



Panduan Developer

Amazon Keyspaces (untuk Apache Cassandra)



Amazon Keyspaces (untuk Apache Cassandra): Panduan Developer

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

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau mungkin tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

Apa itu Amazon Keyspaces?	1
Cara kerjanya	2
High-level arsitektur	2
Model data Cassandra	5
Mengakses Keyspaces Amazon	6
Kasus penggunaan	7
Apa itu CQL?	7
Bandingkan Amazon Keyspaces dengan Cassandra	9
Perbedaan fungsional dengan Apache Cassandra	10
Apache Cassandra API, operasi, dan tipe data	11
Pembuatan asinkron dan penghapusan ruang kunci dan tabel	11
Autentikasi dan otorisasi	11
Batch	11
Change Data Capture (CDC)	11
Konfigurasi kluster	12
Koneksi	12
INKata kunci	13
FROZENkoleksi	13
Transaksi ringan	14
Penyeimbangan beban	14
Paginasi	14
Partisi	15
Pernyataan yang disiapkan	15
Hapus rentang	15
Tabel sistem	16
Stempel waktu	16
User-defined jenis (UDT)	16
API Cassandra yang didukung, operasi, fungsi, dan tipe data	17
Dukungan Cassandra API	17
Dukungan API bidang kontrol Cassandra	19
Dukungan API pesawat data Cassandra	19
Dukungan fungsi Cassandra	20
Dukungan tipe data Cassandra	20
Tingkat konsistensi Cassandra yang didukung	22

Tulis tingkat konsistensi	22
Baca tingkat konsistensi	23
Tingkat konsistensi yang tidak didukung	24
Migrasi ke Amazon Keyspaces	25
Migrasi dari Cassandra	26
Kompatibilitas	27
Perkiraan harga	28
Strategi migrasi	37
Migrasi online	38
Migrasi offline	50
Migrasi hibrida	53
Alat Migrasi	57
Memuat data menggunakan cqlsh	59
Memuat data menggunakan DSBulk	72
Mengakses Keyspaces Amazon	89
Penyiapan AWS Identity and Access Management	89
Mendaftar untuk Akun AWS	89
Menyiapkan Amazon Keyspaces	89
Menggunakan konsol	90
CloudShell integrasi konsol	90
Menggunakan AWS CloudShell	92
AWS CloudShell opsi integrasi	92
Prasyarat	93
Memulai dengan AWS CloudShell	93
Memperoleh izin IAM untuk AWS CloudShell	93
Menginstal klien cqlsh	94
Mengkonfigurasi sertifikat TLS	95
Menghubungkan ke Amazon Keyspaces	95
Menggunakan CQLSH dengan Amazon Keyspaces	95
Data yang bertahan di AWS CloudShell	95
Praktik terbaik	96
Pemecahan masalah	96
Contoh: Cara berinteraksi dengan Amazon Keyspaces menggunakan AWS CloudShell	96
Buat kredensial akses terprogram	98
Buat kredensial khusus layanan	99
Buat kredensial IAM untuk otentikasi AWS	101

Titik akhir layanan	110
Port dan protokol	110
Titik akhir global	111
AWS GovCloud (US) Region Titik akhir FIPS	116
Titik akhir Wilayah Tiongkok	116
Titik akhir aliran	117
Menghubungkan ke titik akhir dual-stack	120
Dukungan IPv6 di Amazon Keyspaces	121
Resolusi DNS di Amazon Keyspaces	121
Menggunakan cqlsh	122
Menggunakan cqlsh-expansion	123
Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS	128
Menggunakan AWS CLI	130
Mengunduh dan Mengkonfigurasi AWS CLI	130
Menggunakan AWS CLI Keyspaces dengan Amazon	130
Menggunakan API	135
Menggunakan driver klien Cassandra	135
Menggunakan driver klien Cassandra Java	136
Menggunakan driver klien Cassandra Python	153
Menggunakan driver klien Cassandra Node.js	157
Menggunakan driver klien Cassandra .NET Core	162
Menggunakan driver klien Cassandra Go	163
Menggunakan driver klien Cassandra Perl	169
Konfigurasi akses lintas akun	171
Konfigurasi akses lintas akun di VPC bersama	171
Konfigurasi akses lintas akun tanpa VPC bersama	175
Memulai	180
Prasyarat	181
Buat ruang kunci	182
Periksa status pembuatan keyspace	185
Membuat tabel	186
Periksa status pembuatan tabel	194
Operasi CRUD	195
Buat	196
Baca	199
Perbarui	204

