



Guía para desarrolladores

Amazon Keyspaces (para Apache Cassandra)



Amazon Keyspaces (para Apache Cassandra): Guía para desarrolladores

Copyright © 2026 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
Funcionamiento	2
High-level arquitectura	2
Modelo de datos de Cassandra	5
Acceso a Amazon Keyspaces	6
Casos de uso	7
¿Qué es CQL?	8
Comparación de Amazon Keyspaces con Cassandra	9
Diferencias funcionales con Apache Cassandra	10
API, operaciones y tipos de datos de Apache Cassandra	11
Creación y eliminación asíncronas de espacios de claves y tablas	11
Autenticación y autorización	11
Lote	11
Captura de datos de cambio (CDC)	12
Configuración del clúster	12
Conexiones	12
Palabra clave IN	13
Colecciones FROZEN	13
Transacciones ligeras	14
Equilibrio de carga	14
Paginación	14
Particionadores	15
Instrucciones preparadas	15
Eliminación de rangos	15
Tablas del sistema	16
Marcas de tiempo	16
User-defined tipos (UDT)	17
API, operaciones, funciones y tipos de datos de Cassandra admitidos	17
Compatibilidad con la API de Cassandra	18
Compatibilidad con la API de plano de control de Cassandra	19
Compatibilidad con la API de plano de datos de Cassandra	20
Compatibilidad con las funciones de Cassandra	20
Compatibilidad con tipos de datos de Cassandra	21
Niveles de coherencia de Cassandra admitidos	22

Niveles de coherencia de escritura	23
Niveles de coherencia de lectura	24
Niveles de coherencia no admitidos	24
Migración a Amazon Keyspaces	26
Migración desde Cassandra	27
Compatibilidad	28
Estimación de precios	29
Estrategia de migración	38
Migración en línea	39
Migración sin conexión	51
Migración híbrida	54
Herramientas de migración	58
Carga de datos con cqlsh	60
Carga de datos con DSBulk	73
Acceso a Amazon Keyspaces	91
Configuración AWS Identity and Access Management	91
Inscríbase para obtener una Cuenta de AWS	91
Configuración de Amazon Keyspaces	91
Uso de la consola	92
CloudShell integración de consolas	93
Usando AWS CloudShell	95
AWS CloudShell opciones de integración	95
Requisitos previos	95
¿Cómo empezar con AWS CloudShell	96
Obtener permisos de IAM para AWS CloudShell	96
Instalación del cliente cqlsh	97
Configuración de certificados TLS	97
Conexión a Amazon Keyspaces	97
Uso de CQLSH con Amazon Keyspaces	98
Datos persistentes en AWS CloudShell	98
Prácticas recomendadas	98
Resolución de problemas	99
Ejemplo: Cómo interactuar con Amazon Keyspaces mediante AWS CloudShell	99
Creación de credenciales de acceso programático	100
Creación de credenciales específicas del servicio	102
Cree credenciales de IAM para la autenticación AWS	104

Puntos de conexión de servicio	113
Puertos y protocolos	113
Puntos de enlace globales	114
AWS GovCloud (US) Region Puntos de conexión de FIPS	118
Puntos de conexión de regiones de China	119
Puntos de enlace de Streams	120
Conexión a puntos finales de doble pila	123
Compatibilidad con IPv6 en Amazon Keyspaces	124
Resolución de DNS en Amazon Keyspaces	124
Uso de <code>cqlsh</code>	125
Uso de <code>cqlsh-expansion</code>	126
Configuración manual de conexiones de <code>cqlsh</code> para TLS	131
Usando el AWS CLI	133
Descargar y configurar el AWS CLI	133
Uso de AWS CLI con Amazon Keyspaces	133
Uso de la API	138
Utilización de un controlador de cliente de Cassandra	138
Uso de un controlador de cliente Java de Cassandra	139
Uso de un controlador de cliente Python para Cassandra	157
Uso de un controlador de cliente Cassandra Node.js	161
Uso de un controlador .NET Core para Cassandra	166
Uso de un controlador de cliente Go para Cassandra	167
Uso de un controlador de cliente Perl para Cassandra	173
Configuración del acceso entre cuentas	175
Configuración del acceso entre cuentas en una VPC compartida	176
Configuración del acceso entre cuentas sin una VPC compartida	180
Introducción	185
Requisitos previos	186
Creación de un espacio de claves	187
Comprobación del estado de creación del espacio de claves	190
Creación de una tabla	191
Comprobación del estado de creación de las tablas	199
Operaciones CRUD	200
Crear	201
Lectura	204
Actualización	209

Eliminación	210
Eliminación de una tabla	212
Eliminación de un espacio de claves	215
Tutoriales y soluciones	219
Conexión con puntos de conexión de VPC	219
Requisitos previos	220
Paso 1: lanzar una instancia de Amazon EC2	220
Paso 2: configurar la instancia de Amazon EC2	222
Paso 3: crear un punto de conexión de VPC para Amazon Keyspaces	225
Paso 4: configurar los permisos para la conexión del punto de conexión de VPC	231
Paso 5: configurar la supervisión	235
Paso 6: (opcional) prácticas recomendadas para las conexiones	236
Paso 7: (opcional) sanear	239
Integración con Apache Spark	240
Requisitos previos	241
Paso 1: Configurar Amazon Keyspaces	242
Paso 2: Configurar Apache Cassandra Spark Connector	243
Paso 3: Crear el archivo de configuración de la aplicación	245
Paso 4: Preparar los datos de origen y la tabla de destino	248
Paso 5: Escribir y leer datos en Amazon Keyspaces	250
Resolución de problemas	253
Conexión desde Amazon EKS	254
Requisitos previos	255
Paso 1: configuración del clúster de Amazon EKS	258
Paso 2: configuración de la aplicación	263
Paso 3: creación de la imagen de la aplicación	266
Paso 4: implementación de la aplicación en Amazon EKS	267
Paso 5: (opcional) eliminación	273
Exportación de datos a Amazon S3	275
Requisitos previos	276
Paso 1: Crear el bucket de Amazon S3, descargar las herramientas y configurar el entorno	277
Paso 2: Configurar el trabajo AWS Glue	280
Paso 3: Ejecute el AWS Glue trabajo de exportación desde AWS CLI	283
Paso 4: (Opcional) Programe el trabajo de exportación	284
Paso 5: (opcional) eliminación	285

Administración de recursos sin servidor	288
Estimación del tamaño de las filas	289
Calcule el tamaño codificado de las columnas	290
Calcule el tamaño codificado de los valores de datos en función del tipo de datos	291
Tenga en cuenta el impacto de las funciones de Amazon Keyspaces en el tamaño de las filas	293
Elija la fórmula correcta para calcular el tamaño codificado de una fila	293
Ejemplo de cálculo del tamaño de fila	294
Estimación del consumo de capacidad	296
Estimación del consumo de capacidad de las consultas de rango	297
Estimación del consumo de capacidad de lectura de las consultas de límites	298
Estimación del consumo de capacidad de lectura de los análisis de tablas	299
Estimación del consumo de capacidad de LWT	299
Estimación del consumo de capacidad de las columnas estáticas	300
Estimación de la capacidad para una tabla multirregión	305
Calcule el consumo de capacidad con CloudWatch	306
Configure los modos read/write de capacidad	307
Configuración del modo de capacidad bajo demanda	307
Configure el modo de capacidad aprovisionada	310
Visualización del modo de capacidad de una tabla	312
Cambio del modo de capacidad	314
Configurar el precalentamiento	316
Administración de la capacidad de rendimiento con escalado automático	331
Funcionamiento del escalado automático de Amazon Keyspaces	332
Funcionamiento del escalado automático para tablas multirregión	333
Notas de uso	334
Configuración y actualización de las políticas de escalado automático	335
Uso de la capacidad de ampliación	350
Uso de las características de Amazon Keyspaces	351
Espacios de claves de sistema	352
system	353
system_schema	355
system_schema_mcs	356
system_multiregion_info	359
User-defined tipos (UDT)	362
Configuración de permisos	363

Cree una UDT	366
Ver los UDT	370
Eliminar una UDT	374
Uso de consultas CQL	375
Uso de IN SELECT	376
Utilice declaraciones por lotes	380
Ordenación de resultados	386
Paginación de resultados	387
Trabajando con las transmisiones de los CDC	388
Cómo funcionan las transmisiones de los CDC	389
Utilice las transmisiones de los CDC	393
Uso de particionadores	428
Cambio de particionador	428
Client-side marcas de tiempo	430
Integración con los servicios AWS	432
Creación de una tabla con marcas de tiempo del cliente	432
Configuración de marcas de tiempo del cliente	435
Uso de marcas de tiempo del cliente en consultas	438
Multi-Region replicación	439
Ventajas	440
Modos de capacidad y precios	441
Funcionamiento	441
Notas de uso	447
Configure la replicación multirregional	449
Copias de seguridad y restauración con recuperación en un momento dado	478
Funcionamiento	479
Uso de la recuperación en un momento dado	485
Caducidad de datos con período de vida	498
Integración con los servicios AWS	500
Creación de una tabla con el valor de TTL predeterminado	500
Actualización del valor de TTL predeterminado de la tabla	504
Creación de una tabla con un TTL personalizado	508
Actualización de una tabla con un TTL personalizado	510
Uso de INSERT para establecer un TTL personalizado para nuevas filas	513
Uso de UPDATE para establecer TTL personalizados para filas y columnas	514
¿Trabajando con AWS los SDK	515

Trabajo con etiquetas	516
Restricciones de etiquetado	517
Etiquete los espacios clave, las tablas y las secuencias	518
Creación de informes de asignación de costos	536
Cree recursos CloudFormation	538
Amazon Keyspaces y plantillas CloudFormation	538
Obtenga más información sobre CloudFormation	538
NoSQL Workbench	539
Download	540
Introducción	540
Visualización de un modelo de datos	542
Creación de un modelo de datos	546
Edición de un modelo de datos	549
Confirmación de un modelo de datos	551
Modelos de datos de ejemplo	562
Historial de versiones	563
Ejemplos de código	565
Conceptos básicos	566
Introducción a Amazon Keyspaces	566
Conceptos básicos	571
Acciones	634
Bibliotecas y herramientas	688
Bibliotecas y ejemplos	688
Kit de herramientas para desarrolladores de Amazon Keyspaces (para Apache Cassandra)	688
Ejemplos de Amazon Keyspaces (para Apache Cassandra)	688
AWS Plugins de autenticación Signature versión 4 (SigV4)	689
Repositorios destacados de ejemplos y herramientas para desarrolladores	689
Búferes de protocolo de Amazon Keyspaces	689
CloudFormation plantilla para crear el CloudWatch panel de control de Amazon para las métricas de Amazon Keyspaces (para Apache Cassandra)	690
Uso de Amazon Keyspaces (para Apache Cassandra) con AWS Lambda	690
Uso de Amazon Keyspaces (para Apache Cassandra) con Spring	690
Uso de Amazon Keyspaces (para Apache Cassandra) con Scala	690
Uso de Amazon Keyspaces (para Apache Cassandra) con AWS Glue	691

Amazon Keyspaces (para Apache Cassandra) Lenguaje de consulta Cassandra (CQL) a conversor CloudFormation	691
Asistentes de Amazon Keyspaces (para Apache Cassandra) para el controlador de Apache Cassandra para Java	691
Demostración de compresión rápida de Amazon Keyspaces (para Apache Cassandra)	691
Demostración del códec de Amazon S3 y Amazon Keyspaces (para Apache Cassandra) ...	691
Prácticas recomendadas	692
Diseño NoSQL	693
NoSQL versus RDBMS	694
Dos conceptos clave	694
Enfoque general	695
Conexiones	696
Cómo funcionan	697
Configuración de conexiones	698
Configuración de las políticas de reintentos	700
Conexiones de puntos de conexión de VPC	700
Monitoreo de conexiones	702
Gestión de errores de conexión	703
Modelado de datos	703
Diseño de claves de partición	704
Optimización de costos	707
Evaluación de los costos en el nivel de tabla	707
Evaluación del modo de capacidad de una tabla	709
Evaluación de los ajustes de Application Auto Scaling de su tabla	714
Identificación de recursos no utilizados	721
Evaluación de los patrones de uso de las tablas	727
Evaluación de la capacidad aprovisionada para lograr un aprovisionamiento del tamaño adecuado	728
Resolución de problemas	738
Errores generales	739
Errores generales	739
Errores de conexión	741
Errores al conectarse a un punto de conexión de Amazon Keyspaces	741
Errores de administración de capacidad	754
Errores de capacidad sin servidor	754
Errores de lenguaje de definición de datos	759

Errores de lenguaje de definición de datos	759
Monitoreo de Amazon Keyspaces	765
Monitorización con CloudWatch	766
Uso de las métricas	767
Métricas y dimensiones	768
Ver las métricas de	793
Creación de alarmas	794
Iniciar sesión con CloudTrail	794
Configuración de las entradas del archivo de registro en CloudTrail	795
Información sobre el DDL en CloudTrail	796
Información sobre el DML en CloudTrail	797
Descripción de las entradas de los archivos de registro	798
Seguridad	810
Protección de datos	811
Cifrado en reposo	812
Cifrado en tránsito	834
Privacidad del tráfico entre redes	834
AWS Identity and Access Management	835
Público	836
Autenticación con identidades	836
Administración del acceso con políticas	837
Funcionamiento de Amazon Keyspaces con IAM	839
Identity-based ejemplos de políticas	845
AWS políticas gestionadas	853
Resolución de problemas	865
Cómo utilizar roles vinculados a servicios	869
Validación de conformidad	878
Resiliencia	879
Seguridad de la infraestructura	880
Uso de los puntos de conexión de VPC de interfaz	881
Uso de puntos finales de VPC de interfaz con transmisiones	888
Configuración y análisis de vulnerabilidades para Amazon Keyspaces	892
Prácticas recomendadas de seguridad	892
Prácticas recomendadas de seguridad preventivas	893
Prácticas recomendadas de detección de seguridad	894
Referencia del lenguaje CQL	897

Elementos del lenguaje	898
Identificadores	898
Constantes	898
Términos	899
Tipos de datos	899
Codificación JSON de los tipos de datos de Amazon Keyspaces	903
Instrucciones DDL	907
Keyspaces	908
Tablas	912
Tipos	927
Instrucciones DML	929
SELECT	930
INSERT	933
UPDATE	934
DELETE	936
Built-in funciones	937
Funciones escalares	937
Cuotas	939
Cuotas de servicio de Amazon Keyspaces	939
Aumento o reducción del rendimiento (para las tablas aprovisionadas)	947
Aumento del rendimiento aprovisionado	947
Reducción de rendimiento aprovisionado	947
Cifrado en reposo de Amazon Keyspaces	947
Cuotas y valores predeterminados para los tipos definidos por el usuario (UDT) en Amazon Keyspaces	948
Cuotas de UDT y valores predeterminados de Amazon Keyspaces	948
Historial de revisión	950
.....	cmlxv

¿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 solo unos clics en la consola de AWS administración o unas pocas líneas de código, puede crear espacios clave 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 con el mismo código de aplicación de Cassandra y las mismas herramientas de desarrollador que AWS utiliza actualmente.

Con la [calculadora de precios de Amazon Keyspaces \(para Apache Cassandra\)](#) disponible en Github, puede estimar los costes mensuales de Amazon Keyspaces en función de su carga de trabajo actual de Apache Cassandra. Introduzca las métricas del resultado de estado de Cassandra nodetool y de la configuración sin servidor prevista para Amazon Keyspaces a fin de comparar los costes directos entre las dos soluciones. Tenga en cuenta que esta calculadora se centra únicamente en los costes operativos de Amazon Keyspaces en comparación con su implementación actual de Cassandra. No incluye los factores del costo total de propiedad (TCO), como el mantenimiento de la infraestructura, los gastos operativos o los costos de soporte de Cassandra.

Para obtener una lista de puntos de enlace Regiones de AWS y disponibles, consulte [Puntos de enlace de servicio 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

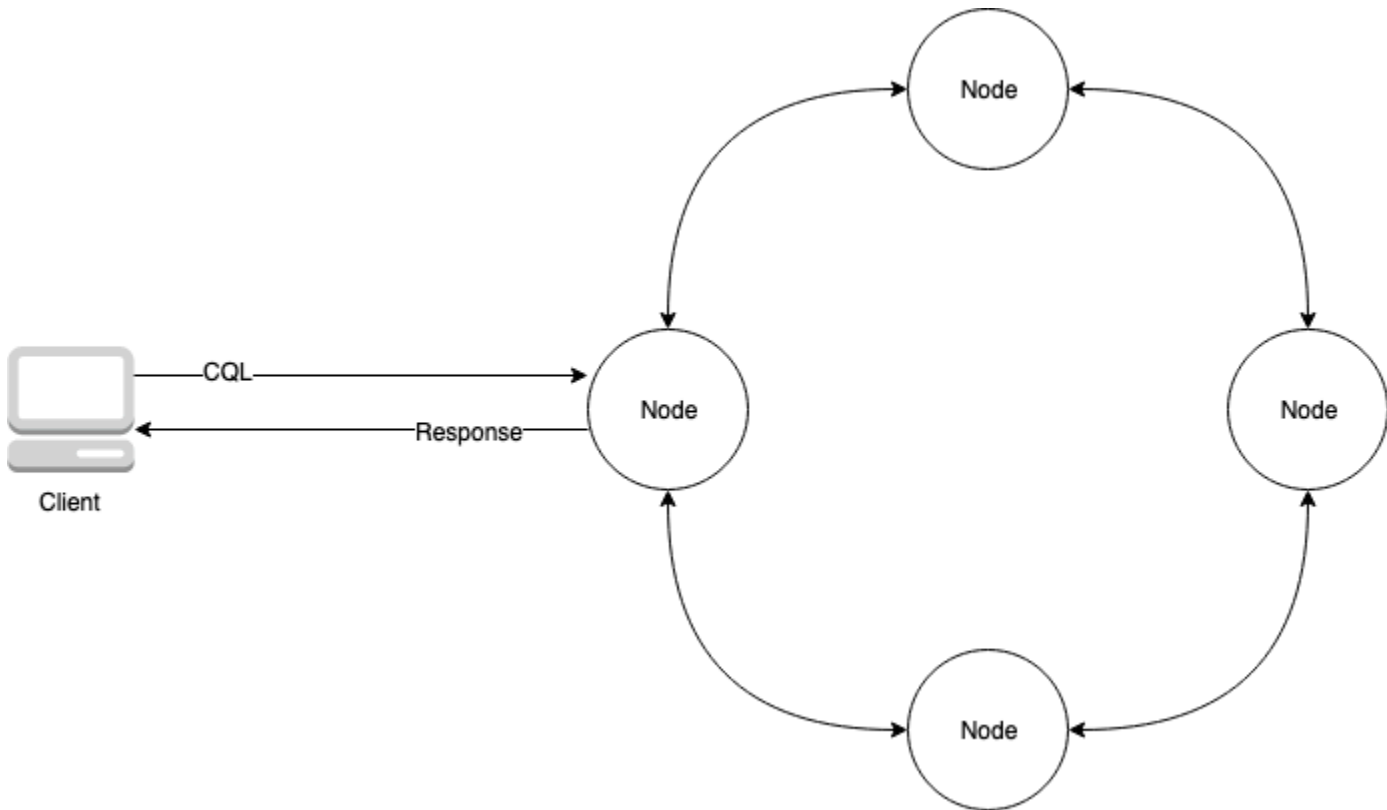
- [High-level arquitectura: Apache Cassandra frente a Amazon Keyspaces](#)
- [Modelo de datos de Cassandra](#)
- [Acceso a Amazon Keyspaces desde una aplicación](#)

High-level arquitectura: Apache Cassandra frente a 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.

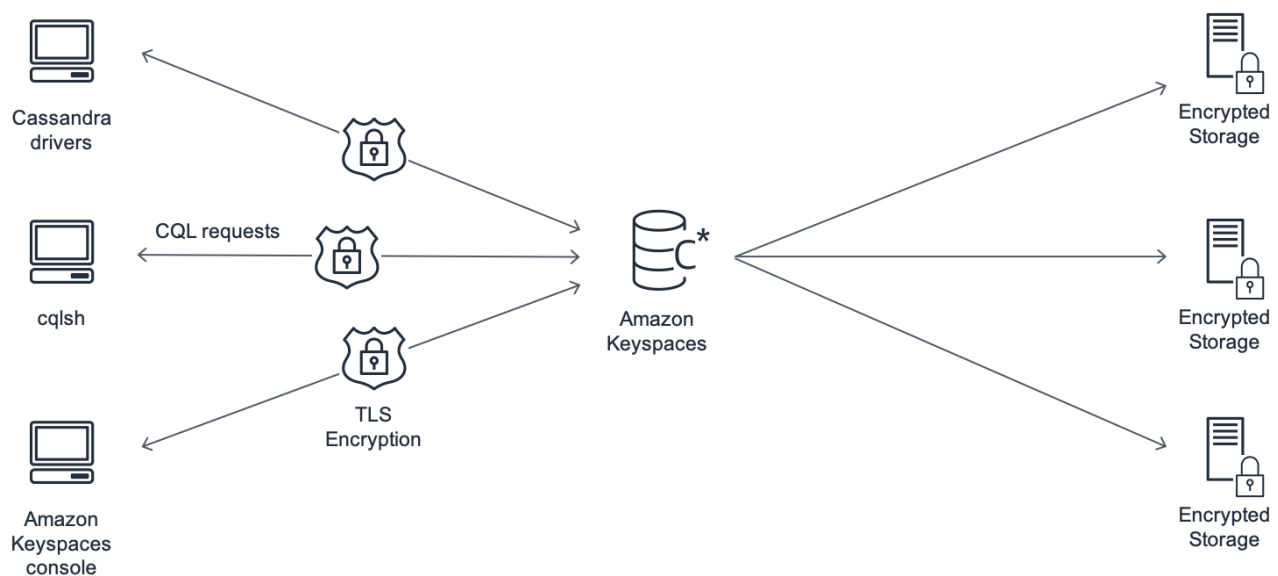
El modo de capacidad aprovisionada le ayuda a optimizar el precio del rendimiento si tiene un tráfico de aplicaciones predecible y puede prever con antelación los requisitos de capacidad de su tabla. Con el modo de capacidad aprovisionada, usted especifica el número de lecturas y escrituras por

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 “Configure los modos read/write de capacidad”](#).

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). Las funciones AWS de seguridad adicionales incluyen la [supervisión](#) y los puntos finales de [nube privada virtual \(VPC\)](#). [AWS Identity and Access Management](#) 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. Para ver las prácticas recomendadas, consulte [the section called “Diseño de claves de partición”](#).

Clave principal

En Cassandra, los datos se almacenan como pares clave-valor. 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. Usaría una clave de partición compuesta si una clave de partición de una sola columna diera como resultado

una sola partición o unas pocas particiones con la mayoría de los datos y, por lo tanto, con la mayoría de las operaciones del disco I/O .

- 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.

Para obtener más información sobre el diseño NoSQL y Amazon Keyspaces, consulte [the section called “Diseño NoSQL”](#). Para obtener más información sobre Amazon Keyspaces y el modelado de datos, consulte [the section called “Modelado de datos”](#).

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. Para obtener más información sobre las credenciales necesarias, consulte [the section called “Cree credenciales de IAM para la autenticación AWS”](#).

Note

Para ayudarle a empezar, puede encontrar ejemplos de código integrales 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](#)

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-1.amazonaws.com`, y el número de puerto por: 9142.
- Añadir la TLS/SSL configuración: para obtener más información sobre cómo añadir la TLS/SSL configuración para conectarse a Amazon Keyspaces mediante un controlador Python del cliente 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.
- Cree aplicaciones con tecnologías de código abierto: cree aplicaciones AWS con 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, PHP Node.js, 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, proteger y escalar las tablas de Cassandra Nube de AWS sin necesidad de 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 comunicarse con Apache Cassandra. Amazon Keyspaces (para Apache Cassandra) es compatible con la API 3.x de CQL (retrocompatible con la versión 2.x).

En CQL, los datos se almacenan en tablas, columnas y filas. En este sentido, CQL es similar al lenguaje Structured Query Language (SQL). A continuación, se describen los conceptos clave de CQL.

- Elementos de CQL: los elementos fundamentales de CQL son los identificadores, las constantes, los términos y los tipos de datos.
- Lenguaje de definición de datos (DDL): las sentencias DDL se utilizan para gestionar estructuras de datos, como los espacios clave y las tablas, que son AWS recursos de Amazon Keyspaces. Las sentencias DDL son operaciones del plano de control en AWS.
- Lenguaje de manipulación de datos (DML): las instrucciones de DML se utilizan para administrar los datos de las tablas. Las instrucciones DML se utilizan para seleccionar, insertar, actualizar y eliminar datos. Son operaciones del plano de datos en AWS.
- Built-in funciones: Amazon Keyspaces admite diversas funciones escalares integradas que puede utilizar en sentencias CQL.

Para obtener más información sobre CQL, consulte [Referencia del lenguaje CQL para Amazon Keyspaces \(para Apache Cassandra\)](#). Para analizar las diferencias funcionales con Apache Cassandra, consulte [the section called “Diferencias funcionales con Apache Cassandra”](#).

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

- Utilizar el editor CQL en la Consola de administración de AWS.
- [El uso AWS CloudShell y la expansión de `clash`](#).
- Utilizar un cliente `cqlsh`.
- Utilizar un controlador de cliente de Cassandra con licencia Apache 2.0.

Además de CQL, puede realizar operaciones de lenguaje de definición de datos (DDL) en Amazon Keyspaces mediante los SDK y el AWS . AWS Command Line Interface

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

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

Para establecer una conexión con Amazon Keyspaces, puede utilizar un [punto de conexión de servicio de AWS](#) público o un punto de conexión privado mediante [puntos de conexión de VPC de interfaz \(AWS PrivateLink\)](#) en [Amazon Virtual Private Cloud](#). Según el punto de conexión utilizado, Amazon Keyspaces puede aparecer ante el cliente de una de las siguientes formas.

AWS conexión de punto final de servicio

Se trata de una conexión establecida a través de cualquier [punto de conexión público](#). En este caso, Amazon Keyspaces le aparece al cliente como un clúster de Apache Cassandra 3.11.2 de nueve nodos.

Conexión con punto de conexión de VPC de interfaz

Se trata de una conexión privada que se establece mediante un [punto de conexión de VPC de interfaz](#). En este caso, Amazon Keyspaces le aparece al cliente como un clúster de Apache Cassandra 3.11.2 de tres nodos.

Independientemente del tipo de conexión y del número de nodos visibles para el cliente, Amazon Keyspaces ofrece un rendimiento y un almacenamiento prácticamente ilimitados. Para ello, Amazon Keyspaces asigna los nodos a equilibradores de carga que direccionan las consultas a una de las diversas particiones de almacenamiento subyacentes. Para obtener más información acerca de las conexiones, consulte [the section called “Cómo funcionan”](#).

Amazon Keyspaces almacena los datos en particiones. Una partición es una asignación de almacenamiento para una tabla basado en discos de estado sólido (SSD). Amazon Keyspaces replica automáticamente sus datos en varias [zonas de disponibilidad](#) dentro de una Región de AWS para garantizar una mayor durabilidad y alta disponibilidad. A medida que aumentan sus necesidades de rendimiento o almacenamiento, Amazon Keyspaces se encarga de la administración de particiones por usted y aprovisiona automáticamente las particiones adicionales necesarias.

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, en Amazon

Keyspaces no necesita usar las operaciones de la API del plano de control de Cassandra para administrar la configuración de clústeres y nodos.

Amazon Keyspaces configura automáticamente ajustes como el factor de réplica y el nivel de coherencia con el fin de ofrecerle alta disponibilidad, durabilidad y un rendimiento de milisegundos de un solo dígito. Para obtener aún más resiliencia y lecturas locales de baja latencia, Amazon Keyspaces ofrece [replicación multirregional](#).

Temas

- [Diferencias funcionales: Amazon Keyspaces vs. Apache Cassandra](#)
- [API, operaciones, funciones y tipos de datos de Cassandra admitidos](#)
- [Niveles de coherencia de lectura y escritura de Apache Cassandra admitidos y costos asociados](#)

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](#)
- [Captura de datos de cambio \(CDC\)](#)
- [Configuración del clúster](#)
- [Conexiones](#)
- [Palabra clave IN](#)
- [Colecciones FROZEN](#)
- [Transacciones ligeras](#)
- [Equilibrio de carga](#)
- [Paginación](#)
- [Particionadores](#)
- [Instrucciones preparadas](#)

- [Eliminación de rangos](#)
- [Tablas del sistema](#)
- [Marcas de tiempo](#)
- [User-defined tipos \(UDT\)](#)

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](#).

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 clave, tablas y tipos de forma asíncrona. Para obtener información sobre cómo monitorear el estado de creación de recursos, consulte [the section called “Comprobación del estado de creación del espacio de claves”](#) y [the section called “Comprobación del estado de creación de las 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 los siguientes comandos por lotes:

- Lote registrado: admite hasta 100 comandos por lote.
- Lote no registrado: admite hasta 30 comandos por lote.

Solo INSERTUPDATE, o DELETE los comandos están permitidos en un lote.

Captura de datos de cambio (CDC)

En Apache Cassandra, la captura de datos de cambios (CDC) copia los segmentos del registro de confirmación que capturan las actualizaciones de las tablas en un directorio específico del nodo. Cada registro de cambios del registro de los CDC puede incluir actualizaciones en varios espacios clave y tablas. Cuando se actualiza o elimina una fila, el registro de los CDC de Apache Cassandra muestra solo las columnas modificadas. El registro no incluye el estado anterior de los datos antes del cambio. Para comprender todos los detalles de las modificaciones de los datos, debe deserializar estos registros con herramientas específicas de Cassandra.

Con las transmisiones de CDC de Amazon Keyspaces, puede seleccionar el tipo de información que recopila la transmisión de CDC para cada fila. De forma predeterminada, las transmisiones CDC de Amazon Keyspaces capturan la versión de la fila anterior y posterior al cambio. Amazon Keyspaces CDC proporciona registros de cambios ordenados y deduplicados a nivel de tabla.

[En Amazon Keyspaces, cada transmisión de CDC es un AWS recurso con un nombre de recurso de Amazon \(ARN\) y su aplicación puede consumir las transmisiones de CDC de Amazon Keyspaces mediante la API de CDC de Amazon Keyspaces utilizando uno de los puntos de enlace disponibles.](#)

Para obtener más información sobre Amazon Keyspaces CDC, consulte [the section called “Trabajando con las transmisiones de los CDC”](#)

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 “Uso de **IN SELECT**”](#).

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 hasta 8 niveles de anidación para colecciones inmovilizadas de forma predeterminada. 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”](#).

En las tablas con columnas estáticas, tanto Apache Cassandra como Amazon Keyspaces establecen el valor de la columna estática de la partición al principio de cada página en una consulta de varias páginas. Cuando una tabla tiene filas de datos de gran tamaño, como resultado del comportamiento de paginación de Amazon Keyspaces, es más probable que el resultado de una operación de lectura de rango devuelva más páginas para Amazon Keyspaces que para Apache Cassandra. En consecuencia, existe una mayor probabilidad en Amazon Keyspaces de que las actualizaciones simultáneas de la columna estática puedan provocar que el valor de la columna estática sea diferente en las distintas páginas del conjunto de resultados de lectura del rango.

Particionadores

El particionador predeterminado en Amazon Keyspaces es el `Cassandra-compatible Murmur3Partitioner`. Además, tiene la opción de utilizar Amazon Keyspaces `DefaultPartitioner` o `Cassandra-compatible RandomPartitioner`.

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

Important

A partir del 1 de julio de 2026, las operaciones de eliminación de rangos pasarán del procesamiento sincrónico al asíncrono. Tras este cambio, una respuesta correcta confirmará que se ha aceptado la solicitud de eliminación, pero es posible que la eliminación siga en curso. Para comprobar que la eliminación se ha completado, consulta los elementos del rango especificado. Si la consulta no arroja ningún resultado, se completa la eliminación.

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.

Las eliminaciones de rangos no son aisladas. Las eliminaciones de filas individuales son visibles para otras operaciones mientras se está realizando un borrado de rango.

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. Para obtener más información, consulte [the section called “Espacios de claves de sistema”](#).

Read-only es necesario acceder a las tablas del sistema y puede controlarlo con las políticas de acceso de IAM. Para obtener más información, consulte [the section called “Administración del acceso con 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 de Cassandra Query Language (CQL) mediante 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 de tiempo

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 [the section called “Client-side marcas de tiempo”](#).

User-defined tipos (UDT)

El operador de desigualdad no es compatible con las UDT en Amazon Keyspaces.

Para obtener información sobre cómo trabajar con las UDT en Amazon Keyspaces, consulte. [the section called “User-defined tipos \(UDT\)”](#)

Para ver cuántas UDT se admiten por espacio de claves, los niveles de anidación compatibles y otros valores y cuotas predeterminados relacionados con las UDT, consulte. [the section called “Cuotas y valores predeterminados para los tipos definidos por el usuario \(UDT\) en Amazon Keyspaces”](#)

API, operaciones, funciones y tipos de datos de Cassandra admitidos

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	Sí
SELECT	Sí
INSERT	Sí
DELETE	Sí
UPDATE	Sí
USE	Sí
CREATE TYPE	Sí
ALTER TYPE	No
DROP TYPE	Sí
CREATE TRIGGER	No

Operación de la API	compatible
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.

Característica	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

Característica	Motivo
Configuración de la caché	No aplicable
Configuración de seguridad	Sustituido por IAM

Compatibilidad con la API de plano de datos de Cassandra

Característica	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 “Built-in funciones”](#).

Función	compatible
Funciones de Aggregate	No
Conversión de Blob	Sí
Cast	Sí
Funciones de Datetime	Sí
Funciones de conversión de tiempo	Sí
Funciones de TimeUuid	Sí
Token	Sí

Función	compatible
User defined functions (UDF)	No
Uuid	Sí

Compatibilidad con tipos de datos de Cassandra

En la siguiente tabla se enumeran los tipos de datos de Apache Cassandra compatibles con Amazon Keyspaces. Para obtener más información sobre los tipos de datos en Amazon Keyspaces, consulte [the section called “Tipos de datos”](#)

Tipo de datos:	compatible	
ascii	Sí	
bigint	Sí	
blob	Sí	
boolean	Sí	
counter	Sí	
date	Sí	
decimal	Sí	
double	Sí	
float	Sí	
frozen	Sí	
inet	Sí	
int	Sí	
list	Sí	

Tipo de datos:	compatible	
map	Sí	
set	Sí	
smallint	Sí	
text	Sí	
time	Sí	
timestamp	Sí	
timeuuid	Sí	
tinyint	Sí	
tuple	Sí	
user-defined types (UDTs)	Sí	
uuid	Sí	
varchar	Sí	
varint	Sí	

Niveles de coherencia de lectura y escritura de Apache Cassandra admitidos y costos asociados

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-1"))
    .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	On-demand
ONE	0,5 CRU	0,5 RRU
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

Migración a Amazon Keyspaces (para Apache Cassandra)

La migración a Amazon Keyspaces (para Apache Cassandra) presenta una serie de ventajas muy interesantes para las empresas y las organizaciones. Estas son algunas de las principales ventajas que convierten a Amazon Keyspaces en una opción atractiva para la migración.

- **Escalabilidad:** Amazon Keyspaces se ha diseñado para administrar cargas de trabajo masivas y escalar sin complicaciones para adaptarse al crecimiento de los volúmenes de datos y del tráfico. Con la versión de Cassandra tradicional, el escalado no se realiza bajo demanda y requiere una planificación para futuros picos. Amazon Keyspaces le permite ampliar o reducir fácilmente sus tablas en función de la demanda, lo que garantiza que sus aplicaciones puedan administrar picos repentinos de tráfico sin que afecte al rendimiento.
- **Rendimiento:** Amazon Keyspaces ofrece acceso a datos de baja latencia, lo que permite a las aplicaciones recuperar y procesar datos con una velocidad excepcional. Su arquitectura distribuida garantiza que las operaciones de lectura y escritura se distribuyan en varios nodos, lo que ofrece tiempos de respuesta uniformes en milisegundos de un solo dígito, incluso con altas tasas de solicitudes.
- **Totalmente administrado:** Amazon Keyspaces es un servicio totalmente administrado que ofrece AWS. Esto significa que AWS gestiona los aspectos operativos de la administración de bases de datos, incluidos el aprovisionamiento, la configuración, los parches, las copias de seguridad y el escalado. Esto le permite centrarse más en el desarrollo de sus aplicaciones y menos en las tareas de administración de bases de datos.
- **Arquitectura sin servidor:** Amazon Keyspaces es un servicio sin servidor. Solo paga por la capacidad consumida, sin necesidad de aprovisionar la capacidad por adelantado. No tendrá que administrar servidores ni seleccionar instancias. Este modelo de pago por solicitud ofrece rentabilidad y unos gastos operativos mínimos, ya que solo paga por los recursos que consume sin que sea necesario aprovisionar ni monitorear la capacidad.
- **Flexibilidad NoSQL con esquemas:** Amazon Keyspaces sigue un modelo de datos NoSQL, lo que proporciona flexibilidad en el diseño de esquemas. Amazon Keyspaces permite almacenar datos estructurados, semiestructurados y sin estructurar, lo que los hace adecuados para administrar tipos de datos diversos y en constante evolución. Además, Amazon Keyspaces realiza la validación del esquema durante la escritura, lo que permite una evolución centralizada del modelo de datos. Esta flexibilidad permite ciclos de desarrollo más rápidos y una adaptación más sencilla a los requisitos empresariales en constante cambio.

- Alta disponibilidad y durabilidad: Amazon Keyspaces replica los datos en varias [zonas de disponibilidad](#) dentro de una Región de AWS, lo que garantiza una alta disponibilidad y durabilidad de los datos. Además, gestiona automáticamente la replicación, la conmutación por error y la recuperación, lo que minimiza el riesgo de pérdida de datos o interrupciones en el servicio. Amazon Keyspaces ofrece un SLA de disponibilidad de hasta el 99,999 %. Para obtener aún más resiliencia y lecturas locales de baja latencia, Amazon Keyspaces ofrece [replicación multirregional](#).
- Seguridad y conformidad: Amazon Keyspaces se integra con AWS Identity and Access Management para controlar el acceso de forma precisa. Proporciona cifrado en reposo y en tránsito, lo que ayuda a garantizar la seguridad de sus datos. Auditores externos han evaluado Amazon Keyspaces en términos de seguridad y conformidad con programas específicos, como HIPAA, PCI DSS y SOC, lo que le permite cumplir los requisitos normativos. Para obtener más información, consulte [the section called “Validación de conformidad”](#).
- Integración con el AWS ecosistema: como parte del AWS ecosistema, Amazon Keyspaces se integra perfectamente con otros Servicios de AWS, por ejemplo CloudWatch, AWS CloudFormation Amazon y. AWS CloudTrail Esta integración le permite crear arquitecturas sin servidor, aprovechar la infraestructura como código y crear aplicaciones basadas en datos en tiempo real. Para obtener más información, consulte [Monitoreo de Amazon Keyspaces](#).

Temas

- [Creación de un plan de migración para migrar de Apache Cassandra a Amazon Keyspaces](#)
- [Selección de la herramienta adecuada para cargar o migrar datos por lotes a Amazon Keyspaces](#)

Creación de un plan de migración para migrar de Apache Cassandra a Amazon Keyspaces

Para que la migración de Apache Cassandra a Amazon Keyspaces se realice correctamente, recomendamos revisar los conceptos y las prácticas recomendadas de migración aplicables, así como comparar las opciones disponibles.

En este tema se describe cómo funciona el proceso de migración y se presentan varios conceptos clave y las herramientas y técnicas que tiene a su disposición. Puede evaluar las diferentes estrategias de migración para seleccionar la que mejor se adapte a sus necesidades.

Temas

- [Compatibilidad funcional](#)

- [Estimación de precios de Amazon Keyspaces](#)
- [Elección de una estrategia de migración](#)
- [Migración online a Amazon Keyspaces: estrategias y prácticas recomendadas](#)
- [Proceso de migración sin conexión: de Apache Cassandra a Amazon Keyspaces](#)
- [Uso de una solución de migración híbrida: de Apache Cassandra a Amazon Keyspaces](#)

Compatibilidad funcional

Analice detenidamente las diferencias funcionales entre Apache Cassandra y Amazon Keyspaces antes de la migración. 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.

Sin embargo, hay algunas API de Cassandra que no son compatibles con Amazon Keyspaces. Para obtener más información acerca de las API admitidas, consulte [the section called “API, operaciones, funciones y tipos de datos de Cassandra admitidos”](#). Para obtener información general sobre todas las diferencias funcionales entre Amazon Keyspaces y Apache Cassandra, consulte [the section called “Diferencias funcionales con Apache Cassandra”](#).

Para comparar las API y el esquema de Cassandra que está utilizando con las funciones compatibles en Amazon Keyspaces, puede ejecutar un script de compatibilidad disponible en el kit de herramientas de Amazon Keyspaces en. [GitHub](#)

Uso del script de compatibilidad

1. Descargue el script de Python de compatibilidad [GitHub](#) y muévelo a una ubicación que tenga acceso a su clúster de Apache Cassandra existente.
2. El script de compatibilidad usa parámetros similares a CQLSH. Para `--host` y `--port`, introduzca la dirección IP y el puerto que utilice para conectarse y ejecutar consultas a uno de los nodos de Cassandra del clúster.

Si su clúster de Cassandra usa autenticación, también debe proporcionar `-username` y `-password`. Para utilizar el script de compatibilidad, puede ejecutar el siguiente comando.

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

Estimación de precios de Amazon Keyspaces

En esta sección se proporciona información general de la información que debe recopilar de las tablas de Apache Cassandra para calcular el costo estimado de Amazon Keyspaces. Cada una de sus tablas requiere diferentes tipos de datos, debe admitir diferentes consultas de CQL y mantiene un tráfico distintivo read/write .

[Pensar en sus requisitos en función de las tablas se alinea con los modos de capacidad de rendimiento y aislamiento read/write de recursos a nivel de tabla de Amazon Keyspaces.](#) Con Amazon Keyspaces, puede definir las [políticas de read/write capacidad y escalado automático para las](#) tablas de forma independiente.

Comprender los requisitos de las tablas le ayuda a priorizarlas para la migración según la funcionalidad, el costo y el esfuerzo de migración.

Recopile las siguientes métricas de la tabla de Cassandra antes de una migración. Esta información ayuda a estimar el costo de la carga de trabajo en Amazon Keyspaces.

- Nombre de la tabla: el nombre completo del espacio de claves y de la tabla.
- Descripción: descripción de la tabla, por ejemplo, cómo se usa o qué tipo de datos se almacenan en ella.
- Promedio de lecturas por segundo: el número medio de lecturas en las coordenadas para la tabla durante un intervalo de tiempo amplio.
- Promedio de escrituras por segundo: el número medio de escrituras en las coordenadas para la tabla durante un intervalo de tiempo amplio.
- Promedio de tamaño de fila en bytes: tamaño medio de fila en bytes.
- Tamaño de almacenamiento en GB: tamaño de almacenamiento sin procesar de una tabla.
- Desglose de coherencia de lectura: el porcentaje de lecturas que utilizan coherencia posterior (LOCAL_ONE o ONE) frente a las que utilizan coherencia sólida (LOCAL_QUORUM).

En esta tabla se muestra un ejemplo de la información relativa a las tablas que debe reunir al planificar una migración.

Nombre de la tabla	Description (Descripción)	Promedio de lecturas por segundo	Promedio de escrituras por segundo	Promedio de tamaño de fila en bytes	Tamaño de almacenamiento en GB	Desglose de coherencia de lectura
mykeyspace.mytable	Se usa para almacenar el historial del carrito de la compra	10 000	5 000	2.200	2,000	100% LOCAL_ONE
mykeyspace.mytable 2	Se utiliza para almacenar la información de perfil más reciente	20 000	1 000	850	1 000	25 % LOCAL_QUORUM 75 % LOCAL_ONE

Recopilación de las métricas de las tablas

En esta sección, se proporcionan instrucciones paso a paso sobre cómo recopilar las métricas de tabla necesarias del clúster de Cassandra existente. Estas métricas incluyen el tamaño de la fila, el tamaño de la tabla y read/write las solicitudes por segundo (RPS). Le permiten evaluar los requisitos de capacidad de rendimiento para una tabla de Amazon Keyspaces y estimar los precios.

Recopilación de las métricas de la tabla de origen de Cassandra

1. Determinación del tamaño de fila

El tamaño de fila es importante para determinar el uso de la capacidad de lectura y escritura en Amazon Keyspaces. En el siguiente diagrama se muestra la distribución de datos típica en un rango de tokens de Cassandra.



Puede usar un script de muestreo de tamaño de fila disponible [GitHub](#) para recopilar métricas de tamaño de fila para cada tabla de su clúster de Cassandra.

El script exporta los datos de la tabla desde Apache Cassandra utilizando `cqlsh` y `awk` para calcular la desviación mínima, máxima, media y estándar del tamaño de la fila sobre un conjunto configurable de datos de tabla de muestra. El muestreador de tamaño de fila pasa los argumentos a `cqlsh`, por lo que se pueden usar los mismos parámetros para conectarse y leer desde el clúster de Cassandra.

La siguiente instrucción es un ejemplo de ello.

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

Para obtener más información sobre cómo se calcula el tamaño de las filas en Amazon Keyspaces, consulte [the section called “Estimación del tamaño de las filas”](#).

2. Determinación del tamaño de las tablas

Con Amazon Keyspaces, no es necesario aprovisionar el almacenamiento por adelantado. Amazon Keyspaces supervisa continuamente el tamaño facturable de sus tablas para determinar sus cargos por almacenamiento. El almacenamiento se factura por GB-month. El tamaño de tabla de Amazon Keyspaces se basa en el tamaño sin procesar (sin comprimir) de una sola réplica.

Para supervisar el tamaño de tabla en Amazon Keyspaces, puede utilizar la métrica `BillableTableSizeInBytes`, que se muestra para cada tabla en la Consola de administración de AWS.

Para calcular el tamaño facturable de la tabla de Amazon Keyspaces, puede usar uno de estos dos métodos:

- Utilizar el tamaño de fila promedio y multiplicar por el número de filas.

Puede estimar el tamaño de la tabla de Amazon Keyspaces multiplicando el tamaño medio de las filas por el número de filas de la tabla de origen de Cassandra. Utilizar el script de muestra de tamaño de fila de la sección anterior para registrar el tamaño medio de las filas. Para registrar el recuento de filas, puede utilizar herramientas como `dsbulk count` para determinar el número total de filas de la tabla de origen.

- Utilizar `nodetool` para recopilar los metadatos de la tabla.

`Nodetool` es una herramienta administrativa incluida en la distribución de Apache Cassandra que proporciona información sobre el estado del proceso de Cassandra y devuelve los metadatos de la tabla. Puede utilizar `nodetool` para muestrear metadatos sobre el tamaño de la tabla y, con ello, extrapolar el tamaño de la tabla en Amazon Keyspaces.

El comando que se debe utilizar es `nodetool tablestats`. `Tablestats` devuelve el tamaño y la relación de compresión de la tabla. El tamaño de la tabla se guarda como el `tablelivespace` de la tabla y se puede dividir por la `compression ratio`. A continuación, multiplique este valor de tamaño por la cantidad de nodos. Por último, divida por el factor de replicación (normalmente es tres).

Esta es la fórmula completa del cálculo que puede utilizar para evaluar el tamaño de la tabla.

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

Supongamos que su clúster de Cassandra tiene 12 nodos. Al ejecutar el comando `nodetool tablestats`, se obtiene un `tablelivespace` de 200 GB y una `compression ratio` de 0,5. El espacio de claves tiene un factor de replicación de tres.

Este es el aspecto que tiene el cálculo para este ejemplo.

```
(200 GB / 0.5) * (12 nodes) / (replication factor of 3)  
= 4,800 GB / 3
```


Keyspaces = 1,600 GB is the table size estimate for Amazon

3. Registro del número de lecturas y escrituras

Para determinar los requisitos de capacidad y escalado de las tablas de Amazon Keyspaces, registre la tasa de solicitudes de lectura y escritura de las tablas de Cassandra antes de la migración.

Amazon Keyspaces se ejecuta sin servidor y solo se paga por lo que se usa. En general, el precio del read/write rendimiento en Amazon Keyspaces depende del número y el tamaño de las solicitudes.

Existen dos modos de capacidad en Amazon Keyspaces:

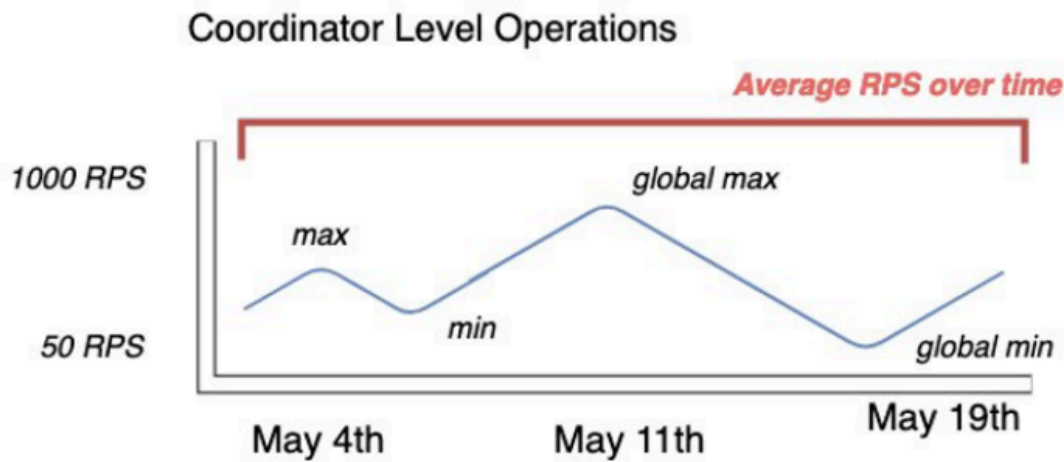
- [On-demand](#)— Se trata de una opción de facturación flexible capaz de atender miles de solicitudes por segundo sin necesidad de planificar la capacidad. Ofrece precios de pago por solicitud para las solicitudes de lectura y escritura, de forma que solo se paga por lo que se utiliza.
- [Aprovisionado](#): 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 ayuda a gestionar el uso de Amazon Keyspaces para que se mantenga igual o inferior a una tasa de solicitudes definida a fin de mantener la previsibilidad.

El modo aprovisionado ofrece [escalado automático](#) para ajustar automáticamente la tasa aprovisionada para escalarla automática o verticalmente con el fin de mejorar la eficiencia operativa. Para obtener más información sobre la administración de recursos sin servidor, consulte [Administración de recursos sin servidor](#).

Como la capacidad de rendimiento de lectura y escritura se aprovisiona por separado en Amazon Keyspaces, debe medir la tasa de solicitudes de lectura y escritura en sus tablas existentes de forma independiente.

Para recopilar las métricas de uso más precisas del clúster de Cassandra actual, registre el promedio de solicitudes por segundo (RPS) para las operaciones de lectura y escritura en el coordinador durante un período prolongado para obtener una tabla en la que aparezcan combinados todos los nodos de un solo centro de datos.

Al registrar el promedio de RPS durante un período de al menos varias semanas, quedan registrados los picos y valles de los patrones de tráfico, como se muestra en el siguiente diagrama.



Tienes dos opciones para determinar la tasa de solicitudes de lectura y escritura de una tabla de Cassandra.

- Utilizar la supervisión de Cassandra existente

Puede usar las métricas que se muestran en la siguiente tabla para observar las solicitudes de lectura y escritura. Tenga en cuenta que los nombres de las métricas pueden cambiar en función de la herramienta de supervisión que utilice.

Dimensión	Métrica de Cassandra JMX
Escrituras	<code>org.apache.cassandra.metrics:type=ClientRequest,scope=Write,name=Latency#Count</code>
Lecturas	<code>org.apache.cassandra.metrics:type=ClientRequest,scope=Read,name=Latency#Count</code>

- Utilizar `nodetool`

Utilice `nodetool tablestats` y `nodetool info` para registrar el promedio de operaciones de lectura y escritura de la tabla. `tablestats` devuelve el recuento total de lecturas y escrituras desde el momento en que se inició el nodo. `nodetool info` proporciona el tiempo de actividad de un nodo en segundos.

Para obtener el promedio de lecturas y escrituras por segundo, divida el recuento de lectura y escritura por el tiempo de actividad del nodo en segundos. A continuación, para las lecturas, se divide por el nivel de consistencia, y para las escrituras, se divide por el factor de replicación. Estos cálculos se expresan en las siguientes fórmulas.

Fórmula para el promedio de lecturas por segundo:

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

Fórmula para el promedio de escrituras por segundo:

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

Supongamos que tenemos un clúster de 12 nodos que ha estado activo durante 4 semanas. `nodetool info` devuelve 2 419 200 segundos de tiempo de actividad y `nodetool tablestats` devuelve 1000 millones de escrituras y 2000 millones de lecturas. Este ejemplo daría como resultado el siguiente cálculo.

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

4. Determinación del uso de la capacidad de la tabla

Para estimar el uso promedio de la capacidad, comience con el promedio de las tasas de solicitud y el promedio del tamaño de fila de la tabla de origen de Cassandra.

Amazon Keyspaces utiliza unidades de capacidad de lectura (RCU) y unidades de capacidad de escritura (WCU) para medir la capacidad de rendimiento aprovisionada para lecturas y escrituras en tablas. Para esta estimación, utilizamos estas unidades para calcular las necesidades de capacidad de lectura y escritura de la nueva tabla de Amazon Keyspaces tras la migración.

Más adelante en este tema analizaremos cómo afecta a la facturación la elección entre el modo de capacidad aprovisionada y el modo de capacidad bajo demanda. Sin embargo, para estimar el uso de la capacidad en este ejemplo, suponemos que la tabla está en modo aprovisionado.

- **Lecturas:** una RCU 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 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 coherencia de lectura `LOCAL_QUORUM` o `LOCAL_ONE`.

Por ejemplo, la lectura de una fila de 8 KB requiere 2 RCU con coherencia de lectura `LOCAL_QUORUM` y 1 RCU si elige coherencia de lectura `LOCAL_ONE`.

- **Escrituras:** una WCU 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).

El número total de WCU necesarias depende del tamaño de la fila. Si necesita escribir una fila de más de 1 KB, la operación de escritura utiliza WCU adicionales. Por ejemplo, si el tamaño de su fila es de 2 KB, necesitará 2 WCU para realizar una solicitud de escritura.

La siguiente fórmula se puede utilizar para estimar las RCU y las WCU necesarias.

- La capacidad de lectura en RCU se puede determinar multiplicando las lecturas por segundo por el número de filas leídas en cada lectura, multiplicado por el tamaño medio de fila dividido entre 4 KB y redondeado al número entero más cercano.
- La capacidad de escritura en WCU se puede determinar multiplicando el número de solicitudes por el tamaño medio de fila dividido por 1 KB y redondeándolo al número entero más cercano.

Esto se expresa en las siguientes fórmulas.

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

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

Por ejemplo, si realiza 4960 solicitudes de lectura con un tamaño de fila de 2,5 KB en la tabla de Cassandra, necesitará 4960 RCU en Amazon Keyspaces. Si actualmente realiza 1653 solicitudes de escritura por segundo con un tamaño de fila de 2,5 KB en la tabla de Cassandra, necesitará 4959 WCU por segundo en Amazon Keyspaces.

Este ejemplo se expresa en las siguientes fórmulas.

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

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

El uso de eventual consistency permite ahorrar hasta la mitad de la capacidad de rendimiento en cada solicitud de lectura. Cada lectura coherente posterior puede consumir hasta 8 KB. Puede calcular las lecturas coherentes posteriores multiplicando el cálculo anterior por 0,5, tal y como se muestra en la siguiente fórmula.

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

5. Cálculo del precio mensual estimado para Amazon Keyspaces

Para calcular la facturación mensual de la tabla en función del rendimiento de la read/write capacidad, puede calcular los precios del modo bajo demanda y del modo aprovisionado mediante distintas fórmulas y comparar las opciones de la tabla.

Modo aprovisionado: el consumo de la capacidad de lectura y escritura se factura según una tarifa por hora basada en las unidades de capacidad por segundo. En primer lugar, divida esa tasa entre 0,7 para representar el objetivo de utilización predeterminado del escalado

automático, que es del 70 %. Luego se debe multiplicar por 30 días naturales, 24 horas por día y los precios de las tarifas regionales.

Este cálculo se resume en las siguientes fórmulas.

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

On-demand modo: la capacidad de lectura y escritura se factura según una tarifa por solicitud. En primer lugar, multiplique la tasa de solicitudes por 30 días naturales y 24 horas por día. A continuación, divida el resultado por un millón de unidades de solicitud. Por último, multiplique el resultado por la tarifa regional.

Este cálculo se resume en las siguientes fórmulas.

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

Elección de una estrategia de migración

Puede elegir entre las siguientes estrategias de migración al migrar de Apache Cassandra a Amazon Keyspaces:

- **En línea:** se trata de una migración en directo que utiliza escrituras duales para empezar a escribir nuevos datos en Amazon Keyspaces y en el clúster de Cassandra de forma simultánea. Este tipo de migración se recomienda para las aplicaciones que requieren cero tiempo de inactividad durante la migración y que requieren coherencia de lectura tras la escritura.

Para obtener más información sobre cómo planificar e implementar una estrategia de migración en línea, consulte [the section called “Migración en línea”](#).

- **Sin conexión:** esta técnica de migración implica copiar un conjunto de datos de Cassandra a Amazon Keyspaces durante un tiempo de inactividad. La migración sin conexión puede simplificar el proceso de migración, ya que no requiere cambios en la aplicación ni la resolución de conflictos entre los datos históricos y las nuevas escrituras.

Para obtener más información sobre cómo planificar una migración sin conexión, consulte [the section called “Migración sin conexión”](#).

- Híbrida: esta técnica de migración permite replicar los cambios en Amazon Keyspaces prácticamente en tiempo real, pero sin coherencia de lectura tras la escritura.

Para obtener más información sobre cómo planificar una migración híbrida, consulte [the section called “Migración híbrida”](#).

Tras revisar las técnicas de migración y las prácticas recomendadas que se describen en este tema, puede colocar las opciones disponibles en un árbol de decisiones para diseñar una estrategia de migración en función de sus requisitos y de los recursos disponibles.

Migración online a Amazon Keyspaces: estrategias y prácticas recomendadas

Si necesita mantener la disponibilidad de las aplicaciones durante una migración de Apache Cassandra a Amazon Keyspaces, puede preparar una estrategia de migración en línea personalizada mediante la implementación de los componentes clave que se describen en este tema. Si sigue estas prácticas recomendadas para las migraciones en línea, puede garantizar que la disponibilidad de las aplicaciones y la coherencia de lectura tras la escritura se mantengan durante todo el proceso de migración, lo que minimiza el impacto sobre sus usuarios.

Al diseñar una estrategia de migración en línea de Apache Cassandra a Amazon Keyspaces, debe tener en cuenta los siguientes pasos clave.

1. Escritura de nuevos datos

- Proxy de escritura doble de ZDM para la migración de Amazon Keyspaces: utilice el proxy de escritura doble de ZDM disponible [en](#) Github para realizar una migración sin tiempo de inactividad de Apache Cassandra a Amazon Keyspaces. El proxy ZDM realiza escrituras duales sin necesidad de refactorizar las aplicaciones existentes y realiza lecturas duales para validar las consultas.
- Escrituras duales de aplicación: puede implementar escrituras duales en su aplicación utilizando las bibliotecas de clientes y los controladores existentes de Cassandra. Designe una base de datos como líder y la otra como seguidora. Los errores de escritura en la base de datos seguidora se registran en una [cola de mensajes fallidos \(DLQ\)](#) para su análisis.

- Escrituras duales en el nivel de mensajería: también puede configurar su plataforma de mensajería existente para enviar escrituras tanto a Cassandra como a Amazon Keyspaces con un consumidor adicional. Con el tiempo, esto crea vistas coherentes en ambas bases de datos.

2. Migración de datos históricos

- Copia de datos históricos: puede migrar datos históricos de Cassandra a Amazon Keyspaces con AWS Glue o mediante scripts de extracción, transformación y carga (ETL) personalizados. Administre la resolución de conflictos entre escrituras duales y cargas en lotes mediante técnicas como las transacciones simplificadas o las marcas de tiempo.
- Uso Time-To-Live (TTL): para períodos de retención de datos más cortos, puede usar TTL tanto en Cassandra como en Amazon Keyspaces para evitar cargar datos históricos innecesarios. A medida que los datos antiguos caducan en Cassandra y se escriben datos nuevos mediante escrituras duales, Amazon Keyspaces acaba poniéndose al día.

3. Validación de los datos

- Lecturas duales: implemente lecturas duales tanto desde las bases de datos de Cassandra (principal) como desde las de Amazon Keyspaces (secundaria) y compare los resultados de forma asíncrona. Las diferencias se registran o se envían a una DLQ.
- Lecturas de ejemplo: utilice funciones λ para muestrear y comparar periódicamente los datos entre ambos sistemas y registrar cualquier discrepancia en una DLQ.

4. Migración de la aplicación

- Blue-green estrategia: cambie su aplicación para tratar Amazon Keyspaces como el almacén de datos principal y Cassandra como el almacén de datos secundario en un solo paso. Supervise el rendimiento y deshaga esas modificaciones en caso de que surjan problemas.
- Implementación canario: primero implemente gradualmente la migración a un subconjunto de usuarios, aumentando gradualmente el tráfico a Amazon Keyspaces como principal hasta que se migre por completo.

5. Retirada de Cassandra

Una vez que su aplicación se haya migrado por completo a Amazon Keyspaces y se haya validado la coherencia de datos, puede planificar la retirada del clúster de Cassandra en función de las políticas de retención de datos.

Si planifica una estrategia de migración en línea con estos componentes, podrá realizar una transición fluida al servicio de Amazon Keyspaces totalmente administrado con un tiempo de

inactividad o interrupción mínimos. En las siguientes secciones se aborda cada componente de manera detallada.

Temas

- [Escritura de nuevos datos durante una migración en línea](#)
- [Carga de datos históricos durante una migración en línea](#)
- [Validación de la coherencia de datos durante una migración en línea](#)
- [Migración de la aplicación durante una migración en línea](#)
- [Retirada de Cassandra tras una migración en línea](#)

Escritura de nuevos datos durante una migración en línea

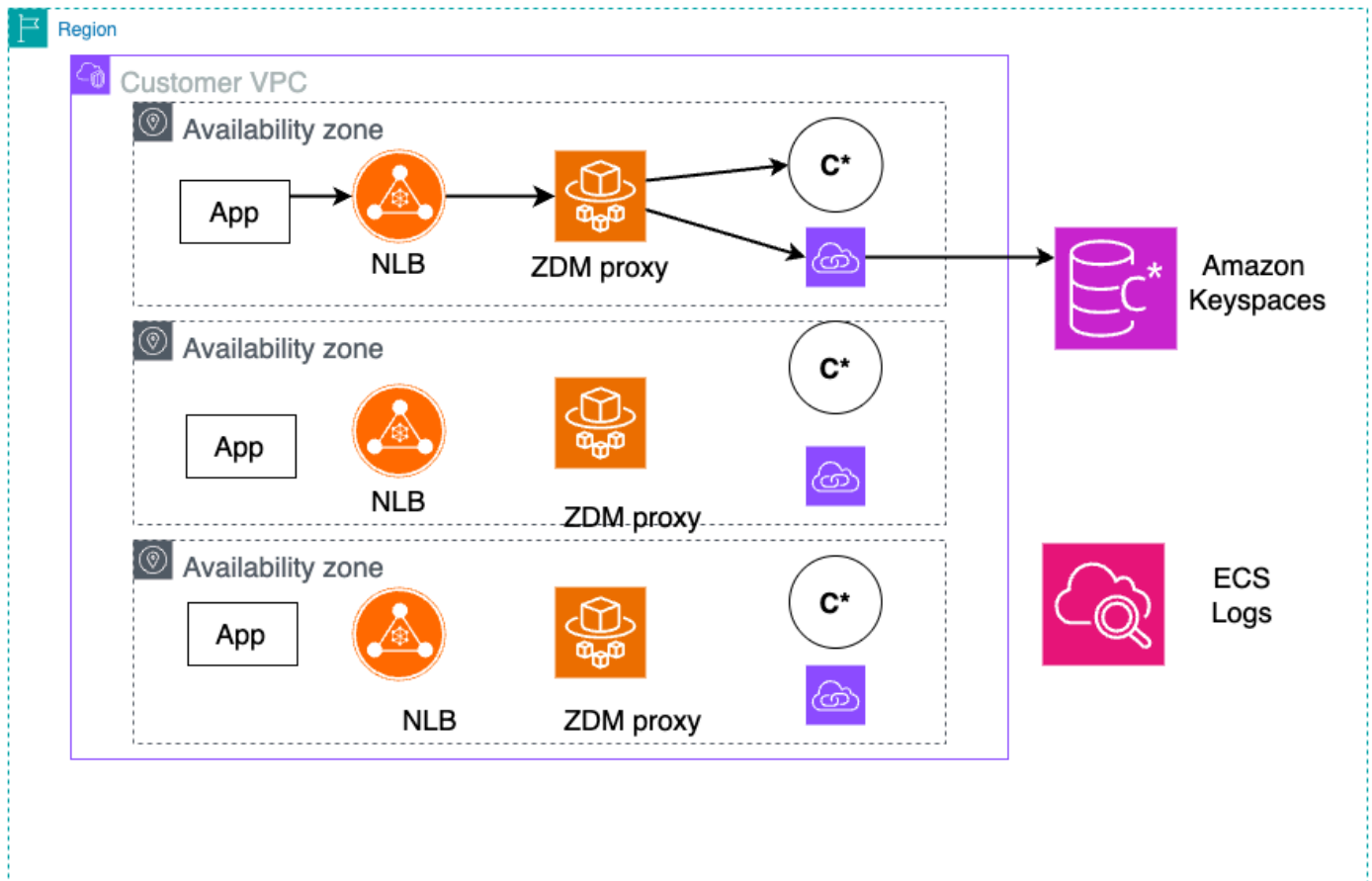
El primer paso de un plan de migración en línea consiste en garantizar que cualquier dato nuevo que escriba la aplicación se almacene en ambas bases de datos, en el clúster de Cassandra existente y en Amazon Keyspaces. El objetivo es proporcionar una visión coherente de los dos almacenes de datos. Para ello, aplique todas las escrituras nuevas a ambas bases de datos. Para implementar escrituras duales, considere una de las tres opciones siguientes.

- Migración de proxy de escritura doble de ZDM para Amazon Keyspaces: con el proxy de ZDM para Amazon Keyspaces disponible [en](#) Github, puede migrar sus cargas de trabajo de Apache Cassandra a Amazon Keyspaces sin tiempo de inactividad de la aplicación. Esta solución mejorada implementa las AWS mejores prácticas y amplía las capacidades oficiales del proxy de ZDM.
- Realice migraciones en línea entre Apache Cassandra y Amazon Keyspaces.
- Escriba datos en las tablas de origen y destino simultáneamente sin refactorizar las aplicaciones.
- Valide las consultas mediante operaciones de doble lectura.

La solución ofrece las siguientes mejoras para trabajar con AWS Amazon Keyspaces.

- Despliegue de contenedores: utilice una imagen de Docker preconfigurada de Amazon Elastic Container Registry (Amazon ECR) para las implementaciones. VPC-accessible
- Infraestructura como código: impleméntela mediante AWS CloudFormation plantillas para una configuración automática. AWS Fargate
- Compatibilidad con Amazon Keyspaces: acceda a las tablas del sistema con adaptaciones personalizadas para Amazon Keyspaces.

La solución se ejecuta en Amazon ECS con Fargate, lo que proporciona escalabilidad sin servidor en función de sus demandas de carga de trabajo. Un balanceador de carga de red distribuye el tráfico entrante de las aplicaciones entre varias tareas de Amazon ECS para lograr una alta disponibilidad.



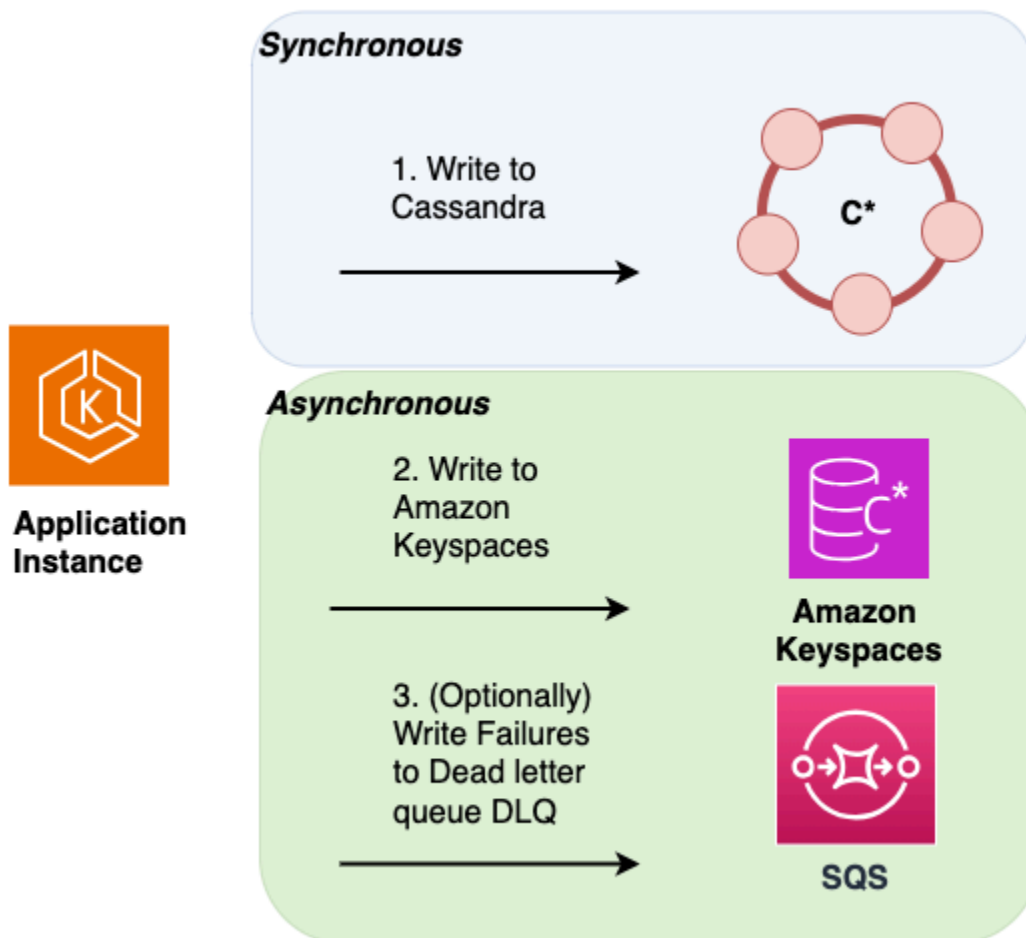
- **Escrituras duales de aplicación:** puede implementar escrituras duales con cambios mínimos en el código de su aplicación aprovechando las bibliotecas de clientes y los controladores existentes de Cassandra. Puede implementar escrituras duales en su aplicación existente o crear una nueva capa en la arquitectura para administrar las escrituras duales. Para obtener más información y un caso práctico de un cliente que demuestra cómo se implementaron las escrituras duales en una aplicación existente, consulte este [caso práctico de migración de Cassandra](#).

Al implementar escrituras duales, puede designar una base de datos como líder y la otra base de datos como seguidora. Esto le permite seguir escribiendo en la base de datos original (o líder) evitando que los errores de escritura en la base de datos de destino (o seguidora) interrumpen la ruta crítica de la aplicación.

En lugar de volver a intentar las escrituras fallidas en la seguidora, puede utilizar Amazon Simple Queue Service para registrar las escrituras fallidas en una [cola de mensajes fallidos \(DLQ\)](#). La DLQ le permite analizar las escrituras fallidas en la seguidora y determinar por qué no se ha realizado correctamente el procesamiento en la base de datos de destino.

[Para una implementación de escritura doble más sofisticada, puede seguir las AWS mejores prácticas para diseñar una secuencia de transacciones locales utilizando el patrón saga.](#) Un patrón saga asegura que, si se produce un error en una transacción, la saga ejecuta transacciones de compensación para revertir los cambios en la base de datos realizados por las transacciones anteriores.

Si utiliza escrituras duales para una migración en línea, puede configurar las escrituras duales para que sigan el patrón saga, de modo que cada escritura sea una transacción local que garantice las operaciones atómicas en bases de datos heterogéneas. Para obtener más información sobre cómo diseñar aplicaciones distribuidas utilizando los patrones de diseño recomendados para ellas Nube de AWS, consulte [Patrones de diseño, arquitecturas e implementaciones en la nube](#).



- Escrituras duales en el nivel de mensajería: en lugar de implementar escrituras duales en la capa de aplicación, puede usar el nivel de mensajería existente para realizar escrituras duales en Cassandra y Amazon Keyspaces.

Para ello, puede configurar un consumidor adicional para su plataforma de mensajería de modo que envíe las escrituras a ambos almacenes de datos. Este enfoque proporciona una estrategia sencilla y con poco código que utiliza el nivel de mensajería para crear dos vistas en ambas bases de datos que, en última instancia, sean coherentes.

Carga de datos históricos durante una migración en línea

Tras implementar la escritura dual para garantizar que los nuevos datos se escriban en ambos almacenes de datos en tiempo real, el siguiente paso del plan de migración consiste en evaluar la cantidad de datos históricos que necesita copiar o cargar por lotes de Cassandra a Amazon Keyspaces. Esto garantiza que tanto los datos nuevos como los históricos estén disponibles en la nueva base de datos de Amazon Keyspaces antes de migrar la aplicación.

En función de sus requisitos de retención de datos, por ejemplo, de la cantidad de datos históricos que necesite conservar en función de las políticas de su organización, puede considerar una de las dos opciones siguientes.

- Carga masiva de datos históricos: la migración de los datos históricos de su implementación actual de Cassandra a Amazon Keyspaces se puede realizar mediante diversas técnicas, por ejemplo, el AWS Glue uso de scripts personalizados para extraer, transformar y cargar (ETL) los datos. Para obtener más información acerca del uso de AWS Glue para cargar datos históricos, consulte [the section called “Migración sin conexión”](#).

Al planificar la carga masiva de datos históricos, hay que tener en cuenta cómo resolver los conflictos que pueden producirse cuando se intenta actualizar con nuevas escrituras los mismos datos que se están cargando. Cabe esperar que la carga por lotes acabe siendo coherente, lo que significa que los datos llegarán a todos los nodos al final.

Si se produce una actualización de los mismos datos al mismo tiempo debido a una nueva escritura, debe asegurarse de que la carga de los datos históricos no los sobrescriba. Para asegurarse de conservar las últimas actualizaciones de sus datos incluso durante la importación por lotes, debe incorporar la solución de conflictos a los scripts de carga por lotes o a la lógica de la aplicación para las escrituras duales.

Por ejemplo, puede usar [the section called “Transacciones ligeras”](#) (LWT) para comparar y configurar las operaciones. Para ello, puede agregar un campo adicional a su modelo de datos que represente el momento de la modificación o el estado.

Además, Amazon Keyspaces admite la función de marca de tiempo `WRITETIME` de Cassandra. Puede utilizar las marcas de tiempo del cliente de Amazon Keyspaces para conservar las marcas de tiempo de la base de datos de origen e implementar una resolución de conflictos en la que prevalezca la última escritura. Para obtener más información, consulte [the section called “Client-side marcas de tiempo”](#).

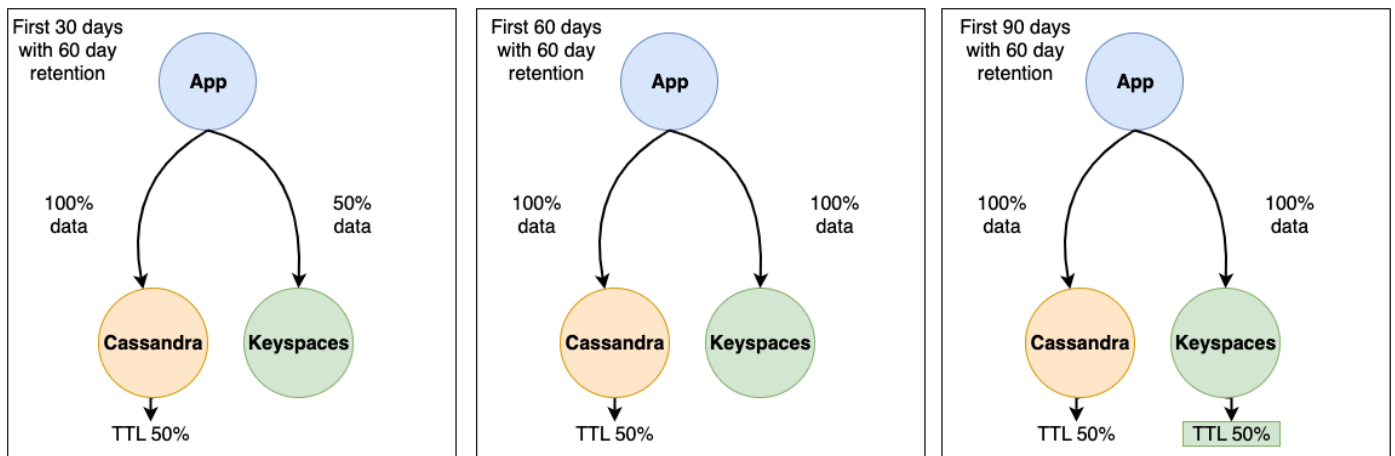
- Uso de Time-to-Live (TTL): para períodos de retención de datos inferiores a 30, 60 o 90 días, puede usar TTL en Cassandra y Amazon Keyspaces durante la migración para evitar cargar datos históricos innecesarios en Amazon Keyspaces. El TTL le permite establecer un período de tiempo tras el cual se quitan automáticamente los datos de la base de datos.

Durante la fase de migración, en lugar de copiar los datos históricos a Amazon Keyspaces, puede configurar los ajustes del TTL para permitir que los datos históricos caduquen automáticamente en el sistema anterior (Cassandra) y aplicar únicamente las nuevas escrituras a Amazon Keyspaces mediante el método de escritura dual. Con el paso del tiempo, y dado que los datos antiguos caducan continuamente en el clúster de Cassandra y los nuevos se escriben con el método de escritura dual, Amazon Keyspaces se pone al día automáticamente para contener los mismos datos que Cassandra.

Este enfoque puede reducir considerablemente la cantidad de datos que se van a migrar, lo que se traduce en un proceso de migración más eficiente y optimizado. Puede plantearse este enfoque cuando se enfrente a conjuntos de datos grandes con requisitos de retención de datos variables. Para obtener más información sobre el TTL, consulte [the section called “Caducidad de datos con período de vida”](#).

Plantéese el siguiente ejemplo de una migración de Cassandra a Amazon Keyspaces mediante la caducidad de datos con TTL. En este ejemplo, establecemos el TTL de ambas bases de datos en 60 días y mostramos cómo progresa el proceso de migración durante un período de 90 días. Ambas bases de datos reciben los mismos datos recién escritos durante este período mediante el método de escritura dual. Vamos a analizar tres fases diferentes de la migración, cada una de las cuales dura 30 días.

En las siguientes imágenes se muestra cómo funciona el proceso de migración para cada fase.



1. Tras los primeros 30 días, el clúster de Cassandra y Amazon Keyspaces han estado recibiendo nuevas escrituras. El clúster de Cassandra también contiene datos históricos que aún no han alcanzado los 60 días de retención, lo que representa el 50 % de los datos del clúster.

Los datos que tienen más de 60 días se eliminan automáticamente en el clúster de Cassandra mediante TTL. En este punto, Amazon Keyspaces contiene el 50 % de los datos almacenados en el clúster de Cassandra, que se compone de las nuevas escrituras menos los datos históricos.

2. Transcurridos 60 días, tanto el clúster de Cassandra como Amazon Keyspaces contienen los mismos datos escritos en los últimos 60 días.
3. En un plazo de 90 días, tanto Cassandra como Amazon Keyspaces contienen los mismos datos y los datos caducan al mismo ritmo.

Este ejemplo ilustra cómo evitar el paso de cargar datos históricos mediante el uso del TTL con una fecha de caducidad establecida en 60 días.

Validación de la coherencia de datos durante una migración en línea

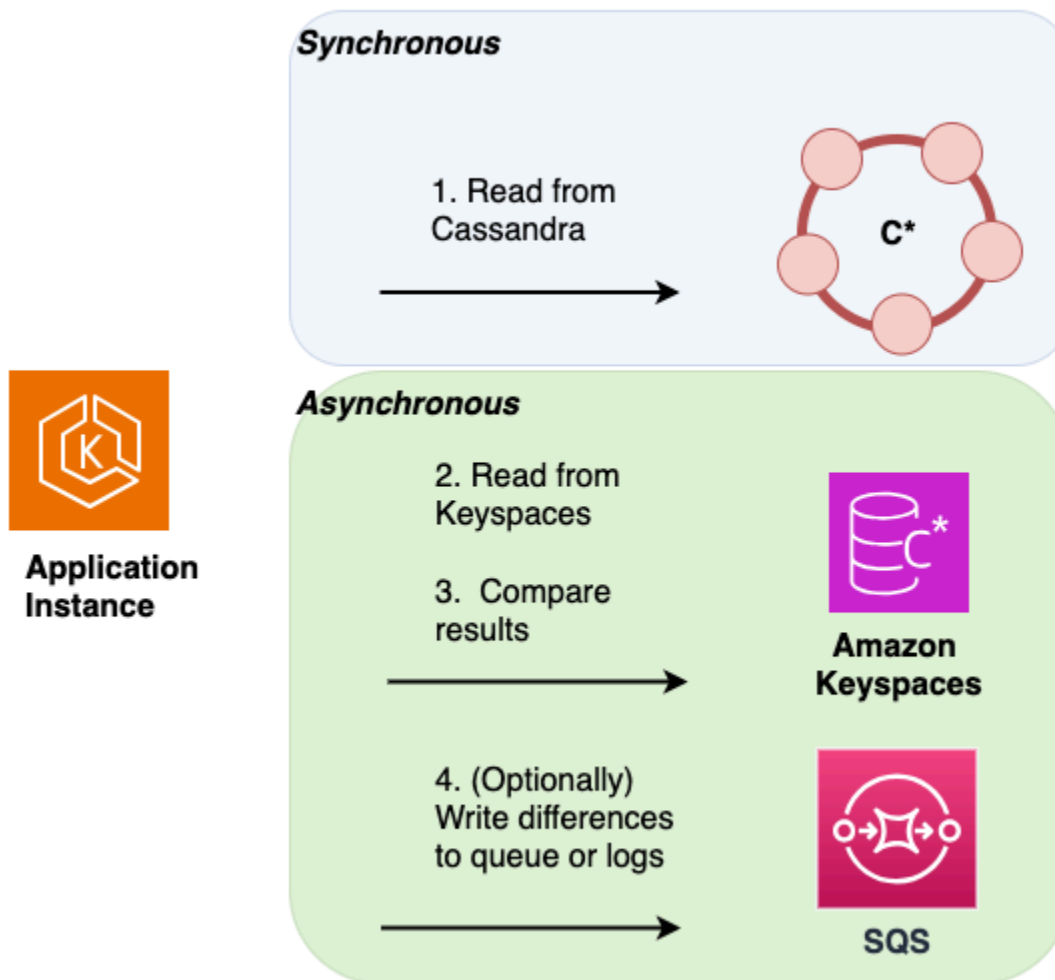
El siguiente paso del proceso de migración en línea es la validación de los datos. Las escrituras duales añaden nuevos datos a su base de datos de Amazon Keyspaces y ha completado la migración de los datos históricos mediante la carga por lotes o la caducidad de los datos con TTL.

Ahora puede utilizar la fase de validación para confirmar que ambos almacenes de datos contienen realmente los mismos datos y devuelven los mismos resultados de lectura. Puede elegir una de las dos opciones siguientes para validar que ambas bases de datos contengan datos idénticos.

- **Lecturas duales:** para validar que tanto la base de datos de origen como la de destino contengan el mismo conjunto de datos históricos y recién escritos, puede implementar lecturas duales. Para ello, debe leer tanto la base de datos principal de Cassandra como la base de datos secundaria de Amazon Keyspaces, de forma similar al método de escritura dual, y comparar los resultados de forma asíncrona.

Los resultados de la base de datos principal se devuelven al cliente y los resultados de la base de datos secundaria se utilizan para validarlos con el conjunto de resultados principal. Las diferencias detectadas se pueden registrar o enviar a una [cola de mensajes fallidos \(DLQ\)](#) para su posterior conciliación.

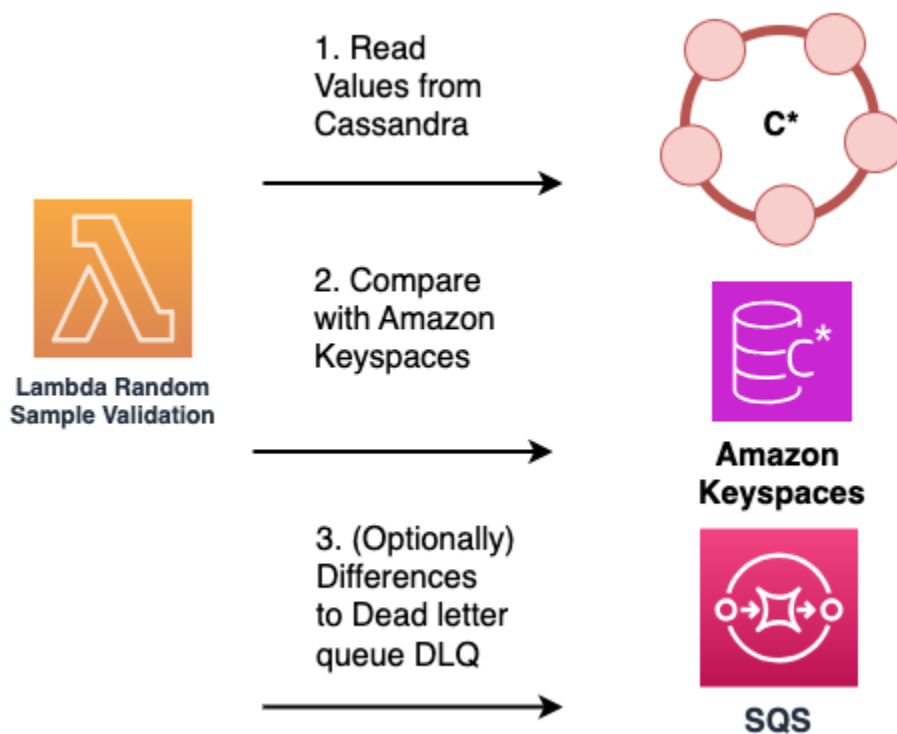
En el siguiente diagrama, la aplicación realiza una lectura sincrónica desde Cassandra (que es el almacén de datos principal) y una lectura asíncrona desde Amazon Keyspaces, que es el almacén de datos secundario.



- **Lecturas de muestra:** una solución alternativa que no requiere cambios en el código de la aplicación consiste en utilizar AWS Lambda funciones para muestrear de forma periódica y aleatoria los datos del clúster Cassandra de origen y de la base de datos Amazon Keyspaces de destino.

Estas funciones de Lambda se pueden configurar para que se ejecuten a intervalos regulares. La función de Lambda recupera un subconjunto aleatorio de datos de los sistemas de origen y destino y, a continuación, realiza una comparación de los datos muestreados. Cualquier discrepancia o discordancia entre los dos conjuntos de datos se puede registrar y enviar a una [cola de mensajes fallidos \(DLQ\)](#) específica para su posterior conciliación.

Este proceso se ilustra en el siguiente diagrama.



Migración de la aplicación durante una migración en línea

En la cuarta fase de una migración en línea, migrará su aplicación y realizará la transición a Amazon Keyspaces como almacén de datos principal. Esto significa que puede cambiar la aplicación para que lea y escriba directamente desde y hacia Amazon Keyspaces. Para garantizar una interrupción mínima para sus usuarios, debe ser un proceso bien planificado y coordinado.

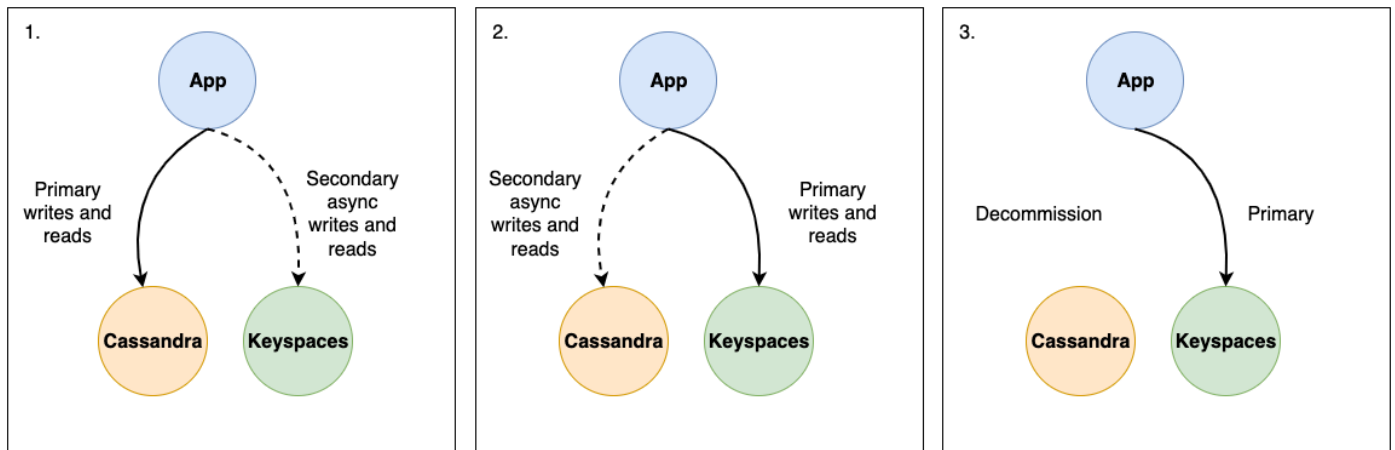
Hay disponibles dos soluciones recomendadas diferentes para la migración de aplicaciones: la estrategia de transición azul-verde y la estrategia de transición canario. En las siguientes secciones se describen estas estrategias con más detalle.

- Estrategia azul-verde: mediante el uso de este enfoque, modifica su aplicación para que trate Amazon Keyspaces como el almacén de datos principal y Cassandra como el almacén de datos secundario en un solo paso. Para ello, utilice un indicador de AWS AppConfig función para controlar la elección de los almacenes de datos principales y secundarios en la instancia de la aplicación. Para obtener más información sobre marcas de características, consulte [Creación de un perfil de configuración de marcas de características en AWS AppConfig](#).

Tras convertir Amazon Keyspaces en el almacén de datos principal, debe supervisar el comportamiento y el rendimiento de la aplicación para garantizar que Amazon Keyspaces cumpla sus requisitos y que la migración se realice correctamente.

Por ejemplo, si ha implementado lecturas duales para su aplicación, durante la fase de migración de la aplicación, pasará las lecturas principales de Cassandra a Amazon Keyspaces y las lecturas secundarias de Amazon Keyspaces a Cassandra. Tras la transición, debe seguir supervisando y comparando los resultados tal y como se describe en la sección de [validación de datos](#) para garantizar la coherencia en ambas bases de datos antes de retirar Cassandra del servicio.

Si detecta algún problema, puede volver rápidamente al estado anterior recuperando Cassandra como almacén de datos principal. Solo pasará a la fase de retirada de la migración si Amazon Keyspaces satisface todas sus necesidades como almacén de datos principal.



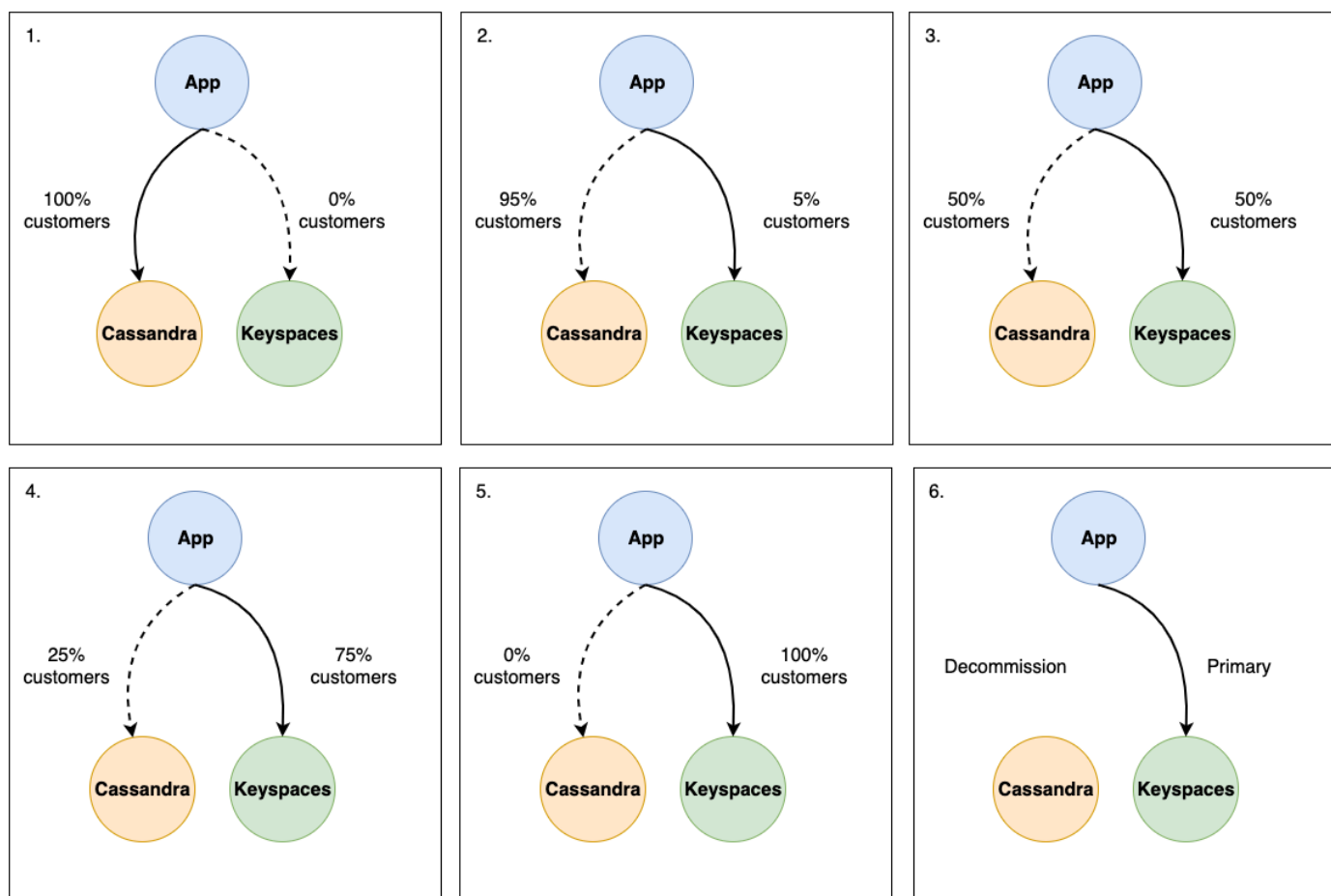
- Estrategia canario: con este enfoque, se lleva a cabo gradualmente la migración para un subconjunto de los usuarios o del tráfico. Inicialmente, un pequeño porcentaje del tráfico de la aplicación (por ejemplo, el 5 % de todo el tráfico) se redirige a la versión que utiliza Amazon

Keyspaces como almacén de datos principal, mientras que el resto del tráfico sigue utilizando Cassandra como almacén de datos principal.

Esto le permite probar exhaustivamente la versión migrada con tráfico real, supervisar su rendimiento y estabilidad e investigar los posibles problemas. Si no detecta ningún problema, puede aumentar gradualmente el porcentaje de tráfico que se dirige a Amazon Keyspaces hasta que se convierta en el almacén de datos principal para todos los usuarios y todo el tráfico.

Esta implementación por etapas minimiza el riesgo de interrupciones generalizadas del servicio y permite un proceso de migración más controlado. Si surge algún problema grave durante la implementación canario, puede volver rápidamente a la versión anterior utilizando Cassandra como almacén de datos principal para el segmento de tráfico afectado. Solo pasará a la fase de retirada de la migración después de haber validado que Amazon Keyspaces procesa el 100 % de los usuarios y el tráfico según lo previsto.

