAM para administrar el acceso a Amazon Keyspaces, debe comprender qué características de IAM están disponibles para uso con Amazon Keyspaces. Para obtener una visión general de cómo funcionan Amazon Keyspaces y otros AWS servicios con IAM, consulte los AWS servicios que funcionan con IAM en la Guía del usuario de IAM.

Temas

- Políticas basadas en identidades de Amazon Keyspaces
- Políticas basadas en recursos de Amazon Keyspaces
- Autorización basada en etiquetas de Amazon Keyspaces
- Roles de IAM de Amazon Keyspaces

Políticas basadas en identidades de Amazon Keyspaces

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. Amazon Keyspaces admite acciones y recursos específicos, así como claves de condición. Para obtener más información acerca de los elementos que utiliza en una política de JSON, consulte Referencia de los elementos de las políticas de JSON de IAM en la Guía del usuario de IAM.

Para ver los recursos y acciones específicos del servicio Amazon Keyspaces y las claves de contexto de condición que se pueden utilizar para las políticas de permisos de IAM, consulte Acciones, recursos y claves de condición para Amazon Keyspaces (para Apache Cassandra) en la Referencia de autorización de servicios.

Acciones

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Las acciones políticas suelen tener el mismo nombre que la operación de AWS API asociada. Hay algunas excepciones, como acciones de solo permiso que no tienen una operación de API coincidente. También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Las acciones de política en Amazon Keyspaces utilizan el siguiente prefijo antes de la acción: `cassandra:`. Por ejemplo, para conceder a alguien permiso para crear un espacio de claves de Amazon Keyspaces con la instrucción CQL `CREATE` de Amazon Keyspaces, incluya la acción `cassandra:Create` en su política. Las instrucciones de la política deben incluir un elemento `Action` o un elemento `NotAction`. Amazon Keyspaces define su propio conjunto de acciones que describen las tareas que puede realizar con este servicio.

Para especificar varias acciones en una única instrucción, sepárelas con comas del siguiente modo:

```
"Action": [
      "cassandra:CREATE",
      "cassandra:MODIFY"
          ]
```

Para ver una lista de las acciones de Amazon Keyspaces, consulte Acciones definidas por Amazon Keyspaces (para Apache Cassandra) en la Referencia de autorización de servicios.

Recursos

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). Puede hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

En Amazon Keyspaces se pueden utilizar espacios de claves y tablas en el elemento `Resource` de los permisos IAM.

El recurso de espacio de claves de Amazon Keyspaces tiene el siguiente ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${KeyspaceName}/
```

El recurso de tabla de Amazon Keyspaces tiene el siguiente ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${KeyspaceName}/table/
${tableName}
```

Para obtener más información sobre el formato de los ARN, consulte Nombres de recursos de Amazon (ARN) y espacios de nombres de AWS servicios.

Por ejemplo, para especificar el espacio de claves `mykeyspace` en su instrucción, utilice el siguiente ARN:

```
"Resource": "arn:aws:cassandra:us-east-1:123456789012:/keyspace/mykeyspace/"
```

Para especificar todos los espacios de claves que pertenezcan a una cuenta en concreto, utilice el comodín (*):

```
"Resource": "arn:aws:cassandra:us-east-1:123456789012:/keyspace/*"
```

Algunas acciones de Amazon Keyspaces, como las de creación de recursos, no se pueden realizar en un recurso específico. En dichos casos, debe utilizar el carácter comodín (*).

```
"Resource": "*"
```

Para conectarse a Amazon Keyspaces mediante programación con un controlador estándar, una entidad principal debe tener acceso SELECT a las tablas del sistema, dado que la mayoría de los controladores leen los espacios de claves/tablas del sistema en la conexión. Por ejemplo, para conceder permisos `SELECT` a un usuario de IAM para `mytable` en `mykeyspace`, la entidad principal debe tener permisos para leer ambos, `mytable` y `system keyspace`. Para especificar varios recursos en una única instrucción, separe los ARN con comas.

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
```

Para ver una lista de los tipos de recursos de Amazon Keyspaces y sus ARN, consulte Recursos definidos por Amazon Keyspaces (para Apache Cassandra) en la Referencia de autorización de

servicios. Para obtener información sobre las acciones con las que puede especificar el ARN de cada recurso, consulte Acciones definidas por Amazon Keyspaces (para Apache Cassandra).

Claves de condición

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` (o bloque de `Condition`) permite especificar condiciones en las que entra en vigor una instrucción. El elemento `Condition` es opcional. Puede crear expresiones condicionales que utilicen operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud.

Si especifica varios elementos de `Condition` en una instrucción o varias claves en un único elemento de `Condition`, AWS las evalúa mediante una operación lógica `AND`. Si especifica varios valores para una única clave de condición, AWS evalúa la condición mediante una `OR` operación lógica. Se deben cumplir todas las condiciones antes de que se concedan los permisos de la instrucción.

También puede utilizar variables de marcador de posición al especificar condiciones. Por ejemplo, puede conceder un permiso de usuario de IAM para acceder a un recurso solo si está etiquetado con su nombre de usuario de IAM. Para más información, consulte Elementos de la política de IAM: variables y etiquetas en la Guía del usuario de IAM.

AWS admite claves de condición globales y claves de condición específicas del servicio. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario de IAM.

Amazon Keyspaces define su propio conjunto de claves de condición y también admite el uso de algunas claves de condición globales. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario de IAM.

Todas las acciones de Amazon Keyspaces admiten las claves de condición `aws:RequestTag/${TagKey}`, `aws:ResourceTag/${TagKey}` y `aws:TagKeys`. Para obtener más información, consulte the section called “ Acceso a recursos de Amazon Keyspaces basado en etiquetas”.

Para ver una lista de las claves de condición de Amazon Keyspaces, consulte Claves de condición para Amazon Keyspaces (para Apache Cassandra) en la Referencia de autorización de servicios. Para obtener información sobre con qué acciones y recursos puede utilizar una clave de condición, consulte Acciones definidas por Amazon Keyspaces (para Apache Cassandra).

Ejemplos

Para ver ejemplos de políticas basadas en identidades de Amazon Keyspaces, consulte Ejemplos de políticas basadas en identidades de Amazon Keyspaces.

Políticas basadas en recursos de Amazon Keyspaces

Amazon Keyspaces no admite políticas basadas en recursos. Para ver un ejemplo de una página detallada de política basada en recursos, consulte https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html.

Autorización basada en etiquetas de Amazon Keyspaces

Puede administrar el acceso a sus recursos de Amazon Keyspaces mediante etiquetas. Para administrar el acceso a recursos en función de etiquetas, proporcione información de etiquetas en el elemento de condición de una política utilizando las claves de condición `cassandra:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`. Para obtener más información sobre el etiquetado de recursos de Amazon Keyspaces, consulte the section called “Trabajar con etiquetas”.

Para ver ejemplos de políticas basadas en identidad para limitar el acceso a un recurso en función de las etiquetas de ese recurso, consulte Acceso a recursos de Amazon Keyspaces basado en etiquetas.

Roles de IAM de Amazon Keyspaces

Un rol de IAM es una entidad dentro de usted Cuenta de AWS que tiene permisos específicos.

Uso de credenciales temporales con Amazon Keyspaces

Puede utilizar credenciales temporales para iniciar sesión con federación, asumir un rol de IAM o asumir un rol de acceso entre cuentas. Para obtener credenciales de seguridad temporales, puede llamar a operaciones de AWS STS API como AssumeRoleo GetFederationToken.

Amazon Keyspaces admite el uso de credenciales temporales con el complemento de autenticación AWS Signature Version 4 (SiGv4) disponible en el repositorio de Github para los siguientes idiomas:

- 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.

- Go: https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin.

Para ver ejemplos y tutoriales que implementan el complemento de autenticación para acceder a Amazon Keyspaces mediante programación, consulte. the section called “Utilización de un controlador de cliente de Cassandra”

Roles vinculados al servicio

Los roles vinculados a un servicio permiten a AWS los servicios acceder a los recursos de otros servicios para completar una acción en su nombre. Los roles vinculados a servicios aparecen en la cuenta de IAM y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Para obtener detalles sobre creación o administración de roles vinculados a servicios de Amazon Keyspaces, consulte the section called “Uso de roles vinculados a servicios”.

Roles de servicio

Amazon Keyspaces no admite roles de servicio.

Ejemplos de políticas basadas en identidades de Amazon Keyspaces

De forma predeterminada, los usuarios y roles de IAM no tienen permiso para crear ni modificar recursos de Amazon Keyspaces. Tampoco pueden realizar tareas mediante la consola, el CQLSH o la API. AWS CLI AWS Un administrador de IAM debe crear políticas de IAM que concedan permisos a los usuarios y a los roles para realizar operaciones de la API concretas en los recursos especificados que necesiten. El administrador debe adjuntar esas políticas a los usuarios o grupos de IAM que necesiten esos permisos.

Para obtener más información acerca de cómo crear una política basada en identidad de IAM con estos documentos de políticas de JSON de ejemplo, consulte Creación de políticas en la pestaña JSON en la Guía del usuario de IAM.

Temas

- Prácticas recomendadas relativas a políticas
- Uso de la consola de Amazon Keyspaces
- Cómo permitir a los usuarios consultar sus propios permisos
- Acceso a las tablas de Amazon Keyspaces
- Acceso a recursos de Amazon Keyspaces basado en etiquetas

Prácticas recomendadas relativas a políticas

Las políticas basadas en identidades determinan si alguien puede crear, acceder o eliminar recursos de Amazon Keyspaces en su cuenta. Estas acciones pueden generar costos adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas AWS administradas y avance hacia los permisos con privilegios mínimos: para empezar a conceder permisos a sus usuarios y cargas de trabajo, utilice las políticas AWS administradas que otorgan permisos para muchos casos de uso comunes. Están disponibles en su. Cuenta de AWS Le recomendamos que reduzca aún más los permisos definiendo políticas administradas por el AWS cliente que sean específicas para sus casos de uso. Con el fin de obtener más información, consulte las políticas administradas por AWS o las políticas administradas por AWS para funciones de trabajo en la Guía del usuario de IAM.
- Aplique permisos de privilegio mínimo: cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Con el fin de obtener más información sobre el uso de IAM para aplicar permisos, consulte Políticas y permisos en IAM en la Guía del usuario de IAM.
- Utilice condiciones en las políticas de IAM para restringir aún más el acceso: puede agregar una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puedes usar condiciones para conceder el acceso a las acciones del servicio si se utilizan a través de una acción específica Servicio de AWS, por ejemplo AWS CloudFormation. Para obtener más información, consulte Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.
- Utilice el analizador de acceso de IAM para validar las políticas de IAM con el fin de garantizar la seguridad y funcionalidad de los permisos: el analizador de acceso de IAM valida políticas nuevas y existentes para que respeten el lenguaje (JSON) de las políticas de IAM y las prácticas recomendadas de IAM. El analizador de acceso de IAM proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. Para más información, consulte Política de validación de Analizador de acceso de IAM en la Guía de usuario de IAM.
- Requerir autenticación multifactor (MFA): si tiene un escenario que requiere usuarios de IAM o un usuario raíz en Cuenta de AWS su cuenta, active la MFA para mayor seguridad. Para solicitar la MFA cuando se invocan las operaciones de la API, agregue las condiciones de la MFA a sus

políticas. Para más información, consulte Configuración del acceso a una API protegido por MFA en la Guía de usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte las Prácticas recomendadas de seguridad en IAM en la Guía del usuario de IAM.

Uso de la consola de Amazon Keyspaces

Amazon Keyspaces no requiere permisos específicos para acceder a la consola de Amazon Keyspaces. Necesita al menos permisos de solo lectura para enumerar y ver detalles sobre los recursos de Amazon Keyspaces que tiene en su sitio. Cuenta de AWS Si crea una política basada en identidad que sea más restrictiva que el mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades (usuarios o roles de IAM) que tengan esa política.

Hay dos políticas AWS administradas disponibles para que las entidades puedan acceder a la consola de Amazon Keyspaces.

- AmazonKeyspacesReadOnlyAccess_v2: esta política otorga acceso de solo lectura a Amazon Keyspaces.
- AmazonKeyspacesFullAccess— Esta política otorga permisos para usar Amazon Keyspaces con acceso total a todas las funciones.

Para obtener más información sobre las políticas administradas de Amazon Keyspaces, consulte the section called “Políticas administradas por AWS”.

Cómo permitir a los usuarios consultar sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas administradas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para completar esta acción en la consola o mediante programación mediante la AWS CLI API o. 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": "*"
        }
    ]
}
```

Acceso a las tablas de Amazon Keyspaces

El siguiente es un ejemplo de política que concede acceso de solo lectura (SELECT) a las tablas del sistema Amazon Keyspaces. Para todas las muestras, sustituya la región y el identificador de cuenta en el Amazon Resource Name (ARN) por los suyos propios.

> **Note**
>
> Para conectarse con un controlador estándar, un usuario debe tener al menos acceso SELECT a las tablas del sistema, dado que la mayoría de los controladores leen los espacios de claves/tablas del sistema en la conexión.

```
{
   "Version":"2012-10-17",
   "Statement":[
```

```
      {
         "Effect":"Allow",
         "Action":[
            "cassandra:Select"
         ],
         "Resource":[
            "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
         ]
      }
   ]
}
```

El siguiente ejemplo de política añade el acceso de solo lectura a la tabla de usuarios del espacio de clavesmytable. 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*"
         ]
      }
   ]
}
```

El siguiente ejemplo de política asigna acceso de lectura/escritura a una tabla de usuario y acceso de lectura a las tablas del sistema.

> **Note**
>
> Las tablas del sistema son siempre de solo lectura.

```
{
   "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*"
         ]
      }
   ]
}
```

El siguiente ejemplo de política permite a un usuario crear tablas en el espacio de claves `mykeyspace`.

```
{
   "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*"
         ]
      }
   ]
}
```

Acceso a recursos de Amazon Keyspaces basado en etiquetas

Puede utilizar condiciones en su política basada en identidades para controlar el acceso a los recursos de Amazon Keyspaces en función de etiquetas. Estas políticas controlan la visibilidad de los espacios de claves y las tablas de la cuenta. Tenga en cuenta que los permisos basados en etiquetas para las tablas del sistema se comportan de manera diferente cuando las solicitudes se

realizan mediante el AWS SDK en comparación con las llamadas a la API de Cassandra Query Language (CQL) mediante los controladores y las herramientas de desarrollo de Cassandra.

- Para realizar solicitudes de recursos `List` y `Get` con el SDK de AWS al utilizar el acceso basado en etiquetas, la persona que realiza la llamada necesita tener acceso de lectura a las tablas del sistema. Por ejemplo, se requieren permisos de acción `Select` para leer datos de las tablas del sistema a través de la operación `GetTable`. Si la persona que llama solo tiene acceso basado en etiquetas a una tabla específica, una operación que requiera acceso adicional a una tabla del sistema fallará.
- Por compatibilidad con el comportamiento establecido de los controladores de Cassandra, las políticas de autorización basadas en etiquetas no se aplican cuando se realizan operaciones en tablas del sistema mediante llamadas a la API de Cassandra Query Language (CQL) a través de los controladores y las herramientas para desarrolladores de Cassandra.

En el siguiente ejemplo se muestra cómo puede crear una política que conceda permisos a un usuario para ver una tabla si el `Owner` de la tabla contiene el valor del nombre de usuario de ese usuario. En este ejemplo, también concede acceso de lectura a las tablas del sistema.

```
{
   "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}"
            }
         }
      }
   ]
}
```

También puede asociar esta política al usuario de IAM en su cuenta. Si un usuario llamado `richard-roe` intenta ver una tabla de Amazon Keyspaces, la tabla debe tener la etiqueta `Owner=richard-roe` o `owner=richard-roe`. De lo contrario, se le deniega el acceso. La clave de la etiqueta de condición `Owner` coincide con los nombres de las claves de condición `Owner` y `owner` porque no distinguen entre mayúsculas y minúsculas. Para obtener más información, consulte Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.

La siguiente política concede permisos a un usuario para crear tablas con etiquetas si el `Owner` de la tabla contiene el valor del nombre de usuario de ese usuario.