Delete	205
Menghapus tabel	207
Hapus ruang kunci	209
Tutorial dan solusi	213
Menghubungkan dengan titik akhir VPC	213
Prasyarat	214
Langkah 1: Luncurkan instans Amazon EC2	214
Langkah 2: Konfigurasi instans Amazon EC2 Anda	216
Langkah 3: Buat titik akhir VPC untuk Amazon Keyspaces	219
Langkah 4: Konfigurasi izin untuk koneksi titik akhir VPC	225
Langkah 5: Konfigurasi pemantauan	228
Langkah 6: (Opsional) Praktik terbaik untuk koneksi	229
Langkah 7: (Opsional) Bersihkan	232
Integrasi dengan Apache Spark	233
Prasyarat	234
Langkah 1: Konfigurasi Amazon Keyspaces	235
Langkah 2: Konfigurasi Konektor Spark Apache Cassandra	236
Langkah 3: Buat file konfigurasi aplikasi	238
Langkah 4: Siapkan data sumber dan tabel target	241
Langkah 5: Tulis dan baca data Amazon Keyspaces	242
Pemecahan masalah	245
Menghubungkan dari Amazon EKS	247
Prasyarat	248
Langkah 1: Konfigurasi cluster Amazon EKS	250
Langkah 2: Konfigurasi aplikasi	255
Langkah 3: Buat gambar aplikasi	258
Langkah 4: Menyebarkan aplikasi ke Amazon EKS	260
Langkah 5: Pembersihan (Opsional)	265
Mengekspor data ke Amazon S3	268
Prasyarat	268
Langkah 1: Buat bucket Amazon S3, unduh alat, dan konfigurasi lingkungan	270
Langkah 2: Konfigurasi AWS Glue pekerjaan	272
Langkah 3: Jalankan AWS Glue pekerjaan ekspor dari AWS CLI	275
Langkah 4: (Opsional) Jadwalkan pekerjaan ekspor	277
Langkah 5: Pembersihan (Opsional)	278
Mengelola sumber daya tanpa server	281

Perkirakan ukuran baris	282
Perkirakan ukuran kolom yang dikodekan	283
Perkirakan ukuran nilai data yang dikodekan berdasarkan tipe data	284
Pertimbangkan dampak fitur Amazon Keyspaces pada ukuran baris	285
Pilih rumus yang tepat untuk menghitung ukuran baris yang dikodekan	286
Contoh perhitungan ukuran baris	287
Perkirakan konsumsi kapasitas	288
Perkirakan konsumsi kapasitas kueri rentang	289
Perkirakan konsumsi kapasitas baca kueri batas	290
Perkirakan konsumsi kapasitas baca pemindaian tabel	291
Perkirakan konsumsi kapasitas LWT	292
Perkirakan konsumsi kapasitas kolom statis	292
Perkirakan kapasitas untuk tabel Multi-wilayah	296
Perkirakan konsumsi kapasitas dengan CloudWatch	298
Konfigurasi mode read/write kapasitas	298
Konfigurasi mode kapasitas sesuai permintaan	299
Konfigurasi mode kapasitas yang disediakan	302
Lihat mode kapasitas tabel	304
Ubah mode kapasitas	305
Konfigurasi pra-pemanasan	308
Kelola kapasitas throughput dengan penskalaan otomatis	322
Cara kerja penskalaan otomatis Amazon Keyspaces	323
Cara kerja penskalaan otomatis untuk tabel Multi-wilayah	325
Catatan penggunaan	326
Konfigurasi dan perbarui kebijakan penskalaan otomatis	327
Gunakan kapasitas burst	340
Bekerja dengan fitur Amazon Keyspaces	342
Ruang kunci sistem	343
system	344
system_schema	346
system_schema_mcs	347
system_multiregion_info	349
User-defined jenis (UDT)	352
Mengatur Konfigurasi Izin	353
Buat UDT	356
Lihat UDT	360