El siguiente diagrama ilustra los pasos individuales de la estrategia canario.



Retirada de Cassandra tras una migración en línea

Una vez finalizada la migración de la aplicación, cuando la aplicación se esté ejecutando completamente en Amazon Keyspaces y se haya validado la coherencia de datos durante un período de tiempo, puede planificar la retirada del clúster de Cassandra. Durante esta fase, puede evaluar si los datos que quedan en su clúster de Cassandra deben archivarse o si se pueden eliminar. Esto depende de las políticas de su organización en materia de tratamiento y retención de datos.

Si sigue esta estrategia y tiene en cuenta las prácticas recomendadas que se describen en este tema al planificar la migración en línea de Cassandra a Amazon Keyspaces, puede garantizar una transición fluida a Amazon Keyspaces y, al mismo tiempo, mantener la coherencia de lectura tras la escritura y la disponibilidad de su aplicación.

La migración de Apache Cassandra a Amazon Keyspaces puede ofrecer numerosas ventajas, como la reducción de los gastos operativos, el escalado automático, la mejora de la seguridad y un marco que le ayude a alcanzar sus objetivos de conformidad normativa. Al planificar una estrategia de migración en línea con escrituras duales, carga de datos históricos, validación de datos e implementación gradual, puede garantizar una transición fluida con una interrupción mínima para su aplicación y sus usuarios.

La implementación de la estrategia de migración en línea que se analiza en este tema le permite validar los resultados de la migración, identificar y abordar cualquier problema y, en última instancia, retirar su implementación actual de Cassandra en favor del servicio de Amazon Keyspaces totalmente administrado.

Proceso de migración sin conexión: de Apache Cassandra a Amazon Keyspaces

Las migraciones sin conexión son adecuadas cuando se pueda permitir un tiempo de inactividad para llevarlas a cabo. Es habitual que las empresas tengan períodos de mantenimiento para la aplicación de parches o lanzamientos de gran tamaño, o tiempos de inactividad para llevar a cabo actualizaciones de hardware o cambios principales. La migración sin conexión puede aprovechar estos periodos para copiar datos y transferir el tráfico de aplicaciones de Apache Cassandra a Amazon Keyspaces.

La migración sin conexión reduce las modificaciones en la aplicación porque no requiere la comunicación simultánea con Cassandra y Amazon Keyspaces. Además, dado que el flujo de datos está pausado, se puede copiar el estado exacto sin mantener las mutaciones.

En este ejemplo, utilizamos Amazon Simple Storage Service (Amazon S3) como espacio provisional para los datos durante la migración sin conexión con el objetivo de minimizar el tiempo de inactividad. Puede importar automáticamente los datos que ha almacenado en formato Parquet en Amazon S3 a una tabla de Amazon Keyspaces mediante Spark Cassandra Connector y AWS Glue. En la siguiente sección se mostrará información general de alto nivel del proceso. Puede encontrar ejemplos de código para este proceso en [Github](#).

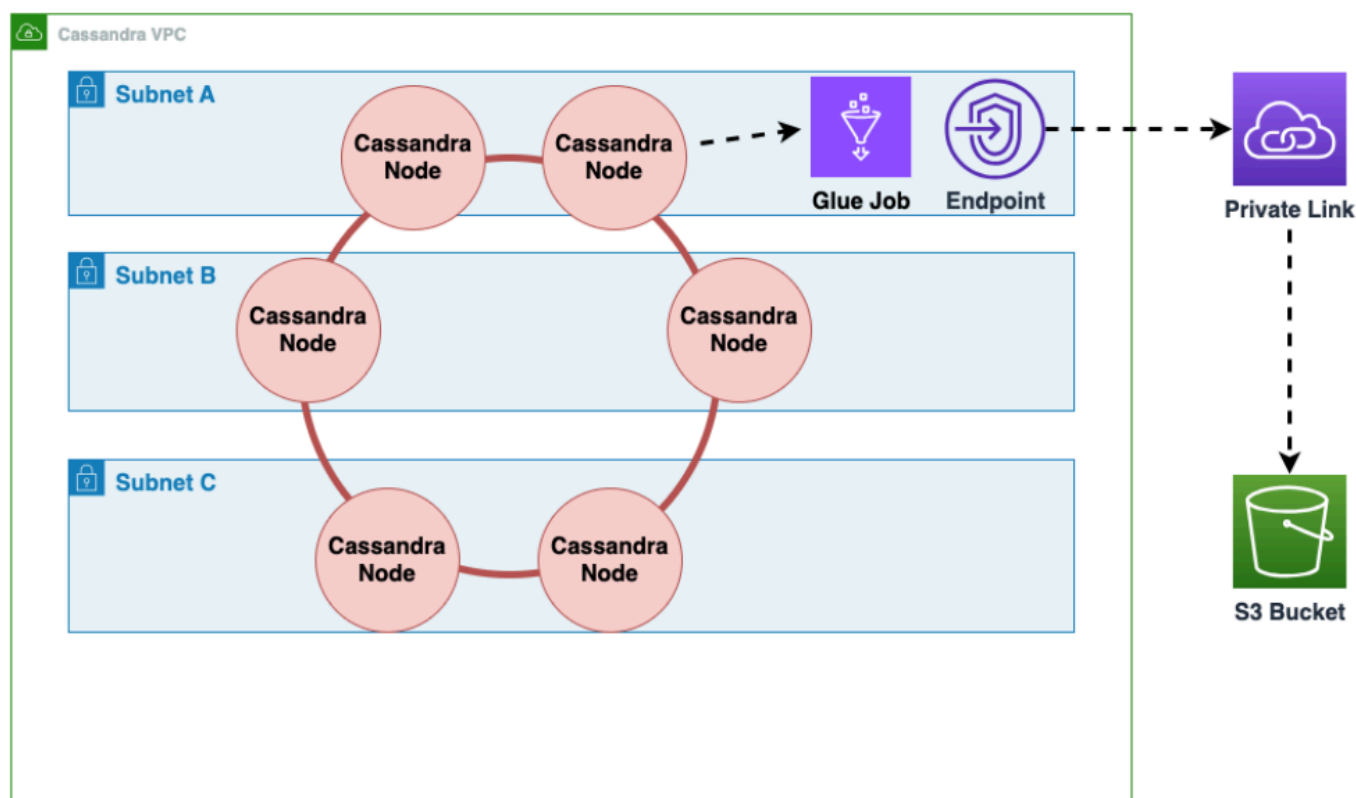
El proceso de migración sin conexión de Apache Cassandra a Amazon Keyspaces mediante Amazon S3 requiere AWS Glue los AWS Glue siguientes trabajos.

1. Un trabajo ETL que extraiga y transforme los datos de CQL y los almacena en un bucket de Amazon S3.
2. Un segundo trabajo que importe los datos del bucket a Amazon Keyspaces.
3. Un tercer trabajo que importe datos incrementales.

Cómo realizar una migración sin conexión a Amazon Keyspaces desde Cassandra cuando se ejecuta en Amazon EC2 en una Amazon Virtual Private Cloud

1. Primero se exportan AWS Glue los datos de las tablas de Cassandra en formato Parquet y se guardan en un bucket de Amazon S3. Debe ejecutar un AWS Glue trabajo mediante un AWS Glue conector a una VPC en la que resida la instancia de Amazon EC2 que ejecuta Cassandra. A continuación, con el punto de conexión privado de Amazon S3, puede guardar los datos en el bucket de Amazon S3.

En el siguiente diagrama se muestran estos pasos.



2. Mezcla de los datos en el bucket de Amazon S3 para mejorar la asignación al azar de los datos. Los datos importados de manera uniforme permiten distribuir más el tráfico en la tabla de destino.

Este paso es obligatorio cuando se exportan datos de Cassandra con particiones grandes (particiones con más de 1000 filas) para evitar patrones de claves sobrecargadas al insertar los datos en Amazon Keyspaces. Los problemas por claves sobrecargadas provocan `WriteThrottleEvents` en Amazon Keyspaces y causan un aumento del tiempo de carga.



3. Utilice otro AWS Glue trabajo para importar datos del bucket de Amazon S3 a Amazon Keyspaces. Los datos mezclados en el bucket de Amazon S3 se almacenan en formato Parquet.



Para obtener más información sobre el proceso de migración fuera de línea, consulte el taller [Amazon Keyspaces with AWS Glue](#)

Uso de una solución de migración híbrida: de Apache Cassandra a Amazon Keyspaces

La siguiente solución de migración puede considerarse un híbrido entre la migración en línea y la migración sin conexión. Con este enfoque híbrido, se escriben los datos en la base de datos de destino casi en tiempo real sin proporcionar coherencia de lectura tras la escritura. Esto significa que los datos recién escritos no estarán disponibles de forma inmediata y que es previsible que se produzcan retrasos. Si necesita coherencia de lectura tras la escritura, consulte [the section called “Migración en línea”](#).

Para realizar una migración casi en tiempo real de Apache Cassandra a Amazon Keyspaces, puede elegir entre dos métodos disponibles.

- **CQLReplicator:** (recomendado) CQLReplicator es una utilidad de código abierto disponible en [Github](#) que le ayuda a migrar datos de Apache Cassandra a Amazon Keyspaces casi en tiempo real.

Para determinar las escrituras y actualizaciones que se van a propagar a la base de datos de destino, CQLReplicator escanea el rango de tokens de Apache Cassandra y utiliza una AWS Glue tarea para eliminar los eventos duplicados y aplicar las escrituras y actualizaciones directamente a Amazon Keyspaces.

- **Captura de datos de cambios (CDC):** si conoce la CDC de Cassandra, otra opción para implementar una migración híbrida es utilizar la característica de CDC integrada en Apache Cassandra, que permite capturar los cambios copiando el registro de confirmaciones en un directorio de CDC independiente.

Para ello, puede replicar los cambios en los datos en Amazon Keyspaces, lo que convierte a la CDC en una opción alternativa para los escenarios de migración de datos.

Si no necesita coherencia de lectura tras escritura, puede utilizar CQLReplicator o una canalización de CDC para migrar los datos de Apache Cassandra a Amazon Keyspaces en función de sus preferencias y de su familiaridad con las herramientas utilizadas en cada solución. Servicios de AWS El uso de estos métodos para migrar datos prácticamente en tiempo real puede considerarse un enfoque de migración híbrido que ofrece una alternativa a la migración en línea.

Esta estrategia se considera un enfoque híbrido porque, además de las opciones descritas en este tema, deben implementarse algunos pasos del proceso de migración en línea, por ejemplo, la copia de los datos históricos y las estrategias de migración de aplicaciones que se analizan en el tema [migración en línea](#).

En las siguientes secciones, se explican las opciones de migración híbrida con más detalle.

Temas

- [Migración de datos con CQLReplicator](#)
- [Migración de datos mediante la captura de datos de cambios \(CDC\)](#)

Migración de datos con CQLReplicator

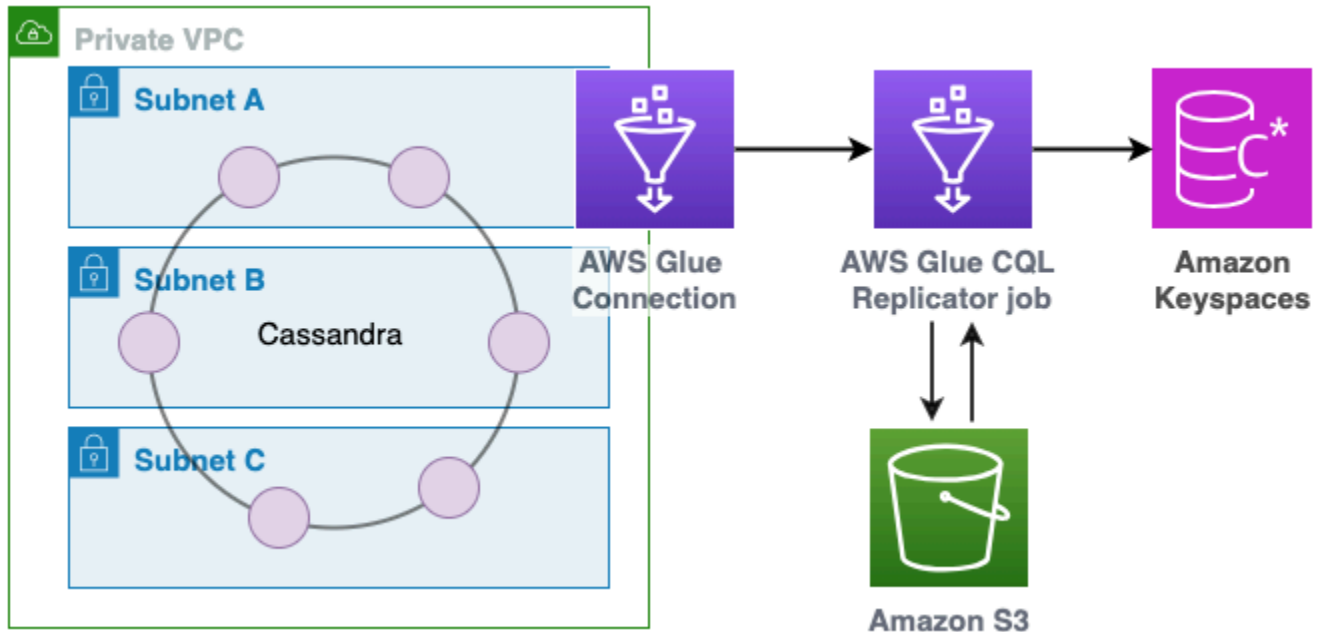
Con [CQLReplicator](#), puede leer los datos de Apache Cassandra prácticamente en tiempo real escaneando de forma inteligente el rango de tokens de Cassandra mediante consultas CQL. CQLReplicator no utiliza la CDC de Cassandra, sino que implementa una estrategia de almacenamiento en caché para reducir las penalizaciones de rendimiento derivadas de los análisis completos.

Para reducir el número de escrituras en el destino, CQLReplicator elimina automáticamente los eventos de replicación duplicados. Con CQLReplicator, puede ajustar la replicación de los cambios de la base de datos de origen a la base de datos de destino, lo que permite una migración de datos casi en tiempo real de Apache Cassandra a Amazon Keyspaces.

En el siguiente diagrama se muestra la arquitectura típica de un trabajo de CQLReplicator en el que se utiliza AWS Glue.

1. Para permitir el acceso a Apache Cassandra que se ejecuta en una VPC privada, configure AWS Glue una conexión con el tipo de conexión Red.

2. Para eliminar duplicados y habilitar el almacenamiento en caché de claves con el trabajo de CQLReplicator, configure Amazon Simple Storage Service (Amazon S3).
3. El trabajo de CQLReplicator transmite los cambios verificados en la base de datos de origen directamente a Amazon Keyspaces.



[Para obtener más información sobre el proceso de migración mediante CQLReplicator, consulte la siguiente publicación en el blog de AWS bases de datos Migración de cargas de trabajo de Cassandra a Amazon Keyspaces mediante CQLReplicator y la AWS guía prescriptiva Migración de cargas de trabajo de Apache Cassandra a Amazon Keyspaces mediante AWS Glue](#)

Migración de datos mediante la captura de datos de cambios (CDC)

Si ya conoce la configuración de una canalización de captura de datos de cambios (CDC) con [Debezium](#), puede usar esta opción para migrar datos a Amazon Keyspaces como alternativa al uso de CQLReplicator. Debezium es una plataforma distribuida de código abierto para la CDC, diseñada para supervisar una base de datos y capturar los cambios en las filas de manera fiable.

El [conector de Debezium para Apache Cassandra](#) carga los cambios en Amazon Managed Streaming para Apache Kafka (Amazon MSK) de modo que los consumidores intermedios puedan consumirlos, procesarlos y escribir a su vez los datos en Amazon Keyspaces. Para obtener más información, consulte [Guidance for continuous data migration from Apache Cassandra to Amazon Keyspaces](#).

Si quiere solucionar cualquier posible problema de coherencia de datos, puede implementar un proceso con Amazon MSK en el que un consumidor compare las claves o particiones de Cassandra con las de Amazon Keyspaces.

Para implementar esta solución correctamente, le recomendamos que tenga en cuenta lo siguiente.

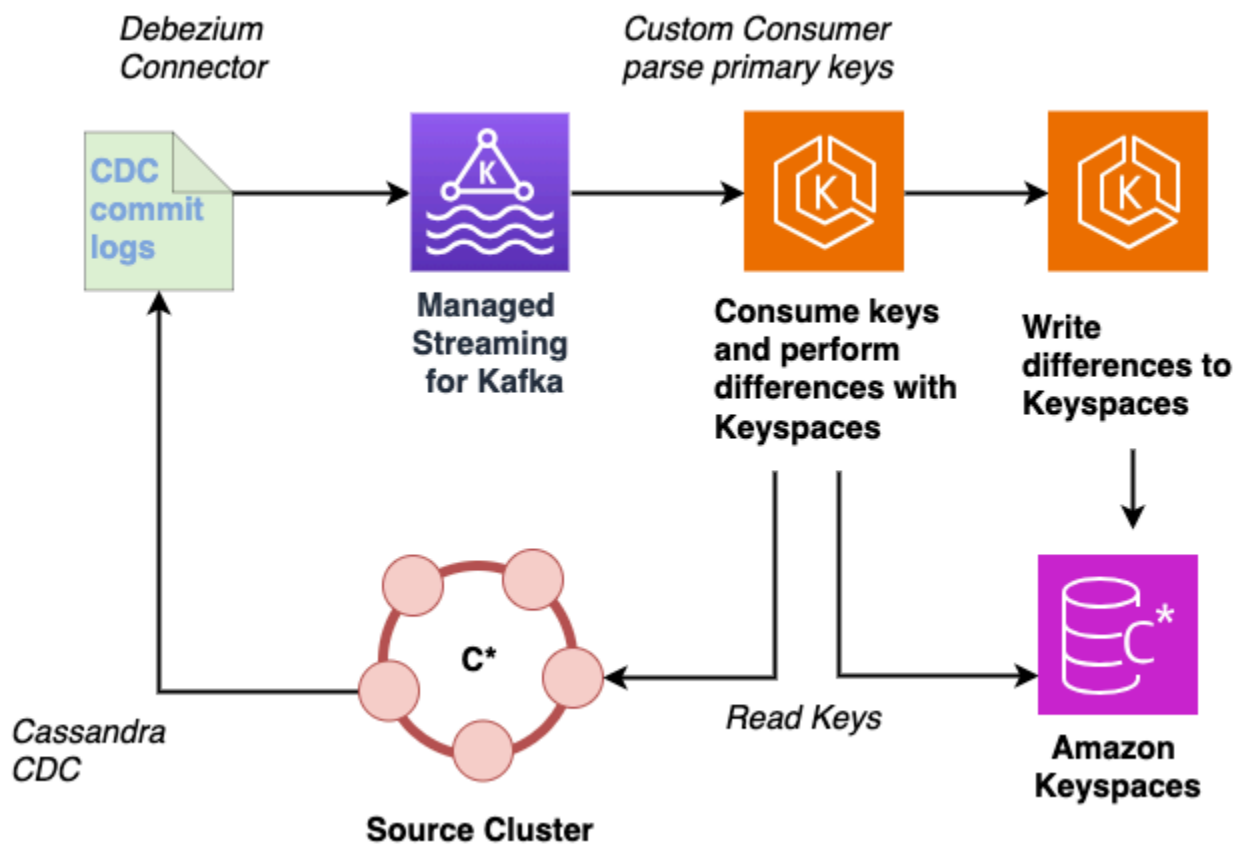
- Cómo analizar el registro de confirmaciones de la CDC; por ejemplo, cómo eliminar los eventos duplicados.
- Cómo mantener el directorio de la CDC; por ejemplo, cómo eliminar los registros antiguos.
- Cómo administrar los errores parciales en Apache Cassandra; por ejemplo, si una escritura solo se realiza correctamente en una de cada tres réplicas.
- Cómo administrar la asignación de recursos, por ejemplo, aumentar el tamaño de la instancia para que admita los requisitos adicionales de CPU, memoria, disco y E/S para el proceso de CDC que se produce en un nodo.

Este patrón trata los cambios de Cassandra como “indicios” de que una clave puede haber cambiado con respecto a su estado anterior. Para determinar si hay cambios que propagar a la base de datos de destino, primero debe leer el clúster de Cassandra de origen mediante una operación `LOCAL_QUORUM` para recibir los registros más recientes y, a continuación, escribirlos en Amazon Keyspaces.

En el caso de eliminaciones o actualizaciones de rangos, es posible que deba realizar una comparación con la totalidad de la partición para determinar qué eventos de escritura o actualización deben escribirse en la base de datos de destino.

En los casos en que las escrituras no tengan idempotencia, también debe compararlas con lo que ya está en la base de datos de destino antes de que se efectúen en Amazon Keyspaces.

En el siguiente diagrama, se muestra la arquitectura típica de una canalización de CDC con Debezium y Amazon MSK.



Selección de la herramienta adecuada para cargar o migrar datos por lotes a Amazon Keyspaces

En esta sección, revisará las diferentes herramientas que puede utilizar para cargar o migrar datos por lotes a Amazon Keyspaces y aprenderá a seleccionar la herramienta correcta en función de sus necesidades. Además, en esta sección se proporciona información general y casos de uso de los tutoriales paso a paso disponibles que muestran cómo importar datos a Amazon Keyspaces.

Para revisar las estrategias disponibles para migrar cargas de trabajo de Apache Cassandra a Amazon Keyspaces, consulte [the section called “Migración desde Cassandra”](#).

- Herramientas de migración
 - Con la [calculadora de precios de Amazon Keyspaces \(para Apache Cassandra\)](#) disponible en Github, puede estimar los costes mensuales de Amazon Keyspaces en función de su carga de trabajo actual de Apache Cassandra. Introduzca las métricas del resultado de estado de Cassandra nodetool y de la configuración sin servidor prevista para Amazon Keyspaces a fin de comparar los costes directos entre las dos soluciones. Tenga en cuenta que esta calculadora

se centra únicamente en los costes operativos de Amazon Keyspaces en comparación con su implementación actual de Cassandra. No incluye los factores del costo total de propiedad (TCO), como el mantenimiento de la infraestructura, los gastos operativos o los costos de soporte de Cassandra.

- Proxy de escritura doble de ZDM para la migración de Amazon Keyspaces: el proxy de escritura doble de ZDM, disponible [en Github](#), permite la migración sin tiempo de inactividad de Apache Cassandra a Amazon Keyspaces.
- CQLReplicator: CQLReplicator es una utilidad de código abierto disponible en [Github](#) que le ayuda a migrar datos de Apache Cassandra a Amazon Keyspaces casi en tiempo real.

Para obtener más información, consulte [the section called “CQLReplicator”](#).

- Para obtener más información sobre cómo usar Amazon Managed Streaming para Apache Kafka para implementar un proceso de [migración en línea](#) con escrituras duales, consulte [Guidance for continuous data migration from Apache Cassandra to Amazon Keyspaces](#).
- Para migraciones grandes, 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 [the section called “Migración sin conexión”](#).
- Para obtener información sobre cómo utilizar el conector Apache Cassandra Spark para escribir datos en Amazon Keyspaces, consulte [the section called “Integración con Apache Spark”](#).
- 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”](#).
- También puede usar DataStax Bulk Loader for Apache Cassandra para cargar datos en Amazon Keyspaces mediante dsbulk el comando. [DSBulk ofrece capacidades de importación más sólidas que cqlsh y está disponible en el repositorio. GitHub](#) Para obtener instrucciones paso a paso, consulte [the section called “Carga de datos con DSBulk”](#).

Consideraciones generales para la carga de datos a Amazon Keyspaces

- Divida la carga de datos en componentes más pequeños.

Considere las siguientes unidades de migración y su huella potencial en términos de tamaño de datos en bruto. Cargar 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 cargar 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.

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

En este tutorial le guiaremos en el proceso de migración de datos de Apache Cassandra a Amazon Keyspaces utilizando el comando `cqlsh COPY FROM`. El comando `cqlsh COPY FROM` resulta útil para cargar rápida y fácilmente pequeños conjuntos de datos a Amazon Keyspaces con fines académicos o de prueba. Para obtener más información acerca de cómo migrar cargas de trabajo de producción, consulte [the section called “Migración sin conexión”](#). En este tutorial, completará los siguientes pasos:

Requisitos previos: configure una AWS cuenta con credenciales, cree un archivo de almacén de confianza JKS para el certificado y configúrelo para conectarse `cqlsh` a Amazon Keyspaces.

1. Creación del CSV de origen y la tabla de destino: prepare un archivo CSV como datos de origen y cree el espacio de claves y la tabla de destino en Amazon Keyspaces.
2. Preparación de los datos: asigne al azar los datos del archivo CSV y analícelos para determinar el tamaño medio y máximo de las filas.

3. Definición de la capacidad de rendimiento: calcule las unidades de capacidad de escritura (WCU) requeridas en función del tamaño de los datos y el tiempo de carga deseado y configure la capacidad aprovisionada de la tabla.
4. Configuración de los parámetros `cqlsh`: determine los valores óptimos para los parámetros `cqlsh COPY FROM` como `INGESTRATE`, `NUMPROCESSES`, `MAXBATCHSIZE` y `CHUNKSIZE` para distribuir la carga de trabajo de manera uniforme.
5. Ejecución del comando **`cqlsh COPY FROM`**: ejecute el comando `cqlsh COPY FROM` para cargar los datos del archivo CSV a la tabla de Amazon Keyspaces y supervise el progreso.

Solución de problemas: resuelva problemas comunes, como solicitudes no válidas, errores del analizador, errores de capacidad y errores de `cqlsh` durante el proceso de carga de datos.

Temas

- [Requisitos previos: pasos que debe completar antes de poder cargar datos mediante `cqlsh COPY FROM`](#)
- [Paso 1: creación del archivo CSV de origen y de una tabla de destino para la carga de datos](#)
- [Paso 2: preparación de los datos de origen para una carga de datos correcta](#)
- [Paso 3: Establecer la capacidad de rendimiento de la tabla](#)
- [Paso 4: Configurar los ajustes de `cqlsh COPY FROM`](#)
- [Paso 5: ejecución del comando `cqlsh COPY FROM` para cargar los datos del archivo CSV a la tabla de destino](#)
- [Resolución de problemas](#)

Requisitos previos: pasos que debe completar antes de poder cargar datos mediante **`cqlsh COPY FROM`**

Para poder comenzar el tutorial, antes debe completar las siguientes tareas.

1. Si aún no lo ha hecho, regístrese Cuenta de AWS siguiendo los pasos que se indican en [the section called “Configuración AWS Identity and Access Management”](#)
2. Cree las credenciales específicas del servicio; para ello, siga los pasos indicados en [the section called “Creación de credenciales específicas del servicio”](#).

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: creación del archivo CSV de origen y de una tabla de destino para la carga de datos

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 T0`, como se muestra en el siguiente ejemplo.

```
cqlsh localhost 9042 -u "username" -p "password" --execute  
"COPY mykeyspace.mytable T0 '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-1.amazonaws.com 9142 -u "111122223333" -p  
"wJa1rXUtnFEMI/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: preparación de los datos de origen para una carga de datos correcta

La preparación de los datos de origen para una transferencia eficaz es un proceso de dos pasos. En primer lugar, asigna al azar los datos. En segundo lugar, analiza los datos para determinar los

valores apropiados del parámetro `cqlsh` y los ajustes necesarios de la tabla para garantizar que la carga de datos se realice correctamente.

Asignación al azar 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 “Estimación 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 sobre 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 WCU
```

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 WCU
```

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: ejecución del comando `cqlsh COPY FROM` para cargar los datos del archivo CSV a la tabla de destino

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.

Resolución 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 tutorial paso a paso le guía a través de la migración de datos de Apache Cassandra a Amazon Keyspaces mediante el cargador DataStax masivo (DSBulk) disponible en [GitHub](#). El uso de DSBulk resulta útil para cargar conjuntos de datos en Amazon Keyspaces con fines académicos o de prueba. Para obtener más información acerca de cómo migrar cargas de trabajo de producción, consulte [the section called “Migración sin conexión”](#). En este tutorial, completará los siguientes pasos:

Requisitos previos: configurar una AWS cuenta con credenciales, crear un archivo de almacén de confianza de JKS para el certificado, configurar `cqlsh`, descargar e instalar DSBulk y configurar un archivo `application.conf`

1. Creación del CSV de origen y la tabla de destino: prepare un archivo CSV como datos de origen y cree el espacio de claves y la tabla de destino en Amazon Keyspaces.
2. Preparación de los datos: asigne al azar los datos del archivo CSV y analícelos para determinar el tamaño medio y máximo de las filas.
3. Definición de la capacidad de rendimiento: calcule las unidades de capacidad de escritura (WCU) requeridas en función del tamaño de los datos y el tiempo de carga deseado y configure la capacidad aprovisionada de la tabla.

4. Configure los ajustes de DSBulk: cree un archivo de configuración de DSBulk con ajustes como la autenticación SSL/TLS, el nivel de coherencia y el tamaño del grupo de conexiones.
5. Ejecución del comando de carga de DSBulk: ejecute el comando de carga de DSBulk para cargar los datos del archivo CSV a la tabla de Amazon Keyspaces y supervise el progreso.

Temas

- [Requisitos previos: pasos que debe completar antes de poder cargar datos mediante DSBulk](#)
- [Paso 1: creación del archivo CSV de origen y de una tabla de destino para la carga de datos con DSBulk](#)
- [Paso 2: preparación de los datos para cargarlos con DSBulk](#)
- [Paso 3: establecimiento de la capacidad de rendimiento de la tabla de destino](#)
- [Paso 4: configuración de los ajustes de DSBulk para cargar los datos del archivo CSV a la tabla de destino](#)
- [Paso 5: ejecución del comando load de DSBulk para cargar los datos del archivo CSV a la tabla de destino](#)

Requisitos previos: pasos que debe completar antes de poder cargar datos mediante DSBulk

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 “Configuración AWS Identity and Access Management”](#)
2. Cree las credenciales; para ello, siga los pasos indicados en [the section called “Cree credenciales de IAM para la autenticación AWS”](#).
3. Cree un archivo de almacén de confianza de JKS.
 - a. Descargue los siguientes certificados digitales y guarde los archivos localmente o en su directorio personal.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4

5. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todas las Regiones de AWS a certificados emitidos en virtud de Amazon Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

- b. Convierta los certificados digitales en archivos TrustStore y agréguelos al almacén de claves.

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

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

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

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

openssl x509 -outform der -in sf-class2-root.crt -out temp_file.der
```

```
keytool -import -alias cassandra -keystore cassandra_truststore.jks -file
temp_file.der
```

En el último paso, debe crear una contraseña para el almacén de claves y confiar en cada certificado. El comando interactivo tiene el siguiente aspecto.

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

Extensions:

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

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

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

```

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

Extensions:

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

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

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

Trust this certificate? [no]: yes
Certificate was added to keystore
Enter keystore password:
Owner: CN=Amazon Root CA 3, O=Amazon, C=US
Issuer: CN=Amazon Root CA 3, O=Amazon, C=US
Serial number: 66c9fd5749736663f3b0b9ad9e89e7603f24a

```

```

Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
Certificate fingerprints:
    SHA1: 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E
    SHA256:
    18:CE:6C:FE:7B:F1:4E:60:B2:E3:47:B8:DF:E8:68:CB:31:D0:2E:BB:3A:DA:27:15:69:F5:03:43:B4
Signature algorithm name: SHA256withECDSA
Subject Public Key Algorithm: 256-bit EC (secp256r1) key
Version: 3

Extensions:

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

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

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

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

```

Subject Public Key Algorithm: 384-bit EC (secp384r1) key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true

BasicConstraints:[

CA:true

PathLen:2147483647

]

#2: ObjectId: 2.5.29.15 Criticality=true

KeyUsage [

DigitalSignature

Key_CertSign

Crl_Sign

]

#3: ObjectId: 2.5.29.14 Criticality=false

SubjectKeyIdentifier [

KeyIdentifier [

0000: D3 EC C7 3A 65 6E CC E1 DA 76 9A 56 FB 9C F3 86 ...:en...v.V....

0010: 6D 57 E5 81 mW..

]

]

Trust this certificate? [no]: yes

Certificate was added to keystore

Enter keystore password:

Owner: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies, Inc.", C=US

Issuer: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies, Inc.", C=US

Serial number: 0

Valid from: Tue Jun 29 17:39:16 UTC 2004 until: Thu Jun 29 17:39:16 UTC 2034

Certificate fingerprints:

SHA1: AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A

SHA256:

14:65:FA:20:53:97:B8:76:FA:A6:F0:A9:95:8E:55:90:E4:0F:CC:7F:AA:4F:B7:C2:C8:67:75:21:FB

Signature algorithm name: SHA1withRSA (weak)

Subject Public Key Algorithm: 2048-bit RSA key

Version: 3

Extensions:

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

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

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

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

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

4. 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.

 Note

Es posible que la versión que se muestra en este tutorial no sea la última disponible. Antes de descargar DSBulk, consulte la [página de descargas de DataStax Bulk Loader](#) para ver la versión más reciente y actualice el número de versión en los siguientes comandos según corresponda.

- 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 [GitHub](#) y 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: creación del archivo CSV de origen y de una tabla de destino para la carga de datos con DSBulk

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  
> keyspace_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-1.amazonaws.com 9142 -u "111122223333" -p  
"wJa1rXUtnFEMI/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 “Comprobación del estado de creación del espacio 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: preparación de los datos para cargarlos con DSBulk

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 “Estimación 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: establecimiento de la capacidad de rendimiento de la tabla de destino

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 sobre 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.

$$\text{ROUNDUP}(3000 / 1024) = \text{ROUNDUP}(2.93) = 3 \text{ WCU}$$

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: configuración de los ajustes de **DSBulk** para cargar los datos del archivo CSV a la tabla de destino

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 de acceso programático”](#).
 - 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-1.amazonaws.com`, entonces establezca el centro de datos local en `us-east-1`. Para ver todas las opciones disponibles Regiones de AWS, consulte [the section called “Puntos de conexión de servicio”](#). Para evitar réplicas, establezca `slow-replica-avoidance` en `false`.

- c. *SSLEngineFactory*— Para configurarlo SSL/TLS, inicialícelo *SSLEngineFactory* añadiendo una sección en el archivo de configuración con una sola línea que especifique la clase con `class = DefaultSslEngineFactory` la que. 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-1.amazonaws.com:9142" ]
  advanced.auth-provider {
    class = PlainTextAuthProvider
    username = "ServiceUserName"
    password = "ServicePassword"
  }

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

  basic.request {
    consistency = LOCAL_QUORUM
    default-idempotence = true
  }

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

2. 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. Recomendamos deshabilitar el modo por lotes, ya que puede provocar situaciones de sobrecarga de claves y provocar *WriteThrottleEvents*.
 - *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: ejecución del comando **load** de DSBulk para cargar los datos del archivo CSV a la tabla de destino

En el último paso de este tutorial, cargará 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 “Configure los modos read/write de capacidad”](#).

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.

Configuración AWS Identity and Access Management

Inscríbase para obtener una Cuenta de AWS

Para empezar AWS, necesitas un Cuenta de AWS. Para obtener información sobre cómo crear un Cuenta de AWS, consulte [Cómo empezar con un Cuenta de AWS](#) en la Guía de AWS Account Management referencia.

Configuración de Amazon Keyspaces

El acceso a los recursos de Amazon Keyspaces se administra con [IAM](#). Con IAM, puede asociar políticas a usuarios, roles e identidades federadas de IAM que concedan permisos de lectura y escritura a recursos específicos en Amazon Keyspaces.

Para empezar a conceder permisos a una identidad de IAM, puede usar una de las políticas administradas de AWS 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 “AWS políticas gestionadas”](#).

Para limitar el alcance de las acciones que puede realizar una identidad de IAM o limitar los recursos a los que la identidad puede acceder, puede crear una política personalizada que utilice la política

administrada `AmazonKeyspacesFullAccess` como plantilla y eliminar todos los permisos que no necesite. También puede limitar el acceso a tablas o espacios de claves determinados. Para obtener más información acerca de 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](#)

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 Consola de administración de AWS acceso, consulte [Control del acceso de los usuarios de IAM Consola de administración de AWS en la](#) Guía del usuario de IAM.

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

- Crear, eliminar y administrar espacios de claves y tablas.
- Supervisar las métricas importantes de una tabla en la pestaña Supervisar de una tabla:
 - Tamaño de la tabla facturable (bytes)
 - Métricas de capacidad
- Ejecute consultas mediante el editor CQL o mediante la [AWS CloudShell experiencia integrada](#), por ejemplo, inserte, actualice y elimine datos.
- Connect a Amazon Keyspaces directamente mediante AWS CloudShell el soporte CQLSH integrado.
- Cambiar la configuración del particionador de la cuenta.
- Consultar las métricas de rendimiento y errores de la cuenta en el panel de control.

Conéctese a Amazon Keyspaces mediante AWS CloudShell desde la consola

AWS CloudShell proporciona una forma simplificada de conectarse a Amazon Keyspaces directamente desde la consola. La AWS CloudShell integración lo inicializa automáticamente y establece una conexión con un espacio de claves específico. `cqlsh-expansion`

Introducción al AWS CloudShell integración

1. Acceda a la AWS CloudShell integración: desde la consola de Amazon Keyspaces, navegue hasta el espacio clave al que desee conectarse y pulse el botón Conectar a Keyspaces. AWS CloudShell
2. Pre-filled comando de configuración: al seleccionar el AWS CloudShell botón, el sistema proporciona un comando relleno previamente que incluye los parámetros de inicialización y conexión necesarios:

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

3. Haga clic en Ejecutar
4. Espere a que Amazon Keyspaces establezca la conexión: espere un tiempo para que se complete el proceso de configuración y se establezca la conexión con el espacio de claves seleccionado.
5. Ejecute comandos: cuando esté conectado, podrá escribir y ejecutar comandos de CQL directamente en el espacio de teclas seleccionado.

Comandos CQLSH disponibles

La AWS CloudShell integración proporciona acceso a la funcionalidad estándar del CQLSH, incluida la compatibilidad con comandos que van más allá del CQL básico.

- Describa: enumere y vea descripciones detalladas de los espacios clave y las tablas.
- Copiar: exporta el resultado de la consulta a un archivo para su posterior procesamiento.
- Fuente: ejecuta sentencias CQL desde un archivo.
- Navegación por el historial de comandos: desplácese por los comandos anteriores mediante atajos de teclado.

- Ejecución de consultas: ejecuta consultas directamente desde la interfaz del teclado.
- Copia de resultados: copia los resultados como texto simple para usarlos en otras aplicaciones.

Aspectos a tener en cuenta sobre las características

Al utilizar la AWS CloudShell integración, tenga en cuenta las siguientes ventajas y desventajas.

No se admiten las siguientes funciones:

- Sugerencias de sintaxis de CQL y funcionalidad de autocompletado.
- Herramientas de edición como mostrar autocompletar, mostrar información sobre herramientas, buscar, buscar y reemplazar, rehacer, bloquear comentarios, activar el doblado de códigos, alternar comentarios con líneas y deshacer.

Funcionalidad modificada:

- Descargas de CSV: para descargar los resultados en formato CSV, especifique el > operador al que se va a canalizar el resultado del comando:

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

- Visualización en JSON: puede ver los elementos en formato JSON sin necesidad de convertir los datos.
- Terminal-style interfaz: la experiencia funciona como una interfaz de terminal y no como un editor de código con todas las funciones.

La AWS CloudShell integración simplifica el proceso de conexión y, al mismo tiempo, proporciona la funcionalidad CQL esencial para administrar los datos de Amazon Keyspaces directamente desde.

Consola de administración de AWS

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 \(para Apache Cassandra\)](#).

Utilización AWS CloudShell para acceder a Amazon Keyspaces

AWS CloudShell es un shell preautenticado y basado en un navegador que puede iniciar directamente desde la Consola de administración de AWS. Puede ejecutar AWS CLI comandos en los AWS servicios mediante el shell que prefiera (Bash o Z shell). PowerShell AWS CloudShell viene preautenticado con las credenciales de la consola e incluye herramientas comunes para trabajar con AWS los servicios, como AWS CLI Python y Node.js.

Se [inicia AWS CloudShell desde la Consola de administración de AWS](#) consola y las AWS credenciales que utilizaste 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).

AWS CloudShell opciones de integración

Amazon Keyspaces ofrece dos formas de uso: AWS CloudShell

1. AWS CloudShell Experiencia integrada: un método de conexión simplificado disponible directamente desde la consola de Amazon Keyspaces que inicializa automáticamente un espacio de claves específico y `cqlsh-expansion` se conecta a él. Para obtener instrucciones completas sobre el uso de esta experiencia integrada, consulte [the section called “CloudShell integración de consolas”](#)
2. AWS CloudShell Configuración manual: instale y configure manualmente `cqlsh` en AWS CloudShell. Los pasos se describen en las siguientes secciones y proporcionan un mayor control sobre el proceso de configuración y permiten configuraciones personalizadas.

Requisitos previos

Antes de usar Amazon Keyspaces con AWS CloudShell, asegúrate de tener:

- Una AWS cuenta con acceso a Amazon Keyspaces y AWS CloudShell
- Permisos de IAM adecuados para usar ambos servicios
- Conocimientos básicos de Cassandra Query Language (CQL)

¿Cómo empezar con AWS CloudShell

Para acceder a AWS CloudShell:

1. Inicie sesión en Consola de administración de AWS
2. AWS CloudShell Para iniciarlo, selecciona el AWS CloudShell icono de la barra de navegación de la parte superior de la consola o navega hasta <https://console.aws.amazon.com/cloudshell/>
3. Espere a que el AWS CloudShell entorno se inicialice

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 gestionada 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 puede limitar el acceso a espacios de claves y tablas específicos para crear una política personalizada. La siguiente política administrada para Amazon Keyspaces se puede asociar 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 administrada, consulte [the section called “AWS políticas gestionadas”](#).

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

Instalación del cliente cqlsh

AWS CloudShell no viene preinstalado. `cqlsh` Debe instalarlo para interactuar con Amazon Keyspaces. Puede instalar la Keyspaces-specific versión de Amazon `decqlsh`. Para ver las instrucciones de instalación de `cqlsh-expansion`, consulte [the section called “Uso de cqlsh-expansion”](#).

Configuración de certificados TLS

Para conectarse a Amazon Keyspaces mediante una conexión TLS, debe descargar los certificados digitales de Amazon necesarios. Para obtener instrucciones detalladas sobre cómo descargar y configurar el certificado SSL, consulte. [the section called “Configuración manual de conexiones de cqlsh para TLS”](#)

Conexión a Amazon Keyspaces

Puede conectarse a Amazon Keyspaces mediante credenciales de IAM (recomendadas) o credenciales específicas del servicio.

Conectarse con credenciales de IAM (recomendado)

AWS CloudShell ya está autenticado con sus credenciales de IAM. Para conectarse con estas credenciales, puede usar el complemento de autenticación SigV4. Para obtener instrucciones detalladas sobre el uso de las credenciales de IAM con `cqlsh`, consulte. [the section called “Uso de cqlsh-expansion”](#)

Conexión con credenciales específicas del servicio

Si ha creado credenciales específicas de un servicio para Amazon Keyspaces, puede utilizarlas para conectarse. Para obtener información sobre la creación y el uso de credenciales específicas de un servicio, consulte [the section called “Creación de credenciales específicas del servicio”](#)

Uso de CQLSH con Amazon Keyspaces

Cuando está conectado, puede utilizar comandos CQL estándar para interactuar con sus espacios de teclas y tablas. Para obtener información detallada sobre los comandos y operaciones de CQL compatibles, consulte [the section called “Uso de cqlsh-expansion”](#)

Datos persistentes en AWS CloudShell

AWS CloudShell proporciona 1 GB de almacenamiento persistente en su directorio principal. Esto significa que los scripts, los certificados y los archivos de configuración que cree permanecen disponibles en todas las sesiones. Puede crear un archivo de configuración `cqlsh` para simplificar las conexiones.

Con un archivo de configuración, puede ejecutarlo `cqlsh` sin especificar parámetros adicionales. Para obtener más información sobre la creación y el uso `cqlsh` de archivos de configuración, consulte [the section called “Uso de cqlsh-expansion”](#).

Prácticas recomendadas

Al utilizar Amazon Keyspaces con AWS CloudShell, recomendamos las siguientes prácticas recomendadas:

- Cree scripts para tareas repetitivas y guárdelos en su directorio principal persistente.
- Utilice variables de entorno para la información confidencial en lugar de codificarlas de forma rígida.
- Recuerde que es posible que AWS CloudShell las sesiones se detengan después de una inactividad.
- Tenga en cuenta las cuotas y AWS CloudShell limitaciones del servicio.
- Considere la posibilidad de AWS CLI utilizarlos para operaciones automatizadas con Amazon Keyspaces.

Resolución de problemas

Si tienes problemas al conectarte a Amazon Keyspaces desde: AWS CloudShell

- Comprueba que tus permisos de IAM incluyan las acciones de Amazon Keyspaces necesarias.
- Asegúrese de utilizar el punto de enlace correcto para su región.
- Compruebe que los certificados SSL se hayan descargado y referenciado correctamente.
- Compruebe que las credenciales específicas del servicio sean correctas y no estén caducadas.
- Si utiliza un archivo `cqlshrc`, compruebe su formato y sus permisos.

Ejemplo: Cómo interactuar con Amazon Keyspaces mediante AWS CloudShell

Tras lanzarlo AWS CloudShell desde Consola de administración de AWS, 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 ha instalado la `cqlsh-expansion`, consulte [the section called “Uso de cqlsh-expansion”](#) para ver los pasos en detalle.

Note

Al usar la `cqlsh-expansion` entrada AWS CloudShell, no necesita configurar las credenciales antes de realizar llamadas, ya que ya está autenticado en el shell.

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

1. Desde Consola de administración de AWS, puede iniciarlo CloudShell seleccionando las siguientes opciones disponibles en la barra de navegación:
 - Selecciona el CloudShell icono.
 - Empieza a escribir `cloudshell` en el cuadro de búsqueda y, a continuación, selecciona la CloudShell opción.
2. Puede establecer una conexión a Amazon Keyspaces mediante el siguiente comando. Asegúrese de `cassandra.us-east-1.amazonaws.com` reemplazarlo por el punto final 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. Creación de un nuevo espacio de claves con el nombre `mykeyspace`. Puede usar el siguiente comando para hacerlo.

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

4. Para confirmar que se ha creado 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)
```

Creación de credenciales para el acceso programático a Amazon Keyspaces

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 utiliza 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.

Warning

Los usuarios de IAM tienen credenciales de larga duración, lo que supone un riesgo de seguridad. Para ayudar a mitigar este riesgo, le recomendamos que brinde a estos usuarios únicamente los permisos que necesitan para realizar la tarea y que los elimine cuando ya no los necesiten.

- Para mejorar la seguridad, recomendamos crear identidades de IAM que se utilicen en todos los AWS servicios y utilizar credenciales temporales. 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

- [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)

- [Creación y configuración de AWS credenciales para Amazon Keyspaces](#)

Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces

Service-specific las credenciales son similares al nombre de usuario y la contraseña tradicionales que Cassandra utiliza para la autenticación y la administración del acceso. Service-specific las credenciales permiten a los usuarios de IAM acceder a un servicio específico AWS. Estas credenciales de larga duración no se pueden utilizar para acceder a otros AWS servicios. Están asociadas a un usuario de IAM específico y no pueden utilizarlas otros usuarios de IAM.

Important

Service-specific las credenciales son credenciales de larga duración asociadas a un usuario de IAM específico y solo se pueden usar 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 mediante credenciales temporales, debe utilizar la [AWS autenticación con el complemento de autenticación SigV4 para Amazon Keyspaces](#).

Utilice uno de los siguientes procedimientos para generar credenciales específicas del servicio.

Console

Creación de credenciales específicas del servicio mediante la consola

1. Inicie sesión en Consola de administración de AWS 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 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.

CLI

Cree credenciales específicas del servicio mediante el AWS CLI

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

1. Descarga la etiqueta. AWS CLI <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 AWS credenciales para Amazon Keyspaces

Para acceder a Amazon Keyspaces mediante programación con el AWS SDK o con los AWS CLI controladores de cliente de Cassandra y el complemento SigV4, necesita un usuario de IAM con claves de acceso. Cuando utiliza Amazon Keyspaces mediante programación, proporciona sus claves de AWS acceso para AWS poder verificar su identidad en las llamadas programáticas. Las claves de acceso constan de un identificador de clave de acceso (por ejemplo, AKIAIOSFODNN7EXAMPLE) y una clave de acceso secreta (por ejemplo, /b CYEXAMPLEKEY). wJalrXUtnFEMI/K7MDENG PxRfi En este tema se explican los pasos necesarios en este proceso.

Las prácticas recomendadas de seguridad proponen crear usuarios de IAM con permisos limitados y asociar roles de IAM a los permisos necesarios para realizar tareas específicas. De este modo, los usuarios de IAM pueden asumir temporalmente esos roles de IAM para llevar a cabo las tareas necesarias. Por ejemplo, los usuarios de IAM de su cuenta que usen la consola de Amazon Keyspaces pueden cambiar a un rol para utilizar temporalmente los permisos del rol en la consola. Los usuarios renuncian a sus permisos originales y adoptan los permisos asignados al rol. Cuando el usuario deja de utilizar el rol, sus permisos originales se restablecen. Las credenciales que emplean los usuarios para asumir el rol son temporales. Por el contrario, los usuarios de IAM tienen credenciales de larga duración, lo que supone un riesgo de seguridad si, en lugar de asumir roles, tienen permisos asignados directamente a ellos. Para ayudar a mitigar este riesgo, le recomendamos que brinde a estos usuarios únicamente los permisos que necesitan para realizar la tarea y que los elimine cuando ya no los necesiten. Para obtener más información, consulte [Situaciones habituales con los roles: usuarios, aplicaciones y servicios](#) en la Guía del usuario de IAM.

Temas

- [Credenciales requeridas por el AWS CLI AWS SDK o el complemento SiGv4 de Amazon Keyspaces para los controladores de cliente de Cassandra](#)
- [Creación de credenciales temporales para conectarse a Amazon Keyspaces mediante un rol de IAM y el complemento SigV4](#)
- [Cree un usuario de IAM para acceder mediante programación a Amazon Keyspaces en su cuenta AWS](#)
- [Creación de nuevas claves de acceso para un usuario de IAM](#)

- [Almacenamiento de claves de acceso para el acceso programático](#)

Credenciales requeridas por el AWS CLI AWS SDK o el complemento SiGv4 de Amazon Keyspaces para los 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 AWS acceso asociada a un usuario o 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 “Creación 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 credenciales temporales para conectarse a Amazon Keyspaces mediante un rol de IAM y el complemento SigV4

La forma recomendada de acceder mediante programación a Amazon Keyspaces es con [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 las funciones de IAM en lugar de las claves de acceso a largo plazo, consulte [Cambiar a una función de IAM \(AWS API\)](#).

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 consola de IAM Consola de administración de AWS y ábrala 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 usar credenciales de seguridad temporales en el código, llama mediante programación a una AWS Security Token Service API similar AssumeRole y extrae las credenciales y el token de sesión resultantes del rol de IAM que creaste en el paso anterior. A continuación, utiliza esos valores como credenciales para las siguientes llamadas 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 la 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 esté trabajando. Puedes encontrar la documentación de todos los AWS SDK en la [página principal de AWS documentación](#), en la sección SDK y kits de herramientas.

Cree un usuario de IAM para acceder mediante programación a Amazon Keyspaces en su cuenta AWS

Para obtener credenciales para el acceso programático a Amazon Keyspaces con AWS CLI el SDK o AWS 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 las Consola de administración de AWS Herramientas para Windows o mediante una operación PowerShell de API. AWS CLI AWS Si crea el usuario en Consola de administración de AWS, 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 de IAM, consulte [Permisos necesarios para acceder a los recursos de IAM](#).

Console

Creación de un usuario de IAM con acceso programático (consola)

1. Inicie sesión en la consola de IAM Consola de administración de AWS y ábrala 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 de 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 se

distingue por caso. 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 AWS administradas 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 “AWS políticas gestionadas”](#).

Con fines de prueba y para seguir los tutoriales de conexión, seleccione la política AmazonKeyspacesReadOnlyAccess para el nuevo usuario de IAM. 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 de permisos personalizadas, debe asociar sus políticas a los roles y dejar que los usuarios asuman temporalmente los roles pertinentes.

Elija 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 Mostrar junto a la contraseña y la clave de acceso. Para guardar las claves de acceso, elija 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.

CLI

Creación de un usuario de IAM con acceso programático (AWS CLI)

1. Cree un usuario con el siguiente AWS CLI código.
 - [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](#)
 - Herramientas para 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 asociando 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 actualizar 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 consola de IAM Consola de administración de AWS y ábrala en <https://console.aws.amazon.com/iam/>.
2. En el panel de navegación, seleccione 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 Claves de acceso, en Prácticas recomendadas y alternativas de las claves de acceso, selecciona el caso de uso Otros. Haga clic en Siguiente, introduzca la información opcional necesaria y seleccione Crear clave de acceso.

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

- ID de clave de acceso: AKIAIOSFODNN7EXAMPLE
- Clave de acceso secreta: wJalrXUtnFEMI/K7MDENG /b PxRfi CYEXAMPLEKEY

Note

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

Tenga en cuenta las siguientes prácticas recomendadas para el key pair que ha creado.

- Nunca guarde la clave de acceso en texto plano, en un repositorio de código o en código.
- Desactive o elimine las claves de acceso cuando ya no las necesite.
- Habilite los permisos con privilegios mínimos.
- Gire las claves de acceso con regularidad.

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

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

Almacenamiento de claves de acceso para el acceso programático

Como práctica recomendada, le sugerimos que no inserte claves de acceso directamente en el código. AWS CLI y las herramientas de línea de AWS comandos le permiten colocar las claves de acceso en ubicaciones conocidas para no tener que guardarlas en el 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 la 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 la 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”](#). Ten en cuenta que esto incluye las opciones predeterminadas Región de AWS. 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>
$ export AWS_DEFAULT_REGION=us-east-1
```

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>
C:\> setx AWS_DEFAULT_REGION us-east-1
```

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>"
PS C:\> $Env:AWS_DEFAULT_REGION="us-east-1"
```

Si establece una variable de entorno en la PowerShell línea de comandos, como se muestra en los ejemplos anteriores, guardará el valor solo durante la sesión actual. Para que la configuración de la variable de entorno sea persistente en todas las sesiones PowerShell y en las de Command Prompt, guárdela mediante la aplicación Sistema del Panel de control. Como alternativa, puede configurar la variable para todas las PowerShell sesiones futuras añadiéndola a su PowerShell perfil. Consulte la [PowerShell documentación](#) para obtener más información sobre cómo almacenar variables de entorno o cómo conservarlas en todas las sesiones.

Puntos de conexión de servicio para Amazon Keyspaces

Temas

- [Puertos y protocolos](#)
- [Puntos de enlace globales](#)
- [AWS GovCloud \(US\) Region Puntos de conexión de FIPS](#)
- [Puntos de conexión de regiones de China](#)
- [Puntos de enlace de Streams](#)
- [Conexión a puntos finales de doble pila](#)

Puertos y protocolos

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

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

Acceso programático	Puerto	Protocolo
CQLSH	9.142	TLS
Controlador de Cassandra	9142	TLS
AWS CLI	443	HTTPS
AWS SDK	443	HTTPS

Para las conexiones TLS, Amazon Keyspaces utiliza certificados emitidos bajo Amazon Trust Services (Amazon Root CA 1—4) para autenticarse en el 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 enlace globales

Amazon Keyspaces admite puntos de enlace públicos IPv4 e IPv6. Puede elegir entre puntos de enlace IPv4 y puntos de enlace de doble pila. Los puntos finales utilizan la siguiente convención de nomenclatura, que se puede sustituir por otra que esté disponible en la *us-east-1* tabla. Región de AWS

- Puntos de conexión IPv4: `cassandra.us-east-1.amazonaws.com`
- Dual-stack puntos finales — `cassandra.us-east-1.api.aws`

Para obtener más información sobre los puntos finales de doble pila y cómo configurar las conexiones, consulte [the section called “Conexión a puntos finales de doble pila”](#)

Amazon Keyspaces está disponible en las siguientes regiones.

Nombre de la región	Región	Punto de conexión	Protocolo	
Este de EE. UU. (Ohio)	us-east-2	<code>cassandra.us-east-2.amazonaws.com</code>	HTTPS y TLS	
		<code>cassandra.us-east-2.api.aws</code>	HTTPS y TLS	
Este de EE. UU. (Norte de Virginia)	us-east-1	<code>cassandra.us-east-1.amazonaws.com</code>	HTTPS y TLS	
		<code>cassandra-fips.us-east-1.api.aws</code>		
		<code>cassandra-fips.us-east-1.amazonaws.com</code>	HTTPS y TLS	
		<code>cassandra.us-east-1.api.aws</code>	HTTPS y TLS	
Oeste de EE. UU.	us-west-1	<code>cassandra.us-west-1.amazonaws.com</code>	HTTPS y TLS	

Nombre de la región	Región	Punto de conexión	Protocolo	
(Norte de California)		cassandra.us-west-1.api.aws	HTTPS y TLS	
Oeste de EE. UU. (Oregón)	us-west-2	cassandra.us-west-2.amazonaws.com cassandra-fips.us-west-2.api.aws cassandra-fips.us-west-2.amazonaws.com cassandra.us-west-2.api.aws	HTTPS y TLS HTTPS y TLS HTTPS y TLS HTTPS y TLS	
África (Ciudad del Cabo)	af-south-1	cassandra.af-south-1.amazonaws.com cassandra.af-south-1.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Hong Kong)	ap-east-1	cassandra.ap-east-1.amazonaws.com cassandra.ap-east-1.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Malasia)	ap-southeast-5	cassandra.ap-southeast-5.amazonaws.com cassandra.ap-southeast-5.api.aws	HTTPS y TLS HTTPS y TLS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Asia-Pacífico (Mumbai)	ap-south-1	cassandra.ap-south-1.amazonaws.com cassandra.ap-south-1.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Seúl)	ap-northeast-2	cassandra.ap-northeast-2.amazonaws.com cassandra.ap-northeast-2.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Singapur)	ap-southeast-1	cassandra.ap-southeast-1.amazonaws.com cassandra.ap-southeast-1.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Sídney)	ap-southeast-2	cassandra.ap-southeast-2.amazonaws.com cassandra.ap-southeast-2.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Tailandia)	ap-southeast-7	cassandra.ap-southeast-7.amazonaws.com cassandra.ap-southeast-7.api.aws	HTTPS y TLS HTTPS y TLS	
Asia-Pacífico (Tokio)	ap-northeast-1	cassandra.ap-northeast-1.amazonaws.com cassandra.ap-northeast-1.api.aws	HTTPS y TLS HTTPS y TLS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Canadá (centro)	ca-central-1	cassandra.ca-central-1.amazonaws.com cassandra.ca-central-1.api.aws	HTTPS y TLS HTTPS y TLS	
Europa (Fráncfort)	eu-central-1	cassandra.eu-central-1.amazonaws.com cassandra.eu-central-1.api.aws	HTTPS y TLS HTTPS y TLS	
Europa (Irlanda)	eu-west-1	cassandra.eu-west-1.amazonaws.com cassandra.eu-west-1.api.aws	HTTPS y TLS HTTPS y TLS	
Europa (Londres)	eu-west-2	cassandra.eu-west-2.amazonaws.com cassandra.eu-west-2.api.aws	HTTPS y TLS HTTPS y TLS	
Europa (París)	eu-west-3	cassandra.eu-west-3.amazonaws.com cassandra.eu-west-3.api.aws	HTTPS y TLS HTTPS y TLS	
Europa (Estocolmo)	eu-north-1	cassandra.eu-north-1.amazonaws.com cassandra.eu-north-1.api.aws	HTTPS y TLS HTTPS y TLS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Medio Oriente (Baréin)	me-south-1	cassandra.me-south-1.amazonaws.com cassandra.me-south-1.api.aws	HTTPS y TLS HTTPS y TLS	
Medio Oriente (EAU)	me-central-1	cassandra.me-central-1.amazonaws.com cassandra.me-central-1.api.aws	HTTPS y TLS HTTPS y TLS	
América del Sur (São Paulo)	sa-east-1	cassandra.sa-east-1.amazonaws.com cassandra.sa-east-1.api.aws	HTTPS y TLS HTTPS y TLS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com cassandra.us-gov-east-1.api.aws	HTTPS y TLS HTTPS y TLS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com cassandra.us-gov-west-1.api.aws	HTTPS y TLS HTTPS y TLS	

AWS GovCloud (US) Region Puntos de conexión de FIPS

Puntos de conexión FIPS disponibles en la AWS GovCloud (US) Region. Amazon Keyspaces es compatible con los puntos de enlace FIPS de IPv4 e IPv6. Puede elegir entre puntos de enlace IPv4

y puntos de enlace de doble pila. Para obtener más información, consulte [Amazon Keyspaces en la Guía del usuario de AWS GovCloud \(US\)](#).

Nombre de la región	Region	Punto final FIPS IPv4	Terminal FIPS de doble pila	Protocolo
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com	cassandra.us-gov-east-1.api.aws	HTTPS y TLS
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com	cassandra.us-gov-west-1.api.aws	HTTPS y TLS

Puntos de conexión de regiones de China

Amazon Keyspaces admite puntos de enlace IPv4 en las regiones de China. 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 conexión	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

Puntos de enlace de Streams

Amazon Keyspaces CDC streams está disponible de la siguiente Regiones de AWS manera. En esta tabla se muestran los puntos finales de servicio de doble pila disponibles para cada región. Para obtener más información al respecto Amazon Keyspaces CDC streams, consulte [the section called “Acceda a los puntos de enlace de transmisión de los CDC”](#).

Nombre de la región	Región	Punto de conexión	Protocolo	
Este de EE. UU. (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS	
Este de EE. UU. (Norte de Virginia)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
Oeste de EE. UU. (Norte de California)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	
Oeste de EE. UU. (Oregón)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
África (Ciudad del Cabo)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	
Asia-Pacífico	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
(Hong Kong)				
Asia-Pacífico (Malasia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Asia-Pacífico (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	
Asia-Pacífico (Seúl)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	
Asia-Pacífico (Singapur)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	
Asia-Pacífico (Sídney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS	
Asia-Pacífico (Tailandia)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS	
Asia-Pacífico (Tokio)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS	
Canadá (centro)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Europa (Fráncfort)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS	
Europa (Irlanda)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS	
Europa (Londres)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS	
Europa (París)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	
Europa (Estocolmo)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Medio Oriente (Baréin)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	
Medio Oriente (EAU)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
América del Sur (São Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra-streams.us-gov-east-1.api.aws	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
AWS GovCloud (US-West)	us-gov-west-1	cassandra-streams.us-gov-west-1.api.aws	HTTPS	

Conexión a puntos finales de doble pila

Los puntos de enlace globales de Amazon Keyspaces son puntos de enlace de doble pila que aceptan solicitudes de IPv4 e IPv6.

Al conectarse a Amazon Keyspaces mediante IPv6, el servicio adapta automáticamente las respuestas de la tabla del sistema para que coincidan con su protocolo de conexión. Esto garantiza que sus aplicaciones reciban información coherente sobre las direcciones de red que coincidan con su tipo de conexión. Esto proporciona al cliente información precisa sobre la topología de la red y, al mismo tiempo, mantiene la compatibilidad con versiones anteriores de las aplicaciones CQL existentes.

Amazon Keyspaces detecta automáticamente el protocolo de red (IPv4 o IPv6) que utiliza la conexión de su cliente y ajusta las respuestas de la tabla del sistema en consecuencia. Esta detección se produce de forma transparente durante el apretón de manos de la conexión inicial y no requiere ninguna configuración adicional por parte de la aplicación cliente.

Amazon Keyspaces devuelve las direcciones IP en función del protocolo de conexión. Por ejemplo, una solicitud de una red IPv4 devuelve la siguiente respuesta.

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

Una conexión desde una red IPv6 a un punto final de doble pila, por ejemplo `cassandra.us-east-1.api.aws`, devuelve la siguiente respuesta.

```
SELECT * FROM system.peers;
```

```
-- Returns IPv6 addresses in peer column
-- Example: 2001:db8::1, 2001:db8::2, etc.
```

Para obtener más información sobre la compatibilidad con IPv6 en Amazon Keyspaces, consulte [the section called “Compatibilidad con IPv6 en Amazon Keyspaces”](#)

Compatibilidad con IPv6 en Amazon Keyspaces

La compatibilidad con IPv6 en Amazon Keyspaces permite a las aplicaciones establecer conexiones mediante el Protocolo de Internet versión 6, el protocolo de Internet de próxima generación que proporciona un espacio de direcciones enormemente ampliado en comparación con IPv4. La implementación utiliza puntos finales de doble pila que admiten IPv4 e IPv6 simultáneamente, lo que garantiza la compatibilidad con versiones anteriores y, al mismo tiempo, permite una conectividad preparada para el futuro. Para obtener una lista de puntos de enlace, consulte [the section called “Puntos de enlace globales”](#).

Amazon Keyspaces implementa la compatibilidad con IPv6 a través de una arquitectura de doble pila que mantiene una total compatibilidad con versiones anteriores y, al mismo tiempo, permite la conectividad IPv6.

Resolución de DNS en Amazon Keyspaces

Cuando las aplicaciones se conectan a puntos finales de doble pila, el proceso de resolución de DNS devuelve ambos tipos de direcciones:

Registros A (IPv4)

Direcciones IPv4 tradicionales para compatibilidad con versiones anteriores

Registros AAAA (IPv6)

Nuevas direcciones IPv6 para una conectividad moderna

El sistema operativo y la pila de redes del cliente seleccionan automáticamente el protocolo más adecuado en función de la configuración local, la disponibilidad de la red y las preferencias del sistema.

El protocolo Cassandra Query Language (CQL) admite sin problemas la conectividad IPv6 sin necesidad de cambiar el código de la aplicación.

Selección automática de protocolos

- Las aplicaciones especifican el punto final de doble pila
- Network Stack elige IPv4 o IPv6 en función de la disponibilidad
- No se requieren modificaciones de código para las aplicaciones CQL existentes

Compatibilidad de controladores

- Todos los principales controladores CQL admiten IPv6 de forma transparente
- DataStax los controladores gestionan las direcciones IPv6 de forma nativa
- Open-source los controladores funcionan sin modificaciones

Consistencia de conexión

- Las tablas del sistema reflejan el protocolo de conexión utilizado
- Las conexiones IPv6 muestran las direcciones IPv6 en `system.peers`
- Las conexiones IPv4 siguen mostrando direcciones IPv4

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.

Note

`Murmur3Partitioner` es el particionador recomendado para Amazon Keyspaces y la `cqlsh-expansion`. La `cqlsh-expansion` no es compatible con el `DefaultPartitioner` de Amazon Keyspaces. Para obtener más información, consulte [the section called “Uso de particionadores”](#).

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`; si quiere averiguar cuál es su versión de Python, utilice `Python --version`. Para la instalación, puede ejecutar el siguiente comando.

```
python3 -m pip 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 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, boto3, 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 la PATH, debe agregarlo siguiendo las instrucciones de su sistema operativo. A continuación se muestra un ejemplo para Ubuntu Linux.

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

Para confirmar que el paquete esté 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 archivo de certificado combinado en el `.cassandra` directorio. 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 obtener más información sobre los certificados, consulte [the section called “Configuración manual de conexiones de cqlsh para TLS”](#).

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, utilizamos `us-east-1`.

```
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 “Cree credenciales de IAM para la autenticación 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 hasta 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](#).

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 encuentra algún error, consulte [the section called “Errores de conexión con cqlsh”](#) para ver información de 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 “Creación de 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 usar la `cqlsh-expansion` utilidad que descarga automáticamente los certificados e instala un archivo de `cqlshrc` configuración preconfigurado. Para obtener más información, consulte [the section called “Uso de cqlsh-expansion”](#) en esta página.

Si desea descargar los certificados y configurar la conexión manualmente, puede hacerlo siguiendo estos pasos.

1. Descargue los siguientes certificados digitales y guarde los archivos localmente o en su directorio principal.
 - a. AmazonRootCA1
 - b. AmazonRootCA2
 - c. AmazonRootCA3
 - d. AmazonRootCA4
 - e. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

 Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todo Regiones de AWS a certificados emitidos bajo Amazon Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

2. Combine todos los certificados descargados en un solo pem archivo con el nombre de nuestros *keyspaces-bundle.pem* ejemplos. Puede hacerlo ejecutando el siguiente comando : Tome nota de la ruta al archivo, la necesitará más adelante.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
    AmazonRootCA4.pem \
    sf-class2-root.crt \
    > keyspaces-bundle.pem
```

3. 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/keyspaces-bundle.pem
```

Uso de AWS CLI para conectarse a Amazon Keyspaces

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

Antes de poder utilizarlos AWS CLI con Amazon Keyspaces, debe obtener un identificador de clave de acceso y una clave de acceso secreta. Para obtener más información, consulte [the section called “Cree credenciales de IAM para la autenticación AWS”](#).

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

Temas

- [Descargar y configurar el AWS CLI](#)
- [Uso de AWS CLI con Amazon Keyspaces](#)

Descargar y configurar el AWS CLI

AWS CLI Está disponible en <https://aws.amazon.com/cli>. Se ejecuta en Windows, macOS o Linux. Tras descargarlo AWS CLI, siga estos pasos para instalarlo y configurarlo:

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 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. AWS CLI Admite una sintaxis abreviada para los valores de los parámetros, así como JSON. Los siguientes ejemplos de Amazon Keyspaces utilizan una sintaxis AWS CLI abreviada. Para obtener más información, consulte [Uso de la sintaxis abreviada con la CLI 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-1:111122223333:/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-1:111122223333:/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-1:111122223333:/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-1:111122223333:/keyspace/catalog/table/
book_awards",
  "creationTimestamp": 1645564368.628,
  "status": "ACTIVE",
  "schemaDefinition": {
    "allColumns": [
      {
        "name": "year",
        "type": "int"
      },
      {
        "name": "award",
        "type": "text"
      },
      {
        "name": "category",
        "type": "text"
      },
      {
        "name": "rank",
        "type": "int"
      },
      {
        "name": "author",
        "type": "text"
      },
      {
        "name": "book_title",
        "type": "text"
      },
      {
        "name": "publisher",
        "type": "text"
      }
    ]
  }
}
```

```
    ],
    "partitionKeys": [
      {
        "name": "year"
      },
      {
        "name": "award"
      }
    ],
    "clusteringKeys": [
      {
        "name": "category",
        "orderBy": "ASC"
      },
      {
        "name": "rank",
        "orderBy": "ASC"
      }
    ],
    "staticColumns": []
  },
  "capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": 1645564368.628
  },
  "encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
  },
  "pointInTimeRecovery": {
    "status": "DISABLED"
  },
  "ttl": {
    "status": "ENABLED"
  },
  "defaultTimeToLive": 0,
  "comment": {
    "message": ""
  }
}
```

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
- habilite Point-in-time la recuperación de 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 de la AWS KMS clave gestionada por el cliente 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,kmsKeyId=arn:aws:kms:us-
east-1:111122223333:key/11111111-2222-3333-4444-555555555555'
```

```
--ttl 'status=ENABLED'
```

Uso de la API para conectarse a Amazon Keyspaces

Puede usar el AWS SDK y 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 servicios y herramientas de infraestructura como código (iAC), como AWS CloudFormation Terraform.

Antes de poder utilizarlos AWS CLI con Amazon Keyspaces, debe obtener un identificador de clave de acceso y una clave de acceso secreta. Para obtener más información, consulte [the section called “Cree credenciales de IAM para la autenticación 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 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. Estos son los controladores y las versiones más recientes que hemos probado y que recomendamos usar con Amazon Keyspaces:

- Java v3.3
- Java v4.17
- Python Cassandra-driver 3.29.1
- Node.js cassandra driver -v 4.7.2
- GO using GOCQL v1.6
- .NET CassandraCSharpDriver -v 3.20.1

Para obtener más información sobre los controladores de Cassandra, consulte [Controladores de cliente de Apache Cassandra](#).

 Note

Como ayuda para empezar, puede ver y descargar ejemplos de código integral que establecen conexiones a 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 tutorial paso a paso que muestra cómo conectarse a Amazon Keyspaces desde un punto de conexión de VPC de Amazon, consulte [the section called “Conexión con puntos de conexión 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 Node.js cliente Cassandra para acceder a Amazon Keyspaces mediante programación](#)
- [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:

- Crear 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 “Cree credenciales de IAM para la autenticación 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](#)
- [Step-by-step tutorial para conectarse a Amazon Keyspaces mediante el controlador DataStax Java para Apache Cassandra con credenciales específicas del servicio](#)
- [Step-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 los siguientes certificados digitales y guarde los archivos de forma local o en su directorio principal.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todo Regiones de AWS a certificados emitidos bajo Amazon Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

- b. Convierta los certificados digitales en archivos TrustStore y agréguelos al almacén de claves.

```
openssl x509 -outform der -in AmazonRootCA1.pem -out temp_file.der
```

```
keytool -import -alias amazon-root-ca-1 -keystore cassandra_truststore.jks -
file temp_file.der

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

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

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

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

En el último paso, debe crear una contraseña para el almacén de claves y confiar en cada certificado. El comando interactivo tiene el siguiente aspecto.

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

Extensions:

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

```
#2: ObjectId: 2.5.29.15 Criticality=true
```

```
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]
```

```
#3: ObjectId: 2.5.29.14 Criticality=false
```

```
SubjectKeyIdentifier [
KeyIdentifier [
0000: 84 18 CC 85 34 EC BC 0C   94 94 2E 08 59 9C C7 B2   ....4.....Y...
0010: 10 4E 0A 08                               .N..
]
]
```

```
Trust this certificate? [no]: yes
```

```
Certificate was added to keystore
```

```
Enter keystore password:
```

```
Owner: CN=Amazon Root CA 2, O=Amazon, C=US
```

```
Issuer: CN=Amazon Root CA 2, O=Amazon, C=US
```

```
Serial number: 66c9fd29635869f0a0fe58678f85b26bb8a37
```

```
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
```

```
Certificate fingerprints:
```

```
SHA1: 5A:8C:EF:45:D7:A6:98:59:76:7A:8C:8B:44:96:B5:78:CF:47:4B:1A
```

```
SHA256:
```

```
1B:A5:B2:AA:8C:65:40:1A:82:96:01:18:F8:0B:EC:4F:62:30:4D:83:CE:C4:71:3A:19:C3:9C:01:1E
```

```
Signature algorithm name: SHA384withRSA
```

```
Subject Public Key Algorithm: 4096-bit RSA key
```

```
Version: 3
```

```
Extensions:
```

```
#1: ObjectId: 2.5.29.19 Criticality=true
```

```
BasicConstraints:[
  CA:true
  PathLen:2147483647
]
```

```
#2: ObjectId: 2.5.29.15 Criticality=true
```

```
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]
```

```
]
```

```
#3: ObjectId: 2.5.29.14 Criticality=false
```

```
SubjectKeyIdentifier [
```

```
KeyIdentifier [
```

```
0000: B0 0C F0 4C 30 F4 05 58    02 48 FD 33 E5 52 AF 4B    ...L0..X.H.3.R.K
```

```
0010: 84 E3 66 52                                ..fR
```

```
]
```

```
]
```

```
Trust this certificate? [no]: yes
```

```
Certificate was added to keystore
```

```
Enter keystore password:
```

```
Owner: CN=Amazon Root CA 3, O=Amazon, C=US
```

```
Issuer: CN=Amazon Root CA 3, O=Amazon, C=US
```

```
Serial number: 66c9fd5749736663f3b0b9ad9e89e7603f24a
```

```
Valid from: Tue May 26 00:00:00 UTC 2015 until: Sat May 26 00:00:00 UTC 2040
```

```
Certificate fingerprints:
```

```
SHA1: 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E
```

```
SHA256:
```

```
18:CE:6C:FE:7B:F1:4E:60:B2:E3:47:B8:DF:E8:68:CB:31:D0:2E:BB:3A:DA:27:15:69:F5:03:43:B4
```

```
Signature algorithm name: SHA256withECDSA
```

```
Subject Public Key Algorithm: 256-bit EC (secp256r1) key
```

```
Version: 3
```

```
Extensions:
```

```
#1: ObjectId: 2.5.29.19 Criticality=true
```

```
BasicConstraints:[
```

```
CA:true
```

```
PathLen:2147483647
```

```
]
```

```
#2: ObjectId: 2.5.29.15 Criticality=true
```

```
KeyUsage [
```

```
DigitalSignature
```

```
Key_CertSign
```

```
Crl_Sign
```

```
]
```

```
#3: ObjectId: 2.5.29.14 Criticality=false
```

```
SubjectKeyIdentifier [
```

```
KeyIdentifier [
```

```
0000: AB B6 DB D7 06 9E 37 AC    30 86 07 91 70 C7 9C C4    .....7.0...p...
```



```

0010: 19 B1 78 C0                                ...x.
]
]

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

Extensions:

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

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

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

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

```

```

Enter keystore password:
Owner: OU=Starfield Class 2 Certification Authority, O="Starfield Technologies,
  Inc.", C=US
Issuer: OU=Starfield Class 2 Certification Authority, O="Starfield
  Technologies, Inc.", C=US
Serial number: 0
Valid from: Tue Jun 29 17:39:16 UTC 2004 until: Thu Jun 29 17:39:16 UTC 2034
Certificate fingerprints:
    SHA1: AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A
    SHA256:
    14:65:FA:20:53:97:B8:76:FA:A6:F0:A9:95:8E:55:90:E4:0F:CC:7F:AA:4F:B7:C2:C8:67:75:21:FB
Signature algorithm name: SHA1withRSA (weak)
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

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

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

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

Warning:

```

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

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

2. 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
```

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

El siguiente tutorial le guía paso a paso en la conexión a Amazon Keyspaces mediante un controlador Java para Cassandra utilizando credenciales específicas del servicio. En concreto, utilizará la versión 4.0 del controlador 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 “Creación de 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 `PlainTextAuthProvider` clase. `ServiceUsername` `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 [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)

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— Inicialízalo `SSLEngineFactory` añadiendo una sección en el archivo de configuración con una sola línea que especifique la clase con la que. `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.

Step-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 “Cree credenciales de IAM para la autenticación 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 “Cree credenciales de IAM para la autenticación 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`. `SigV4AuthProvider` 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 ver todos los disponibles Regiones de AWS, consulte. [???](#) Configure `slow-replica-avoidance = false` para equilibrar la carga con respecto a todos los nodos disponibles.
3. Idempotencia: establezca el valor predeterminado de la aplicación `idempotence` para configurar el controlador para que siempre vuelva `true` a intentar las solicitudes o solicitudes fallidas. `read/write prepare/execute` Esta es una práctica recomendada para las aplicaciones distribuidas que ayuda a administrar los errores transitorios reintentando las solicitudes fallidas.
4. SSL/TLS— Inicialice el `SSLConnectionFactory` añadiendo una sección en el archivo de configuración con una sola línea que especifique la clase con la que. `class = DefaultSslConnectionFactory` 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`.
5. Conexiones: cree al menos 3 conexiones locales por punto de conexión configurándolo en `local.size = 3`. Esta es una práctica recomendada que ayuda a su aplicación a administrar las sobrecargas y las ráfagas de tráfico. Para obtener más información sobre cómo calcular el número de conexiones locales por punto de conexión que necesita su aplicación en función de los patrones de tráfico esperados, consulte [the section called “Configuración de conexiones”](#).
6. Política de reintentos: implementa la política de reintentos de Amazon Keyspaces `AmazonKeyspacesExponentialRetryPolicy` en lugar de la política `DefaultRetryPolicy` que viene con el controlador de Cassandra. Esto le permite configurar el número de reintentos para la política `AmazonKeyspacesExponentialRetryPolicy` que mejor se adapte a sus necesidades. De forma predeterminada, el número de reintentos para la `AmazonKeyspacesExponentialRetryPolicy` está establecido en 3. Para obtener más información, consulte [the section called “Configuración de las políticas de reintentos”](#).
7. Instrucciones preparadas: configure `prepare-on-all-nodes` como `false` para optimizar el uso de la red.

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



```
contact-points = [ "cassandra.us-east-2.amazonaws.com:9142"]
request {
    timeout = 2 seconds
    consistency = LOCAL_QUORUM
    page-size = 1024
    default-idempotence = true
}
load-balancing-policy {
    local-datacenter = "us-east-2"
    class = DefaultLoadBalancingPolicy
    slow-replica-avoidance = false
}
}
advanced {
    auth-provider {
        class = software.aws.mcs.auth.SigV4AuthProvider
        aws-region = us-east-2
    }
    ssl-engine-factory {
        class = DefaultSslEngineFactory
        truststore-path = "./src/main/resources/cassandra_truststore.jks"
        truststore-password = "my_password"
        hostname-validation = false
    }
    connection {
connect-timeout = 5 seconds
max-requests-per-connection = 512
pool {
    local.size = 3
}
}
    retry-policy {
        class = com.aws.ssa.keyspaces.retry.AmazonKeyspacesExponentialRetryPolicy
max-attempts = 3
min-wait = 10 mills
max-wait = 100 mills
    }
    prepared-statements {
prepare-on-all-nodes = false
    }
}
}
```

 Note

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 “Cree credenciales de IAM para la autenticación 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 “Cree credenciales de IAM para la autenticación 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 que se indican [the section called “Antes de empezar”](#) para descargar los certificados digitales, convertirlos en archivos TrustStore y adjuntar el almacén de claves de los argumentos de la JVM a 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:

- Crear 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 “Cree credenciales de IAM para la autenticación 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, debe descargar los certificados digitales de Amazon y configurar el controlador de Python para que utilice TLS.

Descargue los siguientes certificados digitales y guarde los archivos de forma local o en su directorio principal.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todas las Regiones de AWS a certificados emitidos bajo Amazon Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

Combine todos los certificados descargados en un solo pem archivo con el nombre de nuestros *keyspaces-bundle.pem* ejemplos. Puede hacerlo ejecutando el siguiente comando : Tome nota de la ruta al archivo, la necesitará más adelante.

```
cat AmazonRootCA1.pem \  
    AmazonRootCA2.pem \  
    AmazonRootCA3.pem \  
    AmazonRootCA4.pem \  
    sf-class2-root.crt \  
> keyspaces-bundle.pem
```

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/keyspaces-bundle.pem')  
  
ssl_context.verify_mode = CERT_REQUIRED  
auth_provider = PlainTextAuthProvider(username='ServiceUserName',  
    password='ServicePassword')  
cluster = Cluster(['cassandra.us-east-2.amazonaws.com'], ssl_context=ssl_context,  
    auth_provider=auth_provider, port=9142)  
session = cluster.connect()  
r = session.execute('select * from system_schema.keyspaces')  
print(r.current_rows)
```

Notas de uso:

1. "*path_to_file/keyspaces-bundle.pem*" Sustitúyala por la ruta al archivo de certificado combinado guardado en el primer paso.
2. 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. [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)
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 las credenciales del rol de IAM; para ello, siga los pasos que se indican en [the section called “Cree credenciales de IAM para la autenticación 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 “Creación 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/keyspaces-bundle.pem')
ssl_context.verify_mode = CERT_REQUIRED

# use this if you want to use Boto to set the session parameters.
boto_session = boto3.Session(aws_access_key_id="AKIAIOSFODNN7EXAMPLE",
                             aws_secret_access_key="wJalrXUtnFEMI/K7MDENG/
bpXrfiCYEXAMPLEKEY",
                             aws_session_token="AQoDYXdzEJr...<remainder of token>",
                             region_name="us-east-2")
auth_provider = SigV4AuthProvider(boto_session)
```



```
# Use this instead of the above line if you want to use the Default Credentials and not
# bother with a session.
# auth_provider = SigV4AuthProvider()

cluster = Cluster(['cassandra.us-east-2.amazonaws.com'], ssl_context=ssl_context,
                  auth_provider=auth_provider,
                  port=9142)
session = cluster.connect()
r = session.execute('select * from system_schema.keyspaces')
print(r.current_rows)
```

Notas de uso:

1. Sustituya "*path_to_file/keyspaces-bundle.pem*" por la ruta al certificado guardado en el primer paso.
2. Asegúrate de que *aws_access_key_id*, *aws_secret_access_key*, y el *aws_session_token* coincidan con Access Key, Secret Access Key, y Session Token que obtuviste usando `boto3.session`. Para obtener más información, consulte la sección sobre [credenciales](#) en la AWS SDK para 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 Node.js cliente Cassandra para acceder a Amazon Keyspaces mediante programación

En esta sección se muestra cómo conectarse a Amazon Keyspaces mediante un controlador de Node.js cliente. 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:

- Crear 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 "Cree credenciales de IAM para la autenticación AWS"](#).

Temas

- [Antes de empezar](#)
- [Conéctese a Amazon Keyspaces mediante el Node.js DataStax controlador de Apache Cassandra y las credenciales específicas del servicio](#)
- [Conéctese a Amazon Keyspaces mediante el DataStax Node.js controlador 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 los siguientes certificados digitales y guarde los archivos de forma local o en su directorio principal.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todo Regiones de AWS a certificados emitidos bajo Amazon

Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

Combine todos los certificados descargados en un solo pem archivo con el nombre de nuestros *keyspaces-bundle.pem* ejemplos. Puede hacerlo ejecutando el siguiente comando : Tome nota de la ruta al archivo, la necesitará más adelante.

```
cat AmazonRootCA1.pem \  
    AmazonRootCA2.pem \  
    AmazonRootCA3.pem \  
    AmazonRootCA4.pem \  
    sf-class2-root.crt \  
> keyspaces-bundle.pem
```

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

Configure su controlador para que utilice el archivo de certificado combinado *keyspaces-bundle.pem* 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/keyspaces-bundle.pem', 'utf-8')],  
        host: 'cassandra.us-west-2.amazonaws.com',  
        rejectUnauthorized: true  
    };  
const client = new cassandra.Client({  
    contactPoints: ['cassandra.us-west-2.amazonaws.com'],  
    localDataCenter: 'us-west-2',  
    authProvider: auth,  
    sslOptions: sslOptions1,  
    protocolOptions: { port: 9142 }  
});  
const query = 'SELECT * FROM system_schema.keyspaces';
```

```
client.execute(query)
    .then( result => console.log('Row from Keyspaces %s',
    result.rows[0]))
    .catch( e=> console.log(`${e}`));
```

Notas de uso:

1. "*path_to_file/keyspaces-bundle.pem*" Sustitúyalo por la ruta al archivo de certificado combinado guardado en el primer paso.
2. 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. [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)
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 DataStax Node.js controlador 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 DataStax Node.js controlador 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 “Cree credenciales de IAM para la autenticación AWS”](#).

[Agregue el complemento de autenticación Node.js SigV4 a su aplicación desde el repositorio. GitHub](#)

El complemento es compatible con la versión 4.x del DataStax Node.js controlador de Cassandra y depende del AWS SDK que tenga. 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 Region-specific instancia 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_file/keyspaces-bundle.pem', 'utf-8')],
  host: 'cassandra.us-west-2.amazonaws.com',
  rejectUnauthorized: true
};

const client = new cassandra.Client({
  contactPoints: ['cassandra.us-west-2.amazonaws.com'],
  localDataCenter: 'us-west-2',
  authProvider: auth,
  sslOptions: sslOptions1,
  protocolOptions: { port: 9142 }
});

const query = 'SELECT * FROM system_schema.keyspaces';

client.execute(query).then(
  result => console.log('Row from Keyspaces %s', result.rows[0]))
  .catch( e=> console.log(`${e}`));
```

Notas de uso:

1. Sustituya "*path_to_file/keyspaces-bundle.pem*" por la ruta al certificado guardado en el primer paso.
2. Asegúrese de que la *accessKeyId* clave de acceso y la clave de acceso secreta que obtuvo utilizando *secretAccessKey* coincidan con `AWSCredentialsProvider`. Para obtener más información, consulta [Cómo configurar las credenciales Node.js JavaScript en el AWS SDK Node.js](#).
3. Para almacenar las claves de acceso fuera del código, consulte 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, configure el controlador para que utilice el almacén de confianza del sistema, que incluye las CA 1-4 raíz de Amazon.

1. Instale el nuget directo CassandraCSharpDriver mediante la consola nuget.

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

2. En el siguiente ejemplo se utiliza un proyecto de consola .NET Core C# 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)
        {
            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:

1. Asegúrese de utilizar el almacén de confianza del sistema predeterminado, que incluye las CA 1-4 de Amazon Root.
2. 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. [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)
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 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 Cassandra. 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:

- Crear 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 directores 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 “Cree credenciales de IAM para la autenticación 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 Go para utilizar TLS.

Descargue los siguientes certificados digitales y guarde los archivos de forma local o en su directorio principal.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todo Regiones de AWS a certificados emitidos bajo Amazon Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

Combine todos los certificados descargados en un solo pem archivo con el nombre de nuestros *keyspaces-bundle.pem* ejemplos. Puede hacerlo ejecutando el siguiente comando : Tome nota de la ruta al archivo, la necesitará más adelante.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
    AmazonRootCA4.pem \
    sf-class2-root.crt \
    > keyspaces-bundle.pem
```

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 keyspaces-bundle.pem
    cluster.SslOpts = &gocql.SslOptions{
        CaPath: "path_to_file/keyspaces-bundle.pem",
        EnableHostVerification: false,
    }

    // Override default Consistency to LocalQuorum
    cluster.Consistency = gocql.LocalQuorum
    cluster.DisableInitialHostLookup = false

    session, err := cluster.CreateSession()
    if err != nil {
        fmt.Println("err>", err)
    }
    defer session.Close()
}
```

```
// run a sample query from the system keyspace
var text string
iter := session.Query("SELECT keyspace_name FROM system_schema.tables;").Iter()
for iter.Scan(&text) {
    fmt.Println("keyspace_name:", text)
}
if err := iter.Close(); err != nil {
    log.Fatal(err)
}
session.Close()
}
```

Notas de uso:

- a. "*path_to_file/keyspaces-bundle.pem*" Sustitúyala por la ruta al archivo de certificado combinado guardado en el primer paso.
- b. 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. [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)
- 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 la entidad principal de IAM; para ello, siga los pasos que se indican en [the section called "Cree credenciales de IAM para la autenticación AWS"](#).

Si una aplicación se ejecuta en una instancia de Lambda o Amazon EC2, la aplicación utilizará automáticamente las credenciales de la instancia. Para ejecutar este tutorial de forma local, puede almacenar las credenciales como variables de entorno locales.

[Añade 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

    // the authenticator uses the default credential chain to find AWS credentials
    cluster.Authenticator = sigv4.NewAwsAuthenticator()

    cluster.SslOpts = &gocql.SslOptions{
        CaPath: "path_to_file/keyspaces-bundle.pem",
        EnableHostVerification: false,
    }
    cluster.Consistency = gocql.LocalQuorum
    cluster.DisableInitialHostLookup = false

    session, err := cluster.CreateSession()
    if err != nil {
```

```
    fmt.Println("err>", err)
    return
}
defer session.Close()

// doing the query
var text string
iter := session.Query("SELECT keyspace_name FROM system_schema.tables;").Iter()
for iter.Scan(&text) {
    fmt.Println("keyspace_name:", text)
}
if err := iter.Close(); err != nil {
    log.Fatal(err)
}
}
```

Notas de uso:

1. "*path_to_file/keyspaces-bundle.pem*" Sustitúyalo por la ruta al archivo de certificado guardado en el primer paso.
2. Para que este ejemplo se ejecute localmente, debe definir las siguientes variables como variables de entorno:
 - AWS_ACCESS_KEY_ID
 - AWS_SECRET_ACCESS_KEY
 - AWS_DEFAULT_REGION
3. Para almacenar las claves de acceso fuera del código, consulte 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

Los certificados de Amazon Keyspaces están pasando a la jerarquía de Amazon Trust Services (ATS). Asegúrese de que su entorno confíe en las CA raíz de Amazon 1—4 para evitar errores de conexión durante esta rotación. 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 *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* [Creación de credenciales específicas del servicio para el acceso programático a Amazon Keyspaces](#)

 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
```

Configuración del acceso entre cuentas a Amazon Keyspaces con puntos de conexión de VPC

Puede crear y usar recursos separados Cuentas de AWS para aislar y usarlos en diferentes 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 a Amazon Keyspaces mediante puntos de conexión de VPC en una VPC compartida](#)
- [Configuración del acceso entre cuentas a Amazon Keyspaces sin una VPC compartida](#)

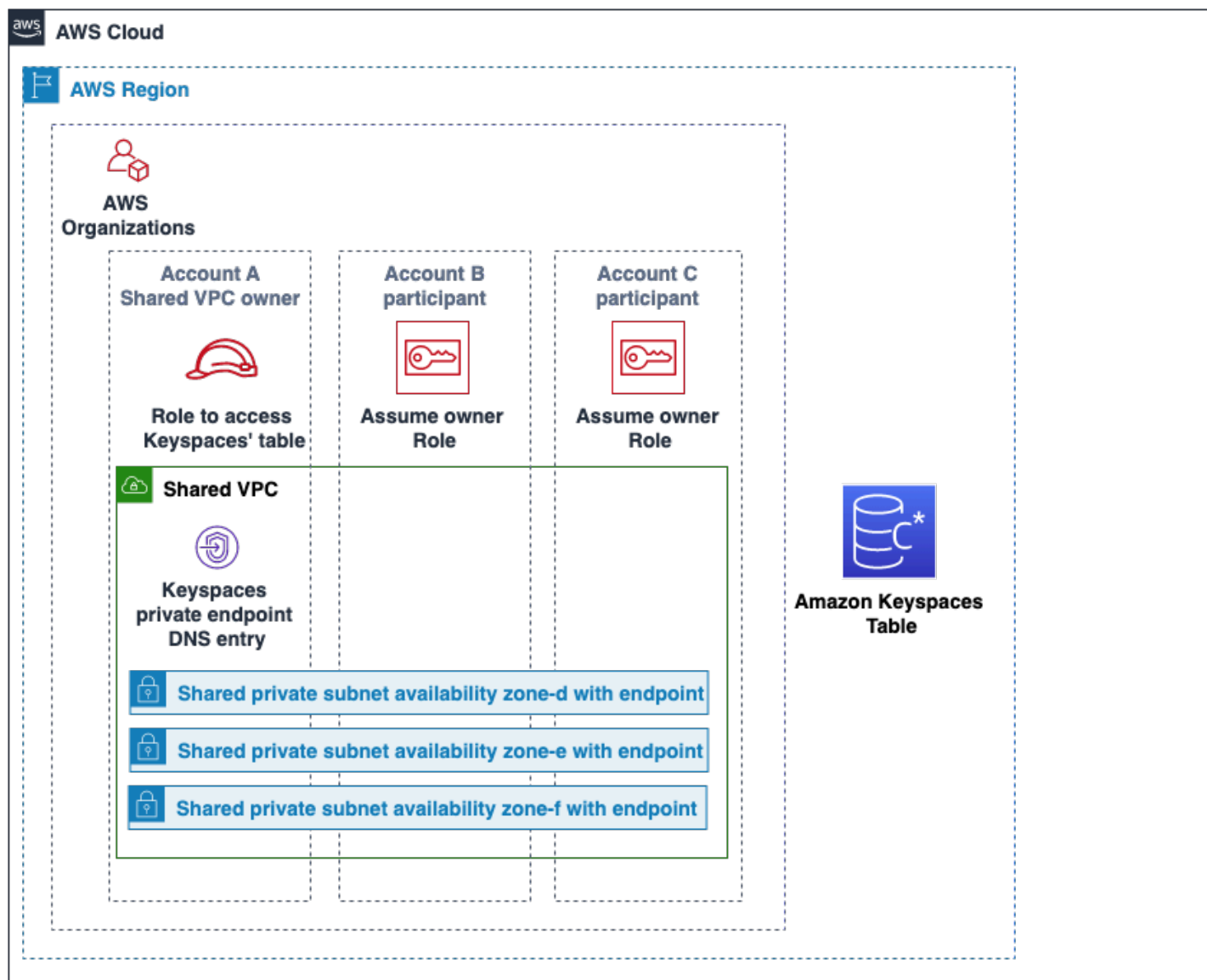
Configuración del acceso entre cuentas a Amazon Keyspaces mediante puntos de conexión de VPC en una VPC compartida

Puede crear diferentes recursos Cuentas de AWS para separarlos 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:111111111111: esta cuenta contiene la infraestructura, incluyendo los puntos de conexión de VPC, las subredes de VPC y las tablas de Amazon Keyspaces.
- Account B:222222222222: esta cuenta contiene una aplicación en un entorno de desarrollo que necesita conectarse a la tabla de Amazon Keyspaces en Account A:111111111111.
- Account C:333333333333: esta cuenta contiene una aplicación en un entorno de producción que necesita conectarse a la tabla de Amazon Keyspaces en Account A:111111111111.