```
{
    "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}"
            }
         }
      }
   ]
}
```

Políticas administradas de AWS para Amazon Keyspaces

Una política administrada de AWS es una política independiente que AWS crea y administra. Las políticas administradas de AWS se diseñan para ofrecer permisos para muchos casos de uso comunes, por lo que puede empezar a asignar permisos a los usuarios, grupos y roles.

Tenga presente que es posible que las políticas administradas de AWS no concedan permisos de privilegio mínimo para los casos de uso concretos, ya que están disponibles para que las utilicen

todos los clientes de AWS. Se recomienda definir políticas administradas por el cliente para los casos de uso a fin de reducir aún más los permisos.

No puede cambiar los permisos definidos en las políticas administradas de AWS. Si AWS actualiza los permisos definidos en un política administrada de AWS, la actualización afecta a todas las identidades de entidades principales (usuarios, grupos y roles) a las que está adjunta la política. Lo más probable es que AWS actualice una política administrada de AWS cuando se lance un nuevo Servicio de AWS o las operaciones de la API nuevas estén disponibles para los servicios existentes.

Para obtener más información, consulte Políticas administradas de AWS en la Guía del usuario de IAM.

Política administrada de AWS: AmazonKeyspacesReadOnlyAccess_v2

Puede adjuntar la política `AmazonKeyspacesReadOnlyAccess_v2` a las identidades de IAM.

Esta política concede acceso de solo lectura a Amazon Keyspaces e incluye los permisos necesarios al conectarse a través de puntos de conexión de VPC privados.

Detalles sobre los permisos

Esta política incluye los siguientes permisos.

- `Amazon Keyspaces`: proporciona acceso de solo lectura a Amazon Keyspaces.
- `Application Auto Scaling`: permite a las entidades principales ver las configuraciones de Application Auto Scaling. Esto es necesario para que los usuarios puedan ver las políticas de escalado automático que se vinculan a una tabla.
- `CloudWatch`: permite a las entidades principales ver los datos métricos y las alarmas configuradas en CloudWatch. Esto es necesario para que los usuarios puedan ver el tamaño facturable de la tabla y las alarmas de CloudWatch que se han configurado para una tabla.

- `AWS KMS`: permite a las entidades principales ver las claves configuradas en AWS KMS. Esto es necesario para que los usuarios puedan ver las claves de AWS KMS que crean y administran en su cuenta para confirmar que la clave asignada a Amazon Keyspaces sea una clave de cifrado simétrica habilitada.
- `Amazon EC2`: permite a las entidades principales que se conectan a Amazon Keyspaces a través de puntos de conexión de VPC consultar la VPC de su instancia de Amazon EC2 para obtener información sobre el punto de conexión y la interfaz en red. Este acceso de solo lectura a la instancia de Amazon EC2 es necesario para que Amazon Keyspaces pueda buscar y almacenar los puntos de conexión de VPC de interfaz disponibles en la tabla `system.peers` que se utiliza para el equilibrio de carga de conexiones.

Para revisar la política en formato `JSON`, consulte AmazonKeyspacesReadOnlyAccess_v2.

Política administrada de AWS: AmazonKeyspacesReadOnlyAccess

Puede adjuntar la política `AmazonKeyspacesReadOnlyAccess` a las identidades de IAM.

Esta política concede acceso de solo lectura a Amazon Keyspaces.

Detalles sobre los permisos

Esta política incluye los siguientes permisos.

- `Amazon Keyspaces`: proporciona acceso de solo lectura a Amazon Keyspaces.
- `Application Auto Scaling`: permite a las entidades principales ver las configuraciones de Application Auto Scaling. Esto es necesario para que los usuarios puedan ver las políticas de escalado automático que se vinculan a una tabla.
- `CloudWatch`: permite a las entidades principales ver los datos métricos y las alarmas configuradas en CloudWatch. Esto es necesario para que los usuarios puedan ver el tamaño facturable de la tabla y las alarmas de CloudWatch que se han configurado para una tabla.
- `AWS KMS`: permite a las entidades principales ver las claves configuradas en AWS KMS. Esto es necesario para que los usuarios puedan ver las claves de AWS KMS que crean y administran en

su cuenta para confirmar que la clave asignada a Amazon Keyspaces sea una clave de cifrado simétrica habilitada.

Para revisar la política en formato `JSON`, consulte AmazonKeyspacesReadOnlyAccess.

Política administrada de AWS: AmazonKeyspacesFullAccess

Puede adjuntar la política `AmazonKeyspacesFullAccess` a las identidades de IAM.

Esta política concede permisos administrativos que permiten a sus administradores acceder sin restricciones a Amazon Keyspaces.

Detalles sobre los permisos

Esta política incluye los siguientes permisos.

- `Amazon Keyspaces`: permite a las entidades principales acceder a cualquier recurso de Amazon Keyspaces y realizar todas las acciones.
- `Application Auto Scaling`: permite a las entidades principales crear, ver y eliminar políticas de escalado automático para las tablas de Amazon Keyspaces. Esto es necesario para que los administradores puedan administrar las políticas de escalado automático de las tablas de Amazon Keyspaces.
- `CloudWatch`: permite a las entidades principales ver el tamaño de la tabla facturable, así como crear, ver y eliminar alarmas de CloudWatch para las políticas de escalado automático de Amazon Keyspaces. Esto es necesario para que los administradores puedan ver el tamaño facturable de la tabla y crear un panel de control de CloudWatch.
- `IAM`: permite a Amazon Keyspaces crear automáticamente roles vinculados a servicios con IAM cuando están activadas las siguientes características:
  - `Application Auto Scaling`: cuando un administrador habilita Application Auto Scaling para una tabla, Amazon Keyspaces crea un rol vinculado a servicios para realizar acciones de escalado automático en su nombre.
  - `Amazon Keyspaces Multi-Region Replication`: cuando un administrador crea un espacio de claves multirregión, se crea automáticamente un rol vinculado a servicios para realizar la réplica de datos a las Regiones de AWS seleccionadas en su nombre.

Para obtener más información acerca de los roles vinculados a servicios, consulte the section called “Uso de roles vinculados a servicios”.

- `AWS KMS`: permite a las entidades principales ver las claves configuradas en AWS KMS. Esto es necesario para que los usuarios puedan ver las claves de AWS KMS que crean y administran en su cuenta para confirmar que la clave asignada a Amazon Keyspaces sea una clave de cifrado simétrica habilitada.
- `Amazon EC2`: permite a las entidades principales que se conectan a Amazon Keyspaces a través de puntos de conexión de VPC consultar la VPC de su instancia de Amazon EC2 para obtener información sobre el punto de conexión y la interfaz en red. Este acceso de solo lectura a la instancia de Amazon EC2 es necesario para que Amazon Keyspaces pueda buscar y almacenar los puntos de conexión de VPC de interfaz disponibles en la tabla `system.peers` que se utiliza para el equilibrio de carga de conexiones.

Para revisar la política en formato `JSON`, consulte AmazonKeyspacesFullAccess.

Actualizaciones de Amazon Keyspaces a las políticas administradas de AWS.

Consulte los detalles de las actualizaciones de las políticas administradas de AWS para Amazon Keyspaces desde que este servicio comenzó a realizar el seguimiento de estos cambios. Para obtener alertas automáticas sobre cambios en esta página, suscríbase a la fuente RSS en la página de Historial de documentos.

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess: actualización de una política existente	Amazon Keyspaces ha añadido nuevos permisos de solo lectura para que los clientes que se conecten a Amazon Keyspaces a través de puntos de conexión de VPC de interfaz puedan acceder a la instancia de	3 de octubre de 2023

Cambio	Descripción	Fecha
	Amazon EC2 para consultar información de red. Amazon Keyspaces almacena los puntos de conexión de VPC de interfaz disponibles en la tabla de `system.peers` para equilibrar la carga de conexión. Para obtener más información, consulte the section called “Uso de los puntos de conexión de VPC de interfaz”.	

Cambio	Descripción	Fecha
AmazonKeyspacesReadOnlyAccess_v2: nueva política	Amazon Keyspaces ha creado una nueva política para añadir permisos de solo lectura para que los clientes que se conecten a Amazon Keyspaces a través de puntos de conexión de VPC de interfaz puedan acceder a la instancia de Amazon EC2 para consultar información de red. Amazon Keyspaces almacena los puntos de conexión de VPC de interfaz disponibles en la tabla de `system.peers` para equilibrar la carga de conexión. Para obtener más información, consulte the section called “Uso de los puntos de conexión de VPC de interfaz”.	12 de septiembre de 2023

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess: actualización de una política existente	Amazon Keyspaces ha añadido nuevos permisos para permitir a Amazon Keyspaces crear un rol vinculado a servicios cuando un administrador crea un espacio de claves multirregión. Amazon Keyspaces utiliza el rol vinculado a servicios para realizar tareas de réplica de datos en su nombre. Para obtener más información, consulte the section called “Réplica multirregión”.	5 de junio de 2023
AmazonKeyspacesReadOnlyAccess: actualización de una política existente	Amazon Keyspaces ha añadido nuevos permisos para permitir a los usuarios ver el tamaño facturable de una tabla mediante CloudWatch. Amazon Keyspaces se integra con Amazon CloudWatch para permitirle monitorear el tamaño facturable de la tabla. Para obtener más información, consulte the section called “Métricas y dimensiones de Amazon Keyspaces”.	7 de julio de 2022

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess: actualización de una política existente	Amazon Keyspaces ha añadido nuevos permisos para permitir a los usuarios ver el tamaño facturable de una tabla mediante CloudWatch. Amazon Keyspaces se integra con Amazon CloudWatch para permitirle monitorear el tamaño facturable de la tabla. Para obtener más informaci ón, consulte the section called “Métricas y dimensiones de Amazon Keyspaces”.	7 de julio de 2022
AmazonKeyspacesReadOnlyAccess: actualización de una política existente	Amazon Keyspaces añadió nuevos permisos para permitir a los usuarios ver las claves de AWS KMS que se han configurado para el cifrado en reposo de Amazon Keyspaces . El cifrado en reposo de Amazon Keyspaces se integra con AWS KMS para proteger y administrar las claves de cifrado utilizadas para cifrar datos en reposo. Para ver la clave de AWS KMS configura da para Amazon Keyspaces, se han añadido permisos de solo lectura.	1 de junio de 2021

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess: actualización de una política existente	Amazon Keyspaces añadió nuevos permisos para permitir a los usuarios ver las claves de AWS KMS que se han configurado para el cifrado en reposo de Amazon Keyspaces . El cifrado en reposo de Amazon Keyspaces se integra con AWS KMS para proteger y administrar las claves de cifrado utilizadas para cifrar datos en reposo. Para ver la clave de AWS KMS configura da para Amazon Keyspaces, se han añadido permisos de solo lectura.	1 de junio de 2021
Amazon Keyspaces ha comenzado a hacer el seguimiento de los cambios	Amazon Keyspaces ha comenzado a hacer el seguimiento de los cambios en sus políticas administradas de AWS.	1 de junio de 2021

Solución de problemas de identidad y acceso a Amazon Keyspaces

Utilice la siguiente información como ayuda para diagnosticar y solucionar problemas comunes que pueda encontrar al trabajar con Amazon Keyspaces e IAM.

Temas

- No tengo autorización para realizar una acción en Amazon Keyspaces
- He modificado un usuario o rol de IAM y los cambios no han surtido efecto de inmediato.
- No puedo restaurar una tabla mediante la point-in-time recuperación de Amazon Keyspaces (PITR)

- No estoy autorizado a realizar iam: PassRole
- Soy administrador y quiero permitir que otros accedan a Amazon Keyspaces
- Quiero permitir que personas ajenas a mí accedan Cuenta de AWS a mis recursos de Amazon Keyspaces

No tengo autorización para realizar una acción en Amazon Keyspaces

Si AWS Management Console te indica que no estás autorizado a realizar una acción, debes ponerte en contacto con tu administrador para que te ayude. Su administrador es la persona que le facilitó su nombre de usuario y contraseña.

En el siguiente ejemplo se produce un error cuando el usuario de IAM `mateojackson` intenta utilizar la consola para ver los detalles de una *`tabla`* pero no tiene permisos `cassandra:`*`Select`* para la tabla.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cassandra:Select on resource: mytable
```

En este caso, Mateo pide a su administrador que actualice sus políticas de forma que pueda obtener acceso al recurso *`mytable`* mediante la acción `cassandra:`*`Select`*.

He modificado un usuario o rol de IAM y los cambios no han surtido efecto de inmediato.

Los cambios en la política de IAM pueden tardar hasta 10 minutos en surtir efecto para las aplicaciones con conexiones existentes y establecidas a Amazon Keyspaces. Los cambios en la política de IAM surten efecto de inmediato cuando las aplicaciones establecen una nueva conexión. Si ha realizado modificaciones en un usuario o rol de IAM existente y no ha surtido efecto de inmediato, espere 10 minutos o desconéctese de Amazon Keyspaces y vuelva a conectarse.

No puedo restaurar una tabla mediante la point-in-time recuperación de Amazon Keyspaces (PITR)

Si está intentando restaurar una tabla de Amazon Keyspaces con point-in-time recuperación (PITR) y observa que el proceso de restauración comienza, pero no se completa correctamente, es posible que no haya configurado todos los permisos necesarios para el proceso de restauración. Debe ponerse en contacto con su administrador para obtener ayuda y pedirle que actualice sus políticas a fin de permitirle restaurar una tabla en Amazon Keyspaces.

Además de los permisos de usuario, Amazon Keyspaces podría requerir permisos para realizar acciones durante el proceso de restauración en nombre de su entidad principal. Este es el caso si la tabla está cifrada con una clave administrada por el cliente, o si está utilizando políticas de IAM que restrinjan el tráfico entrante. Por ejemplo, si utiliza claves de condición en su política de IAM para restringir el tráfico de origen a puntos de conexión o rangos de IP específicos, la operación de restauración falla. Para permitir que Amazon Keyspaces realice la operación de restauración de tablas en nombre de su entidad principal, debe añadir una clave de condición global `aws:ViaAWSService` en la política de IAM.

Para obtener más información sobre los permisos para restaurar tablas, consulte the section called “Permisos de restauración”.

No estoy autorizado a realizar iam: PassRole

Si recibe un error que indica que no está autorizado a realizar la acción `iam:PassRole`, se deben actualizar sus políticas para permitirle pasar un rol a Amazon Keyspaces.

Algunas Servicios de AWS permiten transferir una función existente a ese servicio en lugar de crear una nueva función de servicio o una función vinculada a un servicio. Para ello, debe tener permisos para transferir el rol al servicio.

El siguiente ejemplo de error se produce cuando un usuario de IAM llamado `marymajor` intenta utilizar la consola para realizar una acción en Amazon Keyspaces. Sin embargo, la acción requiere que el servicio cuente con permisos que otorguen un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su administrador. AWS El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

Soy administrador y quiero permitir que otros accedan a Amazon Keyspaces

Para permitir que otros accedan a Amazon Keyspaces, debe crear una entidad de IAM (usuario o rol) para la persona o aplicación que necesita acceso. Esta persona utilizará las credenciales de la

entidad para acceder a AWS. A continuación, debe adjuntar una política a la entidad que les conceda los permisos correctos en Amazon Keyspaces.

Para comenzar de inmediato, consulte Creación del primer grupo y usuario delegado de IAM en la Guía del usuario de IAM.

Quiero permitir que personas ajenas a mí accedan Cuenta de AWS a mis recursos de Amazon Keyspaces

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admitan las políticas basadas en recursos o las listas de control de acceso (ACL), puede utilizar dichas políticas para conceder a las personas acceso a sus recursos.

Para más información, consulte lo siguiente:

- Para informarse de si Amazon Keyspaces admite estas características, consulte Funcionamiento de Amazon Keyspaces con IAM.
- Para obtener información sobre cómo proporcionar acceso a los recursos de su Cuentas de AWS propiedad, consulte Proporcionar acceso a un usuario de IAM en otro Cuenta de AWS de su propiedad en la Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso a tus recursos a terceros Cuentas de AWS, consulta Cómo proporcionar acceso a recursos que Cuentas de AWS son propiedad de terceros en la Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso mediante la federación de identidades, consulte Proporcionar acceso a usuarios autenticados externamente (federación de identidades) en la Guía del usuario de IAM.
- Para obtener información sobre la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulte Cómo los roles de IAM difieren de las políticas basadas en recursos en la Guía del usuario de IAM.

Uso de roles vinculados a servicios para Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM vinculado directamente a Amazon Keyspaces. Los roles vinculados a servicios están predefinidos por Amazon

Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Temas

- Uso de roles para el escalado automático de aplicaciones de Amazon Keyspaces
- Uso de roles para la réplica multirregión de Amazon Keyspaces

Uso de roles para el escalado automático de aplicaciones de Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM vinculado directamente a Amazon Keyspaces. Los roles vinculados a servicios están predefinidos por Amazon Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios facilita la configuración de Amazon Keyspaces dado que no tiene que añadir manualmente los permisos necesarios. Amazon Keyspaces define los permisos de sus roles vinculados a servicios y, a menos que se defina de manera diversa, solo Amazon Keyspaces puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo puede eliminar un rol vinculado a servicios después de eliminar sus recursos relacionados. Esto protege sus recursos de Amazon Keyspaces porque no puede eliminar inadvertidamente el permiso para acceder a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Sí en la columna Roles vinculados a servicios. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon Keyspaces

Amazon Keyspaces utiliza el rol vinculado al servicio denominado AWSServiceRoleForApplicationAutoScaling_CassandraTablepara permitir que Application Auto Scaling llame a Amazon Keyspaces y a Amazon en su nombre. CloudWatch

El rol AWSServiceRoleForApplicationAutoScaling_CassandraTable vinculado al servicio confía en los siguientes servicios para que lo asuman:

- `cassandra.application-autoscaling.amazonaws.com`

La política de permisos del rol permite a Application Auto Scaling realizar las siguientes acciones en los recursos de Amazon Keyspaces especificados:

- Acción: `cassandra:Select` en `arn:*:cassandra:*:*:/keyspace/system/table/*`
- Acción: `cassandra:Select` en el recurso `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
- Acción: `cassandra:Select` en el recurso `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Acción: `cassandra:Alter` en el recurso `arn:*:cassandra:*:*:"*"`

Creación de un rol vinculado a servicios para Amazon Keyspaces

No es necesario crear manualmente un rol vinculado a servicios para el escalado automático de Amazon Keyspaces. Cuando habilita el escalado automático de Amazon Keyspaces en una tabla con el CQLAWS Management Console, la o la AWS APIAWS CLI, Application Auto Scaling crea automáticamente el rol vinculado al servicio.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Cuando habilita el escalado automático de Amazon Keyspaces para una tabla, Application Auto Scaling vuelve a crear el rol vinculado al servicio para usted.