Hapus UDT	363
Bekerja dengan kueri CQL	365
Gunakan IN SELECT	365
Gunakan pernyataan batch	369
Hasil pesanan	375
Hasil paginate	376
Bekerja dengan aliran CDC	377
Cara kerja aliran CDC	378
Gunakan aliran CDC	382
Bekerja dengan partisi	416
Ubah partisi	416
Client-side stempel waktu	418
Integrasi dengan AWS layanan	420
Buat tabel dengan stempel waktu sisi klien	420
Konfigurasi stempel waktu sisi klien	423
Gunakan stempel waktu sisi klien dalam kueri	426
Multi-Region replikasi	427
Manfaat	428
Mode kapasitas dan harga	429
Cara kerjanya	429
Catatan penggunaan	435
Konfigurasi replikasi multi-Region	437
Backup dan restore dengan pemulihan point-in-time	466
Cara kerjanya	466
Gunakan pemulihan point-in-time	472
Data kedaluwarsa dengan Time to Live	485
Integrasi dengan AWS layanan	486
Buat tabel dengan nilai TTL default	487
Perbarui tabel nilai TTL default	491
Buat tabel dengan TTL khusus	494
Perbarui tabel kustom TTL	496
Gunakan INSERT untuk mengatur TTL kustom untuk baris baru	499
Gunakan UPDATE untuk mengatur TTL kustom untuk baris dan kolom	500
Bekerja dengan AWS SDK	501
Bekerja dengan tag	502
Batasan penandaan	503

Tandai ruang kunci, tabel, dan aliran	504
Buat laporan alokasi biaya	522
Buat CloudFormation sumber daya	523
Amazon Keyspaces dan template CloudFormation	524
Pelajari lebih lanjut tentang CloudFormation	524
NoSQL Workbench	524
Unduh	525
Memulai	526
Visualisasikan model data	528
Buat model data	532
Mengedit model data	535
Komit model data	537
Model data sampel	547
Riwayat rilis	548
Contoh kode	550
Hal-hal mendasar	551
Halo Amazon Keyspaces	551
Pelajari dasar-dasarnya	556
Tindakan	618
Perpustakaan dan alat-alat	670
Perpustakaan dan contoh	670
Perangkat pengembang Amazon Keyspaces (untuk Apache Cassandra)	670
Amazon Keyspaces (untuk Apache Cassandra) contoh	670
AWS Plugin otentikasi Sigv4 Versi Tanda Tangan 4 (SiGv4)	670
Contoh yang disorot dan repo alat pengembang	671
Buffer Protokol Amazon Keyspaces	671
CloudFormation template untuk membuat CloudWatch dasbor Amazon untuk metrik	
Amazon Keyspaces (untuk Apache Cassandra)	671
Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan AWS Lambda	672
Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan Spring	672
Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan Scala	672
Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan AWS Glue	672
Amazon Keyspaces (untuk Apache Cassandra) Bahasa kueri Cassandra (CQL) ke konverter	
CloudFormation	673
Amazon Keyspaces (untuk Apache Cassandra) pembantu untuk driver Apache Cassandra	
untuk Java	673