Account A:111111111111 es la cuenta que contiene los recursos (una tabla de Amazon Keyspaces) a los que Account C:333333333333 hay que Account B:222222222222 acceder, al igual que Account A:111111111111 la cuenta de confianza. Account B:222222222222 y Account C:333333333333 son las cuentas con los principales las que necesitan acceso a los recursos (una tabla de Amazon Keyspaces) Account A:111111111111, Account B:222222222222 también lo son Account C:333333333333 las cuentas de confianza. 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:111111111111.

Configuración de **Account A:111111111111**

1. Se utiliza AWS Resource Access Manager para crear un recurso compartido para la subred y compartir la subred privada con y. Account B:222222222222 Account C:333333333333

Account B:222222222222 y Account C:333333333333 ya pueden ver y crear recursos en la subred que se ha compartido con ellos.
2. Cree un punto final de VPC privado de Amazon Keyspaces con la tecnología 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 Account A:111111111111 que tenga acceso total a la tabla Amazon Keyspaces, lea las tablas del sistema Amazon Keyspaces y pueda describir los recursos de la 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 una política de confianza para el rol de IAM para que Account A:111111111111 Account C:333333333333 pueda asumir el rol Account B:222222222222 como cuentas de confianza. Esto se muestra en el siguiente ejemplo.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```

    "Principal": {
      "AWS": [
        "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
        "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
      ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {}
  }
]
}

```

Para obtener más información sobre las políticas de IAM multicuenta, consulte las [Cross-account políticas](#) en la Guía del usuario de IAM.

Configuración de **Account B:222222222222** y **Account C:333333333333**

1. En Account B:222222222222 y Account C:333333333333, cree nuevos roles y vincule la siguiente política que permite a la entidad principal asumir el rol compartido creado en Account A:111111111111.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

Permitir que el director asuma la función compartida se implementa mediante la AssumeRole API de (). AWS Security Token Service AWS STS Para obtener más información, consulte [Proporcionar acceso a un usuario de IAM a otro de su Cuenta de AWS propiedad](#) en la Guía del usuario de IAM.

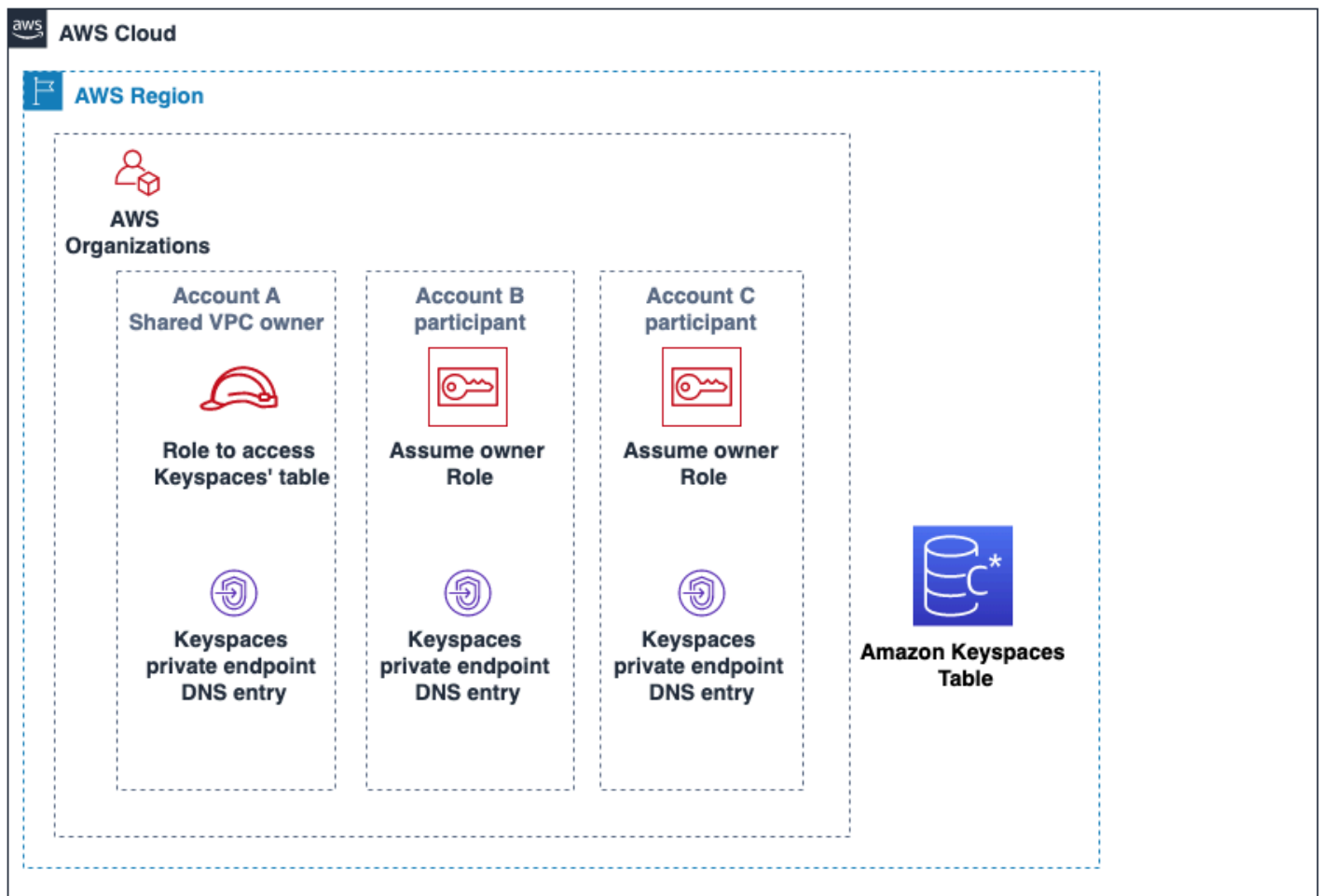
2. En Account B:222222222222 y Account C:333333333333, puede crear aplicaciones que utilicen el complemento de autenticación SIGV4, que permite que una aplicación asuma

el rol compartido para conectarse a la tabla de Amazon Keyspaces ubicada en Account A:111111111111 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 de acceso programático”](#). Para obtener más información sobre cómo configurar una aplicación para que asuma un rol en otra AWS cuenta, consulte [Autenticación y acceso](#) en la guía de referencia de herramientas y AWS SDK.

Configuración del acceso entre cuentas a 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:111111111111, Account B:222222222222 y Account C:333333333333 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 en diferentes cuentas sin un punto de enlace de VPC compartido mediante los puntos de enlace públicos o mediante la implementación de un punto de enlace 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:111111111111, Account B:222222222222 y Account C:333333333333 requieren sus propios puntos de conexión de VPC para acceder a la tabla de Account A:111111111111. 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.



Account A:111111111111 es la cuenta que contiene los recursos (una tabla de Amazon Keyspaces) a los que Account C:333333333333 hay que Account B:222222222222 acceder, al igual que Account A:111111111111 la cuenta de confianza. Account B:222222222222y Account C:333333333333 son las cuentas con los principales las que necesitan acceso a los recursos (una tabla de Amazon Keyspaces)Account A:111111111111, Account B:222222222222 también lo son Account C:333333333333 las cuentas de confianza. 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:111111111111.

Configuración de **Account A:111111111111**

1. Crea un espacio de claves y una tabla en Amazon Keyspaces. Account A:111111111111
2. Cree un rol de IAM Account A:111111111111 que tenga acceso total a la tabla Amazon Keyspaces y acceso de lectura a las tablas del sistema Amazon Keyspaces.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select",
        "cassandra:Modify"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111111111111:/keyspace/mykeyspace/table/mytable",
        "arn:aws:cassandra:us-east-1:111111111111:/keyspace/system*"
      ]
    }
  ]
}
```

3. Configure una política de confianza para la función de IAM de Account A:111111111111 forma que los directores puedan entrar en cuentas de confianza Account B:222222222222 y Account C:333333333333 puedan asumirla como cuentas de confianza. Esto se muestra en el siguiente ejemplo.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
          "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Para obtener más información sobre las políticas de IAM multicuenta, consulte las [Cross-account políticas](#) en la Guía del usuario de IAM.

- Configure el punto de enlace de la VPC Account A:111111111111 y adjunte permisos al punto de enlace que permitan que las funciones se envíen Account B:222222222222 y Account C:333333333333 asuman la función de Account A uso del punto de enlace de la VPC. Estos permisos son válidos para el punto final de la VPC al que están conectados. Para obtener más información sobre las políticas de punto de conexión de VPC, consulte [the section called "Control de acceso a puntos de conexión de VPC de interfaz para Amazon Keyspaces"](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessfromSpecificIAMroles",
      "Effect": "Allow",
      "Action": "cassandra:*",
      "Resource": "*",
      "Principal": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/Cross-Account-Role-B",
            "arn:aws:iam::333333333333:role/Cross-Account-Role-C"
          ]
        }
      }
    }
  ]
}
```

Configuración de **Account B:222222222222** y **Account C:333333333333**

- En Account B:222222222222 y Account C:333333333333, cree nuevos roles y vincule la siguiente política que permite a la entidad principal asumir el rol compartido creado en Account A:111111111111.

```
{
  "Version": "2012-10-17",
  "Statement": {
```

```
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111111111111:role/keyspaces_access"
    }
}
```

Permitir que el director asuma la función compartida se implementa mediante la AssumeRole API de AWS Security Token Service (AWS STS). Para obtener más información, consulte [Proporcionar acceso a un usuario de IAM a otro de su Cuenta de AWS propiedad](#) en la Guía del usuario de IAM.

2. En Account B:222222222222 y Account C:333333333333, puede crear aplicaciones que utilicen el complemento de autenticación SIGV4, que permite que una aplicación asuma la función compartida para conectarse a la tabla Amazon Keyspaces ubicada en Account A:111111111111. Para obtener más información sobre el complemento de autenticación SIGV4, consulte [the section called “Creación de credenciales de acceso programático”](#). Para obtener más información sobre cómo configurar una aplicación para que asuma un rol en otra AWS cuenta, consulte [Autenticación y acceso](#) en la Guía de referencia de herramientas y AWS SDK.

Introducción a Amazon Keyspaces (para Apache Cassandra)

Si es la primera vez que utiliza Apache Cassandra y Amazon Keyspaces, este tutorial le indicará cómo instalar los programas y herramientas necesarios para utilizar Amazon Keyspaces correctamente. Aprenderás a crear un espacio de claves y una tabla con Cassandra Query Language (CQL), the o Consola de administración de AWS the AWS Command Line Interface ().AWS CLI A continuación, utilizará Cassandra Query Language (CQL) para realizar operaciones de creación, lectura, actualización y eliminación (CRUD) de datos en la tabla de Amazon Keyspaces.

Este tutorial abarca los siguientes pasos.

- Requisitos previos: antes de comenzar el tutorial, siga las instrucciones de AWS configuración para registrarse AWS y crear un usuario de IAM con acceso a Amazon Keyspaces. A continuación, configura y. `cqsh-expansion` AWS CloudShell Como alternativa, puede utilizarlos AWS CLI para crear recursos en Amazon Keyspaces.
- Paso 1: creación de un espacio de claves y una tabla; en esta sección, creará un espacio de claves denominado “catalog” y una tabla denominada “book_awards” dentro de él. Especificará las columnas, los tipos de datos, la clave de partición y la columna de agrupamiento de la tabla mediante el Consola de administración de AWS CQL o el. AWS CLI
- Paso 2: Realizar operaciones CRUD: aquí, utilizará la `cqlsh-expansion` entrada CloudShell para insertar, leer, actualizar y eliminar datos de la tabla «book_awards». Aprenderá a usar varias instrucciones CQL, como SELECT, INSERT, UPDATE y DELETE, y practicará el filtrado y la modificación de datos.
- Paso 3: limpieza de los recursos; para evitar incurrir en cargos por los recursos no utilizados, en esta sección se explica cómo eliminar la tabla “book_awards” y el espacio de claves del “catalog” mediante la consola, CQL o la AWS CLI.

Para ver tutoriales para conectarse mediante programación a Amazon Keyspaces utilizando diferentes controladores de cliente de Apache Cassandra, consulte [the section called “Utilización de un controlador de cliente de Cassandra”](#). Para ver ejemplos de código que utilizan distintos AWS SDK, consulta [Ejemplos de código para Amazon Keyspaces AWS](#) que utilizan SDK.

Temas

- [Requisitos y consideraciones previos del tutorial](#)

- [Creación de un espacio de claves en Amazon Keyspaces.](#)
- [Comprobación del estado de creación de los espacios de claves en Amazon Keyspaces](#)
- [Creación de un espacio de claves en Amazon Keyspaces](#)
- [Comprobación del estado de creación de las tablas en Amazon Keyspaces](#)
- [Creación, lectura, actualización y eliminación de datos \(CRUD\) mediante CQL en Amazon Keyspaces](#)
- [Eliminación de una tabla en Amazon Keyspaces](#)
- [Eliminación de un espacio de claves en Amazon Keyspaces](#)

Requisitos y consideraciones previos del tutorial

Antes de empezar a utilizar Amazon Keyspaces, sigue las instrucciones de AWS configuración que se indican en. [Acceso 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.

Para completar todos los pasos del tutorial, necesitará instalar `cqlsh`. Puede seguir las instrucciones de configuración de [Uso de `cqlsh` para conectarse a Amazon Keyspaces](#).

Para acceder a Amazon Keyspaces mediante `cqlsh` o el AWS CLI, le recomendamos que utilice. AWS CloudShell CloudShelles un shell preautenticado y basado en un navegador que se puede iniciar directamente desde. Consola de administración de AWS Puede ejecutar comandos AWS Command Line Interface (AWS CLI) en Amazon Keyspaces utilizando el shell que prefiera (Bash PowerShell o Z shell). Para usar `cqlsh`, debe instalar la `cqlsh-expansion`. Para ver las instrucciones de instalación de `cqlsh-expansion`, consulte [the section called “Uso de `cqlsh-expansion`”](#). Para obtener más información al respecto, consulte CloudShell . [the section called “Usando AWS CloudShell”](#)

Para usar el AWS CLI para crear, ver y eliminar recursos en Amazon Keyspaces, siga las instrucciones de configuración que se encuentran en. [the section called “Descargar y configurar el AWS CLI”](#)

Tras completar los pasos de requisitos previos, continúe con [Creación de un espacio de claves en Amazon Keyspaces](#).

Creación de un espacio de claves en Amazon Keyspaces.

En esta sección, creará un espacio de claves utilizando la consola, `cqlsh` o la AWS CLI.

Note

Antes de comenzar, asegúrese de haber configurado todos los [requisitos previos del tutorial](#).

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:

- Instrucciones en el lenguaje de definición de datos (DDL) en la referencia del lenguaje CQL: [Keyspaces](#)
- [Cuotas para Amazon Keyspaces \(para Apache Cassandra\)](#)

En este tutorial, crearemos un espacio de claves de una sola región y la estrategia de replicación del espacio de claves es `SingleRegionStrategy`. Con `SingleRegionStrategy`, Amazon Keyspaces replica los datos en tres [zonas de disponibilidad](#) en una Región de AWS. Para aprender a crear claves de varias regiones, consulte [the section called “Creación de un espacio de claves multirregional”](#).

Uso de la consola

Para crear un espacio de claves con la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 **catalog** como nombre para su espacio de claves.

Restricciones del nombre:

- El nombre no puede estar vacío.

- Caracteres permitidos: alfanuméricos y guion bajo (_).
 - La longitud máxima es de 48 caracteres.
5. En Regiones de AWS, confirme que la Single-Region replicación es la estrategia de replicación del espacio de claves.
 6. Para crear el espacio de claves, elija Crear espacio de claves.
 7. Para verificar que el espacio de claves `catalog` se haya creado, haga lo siguiente:
 - a. En el panel de navegación, elija Espacios de claves.
 - b. Localice su espacio de claves `catalog` 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. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar `us-east-1` con su propia región.

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

La salida de ese comando debería tener este aspecto.

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

2. Cree su espacio de claves utilizando el siguiente comando CQL.

```
CREATE KEYSPACE catalog 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.

3. Compruebe que se haya creado su espacio de claves.

```
SELECT * from system_schema.keyspaces;
```

La salida de este comando debería ser similar a la siguiente.

```
cqlsh> SELECT * from system_schema.keyspaces;
```

keyspace_name	durable_writes	replication
system_schema	True	{'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
system_schema_mcs	True	{'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
system	True	{'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
system_multiregion_info	True	{'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
catalog	True	{'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}

(5 rows)

Usando el AWS CLI

El siguiente procedimiento crea un espacio de claves utilizando la AWS CLI.

Para crear un espacio de claves mediante el AWS CLI

1. Para confirmar que su entorno está configurado, puede ejecutar el siguiente comando en CloudShell.

```
aws keyspaces help
```

2. Cree su espacio de claves mediante la siguiente AWS CLI instrucción.

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

3. Compruebe que el espacio de claves se haya creado con la siguiente declaración AWS CLI

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

La salida de este comando debería tener un aspecto similar al de este ejemplo.

```
{
  "keyspaceName": "catalog",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/",
  "replicationStrategy": "SINGLE_REGION"
}
```

Comprobación del estado de creación de los 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 el Consola de administración de AWS, 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 claves se hace visible en la tabla `system_schema_mcs keyspaces` 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';
```

Si un espacio de claves se ha creado correctamente, la salida de la consulta tiene el siguiente aspecto.

```
keyspace_name | durable_writes | replication
-----+-----+-----
mykeyspace | true | {...} 1 item
```

Creación de un espacio de claves en Amazon Keyspaces

En este paso, creará una tabla utilizando la consola, `cqlsh` o la AWS CLI.

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 la 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:

- Diseño de claves de partición: [the section called “Diseño de claves de partición”](#)
- Uso de tablas: [the section called “Comprobación del estado de creación de las tablas”](#)
- Instrucciones DDL en la referencia del lenguaje CQL: [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, las claves de partición y las columnas de agrupación especificados.

Uso de la consola

El siguiente procedimiento crea la tabla `book_awards` con estas columnas y tipos de datos.

<code>year</code>	<code>int</code>
<code>award</code>	<code>text</code>
<code>rank</code>	<code>int</code>
<code>category</code>	<code>text</code>
<code>book_title</code>	<code>text</code>
<code>author</code>	<code>text</code>
<code>publisher</code>	<code>text</code>

Para crear una tabla con la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Espacios de claves.
3. Elija `catalog` como el espacio de claves en el que desea crear esta tabla.
4. Seleccione `Create table` (Creación de tabla).
5. En el cuadro `Nombre de la tabla`, introduzca **`book_awards`** como nombre para la tabla.

Restricciones del nombre:

- El 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.

<code>year</code>	<code>int</code>
<code>award</code>	<code>text</code>
<code>rank</code>	<code>int</code>
<code>category</code>	<code>text</code>
<code>book_title</code>	<code>text</code>

author	text
publisher	text

- a. Nombre: introduzca un nombre para la columna.

Restricciones del nombre:

- El 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. Para añadir una columna, elija Agregar columna.

7. Elija `award` y `year` como claves 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 `category` y `rank` como Columnas de agrupación en clústeres. Las columnas de agrupación son opcionales y determinan el orden de clasificación dentro de cada partición.

- a. Para añadir una columna de agrupación, elija Agregar columnas de agrupación en clústeres.

- b. En la lista Columnas, elija la categoría. En la lista Orden, elija ASC para clasificar en orden ascendente los valores de esta columna. (Elija DESC para orden descendente).

- c. A continuación, seleccione Agregar columnas de agrupación en clústeres y elija el rango.

9. En la sección Configuración de la tabla, elija Configuración predeterminada.

10. Seleccione Create table (Creación de 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.

Uso de CQL

Este procedimiento crea una tabla con las siguientes columnas y tipos de datos utilizando CQL. Las columnas `year` y `award` son claves de partición con `category` y `rank` como columnas de agrupación en clústeres, juntas forman la clave principal de la tabla.

<code>year</code>	<code>int</code>
<code>award</code>	<code>text</code>
<code>rank</code>	<code>int</code>
<code>category</code>	<code>text</code>
<code>book_title</code>	<code>text</code>
<code>author</code>	<code>text</code>
<code>publisher</code>	<code>text</code>

Para crear una tabla utilizando CQL

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar `us-east-1` con su propia región.

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

La salida de ese comando debería tener este aspecto.

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

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

```
);
```

Note

ASC es el orden de agrupación predeterminado. También puede especificar DESC para el orden descendente.

Tenga en cuenta que `year` y `award` son columnas de las claves de partición. Además, `category` y `rank` son las columnas de agrupación colocadas por orden ascendente (ASC). Juntas, estas columnas forman la clave principal de la tabla.

3. Verifique que se haya creado su tabla.

```
SELECT * from system_schema.tables WHERE keyspace_name='catalog.book_awards' ;
```

La salida debería ser similar a la siguiente.

```
keyspace_name | table_name | bloom_filter_fp_chance | caching | cdc | comment
| compaction | compression | crc_check_chance | dclocal_read_repair_chance
| default_time_to_live | extensions | flags | gc_grace_seconds | id |
max_index_interval | memtable_flush_period_in_ms | min_index_interval |
read_repair_chance | speculative_retry
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+

(0 rows)
```

4. Verifique la estructura de su tabla.

```
SELECT * FROM system_schema.columns WHERE keyspace_name = 'catalog' AND table_name
= 'book_awards';
```

La salida de esta instrucción debe ser similar al siguiente ejemplo.

```
keyspace_name | table_name | column_name | clustering_order | column_name_bytes
| kind      | position | type
```

```

-----+-----+-----+-----
+-----+-----+-----+-----
      catalog | book_awards |      year |      none |
0x79656172 | partition_key |      0 | int
      catalog | book_awards |      award |      none |
0x6177617264 | partition_key |      1 | text
      catalog | book_awards |      category |      asc |
0x63617465676f7279 | clustering |      0 | text
      catalog | book_awards |      rank |      asc |
0x72616e6b | clustering |      1 | int
      catalog | book_awards |      author |      none |
0x617574686672 | regular |      -1 | text
      catalog | book_awards | book_title |      none |
0x626f666b5f7469746c65 | regular |      -1 | text
      catalog | book_awards | publisher |      none |
0x7075626c6973686572 | regular |      -1 | text

(7 rows)

```

Confirme que todas las columnas y tipos de datos sean los deseados. El orden de las columnas podría diferir al de la instrucción CREATE.

Usando el AWS CLI

El siguiente procedimiento crea una tabla con las siguientes columnas y tipos de datos utilizando la AWS CLI. Las columnas `year` y `award` componen la clave de partición con `category` y `rank` como columnas de agrupación en clústeres.

```

year          int
award         text
rank          int
category      text
book_title    text
author        text
publisher     text

```

Para crear una tabla mediante el AWS CLI

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, la `schema-definition` del comando de creación de tabla de esta sección se divide en líneas separadas).

1. Puede crear la tabla con la siguiente instrucción.

```
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-1:111122223333:/keyspace/catalog/  
table/book_awards"  
}
```

2. 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-1:111122223333:/keyspace/catalog/  
table/book_awards",  
  "creationTimestamp": "2024-07-11T15:12:55.571000+00:00",  
  "status": "ACTIVE",  
  "schemaDefinition": {  
    "allColumns": [  
      {  
        "name": "year",  
        "type": "int"  
      },  
      {  
        "name": "award",  
        "type": "text"  
      },  
      {  
        "name": "rank",  
        "type": "int"  
      },  
      {  
        "name": "category",  
        "type": "text"  
      },  
      {  
        "name": "author",  
        "type": "text"  
      },  
      {  
        "name": "book_title",  
        "type": "text"  
      },  
      {  
        "name": "publisher",  
        "type": "text"  
      }  
    ],  
    "partitionKeys": [  
      {  
        "name": "year",  
        "order": 1  
      },  
      {  
        "name": "award",  
        "order": 2  
      }  
    ],  
    "clusteringKeys": [  
      {  
        "name": "category",  
        "order": 1,  
        "clusteringOrder": "ASC"  
      },  
      {  
        "name": "rank",  
        "order": 2,  
        "clusteringOrder": "ASC"  
      }  
    ]  
  }  
}
```

```
        "name": "award",
        "type": "text"
    },
    {
        "name": "category",
        "type": "text"
    },
    {
        "name": "rank",
        "type": "int"
    },
    {
        "name": "author",
        "type": "text"
    },
    {
        "name": "book_title",
        "type": "text"
    },
    {
        "name": "publisher",
        "type": "text"
    }
],
"partitionKeys": [
    {
        "name": "year"
    },
    {
        "name": "award"
    }
],
"clusteringKeys": [
    {
        "name": "category",
        "orderBy": "ASC"
    },
    {
        "name": "rank",
        "orderBy": "ASC"
    }
],
"staticColumns": []
},
```

```
"capacitySpecification": {
  "throughputMode": "PAY_PER_REQUEST",
  "lastUpdateToPayPerRequestTimestamp": "2024-07-11T15:12:55.571000+00:00"
},
"encryptionSpecification": {
  "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
  "status": "DISABLED"
},
"defaultTimeToLive": 0,
"comment": {
  "message": ""
},
"replicaSpecifications": []
}
```

Para realizar operaciones CRUD (crear, leer, actualizar y eliminar) en los datos de su tabla, vaya a [the section called “Operaciones CRUD”](#).

Comprobación del estado de creación de las 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 el Consola de administración de AWS, lo que indica si 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
```

Creación, lectura, actualización y eliminación de datos (CRUD) mediante CQL en Amazon Keyspaces

En este paso del tutorial, aprenderá a insertar, leer, actualizar y eliminar datos en una tabla de Amazon Keyspaces mediante instrucciones en el lenguaje de manipulación de datos (DML) de CQL. En Amazon Keyspaces, solo puede crear instrucciones DML en lenguaje CQL. En este tutorial, practicará la ejecución de instrucciones DML utilizando la `cqlsh-expansion` con [AWS CloudShell](#) en la Consola de administración de AWS.

- **Inserción de datos:** en esta sección se describe la inserción de registros individuales y múltiples en una tabla mediante la instrucción `INSERT`. Aprenderá a cargar datos desde un archivo CSV y a comprobar que las inserciones se hayan realizado correctamente mediante consultas `SELECT`.
- **Lectura de datos:** aquí explorará diferentes variaciones de la instrucción `SELECT` para recuperar datos de una tabla. Entre los temas se incluyen la selección de todos los datos, la selección de columnas específicas, el filtrado de filas en función de las condiciones mediante la cláusula `WHERE` y la comprensión de las condiciones simples y compuestas.
- **Actualización de datos:** en esta sección, aprenderá a modificar los datos existentes en una tabla mediante la instrucción `UPDATE`. Practicará la actualización de columnas únicas y múltiples y, al mismo tiempo, comprenderá las restricciones relativas a la actualización de las columnas de clave principal.

- Eliminación de datos: la última sección trata sobre la eliminación de datos de una tabla mediante la instrucción DELETE. Aprenderá a eliminar celdas específicas y filas enteras y verá cuáles son las implicaciones de eliminar datos en lugar de eliminar toda la tabla o todo el espacio de claves.

A lo largo del tutorial, encontrará ejemplos, consejos y oportunidades para practicar cómo escribir sus propias consultas CQL para distintos escenarios.

Temas

- [Inserción y carga de datos en una tabla de Amazon Keyspaces](#)
- [Lectura de los datos de una tabla con la instrucción CQL SELECT en Amazon Keyspaces](#)
- [Actualización de datos en una tabla de Amazon Keyspaces con CQL](#)
- [Eliminación de los datos de una tabla mediante la sentencia CQL DELETE](#)

Inserción y carga de datos en una tabla de Amazon Keyspaces

Para crear datos en su tabla `book_awards`, utilice la instrucción INSERT para añadir una única fila.

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar `us-east-1` con su propia región.

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

La salida de ese comando debería tener este aspecto.

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

2. 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 Consola de administración de AWS.

```
CONSISTENCY LOCAL_QUORUM;
```

- Para insertar un único registro, ejecute el siguiente comando en el editor CQL.

```
INSERT INTO catalog.book_awards (award, year, category, rank, author, book_title,
publisher)
VALUES ('Wolf', 2023, 'Fiction',3,'Shirley Rodriguez','Mountain', 'AnyPublisher') ;
```

- Verifique que los datos se hayan añadido correctamente a su tabla; para ello, ejecute el siguiente comando.

```
SELECT * FROM catalog.book_awards;
```

La salida de la sentencia debería tener el siguiente aspecto.

```
year | award | category | rank | author          | book_title | publisher
-----+-----+-----+-----+-----+-----+-----
2023 | Wolf  | Fiction  | 3    | Shirley Rodriguez | Mountain   | AnyPublisher

(1 rows)
```

Para insertar varios registros desde un archivo con cqlsh

- Descargue el archivo CSV de ejemplo (keyspaces_sample_table.csv), contenido en el archivo comprimido [samplmigration.zip](#). Descomprima el archivo y tome nota de la ruta a keyspaces_sample_table.csv.

award	year	category	rank	author	book_title	publisher
Kwesi Manu Prize	2020	Fiction	1	Akua Mansa	Where did you go?	SomePublisher
Kwesi Manu Prize	2020	Fiction	2	John Stiles	Yesterday	Example Books
Kwesi Manu Prize	2020	Fiction	3	Nikki Wolf	Moving to the Chateau	AnyPublisher
Wolf	2020	Non-Fiction	1	Wang Xiulan	History of Ideas	Example Books
Wolf	2020	Non-Fiction	2	Ana Carolina Silva	Science Today	SomePublisher
Wolf	2020	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice	AnyPublisher
Richard Roe	2020	Fiction	1	Alejandro Rosalez	Long Summer	SomePublisher
Richard Roe	2020	Fiction	2	Arnav Desai	The Key	Example Books
Richard Roe	2020	Fiction	3	Mateo Jackson	Inside the Whale	AnyPublisher

- Abre AWS CloudShell Consola de administración de AWS y conéctate a Amazon Keyspaces mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

- En la línea de comandos de cqlsh (cqlsh>), especifique un espacio de claves.

```
USE catalog ;
```

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. AWS CloudShell Selecciona Acciones en la parte superior derecha de la pantalla y, a continuación, selecciona Cargar archivo para cargar el archivo csv descargado anteriormente. Tome nota de la ruta al archivo.
6. En la línea de comandos de espacio de claves (cqlsh:catalog>), ejecute la siguiente instrucción.

```
COPY book_awards (award, year, category, rank, author, book_title, publisher) FROM
'/home/cloudshell-user/keyspaces_sample_table.csv' WITH header=TRUE ;
```

La salida de la instrucción debe ser similar a esta.

```
cqlsh:catalog> COPY book_awards (award, year, category, rank, author,
book_title, publisher) FROM '/home/cloudshell-user/
keyspaces_sample_table.csv' WITH delimiter=',' AND header=TRUE ;
cqlsh current consistency level is LOCAL_QUORUM.
Reading options from /home/cloudshell-user/.cassandra/cqlshrc:[copy]:
{'numprocesses': '16', 'maxattempts': '1000'}
Reading options from /home/cloudshell-user/.cassandra/cqlshrc:[copy-from]:
{'ingestrate': '1500', 'maxparseerrors': '1000', 'maxinserterrors': '-1',
'maxbatchsize': '10', 'minbatchsize': '1', 'chunksize': '30'}
Reading options from the command line: {'delimiter': ',', 'header': 'TRUE'}
Using 16 child processes

Starting copy of catalog.book_awards with columns [award, year, category, rank,
author, book_title, publisher].
OSError: handle is closed      0 rows/s; Avg. rate:      0 rows/s
Processed: 9 rows; Rate:      0 rows/s; Avg. rate:      0 rows/s
9 rows imported from 1 files in 0 day, 0 hour, 0 minute, and 26.706 seconds (0
skipped).
```

7. Verifique que los datos se hayan añadido correctamente a su tabla; para ello, ejecute la siguiente consulta.

```
SELECT * FROM book_awards ;
```

Debería ver la siguiente salida.

```

year | award | category | rank | author | book_title
-----+-----+-----+-----+-----+-----
+-----+-----+-----+-----+-----+-----
2020 |      Wolf | Non-Fiction | 1 | Wang Xiulan | History
of Ideas | Example Books
2020 |      Wolf | Non-Fiction | 2 | Ana Carolina Silva |
Science Today | SomePublisher
2020 |      Wolf | Non-Fiction | 3 | Shirley Rodriguez | The Future of
Sea Ice | AnyPublisher
2020 | Kwesi Manu Prize | Fiction | 1 | Akua Mansa | Where did
you go? | SomePublisher
2020 | Kwesi Manu Prize | Fiction | 2 | John Stiles |
Yesterday | Example Books
2020 | Kwesi Manu Prize | Fiction | 3 | Nikki Wolf | Moving to the
Chateau | AnyPublisher
2020 | Richard Roe | Fiction | 1 | Alejandro Rosalez | Long
Summer | SomePublisher
2020 | Richard Roe | Fiction | 2 | Arnav Desai |
The Key | Example Books
2020 | Richard Roe | Fiction | 3 | Mateo Jackson | Inside
the Whale | AnyPublisher

(9 rows)
```

Para obtener más información sobre cómo usar `cqlsh COPY` para cargar datos desde archivos csv a una tabla de Amazon Keyspaces, consulte [the section called “Carga de datos con cqlsh”](#).

Lectura de los datos de una tabla con la instrucción CQL **SELECT** en Amazon Keyspaces

En la sección [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 la tabla](#)
- [Selección de un subconjunto de columnas](#)
- [Selección de un subconjunto de filas](#)

Selección de todos los datos de la 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 la tabla.

Para seleccionar todos los datos de su tabla

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

2. Ejecute la siguiente consulta.

```
SELECT * FROM catalog.book_awards ;
```

El uso del carácter comodín (*) para *column_list* selecciona todas las columnas. La salida de la instrucción tiene un aspecto semejante al del siguiente ejemplo.

```
year | award          | category    | rank | author          | book_title
    | publisher
-----+-----+-----+-----+-----+-----
+-----+-----+-----+-----+-----+-----
```

```

2020 |           Wolf | Non-Fiction | 1 |           Wang Xiulan |           History
of Ideas | AnyPublisher
2020 |           Wolf | Non-Fiction | 2 | Ana Carolina Silva |
Science Today | SomePublisher
2020 |           Wolf | Non-Fiction | 3 | Shirley Rodriguez | The Future of
Sea Ice | AnyPublisher
2020 | Kwesi Manu Prize | Fiction | 1 |           Akua Mansa |           Where did
you go? | SomePublisher
2020 | Kwesi Manu Prize | Fiction | 2 |           John Stiles |
Yesterday | Example Books
2020 | Kwesi Manu Prize | Fiction | 3 |           Nikki Wolf | Moving to the
Chateau | AnyPublisher
2020 |           Richard Roe | Fiction | 1 | Alejandro Rosalez |           Long
Summer | SomePublisher
2020 |           Richard Roe | Fiction | 2 |           Arnav Desai |
The Key | Example Books
2020 |           Richard Roe | Fiction | 3 |           Mateo Jackson |           Inside
the Whale | AnyPublisher

```

Selección de un subconjunto de columnas

Para consultar un subconjunto de columnas

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

2. Para recuperar solo las columnas award, category y year, ejecute la siguiente consulta.

```
SELECT award, category, year FROM catalog.book_awards ;
```

La salida contiene solo las columnas especificadas en el orden indicado en la instrucción SELECT.

```

award           | category       | year
-----+-----+-----
           Wolf | Non-Fiction | 2020
           Wolf | Non-Fiction | 2020
           Wolf | Non-Fiction | 2020
Kwesi Manu Prize | Fiction       | 2020

```

Kwesi Manu Prize		Fiction		2020
Kwesi Manu Prize		Fiction		2020
Richard Roe		Fiction		2020
Richard Roe		Fiction		2020
Richard Roe		Fiction		2020

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

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

2. Para recuperar solo los registros de los premios de un año determinado, ejecute la siguiente consulta.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Wolf' ;
```

La instrucción SELECT anterior devuelve la siguiente salida.

year	award	category	rank	author	book_title	publisher
2020	Wolf	Non-Fiction	1	Wang Xiulan	History of Ideas	AnyPublisher
2020	Wolf	Non-Fiction	2	Ana Carolina Silva	Science Today	SomePublisher
2020	Wolf	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice	AnyPublisher

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 de partición de la tabla.
- Ha añadido **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 que puede usar en la cláusula **WHERE** deben incluir todas o un subconjunto de las columnas de la clave de partición de la tabla. Si desea utilizar solo un subconjunto de las columnas de la cláusula **WHERE**, debe incluir un conjunto contiguo de columnas de clave de partición de izquierda a derecha, empezando por la primera columna de la clave de partición. Por ejemplo, si las columnas de la clave de partición son `year`, `month` y `award`, puede usar las siguientes columnas en la cláusula **WHERE**:
 - `year`
 - `year Y month`
 - `year Y month Y award`
- 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 `book_awards`:

- Busque los ganadores de los premios Wolf de 2020 y haga que se muestren los títulos y autores de los libros, ordenados por su puesto en la clasificación.
- Haga que se muestren los ganadores del primer puesto de todos los premios de 2020 y que aparezcan los títulos de los libros y los nombres de los premios.

Actualización de datos en una tabla de Amazon Keyspaces con CQL

Para actualizar los datos de su tabla `book_awards`, 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 col3='1' ;
```

- 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

Con la tabla `book_awards`, cambie el nombre de una editorial del ganador de los premios Wolf de no ficción en 2020.

```
UPDATE book_awards SET publisher='new Books' WHERE year = 2020 AND award='Wolf' AND
category='Non-Fiction' AND rank=1;
```

Compruebe que la editorial sea ahora `new Books`.

```
SELECT * FROM book_awards WHERE year = 2020 AND award='Wolf' AND category='Non-Fiction'
AND rank=1;
```

La instrucción debería devolver la siguiente salida.

year	award	category	rank	author	book_title	publisher
2020	Wolf	Non-Fiction	1	Wang Xiulan	History of Ideas	new Books

Pruébelo

Avanzado: el ganador del “Premio Kwezi Manu” de ficción de 2020 ha cambiado de nombre. Actualice este registro para cambiar el nombre a 'Akua Mansa-House'.

Eliminación de los datos de una tabla mediante la sentencia CQL **DELETE**

Para eliminar datos en su tabla `book_awards`, 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. Puede utilizar la recuperación de Amazon Keyspaces Point-in-time (PITR) para restaurar las tablas eliminadas; para obtener más información, consulte [the section called “Copias de seguridad y restauración con recuperación en un momento dado”](#) Para obtener información sobre cómo restaurar una tabla eliminada con la PITR activada, consulte [the section called “Restauración de una tabla eliminada”](#).

Eliminación de celdas

Al eliminar una columna de una fila se eliminan los datos de la celda especificada. Al mostrar esa columna mediante una SELECT sentencia, los datos se muestran como *null*, aunque no se almacene un valor nulo en esa ubicación.

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 la tabla `book_awards`, puede ver que el título del libro que ganó el primer premio “Richard Roe” de 2020 es “Long Summer”. Imagine que se ha retirado del mercado este título y necesita eliminar los datos de esta celda.

Para eliminar una celda en concreto

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

2. Ejecute la siguiente consulta DELETE.

```
DELETE book_title FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe'
AND category='Fiction' AND rank=1;
```

3. Compruebe que la solicitud de eliminación se haya realizado como se esperaba.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND
category='Fiction' AND rank=1;
```

La salida de esta instrucción tiene el siguiente aspecto.

year	award	category	rank	author	book_title	publisher
2020	Richard Roe	Fiction	1	Alejandro Rosalez	null	SomePublisher

Eliminación de filas

Podría darse el caso de que necesite eliminar una fila entera, por ejemplo, para cumplir con una solicitud de eliminación de datos. La sintaxis general para eliminar una fila es la siguiente.

```
DELETE FROM table_name WHERE condition ;
```

Para eliminar una fila

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

2. Ejecute la siguiente consulta DELETE.

```
DELETE FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND  
category='Fiction' AND rank=1;
```

3. Compruebe que la eliminación se haya realizado como se esperaba.

```
SELECT * FROM catalog.book_awards WHERE year=2020 AND award='Richard Roe' AND  
category='Fiction' AND rank=1;
```

El resultado de esta instrucción tiene este aspecto después de eliminar la fila.

```
year | award | category | rank | author | book_title | publisher  
-----+-----+-----+-----+-----+-----+-----  
  
(0 rows)
```

Puede eliminar automáticamente los datos caducados de la tabla con el periodo de vida de Amazon Keyspaces. Para obtener más información, consulte [the section called “Caducidad de datos con período de vida”](#).

Eliminación de una tabla en Amazon Keyspaces

Para evitar que le cobren por tablas y datos que no necesite, elimine todas las tablas 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.

Puede eliminar una tabla utilizando la consola, CQL o la AWS CLI. 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 Consola de administración de AWS.

Para eliminar una tabla con la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 `book_awards` ya no aparezca en la lista.

Uso de CQL

El siguiente procedimiento elimina una tabla y todos sus datos utilizando CQL.

Eliminación de una tabla con CQL

1. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar `us-east-1` con su propia región.

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

2. Elimine la tabla introduciendo la siguiente instrucción.

```
DROP TABLE IF EXISTS catalog.book_awards ;
```

3. Verifique que se haya eliminado su tabla.

```
SELECT * FROM system_schema.tables WHERE keyspace_name = 'catalog' ;
```

La salida debería tener este aspecto. Tenga en cuenta que esto puede llevar algún tiempo, así que vuelva a ejecutar la instrucción después de un minuto si no ve este resultado.

```
keyspace_name | table_name | bloom_filter_fp_chance | caching | cdc | comment
| compaction | compression | crc_check_chance | dclocal_read_repair_chance
| default_time_to_live | extensions | flags | gc_grace_seconds | id |
max_index_interval | memtable_flush_period_in_ms | min_index_interval |
read_repair_chance | speculative_retry
-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+

(0 rows)
```

Usando el AWS CLI

El siguiente procedimiento elimina una tabla y todos sus datos utilizando la AWS CLI.

Para eliminar una tabla mediante el AWS CLI

1. Abra CloudShell
2. Elimine la tabla con la siguiente instrucción.

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

3. Para comprobar que la tabla se ha eliminado, puede enumerar todas las tablas en un espacio de claves.

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

Debería ver la siguiente salida. Tenga en cuenta que esta operación asíncrona puede tardar algún tiempo. Re-run vuelva a ejecutar el comando después de un rato para confirmar que la tabla se ha eliminado.

```
{  
  "tables": []  
}
```

Eliminación de un espacio de claves en Amazon Keyspaces

Para evitar que se le cobren cargos por espacios de claves, elimine todos aquellos que no utilice. Al eliminar un espacio de claves, el espacio de claves y todas sus tablas se eliminan y deja de acumular cargos por ellos.

Puede eliminar un espacio de claves con la consola, CQL o la AWS CLI.

Uso de la consola

El siguiente procedimiento elimina un espacio de claves y todas sus tablas y datos utilizando la consola.

Para eliminar un espacio de claves con la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 `catalog`, 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 `book_awards` 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. Abre Amazon Keyspaces AWS CloudShell y conéctate a ellos mediante el siguiente comando. Asegúrese de actualizar *us-east-1* con su propia región.

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

2. Elimine el espacio de claves introduciendo la siguiente instrucción.

```
DROP KEYSPACE IF EXISTS catalog ;
```

3. Compruebe que el espacio de claves haya sido eliminado.

```
SELECT * from system_schema.keyspaces ;
```

El 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. Una vez eliminado el espacio de claves, el resultado de la instrucción debería tener este aspecto.

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

(4 rows)
```

Usando el AWS CLI

El siguiente procedimiento elimina un espacio de claves y todas sus tablas y datos utilizando la AWS CLI.

Para eliminar un espacio de claves mediante el AWS CLI

1. Abra AWS CloudShell
2. Elimine el espacio de claves introduciendo la siguiente instrucción.

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

3. Compruebe que el espacio de claves haya sido eliminado.

```
aws keyspaces list-keyspaces
```

El resultado de esta sentencia debería tener un aspecto similar al siguiente y mostrar únicamente los espacios de claves del sistema. Tenga en cuenta que, dado que se trata de una operación asíncrona, puede haber un retraso hasta que se elimine el espacio de claves.

```
{
  "keyspaces": [
    {
      "keyspaceName": "system_schema",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system_schema/",
      "replicationStrategy": "SINGLE_REGION"
    },
    {
      "keyspaceName": "system_schema_mcs",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system_schema_mcs/",
      "replicationStrategy": "SINGLE_REGION"
    },
    {
      "keyspaceName": "system",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system/",
      "replicationStrategy": "SINGLE_REGION"
    },
    {
      "keyspaceName": "system_multiregion_info",
      "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system_multiregion_info/",
      "replicationStrategy": "SINGLE_REGION"
    }
  ]
}
```

```
}
```

Tutoriales y soluciones para Amazon Keyspaces

En este tema se incluyen varios tutoriales que explican los pasos detallados necesarios para conectarse a Amazon Keyspaces mediante programación o para utilizar Amazon Keyspaces con otros servicios para crear soluciones. AWS Los tutoriales disponibles cubren diferentes escenarios populares de acceso entre servicios y muestran cómo integrar Amazon Keyspaces con Amazon Virtual Private Cloud, Amazon Elastic Kubernetes Service, Amazon Simple AWS Glue Storage Service o con tecnologías de código abierto como Apache Spark.

Para ver tutoriales paso a paso que muestran cómo conectarse a Amazon Keyspaces mediante los controladores Apache Cassandra, consulte. [the section called “Utilización de un controlador de cliente de Cassandra”](#)

Temas

- [Tutorial: Conéctese a Amazon Keyspaces mediante un punto de enlace de VPC de interfaz](#)
- [Tutorial: Intégrelo con Apache Spark para importar o exportar datos](#)
- [Tutorial: Conectarse desde una aplicación contenerizada alojada en Amazon Elastic Kubernetes Service](#)
- [Tutorial: Exportación de una tabla de Amazon Keyspaces a Amazon S3 mediante AWS Glue](#)

Tutorial: Conéctese a Amazon Keyspaces mediante un punto de enlace 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 de Amazon EC2

1. Abra la consola de Amazon EC2 en <https://console.aws.amazon.com/ec2/>.
2. Elija Lanzar instancia 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 elegible. 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 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 Seleccionar un par de claves 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 Edit (Edición de).
- 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 lo tenga todo 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 Estado de la instancia tenga el valor `running` y de que se hayan superado todas las Comprobaciones de estado.

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 obtener información sobre otras maneras de conectarse, consulte [Conexión con la instancia de Linux](#) en la Guía del usuario de Amazon EC2.

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 Agregar 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, vaya a Grupos de seguridad asociados e introduzca 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 seleccione Guardar.
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.

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. Agregue reglas al grupo de seguridad de su VPC que permitan el acceso HTTP, HTTPS y SSH entrante desde direcciones IPv6.
7. Para confirmar que su punto de enlace de VPC se ha configurado correctamente, debe utilizar una `cqlsh` conexión a Amazon Keyspaces. Si utiliza su entorno local o el editor CQL de Amazon Keyspaces en el Consola de administración de AWS, 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 preparado para crear un punto final de VPC de doble pila para Amazon Keyspaces.

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

En este paso, creará un punto de enlace de VPC de doble pila para Amazon Keyspaces mediante 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"
    "api.aws.us-east-1.cassandra-streams"
  ]
}

```

Si Amazon Keyspaces es uno de los servicios disponibles en el resultado del comando, puede continuar con la creación de un punto de enlace de VPC.

3. Para conectarse a Amazon Keyspaces mediante puntos de enlace de doble pila habilitados para IPv6, confirme que su VPC es compatible con IPv6 y configure las subredes con compatibilidad con IPv6. Para añadir compatibilidad con IPv6 a una VPC existente que actualmente solo admite IPv4, [consulte Compatibilidad con IPv6 para su VPC en la Guía del usuario de Amazon VPC](#);
4. Determine el identificador de su VPC.

```
aws ec2 describe-vpcs
```

```

{
  "Vpcs": [
    {
      "OwnerId": "111122223333",
      "InstanceTenancy": "default",
      "Ipv6CidrBlockAssociationSet": [
        {
          "AssociationId": "vpc-cidr-assoc-0000aaa0a00a00aa0",

```

```

        "Ipv6CidrBlock": "2600:1f18:e19:7d00::/56",
        "Ipv6CidrBlockState": {
            "State": "associated"
        },
        "NetworkBorderGroup": "us-east-1",
        "Ipv6Pool": "Amazon",
        "Ipv6AddressAttribute": "public",
        "IpSource": "amazon"
    }
],
"CidrBlockAssociationSet": [
    {
        "AssociationId": "vpc-cidr-assoc-00a0000a",
        "CidrBlock": "111.11.0.0/16",
        "CidrBlockState": {
            "State": "associated"
        }
    }
],
"IsDefault": true,
"BlockPublicAccessStates": {
    "InternetGatewayBlockMode": "off"
},
"VpcId": "vpc-a1234bcd",
"State": "available",
"CidrBlock": "111.11.0.0/16",
"DhcpOptionsId": "dopt-a00aaaaa"
}
]
}

```

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

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

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

```

{
    "Subnets": [
        {
            "AvailabilityZoneId": "use1-az1",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "111122223333",
            "AssignIpv6AddressOnCreation": false,

```

```

    "Ipv6CidrBlockAssociationSet": [
      {
        "AssociationId": "subnet-cidr-assoc-05d75732736740283",
        "Ipv6CidrBlock": "*****",
        "Ipv6CidrBlockState": {
          "State": "associated"
        },
        "Ipv6AddressAttribute": "public",
        "IpSource": "amazon"
      }
    ],
    "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-70b24b16",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
      "HostnameType": "ip-name",
      "EnableResourceNameDnsARecord": false,
      "EnableResourceNameDnsAAAARecord": false
    },
    "BlockPublicAccessStates": {
      "InternetGatewayBlockMode": "off"
    },
    "SubnetId": "subnet-70b24b16",
    "State": "available",
    "VpcId": "vpc-a1234bcd",
    "CidrBlock": "*****/20",
    "AvailableIpAddressCount": 4089,
    "AvailabilityZone": "us-east-1a",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true
  },
  {
    "AvailabilityZoneId": "use1-az2",
    "MapCustomerOwnedIpOnLaunch": false,
    "OwnerId": "111122223333",
    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [
      {
        "AssociationId": "subnet-cidr-assoc-0ec6fb253e05b17eb",
        "Ipv6CidrBlock": "*****",
        "Ipv6CidrBlockState": {
          "State": "associated"
        }
      },

```

```

        "Ipv6AddressAttribute": "public",
        "IpSource": "amazon"
    },
    ],
    "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/subnet-
c63ffbe7",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
        "HostnameType": "ip-name",
        "EnableResourceNameDnsARecord": false,
        "EnableResourceNameDnsAAAARecord": false
    },
    "BlockPublicAccessStates": {
        "InternetGatewayBlockMode": "off"
    },
    "SubnetId": "subnet-c63ffbe7",
    "State": "available",
    "VpcId": "vpc-a1234bcd",
    "CidrBlock": "*/20",
    "AvailableIpAddressCount": 4087,
    "AvailabilityZone": "us-east-1b",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true
    }
]
}

```

En la salida del ejemplo, hay dos ID de subred disponibles: subnet-70b24b16 y subnet-c63ffbe7.

6. 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-ids`, especifique los ID de subred del paso anterior. Utilice el parámetro `--vpc-endpoint-type` para definir el punto de conexión como interfaz. Para crear un punto final de doble pila, utilice `--ip-address-type dualstack`. 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 \
  --ip-address-type dualstack \

```

```

--service-name com.amazonaws.us-east-1.cassandra \
--subnet-ids subnet-70b24b16 subnet-c63ffbe7

{
  "VpcEndpoint": {
    "VpcEndpointId": "vpce-000000abc111d2ef3",
    "VpcEndpointType": "Interface",
    "VpcId": "vpc-a1234bcd",
    "ServiceName": "com.amazonaws.us-east-1.cassandra",
    "State": "pending",
    "RouteTableIds": [],
    "SubnetIds": [
      "subnet-70b24b16",
      "subnet-c63ffbe7"
    ],
    "Groups": [
      {
        "GroupId": "sg-0123456789",
        "GroupName": "default"
      }
    ],
    "IpAddressType": "dualstack",
    "DnsOptions": {
      "DnsRecordIpType": "dualstack"
    },
    "PrivateDnsEnabled": true,
    "RequesterManaged": false,
    "NetworkInterfaceIds": [
      "eni-08cd525f72ea6f1fa",
      "eni-07b1f6c895169d8fb"
    ],
    "DnsEntries": [
      {
        "DnsName": "vpce-0000000000-1234567.cassandra.us-
east-1.vpce.amazonaws.com",
        "HostedZoneId": "Z7HUB22UULQXV"
      },
      {
        "DnsName": "vpce-0000000000-1234567-us-east-1a.cassandra.us-
east-1.vpce.amazonaws.com",
        "HostedZoneId": "Z7HUB22UULQXV"
      },
      {
        "DnsName": "cassandra.us-east-1.amazonaws.com",

```

```
        "HostedZoneId": "ZONEIDPENDING"
      },
      {
        "DnsName": "cassandra.us-east-1.api.aws",
        "HostedZoneId": "ZONEIDPENDING"
      }
    ],
    "CreationTimestamp": "2025-09-19T15:19:19.266000+00:00",
    "OwnerId": "111122223333",
    "ServiceRegion": "us-east-1"
  }
}
```

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 Administrar grupos de seguridad y, a continuación, elija el grupo de seguridad asociado a este punto final.
3. Elija Reglas de entrada y, a continuación, Editar reglas de entrada.
4. Agregue una regla de entrada con el Tipo CQLSH / CASSANDRA. Esto establecerá automáticamente el Intervalo de puertos en 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 Consola de administración de AWS IAM y ábrala en. <https://console.aws.amazon.com/iam/>
 - b. En el panel de la consola de IAM, elija Usuarios y, a continuación, seleccione el usuario de IAM en la lista.
 - c. En la página Resumen, elija Añadir permisos.
 - d. Elija Asociar 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 la [AWS CLI Referencia de comandos de la](#).

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 “Identity-based ejemplos de políticas”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/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 Consola de administración de AWS 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 Usuarios y, a continuación, seleccione el usuario de IAM en la lista.
 - g. En la página Resumen, elija Añadir permisos.
 - h. Elija Asociar 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 cqlsh-expansion"](#).

```
SELECT * FROM system.peers;
```

El resultado devuelve nodos con direcciones IP IPv6 privadas, según la configuración de VPC y subred en su región. AWS

```
peer                                | data_center | host_id
      | preferred_ip                | rack      | release_version
      | rpc_address                  | schema_version |
tokens
-----+-----
+-----+-----+-----+-----+
+-----+-----+-----+-----+
+-----+-----+-----+-----+
+-----+-----+-----+-----+
+-----+-----+-----+-----+
2600:1111:2222:3333:283b:8e6:d04f | us-east-1 | dddddd-7a22-3582-
a73d-49338a686a53 | 2600:1111:2222:3333:283b:8e6:d04f | us-east-1 |
3.11.2 | 2600:1111:2222:3333:283b:8e6:d04f | 05deae2d-6405-494d-a965-
c0e5836bcb3c | {'85070591730234615865843651857942052863'}
2600:1111:2222:4444:7d26:5a09:1b44 | us-east-1 | 66666666-035d-37ef-
a247-19a6a867ab09 | 2600:1111:2222:4444:7d26:5a09:1b44 | us-east-1 |
3.11.2 | 2600:1111:2222:4444:7d26:5a09:1b44 | 05deae2d-6405-494d-a965-
c0e5836bcb3c | {'170141183460469231731687303715884105726'}
```

 Note

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 Consola de administración de AWS, 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. Abra 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, selecciona `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 controlar si superas la cuota del número máximo de operaciones por segundo y por conexión utilizando la `PerConnectionRequestRateExceeded` CloudWatch métrica del espacio de AWS/Cassandra nombres. 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 con un controlador de Node.js cliente, consulte [the section called “Uso de un controlador de cliente Cassandra Node.js”](#).

El siguiente ejemplo de código utiliza el *keyspaces-bundle.pem* archivo del certificado digital. Para obtener más información, consulte [the section called “Antes de empezar”](#).

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/keyspaces-bundle.pem',
      'utf-8')],
  host: 'cassandra.us-east-1.amazonaws.com',
  rejectUnauthorized: true
};
const client = new cassandra.Client({
  contactPoints: ['cassandra.us-east-1.amazonaws.com'],
  localDataCenter: 'us-east-1',
  pooling: { coreConnectionsPerHost: { [types.distance.local]:
    1 } },
  consistency: types.consistencies.localQuorum,
  queryOptions: { isIdempotent: true },
  authProvider: auth,
  sslOptions: sslOptions1,
```

```
        protocolOptions: { port: 9142 }  
    });
```

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\":\"2000-00-00\",\"Statement\":[{\"Effect\":\"  
\\\"Allow\\\",\\\"Principal\\\":\\\"*\\\",\\\"Action\\\":\\\"*\\\",\\\"Resource\\\":\\\"*\\\"}]}\",  
    "VpcId": "vpc-a1234bcd",  
    "State": "available",  
    "ServiceName": "com.amazonaws.us-east-1.cassandra",  
    "RouteTableIds": [],  
    "VpcEndpointId": "vpce-1a23b4c5",  
    "CreationTimestamp": "2025-07-26T22:00:14Z"  
  }  
}
```

En el resultado de ejemplo, el ID de punto de conexión de VPC es ***vpce-1a23b4c5***. Asegúrese de reemplazar este valor por el suyo propio.

3. Elimine el punto de conexión de VPC.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-1a23b4c5  
  
{  
  "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 Instances (Instancias).
3. Seleccione la instancia de Amazon EC2.
4. Elija Acciones, luego Estado de la instancia y, a continuación, Terminar.
5. En la ventana de confirmación, elija Sí, terminar.

Tutorial: Intégrelo con Apache Spark para importar o exportar datos

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 el conector Cassandra de Spark agiliza la ejecución de las cargas de trabajo de Cassandra en las canalizaciones de Spark-based análisis mediante un servicio de base de datos totalmente gestionado y 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. Lo haría para ejecutar las cargas de trabajo de Cassandra en canalizaciones de análisis. Spark-based

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://d1cdn.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 ayudan a aprovisionar suficiente read/write capacidad 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 el read/write acceso a la tabla de usuarios y el acceso de lectura a las tablas del sistema, tal y como se muestra en el ejemplo de política de IAM que se muestra 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 para configurar una capacidad de read/write rendimiento suficiente para que su tabla de Amazon Keyspaces admita el tráfico del conector Spark Cassandra.
- 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 “Administración de la capacidad de rendimiento con 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 “Configure los modos read/write de capacidad”](#).

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 usar el conector Spark Cassandra 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 de Java. DataStax Para conectarse, puede utilizar credenciales específicas del servicio o el complemento SigV4.

Si aún no lo ha hecho, debe convertir el certificado digital utilizado para crear la conexión TLS 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 `application.conf` archivo de ejemplo que puede usar al conectarse con AWS credenciales y el complemento SigV4. 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 necesarias para la autenticación 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 “Creación de 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: preparación de 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 mediante `cqlsh-expansion` Para ver las instrucciones de instalación de `cqlsh-expansion`, consulte [the section called “Uso de cqlsh-expansion”](#).

Sustituya el punto final del servicio en el siguiente ejemplo por su propio valor.

```
cqlsh-expansion cassandra.us-east-1.amazonaws.com 9142 --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 “Comprobación del estado de creación del espacio 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_table.csv")
```

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 read/write permisos necesarios para acceder a la tabla de usuarios y leer 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, puedes usar CloudWatch las métricas de Amazon para ayudarte a solucionar problemas con la configuración del conector Spark Cassandra 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 superas esta cuota, Amazon Keyspaces emitirá una `PerConnectionRequestRateExceeded` métrica in. CloudWatch

Si observas la presencia de `PerConnectionRequestRateExceeded` eventos junto con otros errores del sistema o del usuario, es probable que Spark esté realizando varios reintentos por encima del número de solicitudes asignado 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 1 000 WCU o WRU por cada second/3 000 RCU o RRU por segundo, por partición. Si ve `StoragePartitionThroughputCapacityExceeded` CloudWatch eventos, podría indicar que los datos no se distribuyen aleatoriamente durante 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.

Tutorial: Conectarse desde una aplicación contenerizada alojada en Amazon Elastic Kubernetes Service

Este tutorial explica los pasos necesarios para configurar un clúster de Amazon Elastic Kubernetes Service (Amazon EKS) para alojar una aplicación en contenedor que se conecte a Amazon Keyspaces mediante la autenticación SigV4.

Amazon EKS es un servicio administrado 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 instrucciones paso a paso para configurar, desarrollar e implementar una aplicación Java en contenedores en Amazon EKS. En el último paso, se ejecuta la aplicación para que escriba datos en una tabla de Amazon Keyspaces.

Temas

- [Requisitos previos para conectarse desde Amazon EKS a Amazon Keyspaces](#)
- [Paso 1: configuración del clúster de Amazon EKS y configuración de los permisos de IAM](#)
- [Paso 2: configuración de la aplicación](#)
- [Paso 3: creación de la imagen de la aplicación y subida del archivo de Docker a su repositorio de Amazon ECR](#)
- [Paso 4: implementación de la aplicación en Amazon EKS y escritura de los datos en la tabla](#)
- [Paso 5: \(opcional\) eliminación](#)

Requisitos previos para conectarse desde Amazon EKS a Amazon Keyspaces

Cree los siguientes AWS recursos antes de empezar con el tutorial

1. Antes de empezar este tutorial, siga las instrucciones AWS de 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.
2. Cree un espacio de claves de Amazon Keyspaces con el nombre `aws` y una tabla con el nombre `user` en la que pueda escribir desde la aplicación en contenedores que se ejecuta en Amazon EKS y que aparecerá 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 el espacio de claves se ha creado, 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 la tabla se ha creado, puede utilizar el siguiente comando.

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

Para obtener más información, consulte [create keyspace](#) y [create table](#) en la Referencia de comandos de la AWS CLI .

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 verificar que la tabla se ha creado, puede utilizar la siguiente instrucción.

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

Su tabla debería aparecer en el resultado de esta instrucción. 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 computación sin servidor que le permite implementar pods de Kubernetes sin administrar instancias de Amazon EC2. Para seguir este tutorial sin tener que actualizar el nombre del clúster en todos los comandos de ejemplo, cree un clúster con el nombre `my-eks-cluster` siguiendo las instrucciones de [Introducción a Amazon EKS — eksctl](#) en la Guía del usuario

de Amazon EKS. Cuando se haya creado el clúster, compruebe que los nodos y los dos pods predeterminados estén funcionando y en buen estado. Puede 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. Instale Docker. Para obtener instrucciones sobre cómo instalar Docker en una instancia de Amazon EC2, consulte [Instalación de 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 utilizar el siguiente comando para crear un repositorio con el 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: configuración del clúster de Amazon EKS”](#).

Paso 1: configuración del clúster de Amazon EKS y configuración de los permisos de IAM

Configuración del clúster de Amazon EKS y cree los recursos de IAM necesarios para permitir que una cuenta de servicio de Amazon EKS se conecte a la tabla de Amazon Keyspaces

1. Cree un proveedor de Open ID Connect (OIDC) para el clúster de Amazon EKS. Esto es necesario para utilizar los roles 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. Este ejemplo asume que el nombre del 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 identidades de OIDC se haya registrado en IAM con el siguiente comando.

```
aws iam list-open-id-connect-providers --region us-east-1
```

La salida debería ser similar a la siguiente. Tome nota del nombre de recurso de Amazon (ARN) del OIDC. Lo necesitará en el siguiente paso al crear una política de confianza para la cuenta de servicio.

```
{
  "OpenIDConnectProviderList": [
    ..
    {
      "Arn": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.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 puede utilizar para implementar una aplicación en

contenedores. A continuación, cree un rol de IAM que la cuenta de servicio pueda asumir para obtener permisos de acceso a los recursos. Puede 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. Cree 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 con 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` en el espacio de nombres `my-eks-namespace` para su clúster 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 una política de confianza que indique al rol de IAM que debe confiar en su cuenta de servicio. Esta relación de confianza es necesaria antes de que una entidad principal pueda asumir un rol. Debe realizar las siguientes modificaciones en el archivo:

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

- En la `condition` declaración, 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 sean correctos.

Debe asociar 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.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount",
          "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
EOF
```

Opcional: también puede agregar varias entradas en las condiciones `StringEquals` y `StringLike` para permitir que varias cuentas de servicio o espacios de nombres asuman el rol. Para permitir que su cuenta de servicio asuma una función de IAM en otra AWS cuenta, consulte los [permisos de Cross-account IAM](#) en la Guía del usuario de Amazon EKS.

3. Cree un rol de IAM con el nombre `my-iam-role` para que lo asuma la cuenta de servicio de Amazon EKS. Asocie el archivo de política de confianza creado en el último paso al rol. La

política de confianza específica la cuenta de servicio y el proveedor de OIDC en los que puede confiar el rol 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. Asigne al rol de IAM permisos para Amazon Keyspaces mediante la asociación de una política de acceso.
 - a. Asocie una política de acceso para definir las acciones que el rol 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 haya asociado correctamente al rol de IAM con la siguiente instrucción.

```
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 en la cuenta de servicio el nombre de recurso de Amazon (ARN) del rol 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 se hayan configurado correctamente.
 - a. Confirme que la política de confianza del rol de IAM se haya configurado correctamente con la siguiente instrucción.

```
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.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
          "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:my-eks-
namespace:my-eks-serviceaccount"
        }
      }
    }
  ]
}
```

- b. Confirme que en la cuenta de servicio de Amazon EKS se haya anotado 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 de Amazon EKS y el rol de IAM, y tras configurar las relaciones y los permisos necesarios, continúe con [the section called “Paso 2: configuración de la aplicación”](#).

Paso 2: configuración de la aplicación

En este paso, creará una aplicación que se conectará a Amazon Keyspaces mediante el complemento SigV4. Puede ver y descargar la aplicación Java de ejemplo desde el repositorio de código de ejemplo de Amazon Keyspaces en [Github](#). O también puede continuar con su propia aplicación, asegurándose de completar todos los pasos de configuración.

Configuración de su aplicación y adición de las dependencias necesarias.

1. Puede 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, descomprima el archivo descargado y navegue hasta el directorio `resources` del archivo `application.conf`.

- a. Configuración de aplicaciones

En este paso, configurará el complemento de autenticación SigV4. Puede utilizar 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 necesarias para la autenticación AWS”](#). Actualice la AWS región y el punto de enlace del servicio para Amazon Keyspaces según sea necesario. Para obtener información acerca de más puntos de conexión del servicio,

consulte [the section called “Puntos de conexión de servicio”](#). Sustituya la ubicación, el nombre y la contraseña del almacén de confianza por los suyos propios.

```
datastax-java-driver {
  basic.contact-points = ["cassandra.us-east-1.amazonaws.com:9142"]
  basic.load-balancing-policy.local-datacenter = "us-east-1"
  advanced.auth-provider {
    class = software.aws.mcs.auth.SigV4AuthProvider
    aws-region = "us-east-1"
  }
  advanced.ssl-engine-factory {
    class = DefaultSslEngineFactory
    truststore-path = "truststore_locationtruststore_name.jks"
    truststore-password = "truststore_password;"
  }
}
```

- b. 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. Para hacerlo, siga este 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 la dependencia de registro.

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

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

Puede utilizar 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" />
  </root>
</configuration>
```

Note

El nivel de `debug` 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: creación de la imagen de la aplicación y subida del archivo de Docker a su repositorio de Amazon ECR

En este paso, compilará la aplicación de ejemplo, creará una imagen de Docker y enviará la imagen a su repositorio de Amazon ECR.

Desarrollo de su aplicación, creación de una imagen de Docker y envío a Amazon Elastic Container Registry

1. Establezca las variables de entorno para la compilación que se definan en su Región de AWS. En los ejemplos, sustituya las regiones por las suyas propias.

```
export CASSANDRA_HOST=cassandra.us-east-1.amazonaws.com:9142
export CASSANDRA_DC=us-east-1
```

2. Compile una aplicación con Apache Maven versión 3.6.3 o superior mediante el siguiente comando.

```
mvn clean install
```

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

3. Recupere el URI del repositorio de ECR que necesita para el próximo paso con el siguiente comando. Asegúrese de actualizar la región y de poner la que ha estado utilizando.

```
aws ecr describe-repositories --region us-east-1
```

El resultado debe tener el aspecto del siguiente ejemplo.

```
"repositories": [
  {
    "repositoryArn": "arn:aws:ecr:us-east-1:111122223333:repository/my-ecr-repository",
    "registryId": "111122223333",
    "repositoryName": "my-ecr-repository",
    "repositoryUri": "111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository",
    "createdAt": "2023-11-02T03:46:34+00:00",
    "imageTagMutability": "MUTABLE",
    "imageScanningConfiguration": {
      "scanOnPush": false
    }
  }
]
```

```
},  
"encryptionConfiguration": {  
  "encryptionType": "AES256"  
}  
},
```

- Desde el directorio raíz de la aplicación, cree la imagen de Docker utilizando el URI del repositorio del último paso. Modifique el archivo de 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.us-east-1.amazonaws.com/my-ecr-repository:latest .
```

- 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 us-east-1 | docker login --username AWS --password-stdin 111122223333.dkr.ecr.us-east-1.amazonaws.com
```

- En primer lugar, compruebe si hay imágenes existentes en su repositorio de Amazon ECR. Puede utilizar el siguiente comando.

```
aws ecr describe-images --repository-name my-ecr-repository --region us-east-1
```

Después, envíe la imagen de Docker al repositorio. Puede utilizar el siguiente comando.

```
docker push 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository:latest
```

Paso 4: implementación de la aplicación en Amazon EKS y escritura de los datos en la tabla

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, utilizaremos `my-keyspaces-app`.
- Espacio de nombres de Kubernetes: el espacio de nombres del clúster de Amazon EKS. Para este tutorial, utilizaremos `my-eks-namespace`.
- Nombre de cuenta de servicio de Amazon EKS: el nombre de la cuenta de servicio de Amazon EKS. Para este tutorial, utilizaremos `my-eks-serviceaccount`.
- Nombre de imagen: el nombre de la imagen de la aplicación. Para este tutorial, utilizaremos `my-keyspaces-app`.
- URI de imagen: el URI de imagen de Docker de Amazon ECR.
- AWS ID de cuenta: su ID AWS de cuenta.
- ARN del rol de IAM: el ARN del rol de IAM creado para que lo asuma la cuenta de servicio. Para este tutorial, utilizaremos `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, implementará y ejecutará la aplicación que se conecta a Amazon Keyspaces y que escribe datos en la tabla.

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

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

Puede utilizar 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.us-east-1.amazonaws.com/my-ecr-repository:latest
          ports:
            - containerPort: 8080
          env:
            - name: CASSANDRA_HOST
              value: "cassandra.us-east-1.amazonaws.com:9142"
            - name: CASSANDRA_DC
              value: "us-east-1"
            - name: AWS_WEB_IDENTITY_TOKEN_FILE
              value: /var/run/secrets/eks.amazonaws.com/serviceaccount/token
            - name: AWS_ROLE_ARN
              value: "arn:aws:iam::111122223333:role/my-iam-role"
            - name: AWS_REGION
              value: "us-east-1"
```

2. Implemente 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 detalles, puede usar el comando siguiente.

```
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://41ff7811d33ae4bc398755800abdc132335d51d74f218ba81da0700a6f8c67b
    Image:                            111122223333.dkr.ecr.us-east-1.amazonaws.com/
    my_eks_repository:latest
    Image ID:                         111122223333.dkr.ecr.us-east-1.amazonaws.com/
    my_eks_repository@sha256:fd3c6430fc5251661efce99741c72c1b4b03061474940200d0524b84a951439c
    Port:                             8080/TCP
    Host Port:                        0/TCP
    State:                            Running
      Started:                        Thu, 23 Nov 2023 12:15:19 +0000
      Finished:                       Thu, 23 Nov 2023 12:16:17 +0000
    Ready:                            True
    Restart Count:                    1
```

```

Environment:
  CASSANDRA_HOST:      cassandra.us-east-1.amazonaws.com:9142
  CASSANDRA_DC:        us-east-1
  AWS_WEB_IDENTITY_TOKEN_FILE: /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
  AWS_ROLE_ARN:         arn:aws:iam::111122223333:role/my-iam-role
  AWS_REGION:          us-east-1
  AWS_STS_REGIONAL_ENDPOINTS: regional
Mounts:
  /var/run/secrets/eks.amazonaws.com/serviceaccount from aws-iam-token (ro)
  /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-fssbf (ro)
Conditions:
  Type              Status
  Initialized        True
  Ready              True
  ContainersReady    True
  PodScheduled       True
Volumes:
  aws-iam-token:
    Type:              Projected (a volume that contains injected data from
multiple sources)
    TokenExpirationSeconds: 86400
  kube-api-access-fssbf:
    Type:              Projected (a volume that contains injected data from
multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:      kube-root-ca.crt
    ConfigMapOptional:  <nil>
    DownwardAPI:        true
QoS Class:           BestEffort
Node-Selectors:      <none>
Tolerations:         node.kubernetes.io/not-ready:NoExecute op=Exists for
300s
                     node.kubernetes.io/unreachable:NoExecute op=Exists for
300s
Events:
  Type    Reason             Age          From              Message
  ----    -
Warning  LoggingDisabled    2m13s        fargate-scheduler Disabled logging
because aws-logging configmap was not found. configmap "aws-logging" not found
Normal   Scheduled          89s          fargate-scheduler Successfully
assigned my-eks-namespace/my-keyspaces-app-6c56fccc56-mgs2m to fargate-
ip-192-168-102-209.ec2.internal

```

```

Normal    Pulled          75s          kubelet
Successfully pulled image "111122223333.dkr.ecr.us-east-1.amazonaws.com/
my_eks_repository:latest" in 13.027s (13.027s including waiting)
Normal    Pulling             54s (x2 over 88s) kubelet          Pulling image
"111122223333.dkr.ecr.us-east-1.amazonaws.com/my_eks_repository:latest"
Normal    Created            54s (x2 over 75s) kubelet          Created container
my-keyspaces-app
Normal    Pulled             54s          kubelet
Successfully pulled image "111122223333.dkr.ecr.us-east-1.amazonaws.com/
my_eks_repository:latest" in 222ms (222ms including waiting)
Normal    Started            53s (x2 over 75s) kubelet          Started container
my-keyspaces-app

```

- Compruebe los registros del pod para confirmar que la aplicación se esté ejecutando y que se pueda conectar a la tabla de Amazon Keyspaces. Puede hacerlo con el siguiente comando. Asegúrese de reemplazar el nombre de su implementación.

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

Debería 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.us-
east-1.amazonaws.com/1.222.333.44:9142, hostId=null, hashCode=e750d92)]
22:47:20.562 [s0-admin-1] DEBUG c.d.o.d.i.c.c.ControlConnection - [s0] Initializing
with event types [SCHEMA_CHANGE, STATUS_CHANGE, TOPOLOGY_CHANGE]
22:47:20.564 [s0-admin-1] DEBUG c.d.o.d.i.core.context.EventBus - [s0] Registering
com.datastax.oss.driver.internal.core.metadata.LoadBalancingPolicyWrapper$$Lambda
$812/0x00000000801105e88@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)

```

- 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: (opcional) eliminación

Siga estos pasos para eliminar todos los recursos que ha creado en este tutorial.

Eliminación de los recursos creados en este tutorial

1. Elimine su implementación. Puede utilizar el siguiente comando para hacerlo.

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

2. Elimine 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 identidades de OIDC. Puede utilizar el siguiente comando para hacerlo.

```
eksctl delete cluster --name my-eks-cluster --region us-east-1
```

3. Elimine el rol de IAM utilizado para la cuenta de servicio de Amazon EKS con permisos de acceso a Amazon Keyspaces. En primer lugar, debe eliminar la política administrada asociada al rol.

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

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

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

Para obtener más información, consulte [Eliminación de 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 con el siguiente comando.

```
aws ecr delete-repository \
```

```
--repository-name my-ecr-repository \  
--force \  
--region us-east-1
```

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

```
aws ecr batch-delete-image \  
  --repository-name my-ecr-repository \  
  --image-ids imageTag=latest \  
  --region us-east-1
```

Para obtener más información, consulte [Eliminación de una imagen](#) en la Guía del usuario de Amazon Elastic Container Registry.

5. Elimine 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. Puede utilizar una de las siguientes opciones para ello.

AWS CLI

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

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

```
aws keyspaces list-keyspaces
```

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

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

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

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

Para obtener más información, consulte [eliminación de un espacio de claves](#) y [eliminación de una tabla](#) en la Referencia de comandos de la AWS CLI .

cqlsh

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

Para verificar que sus espacios de claves se hayan eliminado, puede utilizar la siguiente instrucción.

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

Su espacio de claves no debería aparecer en el resultado de esta instrucción. Tenga en cuenta que puede haber un retraso hasta que se eliminen los espacios de claves. Para obtener más información, consulte [the section called “DROP KEYSPACE”](#).

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

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

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

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

Su tabla no debería aparecer en el resultado de esta instrucción. 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: Exportación de una tabla de Amazon Keyspaces a Amazon S3 mediante AWS Glue

En este tutorial se muestra cómo exportar una tabla de Amazon Keyspaces a un bucket de Amazon S3 utilizando AWS Glue. Para este tutorial, muchos pasos manuales se automatizan mediante scripts de shell disponibles en el [repositorio Github](#) de Amazon Keyspaces. Con este proceso, puede exportar los datos de Amazon Keyspaces a Amazon S3 sin tener que configurar un clúster de Spark.

Temas

- [Requisitos previos para exportar datos de Amazon Keyspaces a Amazon S3](#)

- [Paso 1: Crear el bucket de Amazon S3, descargar las herramientas necesarias y configurar el entorno](#)
- [Paso 2: Configurar el AWS Glue trabajo que exporta la tabla Amazon Keyspaces](#)
- [Paso 3: Ejecute el AWS Glue trabajo para exportar la tabla de Amazon Keyspaces al bucket de Amazon S3 desde AWS CLI](#)
- [Paso 4: \(opcional\) Cree un activador para programar el trabajo de exportación](#)
- [Paso 5: \(opcional\) eliminación](#)

Requisitos previos para exportar datos de Amazon Keyspaces a Amazon S3

Confirme los siguientes requisitos previos y cree los recursos de Amazon Keyspaces antes de empezar con el tutorial

1. Antes de comenzar este tutorial, siga las instrucciones de AWS configuración que se indican 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.
2. Los scripts de este tutorial utilizan sus credenciales y Región de AWS se almacenan de forma predeterminada en una ubicación conocida. Para obtener más información, consulte [the section called "Administración de claves de acceso"](#).

El siguiente ejemplo muestra cómo almacenar los valores necesarios como variables de entorno para el usuario predeterminado.

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-east-1
```

3. Para ejecutar los scripts de este tutorial, necesitará tener el software y las herramientas siguientes instalados en su máquina:
 - [Java](#)
 - [Apache Maven](#)
 - [Git](#)
 - [AWS CLI](#)

Este tutorial se probó con AWS CLI 2, Java 17.0.13 y Apache Maven 3.8.7.

4. Necesitarás una tabla de Amazon Keyspaces con datos de ejemplo para exportarla más adelante en este tutorial. Puedes usar tu propia tabla de Amazon Keyspaces o crear una tabla de ejemplo siguiendo los pasos del tutorial. [Introducción](#)
 - a. Para instalar `cqlsh-expansion`, siga los pasos que se indican en [the section called “Uso de cqlsh-expansion”](#).
 - b. Confirme que el particionador `Murmur3Partitioner` es el particionador predeterminado de su cuenta. Este particionador es compatible con el conector Apache Spark Cassandra y con AWS Glue. Para obtener más información sobre los particionadores, consulte [the section called “Uso de particionadores”](#).

Para cambiar el particionador de su cuenta, puede utilizar la siguiente declaración.

```
SELECT partitioner FROM system.local;  
  
UPDATE system.local set  
partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

- c. Para crear un espacio de claves de Amazon Keyspaces, sigue los pasos que se indican en [the section called “Creación de un espacio de claves”](#).
 - d. Para crear la tabla Amazon Keyspaces, sigue los pasos que se indican en [the section called “Creación de una tabla”](#).
 - e. Para cargar datos de muestra en la tabla para exportarlos a Amazon S3, siga los pasos que se indican en [the section called “Crear”](#).

Tras completar los pasos de requisitos previos, continúe con [the section called “Paso 1: Crear el bucket de Amazon S3, descargar las herramientas y configurar el entorno”](#).

Paso 1: Crear el bucket de Amazon S3, descargar las herramientas necesarias y configurar el entorno

En este paso, descarga las herramientas externas y crea y configura AWS los recursos necesarios para la solución de exportación automática de datos de una tabla de Amazon Keyspaces a un bucket de Amazon S3 mediante un AWS Glue trabajo. Para realizar todas estas tareas de forma eficiente, ejecutamos un script de shell con el nombre `setup-connector.sh` disponible en [Github](#).

El script `setup-connector.sh` automatiza los siguientes pasos.

1. Crea un bucket de Amazon S3 mediante CloudFormation. Este depósito almacena los archivos jar y de configuración descargados, así como los datos de la tabla exportados.
2. Crea un rol de IAM mediante CloudFormation. AWS Glue los trabajos utilizan este rol para acceder a Amazon Keyspaces y Amazon S3.
3. Descarga el [conector Cassandra de Apache Spark](#) y lo carga en el bucket de Amazon S3.
4. Descarga el [complemento de autenticación SigV4](#) y lo carga en el bucket de Amazon S3.
5. Descarga las [extensiones de Apache Spark](#) y las carga en el bucket de Amazon S3.
6. Descarga la [política de reintentos de Keyspaces](#) de Github, compila el código con Maven y carga el resultado en el bucket de Amazon S3.
7. Carga el **keyspaces-application.conf** archivo en el bucket de Amazon S3.

Utilice el script de **setup-connector.sh** shell para automatizar los pasos de instalación y configuración.

1. Copie los archivos del repositorio [aws-glue](#) de Github a su máquina local. Este directorio contiene el script de shell y otros archivos necesarios.
2. Ejecute el script de shell `setup-connector.sh`. Puede especificar los tres parámetros opcionales siguientes.
 1. **SETUP_STACKNAME**— Este es el nombre de la CloudFormation pila utilizada para crear los AWS recursos.
 2. **S3_BUCKET_NAME**— Este es el nombre del bucket de Amazon S3.
 3. **GLUE_SERVICE_ROLE_NAME**— Este es el nombre del rol de servicio de IAM que se AWS Glue utiliza para ejecutar los trabajos que se conectan a Amazon Keyspaces y Amazon S3.

Puede utilizar el siguiente comando para ejecutar el script de shell y proporcionar los tres parámetros con los siguientes nombres.

```
./setup-connector.sh cfn-setup s3-keyspaces iam-export-role
```

Para confirmar que se creó el bucket, puede usar el siguiente AWS CLI comando.

```
aws s3 ls s3://s3-keyspaces
```

El resultado del comando debería tener este aspecto.

```
PRE conf/  
PRE jars/
```

Para confirmar que se creó el rol de IAM y revisar los detalles, puede utilizar la siguiente AWS CLI declaración.

```
aws iam get-role --role-name "iam-export-role"
```

```
{  
  "Role": {  
    "Path": "/",  
    "RoleName": "iam-export-role",  
    "RoleId": "AKIAIOSFODNN7EXAMPLE",  
    "Arn": "arn:aws:iam::111122223333:role/iam-export-role",  
    "CreateDate": "2025-01-28T16:09:03+00:00",  
    "AssumeRolePolicyDocument": {  
      "Version": "2012-10-17",  
      "Statement": [  
        {  
          "Effect": "Allow",  
          "Principal": {  
            "Service": "glue.amazonaws.com"  
          },  
          "Action": "sts:AssumeRole"  
        }  
      ]  
    },  
    "Description": "AWS Glue service role to import and export data from Amazon  
Keyspaces",  
    "MaxSessionDuration": 3600,  
    "RoleLastUsed": {  
      "LastUsedDate": "2025-01-29T12:03:54+00:00",  
      "Region": "us-east-1"  
    }  
  }  
}
```

Si se produce un error en el proceso de CloudFormation pila, puede revisar la información detallada sobre el error de la pila que ha fallado en la CloudFormation consola.

Una vez que se haya creado el bucket de Amazon S3 que contiene todos los scripts y herramientas y se haya configurado la función de IAM, continúe [the section called “Paso 2: Configurar el trabajo AWS Glue ”](#) con.

Paso 2: Configurar el AWS Glue trabajo que exporta la tabla Amazon Keyspaces

En el segundo paso del tutorial, utiliza el script `setup-export.sh` disponible en [Github](#) para crear y configurar el AWS Glue trabajo que se conecta a Amazon Keyspaces mediante el complemento SigV4 y, a continuación, exporta la tabla especificada a su bucket de Amazon S3 creado en el paso anterior. El uso del script le permite exportar datos de Amazon Keyspaces sin necesidad de configurar un clúster de Apache Spark.

Cree un AWS Glue trabajo para exportar una tabla de Amazon Keyspaces a un bucket de Amazon S3.

- En este paso, ejecuta el script de `setup-export.sh` shell ubicado en el `export-to-s3/` directorio que se usará CloudFormation para crear y configurar el trabajo de AWS Glue exportación. El script toma los siguientes parámetros.

```
PARENT_STACK_NAME, EXPORT_STACK_NAME, KEYSPACE_NAME, TABLE_NAME, S3_URI, FORMAT
```

- **PARENT_STACK_NAME**— El nombre de la CloudFormation pila creada en el paso anterior.
- **EXPORT_STACK_NAME**— El nombre de la CloudFormation pila que crea el trabajo de AWS Glue exportación.
- **KEYSPACE_NAME** y **TABLE_NAME** — El nombre completo del espacio de claves y la tabla que se van a exportar. Para este tutorial, lo utilizamos `catalog.book_awards`, pero puede sustituirlo por su propio nombre de tabla completo.
- **S3URI**— El URI opcional del bucket de Amazon S3. El valor predeterminado es el bucket de Amazon S3 de la pila principal.
- **FORMAT**— El formato de datos opcional. El valor predeterminado es `parquet`. En este tutorial, para facilitar la carga y transformación de los datos, utilizamos el predeterminado.

Puede usar el siguiente comando de la como ejemplo.


```
setup-export.sh cfn-setup cfn-glue catalog book_awards
```

Para confirmar que el trabajo se ha creado, puede utilizar la siguiente declaración.

```
aws glue list-jobs
```

La salida de la instrucción debe ser similar a esta.

```
{
  "JobNames": [
    "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue"
  ]
}
```

Para ver los detalles del trabajo, puede usar el siguiente comando.

```
aws glue get-job --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue
```

El resultado del comando muestra todos los detalles del trabajo. Esto incluye los argumentos predeterminados que puede anular al ejecutar el trabajo.

```
{
  "Job": {
    "Name": "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue",
    "JobMode": "SCRIPT",
    "JobRunQueuingEnabled": false,
    "Description": "export to s3",
    "Role": "iam-export-role",
    "CreatedOn": "2025-01-30T15:53:30.765000+00:00",
    "LastModifiedOn": "2025-01-30T15:53:30.765000+00:00",
    "ExecutionProperty": {
      "MaxConcurrentRuns": 1
    },
    "Command": {
      "Name": "glueetl",
      "ScriptLocation": "s3://s3-keyspaces/scripts/cfn-setup-cfn-glue-export.scala",
      "PythonVersion": "3"
    },
  },
}
```

```

    "DefaultArguments": {
      "--write-shuffle-spills-to-s3": "true",
      "--S3_URI": "s3://s3-keyspaces",
      "--TempDir": "s3://s3-keyspaces/shuffle-space/export-sample/",
      "--extra-jars": "s3://s3-keyspaces/jars/spark-cassandra-
connector-assembly_2.12-3.1.0.jar,s3://s3-keyspaces/jars/aws-sigv4-auth-
cassandra-java-driver-plugin-4.0.9-shaded.jar,s3://s3-keyspaces/jars/spark-
extension_2.12-2.8.0-3.4.jar,s3://s3-keyspaces/jars/amazon-keyspaces-helpers-1.0-
SNAPSHOT.jar",
      "--class": "GlueApp",
      "--user-jars-first": "true",
      "--enable-metrics": "true",
      "--enable-spark-ui": "true",
      "--KEYSPACE_NAME": "catalog",
      "--spark-event-logs-path": "s3://s3-keyspaces/spark-logs/",
      "--enable-continuous-cloudwatch-log": "true",
      "--write-shuffle-files-to-s3": "true",
      "--FORMAT": "parquet",
      "--TABLE_NAME": "book_awards",
      "--job-language": "scala",
      "--extra-files": "s3://s3-keyspaces/conf/keyspaces-application.conf",
      "--DRIVER_CONF": "keyspaces-application.conf"
    },
    "MaxRetries": 0,
    "AllocatedCapacity": 4,
    "Timeout": 2880,
    "MaxCapacity": 4.0,
    "WorkerType": "G.2X",
    "NumberOfWorkers": 2,
    "GlueVersion": "3.0"
  }
}

```

Si el proceso de CloudFormation apilado falla, puedes revisar los errores de la pila fallida en la CloudFormation consola. Puede revisar los detalles del trabajo de exportación en la AWS Glue consola seleccionando los trabajos de ETL en el menú de la izquierda.

Una vez que hayas confirmado los detalles del trabajo de AWS Glue exportación, procede [the section called “Paso 3: Ejecute el AWS Glue trabajo de exportación desde AWS CLI”](#) a ejecutar el trabajo para exportar los datos de tu tabla de Amazon Keyspaces.

Paso 3: Ejecute el AWS Glue trabajo para exportar la tabla de Amazon Keyspaces al bucket de Amazon S3 desde AWS CLI

En este paso, se utiliza AWS CLI para ejecutar el AWS Glue trabajo creado en el paso anterior para exportar una tabla de Amazon Keyspaces a su bucket en Amazon S3.

Ejecute el trabajo de exportación desde AWS CLI

1. En el siguiente ejemplo, el AWS CLI comando ejecuta el trabajo creado en el paso anterior.

```
aws glue start-job-run --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue
```

- Puede anular cualquiera de los parámetros del AWS Glue trabajo, incluidos los argumentos predeterminados del AWS CLI comando. Para anular cualquier argumento predeterminado del trabajo, por ejemplo, el espacio de claves o el nombre de la tabla, puede pasarlos como argumentos. Para obtener una lista completa de argumentos, consulte [start-job-run](#) en la Referencia de la línea de comandos. AWS Glue

El siguiente comando ejecuta el trabajo de AWS Glue exportación, pero anula el número de AWS Glue trabajadores, el tipo de trabajador y el nombre de la tabla.

```
aws glue start-job-run --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue \
    --number-of-workers 8 --worker-type G.2X \
    --arguments '{"--TABLE_NAME": "my_table"}'
```

2. Confirma que la tabla se ha exportado a tu bucket de Amazon S3. Según el tamaño de la mesa, esto puede llevar algún tiempo. Cuando finalice el trabajo de exportación, podrá ver las siguientes carpetas en el depósito mediante el comando de ejemplo.

```
aws s3 ls s3://s3-keyspaces
```

El resultado muestra la siguiente estructura en el depósito.

```
PRE conf/
PRE export/
PRE jars/
PRE scripts/
PRE spark-logs/
```

Sus archivos se ubicarán en la siguiente estructura de carpetas `export`, en la que `data/time` los valores mostrarán sus propios valores.

```
\----- export
      \----- keyspace_name
            \----- table_name
                  \----- snapshot
                        \----- year=2025
                              \----- month=01
                                    \----- day=02
                                          \----- hour=09
                                                \----- minute=22
                                                    \--- YOUR DATA HERE
```

Para programar el AWS Glue trabajo que acaba de ejecutar manualmente, continúe con [the section called “Paso 4: \(Opcional\) Programe el trabajo de exportación”](#).

Paso 4: (opcional) Cree un activador para programar el trabajo de exportación

Para ejecutar el trabajo de exportación creado en el paso anterior de forma regular, puede crear un activador programado. Para obtener más información, consulte los [AWS Glue activadores](#) en la Guía para AWS Glue desarrolladores.

Programe un AWS Glue trabajo

1. El siguiente AWS CLI comando es un ejemplo de un activador sencillo con un nombre `KeyspacesExportWeeklyTrigger` que ejecuta la AWS Glue tarea con ese nombre `AmazonKeyspacesExportToS3-cfn-setup-cfn-glue` una vez por semana los lunes a las 12:00 UTC.

```
aws glue create-trigger \
  --name KeyspacesExportWeeklyTrigger \
  --type SCHEDULED \
  --schedule "cron(0 12 ? * MON *)" \
  --start-on-creation \
  --actions '[{
    "JobName": "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue"
```

```
}}'
```

- Para anular cualquiera de los ajustes predeterminados del trabajo programado, puede pasarlos como argumentos. En este ejemplo, sustituimos el nombre del espacio de claves, el nombre de la tabla, el número de trabajadores y el tipo de trabajador pasándolos como argumentos. El siguiente comando es un ejemplo de esto.

```
aws glue create-trigger \  
  --name KeyspacesExportWeeklyTrigger \  
  --type SCHEDULED \  
  --schedule "cron(0 12 ? * MON *)" \  
  --start-on-creation \  
  --actions '[{  
    "JobName": "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue",  
    "Arguments": {  
      "--number-of-workers": "8",  
      "--worker-type": "G.2X"},  
      "--table_name": "my_table",  
      "--keyspace_name": "my_keyspace"  
    }  
  ]}'
```

2. Para confirmar que se ha creado el desencadenador, utilice el siguiente comando.

```
aws glue list-triggers
```

El resultado del comando debería tener un aspecto similar al siguiente.

```
{  
  "TriggerNames": [  
    "KeyspacesExportWeeklyTrigger"  
  ]  
}
```

Para limpiar los AWS recursos creados en este tutorial, continúe con [the section called “Paso 5: \(opcional\) eliminación”](#).

Paso 5: (opcional) eliminación

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

Eliminación de los recursos creados en este tutorial

1. Elimine la segunda CloudFormation pila creada en este tutorial. Esto elimina la AWS Glue tarea y el activador creados en este tutorial. Puede usar el siguiente comando de la para hacerlo.

```
aws cloudformation delete-stack --stack-name cfn-glue
```

2. Elimine el bucket de Amazon S3 junto con todos los datos almacenados en él. Puede utilizar el siguiente comando para hacerlo.

```
aws s3 rm s3://s3-keyspaces --recursive
```

3. Elimine la primera pila creada en este tutorial. De este modo, se eliminan el rol de IAM y los permisos asociados creados en este tutorial. Puede usar el siguiente comando de la como ejemplo.