> **Important**
>
> Este rol vinculado a servicios puede aparecer en su cuenta si se ha completado una acción en otro servicio que utilice las características compatibles con este rol. Para obtener más información, consulte Un nuevo rol ha aparecido en mi Cuenta de AWS.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al habilitar el escalado automático de aplicaciones de Amazon Keyspaces para una tabla, Application Auto Scaling vuelve a crear el rol vinculado a servicios por usted.

Edición de un rol vinculado a servicios para Amazon Keyspaces

Amazon Keyspaces no le permite editar el rol vinculado al AWSServiceRoleForApplicationAutoScaling_CassandraTable servicio. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editar un rol vinculado a servicios en la Guía del usuario de IAM.

Eliminación de un rol vinculado a servicios para Amazon Keyspaces

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma no conservará una entidad no utilizada que no se monitorice ni se mantenga de forma activa. Sin embargo, primero debe deshabilitar el escalado automático en todas las tablas de la cuenta en todas las Regiones de AWS para poder eliminar el rol vinculado a servicios manualmente. Para deshabilitar el escalado automático en las tablas de Amazon Keyspaces, consulte Modificación o deshabilitación de la configuración de escalado automático de Amazon Keyspaces.

> **Note**
>
> Si el escalado automático de Amazon Keyspaces utiliza el rol en el momento de intentar modificar los recursos, es posible que la anulación del registro falle. En tal caso, espere unos minutos e intente de nuevo la operación.

Cómo eliminar manualmente el rol vinculado a servicios mediante IAM

Utilice la consola de IAMAWS CLI, la o la AWS API para eliminar la función vinculada al servicio. AWSServiceRoleForApplicationAutoScaling_CassandraTable Para obtener más información, consulte Eliminar un rol vinculado a un servicio en la Guía del usuario de IAM.

> **Note**
>
> Para eliminar el rol vinculado a servicios utilizado por el escalado automático de Amazon Keyspaces, primero debe deshabilitar el escalado automático en todas las tablas de la cuenta.

Regiones admitidas para roles vinculados a servicios de Amazon Keyspaces

Amazon Keyspaces admite el uso de roles vinculados a servicios en todas las regiones en las que el servicio esté disponible. Para obtener más información, consulte Puntos de conexión del servicio para Amazon Keyspaces.

Uso de roles para la réplica multirregión de Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM vinculado directamente a Amazon Keyspaces. Los roles vinculados a servicios están predefinidos por Amazon Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios facilita la configuración de Amazon Keyspaces dado que no tiene que añadir manualmente los permisos necesarios. Amazon Keyspaces define los permisos de sus roles vinculados a servicios y, a menos que se defina de manera diversa, solo Amazon Keyspaces puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo puede eliminar un rol vinculado a servicios después de eliminar sus recursos relacionados. Esto protege sus recursos de Amazon Keyspaces porque no puede eliminar inadvertidamente el permiso para acceder a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Sí en la columna Roles vinculados a servicios. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon Keyspaces

Amazon Keyspaces utiliza el rol vinculado al servicio denominado para permitir que AWSServiceRoleForAmazonKeyspacesReplicationAmazon Keyspaces replique las escrituras en todas las réplicas de una tabla multirregional en su nombre.

El rol AWSServiceRoleForAmazonKeyspacesReplication vinculado al servicio confía en que los siguientes servicios asuman el rol:

- `replication.cassandra.amazonaws.com`

La política de permisos de roles denominada KeyspacesReplicationServiceRolePolicy permite a Amazon Keyspaces realizar las siguientes acciones:

- Acción: `cassandra:Select`
- Acción: `cassandra:SelectMultiRegionResource`
- Acción: `cassandra:Modify`
- Acción: `cassandra:ModifyMultiRegionResource`

Si bien el rol vinculado al servicio Amazon Keyspaces AWSServiceRoleForAmazonKeyspacesReplication proporciona los permisos: «Acción:» para el nombre de recurso de Amazon (ARN) especificado «arn: *» en la política, Amazon Keyspaces proporciona el ARN de su cuenta.

Debe configurar los permisos para permitir a sus usuarios, grupos o funciones, crear, editar o eliminar la descripción de un rol vinculado al servicio. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Creación de un rol vinculado a servicios para Amazon Keyspaces

No puede crear manualmente un rol vinculado a servicios. Al crear un espacio de claves multirregión en la AWS Management Console, la AWS CLI o la API de AWS, Amazon Keyspaces crea el rol vinculado a servicios por usted.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al crear un espacio de claves multirregión, Amazon Keyspaces vuelve a crear el rol vinculado a servicios para usted.

Edición de un rol vinculado a servicios para Amazon Keyspaces

Amazon Keyspaces no le permite editar el rol vinculado al AWSServiceRoleForAmazonKeyspacesReplication servicio. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editar un rol vinculado a servicios en la Guía del usuario de IAM.

Eliminación de un rol vinculado a servicios para Amazon Keyspaces

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma no tiene una entidad no utilizada que no se

monitoree ni mantenga de forma activa. Sin embargo, primero debe eliminar todos los espacios de claves multirregión de la cuenta en todas las Regiones de AWS para poder eliminar manualmente el rol vinculado a servicios.

Saneamiento de un rol vinculado a servicios

Para poder utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los espacios de claves y tablas multirregión utilizados por el rol.

> **Note**
>
> Si el servicio de Amazon Keyspaces utiliza el rol en el momento en que intenta eliminar los recursos, es posible que la eliminación falle. En tal caso, espere unos minutos e intente de nuevo la operación.

Para eliminar los recursos de Amazon Keyspaces utilizados por la AWSServiceRoleForAmazonKeyspacesReplication (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon Keyspaces en https://console.aws.amazon.com/keyspaces/home.
2. Elija Espacios de claves en el panel izquierdo.
3. Seleccione todos los espacios de claves multirregión de la lista.
4. Elija Eliminar, confirme la eliminación y luego elija Eliminar espacios de claves.

También puede eliminar espacios de claves multirregión mediante programación utilizando cualquiera de los siguientes métodos.

- La instrucción ??? de Cassandra Query Language (CQL).
- La operación delete-keyspace de la CLI de AWS.
- El DeleteKeyspacefuncionamiento de la API de Amazon Keyspaces.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la AWS CLI o la API de AWS para eliminar el rol vinculado a servicios de AWSServiceRoleForAmazonKeyspacesReplication. Para obtener más información, consulte Eliminación de un rol vinculado a servicios en la Guía del usuario de IAM.

Regiones admitidas para roles vinculados a servicios de Amazon Keyspaces

Amazon Keyspaces no admite el uso de roles vinculados a servicios en todas las regiones en las que el servicio esté disponible. Puede utilizar el AWSServiceRoleForAmazonKeyspacesReplication rol en las siguientes regiones.

Nombre de la región	Identidad de la región	Admitida en Amazon Keyspaces
Este de EE. UU. (Norte de Virginia)	us-east-1	Sí
Este de EE. UU. (Ohio)	us-east-2	Sí
Oeste de EE. UU. (Norte de California)	us-west-1	Sí
Oeste de EE. UU. (Oregón)	us-west-2	Sí
Asia Pacífico (Mumbai)	ap-south-1	Sí
Asia Pacífico (Osaka)	ap-northeast-3	Sí
Asia-Pacífico (Seúl)	ap-northeast-2	Sí
Asia Pacífico (Singapur)	ap-southeast-1	Sí
Asia Pacífico (Sídney)	ap-southeast-2	Sí
Asia Pacífico (Tokio)	ap-northeast-1	Sí
Canadá (centro)	ca-central-1	Sí
Europa (Fráncfort)	eu-central-1	Sí
Europa (Irlanda)	eu-west-1	Sí
Europa (Londres)	eu-west-2	Sí
Europa (París)	eu-west-3	Sí
América del Sur (São Paulo)	sa-east-1	Sí
AWS GovCloud (Este de EE. UU.)	us-gov-east-1	No

Nombre de la región	Identidad de la región	Admitida en Amazon Keyspaces
AWS GovCloud (EE. UU.-Oeste)	us-gov-west-1	No

Validación de conformidad para Amazon Keyspaces (para Apache Cassandra)

Auditores externos evalúan la seguridad y conformidad de Amazon Keyspaces (para Apache Cassandra) como parte de múltiples programas de conformidad de AWS. Entre ellos se incluyen:

- ISO/IEC 27001:2013, 27017:2015, 27018:2019 e ISO/IEC 9001:2015. Para obtener más información, consulte Certificaciones y servicios ISO y CSA STAR de AWS.
- Controles del Sistema y Organizaciones (System and Organization Controls, SOC)
- Industria de tarjetas de pago (PCI)
- Programa Federal de Administración de Riesgos y Autorizaciones-Alto (FedRAMP-High)
- Ley de Portabilidad y Responsabilidad de Seguros Médicos de EE. UU (Health Insurance Portability and Accountability Act, HIPAA).

Para saber si un servicio de AWS está incluido en el ámbito de programas de conformidad específicos, consulte Servicios de AWS en el ámbito del programa de conformidad y escoja el programa de conformidad que le interese. Para obtener información general, consulte Programas de conformidad de AWS.

Puede descargar los informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact.

Su responsabilidad de conformidad al utilizar servicios de AWS se determina en función de la sensibilidad de los datos, los objetivos de cumplimiento de su empresa y la legislación y los reglamentos correspondientes. AWS proporciona los siguientes recursos para ayudar con la conformidad:

- Guías de inicio rápido de seguridad y conformidad: estas guías de implementación tratan consideraciones sobre arquitectura y ofrecen pasos para implementar los entornos de referencia centrados en la seguridad y la conformidad en AWS.

- Arquitectura para la seguridad y el cumplimiento de la HIPAA en Amazon Web Services: en este documento técnico, se describe cómo las empresas pueden utilizar AWS para crear aplicaciones aptas para HIPAA.

  > **Note**
  >
  > No todos los Servicios de AWS son aptos para HIPAA. Para obtener más información, consulte la Referencia de servicios aptos para HIPAA.

- Recursos de conformidad de AWS: este conjunto de manuales y guías podría aplicarse a su sector y ubicación.
- Guías de cumplimiento para clientes de AWS: comprenda el modelo de responsabilidad compartida desde el punto de vista del cumplimiento. Las guías resumen las mejores prácticas para garantizar la seguridad de los Servicios de AWS y orientan los controles de seguridad en varios marcos (incluidos el Instituto Nacional de Estándares y Tecnología (NIST, por sus siglas en inglés), el Consejo de Estándares de Seguridad de la Industria de Tarjetas de Pago (PCI, por sus siglas en inglés) y la Organización Internacional de Normalización (ISO, por sus siglas en inglés)).
- Evaluación de recursos con reglas en la Guía para desarrolladores de AWS Config: el servicio AWS Config evalúa en qué medida las configuraciones de sus recursos cumplen las prácticas internas, las directrices del sector y las normativas.
- AWS Security Hub: este servicio de AWS proporciona una visión completa de su estado de seguridad en AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del sector de la seguridad. Para obtener una lista de los servicios y controles compatibles, consulte la Referencia de controles de Security Hub.
- AWS Audit Manager: este Servicio de AWS le ayuda a auditar continuamente el uso de AWS con el fin de simplificar la forma en que administra el riesgo y la conformidad con las normativas y los estándares del sector.

Resiliencia y recuperación de desastres en Amazon Keyspaces

La infraestructura global de AWS se divide en Regiones de AWS y zonas de disponibilidad. Las Regiones de AWS proporcionan varias zonas de disponibilidad físicamente independientes y aisladas que se encuentran conectadas mediante redes con un alto nivel de rendimiento y redundancia, además de baja latencia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre zonas

de disponibilidad sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de centros de datos únicos o múltiples.

Amazon Keyspaces replica tres veces los datos de forma automática en múltiples zonas de disponibilidad de AWS dentro de la misma Región de AWS para conseguir mayor durabilidad y alta disponibilidad.

Para obtener más información sobre las Regiones de AWS y las zonas de disponibilidad, consulte Infraestructura global de AWS.

Además de la infraestructura global de AWS, Amazon Keyspaces ofrece varias características que le ayudan a admitir sus necesidades de resiliencia y copias de seguridad de datos.

Réplica multirregión

Amazon Keyspaces ofrece réplicas multirregión en el caso que necesitase replicar sus datos o aplicaciones en distancias geográficas mayores. Puede replicar sus tablas de Amazon Keyspaces en hasta seis Regiones de AWS diferentes de su elección. Para obtener más información, consulte Réplica multirregión.

Recuperación en un momento dado (PITR)

PITR ayuda a proteger sus tablas de Amazon Keyspaces de operaciones accidentales de escritura o eliminación proporcionándole copias de seguridad continuas de los datos de sus tablas. Para obtener más información, consulte Recuperación en un momento dado para Amazon Keyspaces.

Seguridad de la infraestructura en Amazon Keyspaces

Como se trata de un servicio administrado, Amazon Keyspaces (para Apache Cassandra) está protegido por la seguridad de red global de AWS. Para obtener información sobre los servicios de seguridad de AWS y cómo AWS protege la infraestructura, consulte Seguridad en la nube de AWS. Para diseñar su entorno de AWS con las prácticas recomendadas de seguridad de infraestructura, consulte Protección de la infraestructura en Portal de seguridad de AWS Well-Architected Framework.

Puede utilizar llamadas a la API publicadas de AWS para acceder a Amazon Keyspaces a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Conjuntos de cifrado con confidencialidad directa total (PFS) como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad principal de IAM. También puede utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Amazon Keyspaces admite dos métodos de autenticación de solicitudes de clientes. El primer método utiliza credenciales específicas del servicio, que son credenciales basadas en contraseña generadas para un usuario de IAM específico. Puede crear y administrar la contraseña mediante la consola de IAM, la API AWS CLI o la API AWS. Para obtener más información, consulte Uso de IAM con Amazon Keyspaces.

El segundo método utiliza un complemento de autenticación para el controlador Java DataStax de código abierto para Cassandra. Este complemento permite a los usuarios, roles e identidades federadas de IAM añadir información de autenticación a las solicitudes de la API de Amazon Keyspaces (para Apache Cassandra) mediante el proceso AWS Signature Version 4 (SigV4). Para obtener más información, consulte the section called “Credenciales de IAM para autenticación de AWS.”.

Puede utilizar un punto de conexión de VPC de interfaz para evitar que el tráfico entre su Amazon VPC y Amazon Keyspaces salga de la red de Amazon. Los puntos de conexión de VPC de interfaz funcionan con AWS PrivateLink, una tecnología de AWS que habilita la comunicación privada entre servicios de AWS mediante una interfaz de red elástica con IP privadas en su Amazon VPC. Para obtener más información, consulte the section called “Uso de los puntos de conexión de VPC de interfaz”.

Uso de Amazon Keyspaces con puntos de conexión de VPC de interfaz

Los puntos de conexión de VPC de interfaz habilitan la comunicación privada entre su nube privada virtual (VPC), que se ejecuta en Amazon VPC, y Amazon Keyspaces. Los puntos finales de la interfaz VPC funcionan con AWS PrivateLink, que es un AWS servicio que permite la comunicación privada entre las VPC y los servicios. AWS

AWS PrivateLink lo habilita mediante una interfaz de red elástica con direcciones IP privadas en la VPC para que el tráfico de red no salga de la red de Amazon. Los puntos de conexión de VPC de interfaz no requieren una puerta de enlace de Internet, un dispositivo NAT, una conexión de VPN ni una conexión de AWS Direct Connect . Para obtener más información, consulte Nube privada virtual de Amazon e Interfaz de puntos de conexión de VPC (AWS PrivateLink).