Amazon Keyspaces (untuk Apache Cassandra) demo kompresi tajam	673
Amazon Keyspaces (untuk Apache Cassandra) dan demo codec Amazon S3	673
Praktik terbaik	674
Desain NoSQL	675
NoSQL vs. RDBMS	676
Dua konsep utama	676
Pendekatan umum	677
Koneksi	678
Bagaimana mereka bekerja	678
Cara mengkonfigurasi koneksi	679
Cara mengonfigurasi kebijakan coba lagi	681
Koneksi titik akhir VPC	681
Cara memonitor koneksi	683
Cara menangani kesalahan koneksi	684
Pemodelan data	684
Desain kunci partisi	685
Optimalisasi biaya	687
Mengevaluasi biaya di tingkat tabel	688
Evaluasi mode kapasitas tabel Anda	690
Evaluasi pengaturan Application Auto Scaling tabel Anda	695
Identifikasi sumber daya Anda yang tidak terpakai	702
Evaluasi pola penggunaan tabel Anda	708
Evaluasi kapasitas yang disediakan untuk penyediaan ukuran yang tepat	709
Pemecahan masalah	719
Kesalahan umum	720
Kesalahan umum	720
Kesalahan koneksi	722
Kesalahan saat menghubungkan ke titik akhir Amazon Keyspaces	722
Kesalahan manajemen kapasitas	734
Kesalahan kapasitas tanpa server	734
Kesalahan bahasa definisi data	740
Kesalahan bahasa definisi data	740
Memantau Amazon Keyspaces	746
Pemantauan CloudWatch dengan	747
Menggunakan metrik	748
Metrik dan dimensi	749

Melihat metrik	773
Buat alarm	774
Logging dengan CloudTrail	774
Mengkonfigurasi entri file log di CloudTrail	775
Informasi DDL di CloudTrail	776
Informasi DML di CloudTrail	777
Memahami entri file log	778
Keamanan	790
Perlindungan data	791
Enkripsi saat diam	792
Enkripsi saat bergerak	813
Privasi lalu lintas antarjaringan	814
AWS Identity and Access Management	815
Audiens	816
Mengautentikasi dengan identitas	816
Mengelola akses menggunakan kebijakan	817
Cara Amazon Keyspaces bekerja dengan IAM	819
Identity-based contoh kebijakan	825
AWS kebijakan terkelola	833
Pemecahan masalah	845
Menggunakan Peran Terkait Layanan	849
Validasi kepatuhan	858
Ketahanan	859
Keamanan infrastruktur	860
Menggunakan VPC endpoint antarmuka	861
Menggunakan titik akhir VPC antarmuka dengan aliran	868
Analisis konfigurasi dan kerentanan untuk Amazon Keyspaces	871
Praktik terbaik keamanan	872
Praktik terbaik keamanan pencegahan	872
Praktik terbaik keamanan detektif	874
Referensi bahasa CQL	876
Elemen bahasa	877
Pengidentifikasi	877
Konstanta	877
Istilah	877
Jenis Data	878

Pengkodean JSON dari tipe data Amazon Keyspaces	882
Pernyataan DDL	886
Keyspaces	887
Tabel	891
Tipe	904
Pernyataan DXML	907
SELECT	908
INSERT	910
UPDATE	912
DELETE	913
Built-in fungsi	914
Fungsi skalar	914
Kuota	917
Kuota layanan Amazon Keyspaces	917
Meningkatkan atau mengurangi throughput (untuk tabel yang disediakan)	924
Meningkatkan throughput yang disediakan	924
Menurunkan throughput yang disediakan	924
Enkripsi Amazon Keyspaces saat istirahat	924
Kuota dan nilai default untuk tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces	925
Kuota UDT Amazon Keyspaces dan nilai default	925
Riwayat dokumen	927
.....	cmxlii

Apa itu Amazon Keyspaces (untuk Apache Cassandra)?