```
aws cloudformation delete-stack --stack-name cfn-setup
```

4. Elimine 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. Puede utilizar una de las siguientes opciones para ello.

AWS CLI

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

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

```
aws keyspaces list-keyspaces
```

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

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

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

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

Para obtener más información, consulte [eliminación de un espacio de claves](#) y [eliminación de una tabla](#) en la Referencia de comandos de la AWS CLI .

cqlsh

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

Para verificar que sus espacios de claves se hayan eliminado, puede utilizar la siguiente instrucción.

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

Su espacio de claves no debería aparecer en el resultado de esta instrucción. Tenga en cuenta que puede haber un retraso hasta que se eliminen los espacios de claves. Para obtener más información, consulte [the section called “DROP KEYSPACE”](#).

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

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

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

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

Su tabla no debería aparecer en el resultado de esta instrucción. Tenga en cuenta que puede haber un retraso hasta que se elimine la tabla. Para obtener más información, consulte [the section called “DROP TABLE”](#).

Administración de recursos sin servidor en Amazon Keyspaces (para Apache Cassandra)

Amazon Keyspaces (para Apache Cassandra) opera sin servidor. En lugar de implementar, administrar y mantener los recursos de almacenamiento y procesamiento para su carga de trabajo a través de los nodos de un clúster, Amazon Keyspaces asigna los recursos de almacenamiento y read/write rendimiento directamente a las tablas.

Amazon Keyspaces aprovisiona almacenamiento de forma automática en función de los datos reales almacenados en las tablas. Aumenta y reduce el almacenamiento a medida que se escriben, actualizan y eliminan datos, y solo se paga por el almacenamiento utilizado. Los datos se replican en varias [zonas de disponibilidad](#) para lograr una alta disponibilidad. 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 “Estimación del tamaño de las filas”](#).

En este capítulo se abordan aspectos clave de la administración de recursos en Amazon Keyspaces.

- Estimación del tamaño de las filas: para estimar el tamaño codificado de las filas en Amazon Keyspaces, tenga en cuenta factores como los metadatos de las claves de partición, los metadatos de las columnas agrupadas en clústeres, los identificadores de las columnas, los tipos de datos y los metadatos de las filas. Este tamaño de fila codificado se utiliza para la facturación, la administración de las cuotas y la planificación de la capacidad de rendimiento aprovisionado.
- Estimación del consumo de capacidad: en esta sección se describen ejemplos de cómo estimar el consumo de capacidad de lectura y escritura en escenarios comunes, como las consultas de rango, las consultas de límites, los análisis de tablas, las transacciones simplificadas, las columnas estáticas y las tablas multirregionales. Puedes usar Amazon CloudWatch para monitorear la utilización real de la capacidad. Para obtener más información sobre la supervisión con CloudWatch, consulte [the section called “Monitorización con CloudWatch”](#).
- Configure los modos de read/write capacidad: puede elegir entre dos modos de capacidad para procesar las lecturas y escrituras en las tablas:
 - On-demand modo (predeterminado): paga por solicitud en función del rendimiento de lectura y escritura. Amazon Keyspaces puede escalar la capacidad de forma instantánea hasta cualquier nivel de tráfico alcanzado previamente.

- **Modo aprovisionado:** se especifica por adelantado el número necesario de unidades de capacidad de lectura y escritura. Este modo ayuda a mantener un rendimiento del procesamiento predecible.
- **Administración de la capacidad de rendimiento con el escalado automático:** en el caso de las tablas aprovisionadas, puede habilitar el escalado automático para ajustar la capacidad de rendimiento automáticamente en función del tráfico real de las aplicaciones. Amazon Keyspaces utiliza el seguimiento de objetivos para aumentar o disminuir la capacidad aprovisionada, manteniendo el uso en el objetivo especificado.
- **Uso de la capacidad de ampliación de forma eficaz:** Amazon Keyspaces proporciona capacidad de ampliación mediante la reserva de una parte del rendimiento no utilizado para administrar los picos de tráfico. Esta flexibilidad permite que se produzcan incrementos ocasionales de actividad que superen el rendimiento aprovisionado.

Para solucionar errores de capacidad, consulte [the section called “Errores de capacidad sin servidor”](#).

Temas

- [Estimación del tamaño de las filas en Amazon Keyspaces](#)
- [Estimación del consumo de capacidad del rendimiento de lectura y escritura en Amazon Keyspaces](#)
- [Configurar los modos de read/write capacidad en Amazon Keyspaces](#)
- [Administración automática de la capacidad de rendimiento con el escalado automático de Amazon Keyspaces](#)
- [Uso eficaz de la capacidad de ampliación en Amazon Keyspaces](#)

Estimación del tamaño de las filas en Amazon Keyspaces

Amazon Keyspaces proporciona un almacenamiento totalmente gestionado que ofrece un rendimiento de lectura y escritura en milisegundos de un solo dígito y almacena los datos de forma duradera en varias zonas de disponibilidad. 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 este tema se proporciona información sobre cómo calcular el tamaño codificado de las filas en Amazon Keyspaces. El tamaño codificado de fila se utiliza al calcular su factura y el uso de cuotas. También puede usar el tamaño de fila codificado al estimar los requisitos de capacidad de rendimiento aprovisionados para las tablas.

Para calcular el tamaño codificado de fila en Amazon Keyspaces, puede utilizar las siguientes pautas.

Temas

- [Calcule el tamaño codificado de las columnas](#)
- [Calcule el tamaño codificado de los valores de datos en función del tipo de datos](#)
- [Tenga en cuenta el impacto de las funciones de Amazon Keyspaces en el tamaño de las filas](#)
- [Elija la fórmula correcta para calcular el tamaño codificado de una fila](#)
- [Ejemplo de cálculo del tamaño de fila](#)

Calcule el tamaño codificado de las columnas

En esta sección se muestra cómo estimar el tamaño codificado de las columnas en Amazon Keyspaces.

- Columnas normales: en el caso de las columnas normales, que son columnas que no son claves principales, columnas agrupadas o STATIC columnas, utilice el tamaño original de los datos de las celdas en función del [tipo de datos](#) y añada los metadatos necesarios. Los tipos de datos y algunas diferencias clave en la forma en que Amazon Keyspaces almacena los valores de los tipos de datos y los metadatos se detallan en la siguiente sección.
- Columnas de claves de partición: 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.
- Columnas de agrupamiento: las columnas de agrupamiento 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.

Note

Para facilitar las consultas y la indexación integrada, Amazon Keyspaces almacena dos veces el valor de los datos de cada clave de partición y columna de clave de agrupamiento.

- Nombres de columna: el espacio necesario para cada nombre de columna se almacena mediante 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”](#).

Calcule el tamaño codificado de los valores de datos en función del tipo de datos

En esta sección se muestra cómo estimar el tamaño codificado de distintos tipos de datos en Amazon Keyspaces.

- Tipos de cadenas: Cassandra ASCII y los tipos de datos de VARCHAR cadenas se almacenan en Amazon Keyspaces mediante Unicode con codificación binaria UTF-8 . TEXT El tamaño de una cadena en Amazon Keyspaces es igual al número de bytes UTF-8 codificados.
- Tipos numéricos: Cassandra INT, BIGINT, SMALLINT, TINYINT, y VARINT los tipos de datos se almacenan en Amazon Keyspaces como valores de datos con 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.
- Tipo de blob: A BLOB en Amazon Keyspaces se almacena con la longitud bruta en bytes del valor.
- Tipo booleano: el tamaño de un Boolean valor o un Null valor es de 1 byte.
- Tipos de recopilación: columna que almacena tipos de datos de recopilación, como metadatos LIST o que MAP requiere 3 bytes, 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.

- User-defined tipos: un [tipo definido por el usuario \(UDT\)](#) requiere 3 bytes para los metadatos, independientemente de su contenido. Para cada elemento UDT, Amazon Keyspaces requiere 1 byte adicional de metadatos.

Para calcular el tamaño codificado de una UDT, comience con los `field name` y `field value` para los campos de una UDT:

- nombre de campo: cada nombre de campo de la UDT de nivel superior se almacena mediante un identificador. El valor de almacenamiento del identificador depende del número total de campos de la UDT de nivel superior y puede variar entre 1 y 3 bytes:
 - De 1 a 62 campos: 1 byte
 - 63 a 124 campos: 2 bytes
 - 125 campos como máximo: 3 bytes
- valor de campo: los bytes necesarios para almacenar los valores de campo de la UDT de nivel superior dependen del tipo de datos almacenados:
 - Tipo de datos escalares: los bytes necesarios para el almacenamiento son los mismos que para el mismo tipo de datos almacenado en una columna normal.
 - UDT congelado: para cada UDT anidado congelado, el UDT anidado tiene el mismo tamaño que tendría en el protocolo binario CQL. En el caso de una UDT anidada, se almacenan 4 bytes para cada campo (incluidos los campos vacíos) y el valor del campo almacenado es el formato de serialización del valor del campo mediante el protocolo binario CQL.
 - Colecciones congeladas:
 - LIST y SET: en el caso de una LIST o más congeladas anidadas SET, se almacenan 4 bytes para cada elemento de la colección más el formato de serialización del protocolo binario CQL del valor de la colección.
 - MAPA: en el caso de una congelación anidada MAP, cada par clave-valor tiene los siguientes requisitos de almacenamiento:
 - Para cada clave, asigne 4 bytes y, a continuación, añada el formato de serialización del protocolo binario CQL de la clave.
 - Para cada valor, asigne 4 bytes y, a continuación, añada el formato de serialización del protocolo binario CQL del valor.
- Palabra clave CONGELADA: en el caso de las colecciones congeladas anidadas dentro de colecciones congeladas, Amazon Keyspaces no requiere bytes adicionales para los metadatos.

- Palabra clave estática: los datos de las STATIC columnas no se tienen en cuenta 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”](#).

Tenga en cuenta el impacto de las funciones de Amazon Keyspaces en el tamaño de las filas

En esta sección se muestra cómo las características de Amazon Keyspaces afectan al tamaño codificado de una fila.

- Client-side marcas de tiempo: las marcas de Client-side tiempo se almacenan para cada columna de cada fila cuando la función 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 sobre las marcas de tiempo del lado del cliente, consulte [the section called “Client-side marcas de tiempo”](#)
- Tiempo de vida (TTL): los metadatos TTL ocupan aproximadamente 8 bytes por fila cuando la función está activada. Además, los metadatos TTL se almacenan para cada columna de cada fila. Los metadatos TTL ocupan aproximadamente 8 bytes por cada columna y almacenan un tipo de datos escalares o una colección inmovilizada. Si la columna almacena un tipo de datos de recopilación que no está congelado, para cada elemento de la colección, TTL requiere aproximadamente 8 bytes adicionales para los metadatos. Para una columna que almacena un tipo de datos de recopilación cuando el TTL está activado, puede utilizar la siguiente fórmula.

```
total encoded size of column = (column id) + sum (nested elements + collection  
metadata (1 byte) + TTL metadata (8 bytes)) + collection column metadata (3 bytes)
```

Los metadatos TTL contribuyen al coste de almacenamiento y rendimiento de la fila. Para obtener más información sobre TTL, consulte [the section called “Caducidad de datos con período de vida”](#).

Elija la fórmula correcta para calcular el tamaño codificado de una fila

En esta sección se muestran las distintas fórmulas que puede utilizar para estimar los requisitos de almacenamiento o capacidad de procesamiento de una fila de datos en Amazon Keyspaces.

El tamaño total codificado de una fila de datos se puede estimar en función de una de las siguientes fórmulas, en función de su objetivo:

- Capacidad de rendimiento: para estimar el tamaño codificado de una fila a fin de evaluar las unidades de read/write solicitud (RRUs/WRUs) o las unidades de read/write capacidad (RCUs/WCUs) requeridas:

```
total encoded size of row = partition key columns + clustering columns + regular columns
```

- Tamaño de almacenamiento: para estimar el tamaño codificado de una fila y predecirlo `BillableTableSizeInBytes`, añada los metadatos necesarios para el almacenamiento de la fila:

```
total encoded size of row = partition key columns + clustering columns + regular columns + row metadata (100 bytes)
```

Important

Todos los metadatos de las columnas (por ejemplo, los identificadores de columna, los metadatos de las claves de partición, los metadatos de las columnas agrupadas en clústeres, así como las marcas de tiempo del cliente, el TTL y los metadatos de las filas) cuentan para el tamaño máximo de fila de 1 MB.

Ejemplo de cálculo del tamaño de fila

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, suma los bytes de todas las columnas. Para calcular el tamaño facturable del almacenamiento, añade los 100 bytes adicionales para los metadatos de las filas:

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 supervisar los recursos sin servidor con Amazon CloudWatch, consulte [the section called “Monitorización con CloudWatch”](#).

Estimación del consumo de capacidad del rendimiento de lectura y escritura en Amazon Keyspaces

Al leer o escribir datos en Amazon Keyspaces, la cantidad de unidades de read/write solicitud (RRUs/WRUs) o unidades de read/write capacidad (RCUs/WCUs) que consume la consulta depende de la cantidad total de datos que Amazon Keyspaces tenga que 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 aunque se produzca un error en la comprobación condicional.

Para calcular 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 “Estimación del tamaño de las filas”](#).

Temas

- [Estimación del consumo de capacidad de las consultas de rango en Amazon Keyspaces](#)

- [Estimación del consumo de capacidad de lectura de las consultas de límites](#)
- [Estimación del consumo de capacidad de lectura de los análisis de tablas](#)
- [Estimación del consumo de capacidad de las transacciones ligeras en Amazon Keyspaces](#)
- [Estimación del consumo de capacidad de las columnas estáticas en Amazon Keyspaces](#)
- [Estimación y aprovisionamiento de la capacidad de una tabla multirregional en Amazon Keyspaces](#)
- [Calcule el consumo de capacidad de lectura y escritura con Amazon CloudWatch en Amazon Keyspaces](#)

Estimación del consumo de capacidad de las consultas de rango en Amazon Keyspaces

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;
```

Recibe el siguiente conjunto de resultados de la consulta y la operación de lectura realizada por Amazon Keyspaces consume 2 RRU en el modo de coherencia de LOCAL_QUORUM.

```
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` para procesar la consulta. Sin embargo, Amazon Keyspaces solo devuelve la fila en la que la condición `WHERE` especificada en la consulta es verdadera, que es la fila con el 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.

Estimación del consumo de capacidad de lectura de las consultas de límites

Al procesar una consulta que utilice la cláusula `LIMIT`, Amazon Keyspaces lee las filas hasta el tamaño máximo de página cuando intenta cumplir la condición especificada en la consulta. Si Amazon Keyspaces no encuentra suficientes datos coincidentes que coincidan con el valor de `LIMIT` en la primera página, podrían ser necesarias una o más llamadas paginadas. Para continuar leyendo en la página siguiente, puede usar un token de paginación. El tamaño de página predeterminado es de 1 MB. Para consumir menos capacidad de lectura al usar cláusulas `LIMIT`, puede reducir el tamaño de la página. Para obtener más información sobre paginación, consulte [the section called “Paginación de resultados”](#).

A modo de ejemplo, analicemos la siguiente consulta.

```
SELECT * FROM my_table WHERE partition_key=1234 LIMIT 1;
```

Si no configura el tamaño de la página, Amazon Keyspaces lee 1 MB de datos aunque solo devuelva una fila. Para que Amazon Keyspaces lea solo una fila, puede establecer el tamaño de página en 1 para esta consulta. En este caso, Amazon Keyspaces solo leería una fila siempre que no tenga filas caducadas según la Time-to-live configuración o las marcas horarias del lado del cliente.

El `PAGE SIZE` parámetro determina cuántas filas escanea Amazon Keyspaces del disco para cada solicitud, no cuántas filas devuelve Amazon Keyspaces al cliente. Amazon Keyspaces aplica los filtros que usted proporciona, por ejemplo, la desigualdad en las columnas que no son clave o `LIMIT` después de escanear los datos del disco. Si no lo configuras de forma explícita `PAGE SIZE`, Amazon Keyspaces lee hasta 1 MB de datos antes de aplicar los filtros. Por ejemplo, si lo utiliza `LIMIT 1` sin especificar el `PAGE SIZE`, Amazon Keyspaces podría leer miles de filas del disco antes de aplicar la cláusula de límite y devolver solo una fila.

Para evitar la lectura excesiva, reduzca la cantidad de filas `PAGE SIZE` que Amazon Keyspaces escanea para cada búsqueda. Por ejemplo, si lo define `LIMIT 5` en su consulta, establezca un valor entre 5 y 10 `PAGE SIZE` para que Amazon Keyspaces solo escanee de 5 a 10 filas en cada llamada paginada. Puede modificar este número para reducir el número de búsquedas. Para los límites superiores al tamaño de la página, Amazon Keyspaces mantiene el recuento total de resultados con el estado de paginación. En el caso `LIMIT` de 10 000 filas, Amazon Keyspaces puede obtener estos resultados en dos páginas de 5000 filas cada una. El límite de 1 MB es el límite superior de cualquier conjunto de tamaños de página.

Estimación del consumo de capacidad de lectura de los análisis de tablas

Las consultas que dan como resultado un análisis completo de la tabla, por ejemplo, las consultas que utilicen la opción `ALLOW FILTERING`, son otro ejemplo de consultas que procesan más lecturas de las que devuelven como resultado. Además, el consumo de capacidad de lectura se basa en los datos leídos, no en los datos devueltos.

Para el ejemplo de análisis de tabla, utilizamos la siguiente tabla de ejemplo en 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 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 análisis de tablas en la que Amazon Keyspaces escanea las cuatro particiones de la tabla y consume 6 RRU en el modo de coherencia `LOCAL_QUORUM`. En primer lugar, Amazon Keyspaces consume 3 RRU para leer las tres filas con `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 análisis de la tabla, Amazon Keyspaces escanea todas las particiones de la tabla, incluidas las particiones sin datos.

Estimación del consumo de capacidad de las transacciones ligeras en Amazon Keyspaces

Las transacciones ligeras (LWT) permiten realizar operaciones de escritura condicional en función de los datos de la tabla. Las operaciones de actualización condicional resultan útiles para insertar, actualizar y eliminar registros en función de las condiciones que evalúen el estado actual.

En Amazon Keyspaces, todas las operaciones de escritura requieren coherencia de LOCAL_QUORUM y el uso de LWT no incurre en cargos adicionales. La diferencia con los LWT es que, cuando se comprueba el estado del LWT, FALSE Amazon Keyspaces consume unidades de capacidad de escritura (WCU) o unidades de solicitud de escritura (WRU). La cantidad WCUs/WRUs consumida depende del tamaño de la fila.

Por ejemplo, si el tamaño de la fila es de 2 KB, la escritura condicional fallida consumirá dos WCUs/WRUs. Si la fila no existe actualmente en la tabla, la operación consumirá una fila WCUs/WRUs.

Para determinar el número de solicitudes que provocaron errores en la comprobación de estado, puede supervisar la `ConditionalCheckFailed` métrica en CloudWatch.

Calcule los costos de LWT para las tablas con el tiempo de vida (TTL)

Los LWT pueden requerir unidades de capacidad de lectura (RCU) o unidades de solicitud de lectura (RRU) adicionales para las tablas configuradas con TTL que no utilizan marcas de tiempo del lado del cliente. Cuando se utilizan `IF EXISTS` o utilizan `IF NOT EXISTS` palabras clave como resultado una comprobación de estado, se consumen las siguientes unidades de capacidad FALSE:

- RCUs/RRUs — Si la fila existe, el RCUs/RRUs consumo se basa en el tamaño de la fila existente.
- RCUs/RRUs — Si la fila no existe, RCU/RRU se consume una sola.

Si la condición evaluada da como resultado una operación de escritura correcta, WCUs/WRUs se consumen en función del tamaño de la nueva fila.

Estimación del consumo de capacidad de las 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. Para ver las diferencias funcionales con Apache Cassandra cuando

se utilizan columnas estáticas y resultados de lectura de rangos paginados, consulte [the section called “Paginación”](#).

Temas

- [Cálculo de tamaño de columna estática por partición lógica en Amazon Keyspaces](#)
- [Calcule los requisitos de capacidad y rendimiento para read/write las operaciones con datos estáticos en Amazon Keyspaces](#)

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 “Estimación 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 “Estimación 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 “Estimación 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.`

Calcule los requisitos de capacidad y rendimiento para read/write las operaciones con 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 tanto a los modos de capacidad bajo demanda como a los de read/write capacidad 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 ($2 \times 26 \text{ WCU} + 2 \times 2 \text{ WCU}$).

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 monitorizar los recursos sin servidor con Amazon CloudWatch, consulte [the section called “Monitorización con CloudWatch”](#).

Estimación y aprovisionamiento de la capacidad de una tabla multirregional en Amazon Keyspaces

Puede configurar la capacidad de rendimiento de una tabla multirregión de una de estas dos maneras:

- On-demand modo de capacidad, 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.

Amazon Keyspaces utiliza de forma predeterminada el modo bajo demanda para las tablas multirregionales. En el caso de las tablas en modo bajo demanda, no necesita especificar el rendimiento de lectura y escritura que espera de su aplicación. Amazon Keyspaces se adapta de forma instantánea a sus cargas de trabajo a medida que aumentan o disminuyen a cualquier nivel de tráfico alcanzado previamente. Si el nivel de tráfico de una carga de trabajo alcanza un nuevo nivel máximo, Amazon Keyspaces se adapta rápidamente para acomodar la carga de trabajo.

Si elige el modo de capacidad aprovisionada para una tabla, debe configurar la cantidad de unidades de capacidad de lectura (RCU) y de unidades de capacidad de escritura (WCU) por segundo que requiere 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 obligatorio porque cada escritura que se realiza en una región también debe repetirse en cada región de réplica.

Si la tabla no tiene suficiente capacidad para administrar las escrituras de todas las regiones, se producirán excepciones de capacidad. Además, aumentarán los tiempos de espera de las réplicas interregionales.

Por ejemplo, si tiene una tabla multirregional en la que espera 5 escrituras por segundo en Este de EE. UU. (Norte de Virginia), 10 escrituras por segundo en Este de EE. UU. (Ohio) y 5 escrituras por segundo en Europa (Irlanda), debería esperar que la tabla consuma 20 WCU en cada región: Este de EE. UU. (Norte de Virginia), Este de EE. UU. (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”](#).

Cada escritura se factura como 1 WCU, por lo que en este ejemplo se facturarán un total de 60 WCU. Para obtener más información sobre precios, consulte [Precios de Amazon Keyspaces \(para Apache Cassandra\)](#).

Para obtener más información acerca de la capacidad aprovisionada con el escalado automático de Amazon Keyspaces, consulte [the section called “Administración de la capacidad de rendimiento con escalado automático”](#).

Note

Si una tabla se ejecuta en modo de capacidad aprovisionada con escalado automático, se permite que la capacidad de escritura aprovisionada fluctúe dentro de lo que establezca la configuración de escalado automático para cada región.

Calcule el consumo de capacidad de lectura y escritura con Amazon CloudWatch en Amazon Keyspaces

Para estimar y monitorear el consumo de capacidad de lectura y escritura, puede usar 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 que consume una declaración específica CloudWatch, puedes seguir estos pasos.

1. Creación de una nueva tabla con datos de ejemplo
2. Configure un CloudWatch panel de Amazon Keyspaces para la tabla. Para empezar, puede utilizar una plantilla de panel disponible en [Github](#).
3. Ejecute la instrucción CQL, por ejemplo, utilizando la opción `ALLOW FILTERING`, y compruebe en el panel las unidades de capacidad de lectura consumidas durante el análisis completo de la tabla.

Configurar los modos de read/write capacidad en Amazon Keyspaces

Amazon Keyspaces tiene dos modos de read/write capacidad para procesar lecturas y escrituras en las tablas:

- On-demand (predeterminado)
- Aprovisionado

El modo read/write de capacidad que elija controla cómo se le cobrará por el rendimiento de lectura y escritura y cómo se gestionará la capacidad de rendimiento de la tabla.

Temas

- [Configuración del modo de capacidad bajo demanda](#)
- [Configure el modo de capacidad aprovisionada](#)
- [Visualización del modo de capacidad de una tabla en Amazon Keyspaces](#)
- [Cambiar el modo de capacidad de una tabla en Amazon Keyspaces](#)
- [Configuración de precalentamiento para tablas en Amazon Keyspaces](#)

Configuración del 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 precios de pago por solicitud para las solicitudes de lectura y escritura, de forma que solo paga por lo que utiliza.

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.

On-demand el modo es una buena opción si se cumple alguna de las siguientes condiciones:

- Crea nuevas tablas con cargas de trabajo desconocidas.
- Tiene un tráfico de aplicaciones 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 la aplicación. On-demand el modo de capacidad admite al instante hasta el doble del tráfico máximo anterior en una mesa. 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, consulte [the section called “Estimación del 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.

Configure el modo de capacidad 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 ayuda a gestionar el uso de Amazon Keyspaces para que se mantenga igual o inferior a una tasa de solicitudes definida a fin de mantener la previsibilidad. Para obtener más información sobre el escalado automático para el rendimiento aprovisionado, consulte [the section called “Administración de la capacidad de rendimiento con 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 pronosticar los requisitos de capacidad.

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 acerca de cómo estimar el consumo de capacidad de lectura y escritura de una tabla, consulte [the section called “Estimación del 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 “Estimación 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 ($4 \text{ KB} \times 6 \text{ RCU}$).
- Realizar lecturas LOCAL_ONE de hasta 48 KB por segundo (el doble de rendimiento de lectura).
- Escribir hasta 6 KB por segundo ($1 \text{ KB} \times 6 \text{ WCU}$).

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 para el 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 Consola de administración de AWS 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](#).

Visualización del modo de capacidad de una tabla en Amazon Keyspaces

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.

También puede utilizar el AWS CLI para ver el modo de capacidad de una tabla.

Para cambiar el rendimiento aprovisionado de una tabla, utilice [the section called “Cambio del modo de capacidad”](#).

Cassandra Query Language (CQL)

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.

CLI

Vea el modo de capacidad de rendimiento de una tabla mediante el AWS CLI

```
aws keyspaces get-table --keyspace-name myKeyspace --table-name myTable
```

La salida del comando puede tener un aspecto similar a este para una tabla en modo de capacidad aprovisionada.

```
"capacitySpecification": {
  "throughputMode": "PROVISIONED",
  "readCapacityUnits": 4000,
  "writeCapacityUnits": 2000
}
```

El resultado de una tabla en modo bajo demanda tiene el siguiente aspecto.

```
"capacitySpecification": {
  "throughputMode": "PAY_PER_REQUEST",
  "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
}
```

Cambiar el modo de capacidad de una tabla en Amazon Keyspaces

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.

Al cambiar de modo de capacidad, se aplican los siguientes períodos de espera:

- Puede cambiar una tabla recién creada en el modo bajo demanda al modo de capacidad aprovisionada en cualquier momento. Sin embargo, solo puede volver al modo bajo demanda 24 horas después de la marca de tiempo de creación de la tabla.
- Puede cambiar una tabla existente en el modo bajo demanda al modo de capacidad aprovisionada en cualquier momento. Sin embargo, puede cambiar los modos de capacidad aprovisionada al modo bajo demanda solo una vez cada 24 horas.

Cassandra Query Language (CQL)

Cambio del modo de capacidad de rendimiento de una tabla mediante CQL

1. Para cambiar el modo de capacidad de una tabla a `PROVISIONED`, debe configurar las unidades de capacidad de lectura y escritura en función de los valores máximos esperados de la carga de trabajo. Esta instrucción es un ejemplo de ello. También puede ejecutar esta instrucción para ajustar la capacidad de lectura o las unidades de capacidad de escritura de la tabla.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':  
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 6000,  
  'write_capacity_units': 3000}};
```

Para configurar el modo de capacidad aprovisionada con escalado automático, consulte [the section called “Configuración del escalado automático en una tabla existente”](#).

2. Para cambiar el modo de capacidad de una tabla al modo bajo demanda, defina el modo de rendimiento como `PAY_PER_REQUEST`. La siguiente instrucción es un ejemplo de ello.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':  
{'throughput_mode': 'PAY_PER_REQUEST'}};
```

3. Puede usar la siguiente instrucción para confirmar el modo de capacidad de la tabla.

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'catalog' and  
table_name = 'book_awards';
```

Una tabla configurada con el modo de capacidad bajo demanda devuelve lo siguiente.

```
{  
  "capacity_mode":{  
    "last_update_to_pay_per_request_timestamp":"1727952499092",  
    "throughput_mode":"PAY_PER_REQUEST"  
  }  
}
```

El valor `last_update_to_pay_per_request_timestamp` se mide en milisegundos.

CLI

Cambie el modo de capacidad de procesamiento de una tabla mediante el AWS CLI

1. Para cambiar el modo de capacidad de la tabla a `PROVISIONED`, debe configurar las unidades de capacidad de lectura y escritura en función de los valores máximos esperados de la carga de trabajo. El siguiente comando es un ejemplo de esto. También puede ejecutar este comando para ajustar la capacidad de lectura o las unidades de capacidad de escritura de la tabla.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards  
                           \--capacity-specification  
                           throughputMode=PROVISIONED,readCapacityUnits=6000,writeCapacityUnits=3000
```

Para configurar el modo de capacidad aprovisionada con escalado automático, consulte [the section called “Configuración del escalado automático en una tabla existente”](#).

2. Para cambiar el modo de capacidad de una tabla al modo bajo demanda, defina el modo de rendimiento como `PAY_PER_REQUEST`. La siguiente instrucción es un ejemplo de ello.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards
                             \--capacity-specification
                             throughputMode=PAY_PER_REQUEST
```

3. Puede utilizar el siguiente comando para revisar el modo de capacidad configurado para una tabla.

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

El resultado de una tabla en modo bajo demanda tiene el siguiente aspecto.

```
"capacitySpecification": {
  "throughputMode": "PAY_PER_REQUEST",
  "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
}
```

Configuración de precalentamiento para tablas en Amazon Keyspaces

Amazon Keyspaces escala automáticamente las particiones de almacenamiento en función del rendimiento bajo demanda o aprovisionado, pero en el caso de tablas nuevas o picos de rendimiento repentinos, puede llevar más tiempo asignar las particiones de almacenamiento necesarias. Para garantizar que una mesa nueva o existente tenga la capacidad suficiente para soportar los picos de rendimiento previstos, puede configurar manualmente valores de rendimiento en caliente específicos para precalentar la mesa.

El rendimiento cálido se refiere al número de operaciones de lectura y escritura que su tabla de Amazon Keyspaces puede admitir de forma instantánea. Estos valores están disponibles de forma predeterminada para todas las tablas nuevas y existentes. Si utiliza el modo bajo demanda o si actualiza el rendimiento aprovisionado, Amazon Keyspaces garantiza que su aplicación pueda emitir solicitudes de hasta esos valores al instante.

Amazon Keyspaces ajusta automáticamente los valores de rendimiento cálido a medida que aumenta el uso. Para ajustar la capacidad de procesamiento para los próximos picos de actividad, por ejemplo, al migrar datos de otra base de datos, lo que puede requerir la carga de terabytes de datos en un breve período de tiempo, puede aumentar manualmente los valores de rendimiento en caliente de la tabla. Esto resulta útil para los eventos de mayor actividad planificados, en los que las tasas de solicitudes pueden aumentar 10, 100 veces o más. En primer lugar, evalúe si el

rendimiento en caliente actual es suficiente para gestionar el tráfico esperado. [A continuación, si necesita precalentar la mesa para los picos de carga de trabajo planificados, puede aumentar el valor del rendimiento en caliente manualmente sin cambiar la configuración del rendimiento ni el modo de capacidad.](#)

Puede precalentar las tablas para realizar operaciones de lectura, de escritura o ambas. Puede aumentar este valor para las tablas nuevas y existentes de una sola región y las tablas de varias regiones, y la configuración de rendimiento cálido que establezca se aplicará automáticamente a todas las réplicas de las tablas de varias regiones. No hay límite en cuanto al número de mesas de Amazon Keyspaces que puedes precalentar en cualquier momento. El tiempo necesario para completar el precalentamiento depende de los valores que establezca y del tamaño de la mesa. Puede enviar solicitudes de precalentamiento simultáneas y estas solicitudes no interfieren con ninguna de las operaciones de la mesa. Puedes precalentar tu mesa hasta el límite de la cuota de mesa de tu cuenta en esa región. Utilice la [consola Service Quotas](#) para comprobar sus cuotas actuales y aumentarlas si es necesario.

Los valores de rendimiento cálido que Amazon Keyspaces ajusta en función del uso bajo demanda o de la capacidad aprovisionada están disponibles de forma predeterminada para todas las tablas sin cargos adicionales. Sin embargo, si aumentas manualmente los valores de rendimiento en caliente predeterminados para convertirlos en tablas de precalentamiento en caso de picos de tráfico, se aplicarán cargos adicionales. Para obtener más información, consulta los [precios de Amazon Keyspaces](#).

Estos son algunos escenarios y prácticas recomendadas que puede tener en cuenta a la hora de precalentar las tablas de Amazon Keyspaces.

Rendimiento en caliente y patrones de acceso desiguales

Una tabla puede tener un rendimiento óptimo de 30 000 unidades de lectura por segundo y 10 000 unidades de escritura por segundo, pero aun así es posible que se supere la capacidad en las lecturas o escrituras antes de alcanzar esos valores. Se debe probablemente a una partición activa. Si bien Amazon Keyspaces puede seguir escalando para soportar un rendimiento prácticamente ilimitado, cada partición individual está limitada a 1000 unidades de escritura por segundo y 3000 unidades de lectura por segundo. Si su aplicación dirige demasiado tráfico a una pequeña parte de las particiones de la tabla, los eventos de exceso de capacidad pueden producirse incluso antes de que se alcancen los valores de rendimiento máximo de la tabla. Recomendamos seguir las [prácticas recomendadas de Amazon Keyspaces](#) para garantizar una escalabilidad perfecta y evitar las particiones activas.

Rendimiento en caliente para una tabla aprovisionada

Considere una tabla aprovisionada que tiene un rendimiento en caliente de 30 000 unidades de lectura por segundo y 10 000 unidades de escritura por segundo, pero que actualmente tiene un rendimiento aprovisionado de 4 000 RCU y 8 000 WCU. Puede ampliar al instante el rendimiento aprovisionado de la tabla hasta 30 000 RCU o 10 000 WCU actualizando la configuración de rendimiento aprovisionada. A medida que aumenta el rendimiento aprovisionado por encima de estos valores, el rendimiento en espera se ajusta automáticamente a los nuevos valores más altos, ya que ha establecido un nuevo rendimiento máximo. Por ejemplo, si establece el rendimiento aprovisionado en 50 000 RCU, el rendimiento en caliente aumenta a 50 000 unidades de lectura por segundo.

```
"ProvisionedThroughput":
{
  "ReadCapacityUnits": 4000,
  "WriteCapacityUnits": 8000
}
"WarmThroughput":
{
  "ReadUnitsPerSecond": 30000,
  "WriteUnitsPerSecond": 10000
}
```

Rendimiento en caliente de una tabla bajo demanda

Una nueva tabla bajo demanda comienza con un rendimiento en caliente de 12 000 unidades de lectura por segundo y 4000 unidades de escritura por segundo. La tabla puede acomodar de forma instantánea un tráfico sostenido hasta estos niveles. Cuando sus solicitudes superan las 12 000 unidades de lectura por segundo o las 4000 unidades de escritura por segundo, el rendimiento en caliente se ajusta automáticamente a los valores más altos.

```
"WarmThroughput":
{
  "ReadUnitsPerSecond": 12000,
  "WriteUnitsPerSecond": 4000
}
```

Mejores prácticas para precalentar las tablas de Amazon Keyspaces

Siga estas prácticas recomendadas al implementar el precalentamiento en sus tablas de Amazon Keyspaces:

Calcule con precisión la capacidad requerida

Dado que el precalentamiento implica un coste único, calcule cuidadosamente el rendimiento necesario en función de la carga de trabajo prevista para evitar el aprovisionamiento excesivo.

Tenga en cuenta el esquema de la tabla

Las tablas con filas más grandes pueden requerir más particiones para obtener el mismo rendimiento. Tenga en cuenta el tamaño medio de las filas al estimar los requisitos de precalentamiento.

Controle el rendimiento de la mesa

Tras el precalentamiento, utiliza CloudWatch las métricas para comprobar que la mesa está gestionando la carga según lo previsto. Para obtener más información, consulte [the section called “Controle el precalentamiento con CloudWatch”](#).

Administración de las cuotas

Si su aplicación requiere un rendimiento superior al permitido por las cuotas predeterminadas (40 000 RCUs/WCUs o 2000 particiones), solicite un aumento de la cuota con bastante antelación a la fecha en que se produzca el evento de alto tráfico. Para solicitar un aumento de una cuota, use la [consola de Service Quotas](#).

Optimización de costos

En el caso de eventos temporales con mucho tráfico, considere la posibilidad de utilizar el precalentamiento en lugar de cambiar al modo aprovisionado con gran capacidad, ya que puede resultar más rentable para los eventos de corta duración. Para obtener más información sobre los precios, consulta los precios de [Amazon Keyspaces](#).

Note

Supervise las métricas de rendimiento de su aplicación durante la fase de prueba para validar que la configuración de precalentamiento se adapte adecuadamente a sus requisitos de carga de trabajo.

Temas

- [Creación de una tabla de Amazon Keyspaces nueva con rendimiento en caliente más alto](#)
- [Aumento del rendimiento en caliente de la tabla de Amazon Keyspaces existente](#)

- [Vea el rendimiento cálido de una tabla de Amazon Keyspaces](#)
- [Supervisa el rendimiento de una mesa precalentada con Amazon CloudWatch](#)

Creación de una tabla de Amazon Keyspaces nueva con rendimiento en caliente más alto

Puede ajustar los valores de rendimiento cálido al crear su tabla de Amazon Keyspaces mediante la consola, el CQL o el AWS CLI

Console

¿Cómo crear una tabla nueva con una configuración de rendimiento cálido

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 para la tabla.
5. En la sección Clave principal, defina la clave principal de la tabla y seleccione las columnas de agrupación en clústeres opcionales.
6. En la sección Configuración de la tabla, elija Personalizar configuración.
7. Continúe con la configuración Read/write de capacidad.
8. Para el modo de capacidad, puede elegir entre el modo Aprovisionado On-demando el modo Aprovisionado.
9. En la sección Pre-warming para tablas, puede aumentar los valores de las unidades de lectura por segundo y las unidades de escritura por segundo según sea necesario para preparar la tabla para gestionar los picos de actividad planificados.

Los valores de rendimiento cálido que Amazon Keyspaces ajusta en función del uso bajo demanda o de la capacidad aprovisionada están disponibles de forma predeterminada para todas las tablas sin cargos adicionales. Tenga en cuenta que si aumenta manualmente los valores de rendimiento en caliente predeterminados para precalentar la tabla en caso de picos de tráfico, se aplicarán cargos adicionales.

10. Configure otras funciones opcionales de la tabla según sea necesario. A continuación, selecciona Crear tabla.

Cassandra Query Language (CQL)

- Cree una tabla con un rendimiento óptimo mediante uno de los siguientes métodos:
 - Para el modo aprovisionado, cree una tabla y especifique la capacidad máxima esperada de lectura y escritura mediante la siguiente sintaxis de CQL:

```
CREATE TABLE catalog.book_awards (  
    year int,  
    award text,  
    rank int,  
    category text,  
    book_title text,  
    author text,  
    publisher text,  
    PRIMARY KEY ((year, award), category, rank))  
WITH CUSTOM_PROPERTIES = {  
    'capacity_mode': {  
        'throughput_mode': 'PROVISIONED',  
        'read_capacity_units': 20000,  
        'write_capacity_units': 10000  
    },  
    'warm_throughput': {  
        'read_units_per_second': 40000,  
        'write_units_per_second': 20000  
    }  
};
```

- Para el modo bajo demanda, cree una tabla y especifique la capacidad máxima esperada de lectura y escritura mediante la siguiente sintaxis de CQL:

```
CREATE TABLE catalog.book_awards (  
    year int,  
    award text,  
    rank int,  
    category text,  
    book_title text,  
    author text,  
    publisher text,  
    PRIMARY KEY ((year, award), category, rank))  
WITH CUSTOM_PROPERTIES = {  
    'capacity_mode': {  
        'throughput_mode': 'PAY_PER_REQUEST'
```

```
    },
    'warm_throughput': {
        'read_units_per_second': 40000,
        'write_units_per_second': 20000
    }
};
```

Para confirmar la configuración de capacidad de la tabla, consulte [the section called “Vea el rendimiento en caliente de su mesa”](#).

CLI

1. Cree una tabla con un rendimiento óptimo mediante uno de los siguientes métodos mediante el AWS CLI
 - Cree una tabla nueva en el modo aprovisionado y especifique los valores de la capacidad máxima de lectura y escritura prevista para la nueva tabla. La siguiente instrucción es un ejemplo de ello.

```
aws keyspaces create-table \
--keyspace-name 'catalog' \
--table-name 'book_awards' \
--schema-definition 'allColumns=[{name=year,type=int},
{name=award,type=text},{name=rank,type=int},{name=category,type=text},
{name=book_title,type=text},{name=author,type=text},
{name=publisher,type=text}],partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},
{name=rank,orderBy=ASC}]' \
--capacity-specification
    throughputMode=PROVISIONED,readCapacityUnits=20000,writeCapacityUnits=10000
\
--warm-throughput-specification
    readUnitsPerSecond=40000,writeUnitsPerSecond=20000
```

- Cree una tabla nueva en modo bajo demanda y especifique los valores de capacidad máxima esperados para las lecturas y escrituras de la nueva tabla. La siguiente instrucción es un ejemplo de ello.

```
aws keyspaces create-table \
--keyspace-name 'catalog' \
--table-name 'book_awards' \
```

```
--schema-definition 'allColumns=[{name=year,type=int},
{name=award,type=text},{name=rank,type=int},{name=category,type=text},
{name=book_title,type=text},{name=author,type=text},
{name=publisher,type=text}],partitionKeys=[{name=year},
{name=award}],clusteringKeys=[{name=category,orderBy=ASC},
{name=rank,orderBy=ASC}]' \
--warmThroughputSpecification
readUnitsPerSecond=40000,writeUnitsPerSecond=20000
```

2. El resultado del comando devuelve el ARN de la tabla, como se muestra en el siguiente ejemplo.

```
{
  "resourceArn": "arn:aws::cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards>"
}
```

Para confirmar la configuración de capacidad de la tabla, consulte [the section called “Vea el rendimiento en caliente de su mesa”](#).

Java

Cree una tabla nueva con el SDK for Java.

- Cree una tabla nueva en el modo aprovisionado y especifique los valores de la capacidad máxima de lectura y escritura prevista para la nueva tabla. El siguiente ejemplo de código es un ejemplo de ello.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.*;

public class PreWarmingExample {
    public static void main(String[] args) {
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();

        // Define schema
        List<ColumnDefinition> columns = Arrays.asList(
            ColumnDefinition.builder().name("year").type("int").build(),
            ColumnDefinition.builder().name("award").type("text").build(),
            ColumnDefinition.builder().name("rank").type("int").build(),
            ColumnDefinition.builder().name("category").type("text").build(),
```

```
        ColumnDefinition.builder().name("book_title").type("text").build(),
        ColumnDefinition.builder().name("author").type("text").build(),
        ColumnDefinition.builder().name("publisher").type("text").build()
    );

    List<PartitionKey> partitionKeys = Arrays.asList(
        PartitionKey.builder().name("year").build(),
        PartitionKey.builder().name("award").build()
    );

    List<ClusteringKey> clusteringKeys = Arrays.asList(
        ClusteringKey.builder().name("category").orderBy("ASC").build(),
        ClusteringKey.builder().name("rank").orderBy("ASC").build()
    );

    SchemaDefinition schema = SchemaDefinition.builder()
        .allColumns(columns)
        .partitionKeys(partitionKeys)
        .clusteringKeys(clusteringKeys)
        .build();

    // Define capacity specification
    CapacitySpecification capacitySpec = CapacitySpecification.builder()
        .throughputMode(ThroughputMode.PROVISIONED)
        .readCapacityUnits(20000)
        .writeCapacityUnits(10000)
        .build();

    // Define warm throughput specification
    WarmThroughputSpecification warmThroughput =
    WarmThroughputSpecification.builder()
        .readUnitsPerSecond(40000L)
        .writeUnitsPerSecond(20000L)
        .build();

    // Create table with PreWarming
    CreateTableRequest request = CreateTableRequest.builder()
        .keyspaceName("catalog")
        .tableName("book_awards")
        .schemaDefinition(schema)
        .capacitySpecification(capacitySpec)
        .warmThroughputSpecification(warmThroughput)
        .build();
```

```
CreateTableResponse response = keyspacesClient.createTable(request);
System.out.println("Table created with ARN: " + response.resourceArn());
    }
}
```

Aumento del rendimiento en caliente de la tabla de Amazon Keyspaces existente

Puede aumentar los valores de rendimiento cálido actuales de su tabla de Amazon Keyspaces mediante la consola, el CQL o el AWS CLI

Console

¿Cómo aumentar la configuración de precalentamiento de una mesa mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, seleccione Tablas y, a continuación, elija la tabla que desee actualizar.
3. En la pestaña Capacidad de la tabla, continúe Pre-warming buscando tablas.
4. En la sección Pre-warming para tablas, elija Editar.
5. En la página Editar el precalentamiento de las tablas, puede actualizar los valores de las unidades de lectura por segundo y de las unidades de escritura por segundo.
6. Seleccione Save changes (Guardar cambios). La tabla se está actualizando con los ajustes de precalentamiento especificados.

Cassandra Query Language (CQL)

Aumente la configuración de rendimiento en caliente de una mesa con CQL

- Utilice ALTER TABLE esta afirmación para aumentar el rendimiento en caliente de una mesa. La siguiente instrucción es un ejemplo de ello.

```
ALTER TABLE catalog.book_awards
WITH CUSTOM_PROPERTIES = {
    'warm_throughput': {
        'read_units_per_second': 60000,
        'write_units_per_second': 30000
    }
}
```

```
};
```

Para confirmar la configuración de capacidad actualizada de la tabla, consulte [the section called “Vea el rendimiento en caliente de su mesa”](#)

CLI

Aumente los ajustes de precalentamiento de una mesa con el AWS CLI

- Para aumentar el rendimiento de calentamiento de la mesa, puede utilizar el comando `update-table`. La siguiente instrucción es un ejemplo de ello.

```
aws keyspaces update-table \  
--keyspace-name 'catalog' \  
--table-name 'book_awards' \  
--warmThroughputSpecification readUnitsPerSecond=60000,writeUnitsPerSecond=30000
```

Para confirmar la configuración de capacidad actualizada de la tabla, consulte [the section called “Vea el rendimiento en caliente de su mesa”](#)

Java

Actualice la configuración de precalentamiento de una tabla mediante el SDK for Java.

- Actualice la configuración de rendimiento en caliente de una mesa. El siguiente ejemplo de código es un ejemplo de ello.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;  
import software.amazon.awssdk.services.keyspaces.model.*;  
  
public class UpdatePreWarmingExample {  
    public static void main(String[] args) {  
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();  
  
        // Define new warm throughput specification  
        WarmThroughputSpecification warmThroughput =  
        WarmThroughputSpecification.builder()  
            .readUnitsPerSecond(60000L)  
            .writeUnitsPerSecond(30000L)  
            .build();
```

```
// Update table with new PreWarming settings
UpdateTableRequest request = UpdateTableRequest.builder()
    .keyspaceName("catalog")
    .tableName("book_awards")
    .warmThroughputSpecification(warmThroughput)
    .build();

UpdateTableResponse response = keyspacesClient.updateTable(request);
System.out.println("Table update requested: " + response.resourceArn());
}
}
```

Vea el rendimiento cálido de una tabla de Amazon Keyspaces

Puede ver los valores de rendimiento cálido actuales de su tabla de Amazon Keyspaces mediante la consola, el CQL o el AWS CLI

Console

Cómo ver los ajustes de precalentamiento de la mesa mediante la consola.

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, seleccione Tablas y, a continuación, elija la tabla que desee revisar.
3. En la pestaña Capacidad de la tabla, continúe Pre-warming buscando tablas.

Cassandra Query Language (CQL)

Vea la configuración de rendimiento en caliente de una tabla mediante CQL

- Para ver la configuración de rendimiento en caliente de una tabla, puede utilizar la siguiente sentencia de CQL.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name='catalog' and table_name='book_awards';

// Output:
```

```
...
custom_properties
-----
{
  'warm_throughput':
  {
    'read_units_per_second': '40000',
    'write_units_per_second': '20000',
    'status': 'AVAILABLE'
  }
}
...
```

CLI

Para ver la configuración de rendimiento en caliente de una tabla, utilice el AWS CLI

- Puede ver la configuración de rendimiento en caliente de una tabla mediante el `get-table` comando que se muestra en el siguiente ejemplo.

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

A continuación, se muestra el ejemplo del resultado del `get-table` comando para una tabla de una sola región en modo aprovisionado.

```
{
  "keyspaceName": "catalog",
  "tableName": "book_awards",
  ... Existing Fields ...,
  "capacitySpecificationSummary": {
    "throughputMode": "PROVISIONED",
    "readCapacityUnits": 20000,
    "writeCapacityUnits": 10000
  },
  "warmThroughputSpecificationSummary": {
    "readUnitsPerSecond": 40000,
    "writeUnitsPerSecond": 20000,
    "status": "AVAILABLE"
  }
}
```



```
}
```

A continuación, se muestra el resultado de ejemplo de una tabla de una sola región en modo bajo demanda.

```
{
  "keyspaceName": "catalog",
  "tableName": "book_awards_ondemand",
  ... Existing Fields ...,
  "capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST"
  },
  "warmThroughputSpecificationSummary": {
    "readUnitsPerSecond": 40000,
    "writeUnitsPerSecond": 20000,
    "status": "AVAILABLE"
  }
}
```

Java

Lea la configuración de precalentamiento de una tabla con el SDK for Java.

- Lea los valores de rendimiento en caliente de una tabla utilizando. `get-table` El siguiente ejemplo de código es un ejemplo de ello.

```
import software.amazon.awssdk.services.keyspaces.KeyspacesClient;
import software.amazon.awssdk.services.keyspaces.model.*;

public class GetTableWithPreWarmingExample {
    public static void main(String[] args) {
        KeyspacesClient keyspacesClient = KeyspacesClient.builder().build();

        // Get table details including PreWarming specification
        GetTableRequest request = GetTableRequest.builder()
            .keyspaceName("catalog")
            .tableName("book_awards")
            .build();

        GetTableResponse response = keyspacesClient.getTable(request);
```

```

        // Access PreWarming details
        if (response.warmThroughputSpecification() != null) {
            WarmThroughputSpecificationSummary warmThroughputSummary =
response.warmThroughputSpecification();
            System.out.println("PreWarming Status: " +
warmThroughputSummary.status());
            System.out.println("Read Units: " +
warmThroughputSummary.readUnitsPerSecond());
            System.out.println("Write Units: " +
warmThroughputSummary.writeUnitsPerSecond());

            // Check if PreWarming is active
            if (warmThroughputSummary.status().equals("AVAILABLE")) {
                System.out.println("Table is fully pre-warmed and ready for high
throughput");
            } else if (warmThroughputSummary.status().equals("UPDATING")) {
                System.out.println("Table PreWarming is currently being
updated");
            }
            } else {
                System.out.println("Table does not have PreWarming enabled");
            }
        }
    }
}

```

Supervisa el rendimiento de una mesa precalentada con Amazon CloudWatch

El precalentamiento de Amazon Keyspaces no introduce nuevas CloudWatch métricas, pero puede supervisar el rendimiento de las tablas precalentadas utilizando las métricas existentes de Amazon Keyspaces:

SuccessfulRequestLatency

Supervisa esta métrica para comprobar que la tabla precalentada gestiona las solicitudes con la latencia esperada.

WriteThrottleEvents y ReadThrottleEvents

Estas métricas deberían permanecer bajas para una mesa debidamente precalentada. Si detectas errores de capacidad insuficientes a pesar del precalentamiento, es posible que tengas que ajustar los valores de rendimiento de calentamiento.

ConsumedReadCapacityUnits y ConsumedWriteCapacityUnits

Estas métricas muestran el consumo real de capacidad, lo que puede ayudar a validar si la configuración de precalentamiento es adecuada.

ProvisionedReadCapacityUnits y ProvisionedWriteCapacityUnits

En el caso de las tablas aprovisionadas, estas métricas muestran la capacidad asignada actualmente.

Estas métricas se pueden ver en la CloudWatch consola o consultarse mediante la CloudWatch API. Para obtener más información, consulte [the section called “Monitorización con CloudWatch”](#).

Administración automática de la capacidad de rendimiento 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.

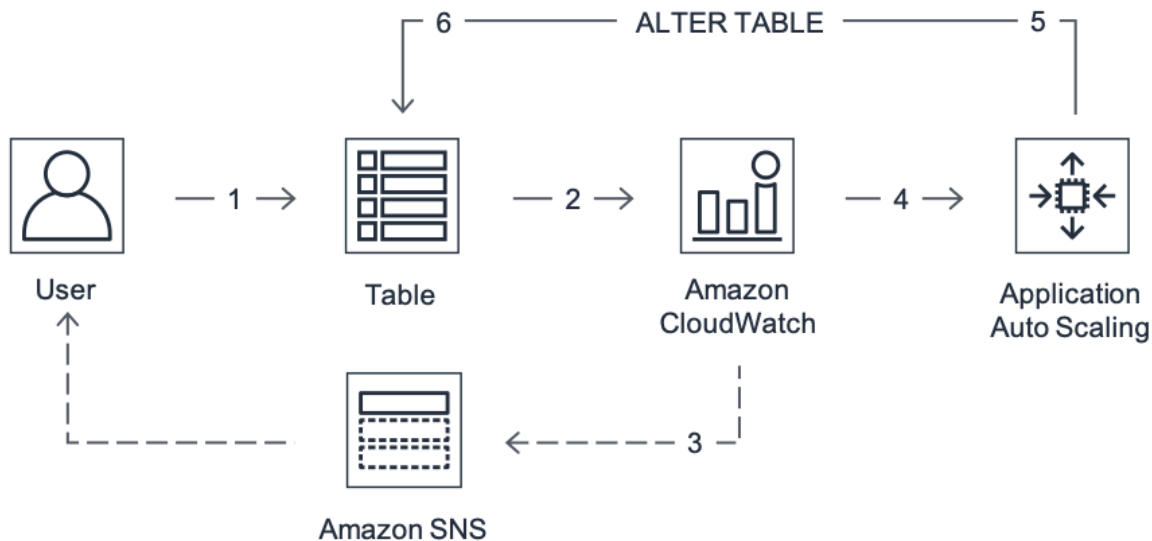
Amazon Keyspaces (para Apache Cassandra) le ayuda a aprovisionar capacidad de rendimiento de forma eficiente para cargas de trabajo variables ajustando de forma automática la capacidad de rendimiento en respuesta al tráfico real de la aplicación. Amazon Keyspaces utiliza el servicio Application Auto 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 ver una introducción rápida al escalado automático de Amazon Keyspaces, consulte [the section called “Configuración y actualización de las políticas de escalado automático”](#).

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 % de su capacidad de lectura y escritura. La tasa de utilización objetivo predeterminada es del 70 %. Puede fijar la utilización objetivo en un porcentaje inferior si su tráfico cambia rápidamente y desea que la capacidad empiece a escalar antes. También puede fijar la tasa de utilización objetivo en un valor más alto 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 de 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 se mantiene elevada (o deprimida) durante un periodo sostenido de varios minutos. El algoritmo de seguimiento de objetivo 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.

Funcionamiento del escalado automático para tablas multirregión

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.

Al utilizar una tabla multirregión en modo aprovisionado con escalado automático, no puede 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 replique la tabla.

Los ajustes de escalado automático 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 Región de AWS en función del uso en esa región. Como resultado, la capacidad aprovisionada en cada región para una tabla multirregión podría ser diferente cuando el escalado automático esté 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 multirregión, consulte [the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional”](#).

Note

Si usa el escalado automático para tablas multirregión, siempre debe usar las operaciones de la API de 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 no está disponible en la región de Oriente Medio (EAU).
- 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 ninguna de 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 “Desactivación del escalado automático de Amazon Keyspaces para una tabla”](#).
- Si utiliza políticas de escalado CloudFormation para crear políticas de escalado, debe administrar las políticas de escalado 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 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.

Configuración y actualización de las políticas de escalado automático de Amazon Keyspaces

Puede usar la consola, CQL o AWS Command Line Interface (AWS CLI) para configurar el escalado automático de Amazon Keyspaces para tablas nuevas y existentes. También puede modificar la configuración del escalado automático o deshabilitarlo.

Para funciones más avanzadas, como establecer tiempos de enfriamiento de escalado vertical y escalable, le recomendamos que utilice CQL o las políticas de escalado para gestionar las políticas de escalado de AWS CLI Amazon Keyspaces.

Temas

- [Configuración de permisos para el escalado automático de Amazon Keyspaces](#)
- [Creación de una nueva tabla con escalado automático](#)
- [Configuración del escalado automático en una tabla existente](#)
- [Visualización de la configuración de escalado automático de Amazon Keyspaces de la tabla](#)
- [Desactivación del escalado automático de Amazon Keyspaces para una tabla](#)

- [Ver la actividad de escalado automático de una tabla de Amazon Keyspaces en Amazon CloudWatch](#)

Configuración de permisos para el escalado automático de Amazon Keyspaces

Para empezar, confirme que la entidad principal 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 o un rol de IAM para el acceso a la consola de Amazon Keyspaces y el escalado automático de Amazon Keyspaces, añada la siguiente política.

Asociación de la política **AmazonKeyspacesFullAccess**

1. Inicie sesión en la consola de IAM Consola de administración de AWS y ábrala en. <https://console.aws.amazon.com/iam/>
2. En el panel de la consola de IAM, elija Usuarios y, a continuación, seleccione el usuario o el rol de IAM en la lista.
3. En la página Resumen, elija Añadir permisos.
4. Elija Asociar 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

Al crear una nueva tabla de Amazon Keyspaces, puede habilitar automáticamente el escalado automático de la capacidad de escritura o lectura de la tabla. 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 multirregión y configurar distintos ajustes de escalado automático para tablas multirregión, consulte [the section called “Creación de una tabla multirregional en modo aprovisionado”](#).

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 obtener más información, consulte [the section called “Cómo utilizar roles vinculados a servicios”](#).

Console

Creación de una nueva tabla con escalado automático habilitado mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 para la tabla.
5. En la sección Clave principal, defina la clave principal de la tabla y seleccione las columnas de agrupación en clústeres opcionales.
6. En la sección Configuración de la tabla, elija Personalizar configuración.
7. Continúe con la configuración Read/write de capacidad.
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 capacidad de lectura o configure los valores de capacidad manualmente.
11. Seleccione Create table (Creación de tabla). Su tabla se crea con los parámetros de escalado automático especificados.

Cassandra Query Language (CQL)

Creación de una nueva tabla con escalado automático de Amazon Keyspaces mediante CQL

Para configurar los ajustes de escalado automático de una tabla mediante programación, utilice la instrucción `AUTOSCALING_SETTINGS` que contiene los parámetros del escalado automático de Amazon Keyspaces. Los parámetros definen las condiciones que indican a Amazon Keyspaces que debe ajustar el rendimiento aprovisionado de la tabla y las acciones opcionales adicionales que debe 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 rendimiento en su nombre. Los siguientes valores son necesarios:

- `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 de objetivos, debe configurar los siguientes parámetros.
 - `target_value`: el escalado automático de Amazon Keyspaces garantiza que la relación entre capacidad consumida y capacidad aprovisionada se mantenga en este valor o cerca del mismo. `target_value` se define como un porcentaje.
 - `disableScaleIn`: (opcional) un valor boolean que especifica si el `scale-in` está deshabilitado o habilitado para la tabla. Este parámetro está deshabilitado de forma predeterminada. Para activar el `scale-in`, establezca el valor boolean como `FALSE`. Esto significa que la capacidad se reduce verticalmente de forma automática para una tabla en su 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 especifica un valor, el valor predeterminado es 0. Para obtener más información acerca del seguimiento de objetivos y los periodos de recuperación, 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 un valor, el valor predeterminado es 0. Para obtener más información acerca del seguimiento de objetivos y los periodos de recuperación, 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 proporción entre capacidad consumida y aprovisionada es ahora de $(150/200)$, es decir, del 75 %. Esta proporción supera su objetivo, de modo que el escalado automático aumenta la capacidad de escritura aprovisionada a 215 para que la proporción sea de $(150/215)$, es decir, del 69,77 %; de esta forma se mantiene lo más próxima posible al valor de `target_value`, pero sin superarlo.

Para `mytable`, establezca `TargetValue` tanto para la lectura como para la escritura en un 50 %. El escalado automático de Amazon Keyspaces ajusta el rendimiento aprovisionado de la tabla dentro del rango comprendido entre 5 y 10 unidades de capacidad, de tal forma que la proporción entre capacidad consumida y aprovisionada se mantiene en el 50 % o en un valor próximo a este. Para la capacidad de lectura, los valores de `ScaleOutCooldown` y `ScaleInCooldown` se establecen en 60 segundos.

Puede usar la siguiente instrucción 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
        }
    }
};
```

CLI

Cree una tabla nueva con el escalado automático de Amazon Keyspaces mediante el AWS CLI

Para configurar los ajustes de escalado automático de una tabla mediante programación, utilice la acción `autoScalingSpecification` que define los parámetros del escalado automático de Amazon Keyspaces. Los parámetros definen las condiciones que indican a Amazon Keyspaces que debe ajustar el rendimiento aprovisionado de la tabla y las acciones opcionales adicionales que debe 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 rendimiento en su nombre. Puede activar el escalado automático de 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 de objetivos, debe configurar los siguientes parámetros.
 - **targetValue:** el escalado automático de Amazon Keyspaces garantiza que la relación entre capacidad consumida y capacidad aprovisionada se mantenga en este valor o cerca del mismo. `targetValue` se define como un porcentaje.
 - **disableScaleIn:** (opcional) un valor boolean que especifica si el `scale-in` está deshabilitado o habilitado para la tabla. Este parámetro está deshabilitado de forma predeterminada. Para activar el `scale-in`, establezca el valor boolean como `FALSE`.

Esto significa que la capacidad se reduce verticalmente de forma automática para una tabla en su 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 acerca del seguimiento de objetivos y los periodos de recuperación, 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 acerca del seguimiento de objetivos y los periodos de recuperación, 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 proporción entre capacidad consumida y aprovisionada es ahora de $(150/200)$, es decir, del 75 %. Esta proporción supera su objetivo, de modo que el escalado automático aumenta la capacidad de escritura aprovisionada a 215 para que la proporción sea de $(150/215)$, es decir, del 69,77 %; de esta forma se mantiene lo más próxima posible al valor de `TargetValue`, pero sin superarlo.

Para `mytable`, establezca `TargetValue` tanto para la lectura como para la escritura en un 50 %. El escalado automático de Amazon Keyspaces ajusta el rendimiento aprovisionado de la tabla dentro del rango comprendido entre 5 y 10 unidades de capacidad, de tal forma que la proporción entre capacidad consumida y aprovisionada se mantiene en el 50 % o en un valor próximo a este. Para la capacidad de lectura, los valores de `ScaleOutCooldown` y `ScaleInCooldown` se establecen en 60 segundos.

Al crear tablas con ajustes de escalado automático complejos, resulta útil cargar los ajustes de escalado automático desde un archivo JSON. Para el siguiente ejemplo, puede descargar el archivo JSON de ejemplo de [auto-scaling.zip](#) y extraer `auto-scaling.json`, tomando nota de la ruta de acceso 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
```

Configuración del escalado automático en una tabla existente

Puede actualizar una tabla de Amazon Keyspaces existente para activar el escalado automático de la capacidad de escritura o lectura de la tabla. Si está actualizando una tabla que actualmente está en modo de capacidad bajo demanda, primero tiene que cambiar el modo de capacidad de la tabla al modo de capacidad aprovisionada.

Para obtener más información sobre cómo actualizar la configuración de escalado automático para una tabla multirregión, consulte [the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional”](#).

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 obtener más información, consulte [the section called “Cómo utilizar roles vinculados a servicios”](#).

Console

Configuración del escalado automático de Amazon Keyspaces para una tabla existente

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. Elija la tabla con la que quiera trabajar y, a continuación, seleccione la pestaña Capacidad.
3. En la sección Configuración de capacidad, elija Editar.

4. En 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 [the section called “Creación de una nueva tabla con escalado automático”](#) para editar la capacidad de lectura y escritura.
6. Cuando se hayan definido los ajustes de escalado automático, elija Guardar.

Cassandra Query Language (CQL)

Configuración de una tabla existente con el escalado automático de Amazon Keyspaces mediante CQL

Puede utilizar la instrucción `ALTER TABLE` para que una tabla de Amazon Keyspaces existente active el escalado automático de la capacidad de escritura o lectura de la tabla. Si va a actualizar una tabla que actualmente esté en modo de capacidad bajo demanda, debe configurar `capacity_mode` como aprovisionada. Si la tabla ya está en el modo de capacidad aprovisionada, puede omitir este campo.

En el siguiente ejemplo, la instrucción actualiza la tabla `mytable`, que está en modo de capacidad bajo demanda. La instrucción 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 5-10 unidades de capacidad con un valor objetivo del 50 %. La capacidad de lectura también se configura dentro del rango de 5-10 unidades de capacidad con un valor objetivo del 50 %. Para la capacidad de lectura, los valores de `scale_out_cooldown` y `scale_in_cooldown` se establecen 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
            }
        }
    }
};

```

CLI

Configure una tabla existente con el escalado automático de Amazon Keyspaces mediante el AWS CLI

Para una tabla de Amazon Keyspaces existente, puede activar el escalado automático de la capacidad de escritura o lectura de la tabla utilizando la operación `UpdateTable`.

Puede utilizar el siguiente comando para habilitar el escalado automático de Amazon Keyspaces para 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 de [auto-scaling.zip](#) y extraer `auto-scaling.json`, tomando nota de la ruta de acceso 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 utilizado en el siguiente ejemplo, consulte [the section called “Creación de una nueva tabla con escalado automático”](#).

```

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

```

Visualización de la configuración de escalado automático de Amazon Keyspaces de la tabla

Puede utilizar la consola, el CQL o el AWS CLI para ver y actualizar la configuración de escalado automático de Amazon Keyspaces de una tabla.

Console

Visualización de la configuración de escalado automático mediante la consola

1. Elija la tabla que quiera ver y seleccione la pestaña Capacidad.
2. En la sección Configuración de capacidad, elija Editar. Ahora puede modificar los ajustes en las secciones Capacidad de lectura o Capacidad de escritura. Para obtener más información sobre estas opciones, consulte [the section called “Creación de una nueva tabla con escalado automático”](#).

Cassandra Query Language (CQL)

Visualización de la política de escalado automático de Amazon Keyspaces de la tabla 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}}}
```

CLI

Consulta la política 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 operación `get-table-auto-scaling-settings`. El siguiente comando de la CLI es un ejemplo de esto.

```
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-1:111122223333:/keyspace/mykeyspace/
table/mytable",
  "autoScalingSpecification": {
    "writeCapacityAutoScaling": {
      "autoScalingDisabled": false,
      "minimumUnits": 5,
      "maximumUnits": 10,
      "scalingPolicy": {
        "targetTrackingScalingPolicyConfiguration": {
          "disableScaleIn": false,
          "scaleInCooldown": 0,
          "scaleOutCooldown": 0,
          "targetValue": 50.0
        }
      }
    },
    "readCapacityAutoScaling": {
      "autoScalingDisabled": false,
      "minimumUnits": 5,
      "maximumUnits": 10,
      "scalingPolicy": {
        "targetTrackingScalingPolicyConfiguration": {
```

```
        "disableScaleIn": false,  
        "scaleInCooldown": 60,  
        "scaleOutCooldown": 60,  
        "targetValue": 50.0  
      }  
    }  
  }  
}
```

Desactivación del escalado automático de Amazon Keyspaces para una tabla

Puede desactivar el escalado automático de Amazon Keyspaces para la tabla en cualquier momento. Si ya no necesita escalar la capacidad de lectura o escritura de la tabla, debería plantearse desactivar el escalado automático para que Amazon Keyspaces deje de modificar continuamente la configuración de la capacidad de lectura o escritura de su tabla. Puede actualizar la tabla con la consola, CQL o la AWS CLI.

Al desactivar el escalado automático, también se eliminan las CloudWatch alarmas que se crearon en tu nombre.

Para eliminar el rol vinculado a servicios utilizado por Application Auto Scaling para acceder a la tabla de 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 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.

Console

Desactivación del escalado automático de Amazon Keyspaces para la tabla mediante la consola

Uso de la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>

2. Elija la tabla que quiera actualizar y seleccione la pestaña Capacidad.
3. En la sección Configuración de capacidad, elija Editar.
4. Para deshabilitar el escalado automático de Amazon Keyspaces, anule la selección de la casilla Escalado automático. Al deshabilitar el escalado automático se anula el registro de la tabla como objetivo escalable con Application Auto Scaling.

Cassandra Query Language (CQL)

Desactivación del escalado automático de Amazon Keyspaces para la tabla mediante CQL

La siguiente instrucción desactiva el escalado automático de la capacidad de escritura de la tabla mytable.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_write_capacity_autoscaling_update': {
        'autoscaling_disabled': true
    }
};
```

CLI

Desactive el escalado automático de Amazon Keyspaces para su tabla mediante el AWS CLI

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}
```

Ver la actividad de escalado automático de una tabla de Amazon Keyspaces en Amazon CloudWatch

Puede supervisar cómo 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.

Uso eficaz de la capacidad de ampliación 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.

Uso de las características de Amazon Keyspaces (para Apache Cassandra)

En este capítulo se proporcionan detalles sobre cómo trabajar con Amazon Keyspaces y diversas funciones de las bases de datos, por ejemplo, la copia de seguridad y la restauración, el tiempo de vida útil y la replicación multirregional.

- **Período de vida (TTL):** Amazon Keyspaces hace caducar los datos de las tablas automáticamente en función del valor del período de vida que establezca. Aprenda a configurar el TTL y a usarlo en sus tablas.
- **PITR:** proteja sus tablas de Amazon Keyspaces de operaciones accidentales de escritura o eliminación mediante la creación de copias de seguridad continuas de los datos de sus tablas. Aprenda a configurar el PITR en sus tablas y a restaurar una tabla a un punto específico en el tiempo o a restaurar una tabla que se haya eliminado accidentalmente.
- **Trabajar con tablas de varias regiones:** las Multi-Region tablas de Amazon Keyspaces deben tener la capacidad de rendimiento de escritura configurada en modo de capacidad a pedido o aprovisionada con escalado automático. Planifique las necesidades de la capacidad de rendimiento calculando las unidades de capacidad de escritura (WCU) necesarias para cada región y aprovisiona la suma de escrituras de todas las regiones para garantizar una capacidad suficiente para las escrituras replicadas.
- **Captura de datos de cambios de Amazon Keyspaces (CDC):** Amazon Keyspaces CDC transmite eventos de cambio registrados a nivel de fila desde su tabla casi en tiempo real. Aprenda a usar la biblioteca de clientes de Kinesis (KCL) para consumir y procesar datos de las transmisiones de Amazon Keyspaces CDC.
- **Consultas y paginación:** Amazon Keyspaces admite funciones de consulta avanzadas, como el uso del operador `IN` con instrucciones `SELECT`, la ordenación de los resultados con `ORDER BY` y la paginación automática de conjuntos de resultados de gran tamaño. En esta sección se explica cómo Amazon Keyspaces procesa estas consultas y se proporcionan ejemplos.
- **Particionadores:** Amazon Keyspaces proporciona tres particionadores: `Murmur3Partitioner` (predeterminado), `RandomPartitioner` y `DefaultPartitioner`. Puede cambiar el particionador por región a nivel de cuenta mediante el lenguaje de consultas Cassandra (CQL) Consola de administración de AWS o el lenguaje de consultas Cassandra.
- **Client-side marcas de tiempo:** las marcas de Client-side tiempo son marcas de tiempo Cassandra-compatible que Amazon Keyspaces conserva para cada celda de la tabla. Utilice las marcas de

tiempo del cliente para resolver conflictos y permita que su aplicación cliente determine el orden de las escrituras.

- **User-defined tipos (UDT):** con los UDT, puede definir estructuras de datos en sus aplicaciones que representen jerarquías de datos del mundo real.
- **Etiquetado de recursos:** puede etiquetar los recursos de Amazon Keyspaces, como los espacios de claves y las tablas, mediante etiquetas. Las etiquetas ayudan a categorizar los recursos, facilitan el seguimiento de los costos y permiten configurar el control de acceso en función de las etiquetas. En esta sección se describen las restricciones de etiquetado, las operaciones y las prácticas recomendadas para Amazon Keyspaces.
- **CloudFormation plantillas:** le CloudFormation ayuda a modelar y configurar sus tablas y espacios clave de Amazon Keyspaces para que pueda dedicar menos tiempo a crear y administrar sus recursos e infraestructura.

Temas

- [Espacios de claves de sistema en Amazon Keyspaces](#)
- [User-defined tipos \(UDT\) en Amazon Keyspaces](#)
- [Uso de consultas CQL en Amazon Keyspaces](#)
- [Trabajar con transmisiones de captura de datos de cambios \(CDC\) en Amazon Keyspaces](#)
- [Uso de particionadores en Amazon Keyspaces](#)
- [Client-side marcas de tiempo en Amazon Keyspaces](#)
- [Multi-Region replicación para Amazon Keyspaces \(para Apache Cassandra\)](#)
- [Copias de seguridad y restauración de datos con recuperación en un momento dado para Amazon Keyspaces](#)
- [Caducidad de datos con período de vida \(TTL\) para Amazon Keyspaces \(para Apache Cassandra\)](#)
- [Uso de este servicio con un SDK AWS](#)
- [Trabajo con etiquetas para los recursos de Amazon Keyspaces](#)
- [Cree recursos de Amazon Keyspaces con AWS CloudFormation](#)
- [Uso de NoSQL Workbench con Amazon Keyspaces \(para Apache Cassandra\)](#)

Espacios de claves de sistema en Amazon Keyspaces

Los espacios de claves del sistema y las tablas del sistema de Amazon Keyspaces (para Apache Cassandra) son recursos de solo lectura que almacenan metadatos sobre sus recursos de

Amazon Keyspaces. Los espacios clave del sistema están presentes en todos Cuenta de AWS, independientemente de si ha creado espacios clave o tablas. Son una función de compatibilidad con Apache Cassandra y se proporcionan sin coste adicional.

No puede modificar ni eliminar los espacios de claves del sistema. La consola Amazon Keyspaces muestra solo los espacios clave creados por el usuario. Se puede acceder a los espacios clave del sistema mediante programación a través de CQL y aparecen en servicios como y. AWS CloudFormation AWS Config

Amazon Keyspaces utiliza cuatro espacios de claves de sistema:

- `system`
- `system_schema`
- `system_schema_mcs`
- `system_multiregion_info`

En las secciones siguientes se ofrece información en detalle sobre los espacios de claves del sistema y las tablas del sistema compatibles con Amazon Keyspaces.

system

Este es un espacio de claves de Cassandra. Amazon Keyspaces utiliza las siguientes tablas.

Nombres de tablas	Nombres de columnas	Comentarios
<code>local</code>	<code>key</code> , <code>bootstrap</code> , <code>ped</code> , <code>broadcast</code> , <code>_address</code> , <code>cluster_name</code> , <code>cql_version</code> , <code>on</code> , <code>data_center</code> , <code>gossip_generation</code> , <code>host_id</code> , <code>listen_address</code> , <code>native_protocol_version</code> , <code>partitioner</code> , <code>rack</code> , <code>release_version</code> , <code>rpc_addre</code>	Información sobre el espacio de claves local.

Nombres de tablas	Nombres de columnas	Comentarios
	ss, schema_version, thrift_version, tokens, truncated_at	
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 de conexión público, verá una lista de nueve direcciones IP disponibles. Si se conecta a través de un punto de conexión FIPS, verá una lista de tres direcciones IP. Si se conecta a través de un punto final de AWS PrivateLink VPC, verá la lista de direcciones IP que ha 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.

Nombres de tablas	Nombres de columnas	Comentarios
<code>prepared_statements</code>	<code>prepared_id</code> , <code>logged_keyspace</code> , <code>query_string</code>	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
<code>keyspaces</code>	<code>keyspace_name</code> , <code>durable_writes</code> , <code>replication</code>	Información sobre un espacio de claves específico.
<code>tables</code>	<code>keyspace_name</code> , <code>table_name</code> , <code>bloom_filter_fp_chance</code> , <code>caching</code> , <code>comment</code> , <code>compaction</code> , <code>compression</code> , <code>crc_check_chance</code> , <code>dclocal_read_repair_chance</code> , <code>default_time_to_live</code> , <code>extensions</code> , <code>flags</code> , <code>gc_grace_seconds</code> , <code>id</code> , <code>max_index_interval</code> , <code>memtable_flush_period_in_ms</code> , <code>min_index_interval</code> , <code>read_repair_chance</code> , <code>speculative_retry</code>	Información sobre una tabla específica.

Nombres de tablas	Nombres de columnas	Comentarios
types	keyspace_name, type_name, field_names, field_types	Información sobre un tipo específico definido por el usuario (UDT).
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 información, consulte the section called “Comprobación del estado de creación del espacio de claves” .
tables	keyspace_name, creation_time, speculative_retry, cdc, gc_grace_seconds, crc_check_chance, min_index_interval, bloom_filter_fp_chance,	Consulte esta tabla para averiguar el estado de una tabla específica. Para obtener más información, consulte the section called “Comprobación del estado de creación de las tablas” .

Nombres de tablas	Nombres de columnas	Comentarios
	flags, custom_properties, dclocal_read_repair_chance, table_name, caching, default_time_to_live, read_repair_chance, max_index_interval, extensions, compaction, comment, id, compression, memtable_flush_period_in_ms, cdc_specification, latest_stream_arn status	También puede consultar esta tabla para ver una lista de ajustes específicos de Amazon Keyspaces y que se almacenan como custom_properties . Por ejemplo: • capacity_mode • cdc_specification • client_side_timestamps • encryption_specification • point_in_time_recovery • ttl
tables_history	keyspace_name, table_name, event_time, creation_time, custom_properties, event	Consulte esta tabla para obtener información sobre los cambios de esquema para una tabla específica.
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 .

Nombres de tablas	Nombres de columnas	Comentarios
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 “Visualización de etiquetas de tabla” .
types	keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types	Consulte esta tabla para obtener información sobre los tipos definidos por el usuario (UDT). Por ejemplo, puede consultar esta tabla para ver una lista de todas las UDT de un espacio de claves determinado. Para obtener más información, consulte the section called “User-defined tipos (UDT)” .

Nombres de tablas	Nombres de columnas	Comentarios
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 esta configuración no estará disponible hasta que se active la tabla. Para consultar esta tabla, debe especificar <code>keyspace_name</code> y <code>table_name</code> en la cláusula <code>WHERE</code> . Para obtener más información, consulte the section called “Visualización de la configuración de escalado automático de Amazon Keyspaces de la tabla” .

system_multiregion_info

Se trata de un espacio de claves de Amazon Keyspaces que almacena información sobre la replicación multirregional.

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 Regiones de AWS se replica la tabla y su estado. También puede consultar esta tabla para ver una lista de ajustes específicos de Amazon Keyspaces que se

Nombres de tablas	Nombres de columnas	Comentarios
		almacenan como <code>custom_properties</code> . Por ejemplo: • <code>capacity_mode</code> Para consultar esta tabla, debe especificar <code>keyspace_name</code> y <code>table_name</code> en la cláusula WHERE. Para obtener más información, consulte the section called “Creación de un espacio de claves multirregional”.
keyspaces	keyspace_name, region, status, tables_replication_progress	Esta tabla contiene información sobre el progreso de una ALTER KEYSPACE operación que agrega una réplica a un espacio de claves, por ejemplo, cuántas tablas ya se han creado en la nueva región y cuántas tablas siguen en curso. Para ver algunos ejemplos, consulte the section called “Compruebe el progreso de la replicación”.

Nombres de tablas	Nombres de columnas	Comentarios
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 multirregión. Tenga en cuenta que esta configuración no estará disponible hasta que se active la tabla. Para consultar esta tabla, debe especificar <code>keyspace_name</code> y <code>table_name</code> en la cláusula <code>WHERE</code> . Para obtener más información, consulte the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional” .
types	keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types, region	Consulte esta tabla para obtener información sobre los tipos definidos por el usuario (UDT) en espacios de claves multirregionales. Por ejemplo, puede consultar esta tabla para ver una lista de todas las réplicas de tablas y sus AWS regiones respectivas que utilizan las UDT para un espacio de claves determinado. Para obtener más información, consulte the section called “User-defined tipos (UDT)” .

User-defined tipos (UDT) en Amazon Keyspaces

Un tipo definido por el usuario (UDT) es una agrupación de campos y tipos de datos que puede utilizar para definir una sola columna en Amazon Keyspaces. Los tipos de datos válidos para las UDT son todos los tipos de datos de Cassandra compatibles, incluidas las colecciones y otras UDT que ya haya creado en el mismo espacio de claves. Para obtener más información sobre los tipos de datos de Cassandra compatibles, consulte [the section called “Compatibilidad con tipos de datos de Cassandra”](#)

Puede utilizar tipos definidos por el usuario (UDT) en Amazon Keyspaces para organizar los datos de forma más eficiente. Por ejemplo, puede crear UDT con colecciones anidadas, lo que le permite implementar modelos de datos más complejos en sus aplicaciones. También puede utilizar la palabra clave congelada para definir las UDT.

Las UDT están vinculadas a un espacio de claves y están disponibles para todas las tablas y las UDT del mismo espacio de claves. Puede crear las UDT en espacios clave de una o varias regiones.

Puede crear tablas nuevas o modificar las existentes y añadir nuevas columnas que utilicen una UDT. Para crear una UDT con una UDT anidada, la UDT anidada debe estar inmovilizada.

Para ver cuántas UDT se admiten por espacio de claves, los niveles de anidación admitidos y otros valores y cuotas predeterminados relacionados con las UDT, consulte [the section called “Cuotas y valores predeterminados para los tipos definidos por el usuario \(UDT\) en Amazon Keyspaces”](#)

Para obtener información sobre cómo calcular el tamaño codificado de las UDT, consulte [the section called “Calcule el tamaño codificado de los valores de datos en función del tipo de datos”](#)

Para obtener más información sobre la sintaxis CQL, consulte [the section called “Tipos”](#).

Para obtener más información sobre las UDT y la restauración puntual, consulte [the section called “PITR y UDT”](#)

Temas

- [Configurar permisos para trabajar con tipos definidos por el usuario \(UDT\) en Amazon Keyspaces](#)
- [Crear un tipo definido por el usuario \(UDT\) en Amazon Keyspaces](#)
- [Ver tipos definidos por el usuario \(UDT\) en Amazon Keyspaces](#)
- [Eliminar un tipo definido por el usuario \(UDT\) en Amazon Keyspaces](#)

Configurar permisos para trabajar con tipos definidos por el usuario (UDT) en Amazon Keyspaces

Al igual que las tablas, las UDT están vinculadas a un espacio de claves específico. Sin embargo, a diferencia de las tablas, no se pueden definir los permisos directamente para las UDT. Los UDT no se consideran recursos AWS y no tienen identificadores únicos en el formato de un nombre de recurso de Amazon (ARN). En su lugar, para conceder a un director de IAM permisos para realizar acciones específicas en una UDT, hay que definir los permisos para el espacio de claves al que está vinculada la UDT. Para trabajar con las UDT en espacios de claves de varias regiones, se requieren permisos adicionales.

Para poder crear, ver o eliminar las UDT, el director, por ejemplo, un usuario o un rol de IAM, necesita los mismos permisos que se requieren para realizar la misma acción en el espacio de claves al que está vinculada la UDT.

Para obtener más información al respecto, consulte [AWS Identity and Access Management the section called “AWS Identity and Access Management”](#)

Permisos para crear una UDT

Para crear una UDT en un espacio de claves de una sola región, el director necesita `Create` permisos para el espacio de claves.

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Create",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Para crear una UDT en un espacio de claves multirregional, además de los permisos, el director también necesita `Create` permisos para la acción `CreateMultiRegionResource` del espacio de claves especificado.

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "cassandra:Create", "cassandra:CreateMultiRegionResource" ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Permisos para ver una UDT

Para ver o enumerar las UDT en un espacio de claves de una sola región, el director necesita permisos de lectura para el espacio de claves del sistema. Para obtener más información, consulte [the section called “system_schema_mcs”](#).

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Select",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}
```

Para ver o enumerar las UDT de un espacio de claves multirregional, el director necesita permisos para las acciones SELECT y SelectMultiRegionResource para el espacio de claves del sistema. Para obtener más información, consulte [the section called “system_multiregion_info”](#).

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["cassandra:Select", "cassandra:SelectMultiRegionResource"],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}
```

Permisos para eliminar una UDT

Para eliminar una UDT de un espacio de claves de una sola región, el director necesita permisos para realizar la Drop acción en el espacio de claves especificado.

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Drop",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Para eliminar una UDT de un espacio de claves multirregional, el director necesita permisos para la Drop acción y para la DropMultiRegionResource acción del espacio de claves especificado.

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```

    {
      "Effect": "Allow",
      "Action": [ "cassandra:Drop", "cassandra:DropMultiRegionResource" ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
      ]
    }
  ]
}
```

Crear un tipo definido por el usuario (UDT) en Amazon Keyspaces

Para crear una UDT en un espacio de claves de una sola región, puede utilizar la `CREATE TYPE` sentencia de CQL, el `create-type` comando de la consola o la AWS CLI consola.

Los nombres UDT deben contener 48 caracteres o menos, deben empezar por un carácter alfabético y solo pueden contener caracteres alfanuméricos y guiones bajos. Amazon Keyspaces convierte automáticamente los caracteres en mayúsculas en minúsculas.

Como alternativa, puede declarar un nombre UDT entre comillas dobles. Al declarar un nombre UDT entre comillas dobles, Amazon Keyspaces conserva las mayúsculas y permite caracteres especiales.

También puede usar comillas dobles como parte del nombre al crear la UDT, pero debe evitar cada comilla doble con una comilla doble adicional.

En la siguiente tabla se muestran ejemplos de nombres UDT permitidos. La primera columna muestra cómo introducir el nombre al crear el tipo, y la segunda columna muestra cómo Amazon Keyspaces formatea el nombre internamente. Amazon Keyspaces espera el nombre formateado para operaciones como `GetType`.

Nombre introducido	Nombre formateado	Nota
MY_UDT	my_udt	Sin comillas dobles, Amazon Keyspaces convierte todos los caracteres en mayúsculas en minúsculas.
"MY_UDT"	MY_UDT	Con las comillas dobles, Amazon Keyspaces respeta los caracteres en mayúscula y elimina las comillas dobles del nombre formateado.

Nombre introducido	Nombre formateado	Nota
"1234"	1234	Con comillas dobles, el nombre puede empezar por un número y Amazon Keyspaces elimina las comillas dobles del nombre formateado.
"Special_Ch@r@cter s<>!!"	Special_Ch@r@cter s<>!!	Con comillas dobles, el nombre puede contener caracteres especiales y Amazon Keyspaces elimina las comillas dobles del nombre formateado.
"nested""""quote s"	nested""""quotes	Amazon Keyspaces elimina las comillas dobles exteriores y las comillas dobles de escape del nombre formateado.

Console

Cree un tipo definido por el usuario (UDT) con la consola Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Keyspaces y, a continuación, elija un espacio clave de la lista.
3. Seleccione la pestaña UDT.
4. Seleccione Crear UDT
5. En Detalles de la UDT, introduzca el nombre de la UDT. En los campos de la UDT, se define el esquema de la UDT.
6. Para terminar, elija Crear UDT.

Cassandra Query Language (CQL)

Cree un tipo definido por el usuario (UDT) con CQL

En este ejemplo, creamos una nueva versión de la tabla de premios de libros utilizada en. [the section called “Creación de una tabla”](#) En esta tabla, almacenamos todos los premios que recibe

un autor por un libro determinado. Creamos dos UDT que están anidadas y contienen información sobre el libro que ha recibido un premio.

1. Crea un espacio de claves con el nombre. `catalog`

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

2. Cree el primer tipo. Este tipo almacena los códigos BISAC, que se utilizan para definir el género de los libros. Un código BISAC consiste en un código alfanumérico y hasta cuatro áreas temáticas.

```
CREATE TYPE catalog.bisac (  
    bisac_code text,  
    subject1 text,  
    subject2 text,  
    subject3 text,  
    subject4 text  
);
```

3. Cree un segundo tipo para los premios de libros que utilice el primer UDT. La UDT anidada debe congelarse.

```
CREATE TYPE catalog.book (  
    award_title text,  
    book_title text,  
    publication_date date,  
    page_count int,  
    ISBN text,  
    genre FROZEN <bisac>  
);
```

4. Cree una tabla con una columna para el nombre del autor y utilice un tipo de lista para los premios de libros. Tenga en cuenta que la UDT utilizada en la lista debe estar inmovilizada.

```
CREATE TABLE catalog.authors (  
    author_name text PRIMARY KEY,  
    awards list <FROZEN <book>>  
);
```

5. En este paso, insertamos una fila de datos en la nueva tabla.


```
CONSISTENCY LOCAL_QUORUM;
```

```
INSERT INTO catalog.authors (author_name, awards) VALUES (
'John Stiles' ,
[
  {
    award_title: 'Wolf',
    book_title: 'Yesterday',
    publication_date: '2020-10-10',
    page_count: 345,
    ISBN: '026204630X',
    genre: { bisac_code: 'FIC014090', subject1: 'FICTION', subject2:
'Historical', subject3: '20th Century', subject4: 'Post-World War II' }
  },
  {
    award_title: 'Richard Roe',
    book_title: 'Who ate the cake?',
    publication_date: '2019-05-13',
    page_count: 193,
    ISBN: '9780262046305',
    genre: { bisac_code: 'FIC022130', subject1: 'FICTION', subject2: 'Mystery &
Detective', subject3: 'Cozy', subject4: 'Culinary' }
  }
]);
```

6. En el último paso leemos los datos de la tabla.

```
SELECT * FROM catalog.authors;
```

El resultado del comando debería tener este aspecto.

```
author_name | awards
-----
+-----+
John Stiles | [{award_title: 'Wolf', book_title: 'Yesterday', publication_date:
2020-10-10, page_count: 345, isbn: '026204630X', genre: {bisac_code:
'FIC014090', subject1: 'FICTION', subject2: 'Historical', subject3: '20th
Century', subject4: 'Post-World War II'}}, {award_title: 'Richard Roe',
book_title: 'Who ate the cake?', publication_date: 2019-05-13, page_count: 193,
isbn: '9780262046305', genre: {bisac_code: 'FIC022130', subject1: 'FICTION',
subject2: 'Mystery & Detective', subject3: 'Cozy', subject4: 'Culinary'}}]
```

(1 rows)

Para obtener más información sobre la sintaxis CQL, consulte [the section called “CREATE TYPE”](#).

CLI

Cree un tipo definido por el usuario (UDT) con el AWS CLI

1. Para crear un tipo, puede utilizar la siguiente sintaxis.

```
aws keyspaces create-type
--keyspace-name 'my_keyspace'
--type-name 'my_udt'
--field-definitions
  '['
    {"name" : "field1", "type" : "int"},
    {"name" : "field2", "type" : "text"}
  ']
```

2. El resultado de ese comando es similar al de este ejemplo. Tenga en cuenta que typeName devuelve el nombre formateado de la UDT.

```
{
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
  "typeName": "my_udt"
}
```

Ver tipos definidos por el usuario (UDT) en Amazon Keyspaces

Para ver o enumerar todas las UDT de un espacio de claves de una sola región, puede consultar la tabla `system_schema_mcs.types` en el espacio de claves del sistema mediante una sentencia en CQL o utilizar los `list-type` comandos `get-type` y con la consola o con la consola. AWS CLI

Para cualquiera de las dos opciones, el director de IAM necesita permisos de lectura en el espacio de claves del sistema. Para obtener más información, consulte [the section called “Configuración de permisos”](#).

Console

Vea los tipos definidos por el usuario (UDT) con la consola Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Keyspaces y, a continuación, elija un espacio clave de la lista.
3. Seleccione la pestaña UDT para revisar la lista de todas las UDT del espacio clave.
4. Para revisar una UDT en detalle, elija una UDT de la lista.
5. En la pestaña Esquema, puede revisar el esquema. En la pestaña Usado en, puede ver si esta UDT se usa en tablas u otras UDT. Tenga en cuenta que solo puede eliminar las UDT que no estén siendo utilizadas por las tablas u otras UDT.

Cassandra Query Language (CQL)

Vea los tipos definidos por el usuario (UDT) de un espacio de claves de una sola región con CQL

1. Para ver los tipos que están disponibles en un espacio de claves determinado, puede utilizar la siguiente instrucción.

```
SELECT type_name
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace';
```

2. Para ver los detalles de un tipo específico, puede usar la siguiente declaración.

```
SELECT
    keyspace_name,
    type_name,
    field_names,
    field_types,
    max_nesting_depth,
    last_modified_timestamp,
    status,
    direct_referring_tables,
    direct_parent_types
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace' AND type_name = 'my_udt';
```

3. Puede enumerar todas las UDT que existen en la cuenta utilizando `DESC TYPE`.

```
DESC TYPES;

Keyspace my_keyspace
-----
my_udt1  my_udt2

Keyspace my_keyspace2
-----
my_udt1
```

4. Puede enumerar todas las UDT del espacio de claves actualmente seleccionado utilizando `DESC TYPE`

```
USE my_keyspace;
my_keyspace DESC TYPES;

my_udt1  my_udt2
```

5. Para enumerar todas las UDT de un espacio de claves multirregional, puede consultar la tabla del sistema en el espacio de claves. `types system_multiregion_info` La siguiente consulta es un ejemplo de ello.

```
SELECT keyspace_name, type_name, region, status FROM
system_multiregion_info.types WHERE keyspace_name = 'mykeyspace' AND table_name
= 'mytable';
```

El resultado de este comando tiene un aspecto similar al siguiente.

keyspace_name	table_name	region	status
mykeyspace	mytable	us-east-1	ACTIVE
mykeyspace	mytable	ap-southeast-1	ACTIVE
mykeyspace	mytable	eu-west-1	ACTIVE

CLI

Vea los tipos definidos por el usuario (UDT) con la AWS CLI

1. Para enumerar los tipos disponibles en un espacio de claves, puede usar el comando. `list-types`

```
aws keyspaces list-types
--keyspace-name 'my_keyspace'
```

El resultado de ese comando tiene un aspecto similar al de este ejemplo.

```
{
  "types": [
    "my_udt",
    "parent_udt"
  ]
}
```

2. Para ver los detalles de un tipo determinado, puede usar el `get-type` comando.

```
aws keyspaces get-type
--type-name 'my_udt'
--keyspace-name 'my_keyspace'
```

El resultado de este comando es similar al de este ejemplo.

```
{
  "keyspaceName": "my_keyspace",
  "typeName": "my_udt",
  "fieldDefinitions": [
    {
      "name": "a",
      "type": "int"
    },
    {
      "name": "b",
      "type": "text"
    }
  ],
  "lastModifiedTimestamp": 1721328225776,
```

```
"maxNestingDepth": 3
"status": "ACTIVE",
"directReferringTables": [],
"directParentTypes": [
  "parent_udt"
],
"keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/"
}
```

Eliminar un tipo definido por el usuario (UDT) en Amazon Keyspaces

Para eliminar una UDT de un espacio de claves, puede utilizar la `DROP TYPE` sentencia de CQL, el `delete-type` comando de o la consola AWS CLI.

Console

Eliminar un tipo definido por el usuario (UDT) con la consola Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Keyspaces y, a continuación, elija un espacio clave de la lista.
3. Seleccione la pestaña UDT.
4. Elija la UDT que desee eliminar. En Usado en, puede confirmar que el tipo que desea eliminar no lo utiliza actualmente ninguna tabla u otra UDT.
5. Selecciona Eliminar encima del resumen.
6. Escriba `Delete` en el cuadro de diálogo que aparece y elija Eliminar UDT.

Cassandra Query Language (CQL)

Elimine un tipo definido por el usuario (UDT) con CQL

- Para eliminar un tipo, puede utilizar la siguiente sentencia.

```
DROP TYPE my_keyspace.my_udt;
```

Para obtener más información sobre la sintaxis CQL, consulte [the section called “DROP TYPE”](#).

CLI

Elimine un tipo definido por el usuario (UDT) con la AWS CLI

1. Para eliminar un tipo, puede utilizar el siguiente comando.

```
aws keyspaces delete-type
--keyspace-name 'my_keyspace'
--type-name 'my_udt'
```

2. El resultado del comando es similar al de este ejemplo.

```
{
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
  "typeName": "my_udt"
}
```

Uso de consultas CQL 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 una consulta en Amazon Keyspaces](#)
- [Usa declaraciones por lotes en Amazon Keyspaces](#)
- [Ordenación de resultados con ORDER BY en Amazon Keyspaces](#)
- [Paginación de resultados en Amazon Keyspaces](#)

Uso del operador **IN** con la instrucción **SELECT** en una consulta 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

Usa declaraciones por lotes en Amazon Keyspaces

Puede combinar varias INSERT DELETE operaciones UPDATE y en una BATCH declaración. LOGGED los lotes son los predeterminados.

```
batch_statement ::=      BEGIN [ UNLOGGED ] BATCH
                        [ USING update_parameter( AND update_parameter)* ]
                        modification_statement ( ';' modification_statement )*
                        APPLY BATCH
modification_statement ::= insert_statement | update_statement | delete_statement
```

Al ejecutar una declaración de lote, el conductor combina todas las declaraciones del lote en una sola operación de lote.

Para decidir qué tipo de operación por lotes utilizar, puede tener en cuenta las siguientes pautas.

Utilice lotes registrados cuando:

- Necesita garantías de transacciones atómicas.
- Las latencias ligeramente más altas son una compensación aceptable.

Utilice lotes no registrados cuando:

- Necesita optimizar las operaciones de una sola partición.
- Desea reducir la sobrecarga de la red.
- Tiene requisitos de alto rendimiento.

Para obtener información sobre las cuotas de extractos de lotes, consulte. [Cuotas](#)

Lotes no registrados

Con los lotes no registrados, Amazon Keyspaces procesa varias operaciones como una sola solicitud sin mantener un registro de lotes. Con una operación por lotes sin registrar, es posible que algunas

de las acciones se realicen correctamente y que otras no. Los lotes no registrados son útiles cuando se quiere:

- Optimice las operaciones dentro de una sola partición.
- Reduzca el tráfico de red agrupando las solicitudes relacionadas.