Temas

- Uso de puntos de conexión de VPC de interfaz para Amazon Keyspaces
- Relleno de entradas de la tabla system.peers con información del punto de conexión de VPC de interfaz
- Control de acceso a puntos de conexión de VPC de interfaz para Amazon Keyspaces
- Disponibilidad
- Políticas de puntos de conexión de VPC y recuperación de Amazon Keyspaces point-in-time (PITR)
- Errores y advertencias comunes

Uso de puntos de conexión de VPC de interfaz para Amazon Keyspaces

Puede crear un punto de conexión de VPC de interfaz para que el tráfico entre Amazon Keyspaces y sus recursos de Amazon VPC comience a fluir a través del punto de conexión de VPC de interfaz. Para comenzar, siga los pasos necesarios para crear un punto de conexión de interfaz. A continuación, edite el grupo de seguridad asociado al punto de conexión que creó en el paso anterior y configure una regla de entrada para el puerto 9142. Para obtener más información, consulte Adición, eliminación y actualización de reglas.

Para ver un step-by-step tutorial sobre cómo configurar una conexión a Amazon Keyspaces a través de un punto de enlace de VPC, consulte. the section called “Conexión con puntos finales de VPC” Para obtener información sobre cómo configurar el acceso multicuenta a los recursos de Amazon Keyspaces separados de las aplicaciones en Cuentas de AWS diferentes áreas de una VPC, consulte. the section called “Acceso entre cuentas”

Relleno de entradas de la tabla `system.peers` con información del punto de conexión de VPC de interfaz

Los controladores de Apache Cassandra utilizan la tabla `system.peers` para consultar información de los nodos sobre el clúster. Los controladores de Cassandra utilizan la información de los nodos

para equilibrar la carga de las conexiones y reintentar las operaciones. Amazon Keyspaces rellena nueve entradas en la tabla `system.peers` de forma automática para los clientes que se conecten a través del punto de conexión público.

Para proporcionar a los clientes que se conecten a través de puntos de conexión de VPC de interfaz una funcionalidad similar, Amazon Keyspaces rellena la tabla `system.peers` de su cuenta con una entrada para cada zona de disponibilidad en la que esté disponible un punto de conexión de VPC. Para buscar y almacenar los puntos de conexión de VPC de interfaz disponibles en la tabla `system.peers`, Amazon Keyspaces requiere que conceda a la entidad de IAM utilizada para conectarse a Amazon Keyspaces permisos de acceso para consultar su VPC en busca de información sobre el punto de conexión y la interfaz en red.

> **Important**
>
> Rellenar la tabla `system.peers` con sus puntos de conexión de VPC de interfaz disponibles mejora el equilibrio de la carga y aumenta el rendimiento de lectura/escritura. Se recomienda para todos los clientes que accedan a Amazon Keyspaces mediante puntos de conexión de VPC de interfaz y es necesario para Apache Spark.

Para conceder a la entidad de IAM utilizada para conectarse a Amazon Keyspaces permisos para buscar la información necesaria del punto de conexión de VPC de interfaz, puede actualizar la política de usuario o el rol de IAM existentes, o crear una nueva política de IAM como se muestra en el siguiente ejemplo.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListVPCEndpoints",
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeVpcEndpoints"
         ],
         "Resource":"*"
      }
   ]
}
```

> **Note**
>
> Las políticas administradas `AmazonKeyspacesReadOnlyAccess_v2` y `AmazonKeyspacesFullAccess` incluyen los permisos necesarios para permitir que Amazon Keyspaces acceda a la instancia de Amazon EC2 para leer la información sobre los puntos de conexión de VPC de interfaz disponibles.

Para confirmar que la política se haya configurado correctamente, consulte la tabla `system.peers` para ver la información de conexión en red. Si la tabla `system.peers` está vacía, podría indicar que la política no se ha configurado correctamente o que ha superado la cuota de tasa de solicitudes para las acciones `DescribeNetworkInterfaces` y `DescribeVPCEndpoints` de la API. `DescribeVPCEndpoints` entra en la categoría `Describe*` y se considera una acción no mutante. `DescribeNetworkInterfaces` entra en el subconjunto de acciones no mutantes, no filtradas y no paginadas, y se aplican cuotas diferentes. Para obtener más información, consulte Tasas de solicitud de llenado y tamaños de bucket de token en la Referencia de la API de Amazon EC2.

Si ve una tabla vacía, inténtelo de nuevo unos minutos más tarde para descartar problemas de cuota de tasa de solicitud. Para comprobar que ha configurado correctamente los puntos de conexión de VPC, consulte the section called “Errores de conexión de punto de conexión de VPC”. Si su consulta devuelve resultados de la tabla, su política se ha configurado correctamente.

Control de acceso a puntos de conexión de VPC de interfaz para Amazon Keyspaces

Con las políticas de punto de conexión de VPC, puede controlar el acceso a los recursos de dos maneras:

- Política de IAM: puede controlar las solicitudes, usuarios o grupos a los que se permite acceder a Amazon Keyspaces a través de un punto de conexión de VPC específico. Puede hacerlo mediante una clave de condición en la política que se vincula a un usuario, grupo o rol de IAM.
- Política de VPC: puede controlar qué puntos de conexión de VPC tienen acceso a sus recursos de Amazon Keyspaces mediante la vinculación de políticas a los mismos. Para restringir el acceso a un espacio de claves o tabla específicos y permitir solo el tráfico que llega a través de un punto de conexión de VPC específico, edite la política de IAM existente que restringe el acceso a los recursos y añada ese punto de conexión de VPC.

A continuación se muestran ejemplos de políticas de punto de conexión para acceder a los recursos de Amazon Keyspaces.

- Ejemplo de política de IAM: Restringir todo el acceso a una tabla específica de Amazon Keyspaces a menos que el tráfico proceda del punto de conexión de VPC especificado: esta política de ejemplo se puede vincular a un usuario, rol o grupo de IAM. Restringe el acceso a una tabla de Amazon Keyspaces especificada a menos que el tráfico entrante se origine en un punto de conexión de VPC especificado.

```
{
   "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**
>
> Para restringir el acceso a una tabla específica, debe incluir también el acceso a las tablas del sistema. Las tablas del sistema son de solo lectura.

- Ejemplo de política VPC: Acceso de solo lectura: esta política de ejemplo puede vincularse a un punto de conexión de VPC. (Para obtener más información, consulte Control de acceso a recursos de Amazon VPC). Restringe las acciones al acceso de solo lectura a los recursos de Amazon Keyspaces a través del punto de conexión de VPC al que esté vinculado.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "ReadOnly",
      "Principal": "*",
      "Action": [
        "cassandra:Select"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

- Ejemplo de política de VPC: Restringir el acceso a una tabla específica de Amazon Keyspaces: esta política de ejemplo puede vincularse a un punto de conexión de VPC. Limita el acceso a una tabla específica a través del punto de conexión de VPC al que esté vinculado.

```
{
   "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**
>
> Para restringir el acceso a una tabla específica, debe incluir también el acceso a las tablas del sistema. Las tablas del sistema son de solo lectura.

Disponibilidad

Amazon Keyspaces admite el uso de puntos de enlace de VPC de interfaz en todos los lugares en los que el servicio Regiones de AWS esté disponible. Para obtener más información, consulte ???.

Políticas de puntos de conexión de VPC y recuperación de Amazon Keyspaces point-in-time (PITR)

Si utiliza políticas de IAM con claves de condición para restringir el tráfico entrante, la operación de restauración de la tabla podría fallar. Por ejemplo, si restringe el tráfico de origen a puntos de conexión de VPC específicos mediante `aws:SourceVpce`claves de condición, la operación de restauración de la tabla falla. Para permitir que Amazon Keyspaces realice una operación de restauración en nombre de su entidad principal, debe añadir una clave de condición `aws:ViaAWSService` a su política de IAM. La clave de `aws:ViaAWSService` condición permite el acceso cuando cualquier AWS servicio realiza una solicitud con las credenciales del director. Para obtener más información, consulte Elementos JSON de la política de IAM: Clave de condición en la Guía del usuario de IAM. La siguiente política es un ejemplo de esto.

```
{
   "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"
            }
         }
      }
   ]
}
```

Errores y advertencias comunes

Si utiliza Amazon Virtual Private Cloud y se conecta a Amazon Keyspaces, es posible que vea la siguiente advertencia.

```
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.
```

Esta advertencia se produce porque la tabla `system.peers` contiene entradas para todos los puntos de conexión de VPC de Amazon para los que Amazon Keyspaces tiene permisos de visualización, incluyendo el punto de conexión de VPC de Amazon a través del cual está conectado. Puede ignorar sin problemas esta advertencia.

Para otros errores, consulte the section called “Errores de conexión de punto de conexión de VPC”.

Configuración y análisis de vulnerabilidades para Amazon Keyspaces

AWS gestiona las tareas de seguridad básicas, como la aplicación de parches en la base de datos y el sistema operativo (SO) de invitado, la configuración del firewall y la recuperación de desastres. Estos procedimientos han sido revisados y certificados por los terceros pertinentes. Para obtener más detalles, consulte los siguientes recursos de :

- Modelo de responsabilidad compartida
- Amazon Web Services: información general de procesos de seguridad (documento técnico)

Prácticas recomendadas de seguridad para Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) proporciona una serie de características de seguridad que debe tener en cuenta a la hora de desarrollar e implementar sus propias políticas de seguridad. Las siguientes prácticas recomendadas son directrices generales y no suponen una solución de seguridad completa. Puesto que es posible que estas prácticas recomendadas no sean adecuadas o suficientes para el entorno, considérelas como consideraciones útiles en lugar de como normas.

Temas

- Prácticas recomendadas de seguridad preventivas para Amazon Keyspaces
- Prácticas recomendadas de detección de seguridad para Amazon Keyspaces

Prácticas recomendadas de seguridad preventivas para Amazon Keyspaces

Las siguientes prácticas recomendadas de seguridad se consideran preventivas porque pueden ayudarle a anticipar y evitar incidentes de seguridad en Amazon Keyspaces.

Uso del cifrado en reposo

Amazon Keyspaces cifra en reposo todos los datos de usuario que se almacenan en tablas mediante claves de cifrado almacenadas en AWS Key Management Service (AWS KMS). Esto proporciona una capa adicional de protección de datos al proteger los datos del acceso no autorizado al almacenamiento subyacente.

De forma predeterminada, Amazon Keyspaces utiliza una Clave propiedad de AWS para cifrar todas sus tablas. Si esta clave no existe, se crea una para usted. Las claves predeterminadas de servicio no se pueden deshabilitar.

Como alternativa, puede utilizar una clave administrada por el cliente para el cifrado en reposo. Para obtener más información, consulte Cifrado en reposo de Amazon Keyspaces.

Uso de roles de IAM para autenticar el acceso a Amazon Keyspaces

Para que los usuarios, las aplicaciones y otros servicios de AWS puedan acceder a Amazon Keyspaces, deben incluir credenciales de AWS válidas en sus solicitudes de la API de AWS. No debe almacenar las credenciales de AWS de forma directa en la aplicación ni en una instancia EC2. Estas son las credenciales a largo plazo que no rotan automáticamente, por lo tanto,

podrían tener un impacto empresarial significativo si se comprometen. Un rol de IAM lo habilita a obtener claves de acceso temporal que se pueden utilizar para tener acceso a los servicios y recursos de AWS.

Para obtener más información, consulte Roles de IAM.

Uso de políticas de IAM para autorización base de Amazon Keyspaces

Al conceder permisos, usted decide quién los obtiene, para qué API de Amazon Keyspaces obtiene permisos y las acciones específicas que desea permitir en esos recursos. La implementación del privilegio mínimo es clave para reducir los riesgos de seguridad y el impacto que podría resultar de errores o intenciones maliciosas.

Vincule políticas de permisos a las identidades de IAM (es decir, usuarios, grupos y roles) y conceda así permisos para realizar operaciones en los recursos de Amazon Keyspaces.

Para hacerlo, utilice lo siguiente:

- Políticas administradas (predefinidas) de AWS
- Políticas administradas por el cliente

Uso de condiciones de políticas de IAM para control de acceso preciso

Al conceder permisos en Amazon Keyspaces, puede especificar condiciones que determinen cómo se aplica una política de permisos. La implementación del privilegio mínimo es clave para reducir los riesgos de seguridad y el impacto que podría resultar de errores o intenciones maliciosas.

Puede especificar las condiciones al conceder permisos utilizando la política de IAM. Por ejemplo, puede hacer lo siguiente:

- Conceder permisos para permitir a los usuarios el acceso de solo lectura a determinados espacios de claves o tablas.
- Conceder permisos para permitir a un usuario el acceso de escritura a una tabla determinada en función de la identidad de dicho usuario.

Para obtener más información, consulte Ejemplos de políticas basadas en identidades.

Tenga en cuenta el cifrado del lado del cliente

Si almacena datos sensibles o confidenciales en Amazon Keyspaces, es posible que desee cifrar dichos datos lo más cerca posible de su origen para que estén protegidos durante todo su ciclo

de vida. El cifrado de su información confidencial en tránsito y en reposo ayuda a garantizar que los datos de texto no cifrado no estén disponibles para ningún tercero.

Prácticas recomendadas de detección de seguridad para Amazon Keyspaces

Las siguientes prácticas recomendadas de seguridad se consideran de detección porque pueden ayudarle a detectar posibles debilidades e incidentes de seguridad.

Uso de AWS CloudTrail para monitorear el uso de claves AWS Key Management Service de AWS KMS (AWS KMS)

Al utilizar una clave AWS KMS gestionada por el cliente para el cifrado en reposo, el uso de esta clave se registra en AWS CloudTrail. CloudTrail proporciona visibilidad de la actividad del usuario mediante el registro de las acciones realizadas en su cuenta. CloudTrail registra información importante acerca de cada acción, incluyendo quién hizo la solicitud, los servicios utilizados, las acciones realizadas, los parámetros de las acciones y los elementos de respuesta devueltos por el servicio de AWS. Esta información le ayuda a realizar un seguimiento de los cambios realizados en sus recursos de AWS y solucionar problemas operativos. CloudTrail facilita la conformidad con las políticas internas y las normas reglamentarias.

Puede utilizar CloudTrail para auditar el uso de las claves. CloudTrail crea archivos de registro que contienen un historial de las llamadas a la API de AWS y de los eventos relacionados de su cuenta. Estos archivos de registro incluyen todas las solicitudes a la API de AWS KMS que se realizaron mediante la consola, los SDK de AWS y las herramientas de línea de comandos, además de aquellas realizadas a través de los servicios de AWS integrados. Puede utilizar estos archivos de registro para obtener información sobre cuándo se utilizó la clave de AWS KMS, la operación que se solicitó, la identidad del solicitante, la dirección IP de la que procedía la solicitud, etc. Para obtener más información, consulte Registro de llamadas a la API de AWS Key Management Service con AWS CloudTrail en la Guía del usuario de AWS CloudTrail.

Uso de CloudTrail para monitorear las operaciones de lenguaje de definición de datos (DDL) de Amazon Keyspaces

CloudTrail proporciona visibilidad de la actividad del usuario mediante el registro de las acciones realizadas en su cuenta. CloudTrail registra información importante acerca de cada acción, incluyendo quién hizo la solicitud, los servicios utilizados, las acciones realizadas, los parámetros de las acciones y los elementos de respuesta devueltos por el servicio de AWS. Esta información

le ayuda a realizar un seguimiento de los cambios realizados en sus recursos de AWS y a solucionar problemas operativos. CloudTrail facilita la conformidad con las políticas internas y las normas reglamentarias.

Todas las operaciones DDL de Amazon Keyspaces se registran automáticamente en CloudTrail. Las operaciones DDL le permiten crear y administrar espacios de claves y tablas de Amazon Keyspaces.

Al producirse una actividad en Amazon Keyspaces, dicha actividad se registra en un evento de CloudTrail junto con otros eventos de servicios de AWS en el historial de eventos. Para obtener más información, consulte Registro de operaciones de Amazon Keyspaces mediante AWS CloudTrail. Puede ver, buscar y descargar los últimos eventos de la Cuenta de AWS. Para obtener más información, consulte Ver eventos con el historial de eventos de CloudTrail en la Guía del usuario de AWS CloudTrail.

Para mantener un registro continuo de los eventos en su Cuenta de AWS, incluyendo los eventos de Amazon Keyspaces, cree un registro de seguimiento. Un seguimiento habilita a CloudTrail a enviar archivos de registro a un bucket de Amazon Simple Storage Service (Amazon S3). De forma predeterminada, al crear un registro de seguimiento en la consola, este se aplica a todas las Regiones de AWS. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail.

Etiquetado de recursos de Amazon Keyspaces para identificación y automatización

Puede asignar metadatos a los recursos de AWS en forma de etiquetas. Cada etiqueta es un simple rótulo que consta de una clave definida por el cliente y un valor opcional que puede facilitar la administración, búsqueda y filtrado de recursos.

El etiquetado permite implementar controles agrupados. Aunque no hay tipos inherentes de etiquetas, lo habilitan a clasificar los recursos según su finalidad, propietario, entorno u otros criterios. A continuación se muestran algunos ejemplos:

- Acceso: se utiliza para controlar el acceso a los recursos de Amazon Keyspaces en función de las etiquetas. Para obtener más información, consulte the section called “Autorización basada en etiquetas de Amazon Keyspaces”.
- Seguridad: se utiliza para determinar requisitos como la configuración de la protección de datos.

- Confidencialidad: identificador del nivel específico de confidencialidad de datos que admite un recurso.
- Entorno: utilizado para distinguir entre el desarrollo, la prueba y la infraestructura de producción.

Para obtener más información, consulte Estrategias de etiquetado de AWS y Adición de etiquetas y rótulos a los recursos.

Referencia del lenguaje CQL para Amazon Keyspaces (para Apache Cassandra)

Tras conectarse a un punto de conexión de Amazon Keyspaces (para Apache Cassandra), utiliza Cassandra Query Language (CQL) para trabajar con su base de datos. CQL es similar en muchos aspectos a Structured Query Language (SQL).

Temas

- Elementos de Cassandra Query Language (CQL) en Amazon Keyspaces
- Instrucciones DDL (lenguaje de definición de datos) en Amazon Keyspaces
- Instrucciones DML (lenguaje de manipulación de datos) en Amazon Keyspaces
- Funciones integradas en Amazon Keyspaces

Elementos de Cassandra Query Language (CQL) en Amazon Keyspaces

Obtenga información sobre los elementos de Cassandra Query Language (CQL) compatibles con Amazon Keyspaces, incluyendo los identificadores, las constantes, los términos y los tipos de datos.

Temas

- Identificadores
- Constantes
- Términos
- Tipos de datos
- Codificación JSON de los tipos de datos de Amazon Keyspaces

Identificadores

Los identificadores (o nombres) se utilizan para identificar tablas, columnas y otros objetos. Un identificador puede estar entre comillas o no. Se aplica lo siguiente.