Amazon Keyspaces (untuk Apache Cassandra) adalah layanan basis data yang kompatibel dengan Apache Cassandra yang dapat diskalakan, sangat tersedia, dan dikelola. Dengan Amazon Keyspaces, Anda tidak perlu menyediakan, menambal, atau mengelola server, dan Anda tidak perlu menginstal, memelihara, atau mengoperasikan perangkat lunak.

Amazon Keyspaces tanpa server, jadi Anda hanya membayar sumber daya yang Anda gunakan, dan layanan secara otomatis menskalakan tabel ke atas dan ke bawah sebagai respons terhadap lalu lintas aplikasi. Anda dapat membangun aplikasi yang melayani ribuan permintaan per detik dengan throughput dan penyimpanan yang hampir tidak terbatas.

Note

Apache Cassandra adalah sumber data open-source, kolom lebar yang dirancang untuk menangani data dalam jumlah besar. Untuk informasi lebih lanjut, lihat [Apache Cassandra](#).

Amazon Keyspaces memudahkan migrasi, menjalankan, dan menskalakan beban kerja Cassandra di file. AWS Cloud Hanya dengan beberapa klik pada Konsol AWS Manajemen atau beberapa baris kode, Anda dapat membuat ruang kunci dan tabel di Amazon Keyspaces, tanpa menggunakan infrastruktur atau menginstal perangkat lunak apa pun.

Dengan Amazon Keyspaces, Anda dapat menjalankan beban kerja Cassandra yang ada AWS menggunakan kode aplikasi Cassandra dan alat pengembang yang sama yang Anda gunakan saat ini.

Dengan [kalkulator harga untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#) yang tersedia di Github, Anda dapat memperkirakan biaya bulanan Anda untuk Amazon Keyspaces berdasarkan beban kerja Apache Cassandra yang ada. Masukkan metrik dari keluaran status nodetool Cassandra Anda dan konfigurasi tanpa server yang dimaksudkan untuk Amazon Keyspaces untuk membandingkan biaya langsung antara kedua solusi. Perhatikan bahwa kalkulator ini hanya berfokus pada biaya operasional Amazon Keyspaces dibandingkan dengan penerapan Cassandra yang ada. Ini tidak termasuk faktor biaya kepemilikan total (TCO) seperti pemeliharaan infrastruktur, overhead operasional, atau biaya dukungan untuk Cassandra.

Untuk daftar titik akhir yang tersedia Wilayah AWS dan titik akhir, lihat [Titik akhir layanan untuk Amazon Keyspaces](#).

Kami menyarankan Anda memulai dengan membaca bagian berikut:

Topik

- [Amazon Keyspaces: Cara kerjanya](#)
- [Kasus penggunaan Amazon Keyspaces](#)
- [Apa itu Cassandra Query Language \(CQL\)?](#)

Amazon Keyspaces: Cara kerjanya

Amazon Keyspaces menghapus overhead administratif pengelolaan Cassandra. Untuk memahami alasannya, akan sangat membantu untuk memulai dengan arsitektur Cassandra dan kemudian membandingkannya dengan Amazon Keyspaces.