La sintaxis de un lote no registrado es similar a la de un lote registrado, con la adición de la `UNLOGGED` palabra clave.

```
BEGIN UNLOGGED BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Lotes registrados

Un lote registrado combina múltiples acciones de escritura en una sola operación atómica. Al ejecutar un lote registrado:

- Todas las acciones se realizan correctamente juntas o fallan juntas.
- La operación es sincrónica e idempotente.
- Puedes escribir en varias tablas de Amazon Keyspaces, siempre y cuando estén en la misma AWS cuenta y. Región de AWS

Los lotes registrados pueden tener latencias ligeramente superiores. Para aplicaciones de alto rendimiento, considere la posibilidad de utilizar lotes no registrados.

El uso de lotes registrados en Amazon Keyspaces no conlleva ningún coste adicional. Solo paga por las escrituras que forman parte de sus operaciones por lotes. Amazon Keyspaces realiza dos escrituras subyacentes en cada fila del lote: una para preparar la fila para el lote y otra para confirmar el lote. Al planificar la capacidad de las tablas que utilizan lotes registrados, recuerde que cada fila de un lote requiere el doble de capacidad que una operación de escritura estándar. Por ejemplo, si su aplicación ejecuta un lote registrado por segundo con tres filas de 1 KB, necesitará aprovisionar seis unidades de capacidad de escritura (WCU), en lugar de solo tres WCU para escrituras individuales o lotes no registrados.

Para obtener información sobre precios, consulte [Precios de Amazon Keyspaces \(para Apache Cassandra\)](#).

Prácticas recomendadas para las operaciones por lotes

Tenga en cuenta las siguientes prácticas recomendadas al utilizar las operaciones por lotes de Amazon Keyspaces.

- Habilite el escalado automático para que sus tablas puedan gestionar las operaciones por lotes y los requisitos de rendimiento adicionales de los lotes registrados.
- Utilice operaciones individuales o lotes no registrados cuando las operaciones se puedan ejecutar de forma independiente sin que ello afecte a la corrección de la aplicación.
- Diseñe su aplicación para minimizar las actualizaciones simultáneas en las mismas filas, ya que las operaciones por lotes simultáneas pueden entrar en conflicto y fallar.
- Para una ingesta masiva de datos de alto rendimiento sin requisitos de atomicidad, utilice operaciones de escritura individuales o lotes no registrados.

Coherencia y simultaneidad

Amazon Keyspaces aplica las siguientes reglas de coherencia y simultaneidad para los lotes registrados:

- Todas las operaciones por lotes utilizan el nivel de coherencia. LOCAL_QUORUM
- Los lotes simultáneos que afectan a diferentes filas se pueden ejecutar simultáneamente.
- DELETELas operaciones INSERT simultáneas UPDATE o en las filas implicadas en un lote en curso fallan y se produce un conflicto.

Operadores y condiciones compatibles

Operadores de WHERE cláusulas compatibles:

- Igualdad (=)

Operadores no compatibles:

- Operadores de rango (>, <, >=, <=)
- operador IN
- operador LIKE
- operador BETWEEN

No se admiten en lotes registrados:

- Varias declaraciones que afectan a la misma fila

- Operaciones de contraataque
- Eliminaciones de rango

Condiciones de error de las declaraciones de lotes registradas

Una operación por lotes registrada puede fallar en cualquiera de los siguientes casos:

- Las expresiones de condición (como `IF NOT EXISTS` o `IF`) se evalúan como falsas.
- Una o más operaciones contienen parámetros no válidos.
- La solicitud entra en conflicto con otra operación por lotes que se ejecuta en las mismas filas.
- La tabla carece de suficiente capacidad aprovisionada.
- Una fila supera el límite de tamaño máximo.
- El formato de datos de entrada no es válido.

Declaraciones por lotes y replicación multirregional

En despliegues multirregionales:

- Las operaciones de la región de origen son sincrónicas y atómicas.
- Las operaciones de la región de destino son asíncronas.
- Todas las operaciones por lotes se replican en las regiones de destino, pero es posible que no mantengan el aislamiento durante la aplicación.

Supervise las operaciones por lotes

Puedes supervisar las operaciones por lotes mediante las CloudWatch métricas de Amazon para hacer un seguimiento del rendimiento, los errores y los patrones de uso. Amazon Keyspaces proporciona las siguientes CloudWatch métricas para supervisar las operaciones por lotes por tabla:

- `SuccessfulRequestCount`— Realice un seguimiento de las operaciones por lotes exitosas.
- `Latency`— Mida el rendimiento de las operaciones por lotes.
- `ConsumedWriteCapacityUnits`— Supervise el consumo de capacidad de las operaciones por lotes.

Para obtener más información, consulte [the section called “Métricas de Amazon Keyspaces”](#).

Además de las CloudWatch métricas, puede utilizarlas AWS CloudTrail para registrar todas las acciones de la API de Amazon Keyspaces. Cada acción de la API del lote se registra, lo CloudTrail que facilita el seguimiento y la auditoría de las operaciones por lotes en las tablas de Amazon Keyspaces.

Ejemplos de operaciones por lotes

A continuación se muestra un ejemplo de una sentencia básica por lotes registrados.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Este es un ejemplo de un lote que incluye INSERTUPDATE, y DELETE declaraciones.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
  UPDATE users SET firstname = 'Johnny' WHERE id = 2;
  DELETE FROM users WHERE id = 3;
APPLY BATCH;
```

Este es un ejemplo de un lote que utiliza marcas de tiempo del lado del cliente.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Stiles') USING
  TIMESTAMP 1669069624;
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Doe') USING
  TIMESTAMP 1669069624;
APPLY BATCH;

BEGIN BATCH
  UPDATE users USING TIMESTAMP 1669069624 SET firstname = 'Carlos' WHERE id = 1;
  UPDATE users USING TIMESTAMP 1669069624 SET firstname = 'Diego' WHERE id = 2;
APPLY BATCH;
```

Este es un ejemplo de lote condicional.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'Jane', 'Doe') IF NOT
  EXISTS;
```



```
INSERT INTO users (id, firstname, lastname) VALUES (2, 'John', 'Doe') IF NOT
EXISTS;
APPLY BATCH;

BEGIN BATCH
  UPDATE users SET lastname = 'Stiles' WHERE id = 1 IF lastname = 'Doe';
  UPDATE users SET lastname = 'Stiles' WHERE id = 2 IF lastname = 'Doe';
APPLY BATCH;
```

Este es un ejemplo de un lote que utiliza Time to Live (TTL).

```
BEGIN BATCH
  INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe') USING TTL
  3600;
  INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith') USING TTL
  7200;
APPLY BATCH;
```

Este es un ejemplo de una sentencia por lotes que actualiza varias tablas.

```
BEGIN BATCH
  INSERT INTO users (id, firstname) VALUES (1, 'John');
  INSERT INTO user_emails (user_id, email) VALUES (1, 'john@example.com');
APPLY BATCH;
```

Este es un ejemplo de una operación por lotes que utiliza tipos definidos por el usuario (UDT). En el ejemplo se supone que existe la address UDT.

```
BEGIN BATCH
  INSERT INTO users (id, firstname, address)
  VALUES (1, 'John', {street: '123 Main St', city: 'NYC', zip: '10001'});
  INSERT INTO users (id, firstname, address)
  VALUES (2, 'Jane', {street: '456 Oak Ave', city: 'LA', zip: '90210'});
APPLY BATCH;

BEGIN BATCH
  UPDATE users SET address.zip = '10002' WHERE id = 1;
  UPDATE users SET address.city = 'Boston' WHERE id = 2;
APPLY BATCH;
```

Ordenación de resultados con **ORDER BY** 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. Para ver un ejemplo de cómo el `PAGE SIZE` de las consultas `LIMIT` puede afectar a la capacidad de lectura, consulte [the section called “Estimación del consumo de capacidad de lectura de las consultas de límites”](#).

Para ver una comparación con la paginación de Apache Cassandra, consulte [the section called “Paginación”](#).

Trabajar con transmisiones de captura de datos de cambios (CDC) en Amazon Keyspaces

La captura de datos de cambios de Amazon Keyspaces (CDC) registra los eventos de cambio a nivel de fila de una tabla de Amazon Keyspaces prácticamente en tiempo real.

Amazon Keyspaces CDC permite casos de uso basados en eventos, como el IoT industrial y la detección de fraudes, así como casos de uso de procesamiento de datos, como la búsqueda de texto completo y el archivo de datos. Los eventos de cambio que los CDC de Amazon Keyspaces capturan en las transmisiones pueden ser consumidos por aplicaciones posteriores que realizan funciones esenciales para la empresa, como el análisis de datos, la búsqueda de texto, el aprendizaje automático training/inference y las copias de seguridad continuas de los datos para archivarlos. Por ejemplo, puede transferir los datos de transmisión a servicios de AWS análisis y almacenamiento como Amazon OpenSearch Service, Amazon Redshift y Amazon S3 para su posterior procesamiento.

Amazon Keyspaces CDC ofrece registros de cambios ordenados por tiempo y deduplicados para tablas, con un escalado automático del rendimiento de los datos y un tiempo de retención de hasta 24 horas.

Las transmisiones de CDC de Amazon Keyspaces no tienen ningún servidor y no es necesario administrar la infraestructura de datos para capturar los eventos de cambio. Además, Amazon Keyspaces CDC no consume ninguna capacidad de mesa ni para cómputo ni para almacenamiento. Para obtener más información, consulte [the section called “Cómo funcionan las transmisiones de los CDC”](#).

Puede utilizar la API Amazon Keyspaces Streams para crear aplicaciones que consuman las transmisiones de Amazon Keyspaces CDC y tomar medidas en función del contenido. Para ver los puntos de enlace disponibles, consulte [the section called “Acceda a los puntos de enlace de transmisión de los CDC”](#)

Para obtener una lista completa de todas las operaciones disponibles para Amazon Keyspaces en la API de Streams, consulte la referencia de la API de Amazon [Keyspaces](#) Streams.

Temas

- [Cómo funcionan las transmisiones de captura de datos de cambios \(CDC\) en Amazon Keyspaces](#)
- [Cómo utilizar las transmisiones de captura de datos de cambios \(CDC\) en Amazon Keyspaces](#)

Cómo funcionan las transmisiones de captura de datos de cambios (CDC) en Amazon Keyspaces

En esta sección se proporciona información general sobre cómo funcionan las transmisiones de captura de datos de cambios (CDC) en Amazon Keyspaces.

La captura de datos de cambios de Amazon Keyspaces (CDC) registra una secuencia ordenada de modificaciones a nivel de fila en las tablas de Amazon Keyspaces y almacena esta información en un registro denominado stream durante un máximo de 24 horas. Cada modificación a nivel de fila genera un nuevo registro CDC que contiene la información de la columna clave principal, así como los estados «antes» y «después» de la fila, incluidas todas las columnas. Las aplicaciones pueden acceder a la transmisión y ver las mutaciones prácticamente en tiempo real.

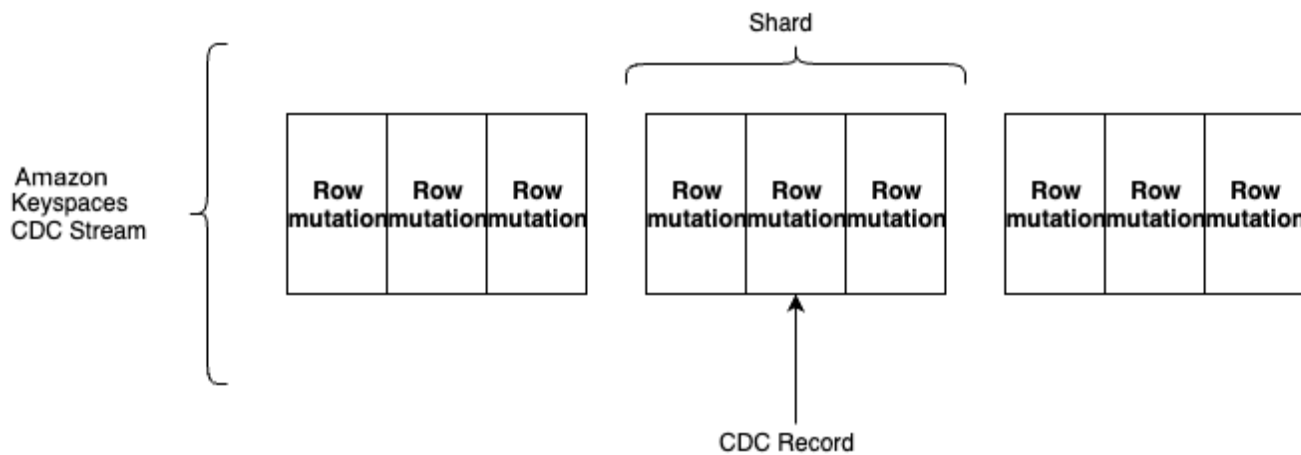
Cuando habilitas los CDC en tu mesa, Amazon Keyspaces crea una nueva transmisión de CDC y comienza a capturar información sobre cada modificación de la tabla. La transmisión de los CDC tiene un nombre de recurso de Amazon (ARN) con el siguiente formato:

```
arn:${Partition}:cassandra:{Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

Puede seleccionar el tipo de información o el tipo de vista que recopila la transmisión de los CDC para cada registro al habilitar la transmisión de los CDC por primera vez. No puede cambiar el tipo de vista de la transmisión posteriormente. Amazon Keyspaces admite los siguientes tipos de vistas:

- **NEW_AND_OLD_IMAGES**— Captura las versiones de la fila antes y después de la mutación. Esta es la opción predeterminada.
- **NEW_IMAGE**— Captura la versión de la fila después de la mutación.
- **OLD_IMAGE**— Captura la versión de la fila anterior a la mutación.
- **KEYS_ONLY**— Captura las claves de partición y agrupamiento de la fila que se ha mutado.

Cada transmisión de los CDC consta de registros. Cada registro representa una modificación de una sola fila en una tabla de Amazon Keyspaces. Los registros se organizan de forma lógica en grupos conocidos como fragmentos. Estos grupos están organizados de forma lógica por rangos de la clave principal (combinación de clave de partición y rangos de claves de agrupamiento) y son una construcción interna de Amazon Keyspaces. Cada fragmento actúa como un contenedor de varios registros y contiene la información necesaria para acceder a estos registros y recorrerlos.



A cada registro de los CDC se le asigna un número de secuencia que refleja el orden en que se publicó el registro en el fragmento. Se garantiza que el número de secuencia será creciente y único dentro de cada fragmento.

Amazon Keyspaces crea y elimina fragmentos automáticamente. En función de las cargas de tráfico, Amazon Keyspaces también puede dividir o fusionar fragmentos a lo largo del tiempo. Por ejemplo, Amazon Keyspaces puede dividir un fragmento en varios fragmentos nuevos o fusionar fragmentos en un solo fragmento nuevo. Las API de Amazon Keyspaces publican la información del fragmento y los CDC transmiten la información para permitir que las aplicaciones consumidoras procesen los registros en el orden correcto al acceder a todo el gráfico de linaje de un fragmento.

Amazon Keyspaces CDC se basa en los siguientes principios en los que puede confiar a la hora de crear su aplicación:

- Cada registro de mutaciones a nivel de fila aparece exactamente una vez en la secuencia de los CDC.
- Al consumir fragmentos por orden de linaje, cada registro de mutación a nivel de fila aparece en la misma secuencia que el orden de mutación real en la clave principal.

Temas

- [Cómo funciona la retención de datos para las transmisiones de los CDC en Amazon Keyspaces](#)
- [Cómo funciona la caducidad de los datos de Time to Live \(TTL\) con las transmisiones de los CDC en Amazon Keyspaces](#)
- [Cómo funcionan las operaciones por lotes para las transmisiones de CDC en Amazon Keyspaces](#)
- [Cómo funcionan las columnas estáticas en las transmisiones de CDC en Amazon Keyspaces](#)

- [Cómo funciona el cifrado en reposo para las transmisiones de los CDC en Amazon Keyspaces](#)
- [Cómo funciona la replicación multirregional para las transmisiones de CDC en Amazon Keyspaces](#)
- [Los CDC transmiten e integran los servicios AWS](#)

Cómo funciona la retención de datos para las transmisiones de los CDC en Amazon Keyspaces

Amazon Keyspaces conserva los registros de la transmisión de los CDC durante un período de 24 horas. No puede cambiar el período de retención. Si desactivas los CDC en una tabla, los datos de la transmisión seguirán siendo legibles durante 24 horas. Transcurrido este tiempo, los datos caducan y los registros se eliminan automáticamente.

Cómo funciona la caducidad de los datos de Time to Live (TTL) con las transmisiones de los CDC en Amazon Keyspaces

Amazon Keyspaces muestra el tiempo de caducidad tanto a column/cell nivel como a nivel de fila en un campo de metadatos denominado `expirationTime` en los registros de cambios de los CDC. Cuando el TTL de Amazon Keyspaces detecta la caducidad de una celda, los CDC crean un nuevo registro de cambios que muestra el TTL como el origen del cambio. Para obtener más información sobre TTL, consulte [the section called “Caducidad de datos con período de vida”](#).

Cómo funcionan las operaciones por lotes para las transmisiones de CDC en Amazon Keyspaces

Las operaciones por lotes se dividen internamente en modificaciones individuales a nivel de fila. Amazon Keyspaces conserva todos los registros de las transmisiones de CDC a nivel de fila, incluso si la modificación se produjo en una operación por lotes. Amazon Keyspaces mantiene el orden de los registros de la transmisión de CDC en la misma secuencia que el orden de mutación que se produjo en el nivel de fila o en la clave principal.

Cómo funcionan las columnas estáticas en las transmisiones de CDC en Amazon Keyspaces

Los valores de las columnas estáticas se comparten entre todas las filas de una partición en Cassandra. Debido a este comportamiento, Amazon Keyspaces captura cualquier actualización de una columna estática como un registro independiente en la transmisión de CDC. Los siguientes ejemplos resumen el comportamiento de las mutaciones en las columnas estáticas:

- Cuando solo se actualiza la columna estática, el flujo CDC contiene una modificación de fila para la columna estática como única columna de la fila.
- Cuando una fila se actualiza sin ningún cambio en la columna estática, la secuencia CDC contiene una modificación de fila que contiene todas las columnas excepto la columna estática.
- Cuando se actualiza una fila junto con la columna estática, el flujo CDC contiene dos modificaciones de fila independientes, una para la columna estática y otra para el resto de la fila.

Cómo funciona el cifrado en reposo para las transmisiones de los CDC en Amazon Keyspaces

Para cifrar los datos en reposo en el registro de pedidos de los CDC, Amazon Keyspaces utiliza la misma clave de cifrado que ya se utiliza para la tabla. Para obtener más información sobre el cifrado en reposo, consulte [the section called “Cifrado en reposo”](#).

Cómo funciona la replicación multirregional para las transmisiones de CDC en Amazon Keyspaces

Puede activar y desactivar las transmisiones de CDC para réplicas individuales de una tabla multirregional mediante la `update-table` API o el comando CQL. `ALTER TABLE` Debido a la replicación asíncrona y a la resolución de conflictos, las transmisiones de los CDC para las tablas multirregionales no son consistentes en todas las tablas. Regiones de AWS Por lo tanto, los registros que Amazon Keyspaces captura en la transmisión pueden aparecer en un orden diferente en distintas regiones.

Para obtener más información sobre la replicación multirregional, consulte. [the section called “Multi-Region replicación”](#)

Los CDC transmiten e integran los servicios AWS

Cómo trabajar con puntos de enlace de VPC para transmisiones de CDC en Amazon Keyspaces

Puede usar puntos de enlace de VPC para acceder a las transmisiones de los CDC de Amazon Keyspaces. Para obtener información sobre cómo crear puntos de enlace de VPC para transmisiones y acceder a ellos, consulte. [the section called “Uso de puntos finales de VPC de interfaz con transmisiones”](#)

Cómo CloudWatch funciona la supervisión de las transmisiones de los CDC en Amazon Keyspaces

Puedes usar Amazon CloudWatch para monitorear las llamadas a la API realizadas al punto de conexión CDC de Amazon Keyspaces. Para obtener más información sobre las métricas disponibles, consulte [the section called “Las métricas de Amazon Keyspaces cambian la captura de datos \(CDC\)”](#).

Cómo CloudTrail funciona el registro con las transmisiones de los CDC en Amazon Keyspaces

Amazon Keyspaces CDC está integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio 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 (DML) 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.

Para obtener más información sobre los eventos de los CDC capturados por CloudTrail, consulte. [the section called “Iniciar sesión con CloudTrail”](#)

Cómo funciona el etiquetado para las transmisiones de los CDC en Amazon Keyspaces

Las transmisiones de los CDC de Amazon Keyspaces son un recurso que se puede etiquetar. Puede etiquetar una transmisión al crear una tabla mediante programación mediante CQL, el SDK o el AWS . AWS CLI También puede etiquetar transmisiones existentes, eliminar etiquetas o ver las etiquetas de una transmisión. Para obtener más información, consulte [the section called “Etiquete los espacios clave, las tablas y las secuencias”](#).

Cómo utilizar las transmisiones de captura de datos de cambios (CDC) en Amazon Keyspaces

Temas

- [Configure los permisos para trabajar con las transmisiones de los CDC en Amazon Keyspaces](#)
- [Cómo acceder a los puntos de enlace de transmisión de los CDC en Amazon Keyspaces](#)
- [Habilita una transmisión de CDC al crear una tabla nueva en Amazon Keyspaces](#)
- [Habilitar una transmisión de CDC para una tabla existente en Amazon Keyspaces](#)
- [Inhabilitar una transmisión de los CDC en Amazon Keyspaces](#)
- [Vea las transmisiones de los CDC en Amazon Keyspaces](#)
- [Acceda a los registros de las transmisiones de los CDC en Amazon Keyspaces](#)

- [Utilice la biblioteca de clientes de Kinesis \(KCL\) para procesar las transmisiones de Amazon Keyspaces](#)

Configure los permisos para trabajar con las transmisiones de los CDC en Amazon Keyspaces

Para habilitar las transmisiones de los CDC, el director, por ejemplo, un usuario o rol de IAM, necesita los siguientes permisos.

Para obtener más información al respecto AWS Identity and Access Management, consulte [the section called “AWS Identity and Access Management”](#).

Permisos para habilitar una transmisión de los CDC para una tabla

Para habilitar una transmisión de CDC para una tabla de Amazon Keyspaces, el director primero necesita permisos para crear o modificar una tabla y, en segundo lugar, los permisos para crear el rol vinculado al servicio. [AWSServiceRoleForAmazonKeyspacesCDC](#) Amazon Keyspaces utiliza el rol vinculado al servicio para publicar CloudWatch métricas en su cuenta en su nombre.

La siguiente política de IAM es un ejemplo de ello.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Create",
        "cassandra:CreateMultiRegionResource",
        "cassandra:Alter",
        "cassandra:AlterMultiRegionResource"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/*",
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    },
    {
      "Sid": "KeyspacesCDCServiceLinkedRole",
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
```

```

    "Resource": "arn:aws:iam::*:role/aws-service-role/cassandra-
streams.amazonaws.com/AWSServiceRoleForAmazonKeyspacesCDC",
    "Condition": {
      "StringLike": {
        "iam:AWSServiceName": "cassandra-streams.amazonaws.com"
      }
    }
  }
]
}

```

Para deshabilitar una transmisión, solo se requieren ALTER TABLE permisos.

Permisos para ver una transmisión de los CDC

Para ver o enumerar las transmisiones de los CDC, el director necesita permisos de lectura para el espacio de claves del sistema. Para obtener más información, consulte [the section called “system_schema_mcs”](#).

La siguiente política de IAM es un ejemplo de ello.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cassandra:Select",
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}

```

Para ver o enumerar las transmisiones de los AWS CLI CDC con la API de Amazon Keyspaces, el director necesita permisos adicionales para las acciones `cassandra:ListStreams` y `cassandra:GetStream`.

La siguiente política de IAM es un ejemplo de ello.

```

{
  "Version": "2012-10-17",

```

```

"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "cassandra:Select",
      "cassandra:ListStreams",
      "cassandra:GetStream"
    ],
    "Resource": "*"
  }
]
}

```

Permisos para leer una transmisión de los CDC

Para leer las transmisiones de los CDC, el director necesita los siguientes permisos.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:GetStream",
        "cassandra:GetShardIterator",
        "cassandra:GetRecords"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/stream_label"
      ]
    }
  ]
}

```

Permisos para procesar las transmisiones de Amazon Keyspaces CDC con la biblioteca de clientes de Kinesis (KCL)

Para procesar las transmisiones de CDC de Amazon Keyspaces con KCL, el director de IAM necesita los siguientes permisos.

- Amazon Keyspaces— Read-only acceso a una transmisión CDC específica de Amazon Keyspaces.

- **DynamoDB— Permisos para crear shard lease tablas, acceso de lectura y escritura a las tablas y acceso de lectura al índice, según sea necesario para el procesamiento de la transmisión de KCL.**
- **CloudWatch— Permisos para publicar datos de métricas de Amazon Keyspaces** Los CDC transmiten el procesamiento con KCL al espacio de nombres de la aplicación cliente de KCL de su cuenta. CloudWatch Para obtener más información sobre la supervisión, consulte [Supervisar la biblioteca de clientes de Kinesis con Amazon](#). CloudWatch

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:GetStream",
        "cassandra:GetShardIterator",
        "cassandra:GetRecords"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/stream_label"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:CreateTable",
        "dynamodb:DescribeTable",
        "dynamodb:UpdateTable",
        "dynamodb:GetItem",
        "dynamodb:UpdateItem",
        "dynamodb:PutItem",
        "dynamodb>DeleteItem",
        "dynamodb:Scan"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME"
      ]
    },
    {
      "Effect": "Allow",
```

```

    "Action":[
        "dynamodb:CreateTable",
        "dynamodb:DescribeTable",
        "dynamodb:GetItem",
        "dynamodb:UpdateItem",
        "dynamodb:PutItem",
        "dynamodb>DeleteItem",
        "dynamodb:Scan"
    ],
    "Resource":[
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME-WorkerMetricStats",
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME-CoordinatorState"
    ]
},
{
    "Effect":"Allow",
    "Action":[
        "dynamodb:Query"
    ],
    "Resource":[
        "arn:aws:dynamodb:us-east-1:111122223333:table/KCL_APPLICATION_NAME/index/
* "
    ]
},
{
    "Effect":"Allow",
    "Action":[
        "cloudwatch:PutMetricData"
    ],
    "Resource":"*"
}
]
}

```

Cómo acceder a los puntos de enlace de transmisión de los CDC en Amazon Keyspaces

Amazon Keyspaces mantiene [puntos de enlace](#) separados para keyspaces/tables y para las transmisiones de CDC en cada uno de los lugares en los que Región de AWS Amazon Keyspaces está disponible. Para acceder a una transmisión de los CDC, seleccione la región de la tabla y

sustituya el `cassandra` prefijo por el nombre del punto final, como se muestra `cassandra-streams` en el siguiente ejemplo:

Nombre de la región	Region	Punto final para espacios clave y tablas	Punto final para transmisiones
Este de EE. UU. (Norte de Virginia)	us-east-1	<i>cassandra</i> .us-east-1.api.aws	<i>cassandra-streams</i> .us-east-1.api.aws

La siguiente tabla contiene una lista completa de los puntos finales públicos disponibles para. Amazon Keyspaces change data capture streams Amazon Keyspaces CDC streams admite IPv4 e IPv6. Todos los puntos finales públicos, por ejemplo `cassandra-streams.us-east-1.api.aws`, son puntos finales de doble pila que se pueden configurar para IPv4 e IPv6.

Nombre de la región	Región	Punto de conexión	Protocolo	
Este de EE. UU. (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS	
Este de EE. UU. (Norte de Virginia)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
Oeste de EE. UU. (Norte de California)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Oeste de EE. UU. (Oregón)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
África (Ciudad del Cabo)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	
Asia-Pacífico (Hong Kong)	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	
Asia-Pacífico (Malasia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Asia-Pacífico (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	
Asia-Pacífico (Seúl)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	
Asia-Pacífico (Singapur)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	
Asia-Pacífico (Sídney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Asia-Pacífico (Tailandia)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS	
Asia-Pacífico (Tokio)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS	
Canadá (centro)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS	
Europa (Fráncfort)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS	
Europa (Irlanda)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS	
Europa (Londres)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS	
Europa (París)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	
Europa (Estocolmo)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Medio Oriente (Baréin)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
Medio Oriente (EAU)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
América del Sur (São Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra-streams.us-gov-east-1.api.aws	HTTPS	
AWS GovCloud (US-West)	us-gov-west-1	cassandra-streams.us-gov-west-1.api.aws	HTTPS	

Habilita una transmisión de CDC al crear una tabla nueva en Amazon Keyspaces

Para habilitar una transmisión de CDC al crear una tabla, puede usar la `CREATE TABLE` instrucción en CQL o el `create-table` comando con la AWS CLI

Para cada fila modificada de la tabla, Amazon Keyspaces puede capturar los siguientes cambios en función `view_type` de los `cdc_specification` que seleccione:

- `NEW_AND_OLD_IMAGES`— ambas versiones de la fila, antes y después del cambio. Este es el valor predeterminado.
- `NEW_IMAGE`— la versión de la fila tras el cambio.
- `OLD_IMAGE`— la versión de la fila anterior al cambio.
- `KEYS_ONLY`— las claves de partición y agrupamiento de la fila que se ha modificado.

Para obtener información sobre cómo etiquetar una transmisión, consulte [the section called “Cree una transmisión con etiquetas con una nueva tabla”](#).

Note

Amazon Keyspaces CDC requiere la presencia de un rol vinculado a un servicio (AWSServiceRoleForAmazonKeyspacesCDC) que publique los datos de métricas de las transmisiones de Amazon Keyspaces CDC en su cuenta "cloudwatch:namespace": "AWS/Cassandra" en su nombre. CloudWatch Este rol se crea automáticamente. Para obtener más información, consulte [the section called “Transmisiones de los CDC”](#).

Cassandra Query Language (CQL)

Habilite una transmisión de CDC al crear una tabla con CQL

1.

```
CREATE TABLE mykeyspace.mytable (a text, b text, PRIMARY KEY(a))
WITH CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'NEW_IMAGE'}} AND CDC
= TRUE;
```
2. Para confirmar la configuración de la transmisión, puede usar la siguiente declaración.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

El resultado de esa declaración debería tener un aspecto similar al siguiente.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';keyspace_name | table_name | cdc | custom_properties
-----+-----+-----
+-----+-----+-----+-----
mykeyspace | mytable | True | {'capacity_mode':
{'last_update_to_pay_per_request_timestamp': '1741383893782',
'throughput_mode': 'PAY_PER_REQUEST'}, 'cdc_specification':
{'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable/stream/2025-03-07T21:44:53.783', 'status': 'ENABLED',
'view_type': 'NEW_IMAGE'}, 'encryption_specification': {'encryption_type':
'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status': 'disabled'}}>
```

CLI

Habilite una transmisión de los CDC al crear una tabla con AWS CLI

1. Para crear una transmisión, puede usar la siguiente sintaxis.

```
aws keyspaces create-table \  
--keyspace-name 'mykeyspace' \  
--table-name 'mytable' \  
--schema-definition 'allColumns=[{name=a,type=text},  
{name=b,type=text}],partitionKeys=[{name=a}]' \  
--cdc-specification status=ENABLED,viewType=NEW_IMAGE
```

2. El resultado de ese comando muestra la create-table respuesta estándar y tiene un aspecto similar al de este ejemplo.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/  
table/mytable" }
```

Habilitar una transmisión de CDC para una tabla existente en Amazon Keyspaces

Para habilitar una transmisión de CDC para una tabla existente, puede usar la ALTER TABLE sentencia en CQL, el update-table comando junto con la AWS CLI tabla o puede usar la consola.

Para cada fila modificada de la tabla, Amazon Keyspaces puede capturar los siguientes cambios en función view_type de los cdc_specification que seleccione:

- NEW_AND_OLD_IMAGES— ambas versiones de la fila, antes y después del cambio. Este es el valor predeterminado.
- NEW_IMAGE— la versión de la fila tras el cambio.
- OLD_IMAGE— la versión de la fila anterior al cambio.
- KEYS_ONLY— las claves de partición y agrupamiento de la fila que se ha modificado.

Para obtener información sobre cómo etiquetar una transmisión, consulte [the section called “Añadir nuevas etiquetas a una transmisión”](#).

Note

Amazon Keyspaces CDC requiere la presencia de un rol vinculado a un servicio (AWSServiceRoleForAmazonKeyspacesCDC) que publique los datos de métricas de las transmisiones de Amazon Keyspaces CDC en su cuenta "cloudwatch:namespace": "AWS/Cassandra" en su nombre. CloudWatch Este rol se crea automáticamente. Para obtener más información, consulte [the section called “Transmisiones de los CDC”](#).

Cassandra Query Language (CQL)

Habilite una transmisión (transmisión de CDC) con CQL

Se puede utilizar ALTER TABLE para habilitar una transmisión en una tabla existente.

1. El siguiente ejemplo crea una secuencia que solo captura los cambios en las claves de partición y agrupamiento de una fila modificada.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = TRUE
AND CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'KEYS_ONLY'}};
```

2. Para comprobar la configuración de la transmisión, puede utilizar la siguiente declaración.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

El resultado de la declaración tiene un aspecto similar al siguiente.

```
keyspace_name | table_name | cdc | custom_properties
-----+-----+-----
+-----+-----+-----
mykeyspace | mytable | True | {'capacity_mode':
{'last_update_to_pay_per_request_timestamp': '1741385897045',
'throughput_mode': 'PAY_PER_REQUEST'}, 'cdc_specification':
{'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable/stream/2025-03-07T22:20:10.454', 'status': 'ENABLED',
'view_type': 'KEYS_ONLY'}, 'encryption_specification': {'encryption_type':
'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status': 'disabled'}}
```

CLI

Cree una transmisión de los CDC con el AWS CLI

1. Para crear una transmisión para una tabla existente, puede usar la siguiente sintaxis.

```
aws keyspaces update-table \  
--keyspace-name 'mykeyspace' \  
--table-name 'mytable' \  
--cdc-specification status=ENABLED,viewType=NEW_AND_OLD_IMAGES
```

2. El resultado de ese comando muestra la `create-table` respuesta estándar y tiene un aspecto similar al de este ejemplo.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/  
table/mytable" }
```

Console

Habilite una transmisión de CDC con la consola Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, selecciona Tablas y, a continuación, selecciona una tabla de la lista.
3. Seleccione la pestaña Streams.
4. Seleccione Editar para activar una transmisión.
5. Seleccione Activar transmisiones.
6. Seleccione Ver el tipo de transmisión. Están disponibles las siguientes opciones. Ten en cuenta que no puedes cambiar el tipo de visualización de una transmisión una vez creada.
 - Imágenes nuevas y antiguas: Amazon Keyspaces captura ambas versiones de la fila, antes y después del cambio. Esta es la opción predeterminada.
 - Imagen nueva: Amazon Keyspaces captura solo la versión de la fila después del cambio.
 - Imagen antigua: Amazon Keyspaces captura solo la versión de la fila anterior al cambio.
 - Solo clave principal: Amazon Keyspaces captura solo las columnas clave de partición y agrupamiento de la fila que se ha modificado.

7. Para terminar, selecciona Guardar cambios.

Inhabilitar una transmisión de los CDC en Amazon Keyspaces

Para deshabilitar una transmisión de CDC en un espacio de claves, puede usar la ALTER TABLE instrucción en CQL, el update-table comando de o la AWS CLI consola.

Cassandra Query Language (CQL)

Inhabilite una transmisión (transmisión CDC) con CQL

1. Para deshabilitar una transmisión, puede usar la siguiente declaración.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = FALSE;
```

2. Para confirmar que la transmisión está deshabilitada, puede usar la siguiente declaración.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

El resultado de esa declaración tiene un aspecto similar al siguiente.

```
keyspace_name | table_name | cdc    | custom_properties
-----+-----+-----+-----
+-----+-----+-----+-----
mykeyspace   | mytable   | False  | {'capacity_mode':
{'last_update_to_pay_per_request_timestamp': '1741385668642',
'throughput_mode': 'PAY_PER_REQUEST'}, 'encryption_specification':
{'encryption_type': 'AWS_OWNED_KMS_KEY'}, 'point_in_time_recovery': {'status':
'disabled'}}
```

CLI

Desactive una transmisión (transmisión CDC) con el AWS CLI

1. Para deshabilitar una transmisión, puede usar el siguiente comando.

```
aws keyspaces update-table \
```

```
--keyspace-name 'mykeyspace' \  
--table-name 'mytable' \  
--cdc-specification status=DISABLED
```

2. El resultado del comando es similar al de este ejemplo.

```
{  
  "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/  
my_keyspace/",  
  "streamName": "my_stream"  
}
```

Console

Inhabilitar una transmisión (transmisión de CDC) con la consola Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, selecciona Tablas y, a continuación, selecciona una tabla de la lista.
3. Seleccione la pestaña Streams.
4. Elija Edit (Edición de).
5. Deselecciona Activar transmisiones.
6. Selecciona Guardar cambios para deshabilitar la transmisión.

Vea las transmisiones de los CDC en Amazon Keyspaces

Para ver o enumerar todas las transmisiones en el espacio de claves, puede consultar la tabla `system_schema_mcs.streams` en el espacio de claves del sistema mediante una sentencia en CQL o utilizar los `list-stream` comandos `get-stream` y con la consola o con la consola AWS CLI.

Para conocer los permisos necesarios, consulte [the section called “Configuración de permisos”](#).

Cassandra Query Language (CQL)

Vea las transmisiones de los CDC con CQL

- Para monitorear el estado de su tabla en los CDC, puede usar la siguiente declaración.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name='my_keyspace' and table_name='my_table';
```

El resultado del comando tiene un aspecto similar al siguiente.

```
...
custom_properties
-----
{'cdc_specification':{'status': 'Enabled', 'view_type': 'NEW_IMAGE',
  'latest_stream_arn': 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table/stream/stream_label'}}
...
```

CLI

Vea las transmisiones de los CDC con el AWS CLI

1. En este ejemplo se muestra cómo ver la información de transmisión de una tabla.

```
aws keyspaces get-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table'
```

El resultado del comando tiene este aspecto.

```
{
  "keyspaceName": "my_keyspace",
  "tableName": "my_table",
  ... Other fields ...,
  "latestStreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/table/my_table/stream/stream_label",
  "cdcSpecification": {
    "status": "ENABLED",
```

```

    "viewType": "NEW_AND_OLD_IMAGES"
  }
}

```

2. Puede enumerar todas las transmisiones de su cuenta de la forma especificada Región de AWS. El siguiente comando es un ejemplo de esto.

```
aws keyspacesstreams list-streams --region us-east-1
```

El resultado del comando podría tener un aspecto similar al siguiente.

```

{
  "Streams": [
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t1/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"
      "TableName": "t1",
    },
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"Create a keyspace with the name catalog. Note
that streams are not supported in multi-Region
keyspaces.
      "TableName": "t2",
    },
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_2/table/t1/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_3"
      "TableName": "t1",
    }
  ]
}

```

3. También puede enumerar las secuencias CDC de un espacio de claves determinado mediante los siguientes parámetros.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --region us-east-1
```

El resultado del comando tiene un aspecto similar al siguiente.

```
{
  "Streams": [
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t1/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"
      "TableName": "t1",
    },
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"
      "TableName": "t2",
    }
  ]
}
```

4. También puede enumerar los flujos CDC de una tabla determinada utilizando los siguientes parámetros.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --table-name t2 --
region us-east-1
```

El resultado del comando tiene un aspecto similar al siguiente.

```
{
  "Streams": [
    {
      "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
ks_1/table/t2/stream/2023-05-11T21:21:33.291",
      "StreamLabel": "2023-05-11T21:21:33.291",
      "KeyspaceName": "ks_1"
      "TableName": "t2",
    }
  ]
}
```

```
}
```

Console

Vea las transmisiones de los CDC en la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, selecciona Tablas y, a continuación, selecciona una tabla de la lista.
3. Seleccione la pestaña Transmisiones para revisar los detalles de la transmisión.

Acceda a los registros de las transmisiones de los CDC en Amazon Keyspaces

Para acceder a los registros de una transmisión, utilice la API [Amazon Keyspaces Streams](#). La siguiente sección contiene ejemplos sobre cómo acceder a los registros mediante AWS CLI

Para conocer los permisos necesarios, consulte [the section called "Configuración de permisos"](#).

Acceda a los registros de una transmisión mediante el AWS CLI

1. Puede utilizar la API Amazon Keyspaces Streams para acceder a los registros de cambios de la transmisión. Para obtener más información, consulte la referencia de la [API de Amazon Keyspaces Streams](#). Para recuperar los fragmentos de la transmisión, puede utilizar la get-stream API, tal y como se muestra en el siguiente ejemplo.

```
aws keyspacesstreams get-stream \  
--stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/  
mytable/stream/STREAM_LABEL '
```

A continuación se muestra un ejemplo de la salida.

```
{  
  "StreamArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/  
table/mytable/stream/2023-05-11T21:21:33.291",  
  "StreamStatus": "ENABLED",  
  "StreamViewType": "NEW_AND_OLD_IMAGES",  
  "CreationRequestDateTime": "<CREATION_TIME>",  
  "KeyspaceName": "mykeyspace",
```

```

    "TableName": "mytable",
    "StreamLabel": "2023-05-11T21:21:33.291",
    "Shards": [
      {
        "SequenceNumberRange": {
          "EndingSequenceNumber": "<END_SEQUENCE_NUMBER>",
          "StartingSequenceNumber": "<START_SEQUENCE_NUMBER>"
        },
        "ShardId": "<SHARD_ID>"
      },
    ]
  }
}

```

2. Para recuperar registros de la transmisión, comience por obtener un iterador que le proporcione el punto de partida para acceder a los registros. Para ello, puedes usar los fragmentos de la transmisión de CDC devueltos por la API en el paso anterior. Para recopilar el iterador, puedes usar la `get-shard-iterator` API. Para este ejemplo, se utiliza un iterador del tipo `TRIM_HORIZON` que recupera datos del último punto recortado (o principio) del fragmento.

```

aws keyspacesstreams get-shard-iterator \
--stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable/stream/STREAM_LABEL' \
--shard-id 'SHARD_ID' \
--shard-iterator-type 'TRIM_HORIZON'

```

El resultado del comando es similar al del siguiente ejemplo.

```

{
  "ShardIterator": "<SHARD_ITERATOR>"
}

```

3. Para recuperar los registros de los CDC mediante la `get-records` API, puede utilizar el iterador devuelto en el último paso. El siguiente comando es un ejemplo de esto.

```

aws keyspacesstreams get-records \
--shard-iterator 'SHARD_ITERATOR' \
--limit 100

```

Utilice la biblioteca de clientes de Kinesis (KCL) para procesar las transmisiones de Amazon Keyspaces

En este tema se describe cómo utilizar la biblioteca de clientes de Kinesis (KCL) para consumir y procesar datos de las transmisiones de captura de datos de cambios (CDC) de Amazon Keyspaces.

En lugar de trabajar directamente con la API Streams de Amazon Keyspaces, trabajar con la biblioteca de clientes de Kinesis (KCL) ofrece muchas ventajas, por ejemplo:

- El seguimiento del linaje de fragmentos y el manejo de iteradores están integrados.
- Equilibrio de carga automático entre los trabajadores.
- Tolerancia a las fallas y recuperación de las fallas de los trabajadores.
- Puntos de control para realizar un seguimiento del progreso del procesamiento.
- Adaptación a los cambios en la capacidad de los arroyos.
- Computación distribuida simplificada para procesar los registros de los CDC.

En la siguiente sección se explica por qué y cómo utilizar la biblioteca de clientes de Kinesis (KCL) para procesar transmisiones y se proporciona un ejemplo de cómo procesar una transmisión CDC de Amazon Keyspaces con la KCL.

Para obtener información sobre precios, consulte [Precios de Amazon Keyspaces \(para Apache Cassandra\)](#).

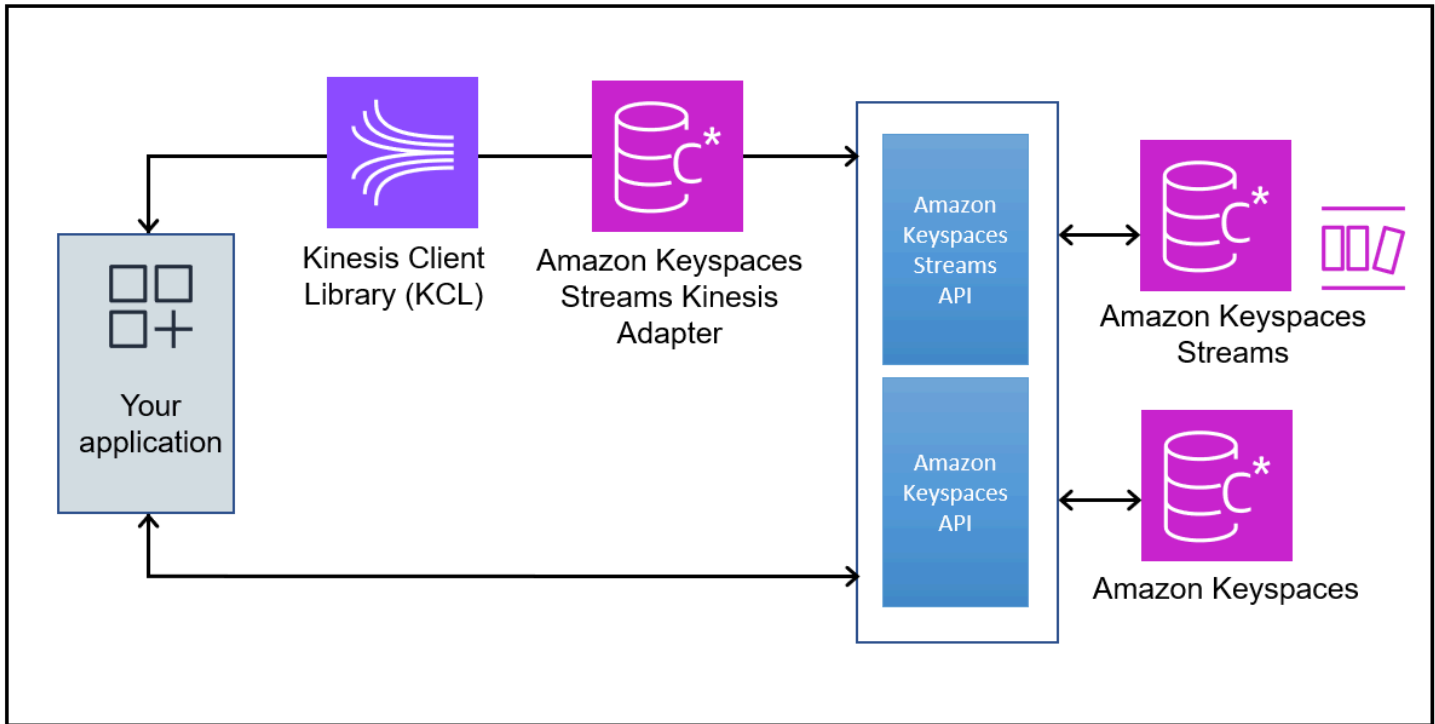
¿Qué es Kinesis Client Library?

La biblioteca de clientes de Kinesis (KCL) es una biblioteca de software Java independiente diseñada para simplificar el proceso de consumo y procesamiento de datos de las transmisiones. KCL gestiona muchas de las tareas complejas asociadas a la informática distribuida, lo que le permite centrarse en implementar la lógica empresarial al procesar los datos de las transmisiones. KCL gestiona actividades como equilibrar la carga entre varios trabajadores, responder a las fallas de los trabajadores, controlar los registros procesados y responder a los cambios en la cantidad de fragmentos de la transmisión.

Para procesar las transmisiones CDC de Amazon Keyspaces, puede utilizar los patrones de diseño que se encuentran en la KCL para trabajar con fragmentos y registros de transmisiones. KCL simplifica la codificación porque proporciona abstracciones útiles por encima del API de bajo nivel de Kinesis Data Streams. Para obtener más información sobre la KCL, consulte [Desarrollar consumidores con la KCL](#) en la Guía para desarrolladores de Amazon Kinesis Data Streams.

Para escribir aplicaciones con la KCL, utilice el adaptador Amazon Keyspaces Streams Kinesis. El adaptador Kinesis implementa la interfaz Kinesis Data Streams para que pueda utilizar la KCL para consumir y procesar registros de las transmisiones de Amazon Keyspaces. Para obtener instrucciones sobre cómo configurar e instalar el adaptador Amazon Keyspaces Streams Kinesis, visite el repositorio. [GitHub](#)

El siguiente diagrama muestra cómo interactúan estas bibliotecas entre sí.



KCL se actualiza con frecuencia para incorporar versiones más recientes de las bibliotecas subyacentes, mejoras de seguridad y correcciones de errores. Recomendamos utilizar la versión más reciente de KCL para evitar los problemas ya conocidos y beneficiarse de las mejoras actuales. [Para encontrar la versión más reciente de KCL, consulte el repositorio de KCL. GitHub](#)

Conceptos de KCL

Antes de implementar una aplicación de consumo mediante KCL, debe comprender los siguientes conceptos:

Aplicación KCL para consumidores

Una aplicación para consumidores de KCL es un programa que procesa datos de una transmisión CDC de Amazon Keyspaces. La KCL actúa como intermediaria entre el código de tu aplicación de consumidor y la transmisión CDC de Amazon Keyspaces.

Trabajador

Un trabajador es una unidad de ejecución de su aplicación de consumo de KCL que procesa datos de la transmisión CDC de Amazon Keyspaces. Su aplicación puede ejecutar varios trabajadores distribuidos en varias instancias.

Procesador de registros

Un procesador de registros es la lógica de su aplicación que procesa los datos de un fragmento de la transmisión CDC de Amazon Keyspaces. Un trabajador crea una instancia de un procesador de registros para cada fragmento que administra.

Arrendamiento

Un arrendamiento representa la responsabilidad de procesar un fragmento. Los trabajadores utilizan los contratos de arrendamiento para coordinar qué trabajador procesa qué fragmento. KCL almacena los datos de arrendamiento en una tabla de Amazon DynamoDB.

Checkpoint

Un punto de control es un registro de la posición del fragmento hasta la que el procesador de registros ha procesado correctamente los registros. Los puntos de control permiten que su solicitud reanude el procesamiento desde donde lo dejó en caso de que un trabajador no lo logre.

Con el adaptador Amazon Keyspaces Kinesis instalado, puede empezar a desarrollar en la interfaz KCL, con las llamadas a la API dirigidas sin problemas al punto de conexión de transmisión de Amazon Keyspaces. Para obtener una lista de los puntos de conexión disponibles, consulte [the section called “Acceda a los puntos de enlace de transmisión de los CDC”](#).

Cuando se inicia la aplicación, llama a KCL para crear una instancia de un proceso de trabajo. Debe proporcionar al trabajador la información de configuración de la aplicación, como el descriptor de la transmisión y AWS las credenciales, y el nombre de la clase de procesador de registros que proporcione. A medida que el proceso de trabajo ejecuta el código en el procesador de registros, lleva a cabo las siguientes tareas:

- Se conecta a la secuencia
- Enumera las particiones del flujo.
- Coordina la asociación de los fragmentos con otros procesos de trabajo (si procede)
- Crea instancias de un procesador de registros para cada fragmento que administra
- Extrae registros del flujo.

- Inserta los registros en el procesador de registros correspondiente
- Genera puntos de comprobación para los registros procesados
- Balancea las asociaciones entre fragmentos y procesos de trabajo cuando cambia el recuento de instancias de procesos de trabajo
- Equilibra las asociaciones entre particiones y procesos de trabajo cuando las particiones se dividen.

Implemente una aplicación de consumo de KCL para las transmisiones de los CDC de Amazon Keyspaces

En este tema se proporciona una guía paso a paso para implementar una aplicación de consumo de KCL para procesar las transmisiones de Amazon Keyspaces CDC.

1. Requisitos previos: antes de empezar, asegúrese de tener:
 - Una tabla de Amazon Keyspaces con una transmisión de CDC
 - Permisos de IAM necesarios para que el director de IAM pueda acceder a la transmisión CDC de Amazon Keyspaces, crear tablas de DynamoDB y acceder a ellas para el procesamiento de transmisiones de KCL y permisos para publicar métricas en ellas. CloudWatch Para obtener más información y un ejemplo de política, consulte [the section called “Permisos para KCL”](#)
 - Asegúrese de que AWS las credenciales válidas estén configuradas en su configuración local. Para obtener más información, consulte [the section called “Administración de claves de acceso”](#).
 - Java Development Kit (JDK) 8 o posterior,
 - Los requisitos figuran en el [archivo readme](#) de Github.
2. En este paso, agregas la dependencia de KCL a tu proyecto. Para Maven, agrega lo siguiente a tu archivo pom.xml:

```
<dependencies>
  <dependency>
    <groupId>software.amazon.kinesis</groupId>
    <artifactId>amazon-kinesis-client</artifactId>
    <version>3.1.0</version>
  </dependency>
  <dependency>
    <groupId>software.amazon.keyspaces</groupId>
```

```

        <artifactId>keyspaces-streams-kinesis-adapter</artifactId>
        <version>1.0.0</version>
    </dependency>
</dependencies>

```

Note

Compruebe siempre la última versión de KCL en el repositorio de [GitHub KCL](#).

3. Cree una clase de fábrica que produzca instancias de procesadores de registros:

```

import software.amazon.awssdk.services.keyspacesstreams.model.Record;
import
    software.amazon.keyspaces.streamsadapter.adapter.KeyspacesStreamsClientRecord;
import
    software.amazon.keyspaces.streamsadapter.model.KeyspacesStreamsProcessRecordsInput;
import
    software.amazon.keyspaces.streamsadapter.processor.KeyspacesStreamsShardRecordProcessor;
import software.amazon.kinesis.lifecycle.events.InitializationInput;
import software.amazon.kinesis.lifecycle.events.LeaseLostInput;
import software.amazon.kinesis.lifecycle.events.ShardEndedInput;
import software.amazon.kinesis.lifecycle.events.ShutdownRequestedInput;
import software.amazon.kinesis.processor.RecordProcessorCheckpoint;

public class RecordProcessor implements KeyspacesStreamsShardRecordProcessor {
    private String shardId;

    @Override
    public void initialize(InitializationInput initializationInput) {
        this.shardId = initializationInput.shardId();
        System.out.println("Initializing record processor for shard: " + shardId);
    }

    @Override
    public void processRecords(KeyspacesStreamsProcessRecordsInput
processRecordsInput) {
        try {
            for (KeyspacesStreamsClientRecord record :
processRecordsInput.records()) {
                Record keyspacesRecord = record.getRecord();
                System.out.println("Received record: " + keyspacesRecord);
            }
        }
    }
}

```

```

        if (!processRecordsInput.records().isEmpty()) {
            RecordProcessorCheckpointier checkpointer =
processRecordsInput.checkpointer();
            try {
                checkpointer.checkpoint();
                System.out.println("Checkpoint successful for shard: " +
shardId);
            } catch (Exception e) {
                System.out.println("Error while checkpointing for shard: " +
shardId + " " + e);
            }
        }
    } catch (Exception e) {
        System.out.println("Error processing records for shard: " + shardId + "
" + e);
    }
}

@Override
public void leaseLost(LeaseLostInput leaseLostInput) {
    System.out.println("Lease lost for shard: " + shardId);
}

@Override
public void shardEnded(ShardEndedInput shardEndedInput) {
    System.out.println("Shard ended: " + shardId);
    try {
        // This is required. Checkpoint at the end of the shard
        shardEndedInput.checkpointer().checkpoint();
        System.out.println("Final checkpoint successful for shard: " +
shardId);
    } catch (Exception e) {
        System.out.println("Error while final checkpointing for shard: " +
shardId + " " + e);
        throw new RuntimeException("Error while final checkpointing", e);
    }
}

@Override
public void shutdownRequested(ShutdownRequestedInput shutdownRequestedInput) {
    System.out.println("Shutdown requested for shard " + shardId);
    try {
        shutdownRequestedInput.checkpointer().checkpoint();
    }
}

```

```

        } catch (Exception e) {
            System.out.println("Error while checkpointing on shutdown for shard: "
+ shardId + " " + e);
        }
    }
}

```

4. Cree una fábrica de discos como se muestra en el siguiente ejemplo.

```

import software.amazon.kinesis.processor.ShardRecordProcessor;
import software.amazon.kinesis.processor.ShardRecordProcessorFactory;

import java.util.Queue;
import java.util.concurrent.ConcurrentLinkedQueue;

public class RecordProcessorFactory implements ShardRecordProcessorFactory {
    private final Queue<RecordProcessor> processors = new
ConcurrentLinkedQueue<>();

    @Override
    public ShardRecordProcessor shardRecordProcessor() {
        System.out.println("Creating new RecordProcessor");
        RecordProcessor processor = new RecordProcessor();
        processors.add(processor);
        return processor;
    }
}

```

5. En este paso, creará la clase base para configurar KCLv3 y el adaptador Amazon Keyspaces.

```

import com.example.KCLExample.utils.RecordProcessorFactory;
import
    software.amazon.keyspaces.streamsadapter.AmazonKeyspacesStreamsAdapterClient;
import software.amazon.keyspaces.streamsadapter.StreamsSchedulerFactory;
import java.util.Arrays;
import java.util.List;
import java.util.concurrent.ExecutionException;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatch.CloudWatchAsyncClient;
import software.amazon.awssdk.services.dynamodb.DynamoDbAsyncClient;
import software.amazon.awssdk.services.dynamodb.model.DeleteTableRequest;
import software.amazon.awssdk.services.dynamodb.model.DeleteTableResponse;

```

```
import software.amazon.awssdk.services.keyspacesstreams.KeyspacesStreamsClient;
import software.amazon.awssdk.services.kinesis.KinesisAsyncClient;
import software.amazon.kinesis.common.ConfigsBuilder;
import software.amazon.kinesis.common.InitialPositionInStream;
import software.amazon.kinesis.common.InitialPositionInStreamExtended;
import software.amazon.kinesis.coordinator.CoordinatorConfig;
import software.amazon.kinesis.coordinator.Scheduler;
import software.amazon.kinesis.leases.LeaseManagementConfig;
import software.amazon.kinesis.processor.ProcessorConfig;
import software.amazon.kinesis.processor.StreamTracker;
import software.amazon.kinesis.retrieval.polling.PollingConfig;

public class KCLTestBase {

    protected KeyspacesStreamsClient streamsClient;
    protected KinesisAsyncClient adapterClient;
    protected DynamoDbAsyncClient dynamoDbAsyncClient;
    protected CloudWatchAsyncClient cloudWatchClient;
    protected Region region;
    protected RecordProcessorFactory recordProcessorFactory;
    protected Scheduler scheduler;
    protected Thread schedulerThread;

    public void baseSetUp() {
        recordProcessorFactory = new RecordProcessorFactory();
        setupKCLBase();
    }

    protected void setupKCLBase() {
        region = Region.US_EAST_1;

        streamsClient = KeyspacesStreamsClient.builder()
            .region(region)
            .build();
        adapterClient = new AmazonKeyspacesStreamsAdapterClient(
            streamsClient,
            region);
        dynamoDbAsyncClient = DynamoDbAsyncClient.builder()
            .region(region)
            .build();
        cloudWatchClient = CloudWatchAsyncClient.builder()
            .region(region)
            .build();
    }
}
```

```
protected void startScheduler(Scheduler scheduler) {
    this.scheduler = scheduler;
    schedulerThread = new Thread(() -> scheduler.run());
    schedulerThread.start();
}

protected void shutdownScheduler() {
    if (scheduler != null) {
        scheduler.shutdown();
        try {
            schedulerThread.join(30000);
        } catch (InterruptedException e) {
            System.out.println("Error while shutting down scheduler " + e);
        }
    }
}

protected Scheduler createScheduler(String streamArn, String leaseTableName) {
    String workerId = "worker-" + System.currentTimeMillis();

    // Create ConfigsBuilder
    ConfigsBuilder configsBuilder = createConfigsBuilder(streamArn, workerId,
leaseTableName);

    // Configure retrieval config for polling
    PollingConfig pollingConfig = new PollingConfig(streamArn, adapterClient);

    // Create the Scheduler
    return StreamsSchedulerFactory.createScheduler(
        configsBuilder.checkpointConfig(),
        configsBuilder.coordinatorConfig(),
        configsBuilder.leaseManagementConfig(),
        configsBuilder.lifecycleConfig(),
        configsBuilder.metricsConfig(),
        configsBuilder.processorConfig(),

configsBuilder.retrievalConfig().retrievalSpecificConfig(pollingConfig),
        streamsClient,
        region
    );
}
```

```
private ConfigsBuilder createConfigsBuilder(String streamArn, String workerId,
String leaseTableName) {
    ConfigsBuilder configsBuilder = new ConfigsBuilder(
        streamArn,
        leaseTableName,
        adapterClient,
        dynamoDbAsyncClient,
        cloudWatchClient,
        workerId,
        recordProcessorFactory);

    configureCoordinator(configsBuilder.coordinatorConfig());
    configureLeaseManagement(configsBuilder.leaseManagementConfig());
    configureProcessor(configsBuilder.processorConfig());
    configureStreamTracker(configsBuilder, streamArn);

    return configsBuilder;
}

private void configureCoordinator(CoordinatorConfig config) {
    config.skipShardSyncAtWorkerInitializationIfLeasesExist(true)
        .parentShardPollIntervalMillis(1000)
        .shardConsumerDispatchPollIntervalMillis(500);
}

private void configureLeaseManagement(LeaseManagementConfig config) {
    config.shardSyncIntervalMillis(0)
        .leasesRecoveryAuditorInconsistencyConfidenceThreshold(0)
        .leasesRecoveryAuditorExecutionFrequencyMillis(5000)
        .leaseAssignmentIntervalMillis(1000L);
}

private void configureProcessor(ProcessorConfig config) {
    config.callProcessRecordsEvenForEmptyRecordList(true);
}

private void configureStreamTracker(ConfigsBuilder configsBuilder, String
streamArn) {
    StreamTracker streamTracker =
StreamsSchedulerFactory.createSingleStreamTracker(
        streamArn,

        InitialPositionInStreamExtended.newInitialPosition(InitialPositionInStream.TRIM_HORIZON)
    );
}
```

```

        configsBuilder.streamTracker(streamTracker);
    }

    public void deleteAllDdbTables(String baseTableName) {
        List<String> tablesToDelete = Arrays.asList(
            baseTableName,
            baseTableName + "-CoordinatorState",
            baseTableName + "-WorkerMetricStats"
        );

        for (String tableName : tablesToDelete) {
            deleteTable(tableName);
        }
    }

    private void deleteTable(String tableName) {
        DeleteTableRequest deleteTableRequest = DeleteTableRequest.builder()
            .tableName(tableName)
            .build();

        try {
            DeleteTableResponse response =
dynamoDbAsyncClient.deleteTable(deleteTableRequest).get();
            System.out.println("Table deletion response " + response);
        } catch (InterruptedException | ExecutionException e) {
            System.out.println("Error deleting table: " + tableName + " " + e);
        }
    }
}

```

6. En este paso, implementa la clase de procesador de registros para que su aplicación comience a procesar los eventos de cambio.

```

import software.amazon.kinesis.coordinator.Scheduler;

public class KCLTest {

    private static final int APP_RUNTIME_SECONDS = 1800;
    private static final int SLEEP_INTERNAL_MS = 60*1000;

    public static void main(String[] args) {
        KCLTestBase kclTestBase;
    }
}

```



```

    kclTestBase = new KCLTestBase();
    kclTestBase.baseSetup();

    // Create and start scheduler
    String leaseTableName = generateUniqueApplicationName();

    // Update below to your Stream ARN
    String streamArn = "arn:aws:cassandra:us-east-1:759151643516:/keyspace/
cdc_sample_test/table/test_kcl_bool/stream/2025-07-01T15:52:57.529";
    Scheduler scheduler = kclTestBase.createScheduler(streamArn,
leaseTableName);
    kclTestBase.startScheduler(scheduler);

    // Wait for specified time before shutting down - KCL applications are
    designed to run forever, however in this
    // example we will shut it down after APP_RUNTIME_SECONDS
    long startTime = System.currentTimeMillis();
    long endTime = startTime + (APP_RUNTIME_SECONDS * 1000);
    while (System.currentTimeMillis() < endTime) {
        try {
            // Print and sleep every minute
            Thread.sleep(SLEEP_INTERNAL_MS);
            System.out.println("Application is running");
        } catch (InterruptedException e) {
            System.out.println("Interrupted while waiting for records");
            Thread.currentThread().interrupt();
            break;
        }
    }

    // Stop the scheduler
    kclTestBase.shutdownScheduler();
    kclTestBase.deleteAllDdbTables(leaseTableName);
}

public static String generateUniqueApplicationName() {
    String timestamp = String.valueOf(System.currentTimeMillis());
    String randomString = java.util.UUID.randomUUID().toString().substring(0,
8);
    return String.format("KCL-App-%s-%s", timestamp, randomString);
}
}

```

Prácticas recomendadas

Siga estas prácticas recomendadas cuando utilice KCL con las transmisiones de los CDC de Amazon Keyspaces:

Error handling (Control de errores)

Implemente una sólida gestión de errores en su procesador de registros para gestionar las excepciones sin problemas. Considere la posibilidad de implementar una lógica de reintento para los errores transitorios.

Frecuencia de los puntos de control

Equilibre la frecuencia de los puntos de control para minimizar el procesamiento duplicado y, al mismo tiempo, garantizar un seguimiento razonable del progreso. Los puntos de control demasiado frecuentes pueden afectar al rendimiento, mientras que los controles demasiado infrecuentes pueden provocar un mayor reprocesamiento si un trabajador no lo logra.

Escalamiento de los trabajadores

Escale la cantidad de trabajadores en función de la cantidad de fragmentos de su transmisión de los CDC. Un buen punto de partida es tener un trabajador por fragmento, pero es posible que tengas que ajustarlo en función de tus requisitos de procesamiento.

Supervisión

Utilice CloudWatch las métricas proporcionadas por KCL para supervisar el estado y el rendimiento de su aplicación de consumo. Las métricas clave incluyen la latencia de procesamiento, la antigüedad de los puntos de control y el recuento de arrendamientos.

Pruebas

Pruebe minuciosamente su aplicación de consumo, incluidos escenarios como los fallos de los trabajadores, la refragmentación de las transmisiones y las condiciones de carga variables.

Uso de KCL con lenguajes distintos de Java

Si bien KCL es principalmente una biblioteca de Java, puede usarla con otros lenguajes de programación a través de. MultiLangDaemon MultiLangDaemon Es un Java-based daemon que gestiona la interacción entre un procesador de registros que no es Java y el KCL.

KCL es compatible con los siguientes idiomas:

- Python
- Ruby
- Node.js
- .NET

[Para obtener más información sobre el uso de KCL con lenguajes distintos de Java, consulte la documentación de KCL. MultiLangDaemon](#)

Resolución de problemas

En esta sección se proporcionan soluciones a los problemas habituales que pueden surgir al utilizar KCL con las transmisiones de los CDC de Amazon Keyspaces.

Procesamiento lento

Si su solicitud de consumidor procesa los registros lentamente, considere lo siguiente:

- Aumentar el número de instancias de trabajadores
- Optimizar la lógica de procesamiento de registros
- Comprobación de cuellos de botella en los sistemas posteriores

Procesamiento duplicado

Si observa un procesamiento duplicado de registros, compruebe la lógica de los puntos de control. Asegúrese de realizar un control después de procesar correctamente los registros.

Fallos de los trabajadores

Si los trabajadores fallan con frecuencia, compruebe:

- Restricciones de recursos (CPU, memoria)
- Problemas de conectividad de red
- Problemas con los permisos

Problemas con la tabla de arrendamiento

Si tienes problemas con la tabla de arrendamientos de KCL:

- Comprueba que tu aplicación tiene los permisos adecuados para acceder a la tabla de Amazon Keyspaces
- Compruebe que la tabla tiene un rendimiento aprovisionado suficiente

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)

Apache Cassandra-compatible `Murmur3Partitioner`. `Murmur3Partitioner` es el particionador predeterminado de Cassandra en Amazon Keyspaces y en Cassandra 1.2 y versiones posteriores.

RandomPartitioner

Apache Cassandra-compatible `RandomPartitioner`. `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.

Cambio del particionador en Amazon Keyspaces

Puede cambiar el particionador utilizando el lenguaje de consulta Cassandra (CQL) Consola de administración de AWS o Cassandra.

Consola de administración de AWS

Para cambiar el particionador con la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija 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.

Client-side marcas de tiempo en Amazon Keyspaces

En Amazon Keyspaces, las marcas de tiempo del lado del cliente son marcas de tiempo que se conservan para cada Cassandra-compatible celda de la 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.

Las marcas de tiempo 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.

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). Para obtener más información, consulte [the section called “Uso de](#)

[marcas de tiempo del cliente en consultas](#)". 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 en el nivel de celda, ya que las marcas de tiempo no pueden ser NULL. Para consultar las marcas de tiempo, puede utilizar la función `WRITETIME` en su instrucción DML.

Amazon Keyspaces no cobra cargos adicionales por activar las marcas de tiempo 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 sobre precios de Amazon Keyspaces, consulte [Precios de Amazon Keyspaces \(para Apache Cassandra\)](#).

Cuando las marcas de tiempo 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 "Estimación 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 "Configure los modos read/write de capacidad"](#).

Una vez activadas las marcas de tiempo del cliente para una tabla, no puede desactivarlas.

Para obtener más información sobre cómo utilizar las marcas de tiempo del cliente en las consultas, consulte [the section called "Uso de marcas de tiempo del cliente en consultas"](#).

Temas

- [Cómo se integran las marcas de tiempo del lado del cliente de Amazon Keyspaces con los servicios AWS](#)
- [Creación de una nueva tabla con marcas de tiempo del cliente en Amazon Keyspaces](#)
- [Configuración de marcas de tiempo del cliente para una tabla en Amazon Keyspaces](#)
- [Uso de marcas de tiempo del cliente en consultas en Amazon Keyspaces](#)

Cómo se integran las marcas de tiempo del lado del cliente de Amazon Keyspaces con los servicios AWS

La siguiente métrica de marcas horarias del lado del cliente está disponible en Amazon CloudWatch para permitir la supervisión continua.

- `SystemReconciliationDeletes`: el número de operaciones de eliminación necesarias para eliminar los datos lapidados.

Para obtener más información sobre cómo supervisar CloudWatch las métricas, consulte [the section called “Monitorización con CloudWatch”](#)

Cuando lo usas CloudFormation, puedes habilitar las marcas de tiempo del lado del cliente al crear una tabla de Amazon Keyspaces. Para obtener más información, consulte la [Guía del usuario de AWS CloudFormation](#).

Creación de una nueva tabla con marcas de tiempo del cliente en Amazon Keyspaces

Siga estos ejemplos para crear una nueva tabla de Amazon Keyspaces con marcas de tiempo del lado del cliente habilitadas mediante Amazon Consola de administración de AWS Keyspaces, Cassandra Query Language (CQL) o AWS Command Line Interface

Console

Creación de una nueva tabla con marcas de tiempo del cliente (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 Esquema, cree el esquema para su tabla.
5. En la sección Configuración de la tabla, elija Personalizar configuración.
6. Continúe con las Client-side marcas de tiempo.

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 cliente activadas.

Cassandra Query Language (CQL)

Creación de una tabla utilizando CQL

1. Para crear una nueva tabla con marcas de tiempo del cliente activadas mediante CQL, puede utilizar el siguiente ejemplo.

```
CREATE TABLE my_keyspace.my_table (  
    userid uuid,  
    time timeuuid,  
    subject text,  
    body text,  
    user inet,  
    PRIMARY KEY (userid, time)  
) WITH CUSTOM_PROPERTIES = {'client_side_timestamps': {'status': 'enabled'}};
```

2. Para confirmar la configuración de las marcas de tiempo 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 cliente.

```
'client_side_timestamps': {'status': 'enabled'}
```

AWS CLI

Cree una tabla nueva con el AWS CLI

1. Para crear una nueva tabla con marcas de tiempo del cliente activadas, puede utilizar el siguiente ejemplo.

```
./aws keyspaces create-table \  
--keyspace-name my_keyspace \  
--table-name my_table \  
--client-side-timestamps 'status=ENABLED' \  
--schema-definition 'allColumns=[{name=id,type=int},{name=date,type=timestamp},  
{name=name,type=text}],partitionKeys=[{name=id}]'
```

2. Para confirmar que las marcas de tiempo 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-1:111122223333:/keyspace/  
my_keyspace/table/my_table",  
  "creationTimestamp": 1662681206.032,  
  "status": "ACTIVE",  
  "schemaDefinition": {  
    "allColumns": [  
      {  
        "name": "id",  
        "type": "int"  
      },  
      {  
        "name": "date",  
        "type": "timestamp"  
      },  
      {  
        "name": "name",  
        "type": "text"  
      }  
    ],  
    "partitionKeys": [  
      {  
        "name": "id"
```

```

        }
    ],
    "clusteringKeys": [],
    "staticColumns": []
},
"capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": 1662681206.032
},
"encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
    "status": "DISABLED"
},
"clientSideTimestamps": {
    "status": "ENABLED"
},
"ttl": {
    "status": "ENABLED"
},
"defaultTimeToLive": 0,
"comment": {
    "message": ""
}
}

```

Configuración de marcas de tiempo del cliente para una tabla en Amazon Keyspaces

Siga estos ejemplos para activar las marcas de tiempo del lado del cliente para las tablas existentes mediante Amazon Keyspaces Consola de administración de AWS, Cassandra Query Language (CQL) o. AWS Command Line Interface

Console

Para activar las marcas de tiempo del cliente para una tabla existente (consola)

1. Inicie sesión en y abra la Consola de administración de AWS 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.

Cassandra Query Language (CQL)

Uso de una instrucción CQL

1. Active las marcas de tiempo del cliente de una tabla existente con una instrucción CQL `ALTER TABLE`.

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps':  
{'status': 'enabled'}};;
```

2. Para confirmar la configuración de las marcas de tiempo 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.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 cliente.

```
'client_side_timestamps': {'status': 'enabled'}
```

AWS CLI

Usando el AWS CLI

1. Puede activar las marcas de tiempo del lado del cliente para una tabla existente AWS CLI mediante el siguiente ejemplo.

```
./aws keyspaces update-table \  
--keyspace-name my_keyspace \  
--table-name my_table \  
--client-side-timestamps 'status=ENABLED'
```

2. Para confirmar que las marcas de tiempo 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 e indicar el estado de las marcas de tiempo del cliente como ENABLED.

```
{  
  "keyspaceName": "my_keyspace",  
  "tableName": "my_table",  
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/  
my_keyspace/table/my_table",  
  "creationTimestamp": 1662681312.906,  
  "status": "ACTIVE",  
  "schemaDefinition": {  
    "allColumns": [  
      {  
        "name": "id",  
        "type": "int"  
      },  
      {  
        "name": "date",  
        "type": "timestamp"  
      },  
      {  
        "name": "name",  
        "type": "text"  
      }  
    ],  
    "partitionKeys": [  
      {  
        "name": "id"  
      }  
    ],  
    "clusteringKeys": [],  
    "staticColumns": []  
  },  
  "capacitySpecification": {  
    "throughputMode": "PAY_PER_REQUEST",
```

```
    "lastUpdateToPayPerRequestTimestamp": 1662681312.906
  },
  "encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
  },
  "pointInTimeRecovery": {
    "status": "DISABLED"
  },
  "clientSideTimestamps": {
    "status": "ENABLED"
  },
  "ttl": {
    "status": "ENABLED"
  },
  "defaultTimeToLive": 0,
  "comment": {
    "message": ""
  }
}
```

Uso de marcas de tiempo del cliente en consultas en Amazon Keyspaces

Una vez activadas las marcas de tiempo 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;
```

Multi-Region replicación 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 activo-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 frente a la degradación regional y a la vez beneficiarse de lecturas y escrituras locales de baja latencia para aplicaciones globales.

Con la replicación multirregional, Amazon Keyspaces replica de forma asíncrona los datos entre regiones y, por lo general, los datos se propagan entre regiones en un segundo. Además, con la replicación multirregional, ya no tiene la difícil tarea de resolver conflictos y corregir los problemas de divergencia de datos, por lo que puede 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 las distintas zonas geográficas de su elección. Regiones de AWS

Temas

- [Ventajas de utilizar la replicación multirregional](#)
- [Modos de capacidad y precios](#)

- [Cómo funciona la replicación multirregional en Amazon Keyspaces](#)
- [Notas de uso de la replicación multirregional de Amazon Keyspaces](#)
- [Configuración de la replicación multirregional para Amazon Keyspaces \(para Apache Cassandra\)](#)

Ventajas de utilizar la replicación multirregional

Multi-Region la replicación ofrece los siguientes beneficios.

- Lecturas y escrituras globales con una latencia de un dígito de milisegundo: en Amazon Keyspaces, la réplica 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 en varias regiones, puede recuperarse de la degradación en una sola región Región de AWS al redirigir la aplicación a una región diferente en 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. Una vez que la región vuelva a estar en línea, Amazon Keyspaces sincroniza automáticamente los cambios que faltan para que pueda recuperarlos sin que la aplicación se vea afectada.

- High-speed replicación entre regiones: la Multi-Region replicación utiliza una replicación física rápida de los datos entre regiones y basada en el almacenamiento, 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 aquellos que vean picos o aumentos repentinos en el rendimiento sin ningún impacto en la aplicación.

- Coherencia y resolución de conflictos: cualquier cambio que se realice en datos en cualquier región se replica en las demás regiones en un espacio de claves multirregión. 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”](#).

Modos de capacidad 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 “Configure los modos read/write de capacidad”](#).

En el modo bajo demanda, se le facturará 1 unidad de solicitud de escritura (WRU) para escribir hasta 1 KB de datos por fila, del mismo modo que en las tablas de una sola región. Sin embargo, se facturan las escrituras en cada región del espacio de claves multirregional. Por ejemplo, escribir una fila de 3 KB de datos en un espacio de claves multirregional con dos regiones requiere 6 WRU: $3 * 2 = 6$ WRU. Además, las escrituras que incluyan datos estáticos y no estáticos requieren operaciones de escritura adicionales.

En el modo aprovisionado, se le factura 1 unidad de capacidad de escritura (WCU) para escribir hasta 1 KB de datos por fila, de la misma manera que en las tablas de una sola región. Sin embargo, se le facturan 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 6 WCU: $3 * 2 = 6$ 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 replicación multirregional en Amazon Keyspaces

En esta sección se proporciona información general sobre cómo funciona la replicación multirregional de Amazon Keyspaces. Para obtener más información sobre precios, consulte [Precios de Amazon Keyspaces \(para Apache Cassandra\)](#).

Temas

- [Cómo funciona la replicación multirregional en Amazon Keyspaces](#)
- [Multi-Region replicación, resolución de conflictos](#)

- [Multi-Region replicación, recuperación ante desastres](#)
- [Multi-Region la replicación Regiones de AWS está deshabilitada de forma predeterminada](#)
- [Multi-Region replicación e integración con recuperación puntual \(PITR\)](#)
- [Multi-Region replicación e integración con servicios AWS](#)

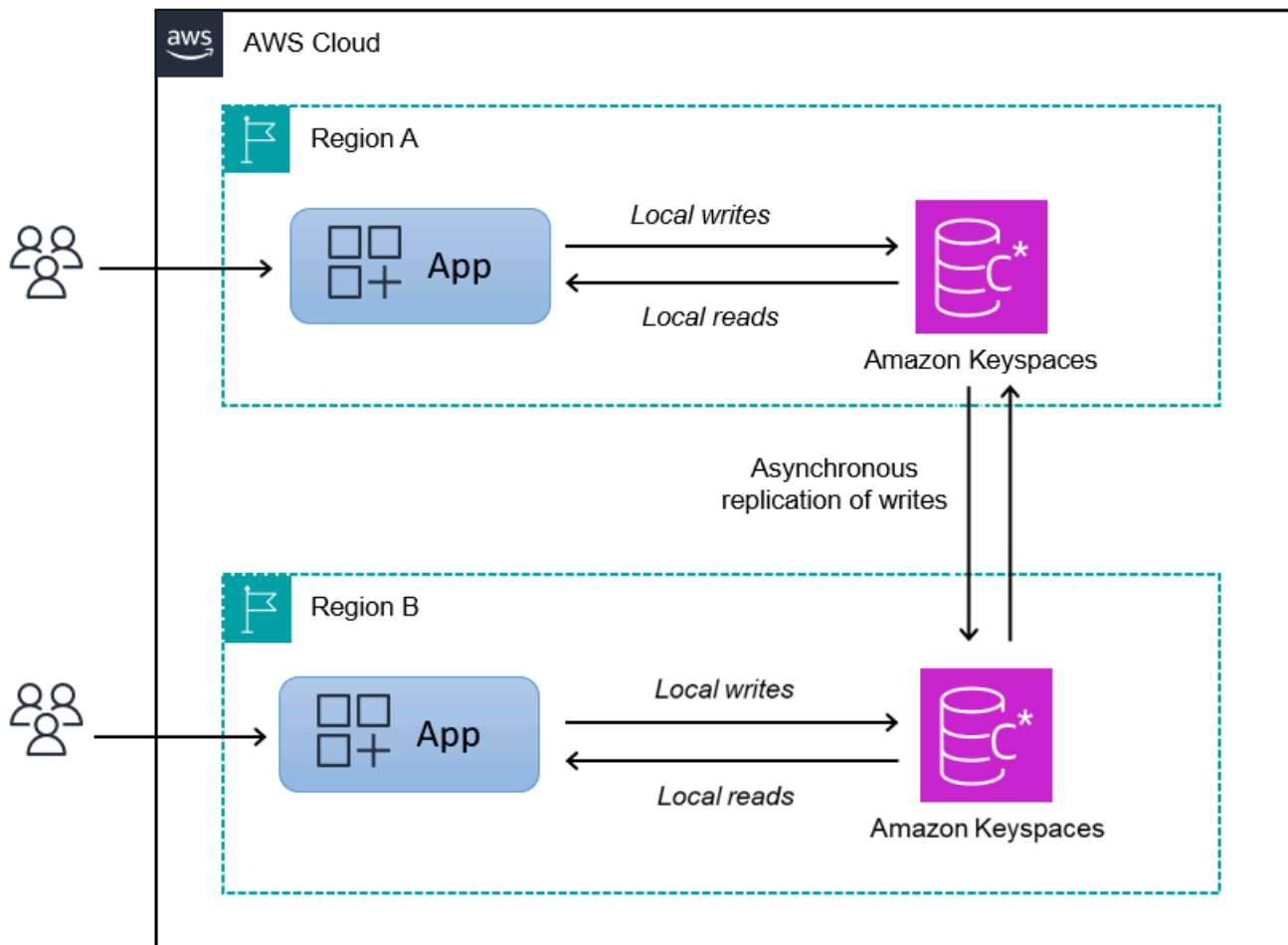
Cómo funciona la replicación multirregional en Amazon Keyspaces

La replicación multirregional de Amazon Keyspaces implementa una arquitectura de resiliencia de datos que distribuye los datos entre personas independientes y distribuidas geográficamente. Regiones de AWS 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 multirregional de Amazon Keyspaces, puede seleccionar regiones adicionales en las que se van a replicar 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”](#).



Multi-Region replicación, resolución de conflictos

La replicación multirregional de Amazon Keyspaces está totalmente gestionada, lo que significa que no tiene que realizar tareas de replicación, como ejecutar operaciones de reparación con regularidad para solucionar problemas de sincronización de datos. Amazon Keyspaces supervisa la coherencia de los datos entre tablas de diferentes tablas Regiones de AWS mediante la detección y la reparación de conflictos, y sincroniza las réplicas automáticamente.

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 [the section called “Client-side marcas de tiempo”](#).

Multi-Region replicación, recuperación ante desastres

Con la replicación multirregional de Amazon Keyspaces, las escrituras se replican de forma asíncrona en cada región. En el raro caso de que una sola región se degrade o falle, la replicación multirregional le ayuda a recuperarse de un 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 el caso de los planes de recuperación ante desastres que utilizan la replicación multirregional para realizar la conmutación por error a una región no afectada, el RTO puede ser prácticamente nulo. 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 el caso de los planes de recuperación ante desastres que utilizan la replicación multirregional para realizar la conmutación por error a una región no afectada, el RPO suele ser de un solo dígito en segundos. 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 sola Región de AWS se aísla o se degrada, la aplicación puede redirigir el tráfico a una región diferente mediante Route 53 para realizar lecturas y escrituras en una tabla de réplicas 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.

Multi-Region la replicación Regiones de AWS está deshabilitada de forma predeterminada

La replicación multirregional de Amazon Keyspaces es compatible con las siguientes aplicaciones, Regiones de AWS que están deshabilitadas de forma predeterminada:

- Región África (Ciudad del Cabo)
- Región Medio Oriente (EAU)
- Región de Asia-Pacífico (Hong Kong)
- Región Medio Oriente (Baréin)

Antes de poder usar una región que está deshabilitada de forma predeterminada con la replicación multirregional de Amazon Keyspaces, primero debe habilitar la región. Para obtener más información, consulte [Habilitar o deshabilitar Regiones de AWS en su cuenta](#) en la Guía del [AWS Organizations usuario](#).

Tras activar una región, puede crear nuevos recursos de Amazon Keyspaces en la región y añadir la región a un espacio de claves multirregional.

Al deshabilitar una región que utiliza la replicación multirregional de Amazon Keyspaces, Amazon Keyspaces inicia un período de gracia de 24 horas. Durante este período de tiempo, puede esperar el siguiente comportamiento:

- Amazon Keyspaces sigue realizando operaciones de lenguaje de manipulación de datos (DML) en las regiones habilitadas.
- Amazon Keyspaces detiene la replicación de las actualizaciones de datos de las regiones habilitadas a la región deshabilitada.
- Amazon Keyspaces bloquea todas las solicitudes de lenguaje de definición de datos (DDL) en la región deshabilitada.

Si ha desactivado la región por error, puede volver a activarla en un plazo de 24 horas. Si vuelves a activar la región durante el período de gracia de 24 horas, Amazon Keyspaces tomará las siguientes medidas:

- Reanuda automáticamente todas las replicaciones en la región que se ha vuelto a activar.
- Replique cualquier actualización de datos que se haya realizado en las regiones habilitadas mientras la región estaba deshabilitada para garantizar la coherencia de los datos.

- Continúe automáticamente con todas las operaciones adicionales de replicación multirregional.

En el caso de que la región permanezca deshabilitada después de que se cierre el período de 24 horas, Amazon Keyspaces toma las siguientes medidas para eliminar permanentemente la región de la replicación multirregional:

- Elimine la región deshabilitada de todos los espacios de claves de replicación multirregional.
- Convierta las réplicas de tablas de replicación multirregionales de la región deshabilitada en tablas y espacios clave de una sola región.
- Amazon Keyspaces no elimina ningún recurso de la región deshabilitada.

Una vez que Amazon Keyspaces haya eliminado permanentemente la región deshabilitada del espacio de claves multirregión, no podrá volver a añadir la región deshabilitada.

Multi-Region replicación e integración con recuperación puntual (PITR)

Point-in-time La recuperación es compatible con las tablas de varias regiones. 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.
- La PITR debe estar habilitada en todas las réplicas de la tabla de origen.

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 la PITR, consulte [the section called “Funcionamiento”](#).

Al crear una tabla multirregional, la configuración de PITR que defina durante el proceso de creación se aplica automáticamente a todas las tablas de todas las regiones. Al cambiar la configuración del PITR mediante `ALTER TABLE` Amazon Keyspaces aplica la actualización únicamente a la tabla local y no a las réplicas de otras regiones. Para habilitar el PITR en una tabla multirregional existente, debe repetir la sentencia para todas las réplicas. `ALTER TABLE`

Multi-Region replicación e integración con servicios AWS

Puede monitorizar el rendimiento de la replicación entre tablas de diferentes tablas Regiones de AWS mediante el uso de CloudWatch las métricas de Amazon. 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 CloudWatch las métricas, consulte [the section called “Monitorización con CloudWatch”](#).

Notas de uso de la replicación multirregional de Amazon Keyspaces

Tenga en cuenta lo siguiente cuando utilice la replicación multirregional con Amazon Keyspaces.

- Puede seleccionar cualquiera de los públicos [disponibles](#). Regiones de AWS Para obtener más información sobre [los Regiones de AWS que están deshabilitados de forma predeterminada](#), consulte [the section called “Regiones deshabilitadas de forma predeterminada”](#).
- AWS GovCloud (US) Regions y las regiones de China no son compatibles.
- Considere las siguientes soluciones alternativas hasta que las características estén disponibles:

Configure el periodo de vida (TTL) al crear la tabla multirregional. No podrá activar ni desactivar el TTL ni ajustar el valor del TTL más adelante. Para obtener más información, consulte [the section called “Caducidad de datos con período de vida”](#).

- Para el cifrado en reposo, utilice una AWS clave propia. Las claves administradas por el cliente actualmente no son compatibles con las tablas multirregionales. Para obtener más información, consulte

[the section called “Funcionamiento”](#).

- Se puede utilizar ALTER KEYSPACE para añadir una región a un espacio de claves de una sola región o de varias regiones. Para obtener más información, consulte [the section called “Agregue una región a un espacio de claves”](#).
- Antes de añadir una región a un espacio de claves de una sola región, asegúrese de que ninguna tabla situada debajo del espacio de claves esté configurada con claves gestionadas por el cliente.

- Las etiquetas existentes configuradas para los espacios de claves o las tablas no se replican en la nueva región.
- Cuando utilice la administración de la capacidad aprovisionada con el escalado automático de Amazon Keyspaces, asegúrese de utilizar las operaciones de la API de Amazon Keyspaces para crear y configurar las tablas multirregionales. Las operaciones subyacentes de la API de Application Auto Scaling a las que Amazon Keyspaces llama en su nombre no tienen capacidades multirregionales.

Para obtener más información, consulte [the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional”](#). Para obtener más información sobre cómo estimar el rendimiento de la capacidad de escritura de las tablas multirregionales aprovisionadas, consulte [the section called “Estimación de la capacidad para una tabla multirregión”](#).

- Si bien los datos se replican automáticamente en las regiones seleccionadas de una tabla multirregional, 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` 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. La propagación de escrituras a las regiones replicadas se realiza entonces 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 multirregional, 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.
- Amazon CloudWatch proporciona una `ReplicationLatency` métrica para cada región replicada. Esta métrica se calcula observando las filas que llegan, comparando su tiempo de llegada con su tiempo de escritura inicial y calculando un promedio. Los tiempos se almacenan CloudWatch en la región de origen. Para obtener más información, consulte [the section called “Monitorización con CloudWatch”](#).

Puede resultar útil visualizar los tiempos medio y máximo para determinar el retraso de replicación promedio y el peor posible. No hay SLA para esta latencia.

- Al utilizar una tabla multirregional en el modo bajo demanda, es posible que observe un aumento en la latencia de la replicación asíncrona de escrituras si la réplica de una tabla experimenta un nuevo pico de tráfico. Al igual que Amazon Keyspaces adapta automáticamente la capacidad de una tabla bajo demanda de una sola región al tráfico de aplicaciones que recibe, Amazon Keyspaces adapta automáticamente la capacidad de una réplica de tabla bajo demanda multirregional al tráfico que reciba. El aumento de la latencia de replicación es transitorio porque Amazon Keyspaces asigna automáticamente más capacidad a medida que aumenta su volumen de tráfico. Una vez que todas las réplicas se hayan adaptado al volumen de tráfico, la latencia de la replicación debería volver a la normalidad. Para obtener más información, consulte [the section called “Picos de tráfico y propiedades de escalado”](#).
- Al utilizar una tabla multirregional en modo aprovisionado, si la aplicación supera la capacidad de rendimiento aprovisionada, es posible que se observen errores por capacidad insuficiente y un aumento en la latencia de replicación. Para garantizar que siempre haya suficiente capacidad de lectura y escritura para todas las réplicas de tablas Regiones de AWS de una tabla multirregional, le recomendamos que configure el escalado automático de Amazon Keyspaces. El escalado automático de Amazon Keyspaces 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. Para obtener más información, consulte [the section called “Funcionamiento del escalado automático para tablas multirregión”](#).

Configuración de la replicación multirregional para Amazon Keyspaces (para Apache Cassandra)

Puede utilizar la consola, Cassandra Query Language (CQL) o AWS Command Line Interface para crear y gestionar tablas y espacios clave multirregionales en Amazon Keyspaces.

En esta sección se proporcionan ejemplos de cómo crear y administrar espacios de claves multirregionales. 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 configuraciones y características compatibles, consulte [the section called “Notas de uso”](#).

Temas

- [Configuración de los permisos de IAM necesarios para crear espacios de claves y tablas multirregionales](#)
- [Configure los permisos de IAM necesarios para añadir una Región de AWS a un espacio de claves](#)
- [Creación de un espacio de claves multirregional en Amazon Keyspaces](#)
- [Añadir un Región de AWS a un espacio de claves en Amazon Keyspaces](#)
- [Compruebe el progreso de la replicación al añadir una nueva región a un espacio de claves](#)
- [Creación de una tabla multirregional con la configuración predeterminada en Amazon Keyspaces](#)
- [Creación de una tabla multirregional en modo aprovisionado con escalado automático en Amazon Keyspaces](#)
- [Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional en Amazon Keyspaces](#)
- [Visualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional en Amazon Keyspaces](#)
- [Desactivación del escalado automático de una tabla en Amazon Keyspaces](#)
- [Configuración manual de la capacidad aprovisionada de una tabla multirregional en Amazon Keyspaces](#)

Configuración de los permisos de IAM necesarios para crear espacios de claves y tablas multirregionales

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 “Multi-Region Replicación”](#).

Para crear la función vinculada al servicio que requiere la replicación multirregional, la política del director 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 AWS servicio al que se puede asignar 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.

Configure los permisos de IAM necesarios para añadir una Región de AWS a un espacio de claves

Para añadir una región a un espacio de claves, el director de IAM necesita los siguientes permisos:

- `cassandra:Alter`
- `cassandra:AlterMultiRegionResource`
- `cassandra:Create`
- `cassandra:CreateMultiRegionResource`
- `cassandra:Select`
- `cassandra:SelectMultiRegionResource`
- `cassandra:Modify`
- `cassandra:ModifyMultiRegionResource`

Si la tabla está configurada en modo aprovisionado con el escalado automático habilitado, se necesitan los siguientes permisos adicionales.

- `application-autoscaling:RegisterScalableTarget`
- `application-autoscaling:DeregisterScalableTarget`
- `application-autoscaling:DescribeScalableTargets`

- `application-autoscaling:PutScalingPolicy`
- `application-autoscaling:DescribeScalingPolicies`

Para añadir correctamente una región a un espacio de claves de una sola región, el director de IAM también debe poder crear un rol vinculado al servicio. 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 “Multi-Region Replicación”](#).

Para crear el rol vinculado al servicio que requiere la replicación multirregional, la política del director 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 AWS servicio al que se puede asignar este rol es Amazon Keyspaces.

El siguiente es un ejemplo de la política que concede los permisos mínimos necesarios a un director para añadir una región a un espacio de claves.

```
{
    "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"}}
}
```

Creación de un espacio de claves multirregional en Amazon Keyspaces

En esta sección se proporcionan ejemplos de cómo crear un espacio de claves multirregional. Puede hacerlo en la consola de Amazon Keyspaces, con CQL o la AWS CLI. Todas las tablas que cree en un espacio de claves multirregional heredan automáticamente la configuración multirregional del 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 “Multi-Region Replicación”](#).

Console

Creación de un espacio de claves multirregional (consola)

1. Inicie sesión en y abra la Consola de administración de AWS 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 Nombre del espacio de claves, introduzca el nombre del espacio de claves.
4. En la sección de Multi-Region replicación, puede añadir las regiones adicionales que están disponibles en la lista.
5. Para terminar, elija Crear espacio de claves.

Cassandra Query Language (CQL)

Creación de un espacio de claves multirregional con CQL

1. Para crear un espacio de claves multirregional, utilice esta `NetworkTopologyStrategy` opción para especificar en qué espacio de claves se va a replicar. Regiones de AWS Debe incluir su región actual y al menos una región adicional.

Todas las tablas del espacio de claves heredan la estrategia de replicación del espacio de claves. No puede cambiar la estrategia de replicación para las tablas.

`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.