```
identifier          ::=  unquoted_identifier | quoted_identifier
```

```
unquoted_identifier ::=  re('[a-zA-Z][a-zA-Z0-9_]*')
quoted_identifier   ::=  '"' (any character where " can appear if doubled)+ '"'
```

Constantes

Se definen las siguientes constantes.

```
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+
```

Términos

Un término denota el tipo de valores que se admiten. Los términos se definen del siguiente modo.

```
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
```

Tipos de datos

Amazon Keyspaces admite los siguientes tipos de datos:

Tipos de cadena

Tipo de datos	Descripción
`ascii`	Representa una cadena de caracteres ASCII.

Tipo de datos	Descripción
`text`	Representa una cadena codificada en UTF-8.
`varchar`	Representa una cadena codificada en UTF-8 (`varchar` es un alias de `text`).

Tipos numéricos

Tipo de datos	Descripción
`bigint`	Representa un entero largo con signo de 64 bits.
`counter`	Representa un contador entero con signo de 64 bits. Para obtener más información, consulte the section called “Contadores”.
`decimal`	Representa un decimal de precisión variable.
`double`	Representa un valor de coma flotante IEEE 754 de 64 bits.
`float`	Representa un valor de coma flotante IEEE 754 de 32 bits.
`int`	Representa un entero con signo de 32 bits.
`varint`	Representa un entero de precisión arbitraria.

Contadores

Una columna `counter` contiene un entero con signo de 64 bits. El valor del contador se incrementa o decrementa mediante la instrucción the section called “UPDATE” y no se puede establecer directamente. Esto hace que las columnas `counter` sean útiles para el seguimiento de recuentos. Por ejemplo, puede utilizar contadores para realizar un seguimiento del número de entradas en un archivo de registro o del número de veces que se ha visto una publicación en una red social. Se aplican las siguientes restricciones a las columnas `counter`:

- Una columna de tipo `counter` no puede formar parte de la `primary key` de una tabla.
- En una tabla que contenga una o más columnas de tipo `counter`, todas las columnas de esa tabla deben ser de tipo `counter`.

En los casos en que la actualización de un contador fallase (por ejemplo, debido a tiempos de espera o a la pérdida de conexión con Amazon Keyspaces), el cliente no sabe si el valor del contador se actualizó. Si se vuelve a intentar la actualización, es posible que la actualización del valor del contador se aplique por segunda vez.

Tipo de blob

Tipo de datos	Descripción
`blob`	Representa bytes arbitrarios.

Tipo booleano

Tipo de datos	Descripción
`boolean`	Representa `true` o `false`.

Tipos relacionados en el tiempo

Tipo de datos	Descripción
`timestamp`	Entero con signo de 64 bits que representa la fecha y la hora desde epoch (1 de enero de 1970 a las 00:00:00 GMT) en milisegundos.
`timeuuid`	Representa un UUID versión 1.

Tipos de colección

Tipo de datos	Descripción
`list`	Representa una colección ordenada de elementos literales.
`map`	Representa una colección desordenada de pares clave-valor.
`set`	Representa una colección desordenada de uno o más elementos literales.

Una columna de colección se declara utilizando el tipo de colección seguido de otro tipo de datos (por ejemplo, `TEXT` o `INT`) entre corchetes angulares. Puede crear una columna con un `SET` de `TEXT` o puede crear un `MAP` de `TEXT` y pares clave-valor `INT`, como se muestra en el siguiente ejemplo.

```
SET <TEXT>
MAP <TEXT, INT>
```

Una colección no congelada le permite realizar actualizaciones en cada elemento individual de la colección. Las marcas de tiempo del lado del cliente y los ajustes de periodo de vida (TTL) se almacenan para elementos individuales.

Al utilizar la palabra clave `FROZEN` en un tipo de colección, los valores de la colección se serializan en un único valor inmutable y Amazon Keyspaces los trata como un `BLOB`. Esto es una colección congelada. Una instrucción `INSERT` o `UPDATE` sobrescribe toda la colección congelada. No se pueden realizar actualizaciones de elementos individuales dentro de una colección congelada.

Las marcas de tiempo del lado del cliente y los ajustes de periodo de vida (TTL) se aplican a toda la colección congelada, no a elementos individuales. Las columnas `Frozen` de la colección pueden formar parte de la `PRIMARY KEY` de una tabla.

Puede anidar colecciones congeladas. Por ejemplo, puede definir un `MAP` dentro de un `SET` si el `MAP` utiliza la palabra clave `FROZEN`, como se muestra en el siguiente ejemplo.

```
SET <FROZEN> <MAP <TEXT, INT>>>
```

Amazon Keyspaces admite el anidamiento de hasta cinco niveles de colecciones congeladas de forma predeterminada. Para obtener más información, consulte the section called “Cuotas de servicio de Amazon Keyspaces”. Para obtener más información sobre las diferencias funcionales con Apache Cassandra, consulte the section called “Colecciones FROZEN”. Para obtener más información sobre la sintaxis CQL, consulte the section called “CREATE TABLE” y the section called “ALTER TABLE”.

Tipo de tupla

El tipo de datos `tuple` representa un grupo acotado de elementos literales. Puede utilizar una tupla como alternativa a un `user defined type`. No es necesario que utilice la palabra clave `FROZEN` para las tuplas. Esto se debe a que una tupla siempre está congelada y no puede actualizar los elementos de forma individual.

Otros tipos

Tipo de datos	Descripción
`inet`	Una cadena que representa una dirección IP, en formato IPv4 o IPv6.

Estático

En una tabla de Amazon Keyspaces con columnas agrupadas en clústeres, puede utilizar la `STATIC` palabra clave para crear una columna estática de cualquier tipo.

La siguiente instrucción es un ejemplo de ello.

```
my_column INT STATIC
```

Para obtener más información sobre cómo trabajar con columnas estáticas, consulte. the section called “Columnas estáticas”

Codificación JSON de los tipos de datos de Amazon Keyspaces

Amazon Keyspaces ofrece las mismas correspondencias de tipos de datos JSON que Apache Cassandra. En la siguiente tabla se describen los tipos de datos que Amazon Keyspaces acepta en las instrucciones `INSERT JSON` y los tipos de datos que Amazon Keyspaces utiliza al devolver datos con la instrucción `SELECT JSON`.

En los tipos de datos de campo único como `float`, `int`, `UUID` y `date` también puede insertar datos como `string`. En los tipos de datos compuestos y colecciones, como `tuple`, `map` y `list` también puede insertar datos como JSON o como `JSON string` codificado.

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones **`INSERT JSON`**	Tipos de datos devueltos en instrucciones **`SELECT JSON`**	Notas
`ascii`	`string`	`string`	Utiliza el escape de caracteres JSON `\u`.
`bigint`	`integer, string`	`integer`	La cadena debe ser un entero válido de 64 bits.
`blob`	`string`	`string`	La cadena debe empezar con `0x` seguido de un número par de dígitos hexadecimales.
`boolean`	`boolean, string`	`boolean`	La cadena debe ser `true` o `false`.
`date`	`string`	`string`	Fecha en formato `YYYY-MM-DD` , zona horaria UTC.
`decimal`	`integer, float, string`	`float`	Puede superar la precisión de punto flotante IEEE-754 de 32 o 64 bits en el decodificador del lado del cliente.
`double`	`integer, float, string`	`float`	La cadena debe ser un entero válido o flotante.

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones **`INSERT JSON`**	Tipos de datos devueltos en instrucciones **`SELECT JSON`**	Notas
`float`	`integer, float, string`	`float`	La cadena debe ser un entero válido o flotante.
`inet`	`string`	`string`	Dirección IPv4 o IPv6.
`int`	`integer, string`	`integer`	La cadena debe ser un entero válido de 32 bits.
`list`	`list, string`	`list`	Utiliza la representación de lista JSON nativa.
`map`	`map, string`	`map`	Utiliza la representación de mapa JSON nativa.
`smallint`	`integer, string`	`integer`	La cadena debe ser un entero válido de 16 bits.
`set`	`list, string`	`list`	Utiliza la representación de lista JSON nativa.
`text`	`string`	`string`	Utiliza el escape de caracteres JSON \u.
`time`	`string`	`string`	Hora del día en formato `HH-MM-SS[.fffffffff]` .

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones **`INSERT JSON`**	Tipos de datos devueltos en instrucciones **`SELECT JSON`**	Notas
`timestamp`	`integer, string`	`string`	Una marca temporal. Las constantes de cadena permiten almacenar las marcas de tiempo como fechas. Se devuelven marcas de fecha con formato `YYYY-MM-DD HH:MM:SS.SSS` .
`timeuuid`	`string`	`string`	UUID tipo 1. Consulte constants para obtener el formato UUID.
`tinyint`	`integer, string`	`integer`	La cadena debe ser un entero válido de 8 bits.
`tuple`	`list, string`	`list`	Utiliza la representación de lista JSON nativa.
`uuid`	`string`	`string`	Consulte constants para obtener el formato UUID.
`varchar`	`string`	`string`	Utiliza el escape de caracteres JSON \u.

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones **`INSERT JSON`**	Tipos de datos devueltos en instrucciones **`SELECT JSON`**	Notas
`varint`	`integer, string`	`integer`	Longitud variable; podría desbordar enteros de 32 o 64 bits en el decodificador del lado del cliente.

Instrucciones DDL (lenguaje de definición de datos) en Amazon Keyspaces

Lenguaje de definición de datos (DDL) es el conjunto de instrucciones de Cassandra Query Language (CQL) que se utiliza para administrar estructuras de datos en Amazon Keyspaces (para Apache Cassandra), como espacios de claves y tablas. Se utiliza DDL para crear estas estructuras de datos, modificarlas una vez creadas y eliminarlas cuando ya no estén en uso. Amazon Keyspaces realiza las operaciones DDL de forma asíncrona. Para obtener más información sobre cómo confirmar si una operación asíncrona se ha completado, consulte the section called “Creación y eliminación asíncronas de espacios de claves y tablas”.

Se admiten las siguientes instrucciones DDL:

- CREATE KEYSPACE
- ALTER KEYSPACE
- DROP KEYSPACE
- CREATE TABLE
- ALTER TABLE
- RESTORE TABLE
- DROP TABLE

Temas

- Keyspaces

- Tablas

Keyspaces

Un espacio de claves agrupa tablas relacionadas que son relevantes para una o más aplicaciones. En términos de un sistema de administración de bases de datos relacionales (RDBMS), los espacios de claves son aproximadamente similares a las bases de datos, espacios de tablas o construcciones similares.

> **Note**
>
> En Apache Cassandra, los espacios de claves determinan cómo se replican los datos entre varios nodos de almacenamiento. Sin embargo, Amazon Keyspaces es un servicio totalmente administrado: los detalles de su capa de almacenamiento se administran en su nombre. Por este motivo, los espacios de claves en Amazon Keyspaces son solo construcciones lógicas y no están relacionados con el almacenamiento físico subyacente.

Para obtener información sobre los límites de cuota y las restricciones de los espacios de claves de Amazon Keyspaces, consulte Cuotas.

Declaraciones para espacios clave

- CREATE KEYSPACE
- ALTER KEYSPACE
- DROP KEYSPACE

CREATE KEYSPACE

Utilice la instrucción `CREATE KEYSPACE` para crear un nuevo espacio de claves.

Sintaxis

```
create_keyspace_statement ::=
    CREATE KEYSPACE [ IF NOT EXISTS ] keyspace_name
    WITH options
```

Donde:

- *keyspace_name* es el nombre del espacio de claves que se va a crear.
- opciones son una o más de las siguientes opciones:
  - `REPLICATION`: un mapa que indica la estrategia de réplica del espacio de claves:
    - `SingleRegionStrategy`: para un espacio de claves de región única. (Obligatorio)
    - `NetworkTopologyStrategy`— Especifique al menos dos y hasta seis Regiones de AWS. El factor de réplica para cada región es tres. (Opcional)
  - `DURABLE_WRITES`: las escrituras en Amazon Keyspaces son siempre duraderas, por lo que esta opción no es necesaria. Sin embargo, si se especifica, el valor debe ser `true`.
  - `TAGS`: una lista de etiquetas de par clave-valor que se vinculan al recurso al crearlo. (Opcional)

Ejemplo

Cree un espacio de claves de la siguiente manera.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class': 'SingleRegionStrategy'} and TAGS ={'key1':'val1',
 'key2':'val2'} ;
```

Para crear un espacio de claves multirregión, especifique `NetworkTopologyStrategy` e incluya al menos dos y hasta seis Regiones de AWS. El factor de réplica para cada región es tres.

```
CREATE KEYSPACE my_keyspace
    WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-
southeast-1':'3','eu-west-1':'3'};
```

ALTER KEYSPACE

Utilice `ALTER KEYSPACE` para añadir o eliminar etiquetas de un espacio de claves.

Sintaxis

```
alter_keyspace_statement ::=
    ALTER KEYSPACE keyspace_name
    [[ADD | DROP] TAGS
```

Donde:

- *keyspace_name* es el nombre del espacio de claves que se va a modificar.

- TAGS: una lista de etiquetas de par clave-valor que se van a añadir o eliminar del espacio de claves.

Ejemplo

Modifique un espacio de claves de la siguiente manera.

```
ALTER KEYSPACE "myGSGKeyspace" ADD TAGS {'key1':'val1', 'key2':'val2'};
```

DROP KEYSPACE

Utilice la DROP KEYSPACE instrucción para eliminar un espacio de claves, incluido todo su contenido, como las tablas.

Sintaxis

```
drop_keyspace_statement ::=
    DROP KEYSPACE [ IF EXISTS ] keyspace_name
```

Donde:

- keyspace_name es el nombre del espacio de claves que se va a eliminar.

Ejemplo

```
DROP KEYSPACE "myGSGKeyspace";
```

Tablas

Las tablas son las estructuras de datos principales de Amazon Keyspaces. Los datos de una tabla se organizan en filas y columnas. Un subconjunto de esas columnas se utiliza para determinar la partición (y, en última instancia, la ubicación de los datos) mediante la especificación de una clave de partición.

Otro conjunto de columnas puede definirse en columnas de agrupación, lo que significa que pueden participar como predicados en la ejecución de consultas.

De forma predeterminada, las tablas nuevas se crean con capacidad de rendimiento bajo demanda. Puede cambiar el modo de capacidad para las tablas nuevas y existentes. Para obtener más

información sobre los modos de capacidad de rendimiento de lectura/escritura, consulte the section called “Modos de capacidad de lectura/escritura”.

Para las tablas en modo aprovisionado, puede configurarlas de forma opcionalAUTOSCALING_SETTINGS. Para obtener más información sobre el escalado automático de Amazon Keyspaces y las opciones disponibles, consulte. the section called “Uso de CQL”

Para obtener información sobre los límites de cuota y las restricciones de las tablas de Amazon Keyspaces, consulte Cuotas.

Declaraciones para tablas

- CREATE TABLE
- ALTER TABLE
- RESTORE TABLE
- DROP TABLE

CREATE TABLE

Utilice la instrucción CREATE TABLE para crear una tabla nueva.

Sintaxis

```
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 ]
```

```
                              | options
                              | CUSTOM_PROPERTIES
                              | AUTOSCALING_SETTINGS
                              | default_time_to_live
                              | TAGS

clustering_order       ::=  column_name (ASC | DESC) ( ',' column_name (ASC | DESC) )*
```

Donde:

- *table_name* es el nombre de la tabla que se va a crear.
- *column_definition* consta de lo siguiente:
  - *column_name*: el nombre de la columna.
  - *cql_type*: un tipo de datos de Amazon Keyspaces (consulte Tipos de datos).
  - *FROZEN*: designa esta columna de tipo `collection` (por ejemplo, `LIST`, `SET`, o `MAP`) como congelada. Una colección congelada se serializa en un único valor inmutable y se trata como un `BLOB`. Para obtener más información, consulte the section called “Tipos de colección”.
  - *STATIC*: designa esta columna como estática. Las columnas estáticas almacenan valores que comparten todas las filas de la misma partición.
  - *PRIMARY KEY*: designa esta columna como clave principal de la tabla.
- *primary_key* consta de lo siguiente:
  - `partition_key`
  - `clustering_columns`
- *partition_key*:
  - La clave de partición puede ser una única columna o un valor compuesto formado por dos o más columnas. La porción de clave de partición de la clave principal es obligatoria y determina cómo almacena Amazon Keyspaces sus datos.
- *clustering_columns*:
  - La porción de columna de agrupación opcional de su clave principal determina cómo se agrupan y clasifican los datos dentro de cada partición.
- *table_options* consta de lo siguiente:
  - *CLUSTERING ORDER BY*: el CLUSTERING ORDER predeterminado en una tabla se compone de sus claves de agrupación en la dirección de ordenación `ASC` (ascendente). Especifíquelo para anular el comportamiento de ordenación predeterminado.
  - *CUSTOM_PROPERTIES*: un mapa de ajustes específicos de Amazon Keyspaces.

- `capacity_mode`: especifica el modo de capacidad de rendimiento de lectura/escritura para la tabla. Las opciones son `throughput_mode:PAY_PER_REQUEST` y `throughput_mode:PROVISIONED`. El modo de capacidad aprovisionada requiere `read_capacity_units` y `write_capacity_units` como entradas. El valor predeterminado es `throughput_mode:PAY_PER_REQUEST`.
- `client_side_timestamps`: especifica si las marcas de tiempo del lado del cliente están habilitadas o deshabilitadas para la tabla. Las opciones son `{'status': 'enabled'}` y `{'status': 'disabled'}`. Si no se especifica, el valor predeterminado es `status:disabled`. Una vez habilitadas las marcas de tiempo del lado del cliente para una tabla, este ajuste no se puede deshabilitar.
- `encryption_specification`: especifica las opciones de cifrado para el cifrado en reposo. Si no se especifica, el valor predeterminado es `encryption_type:AWS_OWNED_KMS_KEY`. La clave gestionada por el cliente de la opción de cifrado requiere la AWS KMS clave en formato Amazon Resource Name (ARN) como entrada::`kms_key_identifier:ARN`. `kms_key_identifier:ARN`
- `point_in_time_recovery`: Especifica si point-in-time la restauración está habilitada o deshabilitada para la tabla. Las opciones son `status:enabled` y `status:disabled`. Si no se especifica, el valor predeterminado es `status:disabled`.
- `replica_updates`: Especifica la configuración de una tabla multirregional que es específica de una Región de AWS. En el caso de una tabla multirregional, puede configurar la capacidad de lectura de la tabla de forma diferente según la tabla. Región de AWS Puede hacerlo configurando los siguientes parámetros. Para obtener más información y ejemplos, consulte the section called “Creación de una tabla multirregional con modo de capacidad aprovisionada y escalado automático (CQL)”.
  - `region`— La réplica Región de AWS de la tabla con los siguientes ajustes:
    - `read_capacity_units`
- TTL: habilita la configuración personalizada del período de vida para la tabla. Para habilitarla, utilice `status:enabled`. El valor predeterminado es `status:disabled`. Una vez habilitado, TTL no se puede deshabilitar para la tabla.

- *`AUTOSCALING_SETTINGS`*incluye la siguiente configuración opcional para las tablas en modo aprovisionado. Para obtener más información y ejemplos, consulte the section called “Cree una tabla nueva con escalado automático mediante CQL”.
  - `provisioned_write_capacity_autoscaling_update`:

- `autoscaling_disabled`— Para habilitar el escalado automático de la capacidad de escritura, defina el valor en`false`. El valor predeterminado es `true`. (Opcional)
- `minimum_units`— El nivel mínimo de rendimiento de escritura que la tabla debería estar siempre lista para soportar. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
- `maximum_units`— El nivel máximo de rendimiento de escritura que la tabla debería estar siempre lista para soportar. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
- `scaling_policy`— Amazon Keyspaces es compatible con la política de seguimiento de objetivos. El objetivo de escalado automático es la capacidad de escritura aprovisionada de la tabla.
  - `target_tracking_scaling_policy_configuration`— Para definir la política de seguimiento del objetivo, debe definir el valor objetivo. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.
    - `target_value`— La tasa de utilización objetivo de la tabla. El escalado automático de Amazon Keyspaces garantiza que la relación entre la capacidad consumida y la capacidad aprovisionada se mantenga en este valor o cerca de él. `target_value` se define como un porcentaje. El doble, entre 20 y 90. (Obligatorio)
    - `scale_in_cooldown`— Un período de enfriamiento en segundos entre las actividades de escalado que permite que la mesa se estabilice antes de que comience otra actividad de escalado. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
    - `scale_out_cooldown`— Un período de enfriamiento en segundos entre las actividades de escalado que permite que la mesa se estabilice antes de que comience otra actividad de escalado horizontal. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
    - `disable_scale_in`: A `boolean` que especifica si `scale-in` está deshabilitada o habilitada para la tabla. Este parámetro está desactivado de forma predeterminada. Para activarlo`scale-in`, defina el `boolean` valor en`FALSE`. Esto significa que la capacidad de una mesa se reduce automáticamente en tu nombre. (Opcional)
- `provisioned_read_capacity_autoscaling_update`:
  - `autoscaling_disabled`— Para habilitar el escalado automático de la capacidad de lectura, defina el valor en`false`. El valor predeterminado es `true`. (Opcional)

- `minimum_units`— El nivel mínimo de rendimiento que la tabla debería estar siempre lista para soportar. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
- `maximum_units`— El nivel máximo de rendimiento que la tabla debería estar siempre lista para soportar. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
- `scaling_policy`— Amazon Keyspaces es compatible con la política de seguimiento de objetivos. El objetivo de escalado automático es la capacidad de lectura aprovisionada de la tabla.
  - `target_tracking_scaling_policy_configuration`— Para definir la política de seguimiento del objetivo, debe definir el valor objetivo. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.
    - `target_value`— La tasa de utilización objetivo de la tabla. El escalado automático de Amazon Keyspaces garantiza que la relación entre la capacidad consumida y la capacidad aprovisionada se mantenga en este valor o cerca de él. `target_value` se define como un porcentaje. El doble, entre 20 y 90. (Obligatorio)
    - `scale_in_cooldown`— Un período de enfriamiento en segundos entre las actividades de escalado que permite que la mesa se estabilice antes de que comience otra actividad de escalado. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
    - `scale_out_cooldown`— Un período de enfriamiento en segundos entre las actividades de escalado que permite que la mesa se estabilice antes de que comience otra actividad de escalado horizontal. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
    - `disable_scale_in`: A `boolean` que especifica si `scale-in` está deshabilitada o habilitada para la tabla. Este parámetro está desactivado de forma predeterminada. Para activarlo`scale-in`, defina el `boolean` valor en`FALSE`. Esto significa que la capacidad de una mesa se reduce automáticamente en tu nombre. (Opcional)
- `replica_updates`: especifica la configuración de escalado automático Región de AWS específica de una tabla multirregión. En el caso de una tabla con varias regiones, puede configurar la capacidad de lectura de la tabla de forma diferente según la tabla. Región de AWS Puede hacerlo configurando los siguientes parámetros. Para obtener más información y ejemplos, consulte the section called “Creación de una tabla multirregional con modo de capacidad aprovisionada y escalado automático (CQL)”.

- `region`— La réplica Región de AWS de la tabla con los siguientes ajustes:
  - `provisioned_read_capacity_autoscaling_update`
    - `autoscaling_disabled`— Para habilitar el escalado automático de la capacidad de lectura de la tabla, defina el valor en`false`. El valor predeterminado es `true`. (Opcional)

      > **Note**
      >
      > El escalado automático de una tabla multirregional debe estar activado o desactivado para todas las réplicas de la tabla.

    - `minimum_units`— El nivel mínimo de rendimiento de lectura que la tabla debe estar siempre lista para soportar. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de tu cuenta (40 000 de forma predeterminada).
    - `maximum_units`— El nivel máximo de rendimiento de lectura que la tabla debería poder soportar en todo momento. El valor debe estar comprendido entre 1 y la cuota máxima de rendimiento por segundo de su cuenta (40 000 de forma predeterminada).
    - `scaling_policy`— Amazon Keyspaces es compatible con la política de seguimiento de objetivos. El objetivo de escalado automático es la capacidad de lectura aprovisionada de la tabla.
      - `target_tracking_scaling_policy_configuration`— Para definir la política de seguimiento del objetivo, debe definir el valor objetivo. Para obtener más información sobre el seguimiento de objetivos y los períodos de enfriamiento, consulte Políticas de escalado de seguimiento de objetivos en la Guía del usuario de Application Auto Scaling.
        - `target_value`— La tasa de utilización objetivo de la tabla. El escalado automático de Amazon Keyspaces garantiza que la relación entre la capacidad de lectura consumida y la capacidad de lectura aprovisionada se mantenga en este valor o cerca de él. `target_value` se define como un porcentaje. El doble, entre 20 y 90. (Obligatorio)
        - `scale_in_cooldown`— Un período de enfriamiento en segundos entre las actividades de escalado que permite que la mesa se estabilice antes de que comience otra actividad de escalado. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)

        - `scale_out_cooldown`— Un período de enfriamiento en segundos entre las actividades de escalado que permite que la mesa se estabilice antes de que comience otra actividad de escalado horizontal. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
        - `disable_scale_in`: A `boolean` que especifica si `scale-in` está deshabilitada o habilitada para la tabla. Este parámetro está desactivado de forma predeterminada. Para activarlo`scale-in`, defina el `boolean` valor en`FALSE`. Esto significa que la capacidad de lectura de una tabla se reduce automáticamente en tu nombre. (Opcional)
    - `default_time_to_live`: la configuración predeterminada de periodo de vida en segundos para la tabla.
    - `TAGS`: una lista de etiquetas de par clave-valor que se vinculan al recurso al crearlo.
- *`clustering_order`* consta de lo siguiente:
    - *`column_name`*: el nombre de la columna.
    - *`ASC | DESC`*: establece el modificador de orden ascendente (`ASC`) o descendente (`DESC`). Si no se especifica, el valor predeterminado es ASC.

Ejemplo

```
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;
```

En una tabla que utilice columnas de agrupación, las columnas que no sean de agrupación pueden declararse como estáticas en la definición de la tabla. Para obtener más información sobre las columnas estáticas, consulte the section called “Columnas estáticas”.

Ejemplo

```
CREATE TABLE "my_keyspace".my_table (
                                                id int,
                                                name text,
                                                region text,
                                                division text,
                                                project text STATIC,
                                                PRIMARY KEY (id,division));
```

ALTER TABLE

Utilice la instrucción ALTER TABLE para añadir nuevas columnas, añadir etiquetas o modificar las propiedades personalizadas de la tabla.

Sintaxis

```
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
```

Donde:

- *table_name* es el nombre de la tabla que se va a modificar.
- *column_definition* es el nombre de la columna y el tipo de datos que se van a añadir.
- *column_definition_list* es una lista separada por comas de columnas colocadas entre paréntesis.
- *table_options* consta de lo siguiente:
  - *CUSTOM_PROPERTIES*: un mapa de ajustes específicos para Amazon Keyspaces.
    - `capacity_mode`: especifica el modo de capacidad de rendimiento de lectura/escritura para la tabla. Las opciones son `throughput_mode:PAY_PER_REQUEST` y `throughput_mode:PROVISIONED`. El modo de capacidad aprovisionada requiere `read_capacity_units` y `write_capacity_units` como entradas. El valor predeterminado es `throughput_mode:PAY_PER_REQUEST`.
    - `client_side_timestamps`: especifica si las marcas de tiempo del lado del cliente están habilitadas o deshabilitadas para la tabla. Las opciones son `{'status': 'enabled'}` y `{'status': 'disabled'}`. Si no se especifica, el valor predeterminado es `status:disabled`. Una vez habilitadas las marcas de tiempo del lado del cliente para una tabla, este ajuste no se puede deshabilitar.
    - `encryption_specification`: especifica la opción de cifrado para el cifrado en reposo. Las opciones son `encryption_type:AWS_OWNED_KMS_KEY` y `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. La opción de cifrado de clave administrada por el cliente requiere la clave AWS KMS en formato de nombre de recurso de Amazon (ARN) como entrada: `kms_key_identifier:ARN`.
    - `point_in_time_recovery`: Especifica si point-in-time la restauración está habilitada o deshabilitada para la tabla. Las opciones son `status:enabled` y `status:disabled`. El valor predeterminado es `status:disabled`.
    - `replica_updates`: especifica la configuración Región de AWS específica de una tabla multirregional. En el caso de una tabla de varias regiones, puede configurar la capacidad de lectura de la tabla de forma diferente según la tabla. Región de AWS Puede hacerlo configurando los siguientes parámetros. Para obtener más información y ejemplos, consulte the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional (CQL)”.
      - `region`— La réplica Región de AWS de la tabla con los siguientes ajustes:
        - `read_capacity_units`

    - `ttl`: habilita la configuración personalizada del período de vida para la tabla. Para habilitarla, utilice `status:enabled`. El valor predeterminado es `status:disabled`. Una vez habilitado, `ttl` no se puede deshabilitar para la tabla.
  - *`AUTOSCALING_SETTINGS`*incluye la configuración de escalado automático opcional para las tablas aprovisionadas. Para ver la sintaxis y las descripciones detalladas, consultethe section called “CREATE TABLE”. Para ver ejemplos, consulte the section called “Habilite el escalado automático en una tabla existente mediante CQL”.
- `default_time_to_live`: la configuración predeterminada de periodo de vida en segundos para la tabla.
- *`TAGS`* es una lista de etiquetas de par clave-valor que se van a vincular al recurso.

> **Note**
>
> Con ALTER TABLE, solo puede cambiar una propiedad personalizada. No puede combinar más de un comando ALTER TABLE en la misma instrucción.

Ejemplos

La siguiente instrucción muestra cómo añadir una columna a una tabla existente.

```
ALTER TABLE mykeyspace.mytable ADD (ID int);
```

Esta instrucción muestra cómo añadir dos columnas de colección a una tabla existente:

- Una columna de colección congelada `col_frozen_list` que contiene una colección congelada anidada
- Una columna de colección no congelada `col_map` que contiene una colección congelada anidada

```
ALTER TABLE my_Table ADD(col_frozen_list FROZEN<LIST<FROZEN<SET<TEXT>>>>, col_map
 MAP<INT, FROZEN<SET<INT>>>);
```

Para cambiar el modo de capacidad de una tabla y especificar unidades de capacidad de lectura y escritura, puede utilizar la siguiente instrucción.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units':
 20}};
```

La siguiente instrucción especifica una clave KMS administrada por el cliente para la tabla.