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' };
```

2. Puede utilizar una instrucción CQL para consultar la tabla `tables` en el espacio de claves `system_multiregion_info` para crear mediante programación una lista de las regiones y el estado de la tabla multirregión que especifique. El siguiente código es un ejemplo de ello.

```
SELECT * from system_multiregion_info.tables WHERE keyspace_name = 'mykeyspace'
AND table_name = 'mytable';
```

La salida de la instrucción tiene un aspecto semejante al siguiente:

keyspace_name	table_name	region	status
mykeyspace	mytable	us-east-1	ACTIVE
mykeyspace	mytable	ap-southeast-1	ACTIVE
mykeyspace	mytable	eu-west-1	ACTIVE

CLI

Cree un nuevo espacio de claves multirregional mediante el AWS CLI

- Para crear un espacio de claves multirregional, 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
```

Para crear una tabla multirregional, consulte [the section called “Creación de una tabla multirregional con la configuración predeterminada”](#) y [the section called “Creación de una tabla multirregional en modo aprovisionado”](#).

Añadir un Región de AWS a un espacio de claves en Amazon Keyspaces

Puede añadir un nuevo espacio de claves Región de AWS a un espacio de claves único o multirregional. La nueva región de réplica se aplica a todas las tablas del espacio de claves.

Para cambiar un espacio de claves de una sola región a uno de varias regiones, debe habilitar las marcas de tiempo del lado del cliente para todas las tablas del espacio de claves. Para obtener más información, consulte [the section called “Client-side marcas de tiempo”](#).

Si va a añadir una región adicional a un espacio de claves multirregional, Amazon Keyspaces tiene que replicar las tablas existentes en la nueva región mediante una restauración única entre regiones para cada tabla existente. Los cargos de restauración de cada tabla se facturan por GB. Para obtener más información, consulte [Backup and restore](#) en la página de precios de Amazon Keyspaces (para Apache Cassandra). La transferencia de datos entre regiones para esta operación de restauración no conlleva ningún cargo. Además de los datos, todas las propiedades de la tabla, excepto las etiquetas, se replicarán en la nueva región.

Puede utilizar la ALTER KEYSPACE sentencia en CQL, el update-keyspace comando de o la AWS CLI consola para añadir una nueva región a un espacio de claves único o multirregional en Amazon Keyspaces. Para que el extracto se ejecute correctamente, la cuenta que utilice debe estar ubicada en una de las regiones en las que el espacio de claves ya esté disponible. Mientras se agrega la réplica, no puede realizar ninguna otra operación de lenguaje de definición de datos (DDL) en los recursos que se están actualizando y replicando.

Para obtener más información sobre los permisos necesarios para añadir una región, consulte [the section called “Configure los permisos de IAM para añadir una región”](#)

Note

Al añadir una región adicional a un espacio de claves de una sola región, Amazon Keyspaces crea un rol vinculado al servicio con el nombre de su cuenta. `AWSServiceRoleForAmazonKeyspacesReplication` Esta función permite a Amazon Keyspaces replicar tablas en nuevas regiones y replicar escrituras de una tabla a todas las réplicas de una tabla de varias regiones en su nombre. Para obtener más información, consulte [the section called “Multi-Region Replicación”](#).

Console

Siga estos pasos para añadir una región a un espacio de claves mediante la consola de Amazon Keyspaces.

Agregue una región a un espacio de claves (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Keyspaces y, a continuación, elija un espacio clave de la lista.
3. Elija la pestaña Regiones de AWS.
4. En la Regiones de AWS pestaña, elija Agregar región.
5. En el cuadro de diálogo Añadir región, elija la región adicional que desee añadir al espacio de claves.
6. Para terminar, elija Agregar.

Cassandra Query Language (CQL)

Añada una región a un espacio de claves mediante CQL

- Para añadir una nueva región a un espacio de claves, puede utilizar la siguiente declaración. En este ejemplo, el espacio de claves ya está disponible en las regiones EE.UU. Este (Norte de Virginia) y EE.UU. Oeste (Oregón), y la instrucción CQL agrega la región Región EE.UU. Oeste (Norte de California).

```
ALTER KEYSPACE my_keyspace
WITH REPLICATION = {
    'class': 'NetworkTopologyStrategy',
    'us-east-1': '3',
    'us-west-2': '3',
    'us-west-1': '3'
} AND CLIENT_SIDE_TIMESTAMPS = {'status': 'ENABLED'};
```

CLI

Añada una región a un espacio de claves mediante el AWS CLI

- Para agregar una nueva región a un espacio de claves mediante la CLI, puede usar el siguiente ejemplo. Tenga en cuenta que el valor predeterminado de `client-side-timestamps` es `DISABLED`. Con el `update-keyspace` comando, debe cambiar el valor a `ENABLED`.


```
aws keyspaces update-keyspace \
--keyspace-name my_keyspace \
--replication-specification '{"replicationStrategy": "MULTI_REGION",
"regionList": ["us-east-1", "eu-west-1", "eu-west-3"] }' \
--client-side-timestamps '{"status": "ENABLED"}'
```

Compruebe el progreso de la replicación al añadir una nueva región a un espacio de claves

Añadir una nueva región a un espacio de claves de Amazon Keyspaces es una operación de larga duración. Para realizar un seguimiento del progreso, puede utilizar las consultas que se muestran en esta sección.

Cassandra Query Language (CQL)

Uso de CQL para verificar el progreso de la adición de una región

- Para comprobar el progreso de la creación de las nuevas réplicas de tablas en un espacio de claves determinado, puede consultar la tabla `system_multiregion_info.keyspaces`. La siguiente instrucción CQL es un ejemplo de ello.

```
SELECT keyspace_name, region, status, tables_replication_progress
FROM system_multiregion_info.keyspaces
WHERE keyspace_name = 'my_keyspace';
```

Mientras hay una operación de replicación en curso, el estado muestra el progreso de la creación de la tabla en la nueva región. Este es un ejemplo en el que 5 de cada 10 tablas se han replicado en la nueva región.

keyspace_name	region	status	tables_replication_progress
my_keyspace	us-east-1	Updating	
my_keyspace	us-west-2	Updating	
my_keyspace	eu-west-1	Creating	50%

Una vez que el proceso de replicación se haya completado correctamente, el resultado debería tener el aspecto que se muestra en este ejemplo.

keyspace_name	region	status
my_keyspace	us-east-1	Active
my_keyspace	us-west-2	Active
my_keyspace	eu-west-1	Active

CLI

Se utiliza AWS CLI para verificar el progreso de la adición de la región

- Para confirmar el estado de la creación de la réplica de la tabla para un espacio de claves determinado, puede utilizar el siguiente ejemplo.

```
aws keyspaces get-keyspace \
--keyspace-name my_keyspace
```

La salida debe ser similar al siguiente ejemplo.

```
{
  "keyspaceName": "my_keyspace",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
  "replicationStrategy": "MULTI_REGION",
  "replicationRegions": [
    "us-east-1",
    "eu-west-1"
  ]
  "replicationGroupStatus": [
    {
      "RegionName": "us-east-1",
      "KeyspaceStatus": "Active"
    },
    {
      "RegionName": "eu-west-1",
      "KeyspaceStatus": "Creating",
      "TablesReplicationProgress": "50.0%"
    }
  ]
}
```

Creación de una tabla multirregional con la configuración predeterminada en Amazon Keyspaces

En esta sección se proporcionan ejemplos de cómo crear una tabla multirregional en el modo bajo demanda con todos los ajustes predeterminados. Puede hacerlo en la consola de Amazon Keyspaces, con CQL o la AWS CLI. Todas las tablas que cree en un espacio de claves multirregional heredan automáticamente la configuración multirregional del espacio de claves.

Para crear un espacio de claves multirregional, consulte [the section called “Creación de un espacio de claves multirregional”](#).

Console

Creación de una tabla multirregional con la configuración predeterminada (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 multirregionales.
 - Modo de capacidad: el modo de capacidad predeterminado es On-demand. Para obtener más información sobre la configuración del modo aprovisionado, consulte [the section called “Creación de una tabla multirregional en modo aprovisionado”](#).
 - Administración de claves de cifrado: solo se admite la opción Clave propiedad de AWS.
 - Client-side marcas de tiempo: esta función es necesaria para las tablas multirregionales.
 - 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.

Cassandra Query Language (CQL)

Creación de una tabla multirregional en el modo bajo demanda con la configuración predeterminada

- Para crear una tabla multirregional con la configuración predeterminada, puede utilizar la siguiente instrucción CQL.

```
CREATE TABLE mykeyspace.mytable(pk int, ck int, PRIMARY KEY (pk, ck))
  WITH CUSTOM_PROPERTIES = {
    'capacity_mode':{
      'throughput_mode':'PAY_PER_REQUEST'
    },
    'point_in_time_recovery':{
      'status':'enabled'
    },
    'encryption_specification':{
      'encryption_type':'AWS_OWNED_KMS_KEY'
    },
    'client_side_timestamps':{
      'status':'enabled'
    }
  };
```

CLI

Uso del AWS CLI

1. 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}{'
```

La salida del comando es:

```
{
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable"
}
```

2. Para confirmar la configuración de la tabla, puede usar la siguiente instrucción.

```
aws keyspaces get-table --keyspace-name mykeyspace --table-name mytable
```

Esta salida 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
      }
    }
  ]
}

```

Creación de una tabla multirregional en modo aprovisionado con escalado automático en Amazon Keyspaces

En esta sección se proporcionan ejemplos de cómo crear una tabla multirregional en el modo aprovisionado con escalado automático. Puede hacerlo en la consola de Amazon Keyspaces, con CQL o la AWS CLI.

Para obtener más información sobre las configuraciones compatibles y las funciones de replicación multirregional, consulte [the section called “Notas de uso”](#).

Para crear un espacio de claves multirregional, consulte [the section called “Creación de un espacio de claves multirregional”](#).

Al crear una nueva tabla multirregional en modo aprovisionado con una configuración de escalado automático, puede especificar una configuración general de la tabla que sea válida para todas las Regiones de AWS en las que se replica la tabla. A continuación, puede sobrescribir la configuración de capacidad de lectura y leer la configuración de escalado automático de cada réplica. Sin embargo, la capacidad de escritura permanece sincronizada en todas las réplicas para garantizar que haya suficiente capacidad para replicar las escrituras en todas las regiones.

 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 obtener más información, consulte [the section called “Cómo utilizar roles vinculados a servicios”](#).

Console

Creación de una nueva tabla multirregión con el escalado automático habilitado

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 para la tabla.
6. En la sección Clave principal, defina la clave principal de la tabla y seleccione las columnas de agrupación en clústeres opcionales.
7. En la sección Configuración de la tabla, elija Personalizar configuración.
8. Continúe con la configuración Read/write de capacidad.
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 Regiones de AWS en las que se replique 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 manera diferente, debe seleccionar las unidades de capacidad de lectura mínima y máxima para cada réplica de 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 verticalmente de manera automática.
- Desactive la casilla Escalado automático si desea aprovisionar la capacidad de lectura de la tabla manualmente. Este ajuste se aplica a todas las réplicas de la tabla.

 Note

Para garantizar que haya suficiente capacidad de lectura para todas las réplicas, le recomendamos el escalado automático de Amazon Keyspaces para las tablas multirregión 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 Escalado automático. A continuación, configure las unidades de capacidad de la tabla. Las unidades de capacidad de escritura permanecen sincronizadas en todas las Regiones de AWS para garantizar que haya suficiente capacidad para replicar los eventos de escritura en todas las regiones.
 - Desactive Escalado automático si desea aprovisionar la capacidad de escritura de la tabla manualmente. Este ajuste se aplica a todas las réplicas de la tabla.

 Note

Para garantizar que haya suficiente capacidad de escritura para todas las réplicas, le recomendamos el escalado automático de Amazon Keyspaces para las tablas multirregión aprovisionadas.

12. Seleccione Create table (Creación de tabla). Su tabla se crea con los parámetros de escalado automático especificados.

Cassandra Query Language (CQL)

Creación de una tabla multirregional con el modo de capacidad aprovisionada y el escalado automático mediante CQL

- Para crear una tabla multirregional en modo aprovisionado con escalado automático, primero debe especificar el modo de capacidad definiendo `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, consulte [the section called “Creación de una nueva tabla con escalado automático”](#).

Para definir la capacidad de lectura de una réplica de tabla en una región específica, puede configurar los siguientes parámetros como parte de las `replica_updates` de la tabla:

- La región
- Las unidades de capacidad de lectura aprovisionadas (opcional)
- La configuración de escalado automático de la capacidad de lectura (opcional)

En el siguiente ejemplo, se muestra una instrucción `CREATE TABLE` para una tabla multirregional en modo aprovisionado. La configuración general de escalado automático de la capacidad de escritura y lectura son los mismos. Sin embargo, la configuración del escalado automático de lectura especifica periodos de recuperación adicionales de 60 segundos antes de aumentar o reducir la capacidad de lectura de la tabla. Además, la configuración de escalado automático de la capacidad de lectura para la región Este de EE. UU. (Norte de Virginia) es superior a la de otras réplicas. Por otra parte, 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  
            }  
        }  
    }  
}  
};
```

CLI

Cree una nueva tabla multirregional en modo aprovisionado con escalado automático mediante el AWS CLI

- Para crear una tabla multirregional en modo aprovisionado con una configuración de escalado automático, puede usar la AWS CLI. Tenga en cuenta que debe usar el comando `create-table` de la CLI de Amazon Keyspaces para configurar los ajustes de escalado automático multirregionales. Esto se debe a que Application Auto Scaling, el servicio que Amazon Keyspaces utiliza para realizar el escalado automático por usted, no es compatible con 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, consulte [the section called “Creación de una nueva tabla con escalado automático”](#).

Para definir la capacidad de lectura de una réplica de tabla en una región específica, puede configurar los siguientes parámetros como parte de las `replicaSpecifications` de la tabla:

- La región
- Las unidades de capacidad de lectura aprovisionadas (opcional)
- La configuración de escalado automático de la capacidad de lectura (opcional)

Al crear tablas aprovisionadas multirregionales con ajustes de escalado automático 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` y `replication.json`. Tome nota de la ruta 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 para una tabla multirregional en Amazon Keyspaces

En esta sección 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 multirregionales aprovisionadas. Para obtener más información sobre las opciones de configuración generales de escalado automático y cómo funcionan, consulte [the section called “Administración de la capacidad de rendimiento con escalado automático”](#).

Tenga en cuenta que si utiliza el modo de capacidad aprovisionada para tablas multirregión, 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 lo son Region-aware.

Para obtener más información sobre cómo estimar el rendimiento de la capacidad de escritura de las tablas multirregionales aprovisionadas, consulte [the section called “Estimación de la capacidad para una tabla multirregión”](#).

Para obtener más información sobre la API de Amazon Keyspaces, consulte la [Referencia de la API de Amazon Keyspaces](#).

Al actualizar el modo aprovisionado o la configuración de escalado automático de una tabla multirregional, puede actualizar la configuración de la capacidad de lectura y la configuración del escalado automático de lectura para cada réplica de la tabla.

Sin embargo, la capacidad de escritura permanece sincronizada en todas las réplicas para garantizar que haya suficiente capacidad para replicar las escrituras en todas las regiones.

Cassandra Query Language (CQL)

Actualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional con CQL

- Puede utilizar `ALTER TABLE` para actualizar el modo de capacidad y la configuración de escalado automático de una tabla existente. Si va a actualizar una tabla que actualmente esté en modo de capacidad bajo demanda, necesita `capacity_mode`. Si la tabla ya está en el modo de capacidad aprovisionada, puede omitir este campo.

Para obtener información detallada sobre la configuración del escalado automático, la política de seguimiento de objetivos, el valor objetivo y la configuración opcional, consulte [the section called “Creación de una nueva tabla con escalado automático”](#).

En la misma instrucción, también puede actualizar la capacidad de lectura y la configuración de escalado automático de las réplicas de las tablas en regiones específicas actualizando la propiedad `replica_updates` 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
                    }
                }
            }
        }
    }
};

```

CLI

Actualice la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional mediante el AWS 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 de la 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 por usted, no es compatible con varias Regiones de AWS.

Para actualizar la capacidad de lectura de una réplica de tabla en una región específica, puede cambiar uno de los siguientes parámetros opcionales de las `replicaSpecifications` de la tabla:

- Las unidades de capacidad de lectura aprovisionadas (opcional)
- La configuración de escalado automático de la capacidad de lectura (opcional)

Al actualizar tablas multirregionales con ajustes de escalado automático 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` y `replication.json`. Tome nota de la ruta 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 para una tabla multirregional en Amazon Keyspaces

Puede ver la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional en la consola de Amazon Keyspaces, con CQL o con la AWS CLI. En esta sección se proporcionan ejemplos de cómo hacerlo con CQL y el AWS CLI

Cassandra Query Language (CQL)

Visualización de la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional con CQL

- Para ver la configuración de escalado automático de una tabla multirregional, utilice el siguiente comando.

```
SELECT * FROM system_multiregion_info.autoscaling WHERE keyspace_name =
'mykeyspace' AND table_name = 'mytable';
```

El resultado para este comando tendrá un aspecto similar al siguiente:

```
keyspace_name | table_name | region          |
provisioned_read_capacity_autoscaling_update

|
provisioned_write_capacity_autoscaling_update
-----+-----+-----
+-----+-----+-----
+-----+-----+-----
mykeyspace    | mytable    | ap-southeast-1 | {'minimum_units': 5,
'maximum_units': 10, 'scaling_policy':
{'target_tracking_scaling_policy_configuration': {'scale_out_cooldown':
60, 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown':
60}}} | {'minimum_units': 5, 'maximum_units': 10, 'scaling_policy':
{'target_tracking_scaling_policy_configuration': {'scale_out_cooldown': 0,
'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown': 0}}}
mykeyspace    | mytable    | us-east-1      | {'minimum_units': 5,
'maximum_units': 20, 'scaling_policy':
{'target_tracking_scaling_policy_configuration': {'scale_out_cooldown':
60, 'disable_scale_in': false, 'target_value': 70, 'scale_in_cooldown':
60}}} | {'minimum_units': 5, 'maximum_units': 10, 'scaling_policy':
{'target_tracking_scaling_policy_configuration': {'scale_out_cooldown': 0,
'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown': 0}}}
mykeyspace    | mytable    | eu-west-1      | {'minimum_units': 5,
'maximum_units': 10, 'scaling_policy':
{'target_tracking_scaling_policy_configuration': {'scale_out_cooldown':
60, 'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown':
60}}} | {'minimum_units': 5, 'maximum_units': 10, 'scaling_policy':
{'target_tracking_scaling_policy_configuration': {'scale_out_cooldown': 0,
'disable_scale_in': false, 'target_value': 50, 'scale_in_cooldown': 0}}}
```


CLI

Vea la capacidad aprovisionada y la configuración de escalado automático de una tabla multirregional mediante el AWS CLI

- Para ver la configuración de escalado automático de una tabla multirregional, puede utilizar la operación `get-table-auto-scaling-settings`. El siguiente comando de la CLI es un ejemplo de esto.

```
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
                }
            }
        }
    }
}

```

Desactivación del escalado automático de una tabla en Amazon Keyspaces

En esta sección se proporcionan ejemplos de cómo desactivar el escalado automático para una tabla multirregional en el modo de capacidad aprovisionada. Puede hacerlo en la consola de Amazon Keyspaces, con CQL o la AWS CLI.

Important

Se recomienda utilizar el escalado automático para las tablas multirregional que utilicen el modo de capacidad aprovisionada. Para obtener más información, consulte [the section called “Estimación de la capacidad para una tabla multirregional”](#).

 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.

Console

Desactivación del escalado automático de Amazon Keyspaces para una tabla multirregional existente en la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. Seleccione la tabla con la que quiera trabajar y elija la pestaña Capacidad.
3. En la sección Configuración de capacidad, elija Editar.
4. Para deshabilitar el escalado automático de Amazon Keyspaces, anule la selección de la casilla Escalado automático. 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 que utiliza Application Auto Scaling para acceder a la tabla Amazon Keyspaces, siga los pasos indicados en [the section called “Eliminación de un rol vinculado a servicios para Amazon Keyspaces”](#).
5. Cuando se hayan definido los ajustes de escalado automático, elija Guardar.

Cassandra Query Language (CQL)

Desactivación del escalado automático para una tabla multirregional mediante CQL

- 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 se desactiva el escalado automático de la capacidad de lectura de la tabla.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
    'provisioned_read_capacity_autoscaling_update': {
```

```
        'autoscaling_disabled': true
    }
};
```

CLI

Desactive el escalado automático de una tabla de varias regiones mediante la AWS 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 se desactiva el escalado automático de la capacidad de lectura de la tabla.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
    \ --auto-scaling-specification
    readCapacityAutoScaling={autoScalingDisabled=true}
```

Configuración manual de la capacidad aprovisionada de una tabla multirregional en Amazon Keyspaces

Si tiene que desactivar el escalado automático para una tabla multirregional, puede aprovisionar la capacidad de lectura de una tabla de réplica manualmente mediante CQL o la AWS CLI.

Note

Se recomienda utilizar el escalado automático para las tablas multirregional que utilicen el modo de capacidad aprovisionada. Para obtener más información, consulte [the section called “Estimación de la capacidad para una tabla multirregión”](#).

Cassandra Query Language (CQL)

Configuración manual de la capacidad aprovisionada de una tabla multirregional con CQL

- Puede utilizar `ALTER TABLE` para aprovisionar manualmente la capacidad de lectura de 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
        }
    }
};
```

CLI

Establezca manualmente la capacidad aprovisionada de una tabla multirregional mediante el AWS CLI

- Si tiene que desactivar el escalado automático para una tabla multirregional, puede usar `update-table` aprovisionar la capacidad de lectura de una tabla de réplica manualmente.

```
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
```

Copias de seguridad y restauración de datos con recuperación en un momento dado para Amazon Keyspaces

Point-in-time recovery (PITR) ayuda a proteger sus tablas de Amazon Keyspaces de operaciones de escritura o eliminación accidentales al proporcionarle copias de seguridad continuas de los datos de las 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 AWS SDK y el lenguaje de consultas AWS Command Line Interface (AWS CLI) o Cassandra Query Language (CQL). Para obtener más información, consulte [Uso de la recuperación en un momento dado en Amazon Keyspaces](#).

Point-in-time las operaciones no afectan al rendimiento ni a la disponibilidad de la tabla base, y la restauración de una tabla no consume un 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

- [Funcionamiento de la recuperación en un momento dado en Amazon Keyspaces](#)
- [Uso de la recuperación en un momento dado en Amazon Keyspaces](#)

Funcionamiento de la recuperación en un momento dado en Amazon Keyspaces

En esta sección se ofrece una descripción general del funcionamiento de la recuperación en un momento dado (PITR) de Amazon Keyspaces. Para obtener más información sobre precios, consulte [Precios de Amazon Keyspaces \(para Apache Cassandra\)](#).

Temas

- [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](#)
- [Restauración mediante PITR de tablas con tipos definidos por el usuario \(UDT\)](#)
- [Tiempo de restauración de la tabla con PITR](#)
- [Amazon Keyspaces, PITR e integración con servicios AWS](#)

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.

- Read/write modo de capacidad
- 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 escalado automático de la tabla. Puede sobrescribir esta configuración mediante el parámetro `autoscaling_settings` en CQL o con `autoScalingSpecification` en la CLI. Para obtener más información sobre la configuración de escalado automático, consulte [the section called “Administración de la capacidad de rendimiento con 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.

- Amazon Keyspaces cambia las transmisiones de captura de datos (CDC)
- AWS Identity and Access Management Políticas (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 se cifró con una Clave propiedad de AWS (opción predeterminada), se restaura automáticamente 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 clave administrada por el cliente, debe proporcionar una clave administrada por el cliente válida y accesible en el momento de la restauración.

Si quieres cambiar de una clave gestionada por el cliente a una Clave propiedad de AWS, confirma que Amazon Keyspaces tiene acceso a la clave gestionada 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 “Funcionamiento”](#).

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 las claves](#) en la Guía para AWS Key Management Service desarrolladores.

Restauración de tablas multirregión con PITR

Puede restaurar una tabla multirregión con PITR. Para que la operación de restauración se realice correctamente, la PITR debe estar habilitada en todas las réplicas de la tabla de origen y tanto la tabla de origen como la de destino deben estar replicadas en la misma. 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 replicación multirregional, consulte [the section called “Funcionamiento”](#)

Restauración mediante PITR de tablas con tipos definidos por el usuario (UDT)

Puede restaurar una tabla que utilice UDT. Para que la operación de restauración se realice correctamente, las UDT a las que se hace referencia deben existir y ser válidas en el espacio de claves.

Si falta alguna UDT requerida al intentar restaurar una tabla, Amazon Keyspaces intentará restaurar el esquema de UDT automáticamente y, a continuación, seguirá restaurando la tabla.

Si ha eliminado y vuelto a crear la UDT, Amazon Keyspaces restaura la UDT con el nuevo esquema de la UDT y rechaza la solicitud de restauración de la tabla con el esquema UDT original. En este caso, si desea restaurar la tabla con el esquema UDT anterior, puede restaurar la tabla en un espacio de claves nuevo. Al eliminar y volver a crear una UDT, incluso si el esquema de la UDT recreada es el mismo que el esquema de la UDT eliminada, la UDT recreada se considera una UDT nueva. En este caso, Amazon Keyspaces rechaza la solicitud de restaurar la tabla con el esquema UDT anterior.

Si falta la UDT y Amazon Keyspaces intenta restaurarla, el intento fallará si se ha alcanzado el número máximo de UDT para la cuenta en la región.

Para obtener más información sobre las cuotas de UDT y los valores predeterminados, consulte [the section called “Cuotas y valores predeterminados para los tipos definidos por el usuario \(UDT\) en](#)

[Amazon Keyspaces](#)” Para obtener más información sobre cómo trabajar con las UDT, consulte. [the section called “User-defined tipos \(UDT\)”](#)

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 ante 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.

Amazon Keyspaces, PITR e integración con servicios AWS

Las siguientes operaciones del PITR se registran AWS CloudTrail para permitir la supervisión y la auditoría continuas.

- 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 [Registrar las llamadas a la API de Amazon Keyspaces con AWS CloudTrail](#).

Puede realizar las siguientes acciones de PITR con 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 CloudFormation](#).

Uso de la recuperación en un momento dado en Amazon Keyspaces

Con Amazon Keyspaces (para Apache Cassandra), puede restaurar tablas a un punto específico en el tiempo mediante Point-in-Time Restore (PITR). La PITR le permite restaurar una tabla a un estado anterior de los últimos 35 días, lo que proporciona capacidades de protección y recuperación de datos. Esta característica resulta útil en casos como la eliminación accidental de datos, los errores de aplicación o la realización de pruebas. Puede recuperar los datos de forma rápida y eficiente, lo que minimiza el tiempo de inactividad y la pérdida de datos. En las siguientes secciones le guiamos a lo largo del proceso de restauración de tablas con la PITR en Amazon Keyspaces para garantizar la integridad de los datos y la continuidad empresarial.

Temas

- [Configuración de los permisos de IAM para la restauración de tablas con la PITR de Amazon Keyspaces](#)
- [Configuración de la PITR para una tabla en Amazon Keyspaces](#)
- [Desactivación de la PITR para una tabla de Amazon Keyspaces](#)
- [Restauración de una tabla desde una copia de seguridad en un momento dado en Amazon Keyspaces](#)
- [Restauración de una tabla eliminada con la PITR de Amazon Keyspaces](#)

Configuración de los permisos de IAM para la restauración de tablas con la PITR de Amazon Keyspaces

En esta sección se resume cómo configurar los permisos para que una entidad principal AWS Identity and Access Management (IAM) restaure las tablas de Amazon Keyspaces. En IAM, la política administrada AmazonKeyspacesFullAccess de AWS incluye los permisos para restaurar tablas de Amazon Keyspaces. Para implementar una política personalizada con los permisos mínimos requeridos, tenga en cuenta los requisitos que se describen en la siguiente sección.

Para restaurar correctamente una tabla, la entidad principal 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.

Este 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 punto de conexión de VPC y recuperación en un momento dado \(PITR\) de Amazon Keyspaces”](#).

Configuración de la PITR para una tabla en Amazon Keyspaces

Puede configurar una tabla en Amazon Keyspaces para las operaciones de copia de seguridad y restauración mediante la PITR con la consola, CQL y la AWS CLI.

Al crear una tabla nueva con CQL o el AWS CLI, debe habilitar explícitamente el PITR en la instrucción de creación de tabla. Al crear una tabla nueva con la consola, la PITR estará habilitada de forma predeterminada.

Para obtener información sobre cómo restaurar una tabla, consulte [the section called “Restauración de una tabla en un momento dado”](#).

Console

Configuración de la PITR para una tabla con la consola

1. Inicie sesión en y abra la Consola de administración de AWS 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 recuperación en un momento dado, selecciona Habilitar la recuperación. Point-in-time
5. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Configuración de la PITR para una tabla con CQL

1. Puede administrar la configuración de PITR para las tablas mediante la propiedad personalizada `point_in_time_recovery`.

Para habilitar la PITR en el momento de crear una nueva tabla, debe configurar el estado de `point_in_time_recovery` como `enabled`. Puede usar 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 propiedad de recuperación en un momento dado, la recuperación en un momento dado se deshabilita de forma predeterminada.

2. 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'}}
```

CLI

Configure el PITR para una tabla mediante el AWS CLI

1. Puede administrar la configuración de PITR para las tablas con la API UpdateTable.

Para habilitar la PITR en el momento de crear una nueva tabla, debe incluir `point-in-time-recovery 'status=ENABLED'` en el comando de creación de la tabla. Puede usar 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 valor de recuperación en un momento dado, la recuperación en un momento dado se deshabilita de forma predeterminada.

2. Para confirmar la configuración de recuperación puntual de una tabla, puede utilizar el siguiente comando. AWS CLI

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

3. Para habilitar el PITR en una tabla existente mediante el AWS CLI, ejecute el siguiente comando.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=ENABLED'
```

Desactivación de la PITR para una tabla de Amazon Keyspaces

Puede desactivar la PITR para una tabla de Amazon Keyspaces en cualquier momento mediante la consola, CQL o la AWS CLI.

 Important

Al deshabilitar la PITR se elimina de inmediato el historial de copias de seguridad, sin importar si vuelve a habilitar la PITR en la tabla en un plazo de 35 días.

Para obtener información sobre cómo restaurar una tabla, consulte [the section called “Restauración de una tabla en un momento dado”](#).

Console

Desactivación de la PITR para una tabla con la consola

1. Inicie sesión en y abra la Consola de administración de AWS 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 recuperación en un momento dado, desactive la casilla Habilitar la Point-in-time recuperación.
5. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Desactivación de la PITR para una tabla con CQL

- Para deshabilitar la PITR para una tabla existente, ejecute el siguiente comando CQL.

```
ALTER TABLE mykeyspace.mytable  
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

CLI

Desactive el PITR para una tabla mediante el AWS CLI

- Para deshabilitar la PITR en una tabla existente, ejecute el siguiente AWS CLI comando.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'  
--point-in-time-recovery 'status=DISABLED'
```

Restauración de una tabla desde una copia de seguridad en un momento dado en Amazon Keyspaces

En la siguiente sección se muestra cómo restaurar una tabla existente de Amazon Keyspaces en un momento dado especificado.

Note

En este procedimiento, se presupone que en la tabla que utiliza se ha configurado la recuperación en un momento dado. Para habilitar la PITR en una tabla, consulte [the section called “Configuración de la PITR”](#).

 Important

Mientras se esté realizando una restauración, no modifique ni elimine las políticas AWS Identity and Access Management (de IAM) que otorgan permiso al director de IAM (por ejemplo, al usuario, grupo o rol) para realizar la restauración. De lo contrario, puede ocurrir un comportamiento inesperado. Por ejemplo, si se quitan los permisos de escritura de una tabla mientras se restaura, la operación `RestoreTableToPointInTime` subyacente 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.

Console

Restauración de una tabla en un momento dado mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS 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 Tablas.
3. En la lista de tablas, elija la tabla que quiera restaurar.
4. En la pestaña Copias de seguridad de la tabla, en la sección Point-in-time de recuperación, selecciona Restaurar.
5. Para el nombre de la nueva tabla, escriba un nombre para la tabla restaurada, por ejemplo, **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 Restaurando. Una vez finalizado el proceso de restauración, el estado de la tabla restaurada cambia a Activa.

Cassandra Query Language (CQL)

Restauración de una tabla en un momento dado con CQL

1. Puede restaurar una tabla activa en 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, consulte la `system_schema_mcs.tables` como se indica en este ejemplo.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Point-in-time la recuperación 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 la hora actual. Si se omite la cláusula `WITH restore_timestamp = ...`, se utiliza la marca de tiempo actual.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

- También puede restaurar a un punto específico en el tiempo, definido por una `restore_timestamp` en formato ISO 8601. 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.

3. 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 Restaurando. Una vez finalizado el proceso de restauración, el estado de la tabla cambia a Activa.

CLI

Restaurar una tabla a un punto en el tiempo mediante el AWS CLI

1. Cree 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-1:111122223333:/keyspace/
myKeyspace/table/myTable",
  "creationTimestamp": "2022-06-20T14:34:57.049000-07:00",
  "status": "ACTIVE",
  "schemaDefinition": {
    "allColumns": [
      {
        "name": "id",
        "type": "int"
      },
      {
        "name": "date",
        "type": "timestamp"
      },
      {
        "name": "name",
        "type": "text"
      }
    ],
    "partitionKeys": [
      {
        "name": "id"
      }
    ],
    "clusteringKeys": [],
    "staticColumns": []
  },
  "capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": "2022-06-20T14:34:57.049000-07:00"
  },
  "encryptionSpecification": {
```

```

    "type": "AWS_OWNED_KMS_KEY"
  },
  "pointInTimeRecovery": {
    "status": "ENABLED",
    "earliestRestorableTimestamp": "2022-06-20T14:35:13.693000-07:00"
  },
  "defaultTimeToLive": 0,
  "comment": {
    "message": ""
  }
}

```

3. • Para restaurar una tabla en 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-1:111122223333:/keyspace/
myKeyspace/table/myTable_restored"
}

```

- Para restaurar la tabla a la hora actual, puede omitir el parámetro `restore-timestamp`.

```

aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
'myTable_restored1'

```

Restauración de una tabla eliminada con la PITR de Amazon Keyspaces

En el siguiente procedimiento se muestra cómo restaurar una tabla eliminada desde una copia de seguridad al momento de su eliminación. Puede hacerlo mediante CQL o el AWS CLI

 Note

En este procedimiento se asume que la PITR estaba habilitado en la tabla eliminada.

Cassandra Query Language (CQL)

Restauración de una tabla eliminada con CQL

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;
```

CLI

Restaure una tabla eliminada mediante el AWS CLI

1. Elimine una tabla que haya creado anteriormente y que tenga la PITR habilitada. El siguiente comando es un ejemplo.

```
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-1:111122223333:/keyspace/myKeyspace/table/myTable_restored2"
}
```

Caducidad de datos con período de vida (TTL) para Amazon Keyspaces (para Apache Cassandra)

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, puedes usar el TTL en una AdTech aplicación para programar cuándo caducan los datos de anuncios específicos y cuándo dejan de estar 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.

Puede establecer, modificar o deshabilitar la configuración de TTL predeterminada para tablas nuevas y existentes mediante la consola, Cassandra Query Language (CQL) o la AWS CLI.

En las tablas con TTL predeterminado configurado, puede utilizar instrucciones CQL para anular la configuración TTL predeterminada de la tabla y aplicar valores de TTL personalizados a filas y columnas. Para obtener más información, consulte [the section called “Uso de INSERT para establecer un TTL personalizado para nuevas filas”](#) y [the section called “Uso de UPDATE para establecer TTL personalizados para filas y columnas”](#).

Los precios de TTL se basan en el tamaño de las filas que se eliminan 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

- [Amazon Keyspaces Time to Live e integración con los servicios AWS](#)
- [Creación de una nueva tabla con la configuración de período de vida \(TTL\) predeterminada](#)
- [Actualización del valor predeterminado del período de vida \(TTL\) de una tabla](#)
- [Creación de una tabla con una configuración personalizada del período de vida \(TTL\) habilitada](#)
- [Actualización de una tabla con un período de vida \(TTL\) personalizado](#)
- [Uso de la instrucción INSERT para establecer valores del período de vida \(TTL\) personalizados para filas nuevas](#)
- [Uso de la instrucción UPDATE para editar la configuración personalizada del período de vida \(TTL\) para filas y columnas](#)

Amazon Keyspaces Time to Live e integración con los servicios AWS

La siguiente métrica TTL está disponible en Amazon CloudWatch para permitir la supervisión continua.

- `TTLDeletes`: las unidades consumidas para eliminar o actualizar datos en una fila utilizando Tiempo de vida (TTL).

Para obtener más información sobre cómo supervisar CloudWatch las métricas, consulte [the section called “Monitorización con CloudWatch”](#).

Si lo usas CloudFormation, puedes activar el TTL al crear una tabla de Amazon Keyspaces. Para obtener más información, consulte la [Guía del usuario de AWS CloudFormation](#).

Creación de una nueva tabla con la configuración de período de vida (TTL) predeterminada

En Amazon Keyspaces, puede establecer un valor de TTL predeterminado para todas las filas de una tabla al crearla.

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.

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.

Después de crear la tabla, puede sobrescribir el valor de TTL predeterminado de la tabla para filas o columnas específicas con instrucciones CQL de DML. Para obtener más información, consulte [the section called “Uso de INSERT para establecer un TTL personalizado para nuevas filas”](#) y [the section called “Uso de UPDATE para establecer TTL personalizados para filas y columnas”](#).

Al habilitar el TTL en una tabla, Amazon Keyspaces comienza a almacenar metadatos TTL-related adicionales 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.

En los siguientes ejemplos se muestra cómo crear una nueva tabla con un valor de TTL predeterminado.

Console

Creación de una tabla nueva con un valor predeterminado de período de vida mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 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 de TTL predeterminada para la tabla.

En Periodo de 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. Seleccione **Create table** (Creación de tabla). Su tabla se crea con el valor de TTL predeterminado especificado.

Cassandra Query Language (CQL)

Creación de una tabla nueva con un valor de TTL predeterminado mediante CQL

1. La siguiente instrucción cree 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;
```

2. 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`.

```
DESC TABLE my_table;
```

```
CREATE TABLE my_keyspace.my_table (  
    userid uuid,  
    time timeuuid,  
    body text,  
    subject text,  
    user inet,  
    PRIMARY KEY (userid, time)  
    ) WITH CLUSTERING ORDER BY (time ASC)  
    AND bloom_filter_fp_chance = 0.01  
    AND caching = {'class': 'com.amazonaws.cassandra.DefaultCaching'}  
    AND comment = ''  
    AND compaction = {'class': 'com.amazonaws.cassandra.DefaultCompaction'}  
    AND compression = {'class': 'com.amazonaws.cassandra.DefaultCompression'}  
    AND crc_check_chance = 1.0  
    AND dclocal_read_repair_chance = 0.0  
    AND default_time_to_live = 3024000
```

```
AND gc_grace_seconds = 7776000
AND max_index_interval = 2048
AND memtable_flush_period_in_ms = 3600000
AND min_index_interval = 128
AND read_repair_chance = 0.0
AND speculative_retry = '99PERCENTILE';
```

CLI

Cree una tabla nueva con un valor TTL predeterminado mediante el AWS CLI

1. Puede utilizar el siguiente comando para crear una tabla nueva con el valor de TTL predeterminado establecido en un año.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable' \
    --schema-definition 'allColumns=[{name=id,type=int},
{name=name,type=text},{name=date,type=timestamp}],partitionKeys=[{name=id}]' \
    --default-time-to-live '31536000'
```

2. Para confirmar el estado de TTL de la tabla, puede utilizar el siguiente comando.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

El resultado del comando tendrá un aspecto similar al del ejemplo siguiente

```
{
  "keyspaceName": "myKeyspace",
  "tableName": "myTable",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
  "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
  "status": "ACTIVE",
  "schemaDefinition": {
    "allColumns": [
      {
        "name": "id",
        "type": "int"
      },
      {
        "name": "date",
        "type": "timestamp"
      }
    ]
  }
}
```

```

        },
        {
            "name": "name",
            "type": "text"
        }
    ],
    "partitionKeys": [
        {
            "name": "id"
        }
    ],
    "clusteringKeys": [],
    "staticColumns": []
},
"capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": "2024-09-02T10:52:22.190000+00:00"
},
"encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
    "status": "DISABLED"
},
"ttl": {
    "status": "ENABLED"
},
"defaultTimeToLive": 31536000,
"comment": {
    "message": ""
},
"replicaSpecifications": []
}

```

Actualización del valor predeterminado del período de vida (TTL) de una tabla

Puede actualizar una tabla existente con un nuevo valor de TTL predeterminado. 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.

Al habilitar el TTL en una tabla, Amazon Keyspaces comienza a almacenar metadatos TTL-related adicionales 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 habilitado el TTL para una tabla, puede sobrescribir el valor de TTL predeterminado de la tabla para filas o columnas específicas con instrucciones CQL de DML. Para obtener más información, consulte [the section called “Uso de INSERT para establecer un TTL personalizado para nuevas filas”](#) y [the section called “Uso de UPDATE para establecer TTL personalizados para filas y columnas”](#).

Una vez habilitada la característica de 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.

Siga estos pasos para actualizar la configuración predeterminada del período de vida (TTL) para tablas existentes con la consola, CQL o la AWS CLI.

Console

Actualización del valor de TTL predeterminado de una tabla mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS 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 de TTL predeterminado, introduzca el tiempo de caducidad y elija la unidad de tiempo, 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.

Cassandra Query Language (CQL)

Actualización del valor de TTL predeterminado de una tabla mediante CQL

1. Puede utilizar `ALTER TABLE` para editar la configuración predeterminada del período de vida (TTL) de una tabla. Para actualizar la configuración del TTL predeterminada de la tabla a 2 592 000 segundos, lo que representa 30 días, puede utilizar la siguiente instrucción.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

2. Para confirmar la configuración del 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`.

```
DESC TABLE my_table;
```

La salida de la instrucción debe ser similar al siguiente ejemplo.

```
CREATE TABLE my_keyspace.my_table (  
    id int PRIMARY KEY,  
    date timestamp,  
    name text  
) WITH bloom_filter_fp_chance = 0.01  
    AND caching = {'class': 'com.amazonaws.cassandra.DefaultCaching'}  
    AND comment = ''  
    AND compaction = {'class': 'com.amazonaws.cassandra.DefaultCompaction'}  
    AND compression = {'class': 'com.amazonaws.cassandra.DefaultCompression'}  
    AND crc_check_chance = 1.0  
    AND dclocal_read_repair_chance = 0.0  
    AND default_time_to_live = 2592000  
    AND gc_grace_seconds = 7776000  
    AND max_index_interval = 2048  
    AND memtable_flush_period_in_ms = 3600000  
    AND min_index_interval = 128  
    AND read_repair_chance = 0.0  
    AND speculative_retry = '99PERCENTILE';
```

CLI

Actualice el valor TTL predeterminado de una tabla mediante el AWS CLI

1. Puede utilizar `update-table` para editar el valor de TTL predeterminado de una tabla. Para actualizar la configuración del TTL predeterminada de la tabla a 2 592 000 segundos, lo que representa 30 días, puede utilizar la siguiente instrucción.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
--default-time-to-live '2592000'
```

2. Para confirmar el valor de TTL predeterminado actualizado, puede utilizar la siguiente instrucción.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

La salida de la instrucción debería tener un aspecto semejante al del siguiente ejemplo.

```
{
  "keyspaceName": "myKeyspace",
  "tableName": "myTable",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
  "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
  "status": "ACTIVE",
  "schemaDefinition": {
    "allColumns": [
      {
        "name": "id",
        "type": "int"
      },
      {
        "name": "date",
        "type": "timestamp"
      },
      {
        "name": "name",
        "type": "text"
      }
    ],
    "partitionKeys": [
      {
```

```
        "name": "id"
      }
    ],
    "clusteringKeys": [],
    "staticColumns": []
  },
  "capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": "2024-09-02T10:52:22.190000+00:00"
  },
  "encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
  },
  "pointInTimeRecovery": {
    "status": "DISABLED"
  },
  "ttl": {
    "status": "ENABLED"
  },
  "defaultTimeToLive": 2592000,
  "comment": {
    "message": ""
  },
  "replicaSpecifications": []
}
```

Creación de una tabla con una configuración personalizada del período de vida (TTL) habilitada

Para crear una nueva tabla con una configuración personalizada del período de vida que se pueda aplicar a filas y columnas sin habilitar la configuración predeterminada del TTL para toda la tabla, puede utilizar los siguientes comandos.

Note

Si se crea una tabla con una configuración personalizada del `ttl` habilitada, no se podrá desactivar la configuración más adelante.

Cassandra Query Language (CQL)

Creación de una tabla nueva con una configuración de TTL personalizada mediante CQL

- ```
CREATE TABLE my_keyspace.my_table (id int primary key) WITH
CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

## CLI

Cree una tabla nueva con una configuración TTL personalizada mediante el AWS CLI

1. Puede utilizar el siguiente comando para crear una nueva tabla con el TTL habilitado.

```
aws keyspaces create-table --keyspace-name 'myKeyspace' --table-name 'myTable' \
 --schema-definition
'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]]' \
 --ttl 'status=ENABLED'
```

2. Para confirmar que el TTL está habilitado para la tabla, puede utilizar la siguiente instrucción.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

La salida de la instrucción debería tener un aspecto semejante al del siguiente ejemplo.

```
{
 "keyspaceName": "myKeyspace",
 "tableName": "myTable",
 "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
 "creationTimestamp": "2024-09-02T10:52:22.190000+00:00",
 "status": "ACTIVE",
 "schemaDefinition": {
 "allColumns": [
 {
 "name": "id",
 "type": "int"
 },
 {
 "name": "date",
 "type": "timestamp"
 }
]
 }
}
```

```

 },
 {
 "name": "name",
 "type": "text"
 }
],
 "partitionKeys": [
 {
 "name": "id"
 }
],
 "clusteringKeys": [],
 "staticColumns": []
},
"capacitySpecification": {
 "throughputMode": "PAY_PER_REQUEST",
 "lastUpdateToPayPerRequestTimestamp": "2024-09-02T11:18:55.796000+00:00"
},
"encryptionSpecification": {
 "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
 "status": "DISABLED"
},
"ttl": {
 "status": "ENABLED"
},
"defaultTimeToLive": 0,
"comment": {
 "message": ""
},
"replicaSpecifications": []
}

```

## Actualización de una tabla con un período de vida (TTL) personalizado

Para habilitar la configuración personalizada del período de vida para una tabla de modo que los valores del TTL se puedan aplicar a filas y columnas individuales sin habilitar un valor del TTL predeterminado para toda la tabla, puede utilizar los siguientes comandos.

**Note**

Una vez habilitado, el `tTL` no se puede deshabilitar para la tabla.

## Cassandra Query Language (CQL)

Habilitación de la configuración del TTL personalizada para una tabla mediante CQL

- ```
ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

CLI

Habilite la configuración TTL personalizada para una tabla mediante el AWS CLI

1. Puede utilizar el siguiente comando para actualizar el valor del TTL personalizado de una tabla.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --ttl 'status=ENABLED'
```

2. Para confirmar que el TTL ya está habilitado para la tabla, puede utilizar la siguiente instrucción.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

La salida de la instrucción debería tener un aspecto semejante al del siguiente ejemplo.

```
{
  "keyspaceName": "myKeyspace",
  "tableName": "myTable",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable",
  "creationTimestamp": "2024-09-02T11:32:27.349000+00:00",
  "status": "ACTIVE",
  "schemaDefinition": {
    "allColumns": [
      {
        "name": "id",
        "type": "int"
```

```
    },
    {
      "name": "date",
      "type": "timestamp"
    },
    {
      "name": "name",
      "type": "text"
    }
  ],
  "partitionKeys": [
    {
      "name": "id"
    }
  ],
  "clusteringKeys": [],
  "staticColumns": []
},
"capacitySpecification": {
  "throughputMode": "PAY_PER_REQUEST",
  "lastUpdateToPayPerRequestTimestamp": "2024-09-02T11:32:27.349000+00:00"
},
"encryptionSpecification": {
  "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
  "status": "DISABLED"
},
"ttl": {
  "status": "ENABLED"
},
"defaultTimeToLive": 0,
"comment": {
  "message": ""
},
"replicaSpecifications": []
}
```


Uso de la instrucción **INSERT** para establecer valores del período de vida (TTL) personalizados para filas nuevas

Note

Antes de poder establecer valores de TTL personalizados para las filas mediante la instrucción INSERT, primero debe habilitar el TTL personalizado en la tabla. Para obtener más información, consulte [the section called “Actualización de una tabla con un TTL personalizado”](#).

Para sobrescribir el valor de TTL predeterminado de una tabla mediante el establecimiento de fechas de caducidad para filas individuales, puede utilizar la instrucción INSERT:

- INSERT: inserta una nueva fila de datos con un valor de TTL establecido.

El establecimiento de valores de TTL para nuevas filas con la instrucción INSERT tiene prioridad sobre el TTL predeterminado para la tabla.

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;
```

Uso de la instrucción **UPDATE** para editar la configuración personalizada del período de vida (TTL) para filas y columnas

Note

Antes de poder establecer valores de TTL personalizados para filas y columnas, primero se debe habilitar el TTL en la tabla. Para obtener más información, consulte [the section called “Actualización de una tabla con un TTL personalizado”](#).

Para sobrescribir el valor de TTL predeterminado de una tabla mediante el establecimiento de fechas de caducidad para filas y columnas individuales, puede utilizar la instrucción UPDATE:

- Filas: puede actualizar una fila de datos existente con un valor de TTL personalizado.
- Columnas: puede actualizar un subconjunto de columnas dentro de las filas existentes con un valor de TTL personalizado.

El establecimiento de valores de TTL para filas y columnas tiene prioridad sobre los ajustes de TTL predeterminados para la tabla.

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	subject	time	body
B79CB3BA-745E-5D9A-8903-4A02327A7E09	Updated Message	96a29100-5e25-11ec-90d7-b5d91eceda0a	

```
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
```

Uso de este servicio 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 para C++	AWS SDK para C++ ejemplos de código
AWS CLI	AWS CLI ejemplos de código
AWS SDK para Go	AWS SDK para Go ejemplos de código
AWS SDK para Java	AWS SDK para Java ejemplos de código
AWS SDK para JavaScript	AWS SDK para JavaScript ejemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin ejemplos de código

Documentación de SDK	Ejemplos de código
AWS SDK para .NET	AWS SDK para .NET ejemplos de código
AWS SDK para PHP	AWS SDK para PHP ejemplos de código
Herramientas de AWS para PowerShell	Herramientas de AWS para PowerShell ejemplos de código
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) ejemplos de código
AWS SDK para Ruby	AWS SDK para 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 Enviar comentarios que se encuentra al final de esta página.

Trabajo con etiquetas 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, ya que le permite crear informes sobre los servicios que llevan una etiqueta determinada.

Para empezar a usar etiquetas, haga lo siguiente:

1. Comprender [Restricciones de uso de etiquetas para rotular recursos de Amazon Keyspaces..](#)
2. Crear etiquetas mediante [Etiquete espacios clave, tablas y transmisiones en Amazon Keyspaces.](#)
3. Úselo [Creación de informes de asignación de costos con etiquetas 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 uso de etiquetas para rotular recursos de 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, tabla o transmisión de Amazon Keyspaces solo puede tener una 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 que se encuentran 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.
- Las etiquetas aplicadas a una tabla no se aplican automáticamente a la secuencia de esa tabla. Para aplicar las mismas etiquetas a una tabla y a la secuencia durante la creación de la tabla, puede usar la `PropagateTagsOnEnable` marca al crear la tabla. Con este indicador, Amazon Keyspaces aplica las etiquetas de la tabla a la transmisión durante la creación de la transmisión. Cuando la transmisión está activa, los cambios en las etiquetas de la tabla no se aplican a la transmisión.
- 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.
- AWS-a los nombres y valores de las etiquetas asignadas se les asigna automáticamente el aws : prefijo, que usted no puede asignar. AWS-los nombres de etiqueta asignados no cuentan para el límite de 50 etiquetas. User-assigned los nombres de las etiquetas llevan el prefijo user : en el informe de asignación de costes.
- No es posible antedatar la aplicación de una etiqueta.

Etiquete espacios clave, tablas y transmisiones en Amazon Keyspaces

Puede añadir, enumerar, editar o eliminar etiquetas para espacios clave, tablas y secuencias mediante la consola Amazon Keyspaces, el lenguaje de consultas Cassandra o Cassandra Query Language (CQL). AWS CLI A continuación, puede activar estas etiquetas definidas por el usuario para que aparezcan en la Administración de facturación y costos de AWS consola a fin de realizar un seguimiento de la asignación de costes. Para obtener más información, consulte [Creación de informes de asignación de costos con etiquetas 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 Grupos de recursos de AWS .

Para obtener más información sobre la estructura de las etiquetas, consulte [Restricciones de uso de etiquetas para rotular recursos de Amazon Keyspaces](#)..

Temas

- [Aplicación de etiquetas al crear un nuevo espacio de claves](#)
- [Aplicación de etiquetas a un espacio de claves](#)
- [Eliminación de etiquetas de un espacio de claves](#)

- [Visualización de las etiquetas de un espacio de claves](#)
- [Aplicación de etiquetas al crear una nueva tabla](#)
- [Aplicación de etiquetas a una tabla](#)
- [Eliminación de etiquetas de una tabla](#)
- [Visualización de las etiquetas de una tabla](#)
- [Agregue etiquetas a una nueva secuencia al crear una tabla](#)
- [Agregue etiquetas a una nueva transmisión para una tabla existente](#)
- [Añadir nuevas etiquetas a una transmisión](#)
- [Eliminar etiquetas de una transmisión](#)
- [Ver las etiquetas de una transmisión](#)

Aplicación de etiquetas al crear un nuevo espacio de claves

Puede utilizar la consola Amazon Keyspaces, CQL o la AWS CLI para añadir etiquetas al crear un nuevo espacio de claves.

Console

Aplicación de una etiqueta al crear un nuevo espacio de claves mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS 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.
4. En Etiquetas, seleccione Agregar nueva etiqueta e introduzca una clave y un valor.
5. Elija Crear espacio de claves.

Cassandra Query Language (CQL)

Aplicación de una etiqueta al crear un nuevo espacio de claves mediante CQL

- En el siguiente ejemplo se crea un nuevo espacio de claves con etiquetas.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

CLI

Establezca una etiqueta al crear un nuevo espacio de claves mediante el AWS CLI

- Con la siguiente instrucción se crea un nuevo espacio de claves con etiquetas.

```
aws keyspaces create-keyspace --keyspace-name 'myKeyspace' --tags  
'key=key1,value=val1' 'key=key2,value=val2'
```

Aplicación de etiquetas a un espacio de claves

En los siguientes ejemplos se muestra cómo aplicar etiquetas a un espacio de claves de Amazon Keyspaces.

Console

Aplicación de una etiqueta a un espacio de claves existente con la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Espacios de claves.
3. Seleccione un espacio de claves de la lista. A continuación, elija la pestaña Etiquetas, donde podrá ver las etiquetas del espacio de claves.
4. Seleccione Administrar etiquetas para agregar, editar o eliminar etiquetas.
5. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Aplicación de una etiqueta a un espacio de claves existente con CQL

- ```
ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```



## CLI

Añada una etiqueta a un espacio de claves existente mediante el AWS CLI

- En el siguiente ejemplo se muestra cómo añadir nuevas etiquetas a un espacio de claves existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

## Eliminación de etiquetas de un espacio de claves

### Console

Eliminación de una etiqueta a un espacio de claves existente con la consola

- Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
- En el panel de navegación, elija Espacios de claves.
- Seleccione un espacio de claves de la lista. A continuación, elija la pestaña Etiquetas, donde podrá ver las etiquetas del espacio de claves.
- Seleccione Administrar etiquetas y elimine las etiquetas que ya no necesite.
- Seleccione Save changes (Guardar cambios).

### Cassandra Query Language (CQL)

Eliminación de una etiqueta a un espacio de claves existente con CQL

- ```
ALTER KEYSPACE mykeyspace DROP TAGS {'key1':'val1', 'key2':'val2'};
```

CLI

Elimine una etiqueta de un espacio de claves existente mediante el AWS CLI

- La siguiente instrucción elimina las etiquetas especificadas de un espacio de claves.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Visualización de las etiquetas de un espacio de claves

En los siguientes ejemplos se muestra cómo leer etiquetas mediante la consola, CQL o la AWS CLI.

Console

Visualización de las etiquetas de espacios de claves mediante la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Espacios de claves.
3. Seleccione un espacio de claves de la lista. A continuación, elija la pestaña Etiquetas, donde podrá ver las etiquetas del espacio de claves.

Cassandra Query Language (CQL)

Visualización de las etiquetas de un espacio de claves con CQL

Para leer las etiquetas adjuntas a un espacio de claves, utilice la siguiente instrucción CQL.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

La cláusula WHERE es obligatoria y debe usar uno de los siguientes formatos:

- `keyspace_name = 'mykeyspace' AND resource_type = 'keyspace'`
- `resource_id = arn`
- La siguiente instrucción 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-1:111122223333:/keyspace/mykeyspace/ | mykeyspace
| mykeyspace | keyspace | {'key1': 'val1', 'key2': 'val2'}

```

CLI

Vea las etiquetas de un espacio de claves mediante el AWS CLI

- En este ejemplo se muestra cómo enumerar las etiquetas del recurso especificado.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/'
```

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"
    }
  ]
}

```

Aplicación de etiquetas al crear una nueva tabla

Puede utilizar la consola Amazon Keyspaces, CQL o la AWS CLI para añadir etiquetas a las tablas nuevas al crearlas.

Console

Aplicación de una etiqueta al crear una nueva tabla con la (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 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. Seleccione Create table (Creación de tabla).

Cassandra Query Language (CQL)

Aplicación de etiquetas al crear una nueva tabla con CQL

- En el siguiente ejemplo se crea una nueva tabla con etiquetas.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

CLI

Añada etiquetas al crear una tabla nueva mediante el AWS CLI

- 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'
```

Aplicación de etiquetas a una tabla

Puede agregar etiquetas a una tabla existente en Amazon Keyspaces con la consola, CQL o la AWS CLI.

Console

Aplicación de etiquetas a una tabla mediante la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Tablas.
3. Seleccione una tabla de la lista y seleccione la pestaña Etiquetas.
4. Seleccione Administrar etiquetas para añadir etiquetas a la tabla.
5. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Aplicación de etiquetas a una tabla con CQL

- La siguiente instrucción muestra cómo agregar etiquetas a una tabla existente.

```
ALTER TABLE mykeyspace.mytable ADD TAGS {'key1':'val1', 'key2':'val2'};
```

CLI

Añada etiquetas a una tabla mediante el AWS CLI

- En el siguiente ejemplo se muestra cómo agregar nuevas etiquetas a una tabla existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/table/myTable' --tags
'key=key3,value=val3' 'key=key4,value=val4'
```

Eliminación de etiquetas de una tabla

Console

Eliminación de etiquetas de una tabla mediante la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Tablas.
3. Seleccione una tabla de la lista y seleccione la pestaña Etiquetas.
4. Seleccione Administrar etiquetas para eliminar etiquetas de la tabla.
5. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Eliminación de etiquetas de una tabla con CQL

- La siguiente instrucción muestra cómo eliminar etiquetas de una tabla existente.

```
ALTER TABLE mytable DROP TAGS {'key3':'val3', 'key4':'val4'};
```

CLI

Añada etiquetas a una tabla mediante el AWS CLI

- La siguiente instrucción elimina las etiquetas especificadas de un espacio de claves.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/table/myTable' --tags  
'key=key3,value=val3' 'key=key4,value=val4'
```

Visualización de las etiquetas de una tabla

Los siguientes ejemplos muestran cómo ver las etiquetas de una tabla en Amazon Keyspaces mediante la consola, el CQL o el AWS CLI

Console

Visualización de las etiquetas de una tabla mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Tablas.
3. Seleccione una tabla de la lista y seleccione la pestaña Etiquetas.

Cassandra Query Language (CQL)

Visualización de las etiquetas de una tabla con CQL

Para leer las etiquetas adjuntas a una tabla, utilice la siguiente instrucción CQL.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

La cláusula WHERE es obligatoria y debe usar uno de los siguientes formatos:

- `keyspace_name = 'mykeyspace' AND resource_name = 'mytable'`
- `resource_id = arn`
- La siguiente consulta devuelve las etiquetas de la tabla especificada.

```
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-1:111122223333:/keyspace/mykeyspace/table/mytable |
mykeyspace   | mytable      | table        | {'key1': 'val1', 'key2':
'val2'}
```

CLI

Para ver las etiquetas de una tabla, utilice el AWS CLI

- En este ejemplo se muestra cómo enumerar las etiquetas del recurso especificado.

```
aws keyspaces list-tags-for-resource --resource-arn
'arn:aws:arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/
table/my_table/stream/2025-05-11T21:21:33.291cassandra:us-east-1:111122223333:/
keyspace/myKeyspace/table/myTable'
```

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"
    }
  ]
}
```

Agregue etiquetas a una nueva secuencia al crear una tabla

Puede añadir etiquetas a las transmisiones al crear una nueva tabla con una transmisión mediante CQL o AWS CLI para etiquetar una transmisión.

Note

Amazon Keyspaces CDC requiere la presencia de un rol vinculado a un servicio (AWSServiceRoleForAmazonKeyspacesCDC) que publique los datos de métricas de las transmisiones de Amazon Keyspaces CDC en su cuenta "cloudwatch:namespace": "AWS/Cassandra" en su nombre. CloudWatch Este rol se crea automáticamente. Para obtener más información, consulte [the section called “Transmisiones de los CDC”](#).

Cassandra Query Language (CQL)

Añada etiquetas a una transmisión al crear una tabla nueva mediante CQL

1. Para crear una nueva tabla con una secuencia y aplicar las etiquetas de la tabla automáticamente a la secuencia, puedes usar la 'propagate_tags': 'TABLE' marca. La siguiente instrucción es un ejemplo de ello.

```
CREATE TABLE mytable (pk int, ck text, PRIMARY KEY(pk))
WITH TAGS={'key1':'val1', 'key2':'val2'}
AND cdc = TRUE
AND CUSTOM_PROPERTIES={
    'cdc_specification': {
        'view_type': 'NEW_IMAGE',
        'propagate_tags': 'TABLE'
    }
};
```

2. Para aplicar nuevas etiquetas a la transmisión, puedes usar el siguiente ejemplo.

```
CREATE TABLE mytable (pk int, ck text, PRIMARY KEY(pk))
WITH TAGS={'key1':'val1', 'key2':'val2'}
AND cdc = TRUE
AND CUSTOM_PROPERTIES={
    'cdc_specification': {
        'view_type': 'NEW_IMAGE',
        'tags': { 'key': 'string', 'value': 'string' },
    }
};
```

CLI

Añada etiquetas a una transmisión al crear una tabla nueva mediante el AWS CLI

1. Para crear una tabla con una secuencia y aplicar las etiquetas de la tabla automáticamente a la transmisión, puedes usar la `propagateTags=Table` marca. El siguiente código es un ejemplo de ello.

```
aws keyspaces create-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--schema-definition 'allColumns=[{name=pk,type=int},  
{name=ck,type=text}],clusteringKeys=[{name=ck,orderBy=ASC}],partitionKeys=[{name=pk}]' \  
\  
--tags key=tag_key,value=tag_value \  
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Para aplicar diferentes etiquetas a la transmisión, puedes usar el siguiente ejemplo.

```
aws keyspaces create-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--schema-definition 'allColumns=[{name=pk,type=int},  
{name=ck,type=text}],clusteringKeys=[{name=ck,orderBy=ASC}],partitionKeys=[{name=pk}]' \  
\  
--tags key=tag_key,value=tag_value \  
--cdc-specification 'status=ENABLED,viewType=NEW_IMAGE,tags=[{key=tag_key,  
value=tag_value}]'
```

Agregue etiquetas a una nueva transmisión para una tabla existente

Puede añadir etiquetas al crear una transmisión nueva para una tabla existente. Puede usar la `PropagateTags` marca para aplicar las etiquetas de la tabla a la transmisión o especificar nuevas etiquetas para la transmisión. Puedes usar CQL o el AWS CLI para etiquetar una nueva transmisión.

Note

Amazon Keyspaces CDC requiere la presencia de un rol vinculado a un servicio (AWSServiceRoleForAmazonKeyspacesCDC) que publique los datos de métricas de las transmisiones de Amazon Keyspaces CDC en su cuenta "cloudwatch:namespace":

"AWS/Cassandra" en su nombre. CloudWatch Este rol se crea automáticamente. Para obtener más información, consulte [the section called "Transmisiones de los CDC"](#).

Console

Añada etiquetas al crear una nueva transmisión mediante la (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, selecciona Tablas y, a continuación, elige la tabla a la que quieres añadir una transmisión.
3. Seleccione la pestaña Streams.
4. En la sección de detalles de la transmisión, selecciona Editar.
5. Seleccione Activar las transmisiones.
6. Seleccione el tipo de vista y vaya a Etiquetas para crear etiquetas para la transmisión.
7. Puede seleccionar una de las siguientes opciones:
 - Sin etiquetas: utilice esta opción si no desea crear ninguna etiqueta para la transmisión.
 - Copiar etiquetas de la tabla: utilice esta opción si desea copiar las etiquetas de la tabla a la transmisión. Tras copiar las etiquetas, puede editarlas para la transmisión. Tenga en cuenta que esta opción solo está disponible si la tabla tiene etiquetas.
 - Añadir etiquetas nuevas: puedes añadir hasta 50 etiquetas para la transmisión seleccionando Añadir nueva etiqueta.
8. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Agrega etiquetas al crear una nueva transmisión

1. Para crear una nueva transmisión para una tabla existente y aplicar las etiquetas de la tabla a la transmisión, puedes usar la 'propagate_tags': 'TABLE' marca. Las siguientes declaraciones son un ejemplo de ello.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':  
  { 'view_type': 'NEW_IMAGE', 'propagate_tags': 'TABLE' } };
```

2. Para crear un flujo nuevo para una tabla existente y especificar nuevas etiquetas, puede usar el siguiente ejemplo.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':  
  { 'view_type': 'NEW_IMAGE', 'tags': { 'key': 'string', 'value': 'string' }} };
```

CLI

Añada etiquetas al crear una nueva transmisión mediante el AWS CLI

1. Para crear una nueva transmisión con etiquetas, puedes usar la `propagateTags=TABLE` marca para aplicar automáticamente las etiquetas de la tabla a la transmisión. El siguiente código es un ejemplo de ello.

```
aws keyspaces update-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Para crear una nueva transmisión para una tabla existente y especificar nuevas etiquetas, puedes usar el siguiente ejemplo.

```
aws keyspaces update-table \  
--keyspace-name 'my_keyspace' \  
--table-name 'my_table' \  
--cdc-specification 'status=ENABLED,viewType=NEW_IMAGE,tags=[{key=tag_key,  
  value=tag_value}]'
```

Añadir nuevas etiquetas a una transmisión

Puede añadir nuevas etiquetas a una transmisión existente en Amazon Keyspaces mediante el CQL o el AWS CLI. Solo puede añadir etiquetas a la última transmisión.

Console

Añadir etiquetas a una transmisión existente (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>

2. En el panel de navegación, seleccione Tablas y, a continuación, elija la tabla con la secuencia que desee etiquetar.
3. Seleccione la pestaña Streams.
4. En la sección Etiquetas, elija Administrar etiquetas.
5. Seleccione Añadir nueva etiqueta para añadir una nueva etiqueta. Puedes crear hasta 50 etiquetas repitiendo este paso.
6. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Añada etiquetas a una transmisión mediante CQL

- La siguiente declaración muestra cómo añadir etiquetas a una transmisión existente.

```
ALTER TABLE mykeyspace.mytable ADD TAGS_FOR_CDC {'key1':'val1', 'key2':'val2'};
```

CLI

Añada etiquetas a una transmisión mediante el AWS CLI

- El siguiente ejemplo muestra cómo añadir nuevas etiquetas a una transmisión existente.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Eliminar etiquetas de una transmisión

Para eliminar etiquetas de una transmisión, puede usar CQL o el AWS CLI. Solo puedes eliminar las etiquetas de la última transmisión.

Console

Eliminación de etiquetas de una tabla mediante la consola de Amazon Keyspaces

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Tablas.
3. Elige una tabla de la lista y selecciona la pestaña Streams.
4. En la sección Etiquetas, elija Administrar etiquetas para eliminar las etiquetas de la tabla.
5. Después de la etiqueta que quieres eliminar, selecciona Eliminar.
6. Seleccione Save changes (Guardar cambios).

Cassandra Query Language (CQL)

Elimine las etiquetas de una transmisión mediante CQL

- La siguiente declaración muestra cómo eliminar etiquetas de una transmisión existente.

```
ALTER TABLE mytable DROP TAGS_FOR_CDC {'key3':'val3', 'key4':'val4'};
```

CLI

Elimine las etiquetas de una transmisión mediante el AWS CLI

- La siguiente instrucción elimina las etiquetas especificadas de un flujo.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Ver las etiquetas de una transmisión

Los siguientes ejemplos muestran cómo ver las etiquetas de una transmisión en Amazon Keyspaces mediante CQL o el. AWS CLI

Console

Vea las etiquetas de una transmisión mediante la consola

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en. <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija Tablas.
3. Elige una tabla de la lista y selecciona la pestaña Streams.
4. Puede ver las etiquetas de la transmisión en la sección Etiquetas.

Cassandra Query Language (CQL)

Vea las etiquetas de una transmisión mediante CQL

Para leer las etiquetas adjuntas a una transmisión, debe especificar el ARN de recurso de la transmisión en la WHERE cláusula. La siguiente sintaxis CQL es un ejemplo de ello.

```
SELECT * FROM system_schema_mcs.tags WHERE resource_id = stream_arn;
```

- La siguiente consulta devuelve las etiquetas del flujo especificado.

```
SELECT tags FROM system_schema_mcs.tags WHERE resource_id =  
'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/  
stream/2025-05-06T17:17:39.800';
```

La salida de esa consulta tiene el siguiente aspecto.

```
resource_id
                                     | keyspace_name | resource_name
resource_type | tags
-----+-----+-----
arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/
stream/2025-04-02T23:00:07.052      |      singleks | 2025-04-02T23:00:07.052 |
stream | {'tagkey': 'tagval'}
```

CLI

Vea las etiquetas de una transmisión mediante el AWS CLI

- En este ejemplo, se muestra cómo enumerar las etiquetas de todos los flujos en el espacio de claves especificado.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291'
```

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"
    }
  ]
}
```

Creación de informes de asignación de costos con etiquetas para Amazon Keyspaces

AWS utiliza etiquetas para organizar los costes de los recursos en el informe de asignación de costes. AWS proporciona dos tipos de etiquetas de asignación de costes:

- Una etiqueta AWS generada por nosotros. AWS define, crea y aplica esta etiqueta por usted.
- User-defined etiquetas. Puede definir, crear y aplicar estas etiquetas.

Debe activar ambos tipos de etiquetas por separado para que puedan aparecer en el Explorador de costos o en un informe de asignación de costos.

Para activar las etiquetas AWS generadas por nosotros:

1. Inicie sesión en la consola de Billing Consola de administración de AWS and Cost Management y ábrala en <https://console.aws.amazon.com/billing/home#/>.
2. En el panel de navegación, seleccione Etiquetas de asignación de costos.
3. En Etiquetas de asignación de costos generada por AWS, elija Activar.

Para activar las etiquetas definidas por el usuario:

1. Inicie sesión en la consola de Billing Consola de administración de AWS and Cost Management y ábrala en <https://console.aws.amazon.com/billing/home#/>.
2. En el panel de navegación, seleccione Etiquetas de asignación de costos.
3. En Etiquetas User-Defined de asignación de costes, seleccione Activar.

Tras crear y activar las etiquetas, AWS genera un informe de asignación de costes con el uso y los costes agrupados por las etiquetas activas. El informe de asignación de costos incluye todos AWS los costos de cada período 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](#).

Cree recursos de Amazon Keyspaces con AWS CloudFormation

Amazon Keyspaces está integrado con AWS CloudFormation un servicio que le ayuda a modelar y configurar sus AWS espacios clave y tablas para que pueda dedicar menos tiempo a crear y administrar sus recursos e infraestructura. Cree una plantilla que describa los espacios clave y las tablas que desee y CloudFormation se encarga de aprovisionar y configurar esos recursos por usted.

Cuando la utilice CloudFormation, podrá reutilizar la plantilla para configurar los recursos de Amazon Keyspaces de forma coherente y repetida. Simplemente describa sus recursos una vez y, a continuación, aprovisiona los mismos recursos una y otra vez en varias Cuentas de AWS regiones.

Amazon Keyspaces y plantillas CloudFormation

Para aprovisionar y configurar recursos para Amazon Keyspaces, debe comprender las [plantillas de CloudFormation](#). Las plantillas son archivos de texto con formato JSON o YAML. Estas plantillas describen los recursos que desea aprovisionar en sus CloudFormation pilas. Si no estás familiarizado con JSON o YAML, puedes usar CloudFormation Designer para ayudarte a empezar con CloudFormation las plantillas. Para obtener más información, consulte [¿Qué es el diseñador CloudFormation ?](#) en la Guía del usuario de AWS CloudFormation .

Amazon Keyspaces permite crear espacios de claves y tablas en CloudFormation. Para las tablas que crees con CloudFormation plantillas, puedes especificar el esquema, el read/write modo, la configuración de rendimiento aprovisionada y otras funciones compatibles. Para obtener más información, incluidos ejemplos de plantillas JSON y YAML para espacios clave y tablas, consulte la referencia del [tipo de recurso Amazon Keyspaces \(para Apache Cassandra\) en la Referencia de plantillas.AWS CloudFormation](#)

Obtenga más información sobre CloudFormation

Para obtener más información CloudFormation, consulte los siguientes recursos:

- [AWS CloudFormation](#)
- [AWS CloudFormation Guía del usuario](#)
- [AWS CloudFormation Guía del usuario de la interfaz de línea de comandos](#)

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 “Creación de un modelo de datos”](#) y [the section called “Edición de un modelo 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 “Visualización de un modelo de datos”](#).

Temas

- [Descargar NoSQL Workbench](#)
- [Introducción a NoSQL Workbench](#)
- [Visualización de modelos de datos con NoSQL Workbench](#)
- [Creación de un nuevo modelo de datos con NoSQL Workbench](#)
- [Edición de modelos de datos existentes con NoSQL Workbench](#)
- [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.

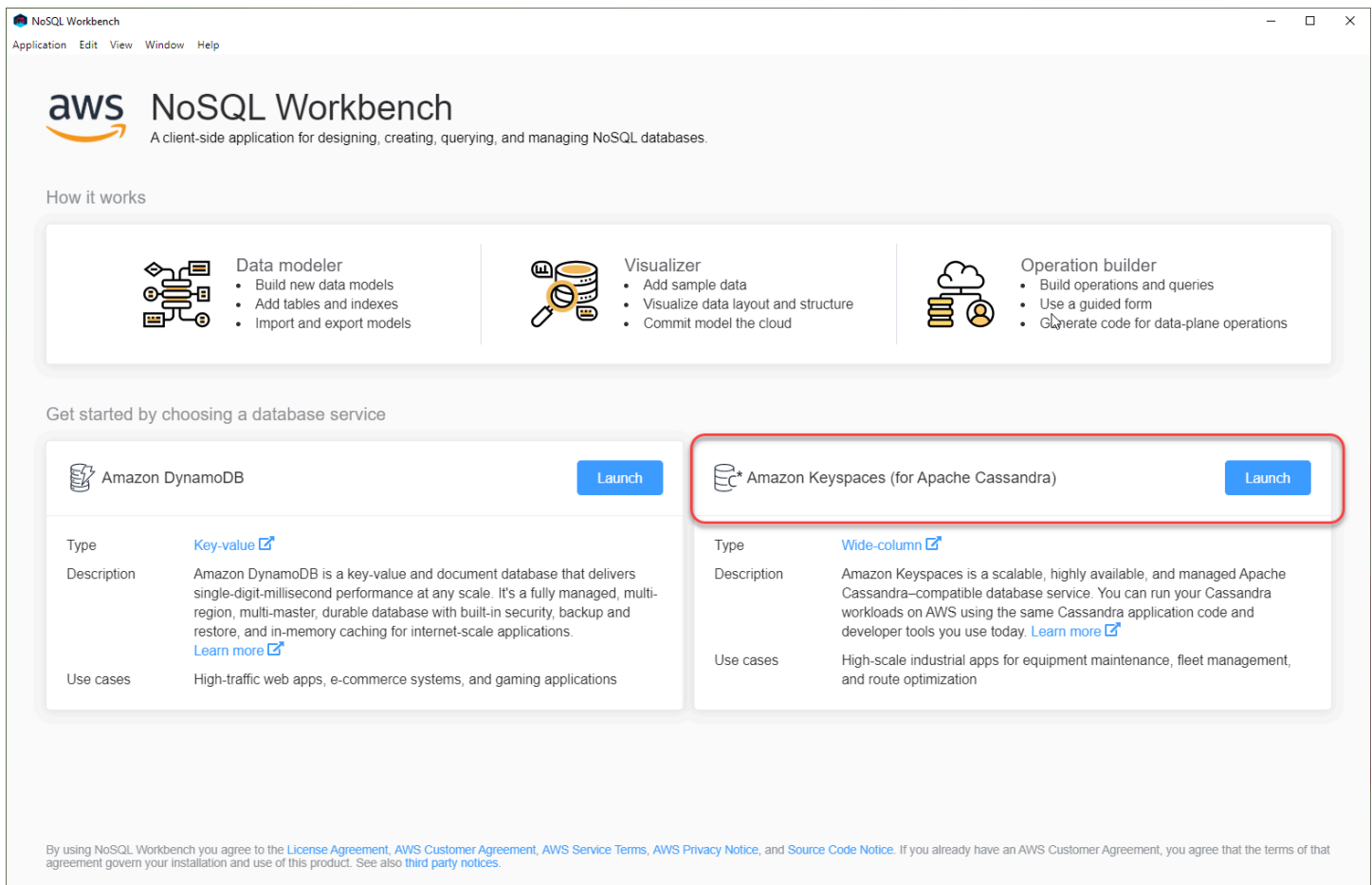
Sistema operativo	Enlace de descarga
macOS	Descargar para macOS
Linux*	Descargar para Linux
Windows	Descargar para Windows

* 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.

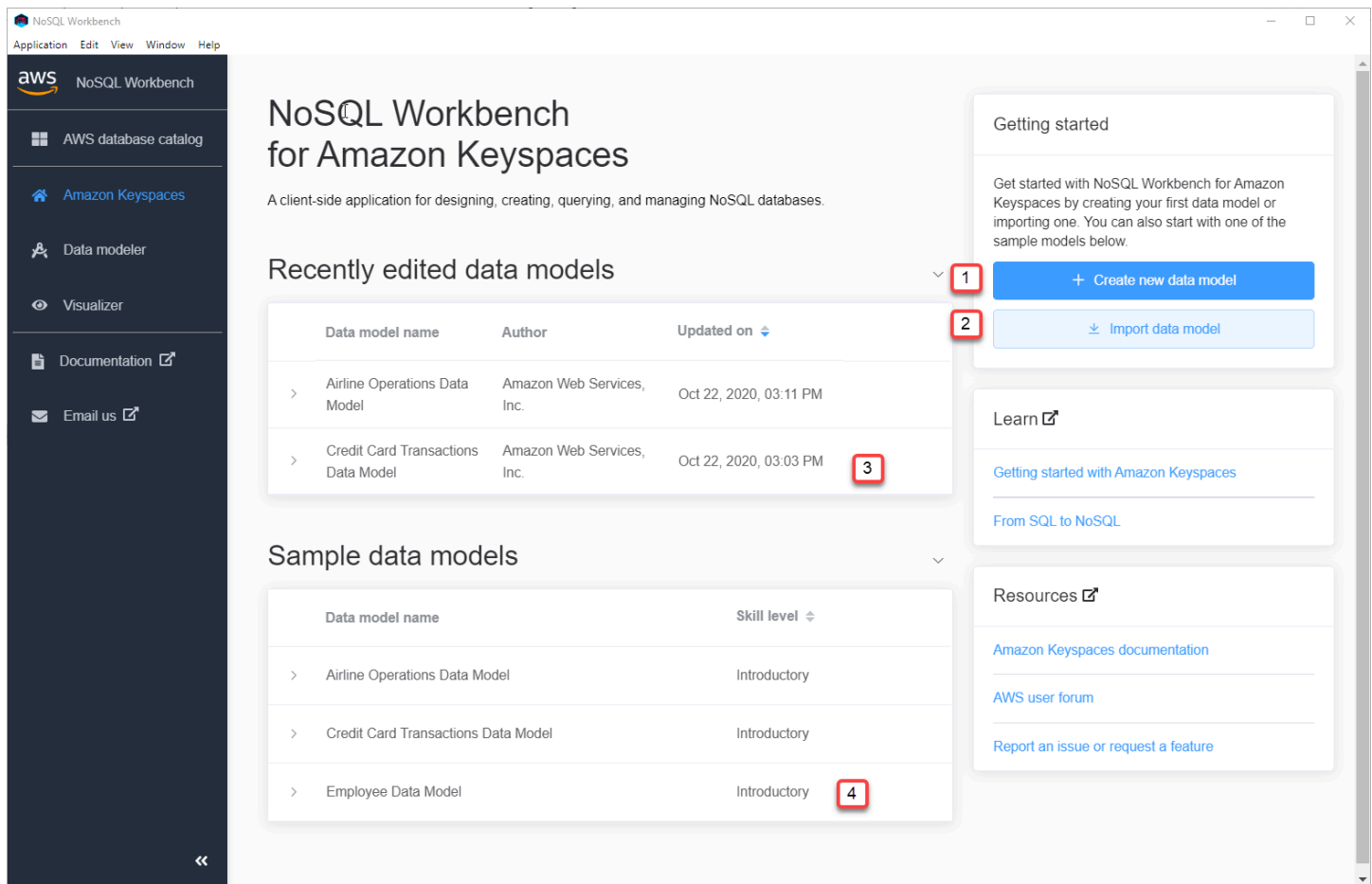
Introducción a NoSQL Workbench

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”](#).

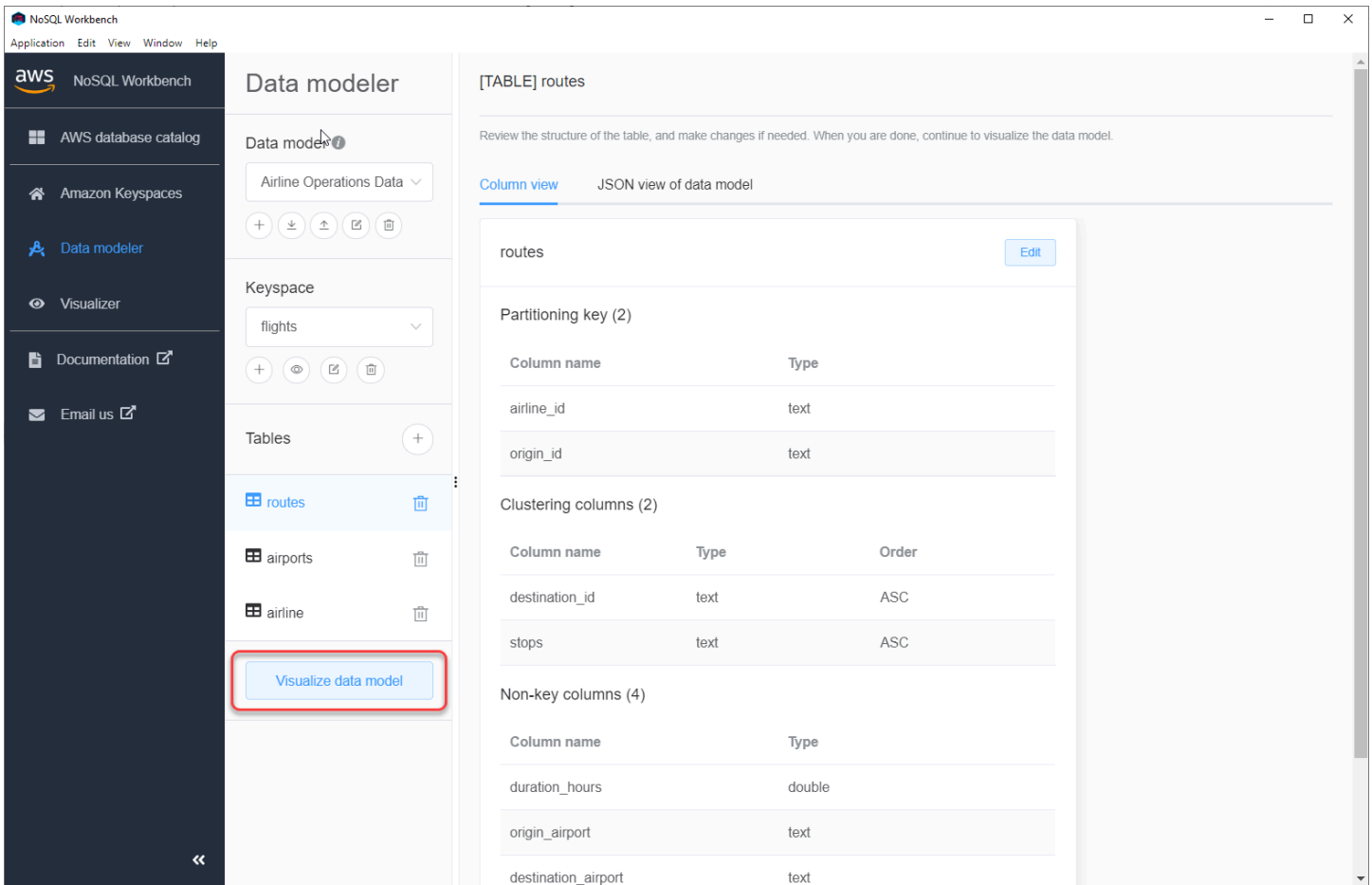
Visualización de modelos de datos con NoSQL Workbench

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.



The screenshot shows the AWS NoSQL Workbench interface. On the left, the 'Data modeler' tab is active. The main workspace shows the 'routes' table structure. The 'Visualize data model' button is highlighted with a red box.

Data modeler

Database: Airline Operations Data

Keyspace: flights

Tables:

- routes
- airports
- airline

Visualize data model

[TABLE] routes

Review the structure of the table, and make changes if needed. When you are done, continue to visualize the data model.

Column view | JSON view of data model

routes [Edit]

Partitioning key (2)

Column name	Type
airline_id	text
origin_id	text

Clustering columns (2)

Column name	Type	Order
destination_id	text	ASC
stops	text	ASC

Non-key columns (4)

Column name	Type
duration_hours	double
origin_airport	text
destination_airport	text

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 Guardar cuando haya terminado.

The screenshot shows the AWS NoSQL Workbench Visualizer interface. On the left is a sidebar with navigation options: AWS database catalog, Amazon Keyspaces, Data modeler, Visualizer (selected), Documentation, and Email us. The main area is titled 'Visualizer' and shows the 'Data model' for 'Airline Operations'. The 'Keyspace' is set to 'flights'. Under 'Tables', 'routes' is selected. The table schema is displayed with columns: 'airline_id (text)' (Primary key), 'destination_id (text)' (Clustering column), 'stops (text)' (Clustering column), 'duration_hours (double)' (Non-key column), and 'origin_airport (text)' (Non-key column). Below the schema, a table of data is shown with three rows. At the bottom of the table, there is a '+ Add new row' button highlighted with a red box. The interface also includes buttons for 'Aggregate view', 'Commit to Amazon Keyspaces', and 'Commit to Apache Cassandra'.

Primary key			Non-key columns	
Partition key	Clustering columns			
airline_id (text)	destination_id (text)	stops (text)	duration_hours (double)	origin_airport (text)
acme_airlines	MCI	1	0.33333333	Newark Liberty
trusted_airlines	MIC	2	0.33333333	Phoenix Sky Ha
freedom_airline	EWR	1	0.33333333	San Francisco I

Vista agregada

Una vez confirmado el esquema de la tabla, puede añadir visualizaciones del modelo de datos.

The screenshot shows the AWS NoSQL Workbench Visualizer interface. On the left sidebar, the 'Visualizer' tab is selected. The main panel displays a data model for 'Airline Operations Data' in the 'flights' keyspace. The table 'routes' is visualized with a primary key and non-key columns.

Primary key				Non-key columns		
Partition key		Clustering columns				
airline_id (text)	origin_id (text)	destination_id (text)	stops (text)	origin_airport (text)	destination_airport (text)	equipment
acme_airlines	EWR	MCI	1	Newark Liberty International	Kansas City International	737
trusted_airlines	PHX	MIC	2	Phoenix Sky Harbor International	Kansas City International	737
freedom_airlines	SFO	EWR	1	San Francisco International	Newark Liberty International	747

The 'Aggregate view' button is highlighted with a red box. Below it are buttons for 'Commit to Amazon Keyspaces' and 'Commit to Apache Cassandra'.

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.

Visualizer

Data model

Airline Operations Data

KeySpace

flights

Tables

- routes
- airports
- airline

Aggregate view

Commit to Amazon Keyspaces

Commit to Apache Cassandra

Aggregate view

[TABLE] routes

Primary key				Non-key columns		
Partition key		Clustering columns				
airline_id (text)	origin_id (text)	destination_id (text)	stops (text)	origin_airport (text)	destination_airport (text)	equipment (text)
acme_airlines	EWR	MCI	1	Newark Liberty International	Kansas City International	737
trusted_airlines	PHX	MIC	2	Phoenix Sky Harbor International	Kansas City International	737
freedom_airlines	SFO	EWR	1	San Francisco International	Newark Liberty International	747

[TABLE] airports

Primary key	Non-key columns			
Partition key				
airport_id (text)	name (text)	city (text)	iata_code (text)	details (map<text;text>)
EWR				

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”](#).

Creación de un nuevo modelo de datos con NoSQL Workbench

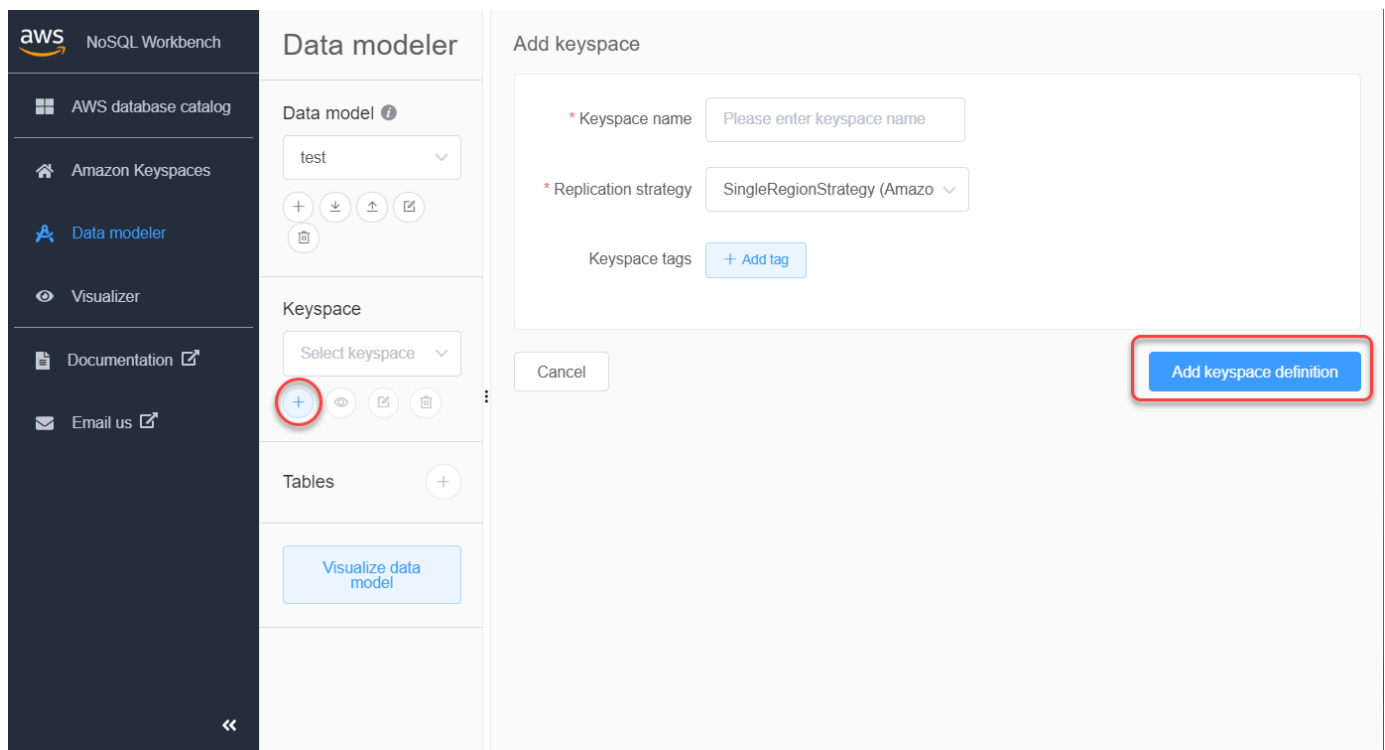
Puede utilizar el modelador de datos de NoSQL Workbench para diseñar nuevos modelos de datos basados en los patrones de acceso a datos de su aplicación. Para crear un nuevo modelo de datos para Amazon Keyspaces, puede utilizar el modelador de datos de NoSQL Workbench para crear espacios de claves, tablas 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 lo utiliza SingleRegionStrategy para replicar datos tres veces de forma automática en varias zonas de AWS disponibilidad. Si planea asignar el modelo de datos a un clúster de Apache Cassandra, puede elegir entre SimpleStrategy una o varias opciones. 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 “Trabajo 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 read/write de capacidad de la tabla. Puede elegir capacidad aprovisionada o bajo demanda. Para obtener más información sobre los modos de capacidad, consulte [the section called “Configure los modos read/write de capacidad”](#).
 - 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 “Trabajo 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.

The screenshot shows the AWS NoSQL Workbench interface. On the left is a dark sidebar with the AWS logo and 'NoSQL Workbench' text. Below it are navigation icons and labels: 'AWS database catalog', 'Amazon Keyspaces', 'Data modeler' (highlighted in blue), 'Visualizer', 'Documentation', and 'Email us'. The main area is titled 'Data modeler' and contains a 'Data model' section with a dropdown set to 'test' and a 'Keyspace' section with a dropdown set to 'key1'. Below these are icons for adding, deleting, and editing models. A 'Tables' section shows a red box around a '+' icon. At the bottom of the sidebar is a 'Visualize data model' button. The main panel is titled 'Add table definition' and contains the following fields: 'Keyspace' (dropdown set to 'key1'), 'Table name' (text input with placeholder 'Please enter table name'), 'Columns' (text input with placeholder 'Enter column name' and a 'Column type' dropdown, with a '+ Add another column' button below), 'Partition key' (text input with placeholder 'Select columns to add to partition' and a dropdown arrow), 'Clustering columns' (text input with placeholder '+ Add clustering column'), 'Capacity mode' (radio buttons for 'Provisioned' and 'On-demand', with 'On-demand' selected), and 'Table tags' (text input with placeholder '+ Add tag'). At the bottom of the main panel are 'Cancel' and 'Add table definition' buttons. A red box highlights the 'Add table definition' button.

Edición de modelos de datos existentes con NoSQL Workbench

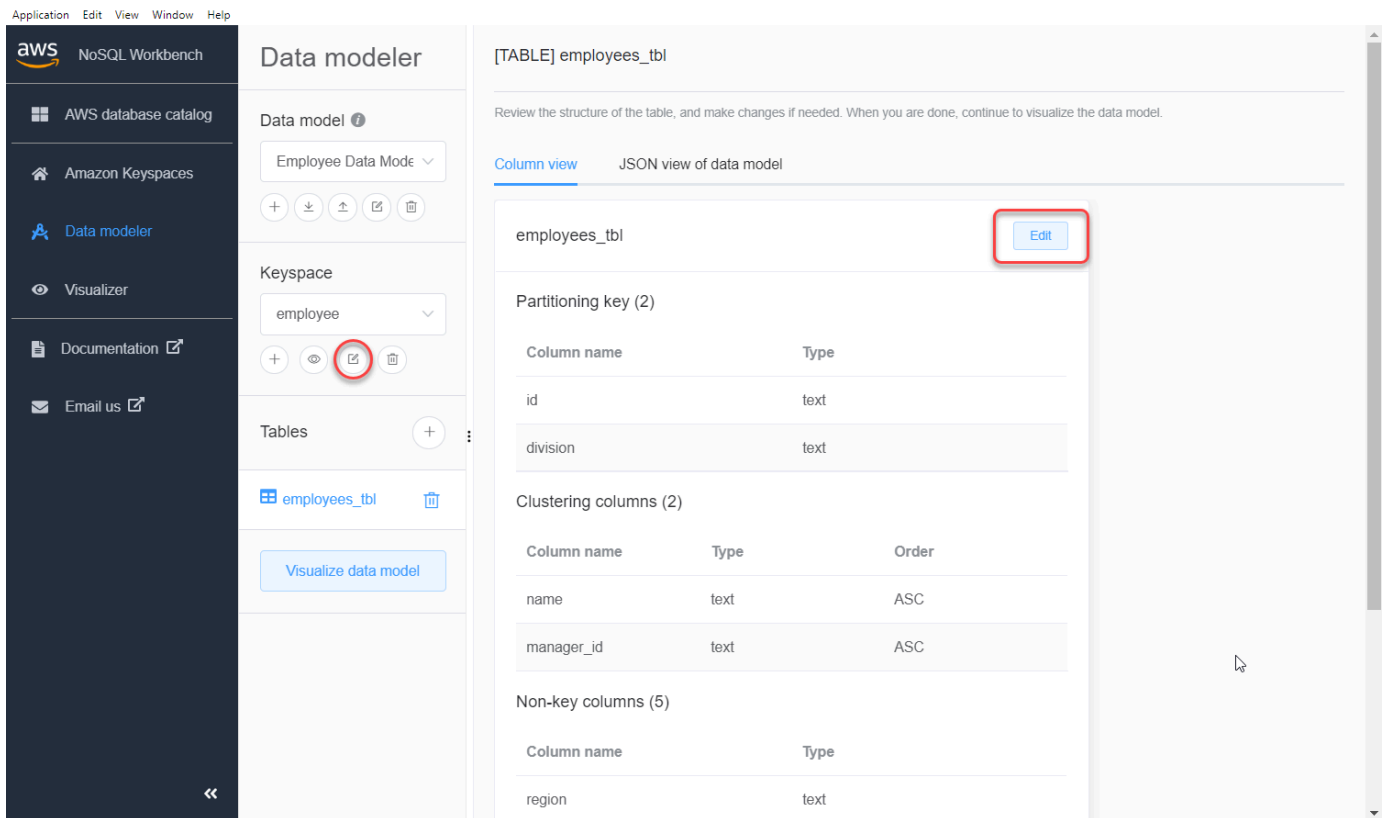
Puede utilizar el modelador de datos para importar y modificar modelos de datos existentes creados con NoSQL Workbench. El modelador de datos también incluye algunos modelos de datos de muestra que le ayudarán a iniciarse en el modelado de datos. Los modelos de datos que se pueden editar con NoSQL Workbench 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 lo utiliza `SingleRegionStrategy` para replicar datos tres veces de forma automática en varias zonas de AWS disponibilidad. Si planea asignar el modelo de datos a un clúster de Apache Cassandra, puede elegir entre `SimpleStrategy` una o varias opciones. `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 “Trabajo 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 read/write de capacidad de la tabla. Puede elegir capacidad aprovisionada o bajo demanda. Para obtener más información sobre los modos de capacidad, consulte [the section called “Configure los modos read/write de capacidad”](#).
 - 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 “Trabajo 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.

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.

Visualizer

Data model: Airline Operations Data

Keyspace: flights

Tables:

- routes
- airports
- airline

Buttons:

- Aggregate view
- Commit to Amazon Keyspaces
- Commit to Apache Cassandra

[TABLE] routes

Use this view to get a visual representation of your tables schema, and add sample data. When you are done, commit your model to Amazon Keyspaces.

Primary key				Non-key columns		
Partition key		Clustering columns				
airline_id (text)	origin_id (text)	destination_id (text)	stops (text)	origin_airport (text)	destination_airport (text)	equipment
acme_airlines	EWR	MCI	1	Newark Liberty International	Kansas City International	737
trusted_airlines	PHX	MIC	2	Phoenix Sky Harbor International	Kansas City International	737
freedom_airlines	SFO	EWR	1	San Francisco International	Newark Liberty International	747

Temas

- [Antes de empezar](#)
- [Conexión con Amazon Keyspaces mediante credenciales específicas del servicio.](#)
- [Connect to Amazon Keyspaces con credenciales 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 los siguientes certificados digitales y guarde los archivos localmente o en su directorio personal.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opcional, para compatibilidad con versiones anteriores)

Para descargar los certificados, puede usar los siguientes comandos.

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

Note

Anteriormente, Amazon Keyspaces utilizaba certificados TLS anclados a la CA de clase 2 de Starfield. AWS está migrando todo Regiones de AWS a certificados emitidos en virtud de Amazon Trust Services (Amazon Root CA 1—4). Durante esta transición, configure los clientes para que confíen en las CA raíz 1—4 de Amazon y en la raíz de Starfield para garantizar la compatibilidad en todas las regiones.

Combine todos los certificados descargados en un solo pem archivo con el nombre de nuestros *keyspaces-bundle.pem* ejemplos. Puede hacerlo ejecutando el siguiente comando : Tome nota de la ruta al archivo, la necesitará más adelante.