```
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'
                   }
         };
```

Para habilitar point-in-time la restauración de una tabla, puede usar la siguiente declaración.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'point_in_time_recovery':
 {'status': 'enabled'}};
```

Para establecer un valor predeterminado de periodo de vida en segundos para una tabla, puede utilizar la siguiente instrucción.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Esta instrucción habilita la configuración personalizada de periodo de vida para una tabla.

```
ALTER TABLE mytable WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

RESTORE TABLE

Use la instrucción `RESTORE TABLE` para restaurar una tabla a un momento dado. Esta declaración requiere que point-in-time la recuperación esté habilitada en una tabla. Para obtener más información, consulte Recuperación a un momento dado.

Sintaxis

```
restore_table_statement ::=
    RESTORE TABLE restored_table_name FROM TABLE source_table_name
```

```
                    [ WITH table_options [ , ... ] ];
```

Donde:

- *restored_table_name* es el nombre de la tabla restaurada.
- *source_table_name* es el nombre de la tabla de origen.
- *table_options* consta de lo siguiente:
  - *restore_timestamp* es el tiempo del punto de restauración en formato ISO 8601. Si no se especifica, se utiliza la marca de tiempo actual.
  - *CUSTOM_PROPERTIES*: un mapa de ajustes específicos para Amazon Keyspaces.
    - `capacity_mode`: especifica el modo de capacidad de rendimiento de lectura/escritura para la tabla. Las opciones son `throughput_mode:PAY_PER_REQUEST` y `throughput_mode:PROVISIONED`. El modo de capacidad aprovisionada requiere `read_capacity_units` y `write_capacity_units` como entradas. La configuración predeterminada es la configuración actual de la tabla de origen.
    - `encryption_specification`: especifica la opción de cifrado para el cifrado en reposo. Las opciones son `encryption_type:AWS_OWNED_KMS_KEY` y `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. La clave gestionada por el cliente de la opción de cifrado requiere la AWS KMS clave en formato Amazon Resource Name (ARN) como entrada:. `kms_key_identifier:ARN` Para restaurar una tabla cifrada con una clave gestionada por el cliente en una tabla cifrada con una Clave propiedad de AWS, Amazon Keyspaces necesita acceder a la AWS KMS clave de la tabla de origen.
    - `point_in_time_recovery`: Especifica si point-in-time la restauración está habilitada o deshabilitada para la tabla. Las opciones son `status:enabled` y `status:disabled`. A diferencia de cuando se crean tablas nuevas, el estado predeterminado para tablas restauradas es `status:enabled` porque la configuración se hereda de la tabla de origen. Para deshabilitar la PITR para las tablas restauradas, debe configurar `status:disabled` de forma explícita.
    - `replica_updates`: especifica la configuración Región de AWS específica de una tabla multirregional. En el caso de una tabla de varias regiones, puede configurar la capacidad de lectura de la tabla de forma diferente según la tabla. Región de AWS Puede hacerlo configurando los siguientes parámetros.
      - `region`— La réplica Región de AWS de la tabla con los siguientes ajustes:
        - `read_capacity_units`

- *AUTOSCALING_SETTINGS*incluye la configuración de escalado automático opcional para las tablas aprovisionadas. Para obtener descripciones y sintaxis detalladas, consultethe section called “CREATE TABLE”.
- *TAGS* es una lista de etiquetas de par clave-valor que se van a vincular al recurso.

> **Note**
>
> Las tablas eliminadas solo se pueden restaurar al momento de la eliminación.

Ejemplo

```
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

Utilice la instrucción `DROP TABLE` para eliminar una tabla del espacio de claves.

Sintaxis

```
drop_table_statement ::=
    DROP TABLE [ IF EXISTS ] table_name
```

Donde:

- `IF EXISTS` evita que `DROP TABLE` falle si la tabla no existe. (Opcional)
- *table_name* es el nombre de la tabla que se va a desechar.

Ejemplo

```
DROP TABLE "myGSGKeyspace".employees_tbl;
```

Instrucciones DML (lenguaje de manipulación de datos) en Amazon Keyspaces

Lenguaje de manipulación de datos (DML) es el conjunto de instrucciones de Cassandra Query Language (CQL) que se utiliza para administrar estructuras de datos en las tablas de Amazon Keyspaces (para Apache Cassandra). Puede usar instrucciones DML para agregar, modificar o eliminar datos de una tabla.

También puede usar instrucciones DML para consultar datos en una tabla. (Tenga en cuenta que CQL no admite uniones ni subconsultas).

Temas

- SELECT
- INSERT
- UPDATE
- DELETE

SELECT

Utilice una instrucción SELECT para consultar datos.

Sintaxis

```
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 ] )*
```

Ejemplos

```
SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;

SELECT JSON name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

Acerca de una tabla que asigne los tipos de datos codificados en JSON a los tipos de datos de Amazon Keyspaces, consulte the section called “Codificación JSON de los tipos de datos de Amazon Keyspaces”.

Uso de la palabra clave **`IN`**

La palabra clave `IN` especifica la igualdad de uno o más valores. Se puede aplicar a la clave de partición y a la columna de agrupación. Los resultados se devuelven en el orden en que se presentan las claves en la instrucción `SELECT`.

Ejemplos

```
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;
```

Para obtener más información sobre la palabra clave `IN` y la forma en que Amazon Keyspaces procesa la instrucción, consulte the section called “Instrucción IN SELECT”.

Ordenación de los resultados

La cláusula `ORDER BY` especifica el orden de clasificación de los resultados devueltos. Toma como argumentos una lista de nombres de columnas junto con el orden de clasificación de cada columna. Solo puede especificar columnas de agrupación en las cláusulas de ordenación. Las columnas no agrupadas no están permitidas. Las opciones de ordenación son `ASC` para ordenación ascendente y `DESC` para ordenación descendente. Si se omite el orden de clasificación, se utiliza el orden predeterminado de la columna de agrupación. Acerca de los posibles criterios de ordenación, consulte. the section called “Ordenación de resultados”

Ejemplo

```
SELECT name, id, division, manager_id FROM "myGSGKeyspace".employees_tbl WHERE id =
 '012-34-5678' ORDER BY division;
```

Al utilizar `ORDER BY` con la palabra clave `IN`, los resultados se ordenan dentro de una página. No se admite la reordenación completa con la paginación deshabilitada.

TOKEN

Puede aplicar la función `TOKEN` a la columna `PARTITION KEY` en las cláusulas `SELECT` y `WHERE`. Con la función `TOKEN`, Amazon Keyspaces devuelve filas basadas en el valor del token asignado de la `PARTITION_KEY` en lugar de en el valor de la `PARTITION KEY`.

Las relaciones `TOKEN` no se admiten con la palabra clave `IN`.

Ejemplos

```
SELECT TOKEN(id) from my_table;

SELECT TOKEN(id) from my_table WHERE TOKEN(id) > 100 and TOKEN(id) < 10000;
```

Función TTL

Puede utilizar la función `TTL` con la instrucción `SELECT` para recuperar el tiempo de caducidad en segundos que se almacena para una columna. Si no se establece ningún valor `TTL`, la función devuelve `null`.

Ejemplo

```
SELECT TTL(my_column) from my_table;
```

La función `TTL` no se puede utilizar en columnas multicelda, como las colecciones.

Función de WRITETIME

Puede utilizar la función `WRITETIME` con la instrucción `SELECT` para recuperar la marca de tiempo que se almacena como metadatos para el valor de una columna solo si la tabla utiliza marcas de tiempo en el lado del cliente. Para obtener más información, consulte Marcas de tiempo del lado del cliente.

```
SELECT WRITETIME(my_column) from my_table;
```

La función `WRITETIME` no se puede utilizar en columnas multicelda, como las colecciones.

> **Note**
>
> Por compatibilidad con el comportamiento establecido de los controladores Cassandra, las políticas de autorización basadas en etiquetas no se aplican cuando se realizan operaciones en tablas del sistema mediante llamadas a la API de Cassandra Query Language (CQL) a través de los controladores y las herramientas para desarrolladores de Cassandra. Para obtener más información, consulte the section called “ Acceso a recursos de Amazon Keyspaces basado en etiquetas”.

INSERT

Utilice la instrucción `INSERT` para añadir una fila a una tabla.

Sintaxis

```
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 )* ')'
```

Ejemplo

```
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') ;
```

Actualización de parámetros

`INSERT` admite los siguientes valores como `update_parameter`:

- TTL: un valor de tiempo en segundos. El valor máximo configurable es de 630 720 000 segundos, lo que equivale a 20 años.

- `TIMESTAMP`: un valor `bigint` que representa el número de microsegundos desde la hora base estándar conocida como epoch: 1 de enero de 1970 a las 00:00:00 GMT. Una marca de tiempo en Amazon Keyspaces debe estar comprendida entre 2 días en el pasado y 5 minutos en el futuro.

Ejemplo

```
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;
```

Compatibilidad con JSON

Acerca de una tabla que asigne los tipos de datos codificados en JSON a los tipos de datos de Amazon Keyspaces, consulte the section called “Codificación JSON de los tipos de datos de Amazon Keyspaces”.

Puede utilizar la palabra clave `JSON` para insertar un mapa codificado con `JSON` como una única fila. Para las columnas que existan en la tabla pero que se omitan en la instrucción de inserción JSON, utilice `DEFAULT UNSET` para conservar los valores existentes. Utilice `DEFAULT NULL` para escribir un valor NULL en cada fila de columnas omitidas y sobrescribir los valores existentes (se aplican los cargos de escritura estándar). `DEFAULT NULL` es la opción predeterminada.

Ejemplo

```
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"}';
```

Si los datos JSON contienen claves duplicadas, Amazon Keyspaces almacena el último valor de la clave (de forma similar a Apache Cassandra). En el siguiente ejemplo, en el que la clave duplicada es `id`, se utiliza el valor `234-56-7890`.

Ejemplo

```
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

Utilice la instrucción UPDATE para modificar una fila en una tabla.

Sintaxis

```
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
```

Ejemplo

```
UPDATE "myGSGKeyspace".employees_tbl SET pay_scale = 5 WHERE id = '567-89-0123' AND
 division = 'Marketing' ;
```

Para incrementar un `counter`, utilice la siguiente sintaxis. Para obtener más información, consulte the section called “Contadores”.

```
UPDATE ActiveUsers SET counter = counter + 1  WHERE user = A70FE1C0-5408-4AE3-
BE34-8733E5K09F14 AND action = 'click';
```

Actualización de parámetros

UPDATE admite los siguientes valores como update_parameter:

- TTL: un valor de tiempo en segundos. El valor máximo configurable es de 630 720 000 segundos, lo que equivale a 20 años.
- TIMESTAMP: un valor bigint que representa el número de microsegundos desde la hora base estándar conocida como epoch: 1 de enero de 1970 a las 00:00:00 GMT. Una marca de tiempo en Amazon Keyspaces debe estar comprendida entre 2 días en el pasado y 5 minutos en el futuro.

Ejemplo

```
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

Utilice la instrucción DELETE para eliminar una fila en una tabla.

Sintaxis

```
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
```

Donde:

- *table_name* es la tabla que contiene la fila que desea eliminar.

Ejemplo

```
DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
 division='Executive' ;
```

`DELETE` admite el siguiente valor como `update_parameter`:

- `TIMESTAMP`: un valor `bigint` que representa el número de microsegundos desde la hora base estándar conocida como epoch: 1 de enero de 1970 a las 00:00:00 GMT.

Funciones integradas en Amazon Keyspaces

Amazon Keyspaces (para Apache Cassandra) admite diversas funciones integradas que puede utilizar en las instrucciones de Cassandra Query Language (CQL).

Temas

- Funciones escalares

Funciones escalares

Una función escalar realiza un cálculo sobre un único valor y devuelve el resultado como un único valor. Amazon Keyspaces soporta las siguientes funciones escalares.

Función	Descripción
`blobAsType`	Devuelve un valor del tipo de datos especific ado.
`cast`	Convierte un tipo de datos nativo en otro tipo de datos nativo.
`currentDate`	Devuelve la fecha/hora actual como fecha.
`currentTime`	Devuelve la fecha/hora actual como tiempo.
`currentTimestamp`	Devuelve la fecha/hora actual como marca de tiempo.
`currentTimeUUID`	Devuelve la fecha/hora actual como `timeuuid`.

Función	Descripción
`fromJson`	Convierte la cadena JSON en el tipo de datos de la columna seleccionada.
`maxTimeuuid`	Devuelve el `timeuuid` más grande posible para la marca de tiempo o la cadena de fecha.
`minTimeuuid`	Devuelve el `timeuuid` más pequeño posible para la marca de tiempo o la cadena de fecha.
`now`	Devuelve un nuevo `timeuuid` único. Se admite en las instrucciones `INSERT`, `UPDATE` y `DELETE` y como parte de la cláusula `WHERE` en instrucciones `SELECT`.
`toDate`	Convierte un `timeuuid` o una marca de tiempo en un tipo de fecha.
`toJson`	Devuelve el valor de la columna seleccionada en formato JSON.
`token`	Devuelve el valor hash de la clave de partición.
`toTimestamp`	Convierte un `timeuuid` o una fecha en una marca de tiempo.
`TTL`	Devuelve el tiempo de caducidad en segundos de una columna.
`typeAsBlob`	Convierte el tipo de datos especificado en un `blob`.
`toUnixTimestamp`	Convierte un `timeuuid` o una marca de tiempo en un `bigInt`.

Función	Descripción
`uuid`	Devuelve un UUID versión 4 aleatorio. Se admite en las instrucciones `INSERT`, `UPDATE` y `DELETE` y como parte de la cláusula `WHERE` en instrucciones `SELECT`.
`writetime`	Devuelve la marca de tiempo del valor de la columna especificada.
`dateOf`	(En desuso) Extrae la marca de tiempo de un `timeuuid` y devuelve el valor como fecha.
`unixTimestampOf`	(En desuso) Extrae la marca de tiempo de un `timeuuid` y devuelve el valor como una marca de tiempo entera de 64 bits sin procesar.

Cuotas para Amazon Keyspaces (para Apache Cassandra)

En esta sección se describen las cuotas actuales y los valores predeterminados de Amazon Keyspaces (para Apache Cassandra).

Temas

- Cuotas de servicio de Amazon Keyspaces
- Aumento o reducción del rendimiento (para las tablas aprovisionadas)
- Cifrado en reposo de Amazon Keyspaces

Cuotas de servicio de Amazon Keyspaces

En la tabla siguiente figuran las cuotas de Amazon Keyspaces (para Apache Cassandra) y los valores predeterminados. La información sobre cuotas que pueden ajustarse está disponible en la consola de Service Quotas, donde también puede solicitar aumentos de cuota. Para obtener más información sobre las cuotas, póngase en contacto con AWS Support.

Cuota	Descripción	Valores predeterminados de Amazon Keyspaces
Máximo de espacios de clave por Región de AWS	El número máximo de espacios de claves para este suscriptor por región. Puede ajustar este valor predeterminado en la consola Service Quotas.	256
Número máximo de tablas por Región de AWS	El número máximo de tablas en todos los espacios de claves para este suscriptor por región. Puede ajustar este valor predeterminado en la consola Service Quotas.	256

Cuota	Descripción	Valores predeterminados de Amazon Keyspaces
Tamaño máximo de esquema de tablas	El tamaño máximo de un esquema de tablas.	350 KB
Número máximo de operaciones DDL concurrentes	El número máximo de operaciones DDL concurrentes permitidas para este suscriptor por región.	50
Número máximo de consultas por conexión	El número máximo de consultas CQL que una única conexión TCP de cliente puede procesar por segundo.	3 000
Tamaño máximo de fila	El tamaño máximo de una fila, excluyendo los datos de columnas estáticas. Para obtener más detalles, consulte the section called “Cálculo del tamaño de las filas”.	1 MB
Número máximo de columnas en las instrucciones `INSERT` y `UPDATE`	El número máximo de columnas permitido en las instrucciones CQL `INSERT` o `UPDATE`. Una instrucción `INSERT` o `UPDATE` admite hasta 225 columnas normales cuando el periodo de vida (TTL) está desactivado. Si TTL está activado, se pueden modificar hasta 166 columnas normales en una única operación.	225/166

Cuota	Descripción	Valores predeterminados de Amazon Keyspaces
Número máximo de datos estáticos por partición lógica	El tamaño agregado máximo de datos estáticos en una partición lógica. Para obtener más detalles, consulte the section called “Cálculo de tamaño de columna estática por partición lógica”.	1 MB
Número máximo de subconsultas por instrucción `IN SELECT`	El número máximo de subconsultas que puede utilizar para la palabra clave `IN` en una instrucción `SELECT`. Puede ajustar este valor predeterminado en la consola Service Quotas.	100
Número máximo de colecciones anidadas congeladas por Región de AWS	El número máximo de colecciones anidadas admitidas cuando se utiliza la palabra clave `FROZEN` para una columna con un tipo de datos de colección. Para obtener más información sobre colecciones anidadas, consulte the section called “Tipos de colección”. Para aumentar el nivel de anidación, póngase en contacto con. AWS Support	5

Cuota	Descripción	Valores predeterminados de Amazon Keyspaces
Rendimiento de lectura máximo por segundo	El rendimiento de lectura máximo por segundo — unidades de solicitud de lectura (RRU) o unidades de capacidad de lectura (RCU)— que se puede asignar a una tabla por región. Puede ajustar este valor predeterminado en la consola Service Quotas.	40 000
Rendimiento de escritura máximo por segundo	El rendimiento de escritura máximo por segundo — unidades de solicitud de escritura (WRU) o unidades de capacidad de escritura (WCU)— que se puede asignar a una tabla por región. Puede ajustar este valor predeterminado en la consola Service Quotas.	40 000
Rendimiento de lectura a nivel de cuenta (aprovisionado)	El número máximo de unidades de capacidad de lectura (RCU) agregadas a la cuenta por región. Esto solo se aplica a las tablas en el modo de capacidad de lectura/escritura aprovisio nada. Puede ajustar este valor predeterminado en la consola Service Quotas.	80 000

Cuota	Descripción	Valores predeterminados de Amazon Keyspaces
Rendimiento de escritura a nivel de cuenta (aprovisionado)	El número máximo de unidades de capacidad de escritura (WCU) agregadas a la cuenta por región. Esto solo se aplica a las tablas en el modo de capacidad de lectura/escritura aprovisionada. Puede ajustar este valor predeterminado en la consola Service Quotas.	80 000
Número máximo de objetivos escalables por región y cuenta	El número máximo de objetivos escalables para la cuenta por región. Una tabla de Amazon Keyspaces cuenta como un objetivo escalable si el escalado automático está habilitado para la capacidad de lectura y como otro objetivo escalable si el escalado automático está habilitado para la capacidad de escritura. Puede ajustar este valor predeterminado en la consola Service Quotas de Application Auto Scaling seleccionando objetivos escalables para Amazon Keyspaces.	1500

Cuota	Descripción	Valores predeterminados de Amazon Keyspaces
Tamaño máximo de clave de partición	Tamaño máximo de la clave de partición compuesta. Se añaden hasta 3 bytes de almacenamiento adicional al tamaño sin procesar de cada columna incluida en la clave de partición para los metadatos.	2048 bytes
Tamaño máx. de clave de clúster	El tamaño máximo combinado de todas las columnas de clúster. Se añaden hasta 4 bytes de almacenamiento adicional al tamaño sin procesar de cada columna de clúster para los metadatos.	850 bytes
El número máximo de restauraciones simultáneas de tablas con P oint-in-time Recovery (PITR)	El número máximo de restauraciones concurrentes de tablas mediante PITR por suscriptor es 4. Puede ajustar este valor predeterminado en la consola Service Quotas.	4
Cantidad máxima de datos restaurados mediante point-in-time la recuperación (PITR)	El tamaño máximo de los datos que se pueden restaurar mediante el PITR en un plazo de 24 horas. Puede ajustar este valor predeterminado en la consola Service Quotas.	5 TB

Aumento o reducción del rendimiento (para las tablas aprovisionadas)

Aumento del rendimiento aprovisionado

Puede aumentarlo `ReadCapacityUnits` o `WriteCapacityUnits` tantas veces como sea necesario utilizando la consola o la `ALTER TABLE` instrucción. La nueva configuración no surtirá efecto hasta que se haya completado la operación `ALTER TABLE`.

No puedes superar las cuotas por cuenta al añadir la capacidad aprovisionada. Además, puede aumentar la capacidad aprovisionada de sus tablas tanto como lo necesite. Para obtener más información sobre las cuotas por cuenta, consulte la sección anterior, the section called “Cuotas de servicio de Amazon Keyspaces”.

Reducción de rendimiento aprovisionado

Para cada tabla, en una instrucción `ALTER TABLE` puede disminuir `ReadCapacityUnits` o `WriteCapacityUnits` (o ambas). La nueva configuración no surtirá efecto hasta que se haya completado la operación `ALTER TABLE`.

Se puede reducir la capacidad hasta cuatro veces al día. Un día se define según la hora universal coordinada (UTC). Además, si no ha habido ninguna reducción en la última hora, se permite una reducción adicional. Esto eleva de manera efectiva el número máximo de disminuciones en un día a 27 (4 disminuciones en la primera hora y 1 disminución para cada periodo subsecuente de 1 hora en un día).

Cifrado en reposo de Amazon Keyspaces

Puede cambiar las opciones de cifrado entre una AWS AWS KMS clave propia y una AWS KMS clave administrada por el cliente hasta cuatro veces en un período de 24 horas, por tabla, a partir del momento en que se creó la tabla. En caso de no producirse ningún cambio en las últimas seis horas, se permite un cambio adicional. Esto eleva efectivamente el número máximo de cambios en un día a 8 (4 cambios en las primeras 6 horas, y un cambio por cada periodo subsecuente de 6 horas en un día).

Puede cambiar la opción de cifrado para utilizar una AWS AWS KMS clave propia tantas veces como sea necesario, incluso si se ha agotado la cuota anterior.

Estas son las cuotas a menos que solicite una cantidad mayor. Para solicitar un aumento de la cuota de servicio, consulte AWS Support.

Historial de documentos de Amazon Keyspaces (para Apache Cassandra)

En la siguiente tabla se describen los cambios importantes realizados en la documentación desde la última versión de Amazon Keyspaces (para Apache Cassandra). Para recibir notificaciones sobre los cambios en esta documentación, puede suscribirse a una fuente RSS.

- Última actualización de la documentación: 7 de febrero de 2024

Cambio	Descripción	Fecha
Conéctese a Amazon Keyspaces desde Amazon Elastic Kubernetes Service	Ahora puede seguir un step-by-step tutorial para conectarse a Amazon Keyspaces desde Amazon EKS.	7 de febrero de 2024
API de escalado automático de Amazon Keyspaces para tablas aprovisionadas	Amazon Keyspaces ahora ofrece CQL compatibilidad con AWS API para configurar el escalado automático con el modo de capacidad aprovisionada.	23 de enero de 2024
Soporte de replicación multirregional de Amazon Keyspaces para tablas aprovisionadas	Amazon Keyspaces ahora admite el modo de capacidad aprovisionada para tablas de varias regiones.	23 de enero de 2024
La actividad de DML de Amazon Keyspaces está incluida en los registros CloudTrail	Ahora puede auditar las llamadas a la API del lenguaje de manipulación de datos (DML) de Amazon Keyspaces en AWS CloudTrail.	20 de diciembre de 2023
Amazon Keyspaces admite la palabra clave FROZEN	Amazon Keyspaces admite ahora la palabra clave FROZEN	15 de noviembre de 2023

	para los tipos de datos de colección.	
Actualización de la política administrada de Amazon Keyspaces	Amazon Keyspaces ha añadido nuevos permisos a la política administrada `AmazonKeyspacesFullAccess` para permitir que los clientes que se conecten a Amazon Keyspaces a través de puntos de conexión de VPC de interfaz accedan a la instancia de Amazon EC2 para actualizar la tabla `system.peers` de Amazon Keyspaces con información de red de la VPC.	3 de octubre de 2023
Actualización de la política administrada de Amazon Keyspaces	Amazon Keyspaces ha creado una nueva política administrada `AmazonKeyspacesReadOnlyAccess_v2` para permitir que los clientes que se conecten a Amazon Keyspaces a través de puntos de conexión de VPC de interfaz accedan a la instancia de Amazon EC2 para actualizar la tabla `system.peers` de Amazon Keyspaces con información de red de la VPC.	12 de septiembre de 2023

Prácticas recomendadas para crear conexiones en Amazon Keyspaces	Obtenga información sobre cómo mejorar y optimizar las configuraciones del controlador de cliente en Amazon Keyspaces.	30 de junio de 2023
Los espacios de claves del sistema ahora se documentan para Amazon Keyspaces	Conozca qué se almacena en los espacios de claves del sistema y cómo consultarlos para obtener información útil en Amazon Keyspaces.	21 de junio de 2023
Amazon Keyspaces admite ahora la réplica multirregión	La réplica multirregión de Amazon Keyspaces le permite mantener aplicaciones distribuidas globalmente al proporcionarle una mayor tolerancia a errores, estabilidad y capacidad de recuperación.	5 de junio de 2023
Actualización de la política administrada de Amazon Keyspaces	Amazon Keyspaces ha añadido nuevos permisos a la política administrada de `AmazonKeyspacesFullAccess` para permitir a Amazon Keyspaces crear un rol vinculado a servicios cuando un administrador crea un espacio de claves multirregión.	5 de junio de 2023
Amazon Keyspaces admite la palabra clave `IN`	Amazon Keyspaces ahora admite la palabra clave `IN` en las instrucciones `SELECT`.	25 de abril de 2023

Acceso entre cuentas de Amazon Keyspaces y puntos de conexión de VPC de interfaz	Obtenga información sobre cómo implementar el acceso entre cuentas para Amazon Keyspaces con puntos de conexión de VPC.	20 de abril de 2023
Amazon Keyspaces admite marcas de tiempo del lado del cliente	Las marcas de tiempo del lado del cliente de Amazon Keyspaces son marcas de tiempo a nivel de celda compatibles con Cassandra que ayudan a las aplicaciones distribuidas a determinar el orden de las operaciones de escritura cuando diferentes clientes realizan cambios en los mismos datos.	14 de marzo de 2023
Introducción a Amazon Keyspaces y los puntos de conexión de VPC de interfaz	En este step-by-step tutorial, aprenda a conectarse a Amazon Keyspaces desde una VPC.	1 de marzo de 2023
Optimización de costos de las tablas de Amazon Keyspaces	Hay a su disposición prácticas recomendadas y orientaciones que le ayudarán a identific ar estrategias para optimizar los costos de sus tablas de Amazon Keyspaces existente s.	17 de febrero de 2023
`Murmur3Partitioner` es ahora el particionador predeterminado	`Murmur3Partitioner` es ahora el particionador predeterminado en Amazon Keyspaces.	17 de noviembre de 2022

Amazon Keyspaces ahora admite `Murmur3Partitioner`	`Murmur3Partitioner` ya está disponible en Amazon Keyspaces.	9 de noviembre de 2022
Actualización de compatibilidad para cadenas vacías y valores blob	Amazon Keyspaces ahora también admite cadenas vacías y valores blob como valores de columna de agrupación.	19 de octubre de 2022
Amazon Keyspaces ya está disponible en AWS GovCloud (US)	Amazon Keyspaces ya está disponible en el FedRAMP High AWS GovCloud (US) Region y cumple con las normas altas de FedRAMP. Para obtener más información sobre los puntos de conexión disponibles, consulte Puntos de conexión FIPS en la AWS GovCloud (US) Region.	4 de agosto de 2022
Supervisa los costos de almacenamiento de tablas de Amazon Keyspaces con Amazon CloudWatch	Amazon Keyspaces ahora le ayuda a supervisar y realizar un seguimiento de los costes de almacenamiento de las tablas a lo largo del tiempo con la `BillableTableSizeInBytes` CloudWatch métrica.	14 de junio de 2022
Amazon Keyspaces ahora admite Terraform	Ahora puede utilizar Terraform para realizar operaciones de lenguaje de definición de datos (DDL) en Amazon Keyspaces.	9 de junio de 2022

Compatibilidad con la función `token` de Amazon Keyspaces	Amazon Keyspaces ahora le ayuda a optimizar las consultas de las aplicaciones mediante la función `token`.	19 de abril de 2022
Integración de Amazon Keyspaces con Apache Spark	Amazon Keyspaces ahora le permite leer y escribir datos en Apache Spark con mayor facilidad mediante el Spark Cassandra Connector de código abierto.	19 de abril de 2022
Referencia de la API de Amazon Keyspaces	Amazon Keyspaces admite las operaciones del plano de control para gestionar los espacios clave y las tablas mediante el AWS SDK y. AWS CLI La guía de referencia de la API describe en detalle las operaciones de plano de control admitidas.	2 de marzo de 2022
Solución de problemas de configuración comunes al utilizar Amazon Keyspaces.	Obtenga más informaci ón sobre cómo solucionar problemas de configuración comunes que podría encontrar al utilizar Amazon Keyspaces.	22 de noviembre de 2021
Compatibilidad de Amazon Keyspaces con Periodo de vida (TTL).	Periodo de vida (TTL) de Amazon Keyspaces le ayuda a simplificar la lógica de su aplicación y a optimizar el precio de almacenamiento al expirar automáticamente los datos de las tablas.	18 de octubre de 2021

Migración de datos a Amazon Keyspaces mediante DSBulk.	Un tep-by-step tutorial para migrar datos de Apache Cassandra a Amazon Keyspaces mediante DataStax Bulk Loader (DSBulk).	9 de agosto de 2021
Amazon Keyspaces admite las entradas de puntos de conexión de VPC en la tabla `system.peers` .	Amazon Keyspaces le permite rellenar la tabla `system.peers` con información del punto de conexión de VPC de interfaz disponible para mejorar el equilibrio de carga y aumentar el rendimiento de lectura/escritura.	29 de julio de 2021
Actualización de las políticas administradas de IAM para admitir las claves administradas por los clientes. AWS KMS	Las políticas gestionadas por IAM para Amazon Keyspaces ahora incluyen permisos para enumerar y ver las claves AWS KMS gestionadas por los clientes disponibles almacenadas en ellas. AWS KMS	1 de junio de 2021
Soporte de Amazon Keyspaces para claves administradas AWS KMS por el cliente.	Amazon Keyspaces le permite tomar el control de AWS KMS las claves gestionadas por los clientes almacenadas AWS KMS para su cifrado en reposo.	1 de junio de 2021
Amazon Keyspaces admite la sintaxis JSON.	Amazon Keyspaces le ayuda a leer y escribir documentos JSON con mayor facilidad al admitir la sintaxis JSON para las operaciones INSERT y SELECT.	21 de enero de 2021

Amazon Keyspaces admite columnas estáticas	Amazon Keyspaces ahora le ayuda a actualizar y almacenar datos comunes entre varias filas de forma eficiente mediante el uso de columnas estáticas.	9 de noviembre de 2020
Compatibilidad de la versión de disponibilidad pública de NoSQL Workbench con Amazon Keyspaces	NoSQL Workbench es una aplicación del lado del cliente que le ayuda a diseñar y visualizar modelos de datos no relacionales para Amazon Keyspaces con mayor facilidad. Los clientes de NoSQL Workbench están disponibles para Windows, macOS y Linux.	28 de octubre de 2020
Compatibilidad de la versión inicial de NoSQL Workbench con Amazon Keyspaces	NoSQL Workbench es una aplicación del lado del cliente que le ayuda a diseñar y visualizar modelos de datos no relacionales para Amazon Keyspaces con mayor facilidad. Los clientes de NoSQL Workbench están disponibles para Windows, macOS y Linux.	5 de octubre de 2020

Nuevos ejemplos de código para el acceso mediante programación a Amazon Keyspaces	Seguimos añadiendo ejemplos de código para el acceso programático a Amazon Keyspaces. Ya hay disponibles ejemplos de controladores Java, Python, Go, C# y Perl para Cassandra que admiten la versión 3.11.2 de Apache Cassandra.	17 de julio de 2020
point-in-time Recuperación de Amazon Keyspaces (PITR)	Amazon Keyspaces ahora ofrece point-in-time recuperación (PITR) para ayudar a proteger sus tablas de operaciones accidentales de escritura o eliminación, ya que le proporciona copias de seguridad continuas de los datos de las tablas.	9 de julio de 2020
Disponibilidad general de Amazon Keyspaces	Con Amazon Keyspaces, conocido anteriormente durante la versión inicial como Amazon Managed Apache Cassandra Service (MCS), puede utilizar el código Cassandra Query Language (CQL), los controladores de Cassandra con licencia Apache 2.0 y las herramientas para desarrolladores que ya utiliza actualmente.	23 de abril de 2020

Escalado automático de Amazon Keyspaces	Amazon Keyspaces (para Apache Cassandra) se integra con Application Auto Scaling para ayudarle a aprovisionar capacidad de rendimiento de manera eficiente para cargas de trabajo variables en respuesta al tráfico real de la aplicación ajustando automáticamente la capacidad de rendimiento.	23 de abril de 2020
Puntos de conexión de nube privada virtual (VPC) de interfaz para Amazon Keyspaces	Amazon Keyspaces ofrece comunicación privada entre el servicio y su VPC para que el tráfico de red no salga de la red de Amazon.	16 de abril de 2020
Políticas de acceso basadas en etiquetas	Ahora puede utilizar etiquetas de recursos en las políticas de IAM para administrar el acceso a Amazon Keyspaces.	8 de abril de 2020
Tipo de datos de contador	Amazon Keyspaces le ayuda ahora a coordinar los incrementos y decrementos de los valores de las columnas mediante el uso de contadores.	7 de abril de 2020
Etiquetado de recursos	Amazon Keyspaces ahora le permite etiquetar y clasificar los recursos mediante etiquetas.	31 de marzo de 2020

AWS CloudFormation soporte	Amazon Keyspaces ahora le ayuda a automatizar la creación y administración de recursos con el uso de AWS CloudFormation.	25 de marzo de 2020
Compatibilidad de roles y políticas de IAM y autenticación SigV4	Se agregó información sobre cómo puede usar AWS Identity and Access Management (IAM) para administrar los permisos de acceso e implementar políticas de seguridad para Amazon Keyspaces y cómo usar el complemento de autenticación del controlador Java para DataStax que Cassandra acceda mediante programación a Amazon Keyspaces mediante roles de IAM e identidades federadas.	17 de marzo de 2020
Modo de capacidad de lectura/escritura	Amazon Keyspaces ahora admite dos modos de capacidad de rendimiento de lectura/escritura. El modo de capacidad de lectura/escritura controla cómo se le cobra por el rendimiento de lectura y escritura y cómo se administra la capacidad de rendimiento de las tablas.	20 de febrero de 2020
Versión inicial	Esta documentación cubre la versión inicial de Amazon Keyspaces (para Apache Cassandra).	3 de diciembre de 2019

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.