```
cat AmazonRootCA1.pem \
    AmazonRootCA2.pem \
    AmazonRootCA3.pem \
```



```
AmazonRootCA4.pem \  
sf-class2-root.crt \  
> keyspaces-bundle.pem
```

Una vez guardado el archivo de certificado, ya puede conectarse a Amazon Keyspaces. Una opción es conectarse mediante credenciales específicas del servicio. Service-specific las credenciales son un nombre de usuario y una contraseña que están asociados a un usuario de IAM específico y que solo se pueden utilizar 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 de acceso programático”](#).

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 con Amazon Keyspaces mediante 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 “Creación de 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

i 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 >

i 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

Connect to Amazon Keyspaces con credenciales 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 [Consola de administración de AWS](#) desde la página de inicio de sesión de IAM. Para obtener información sobre las credenciales AWS de seguridad, incluido el acceso programático y las alternativas a las credenciales a largo plazo, consulte las credenciales de [AWS seguridad en la Guía del usuario de IAM](#). Para obtener más información sobre cómo iniciar sesión en su Cuenta de AWS cuenta, consulte [Cómo iniciar sesión en](#) la AWS Guía del AWS Sign-In usuario.
 - 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 recursos](#) 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 [Administrar las claves de acceso para 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.
 - AWS certificado público: señale el AWS certificado que se descargó en el primer paso.
 - Mantener la conexión: seleccione esta casilla de verificación si desea guardar los secretos de la AWS conexión de forma local.
2. Elija Confirmar para actualizar Amazon Keyspaces con el modelo de datos.

i 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. **i**

Persist connection

☒

If you select this check box, AWS connection secrets will be persisted in

C:\Users\alexis.fid\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.

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 : >

* Saved connections

default



* 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.

Cancel

Reset

Commit

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

Password

* Contact points

[+ Add another contact point](#)

* Local data center

* Port

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 datos no relacionales para Amazon Keyspaces con	5 de octubre de 2020

Cambio	Descripción	Fecha
	mayor facilidad. Los clientes de NoSQL Workbench están disponibles para Windows, macOS y Linux. Para obtener más información, consulte NoSQL Workbench para Amazon Keyspaces .	

Ejemplos de código para Amazon Keyspaces con SDK AWS

Los siguientes ejemplos de código muestran cómo utilizar Amazon Keyspaces con un kit de desarrollo de AWS software (SDK).

Los conceptos básicos son ejemplos de código que muestran cómo realizar las operaciones esenciales dentro de un servicio.

Las acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Mientras las acciones muestran cómo llamar a las distintas funciones de servicio, es posible ver las acciones en contexto en los escenarios relacionados.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Ejemplos de código

- [Ejemplos básicos de Amazon Keyspaces que utilizan SDK AWS](#)
 - [Introducción a Amazon Keyspaces](#)
 - [Aprenda los conceptos básicos de Amazon Keyspaces con un SDK AWS](#)
 - [Acciones para Amazon Keyspaces mediante SDK AWS](#)
 - [CreateKeyspaceÚsalo con un AWS SDK](#)
 - [CreateTableÚselo con un AWS SDK](#)
 - [DeleteKeyspaceÚselo con un AWS SDK](#)
 - [DeleteTableÚselo con un AWS SDK](#)
 - [GetKeyspaceÚselo con un AWS SDK](#)
 - [GetTableÚselo con un AWS SDK](#)
 - [ListKeyspacesÚselo con un AWS SDK](#)
 - [ListTablesÚselo con un AWS SDK](#)
 - [RestoreTableÚselo con un AWS SDK](#)
 - [UpdateTableÚselo con un AWS SDK](#)

Ejemplos básicos de Amazon Keyspaces que utilizan SDK AWS

En los siguientes ejemplos de código se muestra cómo utilizar los conceptos básicos de Amazon Keyspaces (para Apache Cassandra) con los AWS SDK.

Ejemplos

- [Introducción a Amazon Keyspaces](#)
- [Aprenda los conceptos básicos de Amazon Keyspaces con un SDK AWS](#)
- [Acciones para Amazon Keyspaces mediante SDK AWS](#)
 - [CreateKeyspaceUtilízalo con un AWS SDK](#)
 - [CreateTableÚselo con un AWS SDK](#)
 - [DeleteKeyspaceÚselo con un AWS SDK](#)
 - [DeleteTableÚselo con un AWS SDK](#)
 - [GetKeyspaceÚselo con un AWS SDK](#)
 - [GetTableÚselo con un AWS SDK](#)
 - [ListKeyspacesÚselo con un AWS SDK](#)
 - [ListTablesÚselo con un AWS SDK](#)
 - [RestoreTableÚselo con un AWS SDK](#)
 - [UpdateTableÚselo con un AWS SDK](#)

Introducción a Amazon Keyspaces

En los siguientes ejemplos de código se muestra cómo empezar a utilizar Amazon Keyspaces.

.NET

SDK para .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 [ListKeyspaces](#) la Referencia AWS SDK para .NET de la API.

Java

SDK para Java 2.x

Note

Hay más información al respecto en 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 [ListKeyspaces](#) la 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.listKeyspaces(keyspacesRequest)
        response.keyspaces?.forEach { keyspace ->
            println("The name of the keyspace is ${keyspace.keyspaceName}")
        }
    }
}
```

- Para obtener más información sobre la API, consulta [ListKeyspaces](#) la 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 [ListKeyspaces](#) la 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 este servicio con un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Aprenda los conceptos básicos de Amazon Keyspaces con un SDK AWS

En el siguiente ejemplo de código, se muestra cómo:

- Crear un espacio de claves y una tabla. El esquema de tabla contiene datos de películas y tiene habilitada la recuperación a un momento determinado o PITR.
- Conectarse al espacio de claves mediante una conexión TLS segura con autenticación SigV4.
- Consultar la tabla. Agregar, recuperar y actualizar datos de películas.
- Actualizar la tabla. Añadir una columna para llevar un seguimiento de las películas vistas.
- Restaurar la tabla a su estado anterior y limpiar los recursos.

.NET

SDK para .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.WriteLine($"Created {keyspaceName}. Waiting for it to become
available. ");
    do
    {

```

```
        getKeyspaceArn = await
keyspacesWrapper.GetKeyspace(keyspaceName);
        Console.WriteLine(". ");
    } 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.WriteLine("Waiting for the new table to be active. ");
}
```

```
do
{
    try
    {
        resp = await keyspacesWrapper.GetTable(keyspaceName,
tableName);

        Console.WriteLine(".");
    }
    catch (ResourceNotFoundException)
    {
        Console.WriteLine(".");
    }
} 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\t{title}\t\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.WriteLine(".");
        }
    }

    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 sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para .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)
                .addColumnns(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 sobre 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteKeyspace(deleteKeyspaceRequest)
    }
}

suspend fun checkTableDelete(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var status: String
    var response: GetTableResponse
    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    try {
        KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient
->
            // Keep looping until the table cannot be found and a
ResourceNotFoundException is thrown.
            while (true) {
                response = keyClient.getTable(tableRequest)
                status = response.status.toString()
                println(". The table status is $status")
                delay(500)
            }
        }
    } catch (e: ResourceNotFoundException) {
```



```
        println(e.message)
    }
    println("The table is deleted")
}

suspend fun deleteTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    val tableRequest =
        DeleteTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.deleteTable(tableRequest)
    }
}

suspend fun checkRestoredTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println("The table status is $status")

            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
        }
        delay(500)
    }
}
```

```

    }

    val cols = response!!.schemaDefinition?.allColumns
    if (cols != null) {
        for (def in cols) {
            println("The column name is ${def.name}")
            println("The column type is ${def.type}")
        }
    }
}

suspend fun restoreTable(
    keyspaceName: String?,
    utc: ZonedDateTime,
) {
    // Create an aws.smithy.kotlin.runtime.time.Instant value.
    val timeStamp =
        aws.smithy.kotlin.runtime.time
            .Instant(utc.toInstant())
    val restoreTableRequest =
        RestoreTableRequest {
            restoreTimestamp = timeStamp
            sourceTableName = "MovieKotlin"
            targetKeyspaceName = keyspaceName
            targetTableName = "MovieRestore"
            sourceKeyspaceName = keyspaceName
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.restoreTable(restoreTableRequest)
        println("The ARN of the restored table is ${response.restoredTableArn}")
    }
}

fun getWatchedData(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet = session.execute("SELECT * FROM \"${keyspaceName}\".
    \"MovieKotlin\" WHERE watched = true ALLOW FILTERING;")
    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
    }
}

```

```
        println("The plot is ${item.getString("plot")}")
    }
}

fun updateRecord(
    session: CqlSession,
    keySpace: String,
    titleUpdate: String?,
    yearUpdate: Int,
) {
    val sqlStatement =
        "UPDATE \"$keySpace\".\"MovieKotlin\" SET watched=true WHERE title = :k0
AND year = :k1;"
    val builder = BatchStatement.builder(DefaultBatchType.UNLOGGED)
    builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
    val preparedStatement = session.prepare(sqlStatement)
    builder.addStatement(
        preparedStatement
            .boundStatementBuilder()
            .setString("k0", titleUpdate)
            .setInt("k1", yearUpdate)
            .build(),
    )
    val batchStatement = builder.build()
    session.execute(batchStatement)
}

suspend fun updateTable(
    keySpace: String?,
    tableNameVal: String?,
) {
    val def =
        ColumnDefinition {
            name = "watched"
            type = "boolean"
        }

    val tableRequest =
        UpdateTableRequest {
            keyspaceName = keySpace
            tableName = tableNameVal
            addColumns = listOf(def)
        }
}
```

```

        KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
            keyClient.updateTable(tableRequest)
        }
    }

fun getSpecificMovie(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet =
        session.execute("SELECT * FROM \"\$keyspaceName\".\"MovieKotlin\" WHERE
title = 'The Family' ALLOW FILTERING ;")

    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
    }
}

// Get records from the Movie table.
fun getMovieData(
    session: CqlSession,
    keyspaceName: String,
) {
    val resultSet = session.execute("SELECT * FROM \"\$keyspaceName\".
\"MovieKotlin\";")
    resultSet.forEach { item: Row ->
        println("The Movie title is ${item.getString("title")}")
        println("The Movie year is ${item.getInt("year")}")
        println("The plot is ${item.getString("plot")}")
    }
}

// Load data into the table.
fun loadData(
    session: CqlSession,
    fileName: String,
    keySpace: String,
) {
    val sqlStatement =
        "INSERT INTO \"\$keySpace\".\"MovieKotlin\" (title, year, plot) values
(:k0, :k1, :k2)"
    val parser = JsonFactory().createParser(File(fileName))

```

```

val rootNode = ObjectMapper().readTree<JsonNode>(parser)
val iter: Iterator<JsonNode> = rootNode.iterator()
var currentNode: ObjectNode

var t = 0
while (iter.hasNext()) {
    if (t == 50) {
        break
    }

    currentNode = iter.next() as ObjectNode
    val year = currentNode.path("year").asInt()
    val title = currentNode.path("title").asText()
    val info = currentNode.path("info").toString()

    // Insert the data into the Amazon Keyspaces table.
    val builder = BatchStatement.builder(DefaultBatchType.UNLOGGED)
    builder.setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM)
    val preparedStatement: PreparedStatement = session.prepare(sqlStatement)
    builder.addStatement(
        preparedStatement
            .boundStatementBuilder()
            .setString("k0", title)
            .setInt("k1", year)
            .setString("k2", info)
            .build(),
    )

    val batchStatement = builder.build()
    session.execute(batchStatement)
    t++
}

suspend fun listTables(keyspaceNameVal: String?) {
    val tablesRequest =
        ListTablesRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
    }
}

```

```

        .collect { obj ->
            println(" ARN: ${obj.resourceArn} Table name: ${obj.tableName}")
        }
    }
}

suspend fun checkTable(
    keyspaceNameVal: String?,
    tableNameVal: String?,
) {
    var tableStatus = false
    var status: String
    var response: GetTableResponse? = null

    val tableRequest =
        GetTableRequest {
            keyspaceName = keyspaceNameVal
            tableName = tableNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println(". The table status is $status")
            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
        val cols: List<ColumnDefinition>? =
            response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}

suspend fun createTable(
    keySpaceVal: String?,
    tableNameVal: String?,
) {

```

```
// Set the columns.
val defTitle =
    ColumnDefinition {
        name = "title"
        type = "text"
    }

val defYear =
    ColumnDefinition {
        name = "year"
        type = "int"
    }

val defReleaseDate =
    ColumnDefinition {
        name = "release_date"
        type = "timestamp"
    }

val defPlot =
    ColumnDefinition {
        name = "plot"
        type = "text"
    }

val collist = ArrayList<ColumnDefinition>()
collist.add(defTitle)
collist.add(defYear)
collist.add(defReleaseDate)
collist.add(defPlot)

// Set the keys.
val yearKey =
    PartitionKey {
        name = "year"
    }

val titleKey =
    PartitionKey {
        name = "title"
    }

val keyList = ArrayList<PartitionKey>()
keyList.add(yearKey)
```

```
keyList.add(titleKey)

val schemaDefinition0b =
    SchemaDefinition {
        partitionKeys = keyList
        allColumns = colList
    }

val timeRecovery =
    PointInTimeRecovery {
        status = PointInTimeRecoveryStatus.Enabled
    }

val tableRequest =
    CreateTableRequest {
        keyspaceName = keySpaceVal
        tableName = tableNameVal
        schemaDefinition = schemaDefinition0b
        pointInTimeRecovery = timeRecovery
    }

KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
    val response = keyClient.createTable(tableRequest)
    println("The table ARN is ${response.resourceArn}")
}

suspend fun listKeyspacesPaginator() {
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listKeyspacesPaginated(ListKeyspacesRequest {})
            .transform { it.keyspaces?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println("Name: ${obj.keyspaceName}")
            }
    }
}

suspend fun checkKeyspaceExistence(keyspaceNameVal: String?) {
    val keyspaceRequest =
        GetKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }
    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
```



```
        val response: GetKeyspaceResponse =
            keyClient.getKeyspace(keyspaceRequest)
        val name = response.keyspaceName
        println("The $name KeySpace is ready")
    }
}

suspend fun createKeySpace(keyspaceNameVal: String) {
    val keyspaceRequest =
        CreateKeyspaceRequest {
            keyspaceName = keyspaceNameVal
        }

    KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        val response = keyClient.createKeyspace(keyspaceRequest)
        println("The ARN of the KeySpace is ${response.resourceArn}")
    }
}
```

- Para obtener detalles sobre 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 keyspace 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 de 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 información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK 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, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Acciones para Amazon Keyspaces mediante SDK AWS

Los siguientes ejemplos de código muestran cómo realizar acciones individuales de Amazon Keyspaces con AWS los SDK. Cada ejemplo incluye un enlace a GitHub, donde puede encontrar instrucciones para configurar y ejecutar el código.

Los siguientes ejemplos 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

- [CreateKeyspaceÚsalo con un AWS SDK](#)
- [CreateTableÚselo con un AWS SDK](#)
- [DeleteKeyspaceÚselo con un AWS SDK](#)
- [DeleteTableÚselo con un AWS SDK](#)
- [GetKeyspaceÚselo con un AWS SDK](#)
- [GetTableÚselo con un AWS SDK](#)
- [ListKeyspacesÚselo con un AWS SDK](#)
- [ListTablesÚselo con un AWS SDK](#)
- [RestoreTableÚselo con un AWS SDK](#)
- [UpdateTableÚselo con un AWS SDK](#)

CreateKeyspaceÚsalo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar CreateKeyspace.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [CreateKeyspace](#) la Referencia AWS SDK para .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 [CreateKeyspace](#) la 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.fromEnvironment { 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 [CreateKeyspace](#) la 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 [CreateKeyspace](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```

TRY.
    oo_result = lo_kys->createkeyspace(
        iv_keyspacename = iv_keyspace_name ).
    MESSAGE 'Keyspace created successfully.' TYPE 'I'.
CATCH /aws1/cx_kysconflictexception.
    MESSAGE 'Keyspace already exists.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.

```

```
ENDTRY.
```

- Para obtener más información sobre la API, consulte [CreateKeyspace](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

CreateTableÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar CreateTable.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [CreateTable](#) la Referencia AWS SDK para .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 [CreateTable](#) la 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 schemaDefinition0b =
    SchemaDefinition {
        partitionKeys = keyList
        allColumns = collist
    }

val timeRecovery =
    PointInTimeRecovery {
        status = PointInTimeRecoveryStatus.Enabled
    }
```

```

val tableRequest =
    CreateTableRequest {
        keyspaceName = keySpaceVal
        tableName = tableNameVal
        schemaDefinition = schemaDefinition0b
        pointInTimeRecovery = timeRecovery
    }

KeyspacesClient.fromEnvironment { 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 [CreateTable](#) la 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 [CreateTable](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```

TRY.
    " Define schema with columns
    DATA(lt_columns) = VALUE /aws1/
cl_kyscolumndefinition=>tt_columndefinitionlist(
    ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'title' iv_type =
'text' ) )
    ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'year' iv_type =
'int' ) )
    ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'release_date' iv_type =
'timestamp' ) )
    ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'plot' iv_type =
'text' ) )
    ).

    " Define partition keys
    DATA(lt_partition_keys) = VALUE /aws1/
cl_kyspartitionkey=>tt_partitionkeylist(
    ( NEW /aws1/cl_kyspartitionkey( iv_name = 'year' ) )
    ( NEW /aws1/cl_kyspartitionkey( iv_name = 'title' ) )
    ).

    " Create schema definition
    DATA(lo_schema) = NEW /aws1/cl_kysschemadefinition(
        it_allcolumns = lt_columns
        it_partitionkeys = lt_partition_keys ).

    " Enable point-in-time recovery
    DATA(lo_pitr) = NEW /aws1/cl_kyspointintimerec(

```

```

        iv_status = 'ENABLED' );

        oo_result = lo_kys->createtable(
            iv_keyspacename = iv_keyspace_name
            iv_tablename = iv_table_name
            io_schemadefinition = lo_schema
            io_pointintimerecovery = lo_pitr );
        MESSAGE 'Table created successfully.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.

```

- Para obtener más información sobre la API, consulte [CreateTable](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

DeleteKeyspaceÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar DeleteKeyspace.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [DeleteKeyspace](#) la Referencia AWS SDK para .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 [DeleteKeyspace](#) la 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->  
        keyClient.deleteKeyspace(deleteKeyspaceRequest)  
    }  
}
```

- Para obtener más información sobre la API, consulta [DeleteKeyspace](#) la 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 [DeleteKeyspace](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```
TRY.  
    lo_kys->deletekeyspace(  
        iv_keyspacename = iv_keyspace_name ).  
    MESSAGE 'Keyspace deleted successfully.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
        MESSAGE lv_error TYPE 'E'.  
    ENDTRY.
```

- Para obtener más información sobre la API, consulte [DeleteKeyspace](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

DeleteTableÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar DeleteTable.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [DeleteTable](#) la Referencia AWS SDK para .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 [DeleteTable](#) la 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->

```

```

        keyClient.deleteTable(tableRequest)
    }
}

```

- Para obtener más información sobre la API, consulta [DeleteTable](#) la 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 [DeleteTable](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```

TRY.
    lo_kys->deletetable(
        iv_keyspacename = iv_keyspace_name
        iv_tablename = iv_table_name ).
    MESSAGE 'Table deleted successfully.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obtener más información sobre la API, consulte [DeleteTable](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

GetKeyspaceÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar GetKeyspace.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [GetKeyspace](#) la Referencia AWS SDK para .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 [GetKeyspace](#) la 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.fromEnvironment { 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 [GetKeyspace](#) la 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 [GetKeyspace](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```
TRY.
    oo_result = lo_kys->getkeyspace(
        iv_keyspacename = iv_keyspace_name ).
    MESSAGE 'Keyspace retrieved successfully.' TYPE 'I'.
CATCH /aws1/cx_kysresourcenotfoundex.
    MESSAGE 'Keyspace does not exist.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obtener más información sobre la API, consulte [GetKeyspace](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

GetTableÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar GetTable.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [GetTable](#) la Referencia AWS SDK para .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 [GetTable](#) la 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        while (!tableStatus) {
            response = keyClient.getTable(tableRequest)
            status = response!!.status.toString()
            println(". The table status is $status")
            if (status.compareTo("ACTIVE") == 0) {
                tableStatus = true
            }
            delay(500)
        }
    }
}
```

```

        val cols: List<ColumnDefinition>? =
response!!.schemaDefinition?.allColumns
        if (cols != null) {
            for (def in cols) {
                println("The column name is ${def.name}")
                println("The column type is ${def.type}")
            }
        }
    }
}

```

- Para obtener más información sobre la API, consulta [GetTable](#) la 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 [GetTable](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```
TRY.
    oo_result = lo_kys->gettable(
        iv_keyspacename = iv_keyspace_name
        iv_tablename = iv_table_name ).
    MESSAGE 'Table information retrieved successfully.' TYPE 'I'.
CATCH /aws1/cx_kysresourcenotfoundex.
    MESSAGE 'Table does not exist.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obtener más información sobre la API, consulte [GetTable](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

ListKeyspacesÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar ListKeyspaces.

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:

- [Conceptos básicos](#)

.NET

SDK para .NET

Note

Hay más información al respecto en 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 [ListKeyspaces](#) la Referencia AWS SDK para .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 [ListKeyspaces](#) la 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.fromEnvironment { 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 [ListKeyspaces](#) la 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 [ListKeyspaces](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```
TRY.  
    oo_result = lo_kys->listkeyspaces(  
        iv_maxresults = iv_max_results ).  
    MESSAGE 'Keyspaces listed successfully.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obtener más información sobre la API, consulte [ListKeyspaces](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

ListTablesÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar ListTables.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [ListTables](#) la Referencia AWS SDK para .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 [ListTables](#) la 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient
            .listTablesPaginated(tablesRequest)
            .transform { it.tables?.forEach { obj -> emit(obj) } }
            .collect { obj ->
                println(" ARN: ${obj.resourceArn} Table name: ${obj.tableName}")
            }
    }
}
```

- Para obtener más información sobre la API, consulta [ListTables](#) la 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 [ListTables](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```
TRY.  
    oo_result = lo_kys->listtables(  
        iv_keyspacename = iv_keyspace_name ).  
    MESSAGE 'Tables listed successfully.' TYPE 'I'.  
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obtener más información sobre la API, consulte [ListTables](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

RestoreTableÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar RestoreTable.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [RestoreTable](#) la Referencia AWS SDK para .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 [RestoreTable](#) la 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.fromEnvironment { 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 [RestoreTable](#) la 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 [RestoreTable](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

TRY.

```

oo_result = lo_kys->restoretale(
    iv_sourcekeyspacename = iv_source_keyspace_name
    iv_sourcetablename = iv_source_table_name
    iv_targetkeyspacename = iv_target_keyspace_name
    iv_targettablename = iv_target_table_name
    iv_restoretimestamp = iv_restore_timestamp ).
MESSAGE 'Table restore initiated successfully.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).

```

```
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obtener más información sobre la API, consulte [RestoreTable](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio con un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

UpdateTableÚselo con un AWS SDK

Los siguientes ejemplos de código muestran cómo utilizar UpdateTable.

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:

- [Conceptos básicos](#)

.NET

SDK para .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 [UpdateTable](#) la Referencia AWS SDK para .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 [UpdateTable](#) la 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.fromEnvironment { region = "us-east-1" }.use { keyClient ->
        keyClient.updateTable(tableRequest)
    }
}
```

```
}
}
```

- Para obtener más información sobre la API, consulta [UpdateTable](#) la 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 [UpdateTable](#) la AWS Referencia de API de SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

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](#).

```

TRY.
    " Add a new column to track watched movies
    DATA(lt_add_columns) = VALUE /aws1/
cl_kyscolumndefinition=>tt_columndefinitionlist(
    ( NEW /aws1/cl_kyscolumndefinition( iv_name = 'watched' iv_type =
'boolean' ) )
    ).

    oo_result = lo_kys->updatetable(

```

```
        iv_keyspacename = iv_keyspace_name
        iv_tablename = iv_table_name
        it_addcolumns = lt_add_columns ).
    MESSAGE 'Table updated successfully.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obtener más información sobre la API, consulte [UpdateTable](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Uso de este servicio 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

[Puedes encontrar las bibliotecas de código abierto y las herramientas para desarrolladores de Amazon Keyspaces GitHub en los repositorios AWSy AWS samples.](#)

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 las prácticas recomendadas, una expansión de AWS autenticación opcional para cqlsh y herramientas auxiliares para realizar tareas habituales. 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/>.

AWS Plugins de autenticación Signature versión 4 (SigV4)

Los complementos le permiten administrar el acceso a Amazon Keyspaces mediante usuarios y roles 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 BLOB 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>

CloudFormation plantilla para crear el CloudWatch panel de control de Amazon para las métricas de Amazon Keyspaces (para Apache Cassandra)

Este repositorio proporciona CloudFormation plantillas para configurar rápidamente CloudWatch las métricas de Amazon Keyspaces. El uso de esta plantilla le permitirá empezar más fácilmente al proporcionar CloudWatch paneles prediseñados y desplegados con las métricas más utilizadas.

<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 usar Amazon Keyspaces con. AWS Glue

<https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue>

Amazon Keyspaces (para Apache Cassandra) Lenguaje de consulta Cassandra (CQL) a conversor CloudFormation

Este paquete implementa una herramienta de línea de comandos para convertir los scripts de Apache Cassandra Query Language (CQL) en plantillas CloudFormation (CloudFormation), lo que permite gestionar fácilmente los esquemas de Amazon Keyspaces en pilas. 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 de controladores, ejemplos y prácticas recomendadas para el uso del 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 read/write agrandar objetos para lograr un rendimiento más rápido y reducir los costos de procesamiento y 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>.

Prácticas recomendadas de diseño y arquitectura con Amazon Keyspaces (para Apache Cassandra)

Utilice esta sección para encontrar con rapidez recomendaciones para maximizar el desempeño y minimizar los costos de rendimiento al trabajar con Amazon Keyspaces.

Contenido

- [Diferencias clave y principios de diseño del diseño NoSQL](#)
 - [Diferencias entre el diseño de datos relacionales y NoSQL](#)
 - [Dos conceptos clave del diseño NoSQL](#)
 - [Aproximación al diseño NoSQL](#)
- [Optimización de las conexiones de los controladores del cliente para el entorno sin servidor](#)
 - [Funcionamiento de las conexiones en Amazon Keyspaces](#)
 - [Configuración de conexiones en Amazon Keyspaces](#)
 - [Configuración de la política de reintentos para las 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](#)
- [Prácticas recomendadas de modelado de datos: recomendaciones para diseñar modelos de datos](#)
 - [Uso eficaz de las claves de partición en Amazon Keyspaces](#)
 - [Uso de la partición de escritura para distribuir uniformemente las cargas de trabajo entre las particiones](#)
 - [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](#)
 - [Evaluación de 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 las tablas con un nivel de utilización objetivo bajo \(\$\leq 50\%\$ \)](#)
 - [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 optimizar los costos en Amazon Keyspaces](#)
 - [Cómo identificar los recursos sin utilizar](#)
 - [Identificación de recursos de tabla sin utilizar](#)
 - [Limpieza de los recursos de tabla no utilizados](#)
 - [Saneamiento de copias de seguridad de recuperación en un momento dado \(PITR\) no utilizadas](#)
- [Evaluación de los patrones de uso de las tablas para optimizar el rendimiento y los costos](#)
 - [Realización de menos operaciones de lectura altamente coherente](#)
 - [Habilitación del período de vida \(TTL\)](#)
- [Evaluación de 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](#)

Diferencias clave y principios de diseño del diseño NoSQL

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 se 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.

Diferencias del 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 corresponda 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. Saber de antemano cuáles serán los picos de carga de consultas podría ayudar a determinar cómo particionar los datos para aprovechar al máximo I/O la capacidad.

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 enrutamiento, 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 un gran volumen de consultas no se centre en una parte de la base de datos, donde pueden superar I/O su capacidad. 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.

Optimización de las conexiones de los controladores del cliente para el entorno sin servidor

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

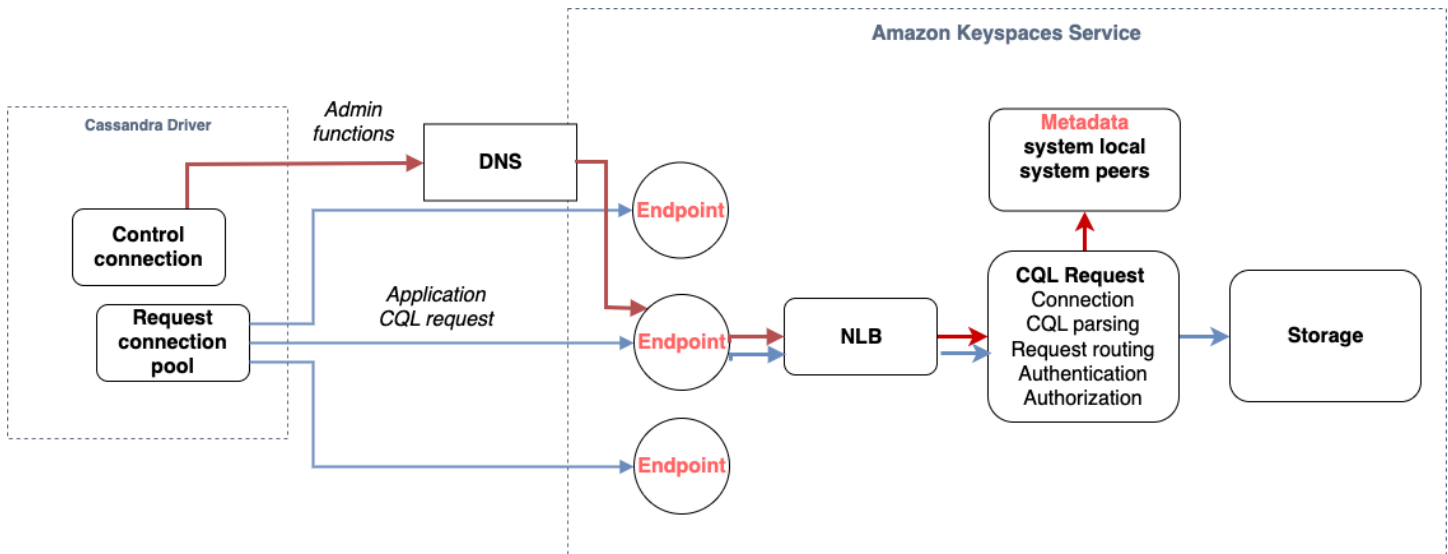
- [Funcionamiento de las conexiones en Amazon Keyspaces](#)
- [Configuración de conexiones en Amazon Keyspaces](#)
- [Configuración de la política de reintentos para las conexiones en Amazon Keyspaces](#)

- [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 fijar como objetivo solo 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 la política de reintentos para las conexiones en Amazon Keyspaces

Al configurar la política de reintentos para una conexión a Amazon Keyspaces, le recomendamos que implemente la política de reintentos de Amazon Keyspaces `AmazonKeyspacesExponentialRetryPolicy`. Esta política de reintentos es más adecuada para volver a intentarlo a través de conexiones a Amazon Keyspaces distintas a la `DefaultRetryPolicy` del controlador.

Con la `AmazonKeyspacesExponentialRetryPolicy`, puede configurar el número de reintentos de la conexión que mejor se ajuste a sus necesidades. De forma predeterminada, el número de reintentos para la `AmazonKeyspacesExponentialRetryPolicy` está establecido en 3.

Una ventaja adicional es que la política de reintentos de Amazon Keyspaces devuelve la excepción original devuelta por el servicio, lo que indica por qué falló el intento de solicitud. La política de reintentos predeterminada solo devuelve la `NoHostAvailableException` genérica, lo que podría ocultar información sobre el error de la solicitud.

Para configurar la política de reintentos de solicitudes mediante la `AmazonKeyspacesExponentialRetryPolicy`, le recomendamos que configure un número reducido de reintentos y que administre cualquier excepción devuelta en el código de la aplicación.

Para ver ejemplos de código en el que se implementan políticas de reintentos, consulte [Amazon Keyspaces retry policies](#) en Github.

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 3 puntos de conexión. 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 finales disponibles en los distintos Regiones de AWS puntos finales.

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 “Configuración del 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.

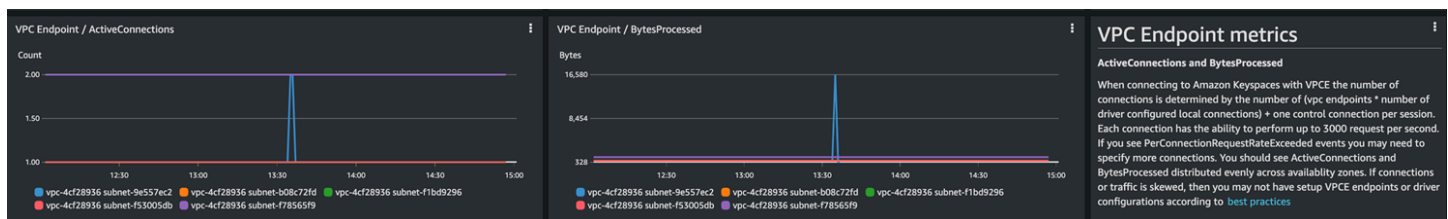
```
ResultSet result = session.execute(new SimpleStatement("SELECT * FROM system.peers"));

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 después de 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 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 [Amazon Keyspaces code examples](#).

Prácticas recomendadas de modelado de datos: recomendaciones para diseñar modelos de datos

Un modelado de datos efectivo es esencial para optimizar el rendimiento y minimizar los costos al trabajar con Amazon Keyspaces (para Apache Cassandra). En este tema se tratan las principales consideraciones y recomendaciones para diseñar modelos de datos que se adapten a los patrones de acceso a los datos de su aplicación.

- **Diseño de claves de partición:** la clave de partición desempeña un papel fundamental a la hora de determinar cómo se distribuyen los datos entre las particiones de Amazon Keyspaces. La elección de una clave de partición adecuada puede afectar considerablemente al desempeño de las consultas y a los costos de rendimiento. En esta sección se analizan las estrategias para diseñar claves de partición que promuevan una distribución uniforme de la actividad de lectura y escritura entre las particiones.
- **Consideraciones clave:**
 - **Distribución uniforme de la actividad:** procure que la actividad de lectura y escritura sea uniforme en todas las particiones para minimizar los costos de rendimiento y aprovechar la capacidad de ampliación de forma eficaz.

- Patrones de acceso: alinee el diseño de la clave de partición con los patrones de acceso a los datos principales de la aplicación.
- Tamaño de la partición: evite crear particiones que crezcan demasiado, ya que esto puede afectar al rendimiento y aumentar los costos.

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 la aplicación de manera uniforme en todas las particiones ayuda a minimizar los costos de rendimiento, y esto se aplica tanto a los modos de capacidad bajo demanda como a los de capacidad aprovisionada. read/write 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.

Cuando una partición presenta un rendimiento de lectura o escritura elevado y sostenido, en función de los patrones de tráfico, Amazon Keyspaces puede dividir automáticamente la partición en dos nuevas particiones. Cada nueva partición contiene un subconjunto de las filas de la partición original, lo que distribuye el rendimiento de manera uniforme entre ambas particiones.

Amazon Keyspaces ofrece flexibilidad adicional en su aprovisionamiento de rendimiento por partición proporcionando capacidad de ampliación. Para obtener más información, consulte [the section called “Uso de la capacidad de ampliación”](#).

Temas

- [Uso de la partición de escritura para distribuir uniformemente las cargas de trabajo entre las particiones](#)

Uso de la partición de escritura para distribuir uniformemente las cargas de trabajo entre las particiones

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 baste con un cálculo simple, como el producto de los valores de los puntos de UTF-8 código 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

- [Evaluación de 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 optimizar los costos en Amazon Keyspaces](#)
- [Evaluación de los patrones de uso de las tablas para optimizar el rendimiento y los costos](#)
- [Evaluación de la capacidad aprovisionada para lograr un aprovisionamiento del tamaño adecuado](#)

Evaluación de los costos en el nivel de tabla

La herramienta Cost Explorer que se encuentra en el Consola de administración de AWS le permite ver los costos desglosados por tipo, por ejemplo, los cargos de lectura, escritura, almacenamiento y respaldo. También puede ver estos costos resumidos por periodo, por ejemplo, por mes o por día.

Uno de los problemas comunes del Explorador de costos es que no permite revisar fácilmente los costos de una sola tabla en particular, ya que el Explorador de costos no le permite filtrar o agrupar por costos de una tabla específica. Puede ver la métrica Tamaño de tabla facturable (bytes) de cada tabla en la consola de Amazon Keyspaces, en la pestaña Supervisar la tabla. Si necesita más información relacionada con los costos por tabla, en esta sección se muestra 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](#)

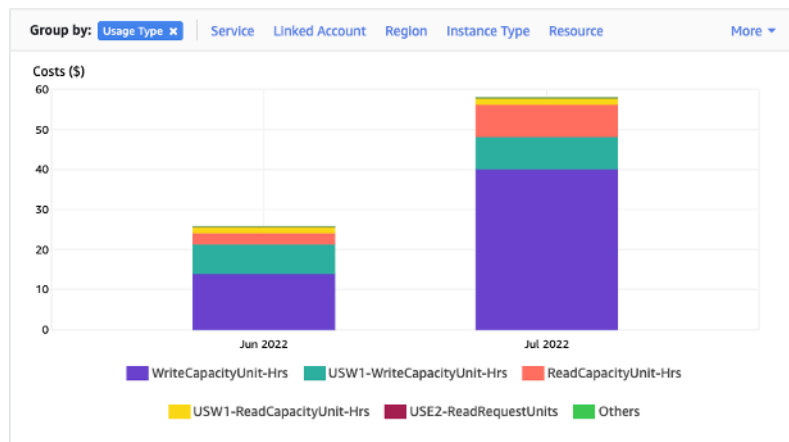
Visualización de costos de una única tabla de Amazon Keyspaces

Puede ver información básica sobre una tabla de Amazon Keyspaces en la consola, incluido el esquema de la clave principal, el tamaño de tabla facturable y las métricas relacionadas con la capacidad. El tamaño de la tabla se puede utilizar para calcular el costo mensual de almacenamiento de la tabla. Por ejemplo, 0,25\$ por GB en el este de EE. UU. (Virginia del Norte). Región de AWS

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). Puede utilizar esta información para calcular los costos actuales de lectura y escritura de la tabla. Tenga en cuenta que estos costos podrían cambiar, especialmente si ha configurado la tabla con el escalado automático de Amazon Keyspaces.

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 obtener más información, consulte [the section called “Trabajo con etiquetas”](#).

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 CQL, el o el AWS CLI SDK. 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. Para obtener más información, consulte [the section called “Creación de informes de asignación de costos”](#).

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 “Configure los modos read/write de capacidad”](#).

On-demand modo de capacidad

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).

On-demand las tablas se facturan contando el número de solicitudes reales con respecto a la tabla, por lo que solo se paga por lo que se utiliza y no por lo que se ha aprovisionado.

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.

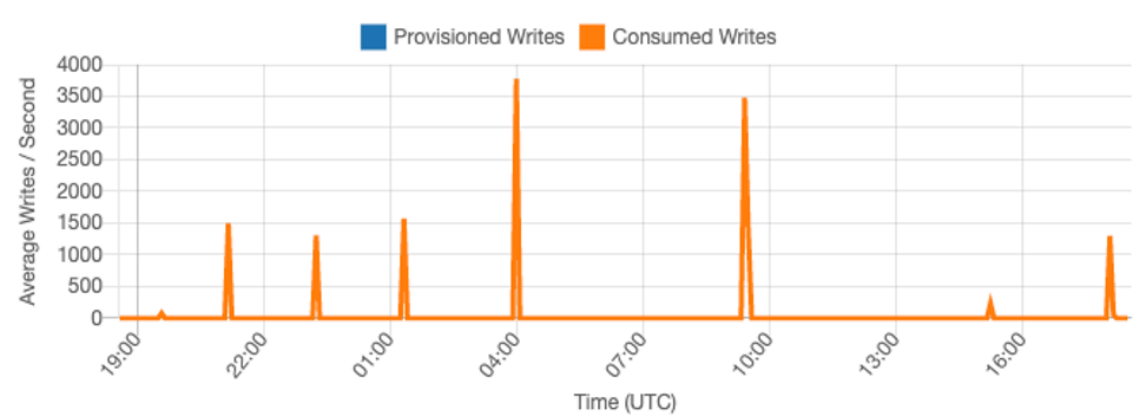
El modo de capacidad aprovisionada requiere un equilibrio entre no sobreaprovisionar ni subaprovisionar la tabla a fin de lograr tanto una baja incidencia de errores de capacidad de rendimiento insuficiente como costos optimizados.

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.

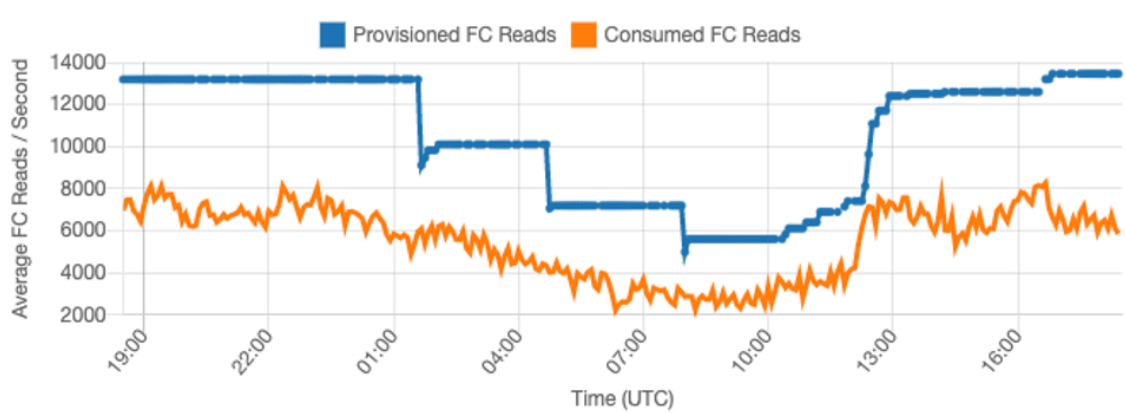
Dado que las tablas bajo demanda se facturan por solicitud, no necesita hacer nada más a nivel de tabla para optimizar costos. Debe evaluar periódicamente sus tablas bajo demanda para comprobar que la carga de trabajo siga teniendo las características mencionadas. Si la carga de trabajo se ha estabilizado, 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:

- Predicable/cyclical tráfico para una hora o un día determinados
- 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.

Utilización objetivo

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 sobreaprovisionada con capacidad reservada.

Mejora de 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 de 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 las tablas con un nivel de utilización objetivo bajo \(\$\leq 50\%\$ \)](#)
- [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 las tablas con un nivel de utilización objetivo bajo ($\leq 50\%$)

Puede usar AWS CLI o Consola de administración de AWS para monitorear e identificar las TargetValues políticas de Auto Scaling de su aplicación en sus recursos de Amazon Keyspaces.

Note

Cuando utilice tablas multirregión en el modo de capacidad aprovisionada con el escalado automático de Amazon Keyspaces, asegúrese de utilizar las operaciones de la API de Amazon Keyspaces para configurar el escalado automático. Las operaciones subyacentes de la API de Application Auto Scaling a las que Amazon Keyspaces llama en su nombre no tienen capacidades multirregionales. Para obtener más información, consulte [the section called “Visualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional”](#).

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:us-
east-1:111122223333:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-
scaling-policy",
      "PolicyName": "$<full-table-name>",
      "ServiceNamespace": "cassandra",
      "ResourceId": "keyspace/keyspace-name/table/table-name",
      "ScalableDimension": "cassandra:table:WriteCapacityUnits",
      "PolicyType": "TargetTrackingScaling",
      "TargetTrackingScalingPolicyConfiguration": {
        "TargetValue": 70.0,
        "PredefinedMetricSpecification": {
          "PredefinedMetricType": "KeyspacesWriteCapacityUtilization"
        }
      },
      "Alarms": [
        ...
      ],
      "CreationTime": "2022-03-04T16:23:48.641000+10:00"
    },
    {
      "PolicyARN": "arn:aws:autoscaling:us-
east-1:111122223333:scalingPolicy:<uuid>:resource/keyspaces/table/table-name-
scaling-policy",
      "PolicyName": "$<full-table-name>",
      "ServiceNamespace": "cassandra",
      "ResourceId": "keyspace/keyspace-name/table/table-name",
      "ScalableDimension": "cassandra:table:ReadCapacityUnits",
      "PolicyType": "TargetTrackingScaling",
      "TargetTrackingScalingPolicyConfiguration": {
        "TargetValue": 70.0,
        "PredefinedMetricSpecification": {
          "PredefinedMetricType": "KeyspacesReadCapacityUtilization"
        }
      },
      "Alarms": [
        ...
      ],
      "CreationTime": "2022-03-04T16:23:47.820000+10:00"
    }
  ]
}

```

```
}
```

Consola de administración de AWS

1. Inicie sesión Consola de administración de AWS y navegue hasta la página del CloudWatch servicio en [Cómo empezar con](#). Consola de administración de AWS Seleccione lo apropiado Región de AWS si es necesario.
2. En la barra de navegación izquierda, seleccione Tablas. En la página Tablas, seleccione el 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:

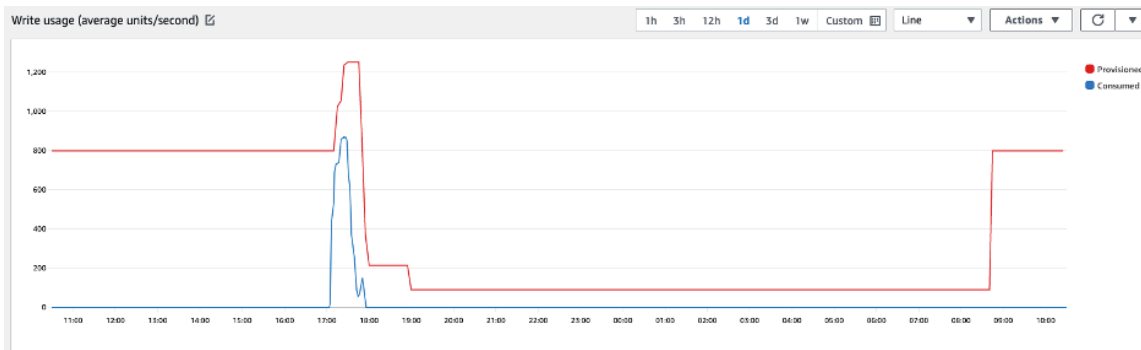
```
{
  "ScheduledActions": [
    {
      "ScheduledActionName": "my-5-8-scheduled-down-action",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:111122223333:scheduledAction:<uuid>:resource/keyspaces/table/table-
name:scheduledActionName/my-5-8-scheduled-down-action",
      "ServiceNamespace": "cassandra",
      "Schedule": "cron(15 17 ? * MON-FRI *)",
      "Timezone": "Australia/Brisbane",
      "ResourceId": "keyspace/keyspace-name/table/table-name",
      "ScalableDimension": "cassandra:table:WriteCapacityUnits",
      "ScalableTargetAction": {
        "MinCapacity": 90,
        "MaxCapacity": 1500
      },
      "CreationTime": "2022-03-15T17:30:25.100000+10:00"
    },
    {
      "ScheduledActionName": "my-8-5-scheduled-action",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:111122223333:scheduledAction:<uuid>:resource/keyspaces/table/table-
name:scheduledActionName/my-8-5-scheduled-action",
      "ServiceNamespace": "cassandra",
      "Schedule": "cron(45 8 ? * MON-FRI *)",
      "Timezone": "Australia/Brisbane",
      "ResourceId": "keyspace/keyspace-name/table/table-name",
      "ScalableDimension": "cassandra:table:WriteCapacityUnits",
      "ScalableTargetAction": {
        "MinCapacity": 800,
        "MaxCapacity": 1500
      },
    }
  ]
}
```

```

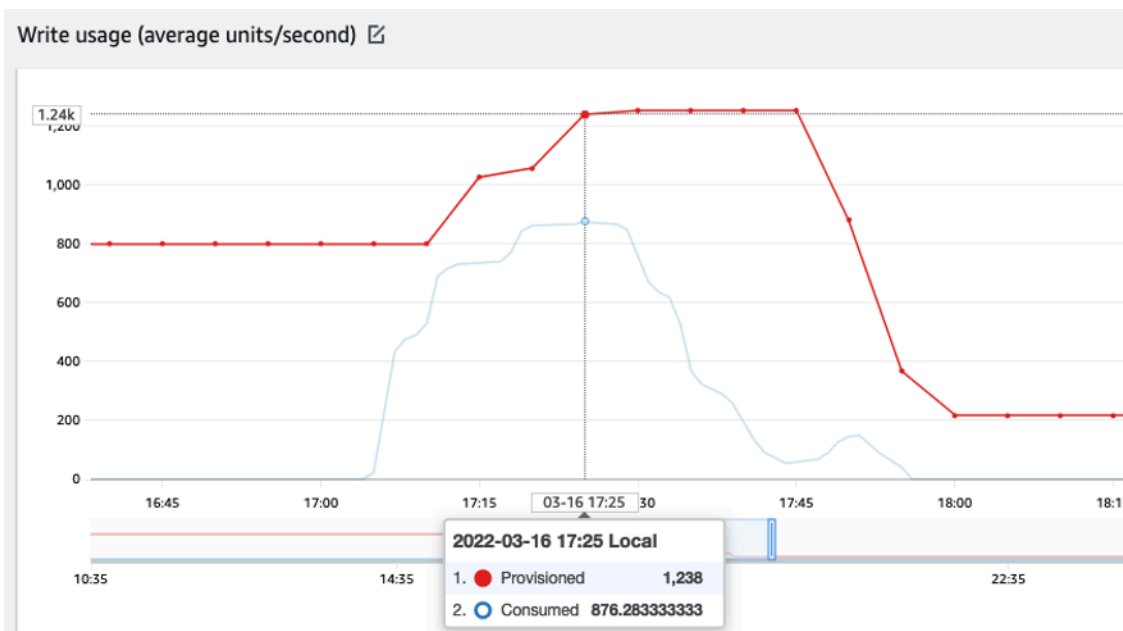
    "CreationTime": "2022-03-15T17:28:57.816000+10:00"
  }
]
}

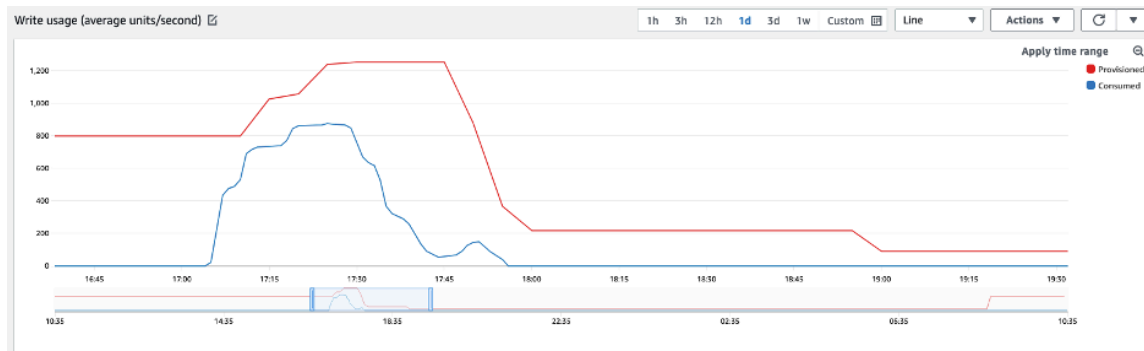
```

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.

En su lugar, considere utilizar el [modo de capacidad bajo demanda](#). On-demand las tablas son perfectas para cargas de trabajo con picos de actividad en las 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 optimizar los costos en Amazon Keyspaces

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 métricos 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](#)
- [Saneamiento de copias de seguridad de recuperación en un momento dado \(PITR\) no utilizadas](#)

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 Consola de administración de AWS 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"  
},  
}
```

Consola de administración de AWS

Los pasos siguientes le permiten evaluar la utilización de sus recursos a través de la Consola de administración de AWS.

1. Inicie sesión en Consola de administración de AWS 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 read/write de capacidad controla cómo se le cobra por el rendimiento de lectura y escritura y cómo 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 Consola de administración de AWS o la AWS Command Line Interface.

Saneamiento de copias de seguridad de recuperación en un momento dado (PITR) no utilizadas

Amazon Keyspaces ofrece Point-in-time recuperación, 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 [the section called “Copias de seguridad y restauración con recuperación en un momento dado”](#) para determinar si sus tablas tienen habilitadas copias de seguridad que tal vez ya no sean necesarias.

Evaluación de los patrones de uso de las tablas para optimizar el rendimiento y los costos

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

- [Realización de menos operaciones de lectura altamente coherente](#)
- [Habilitación del período de vida \(TTL\)](#)

Realización de 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.

Habilitación del período 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.

Evaluación de 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 Consola de administración de AWS.

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
<code><table-name></code>	El nombre de la tabla que vaya a analizar	SampleTable

Nombre del campo	Definición	Ejemplo
<period>	El periodo que utilizará para evaluar el objetivo de utilización, en segundos	Para un periodo de 1 hora, debe especificar: 3600
<start-time>	El comienzo del intervalo de evaluación, especificado en formato ISO8601	21-02-2022 T23:00:00
<end-time>	El final del intervalo de evaluación, especificado en formato ISO8601	2022-02-22 T06:00:00

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,
        40.947256944444445,
        25.618194444444444,
        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.

Consola de administración de AWS

1. Inicie sesión Consola de administración de AWS y navegue hasta la página de CloudWatch servicio en [Cómo empezar](#) con. Consola de administración de AWS 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 Métricas diagramadas (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 Agregar matemática, seguido de Común y, a continuación, seleccione la función 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 Media a 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 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 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 $100 * (m1 / \text{provisionedWCU})$.
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 ráfaga](#) es una función de Amazon Keyspaces que permite a los clientes consumir temporalmente RCUs/WCUs más de lo que se había aprovisionado originalmente (más del 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 “Uso de la 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 revisar 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 una relación de utilización baja y aceleración alta (máxima (ThrottleEvents) /minuto (ThrottleEvents) en el intervalo), esto puede ocurrir cuando tiene una carga de trabajo muy intensa en la que el tráfico aumenta significativamente en días (u horas del día) específicos, pero por lo demás es constantemente bajo. En estos escenarios, podría ser beneficioso utilizar [Application Auto Scaling programado](#).

Solución de problemas de Amazon Keyspaces (para Apache Cassandra)

En esta guía se describen pasos de solución de problemas para varios escenarios al trabajar con Amazon Keyspaces (para Apache Cassandra). Incluye información sobre cómo resolver errores generales, problemas de conexión, problemas de administración de la capacidad y errores del lenguaje de definición de datos (DDL).

- Errores generales
 - Solución de problemas de excepciones de alto nivel como `NoNodeAvailableException`, `NoHostAvailableException` y `AllNodesFailedException`.
 - Aislamiento de errores subyacentes de las excepciones del controlador de Java.
 - Implementación de políticas de reintentos y configuración de conexiones correctamente.
- Problemas de conectividad
 - Resolución de errores al conectarse a puntos de conexión de Amazon Keyspaces mediante `cqlsh` o controladores de cliente de Apache Cassandra.
 - Solución de problemas de conexiones de puntos finales de VPC, Cassandra-stress conexiones y errores de configuración de IAM.
 - Administración de las pérdidas de conexión durante la importación de datos.
- Errores de administración de capacidad
 - Reconocimiento y resolución de los errores de capacidad insuficiente relacionados con las tablas, las particiones y las conexiones.
 - Supervisión de las métricas relevantes de Amazon Keyspaces en Amazon CloudWatch Logs.
 - Optimización de las conexiones y el rendimiento para mejorar el desempeño.
- Errores de lenguaje de definición de datos (DDL)
 - Solución de errores de creación, acceso o restauración de tablas y espacios de claves.
 - Administración de errores relacionados con la configuración personalizada del período de vida (TTL), los límites de columnas y las eliminaciones de rangos.
 - Consideraciones para las cargas de trabajo de borrado pesadas.

Para obtener una guía de solución de problemas específica para el acceso a IAM, consulte [the section called “Resolución de problemas”](#). Para obtener más información sobre prácticas recomendadas de seguridad, consulte [the section called “Prácticas recomendadas de seguridad”](#).

Temas

- [Solución de errores generales en Amazon Keyspaces](#)
- [Solución de errores de conexión en Amazon Keyspaces](#)
- [Solución de errores de administración de capacidad en Amazon Keyspaces](#)
- [Solución de errores de lenguaje de definición de datos en Amazon Keyspaces](#)

Solución de errores generales en Amazon Keyspaces

¿Ha recibido errores generales? A continuación se indican algunos problemas comunes y la forma de resolverlos.

Errores generales

Recibe una de las siguientes excepciones de nivel superior que pueden producirse por muchos motivos diferentes.

- `NoNodeAvailableException`
- `NoHostAvailableException`
- `AllNodesFailedException`

Estas excepciones las genera el controlador del cliente y pueden producirse al establecer la conexión de control o al realizar solicitudes de `read/write prepare/execute /lotes`.

Cuando el error se produce al establecer la conexión de control, es una señal de que no se puede acceder a todos los puntos de contacto especificados en la aplicación. Cuando el error se produce al realizar `prepare/execute` consultas `read/write/`, indica que se han agotado todos los reintentos de esa solicitud. Cada reintento se realiza en un nodo diferente cuando se utiliza la política de reintentos predeterminada.

Cómo aislar el error subyacente de las excepciones de los controladores Java de nivel superior

Estos errores generales pueden deberse a problemas de conexión o al realizar `prepare/execute` las operaciones `read/write/`. Cabe esperar que se produzcan errores transitorios en los sistemas distribuidos, y deben solucionarse reintentando la solicitud. El controlador Java no vuelve a intentarlo automáticamente cuando se producen errores de conexión, por lo que se recomienda implementar la política de reintentos al establecer la conexión del controlador en la aplicación. Para obtener una descripción detallada de las prácticas recomendadas para la conexión, consulte [the section called “Conexiones”](#).

De forma predeterminada, el controlador Java se establece en `false idempotence` para todas las solicitudes, lo que significa que el controlador Java no reintenta automáticamente la solicitud `read/write /prepare` fallida. Para configurar `idempotence` como `true` e indicarle al controlador que vuelva a intentar las solicitudes fallidas, puede hacerlo de diferentes maneras. Este es un ejemplo de cómo se puede configurar la idempotencia mediante programación para una sola solicitud en su aplicación Java.

```
Statement s = new SimpleStatement("SELECT * FROM my_table WHERE id = 1");
s.setIdempotent(true);
```

También puede establecer la idempotencia predeterminada mediante programación para toda la aplicación Java, tal y como se muestra en el siguiente ejemplo.

```
// Make all statements idempotent by default:
cluster.getConfiguration().getQueryOptions().setDefaultIdempotence(true);
//Set the default idempotency to true in your Cassandra configuration
basic.request.default-idempotence = true
```

Otra recomendación es crear una política de reintentos en el nivel de la aplicación. En este caso, la aplicación debe detectar la `NoNodeAvailableException` y reintentar la solicitud. Recomendamos realizar 10 reintentos con un retroceso exponencial a partir de 10 ms y continuar hasta los 100 ms con un tiempo total de 1 segundo para todos los reintentos.

Otra opción es aplicar la política de reintentos exponenciales de Amazon Keyspaces al establecer la conexión del controlador Java disponible en [Github](#).

Confirme que ha establecido conexiones con más de un nodo cuando utilice la política de reintentos predeterminada. Para ello, utilice la siguiente consulta en Amazon Keyspaces.

```
SELECT * FROM system.peers;
```

Si la respuesta de esta consulta está vacía, esto indica que está trabajando con un único nodo para Amazon Keyspaces. Si utiliza la política de reintentos predeterminada, no se producirán reintentos porque el reintento predeterminado siempre se produce en un nodo diferente. Para obtener más información sobre cómo establecer conexiones a través de puntos de conexión de VPC, consulte [the section called “Conexiones de puntos de conexión de VPC”](#).

Para ver un tutorial paso a paso que muestra cómo establecer una conexión con Amazon Keyspaces mediante el controlador Cassandra Datastax 4.x, consulte [the section called “Complemento de autenticación para Java 4.x”](#).

Solución de errores 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](#)

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 de conexión 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.amazonaws.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 SSL/TLS certificado, 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. Para obtener más información, consulte [the section called “Configuración manual de conexiones de `cqlsh` para TLS”](#).

```
certfile = path_to_file/keyspaces-bundle.pem
```

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/keyspaces-bundle.pem
#

# cqlsh cassandra.us-east-1.amazonaws.com -u "USERNAME" -p "PASSWORD" --ssl
Connection error: ('Unable to connect to any servers', {'3.234.248.192': IOError(2, 'No
such file or directory')})
#
```

Para resolver este problema, compruebe que la ruta al archivo de certificado en su ordenador sea correcta. Para obtener más información, consulte [the section called “Configuración manual de conexiones de cqlsh para TLS”](#).

Está intentando conectarse a Amazon Keyspaces con `cqlsh`, pero recibe un error **[X509] PEM lib**.

Este podría ser el caso si el pem archivo del SSL/TLS certificado no es válido y se produce 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, asegúrese de haber descargado los certificados digitales necesarios. Para obtener más información, consulte [the section called “Configuración manual de conexiones de cqlsh para TLS”](#).

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error de SSL **unknown**.

Este puede ser el caso si el pem archivo del SSL/TLS certificado está vacío y se produce 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, asegúrese de haber descargado los certificados digitales necesarios. Puede confirmarlo siguiendo los pasos del tema siguiente [the section called “Configuración manual de conexiones de cqlsh para TLS”](#).

Está intentando conectarse a Amazon Keyspaces con cqlsh, pero recibe un error **SSL: CERTIFICATE_VERIFY_FAILED**.

Este podría ser el caso si no se pudo comprobar el archivo de SSL/TLS certificado, 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, asegúrese de haber descargado los certificados digitales necesarios. Puede confirmarlo siguiendo los pasos del tema siguiente [the section called “Configuración manual de conexiones de cqlsh para TLS”](#).

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 con este comando también se 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, necesitará 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. De no ser así, debe crear un nuevo grupo de seguridad para la instancia de 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, CQLSH/CASSANDRA elija una opción 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 finalmente 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](#).

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 “Identity-based ejemplos de políticas”](#).

Para obtener una guía de solución de problemas específica para el acceso a IAM, consulte [the section called “Resolución 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 el **USERNAME** y **PASSWORD** del código coinciden 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 Soporte.

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.

Estás intentando conectarte a una tabla de Amazon Keyspaces mediante el controlador DataStax Java, pero aparece un **NodeUnavailableException** error.

Si se interrumpe la conexión en la que se intenta realizar la solicitud, se produce el siguiente error.

```
[com.datastax.oss.driver.api.core.NodeUnavailableException: No connection
was available to Node(endPoint=vpce-22ff22f2f22222fff-aa1bb234.cassandra.us-
west-2.vpce.amazonaws.com/11.1.1111.222:9142, hostId=1a23456b-
c77d-8888-9d99-146cb22d6ef6, hashCode=123ca4567)]
```

Para resolver este problema, busque el valor de latido y bájelo a 30 segundos si es más alto.

```
advanced.heartbeat.interval = 30 seconds
```

A continuación, busque el tiempo de espera asociado y asegúrese de que el valor esté establecido en al menos 5 segundos.

```
advanced.connection.init-query-timeout = 5 seconds
```

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 “Cree credenciales de IAM para la autenticación 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 “Cree credenciales de IAM para la autenticación AWS”](#).

Está intentando conectarse a una tabla de Amazon Keyspaces con un controlador, pero recibe un error **Invalid signature**.

Esto podría ocurrir si alguno de los componentes necesarios para la firma es incorrecto o no está definido correctamente para la sesión.

- `AWS_ACCESS_KEY_ID`
- `AWS_SECRET_ACCESS_KEY`
- `AWS_DEFAULT_REGION`

El siguiente error es un ejemplo de claves de acceso no válidas.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
{'11.234.11.234:9142':
AuthenticationFailed('Failed to authenticate to 11.234.11.234:9142: Error from server:
code=0100
[Bad credentials] message="Authentication failure: Invalid signature"')})
```

Para resolver este problema, compruebe que las claves de acceso y las se Región de AWS hayan configurado correctamente para que el complemento SiGv4 acceda a Amazon Keyspaces. Para obtener más información sobre claves de acceso y el complemento SigV4, consulte [the section called “Cree credenciales de IAM para la autenticación 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 ver un tutorial paso a paso para obtener información sobre cómo configurar una conexión a través de puntos de conexión de VPC de interfaz para Amazon Keyspaces, consulte [the section called “Conexión con puntos de conexión de VPC”](#).

1. Confirme que la entidad de IAM utilizada para conectarse a Amazon Keyspaces read/write tenga acceso a la tabla de usuarios 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 debe tener un aspecto similar al de este ejemplo y devolver entre 2 y 6 nodos con direcciones IPv4 privadas al conectarse desde una red IPv4, según la configuración de la VPC y la región. AWS

```
peer
-----
192.0.2.0.15
192.0.2.0.24
192.0.2.0.13
192.0.2.0.7
192.0.2.0.8

(5 rows)
```

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, el usuario debe tener al menos SELECT acceso a las tablas del sistema, ya que la mayoría de los controladores leen el sistema keyspaces/tables cuando establecen 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 “Resolución 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 errores 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. `Insufficient-capacity` Los eventos que provocan errores del lado del cliente se pueden clasificar en estos tres grupos según el 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 “Configure los modos read/write de capacidad”](#).
- **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 aprender cómo configurar conexiones para Amazon Keyspaces, consulte [the section called “Configuración de conexiones”](#). Para obtener más información sobre cómo optimizar conexiones con puntos de conexión de VPC, consulte [the section called “Conexiones de puntos de conexión de VPC”](#).

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 read/write tráfico entre más particiones.
- `WriteThrottleEvents`: solicitudes a Amazon Keyspaces que superan la capacidad de escritura de una tabla.

Para obtener más información CloudWatch, consulte [the 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 “Administración de la capacidad de rendimiento con escalado automático”](#).
- Elija el modo On-demand de capacidad para la tabla. Para obtener más información, consulte [the section called “Configuración del 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.

Cuando aparece el error `StoragePartitionThroughputCapacityExceeded`, la capacidad de la partición se ha excedido temporalmente. Esto podría administrarse automáticamente mediante el uso de capacidad adaptativa o de capacidad bajo demanda. Recomendamos revisar el modelo de datos

para distribuir read/write el tráfico entre más particiones y mitigar estos errores. 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 read/write el tráfico 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 aparición de `PerConnectionRequestRateExceeded` se debe 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. Para obtener más información, consulte

[the section called “Configuración de las políticas de reintentos”](#).

- 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 carga y aumenta read/write el rendimiento. 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 instrucciones paso a paso, consulte [the 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.

Para obtener más información sobre el tamaño de almacenamiento de la tabla, consulte [the section called “Evaluación de los costos en el nivel de tabla”](#). También 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 “Estimación del tamaño de las filas”](#).

Solución de errores 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 nuevos espacios de claves y tablas en el Consola de administración de AWS, lo que indica si un espacio de claves 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 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 recuperación en un momento dado \(PITR\) de Amazon Keyspaces, pero la restauración falla.](#)
- [Estoy intentando editar la configuración personalizada de Time to Live \(TTL\), pero la operación falla INSERT/UPDATE](#)
- [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 “Comprobación del estado de creación del espacio 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 “Comprobación del estado de creación de las tablas”](#).

Estoy intentando restaurar una tabla mediante la recuperación en un momento dado (PITR) de Amazon Keyspaces, pero la restauración falla.

Si al intentar restaurar una tabla de Amazon Keyspaces con la recuperación en un momento dado (PITR) ve que el proceso de restauración comienza pero no finaliza correctamente, es posible que no haya configurado todos los permisos necesarios para el proceso de restauración en esta tabla en particular.

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 “Configuración de los permisos de IAM para la restauración”](#).

Estoy intentando editar la configuración personalizada de Time to Live (TTL), pero la operación falla INSERT/UPDATE

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 “Actualización de una tabla con un TTL personalizado”](#).

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.

Monitoreo de Amazon Keyspaces (para Apache Cassandra)

La supervisión es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el rendimiento de Amazon Keyspaces y del resto AWS de sus soluciones. AWS proporciona las siguientes herramientas de supervisión para ver Amazon Keyspaces, informar cuando algo va mal y tomar medidas automáticas cuando sea necesario:

- Amazon Keyspaces ofrece un panel preconfigurado Consola de administración de AWS que muestra la latencia y los errores agregados en todas las tablas de la cuenta.
- Amazon CloudWatch monitorea tus AWS recursos y las aplicaciones en las que AWS ejecutas 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, consulta la [Guía del CloudWatch usuario de Amazon](#).
- CloudWatch Las alarmas de Amazon supervisan una única métrica durante un período de tiempo que usted especifique y realizan una o más acciones en función del valor de la métrica en relación con un umbral determinado durante varios períodos 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.

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, consulte [Supervisión de Amazon Keyspaces con Amazon CloudWatch](#).

- Amazon CloudWatch Logs le permite supervisar, almacenar y acceder a sus archivos de registro desde Amazon Keyspaces CloudTrail, tablas y otras fuentes. CloudWatch Los registros pueden monitorear la información de los archivos de registro y notificarle cuando se alcancen ciertos umbrales. También se pueden archivar los datos del registro en un almacenamiento de larga duración. Para obtener más información, consulta la [Guía del usuario CloudWatch de Amazon Logs](#).

- AWS CloudTrail captura las llamadas a la API y otros eventos relacionados que realiza la Cuenta de AWS o que se realizan en nombre de esta. Además, entrega los archivos de registro 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 estas se realizaron. Para obtener más información, consulte la [Guía del usuario de AWS CloudTrail](#).

Amazon EventBridge es un servicio de bus de eventos sin servidor que facilita la conexión de sus aplicaciones con datos de diversas fuentes. EventBridge ofrece un flujo de datos en tiempo real desde sus propias aplicaciones, aplicaciones Software-as-a-Service (SaaS) y AWS servicios, y dirige esos 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, consulta la [Guía del EventBridge usuario de Amazon](#).

Temas

- [Supervisión de Amazon Keyspaces con Amazon CloudWatch](#)
- [Registrar las 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 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](#)
- [Visualización de las métricas de Amazon Keyspaces en CloudWatch](#)
- [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 <code>SystemErrors</code> 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 <code>ConsumedReadCapacityUnits</code> y <code>ProvisionedReadCapacityUnits</code> en el intervalo que desee monitorear. 2. Cambie la Estadística para <code>ConsumedReadCapacityUnits</code> 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 <code>ConsumedReadCapacityUnits</code> y divida la métrica por la función CloudWatch PERIOD de la métrica (<code>metric_id/ (PERIOD (metric_id))</code>). 5. Deseleccione <code>ConsumedReadCapacityUnits</code>.

¿Cómo...?	Métricas relevantes
	Ahora puede comparar su capacidad de lectura 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.
¿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, este envía métricas y dimensiones a Amazon. CloudWatch Todas las métricas se agregan y notifican cada minuto. En las siguientes secciones se ofrece un resumen de todas las métricas disponibles para Amazon Keyspaces.

Temas

- [Métricas de Amazon Keyspaces](#)
- [Dimensiones de las métricas de Amazon Keyspaces](#)
- [Las métricas de Amazon Keyspaces cambian la captura de datos \(CDC\)](#)
- [Dimensions for Amazon Keyspaces cambia las métricas de captura de datos \(CDC\)](#)

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	Description (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. 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

Métrica	Description (Descripción)
	Estadísticas válidas: • Maximum: el número máximo de unidades de capacidad de escritura que puede utilizar una tabla de la cuenta.
<code>AccountProvisionedReadCapacityUtilization</code>	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	Description (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	Description (Descripción)
<code>BillableTableSizeInBytes</code>	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: • <code>Maximum</code>: el tamaño máximo de almacenamiento de la tabla. • <code>Minimum</code>: el tamaño mínimo de almacenamiento de la tabla. • <code>Average</code>: el tamaño promedio de almacenamiento de la tabla. Esta métrica se calcula en intervalos de 4 a 6 horas.

Métrica	Description (Descripción)
<code>ConditionalCheckFailedRequests</code>	El número de solicitudes de escritura de transacciones ligeras (LWT) fallidas. Las 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, <code>ConditionalCheckFailedRequests</code> se incrementa en uno. Las comprobaciones de condiciones que se evalúan como falsas consumen unidades de capacidad de escritura en función del tamaño de la fila. Para obtener más información, consulte the section called “Estimación del consumo de capacidad de LWT”. Unidades: Count Dimensiones: Keyspace, TableName Estadísticas válidas: • Minimum • Maximum • Average • SampleCount • Sum

Métrica	Description (Descripción)
ConsumedReadCapacityUnits	El número de unidades de capacidad de lectura consumidas durante el periodo especificado. Para obtener más información, consulte el modo de Read/Write capacidad.  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	Description (Descripción)
	• Average: la capacidad de lectura promedio por solicitud consumida.  Note El valor <code>Average</code> 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 <code>ConsumedReadCapacityUnits</code>. • SampleCount: el número de solicitudes a Amazon Keyspaces, aunque no se haya consumido capacidad de lectura.  Note El valor <code>SampleCount</code> está influenciado por periodos de inactividad donde el valor de la muestra será cero.

Métrica	Description (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 Read/Write de capacidad.  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	Description (Descripción)
	• <code>Maximum</code>: el número máximo de unidades de capacidad de escritura consumidas por cualquier solicitud individual a la tabla. • <code>Average</code>: la capacidad de escritura promedio por solicitud consumida.  Note El valor <code>Average</code> está influenciado por periodos de inactividad donde el valor de la muestra será cero. • <code>Sum</code>: el total de unidades de capacidad de escritura consumidas. Esta es la estadística más útil para la métrica <code>ConsumedWriteCapacityUnits</code>. • <code>SampleCount</code>: el número de solicitudes a Amazon Keyspaces, aunque no se haya consumido capacidad de escritura.  Note El valor <code>SampleCount</code> está influenciado por periodos de inactividad donde el valor de la muestra será cero.

Métrica	Description (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	Description (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	Description (Descripción)
<code>PerConnectionRequestRateExceeded</code>	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. Cuando se utiliza la replicación multirregional, 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 <code>PerConnectionRequestRateExceeded</code>. No existe límite para el número de conexiones que pueda tener en Amazon Keyspaces. Unidades: Count Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: • <code>SampleCount</code> • <code>Sum</code>

Métrica	Description (Descripción)
ProvisionedReadCapacityUnits	El número de unidades de capacidad de lectura aprovisionadas para una tabla. La dimensión <code>TableName</code> devuelve las <code>ProvisionedReadCapacityUnits</code> de la tabla. Unidades: Count Dimensiones: Keyspace, TableName Estadísticas válidas: • Minimum: la configuración más baja para la capacidad de lectura aprovisionada. Si utiliza <code>ALTER TABLE</code> para aumentar la capacidad de lectura, esta métrica muestra el valor más bajo de <code>ReadCapacityUnits</code> aprovisionado durante este periodo de tiempo. • Maximum: la configuración más alta para la capacidad de lectura aprovisionada. Si utiliza <code>ALTER TABLE</code> para reducir la capacidad de lectura, esta métrica muestra el valor más alto de <code>ReadCapacityUnits</code> aprovisionado durante este período de tiempo. • Average: la capacidad de lectura aprovisionada promedio. La métrica <code>ProvisionedReadCapacityUnits</code> 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	Description (Descripción)
ProvisionedWriteCapacityUnits	El número de unidades de capacidad de escritura aprovisionadas para una tabla. La dimensión <code>TableName</code> devuelve las <code>ProvisionedWriteCapacityUnits</code> de la tabla. Unidades: Count Dimensiones: Keyspace, TableName Estadísticas válidas: • Minimum: la configuración más baja para la capacidad de escritura aprovisionada. Si utiliza <code>ALTER TABLE</code> para aumentar la capacidad de escritura, esta métrica muestra el valor más bajo de <code>WriteCapacityUnits</code> aprovisionado durante este período de tiempo. • Maximum: la configuración más alta para la capacidad de escritura aprovisionada. Si utiliza <code>ALTER TABLE</code> para reducir la capacidad de escritura, esta métrica muestra el valor más alto de <code>WriteCapacityUnits</code> aprovisionado durante este período de tiempo. • Average: la capacidad de escritura aprovisionada promedio. La métrica <code>ProvisionedWriteCapacityUnits</code> 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	Description (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	Description (Descripción)
ReturnedItemCountBySelect	El número de filas devueltas por las SELECT consultas de varias filas durante el período de tiempo especificado. Multi-row SELECTLas consultas son consultas que no contienen la clave principal completa, como las consultas de rangos o escaneos de tablas completas. El número de filas devueltas no es necesariamente el mismo que el número de filas que se evaluaron. Por ejemplo, suponga que solicitó una <code>SELECT *</code> con <code>ALLOW FILTERING</code> en una tabla que tiene 100 filas, pero especificó una cláusula <code>WHERE</code> que redujo los resultados de forma que solo se devuelven 15 filas. En este caso, la respuesta de SELECT contendría un <code>ScanCount</code> de 100 y un <code>Count</code> de 15 filas devueltas. Unidades: Count Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: • Minimum • Maximum • Average • SampleCount • Sum

Métrica	Description (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 revisar su modelo de datos para distribuir read/write el tráfico 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	Description (Descripción)
SuccessfulRequestLatency	Las solicitudes exitosas a Amazon Keyspaces durante el periodo de tiempo especificado. <code>SuccessfulRequestLatency</code> puede proporcionar dos tipos diferentes de información: • El tiempo transcurrido para las solicitudes correctas (<code>Minimum</code>, <code>Maximum</code>, <code>Sum</code> o <code>Average</code>). • El número de solicitudes realizadas correctamente (<code>SampleCount</code>). <code>SuccessfulRequestLatency</code> refleja la actividad únicamente en Amazon Keyspaces y no tiene en cuenta la latencia de la red ni la actividad del lado del cliente. Unidades: <code>Milliseconds</code> Dimensiones: <code>Keyspace</code>, <code>TableName</code>, <code>Operation</code> Estadísticas válidas: • <code>Minimum</code> • <code>Maximum</code> • <code>Average</code> • <code>SampleCount</code>

Métrica	Description (Descripción)
SystemErrors	Las solicitudes a Amazon Keyspaces que generan un <code>ServerError</code> durante el periodo especificado. Un <code>ServerError</code> 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 <code>SystemReconciliationDelete</code> 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 <code>SystemReconciliationDeletes</code>. O bien, para eliminar una fila completa que contenga 3,5 KB de datos se requieren 4 <code>SystemReconciliationDeletes</code>. Unidades: Count Dimensiones: Keyspace, TableName Estadísticas válidas: • Sum: el número total de <code>SystemReconciliationDeletes</code> consumidos en un periodo de tiempo.

Métrica	Description (Descripción)
TTLDeletes	Las unidades consumidas para eliminar o actualizar datos en una fila utilizando Tiempo de vida (TTL). Cada <code>TTLDelete</code> 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 <code>TTLDeletes</code> consumidos en un periodo de tiempo.

Métrica	Description (Descripción)
UserErrors	Las solicitudes a Amazon Keyspaces que generan un error <code>InvalidRequest</code> durante el periodo especificado. <code>InvalidRequest</code> los errores suelen representar un error del lado del cliente, por ejemplo, una combinación de parámetros no válida, un intento de actualizar una tabla que no existe o una firma de solicitud incorrecta. <code>UserErrors</code> representan 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	Description (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.

Las métricas de Amazon Keyspaces cambian la captura de datos (CDC)

Las siguientes métricas están disponibles en su CloudWatch cuenta para monitorear las llamadas a la API al punto final de captura de datos de cambios (CDC) de Amazon Keyspaces.

Métrica	Description (Descripción)
ReturnedRecordsCount	La cantidad de registros de transmisión devueltos por <code>GetRecords</code> las operaciones durante el período de tiempo especificado. Unidades: Count Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: Minimum

Métrica	Description (Descripción)
	• <code>Maximum</code> • <code>Average</code> • <code>SampleCount</code> • <code>Sum</code>
SuccessfulRequestLatency	Las solicitudes exitosas a Amazon Keyspaces durante el periodo de tiempo especificado. <code>SuccessfulRequestLatency</code> puede proporcionar dos tipos diferentes de información: • El tiempo transcurrido para las solicitudes correctas (<code>Minimum</code>, <code>Maximum</code>, <code>Sum</code> o <code>Average</code>). • El número de solicitudes realizadas correctamente (<code>SampleCount</code>). <code>SuccessfulRequestLatency</code> refleja la actividad únicamente dentro de las API CDC de Amazon Keyspaces y no tiene en cuenta la latencia de la red ni la actividad del lado del cliente. Unidades: <code>Milliseconds</code> Dimensiones: <code>Keyspace</code>, <code>TableName</code>, <code>Operation</code> Estadísticas válidas: • <code>Minimum</code> • <code>Maximum</code> • <code>Average</code> • <code>SampleCount</code>

Métrica	Description (Descripción)
<code>SystemErrors</code>	Las solicitudes a Amazon Keyspaces CDC que generan un código de HTTP 500 estado durante el período de tiempo especificado. Un código HTTP 500 de estado suele indicar un error de servicio interno. Unidades: Count Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: • Sum • SampleCount
<code>UserErrors</code>	Solicitudes a Amazon Keyspaces CDC que generan un código de HTTP 400 estado durante el período de tiempo especificado. <code>UserErrors</code> representa la suma de HTTP 400 errores de Amazon Keyspaces CDC en el momento actual Región de AWS y en el actual. Cuenta de AWS Unidades: Count Dimensiones: Keyspace, TableName, Operation Estadísticas válidas: • Sum • SampleCount

Dimensions for Amazon Keyspaces cambia las métricas de captura de datos (CDC)

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	Description (Descripción)
<code>StreamLabel</code>	Esta dimensión limita los datos a una transmisión de captura de datos de cambios (CDC) específica cuando se especifican <code>KeyspaceName</code> y <code>TableName</code> se especifican.
<code>Keyspace</code>	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.
<code>TableName</code>	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 <code>Keyspace</code> .
<code>Operation</code>	Esta dimensión limita los datos a una de las siguientes operaciones de la API Amazon Keyspaces Streams: <code>GetRecords</code> , <code>GetStream</code> , <code>GetShardIterator</code> , y <code>ListStreams</code> .

Visualización de las métricas de Amazon Keyspaces en CloudWatch

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 de AWS. 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"
```

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 periodos 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, consulte [the 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.

Registrar las llamadas a la API de Amazon Keyspaces con AWS CloudTrail

Amazon Keyspaces está integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio 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 (DML) 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, puede habilitar la entrega continua de CloudTrail eventos a un bucket de Amazon Simple Storage Service (Amazon S3), incluidos los eventos de Amazon Keyspaces.

Si no configura una ruta, podrá ver los eventos compatibles más recientes en la CloudTrail consola, en el historial de eventos. Con 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 la hizo, cuándo se realizó y detalles adicionales.

Para obtener más información CloudTrail, consulte la [Guía del AWS CloudTrail usuario](#).

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 \(DML\) 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 Amazon Keyspaces iniciada CloudTrail incluye parámetros de solicitud que se expresan en el lenguaje de consulta CQL. Para obtener más información, consulte la [Referencia del lenguaje CQL](#).

Puede ver, buscar y descargar eventos recientes en su Cuenta de AWS. Para obtener más información, consulte [Visualización de eventos con el historial de CloudTrail eventos](#).

Para tener un registro continuo de los eventos en su Cuenta de AWS entorno, incluidos los eventos de Amazon Keyspaces, cree una ruta. Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando crea una ruta en la consola, la ruta se aplica a todas AWS las regiones. La ruta registra los eventos de todas las regiones de la AWS partición y envía los archivos de registro al bucket de Amazon S3 que especifique. Además, puede configurar otros AWS servicios para analizar más a fondo los datos de eventos recopilados en los CloudTrail registros y actuar en función de ellos.

Para obtener más información, consulte los siguientes temas en la Guía del usuario de AWS CloudTrail :

- [Introducción a la creación de registros de seguimiento](#)
- [CloudTrail servicios e integraciones compatibles](#)
- [Configuración de las notificaciones de Amazon SNS para CloudTrail](#)
- [Recibir archivos de CloudTrail registro de varias regiones](#)
- [Recibir archivos de CloudTrail registro de varias cuentas](#)

Cada entrada de registro o evento contiene información sobre 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 root o AWS Identity and Access Management (IAM).
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro AWS servicio.

Para obtener más información, consulte el [elemento userIdentity de CloudTrail](#).

Información sobre el lenguaje de definición de datos (DDL) de Amazon Keyspaces en CloudTrail

CloudTrail está habilitada en tu cuenta Cuenta de AWS al crear la cuenta. Cuando se produce una actividad de DDL en Amazon Keyspaces, esa actividad se registra automáticamente como CloudTrail un evento junto con AWS otros eventos de servicio en el historial de eventos. En la tabla siguiente se muestran las instrucciones DDL que se registran para Amazon Keyspaces.

CloudTrail eventName	Instrucción	Acción de CQL	AWS Acción del SDK
CreateKeyspace	DDL	CREATE KEYSPACE	CreateKeyspace
AlterKeyspace	DDL	ALTER KEYSPACE	UpdateKeyspace
DropKeyspace	DDL	DROP KEYSPACE	DeleteKeyspace

CloudTrail eventName	Instrucción	Acción de CQL	AWS Acción del SDK
CreateTable	DDL	CREATE TABLE	CreateTable
DropTable	DDL	DROP TABLE	DeleteTable
AlterTable	DDL	ALTER TABLE	UpdateTable , TagResource , UntagResource
CreateUdt	DDL	CREATE TYPE	CreateType
DropUdt	DDL	DROP TYPE	DeleteType
GetStream	DDL	no hay ninguna acción de CQL disponible	GetStream
ListStreams	DDL	no hay ninguna acción de CQL disponible	ListStreams

Información sobre el lenguaje de manipulación de datos (DML) de Amazon Keyspaces en CloudTrail

Para habilitar el registro de las declaraciones DML de Amazon Keyspaces CloudTrail, primero debe habilitar el registro de la actividad de la API del plano de datos. CloudTrail Puede empezar a registrar los eventos DML de Amazon Keyspaces en rutas nuevas o existentes si elige registrar la actividad del tipo de evento de datos de la tabla Cassandra mediante la CloudTrail consola o configurando el `resources.type` valor para que utilice las operaciones de la AWS CLI o `AWS::Cassandra::Table` la API. CloudTrail Para obtener más información, consulte [Registro de eventos de datos](#).

Para obtener más información y un ejemplo que muestre cómo crear alarmas para eventos de datos, consulte la siguiente publicación en el blog de AWS bases de datos [Using DML auditing for Amazon Keyspaces \(para Apache Cassandra\)](#).

En la siguiente tabla se muestran los eventos de datos que registra. CloudTrail Cassandra table

CloudTrail eventName	Instrucción	Acción de CQL	AWS Acciones del SDK
Select	DML	SELECT	GetKeyspace , GetTable, GetType, ListKeyspaces , ListTables , ListTypes , ListTagsForResource
Inserción	DML	INSERT	no hay acciones AWS del SDK disponibles
Actualización	DML	UPDATE	no hay acciones AWS del SDK disponibles
Eliminar	DML	DELETE	no hay acciones AWS del SDK disponibles
GetShardIterator	DML	no hay ninguna acción de CQL disponible	GetShardIterator
GetRecords	DML	no hay ninguna acción de CQL disponible	GetRecords

Comprensión de las entradas del archivo de registro de Amazon Keyspaces

CloudTrail los archivos de registro contienen una o más entradas de registro. Un evento representa una solicitud única de cualquier fuente e incluye información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra una entrada de CloudTrail registro que muestra las DropTable acciones CreateKeyspace DropKeyspaceCreateTable,, y:

```
{
  "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"
          }
        },
        "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": "e5aebac-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\n\t\tKEY((\"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 AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted 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 AWS los servicios en la 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 AWS servicio 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 aprenderá a utilizar otros AWS servicios que pueden ayudarle a supervisar y proteger 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](#) 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 en la que se ejecutan todos los. Nube de AWS Eres responsable de mantener el control sobre el contenido alojado en esta infraestructura. También eres 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 privacidad de datos](#) y los . Para obtener más información sobre la protección de datos en Europa, consulte el [Centro del Reglamento General de Protección de Datos \(RGPD\)](#).

Para fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.
- Se utiliza SSL/TLS para comunicarse con AWS los recursos. Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad de los usuarios con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte [Cómo trabajar con CloudTrail senderos](#) en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utiliza servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger la información confidencial almacenada en Amazon S3.
- Si necesita módulos criptográficos validados por FIPS 140-3 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte [Estándar de procesamiento de la información federal \(FIPS\) 140-3](#).

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 incluye cuando trabajas con Amazon Keyspaces u otros dispositivos Servicios de AWS mediante la consola, la API o AWS los AWS CLI SDK. Cualquier

dato que introduzca en etiquetas o campos de formato libre utilizados para los 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 los datos mediante el estándar de cifrado avanzado (AES-256) de 256 bits. Esto ayuda a proteger los datos del acceso no autorizado al almacenamiento subyacente.

Amazon Keyspaces cifra y descifra los datos de las tablas y las transmisiones de forma transparente. Amazon Keyspaces utiliza el cifrado de sobres 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 “Funcionamiento”](#). Para obtener más información sobre AWS KMS conceptos como el cifrado de sobres, consulte los conceptos de los [servicios AWS KMS de administración en la Guía para desarrolladores](#). 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 tiene el control total sobre la clave gestionada por el cliente (de AWS KMS pago).

Amazon Keyspaces cifra automáticamente las transmisiones de captura de datos de cambios (CDC) con la misma clave que la tabla subyacente. Para obtener más información sobre los CDC, consulte [the section called “Trabajando con las transmisiones de los CDC”](#)

Puede cambiar entre la clave gestionada por el cliente Clave propiedad de AWS y la clave gestionada 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 aprender a hacerlo, consulte [Cifrado en reposo: Cómo utilizar claves administradas por el cliente para cifrar tablas en Amazon Keyspaces..](#)

El cifrado en reposo mediante la opción predeterminada de Claves propiedad de AWS se ofrece sin coste 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 Regiones de AWS, incluidas las regiones de AWS China (Pekín) y AWS China (Ningxia). 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 Keyspaces.](#)

Cifrado en reposo: Cómo funciona en Amazon Keyspaces

El cifrado de Amazon Keyspaces (para Apache Cassandra) en reposo cifra los datos mediante 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 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 tiene el control total sobre la clave gestionada por el cliente (de AWS KMS pago).

AWS KMS clave (clave KMS)

El cifrado en reposo protege todos los datos de Amazon Keyspaces con una AWS KMS clave. 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.

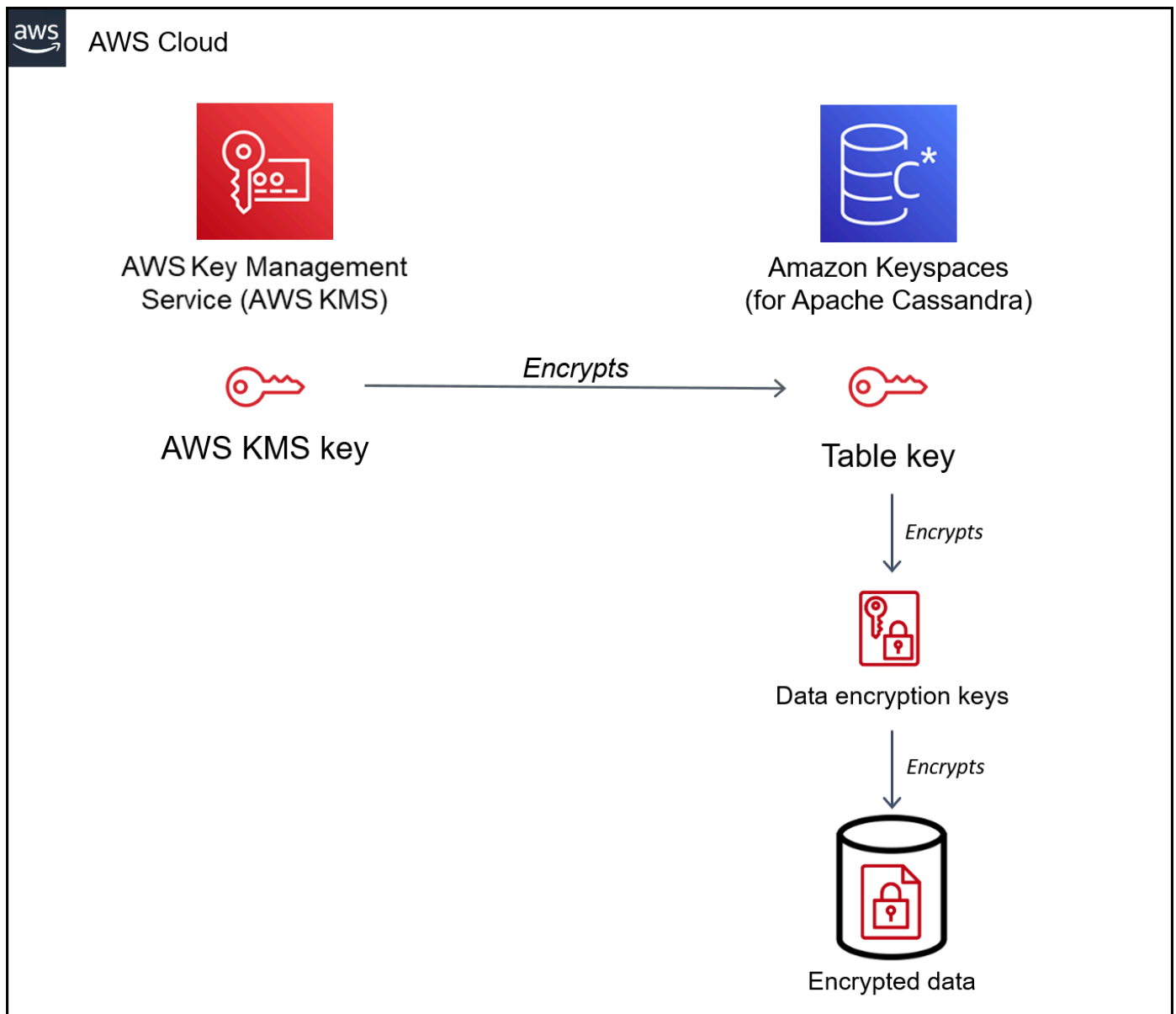
Sin embargo, puede cifrar sus tablas de Amazon Keyspaces utilizando una [clave administrada por el cliente](#) en su Cuenta de AWS. Puede seleccionar una clave KMS diferente para cada tabla de un espacio de claves. La clave KMS que seleccione para una tabla también se utiliza para cifrar todos sus metadatos 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.

La primera vez que configura la clave de KMS como una clave administrada por el cliente, AWS KMS genera una clave de datos. La clave AWS KMS de datos hace referencia a la clave de tabla de Amazon Keyspaces.

Cuando accedes a una tabla cifrada, Amazon Keyspaces envía una solicitud para usar la clave de KMS AWS KMS para descifrar la clave de la 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 usa y almacena la clave de tabla y las claves de cifrado de datos fuera de ella. 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 tener que realizar AWS KMS todas las operaciones de Amazon Keyspaces, Amazon Keyspaces almacena en caché las claves de tabla de texto sin formato de cada conexión en la memoria. Si Amazon Keyspaces recibe una solicitud para la clave de tabla almacenada en caché después de cinco minutos de inactividad, envía una nueva solicitud AWS KMS a para descifrar la clave de tabla. Esta llamada captura cualquier cambio realizado en las políticas de acceso de la clave KMS AWS Identity and Access Management (AWS KMS o IAM) desde la última solicitud para descifrar la clave de la 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

- [AWS claves propias](#)
- [Claves administradas por el cliente](#)
- [Notas de uso del cifrado en reposo](#)

AWS claves propias

Claves propiedad de AWS no están almacenados en su. Cuenta de AWS Forman parte de una colección de claves de KMS que AWS posee y administra para su uso en múltiples direcciones Cuentas de AWS. AWS los servicios que puede utilizar Claves propiedad de AWS para proteger sus datos.

No puede ver, administrar Claves propiedad de AWS, usar ni auditar su uso. Sin embargo, no es necesario que realice ninguna acción ni que cambie programas para proteger las claves que cifran sus datos.

No se te cobra una cuota mensual ni una cuota de uso por el uso de tu cuenta Claves propiedad de AWS, y estas no se tienen en cuenta para AWS KMS las cuotas de tu cuenta.

Claves administradas por el cliente

Las claves administradas por el cliente son claves Cuenta de AWS tuyas que tú creas, posees y administras. Usted tiene el control total sobre estas claves 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.
- Puedes usar AWS CloudTrail Amazon CloudWatch Logs para realizar un seguimiento de las solicitudes que Amazon Keyspaces envía AWS KMS en tu nombre. Para obtener más información, consulte [the section called “Paso 6: Configure la supervisión con AWS CloudTrail”](#).

Las claves administradas por el cliente [conllevar un cargo](#) por cada llamada a la API y se aplican AWS KMS cuotas a estas claves de 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.

- Server-side el cifrado en reposo está activado en todas las tablas de Amazon Keyspaces y no se puede desactivar. 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 realizan actualizaciones atómicas de varias filas cifran los datos que se utilizan temporalmente 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 Keyspaces.

Puede utilizar la consola o las sentencias CQL AWS KMS key para especificar 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 gestionada por el cliente 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: Configure la supervisión con AWS CloudTrail](#)

Requisitos previos: Crear una clave gestionada por el cliente AWS KMS y conceder permisos a Amazon Keyspaces

Para poder proteger una tabla de Amazon Keyspaces con una [clave gestionada por el cliente](#), primero debe crear la clave en AWS Key Management Service (AWS KMS) y, a continuación, autorizar a Amazon Keyspaces a utilizarla.

Paso 1: Cree una clave gestionada por el cliente mediante AWS KMS

Para crear una clave gestionada por el cliente que se utilice para proteger una tabla de Amazon Keyspaces, puede seguir los pasos que se indican en [Creación de claves KMS de cifrado simétrico](#) mediante la consola o la API. 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](#)
- [Ejemplo de política de claves de](#)
- [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](#))
- [kilómetros: DescribeKey](#)
- [km: CreateGrant](#)

Ejemplo de política de claves de

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 [kms: ViaService](#) condition permite los permisos solo cuando la solicitud proviene de Amazon Keyspaces en nombre de las entidades principales que figuran en la declaración de 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 requiere el permiso para ser independiente de cualquier elemento concreto. Región de AWS
- 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 un ejemplo de política de claves, sustituya los principios de ejemplo por los principios reales de su empresa. 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 que tiene 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 Cuenta de AWS el ID. 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 y abra la Consola de administración de AWS consola de Amazon Keyspaces 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 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 de KMS diferente (avanzada) y, en el campo de búsqueda, elija AWS KMS key o introduzca un nombre de recurso de Amazon (ARN).

 Note

Si no se puede acceder a la clave que ha seleccionado o le faltan los permisos necesarios, consulte [Solución de problemas de acceso a claves](#) en la Guía para AWS Key Management Service desarrolladores.

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-1:555555555555: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 ha configurado correctamente, consulte [Solución de problemas de acceso a claves](#) en la Guía para AWS Key Management Service desarrolladores.

Paso 4: Actualizar la clave de cifrado de una tabla existente

También puede utilizar la consola Amazon Keyspaces o el CQL para cambiar las claves de cifrado de una tabla existente entre una clave de KMS gestionada por el cliente Clave propiedad de AWS y una clave de KMS gestionada por el cliente en cualquier momento.

Actualización de una tabla existente con la nueva clave administrada por el cliente (consola)

1. Inicie sesión en y abra la Consola de administración de AWS consola de Amazon Keyspaces en <https://console.aws.amazon.com/keyspaces/home>
2. En el panel de navegación, elija 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, selecciona la opción Elige una clave de KMS diferente (avanzada) y, en el campo de búsqueda, selecciona AWS KMS key o introduce un nombre de recurso de Amazon (ARN).

Note

Si la clave que ha seleccionado no es válida, consulte [Solución de problemas de acceso a claves](#) en la Guía para AWS Key Management Service desarrolladores.

Como alternativa, puede elegir una Clave propiedad de AWS para una tabla cifrada con una clave gestionada 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-1:555555555555: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 ha configurado correctamente, consulte [Solución de problemas de acceso a claves](#) en la Guía para AWS Key Management Service desarrolladores.

Para volver a cambiar la clave de cifrado por la opción de cifrado en reposo predeterminada Claves propiedad de AWS, puede utilizar la `ALTER TABLE` sentencia que 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. Al incluir un contexto de cifrado en una solicitud de cifrado de datos, vincula AWS KMS 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 AWS KMS criptográficas. 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 plano en los registros, como en los registros de Amazon Logs [AWS CloudTrail](#) [Amazon CloudWatch Logs](#).

En sus solicitudes 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: Configure la supervisión con AWS CloudTrail

Si utilizas una [clave gestionada por el cliente](#) para proteger tus tablas de Amazon Keyspaces, puedes utilizar AWS CloudTrail los registros para realizar un seguimiento de las solicitudes que Amazon Keyspaces envía en tu nombre. AWS KMS

Las solicitudes `GenerateDataKey`, `DescribeKey`, `Decrypt` y `CreateGrant` se tratan en esta sección. Además, Amazon Keyspaces utiliza una [RetireGrant](#) operación para eliminar una concesión al eliminar una tabla.

Note

Cuando trabaja con Amazon Keyspaces, algunas operaciones pueden generar CloudTrail eventos con un `invokedBy` campo de `dynamodb.amazonaws.com`. Esto es de esperar y se debe a que Amazon Keyspaces se integra con Amazon DynamoDB para proporcionar su servicio.

GenerateDataKey

Amazon Keyspaces crea una clave de tabla única para cifrar los datos en reposo. Envía una [GenerateDataKey](#) solicitud a la AWS KMS que se 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-1:555555555555: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-1:555555555555:key/11111111-1111-111-1111-111111111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123SAMPLE012",
  "sharedEventID": "84fbaaf0-9641-4e32-9147-57d2cb08792e"
}

```

DescribeKey

Amazon Keyspaces utiliza una [DescribeKey](#) operación para determinar si la clave de KMS que ha seleccionado existe en la cuenta y la 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-1:555555555555: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-1:555555555555: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 de descifrado AWS KMS a la que se especifica la clave de KMS de la tabla.

El evento que registra la operación Decrypt es similar al siguiente evento de ejemplo. El usuario principal que accede a la tabla es el Cuenta de AWS usuario principal. Los parámetros incluyen la clave de la tabla cifrada (como un bloque de texto cifrado) y el [contexto de cifrado](#) que identifica la tabla y la. Cuenta de AWS AWS KMS obtiene el ID de la clave gestionada 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-1:555555555555: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. La entidad 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-1:555555555555:key/11111111-1111-111-1111-111111111111:user/admin",
    "accountId": "arn:aws:kms:eu-west-1:555555555555: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-1:555555555555: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) protege las conexiones de las aplicaciones locales a Amazon Keyspaces y entre Amazon Keyspaces AWS y otros recursos dentro de las mismas. Región de AWS

Tráfico entre el servicio y las aplicaciones y clientes locales

Dispone de dos opciones de conectividad entre su red privada y: AWS

- Una AWS Site-to-Site VPN conexión. Para obtener más información, consulte [¿Qué es AWS Site-to-Site VPN?](#) en la Guía del usuario de AWS Site-to-Site VPN .
- Una Direct Connect conexión. Para obtener más información, consulte [¿Qué es Direct Connect?](#) en la Guía del usuario de Direct Connect .

Como servicio gestionado, Amazon Keyspaces (para Apache Cassandra) está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad [AWS en la nube](#). Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte [Protección de infraestructuras en un marco](#) de buena AWS arquitectura basado en el pilar de la seguridad.

Utiliza las llamadas a la API AWS publicadas 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.
- Paquetes de cifrado con perfecto secreto directo (PFS), como el DHE (Ephemeral) o el ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

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 o la API. AWS CLI AWS Para obtener más información, consulte [Uso de IAM con Amazon Keyspaces](#).

El segundo método utiliza un complemento de autenticación para el controlador DataStax Java 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 “Cree credenciales de IAM para la autenticación AWS”](#).

Tráfico entre AWS recursos de 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 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 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 finales de la interfaz VPC no requieren una puerta de enlace a Internet, un dispositivo NAT, una conexión VPN o una conexión. 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 del acceso con políticas](#)
- [Funcionamiento de Amazon Keyspaces con IAM](#)
- [Ejemplos de políticas basadas en identidades de Amazon Keyspaces](#)
- [AWS políticas gestionadas 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 según la función que desempeñes:

- Usuario del servicio: solicite permisos al administrador si no puede acceder a las características (consulte [Solución de problemas de identidad y acceso a Amazon Keyspaces](#)).
- Administrador del servicio: determine el acceso de los usuarios y envíe las solicitudes de permiso (consulte [Funcionamiento de Amazon Keyspaces con IAM](#)).
- Administrador de IAM: escribe las políticas para administrar el acceso (consulte [Ejemplos de políticas basadas en identidades de Amazon Keyspaces](#)).

Autenticación con identidades

La autenticación es la forma en que inicias sesión AWS con tus credenciales de identidad. Debe autenticarse como usuario de Usuario raíz de la cuenta de AWS IAM o asumir una función de IAM.

Puede iniciar sesión como una identidad federada con las credenciales de una fuente de identidad, como AWS IAM Identity Center (IAM Identity Center), la autenticación de inicio de sesión único o las credenciales. Google/Facebook Para obtener más información sobre el inicio de sesión, consulte [Cómo iniciar sesión en la Cuenta de AWS](#) en la Guía del usuario de AWS Sign-In .

Para el acceso programático, AWS proporciona un SDK y una CLI para firmar criptográficamente las solicitudes. Para obtener más información, consulte [AWS Signature Version 4 para solicitudes de API](#) en la Guía del usuario de IAM.

Cuenta de AWS usuario root

Al crear un Cuenta de AWS, se comienza con una identidad de inicio de sesión denominada usuario Cuenta de AWS raíz que tiene acceso completo a todos Servicios de AWS los recursos. Se recomienda encarecidamente que no utilice el usuario raíz para las tareas diarias. Para ver la lista completa de las tareas que requieren credenciales de 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 con permisos específicos para una sola persona o aplicación. Recomendamos el uso de credenciales temporales en lugar de usuarios de IAM con credenciales de larga duración. Para obtener más información, consulte [Exigir a los usuarios humanos que utilicen la federación con un proveedor de identidad para acceder AWS mediante credenciales temporales](#) en la Guía del usuario de IAM.

Un [grupo de IAM](#) especifica un conjunto de usuarios de IAM y facilita la administración de los permisos para grupos grandes de usuarios. Para obtener más información, consulte [Casos de uso para usuarios de IAM](#) en la Guía del usuario de IAM.

Roles de IAM

Un [Rol de IAM](#) es una identidad con permisos específicos que proporciona credenciales temporales. Puede asumir un rol [cambiando de un rol de usuario a uno de IAM \(consola\)](#) o llamando a una AWS CLI operación de AWS API. Para obtener más información, consulte [Métodos para asumir un rol](#) en la Guía del usuario de IAM.

Los roles de IAM son útiles para el acceso de usuario federado, los permisos de usuario de IAM temporales, el acceso entre cuentas, el acceso entre servicios y las aplicaciones que se ejecutan en Amazon EC2. Para obtener más información, consulte [Acceso a recursos entre cuentas en IAM](#) en la Guía del usuario de IAM.

Administración del acceso con políticas

AWS Para controlar el acceso, puede crear políticas y adjuntarlas a AWS identidades o recursos. Una política define los permisos cuando están asociados a una identidad o un recurso. AWS evalúa

estas políticas cuando un director hace una solicitud. La mayoría de las políticas se almacenan AWS como documentos JSON. Para obtener más información sobre los documentos de políticas de JSON, consulte [Información general de políticas de JSON](#) en la Guía del usuario de IAM.

Mediante las políticas, los administradores especifican quién tiene acceso a qué, definiendo qué entidad principal puede realizar acciones sobre qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Un administrador de IAM crea políticas de IAM y las agrega a roles, que los usuarios pueden asumir posteriormente. Las políticas de IAM definen permisos independientemente del método que se utilice para realizar la operación.

Identity-based políticas

Identity-based las políticas son documentos de política de permisos de JSON que se adjuntan a una identidad (usuario, grupo o rol). Estas políticas controlan qué acciones pueden realizar las identidades, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en la identidad, consulte [Definición de permisos de IAM personalizados con políticas administradas por el cliente](#) en la Guía del usuario de IAM.

Identity-based las políticas pueden ser políticas integradas (integradas directamente en una sola identidad) o políticas administradas (políticas independientes asociadas a varias identidades). Para obtener información sobre cómo elegir entre políticas administradas e insertadas, consulte [Selección entre políticas administradas y políticas insertadas](#) en la Guía del usuario de IAM.

Resource-based políticas

Resource-based las políticas son documentos de políticas de JSON que se adjuntan a un recurso. Los ejemplos incluyen 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. Debe [especificar una entidad principal](#) en una política basada en recursos.

Resource-based las políticas son políticas en línea 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 que pueden establecer los permisos máximos que conceden los tipos de políticas más comunes:

- Límites de permisos: establecen los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM. Para obtener más información, consulte [Límites de permisos para las entidades de IAM](#) en la Guía del usuario de IAM.
- Políticas de control de servicios (SCP): especifican los permisos máximos para una organización o unidad organizativa en AWS Organizations. Para obtener más información, consulte [Políticas de control de servicios](#) en la Guía del usuario de AWS Organizations .
- Políticas de control de recursos (RCP): definen los permisos máximos disponibles para los recursos de las cuentas. Para obtener más información, consulte [Políticas de control de recursos \(RCP\)](#) en la Guía del usuario de AWS Organizations .
- Políticas de sesión: políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal para un rol o un usuario federado. Para obtener 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 se AWS determina 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. 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. Como práctica recomendada, especifique un recurso utilizando el [Nombre de recurso de Amazon \(ARN\)](#). En el caso de las acciones que no admiten permisos por recurso, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

En Amazon Keyspaces, los espacios clave, las tablas y las transmisiones se pueden utilizar como `Resource` elemento de los permisos de IAM.

Note

Para acceder a los espacios clave y las tablas de los usuarios en Amazon Keyspaces, su política de IAM debe `cassandra:Select` incluir permisos en las tablas del sistema:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
```

Esto se aplica a los siguientes escenarios:

- AWS Acceso a la consola de administración
- Las operaciones de recursos del SDK `GetKeyspace`, por ejemplo `GetTable`, `ListKeyspaces`, y `ListTables`
- Conexiones estándar del controlador del cliente Apache Cassandra, ya que los controladores leen automáticamente las tablas del sistema durante la inicialización de la conexión

Las tablas del sistema son de solo lectura y no se pueden modificar.

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}
```

El recurso de transmisión de Amazon Keyspaces tiene el siguiente ARN:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

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-1:111122223333:/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-1:111122223333:/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": ""
```

Por ejemplo, para conceder SELECT permisos a una entidad principal de IAM para mytable entrarmykeyspace, la entidad principal debe tener permisos para leer ambos, y. mytable keyspace/system* 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` especifica cuándo se ejecutan las instrucciones en función de criterios definidos. Puede crear expresiones condicionales que utilizan [operadores de condición](#), tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud. 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 “Trabajo 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 [AssumeRole](#) o [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 en los que se implementa el complemento de autenticación para acceder a Amazon Keyspaces mediante programación, consulte [the section called “Utilización de un controlador de cliente de Cassandra”](#).

Service-linked roles

[Service-linked los roles](#) permiten a los AWS servicios acceder a los recursos de otros servicios para completar una acción en su nombre. Service-linked los roles aparecen en su 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 “Cómo utilizar 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 AWS la AWS CLI API. 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

Identity-based las políticas determinan si alguien puede crear, acceder o eliminar los recursos de Amazon Keyspaces de 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 tarea](#) en la Guía de 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 CloudFormation. Para obtener más información, consulte [Elementos de la política de JSON de IAM: Condición](#) en la Guía del usuario de IAM.
- Utiliza 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 [Validación de políticas con el Analizador de acceso de IAM](#) en la Guía del 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 exigir la MFA cuando se invoquen las operaciones de la API, añada condiciones de MFA a sus políticas. Para más información, consulte [Acceso seguro a la API con MFA](#) en la Guía del usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte [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 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 “AWS políticas gestionadas”](#).

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

Note

Para acceder a los espacios clave y las tablas de los usuarios en Amazon Keyspaces, su política de IAM debe `cassandra:Select` incluir permisos en las tablas del sistema:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
```

Esto se aplica a los siguientes escenarios:

- AWS Acceso a la consola de administración
- Las operaciones de recursos del SDK `GetKeyspace`, por ejemplo `getTable`, `ListKeyspaces`, y `ListTables`
- Conexiones estándar del controlador del cliente Apache Cassandra, ya que los controladores leen automáticamente las tablas del sistema durante la inicialización de la conexión

Las tablas del sistema son de solo lectura y no se pueden modificar.

A continuación se muestra un ejemplo de política que concede acceso de solo lectura (SELECT) a las tablas del sistema de Amazon Keyspaces. En todos los ejemplos, sustituya la región y el ID de cuenta en el nombre de recurso de Amazon (ARN) por los suyos.


```
{
  "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 acceso de solo lectura a la tabla de usuario mytable en el espacio de claves 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 el read/write acceso a una tabla de usuarios y el acceso de lectura a las tablas del sistema.

```
{
  "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*"
      ]
    }
  ]
}

```

El siguiente ejemplo de política asigna el acceso de lectura a las tablas del sistema, pero restringe el acceso SELECT (lectura) y MODIFY (escritura) a la tabla de usuarios. mytable

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",

```

```
    "Action": [
      "cassandra:Select"
    ],
    "Resource": [
      "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
    ]
  },
  {
    "Effect": "Deny",
    "Action": [
      "cassandra:Select",
      "cassandra:Modify"
    ],
    "Resource": [
      "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/mytable"
    ]
  }
]
```

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}"
        }
    }
}
]
}

```

AWS políticas gestionadas para Amazon Keyspaces

Una política AWS administrada es una política independiente creada y administrada por AWS. Las políticas administradas están diseñadas para proporcionar permisos para muchos casos de uso comunes, de modo que pueda empezar a asignar permisos a usuarios, grupos y funciones.

Ten en cuenta que es posible que las políticas AWS administradas no otorguen permisos con privilegios mínimos para tus casos de uso específicos, ya que están disponibles para que los usen todos los AWS clientes. Se recomienda definir [políticas administradas por el cliente](#) específicas para sus casos de uso a fin de reducir aún más los permisos.

No puedes cambiar los permisos definidos en AWS las políticas administradas. Si AWS actualiza los permisos definidos en una política AWS administrada, la actualización afecta a todas las identidades principales (usuarios, grupos y roles) a las que está asociada la política. AWS es más probable que actualice una política AWS administrada cuando Servicio de AWS se lance una nueva o cuando estén disponibles nuevas operaciones de API para los servicios existentes.

Para obtener más información, consulte [Políticas administradas por AWS](#) en la Guía del usuario de IAM.

AWS política gestionada: AmazonKeyspacesReadOnlyAccess_v2

Puede asociar 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 de los permisos

Esta política incluye los siguientes permisos.

- **Amazon Keyspaces:** proporciona acceso de solo lectura a Amazon Keyspaces.
- **Amazon Keyspaces CDC streams—** Permite a los directores ver las transmisiones de los CDC de 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 los directores ver los datos métricos y las alarmas configuradas en CloudWatch. Esto es necesario para que los usuarios puedan ver el tamaño de la tabla facturable y CloudWatch las alarmas que se han configurado para una tabla.
- **AWS KMS—** Permite a los directores ver las claves configuradas en AWS KMS. Esto es necesario para que los usuarios puedan ver AWS KMS las claves que crean y administran en su cuenta y confirmar que la clave asignada a Amazon Keyspaces es una clave de cifrado simétrica que está 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 el JSON formato de la política, consulte `AmazonKeyspacesReadOnlyAccess\_v2`.](#)

AWS política gestionada: `AmazonKeyspacesReadOnlyAccess`

Puede asociar la política `AmazonKeyspacesReadOnlyAccess` a las identidades de IAM.

Esta política concede acceso de solo lectura a Amazon Keyspaces.

Detalles de los permisos

Esta política incluye los siguientes permisos.

- `Amazon Keyspaces`: proporciona acceso de solo lectura a Amazon Keyspaces.
- `Amazon Keyspaces CDC streams`— Permite a los directores ver las transmisiones de los CDC de 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 los directores ver los datos métricos y las alarmas configuradas en CloudWatch. Esto es necesario para que los usuarios puedan ver el tamaño de la tabla facturable y CloudWatch las alarmas que se han configurado para una tabla.
- `AWS KMS`— Permite a los directores ver las claves configuradas en AWS KMS. Esto es necesario para que los usuarios puedan ver AWS KMS las claves que crean y administran en su cuenta y confirmar que la clave asignada a Amazon Keyspaces es una clave de cifrado simétrica que está habilitada.

Para revisar el JSON formato de la política, consulte. [AmazonKeyspacesReadOnlyAccess](#)

AWS política gestionada: `AmazonKeyspacesFullAccess`

Puede asociar 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 de 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 los directores ver el tamaño de la tabla facturable, así como crear, ver y eliminar CloudWatch alarmas para las políticas de escalado automático de Amazon Keyspaces. Esto es necesario para que los administradores puedan ver el tamaño de la tabla facturable y crear un panel. CloudWatch
- **IAM:** permite a Amazon Keyspaces crear automáticamente roles vinculados a servicios con IAM cuando están activadas las siguientes características:
 - **Amazon Keyspaces CDC streams—** Cuando un administrador habilita una transmisión para una tabla, Amazon Keyspaces crea el rol vinculado al servicio [AWSServiceRoleForAmazonKeyspacesCDC](#) para publicar CloudWatch las métricas en su cuenta en su nombre.
 - **Application Auto Scaling—** Cuando un administrador habilita Application Auto Scaling para una tabla, Amazon Keyspaces crea el rol vinculado al servicio [AWSServiceRoleForApplicationAutoScaling_CassandraTable](#) para realizar acciones de escalado automáticas en su nombre.
 - **Amazon Keyspaces multi-Region replication—** Cuando un administrador crea un nuevo espacio de claves multirregión o añade uno nuevo Región de AWS a un espacio de claves de una sola región existente, Amazon Keyspaces crea la [AWSServiceRoleForAmazonKeyspacesReplication](#) función vinculada al servicio para realizar la replicación de tablas, datos y metadatos en las regiones seleccionadas en su nombre.
- **AWS KMS:** permite a las entidades principales ver las claves configuradas en AWS KMS. Esto es necesario para que los usuarios puedan ver AWS KMS las claves que crean y administran en su cuenta y confirmar que la clave asignada a Amazon Keyspaces es una clave de cifrado simétrica que está 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 el JSON formato de la política, consulte. [AmazonKeyspacesFullAccess](#)

AWS política gestionada: KeyspacesCDCServiceRolePolicy

No puede asociar KeyspacesCDCServiceRolePolicy a sus entidades IAM. Esta política se adjunta a un rol vinculado a un servicio que permite a Amazon Keyspaces realizar acciones en su nombre. Para obtener más información, consulte [the section called “Transmisiones de los CDC”](#).

Esta política otorga los permisos necesarios al rol vinculado al servicio para publicar en su nombre AWSServiceRoleForAmazonKeyspacesCDC los datos de las métricas de transmisión de los CDC de Amazon Keyspaces. CloudWatch

Detalles de los permisos

Esta política incluye los siguientes permisos.

- CloudWatch— Permite que la función vinculada al servicio publique datos de métricas de [AWSServiceRoleForAmazonKeyspacesCDC](#) las transmisiones de los CDC de Amazon Keyspaces en su cuenta en su nombre. "cloudwatch:namespace": "AWS/Cassandra" CloudWatch

Para revisar el formato de la política, consulte. JSON [KeyspacesCDCServiceRolePolicy](#)

Amazon Keyspaces actualiza las políticas gestionadas AWS

Consulta los detalles sobre las actualizaciones de las políticas AWS gestionadas para Amazon Keyspaces desde que este servicio comenzó a realizar el seguimiento de estos cambios. Para obtener alertas automáticas sobre cambios en esta página, suscríbese a la fuente RSS en la página de [Historial de revisión](#).

Cambio	Descripción	Fecha
KeyspacesCDCServiceRolePolicy : política nueva	Amazon Keyspaces ha añadido una nueva política gestionada Keyspaces <code>CDCServiceRolePolicy</code> que concede los permisos necesarios al rol vinculado al servicio para publicar los datos de las métricas de streaming de los CDC de <code>AWSServiceRoleForAmazonKeyspacesCDC</code> Amazon Keyspaces en su nombre. CloudWatch Para obtener más información, consulte the section called “Transmisiones de los CDC” .	2 de julio de 2025
AmazonKeyspacesReadOnlyAccess_v2 : actualización de una política existente	Amazon Keyspaces agregó nuevos permisos para permitir a los directores de IAM ver las transmisiones de Amazon Keyspaces CDC. Para obtener más información, consulte the section called “Vea las transmisiones de los CDC” .	2 de julio de 2025
AmazonKeyspacesReadOnlyAccess : actualización de una política actual	Amazon Keyspaces agregó nuevos permisos para permitir a los directores de IAM ver las transmisiones de Amazon Keyspaces CDC. Para obtener más información, consulte the section called “Vea las transmisiones de los CDC” .	2 de julio de 2025

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess : actualización de una política actual	Amazon Keyspaces creó la política Keyspaces CDCServiceRolePolicy gestionada para el rol vinculado al servicio con el AWSServiceRoleForAmazonKeyspacesCDC fin de añadir los permisos que se requieren cuando un administrador habilita una transmisión para una tabla. Amazon Keyspaces utiliza la función vinculada al servicio AWSServiceRoleForAmazonKeyspacesCDC para publicar CloudWatch las métricas en su cuenta en su nombre. Para obtener más información, consulte the section called “Transmisiones de los CDC”.	2 de julio de 2025

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess : actualización de una política actual	Amazon Keyspaces actualizó el <code>KeyspacesReplicationServiceRolePolicy</code> y rol vinculado al servicio AWSServiceRoleForAmazonKeyspacesReplication para añadir los permisos necesarios cuando un administrador añade un espacio de claves nuevo Región de AWS a un espacio de claves único o multirregional. Amazon Keyspaces utiliza la función vinculada al servicio <code>AWSServiceRoleForAmazonKeyspacesReplication</code> para replicar las tablas, su configuración y los datos en su nombre. Para obtener más información, consulte the section called “Multi-Region Replicación”.	19 de noviembre de 2024

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess : actualización de una política actual	Amazon Keyspaces ha añadido nuevos permisos para permitir a Amazon Keyspaces crear un rol vinculado a un servicio cuando un administrador añade una nueva región a un espacio de claves único o multirregional. 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 “Multi-Region Replicación”.	3 de octubre de 2023

Cambio	Descripción	Fecha
AmazonKeyspacesReadOnlyAccess_v2: nueva política	Amazon Keyspaces creó una nueva política para añadir permisos de solo lectura a los clientes que se conecten a Amazon Keyspaces a través de puntos de enlace de la interfaz de la VPC para acceder a la instancia de Amazon EC2 y buscar información de la red. Amazon Keyspaces almacena los puntos de conexión de VPC de interfaz disponibles en la tabla de <code>system.peers</code> 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 actual	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 la función vinculada al servicio <code>AWSServiceRoleForAmazonKeyspacesReplication</code> para realizar tareas de replicación de datos en su nombre. Para obtener más información, consulte the section called “Multi-Region Replicación”.	5 de junio de 2023
AmazonKeyspacesReadOnlyAccess : actualización de una política actual	Amazon Keyspaces agregó nuevos permisos para permitir a los usuarios ver el tamaño facturable de una tabla utilizando CloudWatch. Amazon Keyspaces se integra con Amazon CloudWatch para que puedas controlar el tamaño de la mesa facturable. Para obtener más información, consulte the section called “Métricas de Amazon Keyspaces”.	7 de julio de 2022

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess : actualización de una política actual	Amazon Keyspaces agregó nuevos permisos para permitir a los usuarios ver el tamaño facturable de una tabla utilizando CloudWatch. Amazon Keyspaces se integra con Amazon CloudWatch para que puedas controlar el tamaño de la mesa facturable. Para obtener más información, consulte the section called “Métricas de Amazon Keyspaces”.	7 de julio de 2022
AmazonKeyspacesReadOnlyAccess : actualización de una política actual	Amazon Keyspaces agregó nuevos permisos para permitir a los usuarios ver AWS KMS las claves que se han configurado para el cifrado de Amazon Keyspaces en reposo. El cifrado en reposo de Amazon Keyspaces se integra AWS KMS para proteger y administrar las claves de cifrado utilizadas para cifrar los datos en reposo. Para ver la AWS KMS clave configurada para Amazon Keyspaces, se han añadido permisos de solo lectura.	1 de junio de 2021

Cambio	Descripción	Fecha
AmazonKeyspacesFullAccess: actualización de una política actual	Amazon Keyspaces agregó nuevos permisos para permitir a los usuarios ver AWS KMS las claves que se han configurado para el cifrado de Amazon Keyspaces en reposo. El cifrado en reposo de Amazon Keyspaces se integra AWS KMS para proteger y administrar las claves de cifrado utilizadas para cifrar los datos en reposo. Para ver la AWS KMS clave configurada 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 comenzó a realizar un seguimiento de los cambios en sus políticas AWS gestionadas.	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 con la recuperación en un momento dado \(PITR\) de Amazon Keyspaces](#)

- [No estoy autorizado a realizar un 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 Consola de administración de AWS te indica que no estás autorizado a realizar una acción, debes ponerte en contacto con tu administrador para que te ayude. El administrador es la persona que le facilitó el nombre de usuario y la contraseña.

El siguiente ejemplo de error se produce cuando el usuario de mateojackson IAM intenta utilizar la consola para ver los detalles de una tabla *table*, pero no tiene `cassandra:Select` permisos para acceder a ella.

```
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 con la recuperación en un momento dado (PITR) de Amazon Keyspaces

Si al intentar restaurar una tabla de Amazon Keyspaces con la recuperación en un momento dado (PITR) ve que el proceso de restauración comienza pero no finaliza 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 “Configuración de los permisos de IAM para la restauración”](#).

No estoy autorizado a realizar un 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.

Algunos 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 la función 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 AWS administrador. 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 otras personas accedan a Amazon Keyspaces, debe conceder permiso a las personas o aplicaciones que lo necesiten. Si usa AWS IAM Identity Center para administrar las

personas y las aplicaciones, debe asignar conjuntos de permisos a los usuarios o grupos para definir su nivel de acceso. Los conjuntos de permisos crean políticas de IAM y las asignan a los roles de IAM asociados a la persona o aplicación de forma automática. Para obtener más información, consulte la sección [Conjuntos de permisos](#) en la Guía del usuario de AWS IAM Identity Center .

Si no utiliza IAM Identity Center, debe crear entidades de IAM (usuarios o roles) para las personas o aplicaciones que necesitan acceso. A continuación, debe asociar una política a la entidad que les conceda los permisos correctos en Amazon Keyspaces. Una vez concedidos los permisos, proporcione las credenciales al usuario o al desarrollador de la aplicación. Utilizarán esas credenciales para acceder a AWS. Para obtener más información sobre la creación de usuarios, grupos, políticas y permisos de IAM, consulte [Identidades de IAM](#) y [Políticas y permisos en 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

Se puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Se 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 obtener 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 una federación de identidades, consulte [Proporcionar acceso a usuarios autenticados externamente \(identidad federada\)](#) en la Guía del usuario de IAM.
- Para conocer sobre la diferencia entre las políticas basadas en roles y en recursos para el acceso entre cuentas, consulte [Acceso a recursos entre cuentas en IAM](#) en la Guía del usuario de IAM.

Uso de roles vinculados a servicios para Amazon Keyspaces

[Amazon Keyspaces \(para Apache Cassandra\) usa roles vinculados a servicios AWS Identity and Access Management \(IAM\)](#). Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a Amazon Keyspaces. Service-linked Los roles están predefinidos por Amazon Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros AWS servicios en su nombre.

Para obtener información sobre otros servicios que admiten funciones vinculadas a servicios, consulte los [AWS servicios que funcionan con IAM](#) y busque los servicios que tienen la palabra Sí en la columna de funciones. Service-linked Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Temas

- [Uso de roles para el escalado automático de aplicaciones de Amazon Keyspaces](#)
- [Uso de roles para la replicación de Amazon Keyspaces Multi-Region](#)
- [Uso de roles para las transmisiones de los CDC de Amazon Keyspaces](#)

Uso de roles para el escalado automático de aplicaciones de Amazon Keyspaces

[Amazon Keyspaces \(para Apache Cassandra\) usa roles vinculados a servicios AWS Identity and Access Management \(IAM\)](#). Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a Amazon Keyspaces. Service-linked Los roles están predefinidos por Amazon Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros AWS servicios 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 asociar 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.

Service-linked permisos de rol para Amazon Keyspaces

Amazon Keyspaces utiliza el rol vinculado al servicio denominado `AWSServiceRoleForApplicationAutoScaling_CassandraTable` para 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 que los siguientes servicios asuman el rol:

- `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 CQL Consola de administración de AWS, la o la AWS API AWS 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. Al habilitar el escalado automático de Amazon Keyspaces para una tabla, Application Auto Scaling vuelve a crear el rol vinculado a servicios por 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 Apareció [un nuevo rol](#) en mi. Cuenta de AWS

Edición de un rol vinculado a servicios para Amazon Keyspaces

Amazon Keyspaces no le permite editar el rol `AWSServiceRoleForApplicationAutoScaling_CassandraTable` vinculado al `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 para Regiones de AWS poder eliminar manualmente el rol vinculado al servicio. Para deshabilitar el escalado automático en las tablas de Amazon Keyspaces, consulte [the section called “Desactivación del escalado automático de Amazon Keyspaces para una tabla”](#).

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.

Para eliminar manualmente el rol vinculado a servicios mediante IAM

Utilice la consola de IAM AWS CLI, la o la AWS API para eliminar la función `AWSServiceRoleForApplicationAutoScaling_CassandraTable` vinculada al servicio. Para obtener más información, consulte [Eliminar un Service-Linked rol](#) 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 replicación de Amazon Keyspaces Multi-Region

[Amazon Keyspaces \(para Apache Cassandra\) usa roles vinculados a servicios AWS Identity and Access Management \(IAM\)](#). Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a Amazon Keyspaces. Service-linked Los roles están predefinidos por Amazon Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros AWS servicios 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 asociar 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.

Service-linked permisos de rol para Amazon Keyspaces

Amazon Keyspaces utiliza el rol vinculado al servicio denominado para permitir que `AWSServiceRoleForAmazonKeyspacesReplication` Amazon Keyspaces añada nuevos elementos Regiones de AWS a un espacio de claves en su nombre y replique las tablas y todos sus datos y configuraciones en la nueva región. El rol también permite a Amazon Keyspaces replicar escrituras en tablas de todas las regiones en su nombre.

El rol `AWSServiceRoleForAmazonKeyspacesReplication` vinculado al servicio confía en los siguientes servicios para asumir 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`
- Acción: `cassandra:AlterMultiRegionResource`
- Acción: `application-autoscaling:RegisterScalableTarget` — Amazon Keyspaces utiliza los permisos de escalado automático de la aplicación cuando se añade una réplica a una sola tabla de regiones en el modo aprovisionado con el escalado automático activado.
- Acción: `application-autoscaling:DeregisterScalableTarget`
- Acción: `application-autoscaling:DescribeScalableTargets`
- Acción: `application-autoscaling:PutScalingPolicy`
- Acción: `application-autoscaling:DescribeScalingPolicies`
- Acción: `cassandra:Alter`
- Acción: `cloudwatch>DeleteAlarms`
- Acción: `cloudwatch:DescribeAlarms`
- Acción: `cloudwatch:PutMetricAlarm`

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.

Los permisos para crear el rol vinculado al servicio se incluyen en la política gestionada.

`AWSServiceRoleForAmazonKeyspacesReplication` `AmazonKeyspacesFullAccess` Para obtener más información, consulte [the section called “AmazonKeyspacesFullAccess”](#).

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 [los permisos de los Service-linked roles](#) 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 multirregional en la Consola de administración de AWS, la o la AWS API AWS CLI, Amazon Keyspaces 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. 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. Así no tendrá una entidad no utilizada que no se supervise ni mantenga de forma activa. Sin embargo, primero debe eliminar todos los espacios de claves multirregionales de la cuenta para Regiones de AWS poder eliminar manualmente el rol vinculado al servicio.

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 y abra la Consola de administración de AWS 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 de [eliminación del espacio de claves](#) de la CLI. AWS
- El [DeleteKeyspace](#) funcionamiento de la API de Amazon Keyspaces.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM AWS CLI, la o la AWS API para eliminar el rol vinculado al `AWSServiceRoleForAmazonKeyspacesReplication` servicio. 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 usar 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

Nombre de la región	Identidad de la región	Admitida en Amazon Keyspaces
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
África (Ciudad del Cabo)	af-south-1	Sí
América del Sur (São Paulo)	sa-east-1	Sí
AWS GovCloud (US-East)	us-gov-east-1	No
AWS GovCloud (US-West)	us-gov-west-1	No

Uso de roles para las transmisiones de los CDC de Amazon Keyspaces

[Amazon Keyspaces \(para Apache Cassandra\) usa roles vinculados a servicios AWS Identity and Access Management \(IAM\)](#). Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a Amazon Keyspaces. Service-linked Los roles están predefinidos por Amazon Keyspaces e incluyen todos los permisos que el servicio requiere para llamar a otros AWS servicios 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 asociar a ninguna otra entidad de IAM.

No puede eliminar el rol vinculado al servicio.

Service-linked permisos de rol para Amazon Keyspaces

Amazon Keyspaces utiliza el rol vinculado al servicio denominado para permitir que las transmisiones de `AWSServiceRoleForAmazonKeyspacesCDC` Amazon Keyspaces CDC publiquen CloudWatch métricas en su cuenta en su nombre.

El rol `AWSServiceRoleForAmazonKeyspacesCDC` vinculado al servicio confía en que el siguiente servicio asuma el rol:

- `cassandra-streams.amazonaws.com`

La política de permisos de roles denominada [KeyspacesCDCServiceRolePolicy](#) permite a Amazon Keyspaces realizar la siguiente acción en los recursos del CloudWatch espacio de nombres: `AWS/Cassandra`

- Acción: `cloudwatch:PutMetricData` en *

`AWSServiceRoleForAmazonKeyspacesCDC` Proporciona los permisos: Acción:
`cloudwatch:PutMetricData` en todos los recursos que cumplan la siguiente condición:
`"cloudwatch:namespace": "AWS/Cassandra"`

Para obtener más información sobre `KeyspacesCDCServiceRolePolicy`, consulte [the section called "KeyspacesCDCServiceRolePolicy"](#).

Para habilitar las transmisiones de los CDC para una tabla, lo que crea automáticamente la función vinculada al servicio `AWSServiceRoleForAmazonKeyspacesCDC`, el director de IAM necesita los siguientes permisos.

```
{
  "Sid": "KeyspacesCDCServiceLinkedRole",
  "Effect": "Allow",
  "Action": "iam:CreateServiceLinkedRole",
  "Resource": "arn:aws:iam::*:role/aws-service-role/cassandra-streams.amazonaws.com/AWSServiceRoleForAmazonKeyspacesCDC",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "cassandra-streams.amazonaws.com"
    }
  }
}
```

Los permisos para crear el rol vinculado al servicio `AWSServiceRoleForAmazonKeyspacesCDC` se incluyen en la política administrada `AmazonKeyspacesFullAccess`. Para obtener más información, consulte [the section called “AmazonKeyspacesFullAccess”](#).

Creación de un rol vinculado a servicios para Amazon Keyspaces

No necesitas crear manualmente un rol vinculado a un servicio para las transmisiones de Amazon Keyspaces CDC. Cuando habilita las transmisiones CDC de Amazon Keyspaces en una tabla con el CQL Consola de administración de AWS, la o la API AWS CLI AWS , Amazon Keyspaces crea el rol vinculado al servicio para 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 habilitar las transmisiones de Amazon Keyspaces CDC para una tabla, Amazon Keyspaces vuelve a crear el rol vinculado al servicio para usted.

Edición de un rol vinculado a servicios para Amazon Keyspaces

Amazon Keyspaces no le permite editar el rol vinculado al `AWSServiceRoleForAmazonKeyspacesCDC` 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.

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 y Regiones de AWS](#).

Validación de conformidad para Amazon Keyspaces (para Apache Cassandra)

Third-party los auditores evalúan la seguridad y el cumplimiento de Amazon Keyspaces (para Apache Cassandra) como parte de varios AWS programas de cumplimiento. Entre ellos se incluyen:

- ISO/IEC 27001:2013, 27017:2015, 27018:2019 y 9001:2015. ISO/IEC 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 programa de cumplimiento Servicio de AWS está dentro del ámbito de aplicación de programas de cumplimiento específicos, consulte Alcance por programa de cumplimiento [Servicios de AWS en Alcance por programa de cumplimiento y elija el programa](#) Servicios de AWS de . Para obtener información general, consulte Programas de [AWS cumplimiento > Programas AWS](#) .

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte [Descarga de informes en AWS Artifact](#) .

Su responsabilidad de cumplimiento al Servicios de AWS utilizarlos viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. Para obtener más información sobre su responsabilidad de conformidad al utilizarlos Servicios de AWS, consulte [AWS la documentación de seguridad](#).

Resiliencia y recuperación de desastres en Amazon Keyspaces

La infraestructura AWS global se basa Regiones de AWS en zonas de disponibilidad. Regiones de AWS proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. 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 los datos automáticamente tres veces en varias zonas de AWS disponibilidad dentro de la misma Región de AWS para garantizar una mayor durabilidad y alta disponibilidad.

Para obtener más información sobre las zonas de disponibilidad Regiones de AWS y las zonas de disponibilidad, consulte la infraestructura [AWS global](#).

Además de la infraestructura AWS global, Amazon Keyspaces ofrece varias funciones para respaldar sus necesidades de respaldo y resiliencia de datos.

Replicación multirregional

Amazon Keyspaces ofrece replicación multirregional si necesita replicar sus datos o aplicaciones a grandes distancias geográficas. Puede replicar sus tablas de Amazon Keyspaces en las diferentes tablas que Regiones de AWS desee. Para obtener más información, consulte [the section called “Multi-Region replicación”](#).

Point-in-time recuperación (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 [Point-in-time la recuperación de Amazon Keyspaces](#).

Seguridad de la infraestructura en Amazon Keyspaces

Como servicio gestionado, Amazon Keyspaces (para Apache Cassandra) está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad [AWS en la nube](#). Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte [Protección de infraestructuras en un marco](#) de buena AWS arquitectura basado en el pilar de la seguridad.

Utiliza las llamadas a la API AWS publicadas 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.
- Paquetes de cifrado con perfecto secreto directo (PFS), como el DHE (Ephemeral) o el ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

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 o la API. AWS CLI AWS Para obtener más información, consulte [Uso de IAM con Amazon Keyspaces](#).

El segundo método utiliza un complemento de autenticación para el controlador DataStax Java 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 “Cree credenciales de IAM para la autenticación AWS”](#).

Puede llamar a estas operaciones de API desde cualquier ubicación de la red, pero Amazon Keyspaces admite políticas de acceso basadas en recursos, que pueden incluir restricciones basadas en la dirección IP de origen. También puede utilizar las políticas de Amazon Keyspaces para controlar el acceso desde puntos de enlace o VPC específicos de Amazon Virtual Private Cloud (Amazon VPC). De hecho, esto aísla el acceso a la red a un recurso de Amazon Keyspaces determinado únicamente de la VPC específica de la red. 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 enlace de la interfaz VPC funcionan con una AWS tecnología que permite la comunicación privada entre AWS servicios mediante una interfaz de red elástica con direcciones IP privadas en su Amazon VPC. AWS PrivateLink 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 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 punto de conexión de VPC y recuperación en un momento dado \(PITR\) de Amazon Keyspaces](#)
- [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 tutorial paso a paso para configurar una conexión a Amazon Keyspaces a través de un punto de conexión de VPC, consulte [the section called “Conexión con puntos de conexión 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 “Configuración del 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 `system.peers` tabla con los puntos finales de VPC de la interfaz disponibles mejora el equilibrio de carga y aumenta el rendimiento. read/write 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": "*"
    }
  ]
}
```

ℹ 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 de VPC: Read-only acceso: este ejemplo de política se puede adjuntar a un punto final 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 punto de conexión de VPC y recuperación en un momento dado (PITR) de Amazon Keyspaces

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”](#).

Uso de Amazon Keyspaces: transmisiones de CDC con puntos de enlace 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 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 enlace de VPC de interfaz para transmisiones CDC de Amazon Keyspaces](#)
- [Amazon Keyspaces CDC transmite puntos de enlace de VPC de interfaz](#)
- [Cree Amazon Keyspaces, interfaz de transmisión de CDC, punto final de VPC](#)
- [Actualizar un punto final de VPC de la interfaz de transmisiones CDC de Amazon Keyspaces](#)
- [Enumere las transmisiones mediante un punto final de VPC de la interfaz de transmisiones CDC de Amazon Keyspaces](#)
- [Cree una política para un punto final de VPC de la interfaz de transmisiones CDC de Amazon Keyspaces](#)

Uso de puntos de enlace de VPC de interfaz para transmisiones CDC de Amazon Keyspaces

Puede usar un punto de enlace de VPC de interfaz para que el tráfico entre las transmisiones de CDC de Amazon Keyspaces y sus recursos de Amazon VPC comience a fluir a través del punto de enlace de VPC de interfaz. Puede usar las políticas de puntos finales de VPC para restringir el acceso a sus transmisiones de los CDC.

Para obtener más información sobre las transmisiones de los CDC de Amazon Keyspaces, consulte [the section called “Trabajando con las transmisiones de los CDC”](#)

Amazon Keyspaces CDC transmite puntos de enlace de VPC de interfaz

Al crear un punto de enlace de interfaz, las transmisiones de CDC de Amazon Keyspaces generan dos tipos de nombres DNS específicos del punto de enlace para la transmisión: regional y zonal.

Regional

El nombre DNS regional incluye la siguiente información:

- un ID de punto final de Amazon VPC único
- un identificador de servicio
- el Región de AWS
- el `vpce.amazonaws.com` sufijo

Para un punto final de Amazon VPC con el ID `vpce-1a2b3c4d`, el nombre de DNS generado podría tener un aspecto similar al siguiente ejemplo: `vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com`

Zonal

El nombre DNS zonal incluye la [zona de disponibilidad](#) además de la información del nombre DNS regional. El nombre de DNS generado para el punto de enlace de Amazon VPC con el ID `vpce-1a2b3c4d` vería como en el siguiente ejemplo. Tenga en cuenta que Región de AWS ahora incluye la zona de disponibilidad: `vpce-1a2b3c4d-5e6f-us-east-1a.cassandra-streams.us-east-1.vpce.amazonaws.com`

Puede usar esta opción si su arquitectura aísla las zonas de disponibilidad. Por ejemplo, podría usarlo para la contención de fallos o para reducir los costos de transferencia de datos regionales.

Note

Para lograr una confiabilidad óptima, recomendamos implementar el servicio en un mínimo de tres zonas de disponibilidad.

Cree Amazon Keyspaces, interfaz de transmisión de CDC, punto final de VPC

Puede utilizar el AWS SDK AWS CLI o el mismo para acceder a las operaciones de la API CDC Streams de Amazon Keyspaces a través de los puntos de enlace de la interfaz CDC Streams de Amazon Keyspaces. Para obtener una lista completa de todas las operaciones de API disponibles, consulte [Amazon Keyspaces Streams API Reference](#).

Para obtener más información sobre cómo crear puntos de enlace de VPC, consulte [crear un punto de enlace de interfaz](#) en la Guía del usuario de Amazon VPC.

Para crear un punto final de VPC, puede usar la sintaxis del siguiente ejemplo.

```
aws ec2 create-vpc-endpoint \  
  --region us-east-1 \  
  --service-name api.aws.us-east-1.cassandra-streams \  
  --vpc-id client-vpc-id \  
  --subnet-ids client-subnet-id \  
  --vpc-endpoint-type Interface \  
  --security-group-ids client-sg-id
```

Actualizar un punto final de VPC de la interfaz de transmisiones CDC de Amazon Keyspaces

Para actualizar un punto final de VPC, puede usar la sintaxis del siguiente ejemplo.

```
aws ec2 modify-vpc-endpoint \  
  --region us-east-1 \  
  --vpc-endpoint-id client-vpc-id \  
  --policy-document policy-document \ #example optional parameter  
  --add-security-group-ids security-group-ids \ #example optional parameter
```

Enumere las transmisiones mediante un punto final de VPC de la interfaz de transmisiones CDC de Amazon Keyspaces

Para enumerar las transmisiones que utilizan un punto final de VPC, puede utilizar la sintaxis del siguiente ejemplo. Asegúrese de reemplazar la región y el nombre DNS del ID del punto final de la VPC por su propia información.

```
aws keyspacesstreams \
  --endpoint https://vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com \
  --region us-east-1 \
  list-streams
```

Cree una política para un punto final de VPC de la interfaz de transmisiones CDC de Amazon Keyspaces

Puede adjuntar una política de punto final a su punto de enlace de Amazon VPC que controle el acceso a las transmisiones de los CDC de Amazon Keyspaces. La política especifica la siguiente información:

- El director AWS Identity and Access Management (IAM) que puede realizar acciones
- Las acciones que se pueden realizar
- Los recursos en los que se pueden llevar a cabo las acciones

Para restringir el acceso a transmisiones de CDC específicas de Amazon Keyspaces y permitir solo AWS servicios específicos en tu acceso a Amazon VPC, puedes usar el siguiente ejemplo.

La siguiente política de transmisión otorga acceso a cualquier entidad principal de IAM para las acciones `cassandra:GetStream` y `cassandra:GetRecords` para la transmisión especificada `2025-02-20T11:22:33.444` asociada al recurso que `/keyspace/mykeyspace/table/mytable/` pertenece a la cuenta. `123456788901` Para usar esta política de puntos finales, asegúrate de reemplazar la región, el ID de cuenta y el recurso por la etiqueta de transmisión.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1216114807515",
  "Statement": [
    { "Sid": "Access-to-specific-stream-only",
```

```
{
  "Principal": "*",
  "Action": [
    "cassandra:GetStream",
    "cassandra:GetRecords"
  ],
  "Effect": "Allow",
  "Resource": ["arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable/stream/2025-02-20T11:22:33.444"]
}
```

Note

Amazon Keyspaces no admite puntos de enlace Gateway para las transmisiones de CDC.

Configuración y análisis de vulnerabilidades para Amazon Keyspaces

AWS gestiona las tareas de seguridad básicas, como la aplicación de parches al sistema operativo (SO) huésped y a las bases de datos, la configuración del firewall y la recuperación ante 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 constituyen 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 un Clave propiedad de AWS para cifrar todas las 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 AWS servicios puedan acceder a Amazon Keyspaces, deben incluir AWS credenciales válidas en sus solicitudes de AWS API. No debe almacenar AWS las credenciales directamente en la aplicación o en la instancia de 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:

- [AWS políticas gestionadas \(predefinidas\)](#)
- [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 Identity-Based de políticas](#).

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.

Se usa AWS CloudTrail para monitorear el uso de las AWS KMS claves AWS Key Management Service (AWS KMS)

Si utilizas una [AWS KMS clave gestionada por el cliente](#) para el cifrado en reposo, se inicia sesión en el uso de esta clave AWS CloudTrail. CloudTrail proporciona visibilidad de la actividad de los usuarios al registrar las acciones realizadas en tu cuenta. CloudTrail registra información importante sobre cada acción, como quién realizó la solicitud, los servicios utilizados, las acciones realizadas, los parámetros de las acciones y los elementos de respuesta devueltos por el AWS servicio. Esta información le ayuda a realizar un seguimiento de los cambios realizados en sus AWS recursos y a solucionar problemas operativos. CloudTrail facilita la garantía del cumplimiento de las políticas internas y las normas reglamentarias.

Se puede utilizar CloudTrail para auditar el uso de las claves. CloudTrail crea archivos de registro que contienen un historial de llamadas a la AWS API y eventos relacionados para tu cuenta. Estos archivos de registro incluyen todas las solicitudes de AWS KMS API que se realizaron mediante la consola, AWS los SDK y las herramientas de línea de comandos, además de las realizadas a través de AWS servicios integrados. Puede utilizar estos archivos de registro para obtener información sobre cuándo se utilizó la AWS KMS clave, 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](#).

Úselo CloudTrail para supervisar las operaciones del lenguaje de definición de datos (DDL) de Amazon Keyspaces

CloudTrail proporciona visibilidad de la actividad de los usuarios al registrar las acciones realizadas en su cuenta. CloudTrail registra información importante sobre cada acción, como quién realizó la solicitud, los servicios utilizados, las acciones realizadas, los parámetros de las acciones y los elementos de respuesta devueltos por el AWS servicio. Esta información le ayuda a realizar un seguimiento de los cambios realizados en sus AWS recursos y a solucionar problemas operativos. CloudTrail facilita la garantía del cumplimiento de las políticas internas y las normas reglamentarias.

Todas las [operaciones de DDL](#) de Amazon Keyspaces se registran automáticamente. CloudTrail Las operaciones DDL le permiten crear y administrar espacios de claves y tablas de Amazon Keyspaces.

Cuando se produce actividad en Amazon Keyspaces, esa actividad se registra en un CloudTrail evento junto con otros eventos de AWS servicio 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 eventos recientes en su Cuenta de AWS. Para obtener más información, consulte [Visualización de eventos con el historial de CloudTrail eventos](#) en la Guía del AWS CloudTrail usuario.

[Para tener un registro continuo de los eventos en su Cuenta de AWS entorno, incluidos los eventos de Amazon Keyspaces, cree una ruta](#). Un registro permite CloudTrail entregar los 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. Además, puede configurar otros AWS servicios para analizar más a fondo los datos de eventos recopilados en los CloudTrail registros y actuar en función de ellos.

Etiquetado de recursos de Amazon Keyspaces para identificación y automatización

Puede asignar metadatos a sus AWS recursos 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, utiliza Cassandra Query Language (CQL) para trabajar con su base de datos. CQL es similar en muchos aspectos a Structured Query Language (SQL).

- **Elementos de CQL:** en esta sección se describen los elementos fundamentales de CQL compatibles con Amazon Keyspaces, incluidos los identificadores, las constantes, los términos y los tipos de datos. También se explican conceptos como los tipos de cadenas, los tipos numéricos, los tipos de colecciones, etc.
- **Lenguaje de definición de datos (DDL):** las instrucciones DDL se utilizan para administrar estructuras de datos, como los espacios de claves y las tablas, en Amazon Keyspaces. En esta sección se describen las instrucciones para crear, modificar y eliminar tablas y espacios de claves, así como la restauración de tablas a partir de una copia de seguridad en un momento dado.
- **Lenguaje de manipulación de datos (DML):** las instrucciones de DML se utilizan para administrar los datos de las tablas. En esta sección se explican las instrucciones para seleccionar, insertar, actualizar y eliminar datos. También explica las capacidades de consulta avanzadas, como el uso del operador IN, la ordenación de los resultados y la paginación.
- **Built-in funciones:** Amazon Keyspaces admite diversas funciones escalares integradas que puede utilizar en sentencias CQL. En esta sección se proporciona información general sobre estas funciones y se incluyen ejemplos de su uso.

A lo largo de este tema, encontrará información detallada sobre la sintaxis, ejemplos y prácticas recomendadas para utilizar CQL de forma eficaz en Amazon Keyspaces.

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](#)
- [Built-in funciones 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:	Description (Descripción)
<code>ascii</code>	Representa una cadena de caracteres ASCII.
<code>text</code>	Representa una cadena codificada UTF-8 .
<code>varchar</code>	Representa una cadena UTF-8 codificada (varchares un alias de <code>text</code>).

Tipos numéricos

Tipo de datos:	Description (Descripción)
<code>bigint</code>	Representa un entero largo con signo de 64 bits.
<code>counter</code>	Representa un contador entero con signo de 64 bits. Para obtener más información, consulte the section called “Contadores” .
<code>decimal</code>	Representa un decimal de precisión variable.

Tipo de datos:	Description (Descripción)
<code>double</code>	Representa un valor de coma flotante IEEE 754 de 64 bits.
<code>float</code>	Representa un valor de coma flotante IEEE 754 de 32 bits.
<code>int</code>	Representa un entero con signo de 32 bits.
<code>varint</code>	Representa un valor entero dentro del rango $\pm 10^{38}$.

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:	Description (Descripción)
<code>blob</code>	Representa bytes arbitrarios.

Tipo booleano

Tipo de datos:	Description (Descripción)
boolean	Representa true o false.

Time-related tipos

Tipo de datos:	Description (Descripción)
date	Una cadena en el formato <yyyy>-<mm>-<dd>.
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:	Description (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 te permite realizar actualizaciones en cada elemento individual de la colección. Client-side Las marcas de tiempo y los ajustes de Time to Live (TTL) se almacenan para los 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.

Client-side Las marcas de tiempo y los ajustes de Time to Live (TTL) se aplican a toda la colección congelada, no a elementos individuales. Frozenlas columnas de recopilación pueden formar parte de una tablaPRIMARY KEY.

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 8 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:	Description (Descripción)
inet	Una cadena que representa una dirección IP, en formato IPv4 o IPv6.

Estático

En una tabla de Amazon Keyspaces con columnas de agrupación, puede utilizar la palabra clave `STATIC` 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 el trabajo con columnas estáticas, consulte [the section called “Estimación del consumo de capacidad de las columnas estáticas”](#).

User-defined tipos (UDT)

Amazon Keyspaces admite tipos definidos por el usuario (UDT). Puede usar cualquier tipo de datos de Amazon Keyspaces válido para crear una UDT, incluidas las colecciones y otras UDT existentes. Puede crear las UDT en un espacio de claves y utilizarlas para definir las columnas de cualquier tabla del espacio de claves.

Para obtener más información sobre la sintaxis CQL, consulte [the section called “Tipos”](#). Para obtener más información sobre cómo trabajar con las UDT, consulte [the section called “User-defined tipos \(UDT\)”](#).

Para ver cuántas UDT se admiten por espacio de claves, los niveles de anidación admitidos y otros valores y cuotas predeterminados relacionados con las UDT, consulte [the section called “Cuotas y valores predeterminados para los tipos definidos por el usuario \(UDT\) en Amazon Keyspaces”](#).

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
<code>ascii</code>	<code>string</code>	<code>string</code>	Utiliza el escape de caracteres JSON <code>\u</code> .
<code>bigint</code>	<code>integer</code> , <code>string</code>	<code>integer</code>	La cadena debe ser un entero válido de 64 bits.
<code>blob</code>	<code>string</code>	<code>string</code>	La cadena debe empezar con <code>0x</code> seguido de un número par de dígitos hexadecimales.
<code>boolean</code>	<code>boolean</code> , <code>string</code>	<code>boolean</code>	La cadena debe ser <code>true</code> o <code>false</code> .
<code>date</code>	<code>string</code>	<code>string</code>	Fecha en formato <code>YYYY-MM-DD</code> , zona horaria UTC.
<code>decimal</code>	<code>integer</code> , <code>float</code> , <code>string</code>	<code>float</code>	Puede superar la precisión de punto IEEE-754 flotante de 32 o 64 bits en el decodificador del lado del cliente.

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones INSERT JSON	Tipos de datos devueltos en instrucciones SELECT JSON	Notas
double	integer, float, string	float	La cadena debe ser un entero válido o flotante.
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.

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones INSERT JSON	Tipos de datos devueltos en instrucciones SELECT JSON	Notas
time	string	string	Hora del día en formato HH-MM-SS[.fffffffffff] .
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.
UDT	map, string	map	Utiliza la representación cartográfica JSON nativa con los nombres de los campos como claves.

Tipos de datos de JSON	Tipos de datos aceptados en instrucciones INSERT JSON	Tipos de datos devueltos en instrucciones SELECT JSON	Notas
uuid	string	string	Consulte constants para obtener el formato UUID.
varchar	string	string	Utiliza el escape de caracteres JSON \u.
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](#)
- [USA](#)
- [CREATE TABLE](#)

- [ALTER TABLE](#)
- [RESTORE TABLE](#)
- [DROP TABLE](#)
- [CREAR TIPO](#)
- [TIPO DE CAÍDA](#)

Temas

- [Keyspaces](#)
- [Tablas](#)
- [User-defined tipos \(UDT\)](#)

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](#).

Instrucciones para espacios de claves

- [CREATE KEYSPACE](#)
- [ALTER KEYSPACE](#)
- [DROP KEYSPACE](#)

- [USE](#)

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 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 multirregional, especifique `NetworkTopologyStrategy` e incluya al menos dos. 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

Puede usar la `ALTER KEYSPACE WITH` instrucción para las siguientes opciones

- **REPLICATION**— Utilice esta opción para añadir una nueva Región de AWS réplica a un espacio de claves. Puede añadir una nueva región a un espacio de claves de una sola región o de varias regiones.
- **TAGS**— Utilice esta opción para añadir o eliminar etiquetas de un espacio de teclas.

Sintaxis

```
alter_keyspace_statement ::=  
  ALTER KEYSPACE keyspace_name  
  WITH options
```

Donde:

- *keyspace_name* es el nombre del espacio de claves que se va a modificar.
- las opciones son una de las siguientes:
 - **ADD | DROP TAGS**: una lista de etiquetas de par clave-valor que se van a añadir o eliminar del espacio de claves.
 - **REPLICATION**— Un mapa que indica la estrategia de replicación del espacio de claves;
 - **class**— `NetworkTopologyStrategy` define el espacio de claves como un espacio de claves multirregional.
 - **region**— Especifique uno adicional Región de AWS para este espacio de claves. El factor de réplica para cada región es tres.
 - **CLIENT_SIDE_TIMESTAMPS**— El valor predeterminado es `DISABLED`. Solo puede cambiar el estado a `ENABLED`.

Ejemplos

Modifique un espacio de teclas como se muestra en el siguiente ejemplo para añadir etiquetas.

```
ALTER KEYSPACE my_keyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

Para añadir una tercera región a un espacio de claves de varias regiones, puede utilizar la siguiente instrucción.

```
ALTER KEYSPACE my_keyspace
WITH REPLICATION = {
  'class': 'NetworkTopologyStrategy',
  'us-east-1': '3',
  'us-west-2': '3',
  'us-west-1': '3'
} AND CLIENT_SIDE_TIMESTAMPS = {'status': 'ENABLED'};
```

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 my_keyspace;
```

USE

Utilice la USE sentencia para definir el espacio de claves actual. Esto le permite hacer referencia a objetos enlazados a un espacio de claves específico, por ejemplo, tablas y tipos, sin utilizar el nombre completo que incluye el prefijo del espacio de claves.

Sintaxis

```
use_statement ::=
```

```
USE keyspace_name
```

Donde:

- `keyspace_name` es el nombre del espacio de claves que se va a utilizar.

Ejemplo

```
USE my_keyspace;
```

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 read/write de rendimiento de capacidad, consulte [the section called “Configure los modos read/write de capacidad”](#)

Para las tablas en modo aprovisionado, puede configurar `AUTOSCALING_SETTINGS` de manera opcional. Para obtener más información sobre el escalado automático de Amazon Keyspaces y las opciones disponibles, consulte [the section called “Configuración del escalado automático en una tabla existente”](#).

Para obtener información sobre los límites de cuota y las restricciones de las tablas de Amazon Keyspaces, consulte [Cuotas](#).

Instrucciones 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 ]
                           | cdc
                           | 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. El nombre completo incluye el prefijo de espacio clave. Como alternativa, puede establecer el espacio de claves actual con la instrucción de espacio de `USE` claves.
- *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 definida por el usuario o cuyo tipo *collection* (por ejemplo, *LISTSET*, *oMAP*) está inmovilizada. 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.
 - *cdc*— Un booleano que especifica si Amazon Keyspaces crea un flujo de captura de datos de cambios (CDC) para la tabla. El valor predeterminado es *false*. Para especificar *view type* cuándo se habilita una transmisión, defina el valor *with. cdc_specification* *CUSTOM_PROPERTIES*
 - *CUSTOM_PROPERTIES*: un mapa de ajustes específicos de Amazon Keyspaces.
 - *capacity_mode*: Especifica el modo de capacidad de read/write rendimiento de 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*.

- `cdc_specification`: Especifica el flujo `view_type` de la CDC. Las opciones son:
 - `NEW_AND_OLD_IMAGES`— ambas versiones de la fila, antes y después del cambio. Este es el valor predeterminado.
 - `NEW_IMAGE`— la versión de la fila tras el cambio.
 - `OLD_IMAGE`— la versión de la fila anterior al cambio.
 - `KEYS_ONLY`— las claves de partición y agrupamiento de la fila que se ha modificado.

Para obtener más información sobre las transmisiones de los CDC, consulte [the section called “Trabajando con las transmisiones de los CDC”](#). Para ver ejemplos de código, consulte [the section called “Habilite una transmisión de los CDC para una tabla nueva”](#).

- `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 el formato Amazon Resource Name (ARN) como entrada: `kms_key_identifier:ARN`.
- `point_in_time_recovery`: especifica si la recuperación en un momento dado 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 Región de AWS. Para ello, configure los siguientes parámetros. Para obtener más información y ejemplos, consulta [the section called “Creación de una tabla multirregional en modo aprovisionado”](#).
 - `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, consulta [the section called “Creación de una nueva tabla con escalado automático”](#).
- *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 para el que la tabla debería estar siempre preparada. 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 para el que la tabla debería estar siempre preparada. 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 de objetivos, debe definir el valor objetivo. Para obtener más información acerca del seguimiento de objetivos y los periodos de recuperación, 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 capacidad consumida y capacidad aprovisionada se mantenga en este valor o cerca del mismo. *target_value* se define como un porcentaje. Un número de dos dígitos entre 20 y 90. (Obligatorio)
 - *scale_in_cooldown*: un periodo de recuperación en segundos entre actividades de escalado permite que la tabla se estabilice antes de que comience otra actividad de reducción horizontal. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
 - *scale_out_cooldown*: un periodo de recuperación en segundos entre actividades de escalado permite que la tabla 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*: un valor boolean que especifica si el `scale-in` está deshabilitado o habilitado para la tabla. Este parámetro está deshabilitado de forma predeterminada. Para activar el `scale-in`, establezca el valor boolean como `FALSE`.

Esto significa que la capacidad se reduce verticalmente de forma automática para una tabla en su 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 para el que la tabla debería estar siempre preparada. 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 para el que la tabla debería estar siempre preparada. 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 de objetivos, debe definir el valor objetivo. Para obtener más información acerca del seguimiento de objetivos y los periodos de recuperación, 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 capacidad consumida y capacidad aprovisionada se mantenga en este valor o cerca del mismo. `target_value` se define como un porcentaje. Un número de dos dígitos entre 20 y 90. (Obligatorio)
 - `scale_in_cooldown`: un periodo de recuperación en segundos entre actividades de escalado permite que la tabla se estabilice antes de que comience otra actividad de reducción horizontal. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
 - `scale_out_cooldown`: un periodo de recuperación en segundos entre actividades de escalado permite que la tabla 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`: un valor boolean que especifica si el `scale-in` está deshabilitado o habilitado para la tabla. Este parámetro está deshabilitado de forma predeterminada. Para activar el `scale-in`, establezca el valor boolean como `FALSE`.

Esto significa que la capacidad se reduce verticalmente de forma automática para una tabla en su nombre. (Opcional)

- `replica_updates`: especifica la configuración de escalado automático Región de AWS específica de una tabla multirregión. En el caso de una tabla multirregional, puede configurar la capacidad de lectura de la tabla de forma diferente según la Región de AWS. Para ello, configure los siguientes parámetros. Para obtener más información y ejemplos, consulta [the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional”](#).
- `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 multirregión debe estar activado o desactivado para todas las réplicas de la tabla.

- `minimum_units`: el nivel mínimo de rendimiento de lectura para el que la tabla debería estar siempre preparada. 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 lectura para el que la tabla debería estar siempre preparada. 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 de objetivos, debe definir el valor objetivo. Para obtener más información acerca del seguimiento de objetivos y los periodos de recuperación, 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 capacidad de lectura

consumida y capacidad de lectura aprovisionada se mantenga en este valor o cerca del mismo. `target_value` se define como un porcentaje. Un número de dos dígitos entre 20 y 90. (Obligatorio)

- `scale_in_cooldown`: un periodo de recuperación en segundos entre actividades de escalado permite que la tabla se estabilice antes de que comience otra actividad de reducción horizontal. Si no se proporciona ningún valor, el valor predeterminado es 0. (Opcional)
- `scale_out_cooldown`: un periodo de recuperación en segundos entre actividades de escalado permite que la tabla 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`: un valor boolean que especifica si el `scale-in` está deshabilitado o habilitado para la tabla. Este parámetro está deshabilitado de forma predeterminada. Para activar el `scale-in`, establezca el valor boolean como `FALSE`. Esto significa que la capacidad de lectura se reduce verticalmente de forma automática para una tabla en su 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-1:555555555555: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 “Estimación del consumo de capacidad de las columnas estáticas”](#).

Ejemplo

```

CREATE TABLE my_keyspace.my_table (
    id int,
    name text,
    region text,
    division text,
    project text STATIC,
    PRIMARY KEY (id,division));

```

Puede crear una tabla con una columna que utilice un tipo definido por el usuario (UDT). La primera sentencia de los ejemplos crea un tipo, la segunda crea una tabla con una columna que utiliza ese tipo.

Ejemplo

```

CREATE TYPE my_keyspace."udt""N@ME" (my_field int);

```



```
CREATE TABLE my_keyspace.my_table (my_col1 int pri key, my_col2 "udt""N@ME");
```

ALTER TABLE

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:
 - *AUTOSCALING_SETTINGS* incluye la configuración de escalado automático opcional para las tablas aprovisionadas. Para ver la sintaxis y descripciones detalladas, consulte [the section called “CREATE TABLE”](#). Para ver ejemplos, consulte [the section called “Configuración del escalado automático en una tabla existente”](#).
 - *cdc*— Un booleano que especifica si Amazon Keyspaces crea un flujo de captura de datos de cambios (CDC) para la tabla. El valor predeterminado es `false`. Para especificar `view type` cuándo se habilita una transmisión, defina el valor `with. cdc_specification`
 - *CUSTOM_PROPERTIES*
 - *CUSTOM_PROPERTIES*: un mapa de ajustes específicos para Amazon Keyspaces.
 - *capacity_mode*: Especifica el modo de capacidad de read/write rendimiento de 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`.

- `cdc_specification`: Especifica el flujo `view_type` de la CDC. Las opciones son:
 - `NEW_AND_OLD_IMAGES`— ambas versiones de la fila, antes y después del cambio. Este es el valor predeterminado.
 - `NEW_IMAGE`— la versión de la fila tras el cambio.
 - `OLD_IMAGE`— la versión de la fila anterior al cambio.
 - `KEYS_ONLY`— las claves de partición y agrupamiento de la fila que se ha modificado.

Para obtener más información sobre las transmisiones de los CDC, consulte [the section called “Trabajando con las transmisiones de los CDC”](#). Para ver ejemplos de código, consulte [the section called “Habilite una transmisión de CDC para una tabla existente”](#).

- `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 la recuperación en un momento dado 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 multirregional, puede configurar la capacidad de lectura de la tabla de forma diferente según la Región de AWS. Para ello, configure los siguientes parámetros. Para obtener más información y ejemplos, consulta [the section called “Actualización de la capacidad aprovisionada y la configuración de escalado automático para una tabla multirregional”](#).
 - `region`— La réplica Región de AWS de la tabla con los siguientes ajustes:
 - `read_capacity_units`

- `t1`: 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, `t1` no se puede deshabilitar para la tabla.
- `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>>>>);
```

El siguiente ejemplo muestra cómo añadir una columna que utilice un tipo definido por el usuario (UDT) a una tabla.

```
ALTER TABLE my_keyspace.my_table ADD (my_column, my_udt);
```

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-1:555555555555:key/11111111-1111-111-1111-111111111111'
    }
};
```

Para habilitar la restauración en un momento dado para una tabla, puede utilizar la siguiente instrucción.

```
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 instrucción requiere que la recuperación en un momento dado esté habilitada en una tabla. Para obtener más información, consulte [the section called “Copias de seguridad y restauración con recuperación en 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 read/write rendimiento de 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 el 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 la recuperación en un momento dado 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 multirregional, puede configurar la capacidad de lectura de la tabla de forma diferente según la Región de AWS. Para ello, configure 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 ver en detalle la sintaxis y las descripciones, consulte [the 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 my_keyspace.my_table;
```

User-defined tipos (UDT)

UDT: agrupación de campos y tipos de datos que puede utilizar para definir una sola columna en Amazon Keyspaces. Los tipos de datos válidos para las UDT son todos los tipos de datos de Cassandra compatibles, incluidas las colecciones y otras UDT que ya haya creado en el mismo espacio de claves. Para obtener más información sobre los tipos de datos de Cassandra compatibles, consulte [the section called “Compatibilidad con tipos de datos de Cassandra”](#)

```
user_defined_type ::= udt_name
udt_name ::= [ keyspace_name '.' ] identifier
```

Declaraciones de tipos

- [CREATE TYPE](#)
- [DROP TYPE](#)

CREATE TYPE

Utilice la CREATE TYPE sentencia para crear un tipo nuevo.

Sintaxis

```
create_type_statement ::= CREATE TYPE [ IF NOT EXISTS ] udt_name
    '(' field_definition ( ',' field_definition ) * ')'
    field_definition ::= identifier cql_type
```

Donde:

- IF NOT EXISTS CREATE TYPE evita que se produzca un error si el tipo ya existe. (Opcional)
- *udt_name* es el nombre completo de la UDT en formato tipográfico, por ejemplo.
my_keyspace.my_type Si define el espacio de claves actual con la USE instrucción, no necesita especificar el nombre del espacio de claves.
- *field_definition* consta de un nombre y un tipo.

En la siguiente tabla se muestran ejemplos de nombres UDT permitidos. La primera columna muestra cómo introducir el nombre al crear el tipo, y la segunda columna muestra cómo Amazon Keyspaces formatea el nombre internamente. Amazon Keyspaces espera el nombre formateado para operaciones como. GetType

Nombre introducido	Nombre formateado	Nota
MY_UDT	my_udt	Sin comillas dobles, Amazon Keyspaces convierte todos los caracteres en mayúsculas en minúsculas.
"MY_UDT"	MY_UDT	Con las comillas dobles, Amazon Keyspaces respeta los caracteres en mayúscula y elimina las comillas dobles del nombre formateado.
"1234"	1234	Con comillas dobles, el nombre puede empezar por un número y Amazon Keyspaces elimina las comillas dobles del nombre formateado.
"Special_Ch@r@cter s<>!!"	Special_Ch@r@cter s<>!!	Con comillas dobles, el nombre puede contener caracteres especiales y Amazon Keyspaces elimina las comillas dobles del nombre formateado.
"nested""""quote s"	nested""""quotes	Amazon Keyspaces elimina las comillas dobles exteriores y las comillas dobles de escape del nombre formateado.

Ejemplos

```
CREATE TYPE my_keyspace.phone (
    country_code int,
    number text
);
```

Puede anidar las UDT si las UDT anidadas están inmovilizadas. Para obtener más información sobre los valores predeterminados y las cuotas de los tipos, consulte. [the section called “Cuotas de UDT y valores predeterminados de Amazon Keyspaces”](#)

```
CREATE TYPE my_keyspace.user (
    first_name text,
    last_name text,
    phones FROZEN<phone>
);
```


Para ver más ejemplos de código que muestran cómo crear UDT, consulte [the section called “User-defined tipos \(UDT\)”](#).

DROP TYPE

Utilice la `DROP TYPE` sentencia para eliminar una UDT. Solo puede eliminar un tipo que no esté siendo utilizado por otro tipo o tabla.

Sintaxis

```
drop_type_statement ::= DROP TYPE [ IF EXISTS ] udt_name
```

Donde:

- `IF EXISTS` `DROP TYPE` evita que se produzca un error si el tipo no existe. (Opcional)
- *udt_name* es el nombre completo de la UDT en formato tipográfico, por ejemplo. `my_keyspace.my_type` Si define el espacio de claves actual con la `USE` instrucción, no necesita especificar el nombre del espacio de claves.

Ejemplo

```
DROP TYPE udt_name;
```

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 ;

```

Para ver una tabla que asigna JSON-encoded los tipos de datos 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 “Uso de 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 agrupamiento en clústeres en las cláusulas de ordenación. Non-clustering no se permiten columnas. 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 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 [the section called “Client-side marcas de tiempo”](#).

```
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

Para ver una tabla que asigna JSON-encoded los tipos de datos 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.

Built-in funciones 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	Description (Descripción)
<code>blobAsType</code>	Devuelve un valor del tipo de datos especificado.
<code>cast</code>	Convierte un tipo de datos nativo en otro tipo de datos nativo.
<code>currentDate</code>	Devuelve la fecha actual date/time .
<code>currentTime</code>	Devuelve la corriente en date/time forma de hora.
<code>currentTimestamp</code>	Devuelve la corriente date/time como una marca de tiempo.
<code>currentTimeUUID</code>	Devuelve la corriente date/time como un. timeuuid
<code>fromJson</code>	Convierte la cadena JSON en el tipo de datos de la columna seleccionada.
<code>maxTimeuuid</code>	Devuelve el timeuuid más grande posible para la marca de tiempo o la cadena de fecha.

Función	Description (Descripción)
<code>minTimeuuid</code>	Devuelve el <code>timeuuid</code> más pequeño posible para la marca de tiempo o la cadena de fecha.
<code>now</code>	Devuelve un nuevo <code>timeuuid</code> único.
<code>toDate</code>	Convierte un <code>timeuuid</code> o una marca de tiempo en un tipo de fecha.
<code>toJson</code>	Devuelve el valor de la columna seleccionada en formato JSON.
<code>token</code>	Devuelve el valor hash de la clave de partición.
<code>toTimestamp</code>	Convierte un <code>timeuuid</code> o una fecha en una marca de tiempo.
<code>TTL</code>	Devuelve el tiempo de caducidad en segundos de una columna.
<code>typeAsBlob</code>	Convierte el tipo de datos especificado en un blob.
<code>toUnixTimestamp</code>	Convierte un <code>timeuuid</code> o una marca de tiempo en un <code>bigInt</code> .
<code>uuid</code>	Devuelve un UUID versión 4 aleatorio.
<code>writetime</code>	Devuelve la marca de tiempo del valor de la columna especificada.
<code>dateOf</code>	(En desuso) Extrae la marca de tiempo de un <code>timeuuid</code> y devuelve el valor como fecha.
<code>unixTimestampOf</code>	(En desuso) Extrae la marca de tiempo de un <code>timeuuid</code> 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 y valores predeterminados para los tipos definidos por el usuario \(UDT\) en 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	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Espacio máximo de teclas por Región de AWS	El número máximo de espacios de claves para este suscriptor por región. Puede ajustar este valor predeterminado en la consola de Service Quotas .	256
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 de Service Quotas .	256

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Cantidad máxima de datos restaurados mediante ADD REGION operaciones	El tamaño máximo de los datos que las ADD REGION operaciones pueden restaurar simultáneamente. Para aumentar la cantidad de datos que se van a restaurar simultáneamente, póngase en contacto con. Soporte	10 TB
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 información, consulte the section called “Estimación del tamaño de las filas” .	1 MB

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
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
Número máximo de datos estáticos por partición lógica	El tamaño agregado máximo de datos estáticos en una partición lógica. Para obtener más información, 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 de Service Quotas .	100

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Número máximo de colecciones congeladas anidadas 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 Soporte	8
Número máximo de declaraciones en un lote registrado	El número máximo de sentencias CQL admitidas cuando se utiliza una sentencia por lotes registrada. Para obtener más información sobre las declaraciones por lotes, consulte the section called “Utilice declaraciones por lotes”	100
Número máximo de declaraciones en un lote registrado mediante columnas estáticas	El número máximo de sentencias CQL admitidas cuando se utilizan columnas estáticas en una sentencia por lotes registrada. Para obtener más información sobre las declaraciones por lotes, consulte the section called “Utilice declaraciones por lotes”	50

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Tamaño máximo de la declaración de lote registrada	El tamaño máximo de una declaración de lote registrada. Para obtener más información sobre las declaraciones de lotes, consulte the section called “Utilice declaraciones por lotes” .	4 MB
Número máximo de declaraciones en un lote no registrado	El número máximo de sentencias CQL admitidas cuando se utiliza una sentencia por lotes no registrada. Para obtener más información sobre las declaraciones por lotes, consulte the section called “Utilice declaraciones por lotes” .	30
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 de Service Quotas .	40 000

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
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 de Service Quotas .	40 000
Rendimiento máximo de escritura de una tabla mediante la captura de datos de cambios (CDC)	La capacidad máxima de escritura de una tabla de Amazon Keyspaces con CDC activado. Esta cuota predeterminada solo se aplica a las tablas en el modo de read/write capacidad aprovisionada. Puede ajustar este valor predeterminado en la consola de Service Quotas .	40 000
Account-level rendimiento de lectura (aprovisionado)	El número máximo de unidades de capacidad de lectura (RCU) agregadas para la cuenta por región. Esto solo se aplica a las tablas en el modo de capacidad aprovisionada read/write . Puede ajustar este valor predeterminado en la consola de Service Quotas .	80 000

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Account-level rendimiento de escritura (aprovisionado)	El número máximo de unidades de capacidad de escritura (WCU) agregadas para la cuenta por región. Esto solo se aplica a las tablas en el modo de capacidad aprovisionada read/write . Puede ajustar este valor predeterminado en la consola de Service Quotas .	80 000
Número máximo de objetivos escalables por región y por 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 de Service Quotas de Application Auto Scaling seleccionando Objetivos escalables para Amazon Keyspaces.	1500

Cuota	Description (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
Número máximo de restauraciones simultáneas de tablas mediante Point-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 de Service Quotas .	4
Cantidad máxima de datos restaurados mediante recuperación en un momento dado (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 de Service Quotas .	5 TB

Aumento o reducción del rendimiento (para las tablas aprovisionadas)

Aumento del rendimiento aprovisionado

Puede aumentar `ReadCapacityUnits` o `WriteCapacityUnits` tantas veces como sea necesario mediante la consola o la instrucción `ALTER TABLE`. La nueva configuración no surtirá efecto hasta que se haya completado la operación `ALTER TABLE`.

No puede superar sus cuotas por cuenta al añadir capacidad aprovisionada. Y puede aumentar la capacidad aprovisionada de las 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 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 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 [Soporte](#).

Cuotas y valores predeterminados para los tipos definidos por el usuario (UDT) en Amazon Keyspaces

Cuotas de UDT y valores predeterminados de Amazon Keyspaces

La siguiente tabla contiene las cuotas y los valores predeterminados relacionados con las UDT en Amazon Keyspaces. 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 estas cuotas, póngase en contacto con AWS Support

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Número máximo de UDT por Región de AWS	El número máximo de UDT en todos los espacios clave para este suscriptor por región.	256
Número máximo de tablas por UDT	El número máximo de tablas que pueden hacer referencia a la misma UDT.	100
Número máximo de UDT por tabla	El número máximo de UDT a los que puede hacer referenci a una tabla.	50
Nivel máximo de anidación de las UDT	La profundidad máxima de anidación admitida por las UDT.	8
Cantidad máxima de UDT secundarias directas por UDT	El número máximo de UDT secundarios que se admiten para una UDT.	10

Cuota	Description (Descripción)	Valores predeterminados de Amazon Keyspaces
Cantidad máxima de UDT principales directas por UDT	El número máximo de UDT principales compatibles con una UDT.	10
Tamaño máximo del esquema UDT	El tamaño máximo del esquema de una UDT.	25 KB
Longitud máxima del nombre UDT	El número máximo de caracteres del nombre UDT.	48
Longitud máxima del nombre de campo UDT	El número máximo de caracteres de un nombre de campo UDT.	128

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: 28 de enero de 2026

Cambio	Descripción	Fecha
Ahora puedes precalentar tablas nuevas y existentes en Amazon Keyspaces	Ahora puede configurar warm-throughput las tablas nuevas y existentes en Amazon Keyspaces a fin de precalentarlas para los picos de capacidad previstos.	28 de enero de 2026
Amazon Keyspaces ahora admite una experiencia integrada AWS CloudShell	Ahora puede lanzar un shell previamente autenticado directamente desde la consola de Amazon Keyspaces.	26 de enero de 2026
Amazon Keyspaces ahora admite lotes registrados	Ahora puede combinar varias declaraciones en lotes registrados en Amazon Keyspaces.	11 de noviembre de 2025
Cambios en los certificados TLS de Amazon Keyspaces	Se actualizaron las instrucciones para incluir los certificados emitidos en el marco de la cadena Amazon Trust Services (ATS).	25 de octubre de 2025

[Amazon Keyspaces ahora admite solicitudes de IPv4 e IPv6](#)

Los puntos de enlace de doble pila de Amazon Keyspaces admiten configuraciones de IPv4 e IPv6.

30 de septiembre de 2025

[NOWy ahora UUID se admiten en todas las declaraciones de CQL de Amazon Keyspaces](#)

Ahora puede utilizar las UUID funciones NOW y en todas las sentencias CQL de Amazon Keyspaces.

8 de septiembre de 2025

[Ya está disponible el adaptador Amazon Keyspaces Streams Kinesis](#)

Ahora puede procesar las transmisiones de los CDC de Amazon Keyspaces con la biblioteca de clientes de Kinesis (KCL).

2 de julio de 2025

[Nueva política gestionada de Amazon Keyspaces](#)

Amazon Keyspaces ha añadido una nueva política gestionada Keyspaces `CDCServiceRolePolicy` que concede los permisos necesarios al rol vinculado al servicio para publicar los datos de las métricas de streaming de los CDC de `AWSServiceRoleForAmazonKeyspacesCDC` Amazon Keyspaces en su nombre. CloudWatch Para obtener más información, consulte [Uso de roles para las transmisiones de los CDC de Amazon Keyspaces](#).

2 de julio de 2025

[Actualización de la política administrada de Amazon Keyspaces](#)

Amazon Keyspaces agregó nuevos permisos a las políticas administradas AmazonKeyspacesReadOnlyAccess_v2 y a las políticas AmazonKeyspacesReadOnlyAccess administradas para permitir a los directores de IAM ver las transmisiones. Amazon Keyspaces actualizó la política AmazonKeyspacesFullAccess gestionada con permisos para crear un rol vinculado a un servicio AWSServiceRoleForAmazonKeyspacesCDC que publique CloudWatch las métricas en nombre del director.

2 de julio de 2025

[Ya están disponibles las transmisiones de captura de datos de cambios \(CDC\) de Amazon Keyspaces](#)

Habilita las transmisiones de CDC para registrar los cambios en tus tablas de Amazon Keyspaces.

2 de julio de 2025

[La replicación multirregional de Amazon Keyspaces ahora es compatible con la región de África \(Ciudad del Cabo\).](#)

Ahora puede definir la región de África (Ciudad del Cabo) para los espacios clave multirregionales.

31 de marzo de 2025

[La replicación multirregional de Amazon Keyspaces ahora admite de forma ilimitada. Regiones de AWS](#)

Ahora puede definir tantas regiones como desee para los espacios de claves multirregionales.

25 de marzo de 2025

[Compatibilidad con Amazon Keyspaces para tipos definidos por el usuario \(UDT\) en espacios clave multirregionales](#)

Ahora puede crear y utilizar las UDT en espacios de claves de varias regiones.

24 de marzo de 2025

[Actualización de la política administrada de Amazon Keyspaces](#)

Amazon Keyspaces ha añadido nuevos permisos a la política AmazonKey spacesFullAccess gestionada para permitir a los directores de IAM añadir nuevas regiones a un espacio de claves existente. Esto incluye una actualización de la función vinculada al servicio. [AWSServiceRoleForAmazonKeyspacesReplication](#)

19 de noviembre de 2024

[Añadir réplicas a tablas de varias regiones en Amazon Keyspaces](#)

Ahora puede añadir nuevos espacios de claves de una o Regiones de AWS varias regiones a los existentes.

19 de noviembre de 2024

[Support para tipos definidos por el usuario \(UDT\) en Amazon Keyspaces](#)

Gracias a la compatibilidad nativa con las UDT en Amazon Keyspaces, ahora puede definir estructuras de datos en sus aplicaciones que representen jerarquías de datos del mundo real.

30 de octubre de 2024

[ADD COLUMNcompatibilidad con la replicación multirregional de Amazon Keyspaces](#)

Amazon Keyspaces ahora es compatible con los cambios de esquema de las tablas multirregión.

17 de septiembre de 2024

[Se ha actualizado la guía de migración de Amazon Keyspaces](#)

La guía de migración actualizada describe cómo crear un plan de migración para migrar correctamente de Apache Cassandra a Amazon Keyspaces, incluidas diferentes estrategias para las migraciones online y offline.

28 de junio de 2024

[Conexión a Amazon Keyspaces desde Amazon Elastic Kubernetes Service](#)

Ahora puede seguir un tutorial paso a paso para conectarse a Amazon Keyspaces desde Amazon EKS.

7 de febrero de 2024

[Soporte de replicación multirregional de Amazon Keyspaces para tablas aprovisionadas](#)

Ahora Amazon Keyspaces es compatible con el modo de capacidad aprovisionada para las tablas multirregión.

23 de enero 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

[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

[Compatibilidad de Amazon Keyspaces con la palabra clave FROZEN](#)

Amazon Keyspaces admite ahora la palabra clave FROZEN para los tipos de datos de colección.

15 de noviembre de 2023

[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 ahora admite la replicación multirregional](#)

La replicación multirregional de Amazon Keyspaces le ayuda a mantener las aplicaciones distribuidas por todo el mundo al proporcionarle una mejor tolerancia a errores, estabilidad y resiliencia.

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

[Cross-account acceso a Amazon Keyspaces y puntos de enlace 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 Cassandra-compatible nivel de celda 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 tutorial paso a paso puede informarse de cómo 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 identificar estrategias para optimizar los costos de sus tablas de Amazon Keyspaces existentes.

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 AWS GovCloud (US) Region y está sujeto a conformidad con las FedRAMP-High normas. 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 <code>BillableTableSizeInBytes</code> 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 información 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.	Step-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 `system.peers` tabla con información sobre los puntos de conexión de VPC de la interfaz disponibles para mejorar el equilibrio de carga y aumentar el rendimiento. `read/write`

29 de julio de 2021

[Actualice 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

[Recuperación en un momento dado \(PITR\) de Amazon Keyspaces](#)

Amazon Keyspaces ofrece ahora la recuperación en un momento dado (PITR) para ayudarle a proteger sus tablas de operaciones accidentales de escritura o eliminación, proporcionándole copias de seguridad continuas de los datos de sus 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

<u>Escalado automático de Amazon Keyspaces</u>	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
<u>Puntos de conexión de nube privada virtual (VPC) de interfaz para Amazon Keyspaces</u>	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
<u>Tag-based políticas de acceso</u>	Ahora puede utilizar etiquetas de recursos en las políticas de IAM para administrar el acceso a Amazon Keyspaces.	8 de abril de 2020
<u>Tipo de datos de contador</u>	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
<u>Etiquetado de recursos</u>	Amazon Keyspaces ahora le permite etiquetar y clasificar los recursos mediante etiquetas.	31 de marzo de 2020

CloudFormation soporte	Amazon Keyspaces ahora le ayuda a automatizar la creación y administración de recursos con el uso de 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
Read/write modo de capacidad	Amazon Keyspaces ahora admite dos modos de read/write e capacidad de rendimiento. El modo read/write de capacidad controla cómo se le cobra por el rendimiento de lectura y escritura y cómo se administra la capacidad de rendimiento de la tabla.	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.