Topik

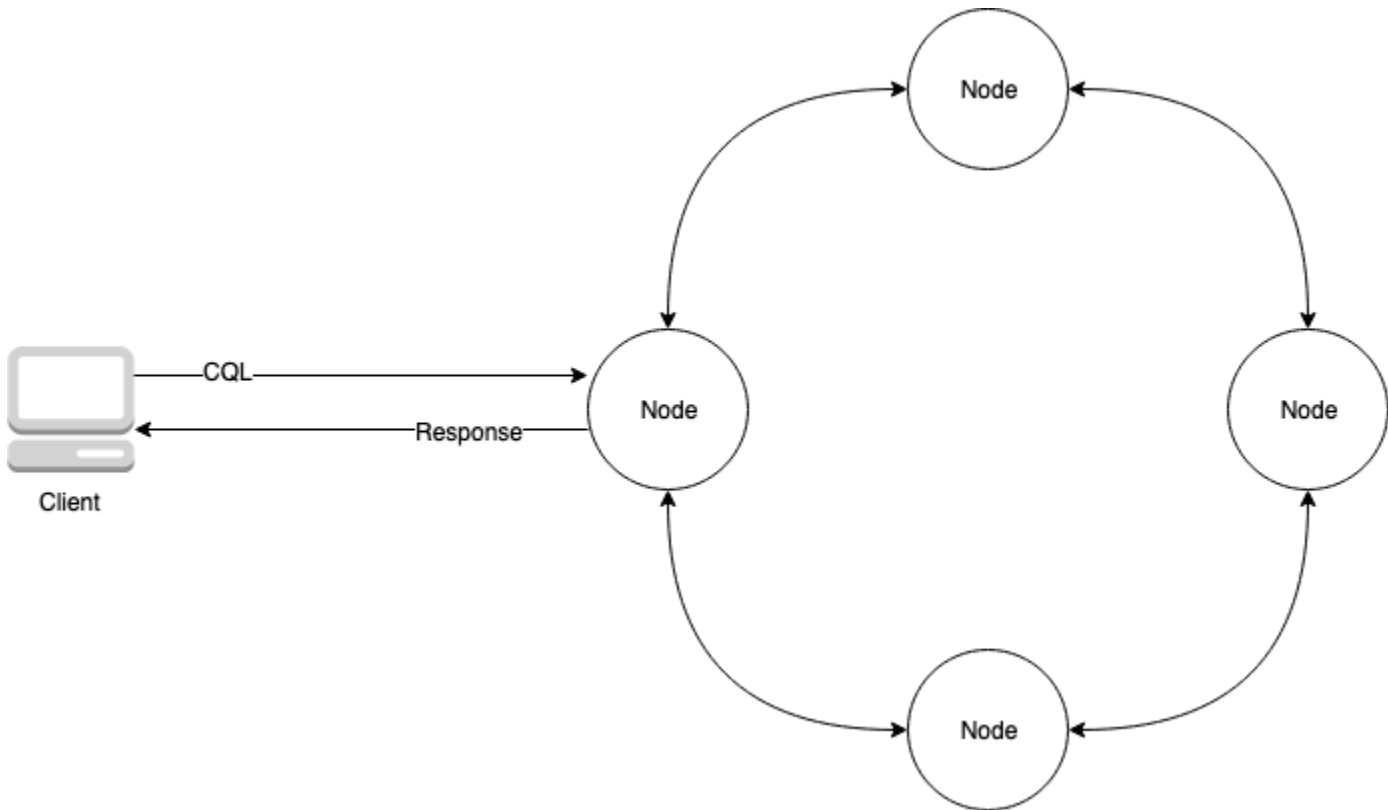
- [High-level arsitektur: Apache Cassandra vs Amazon Keyspaces](#)
- [Model data Cassandra](#)
- [Mengakses Amazon Keyspaces dari aplikasi](#)

High-level arsitektur: Apache Cassandra vs Amazon Keyspaces

Apache Cassandra tradisional digunakan dalam sebuah cluster yang terdiri dari satu atau lebih node. Anda bertanggung jawab untuk mengelola setiap node dan menambahkan serta menghapus node sebagai skala cluster Anda.

Program klien mengakses Cassandra dengan menghubungkan ke salah satu node dan mengeluarkan pernyataan Cassandra Query Language (CQL). CQL mirip dengan SQL, bahasa populer yang digunakan dalam database relasional. Meskipun Cassandra bukan database relasional, CQL menyediakan antarmuka yang akrab untuk query dan memanipulasi data di Cassandra.

Diagram berikut menunjukkan cluster Apache Cassandra sederhana, yang terdiri dari empat node.



Penyebaran Cassandra produksi mungkin terdiri dari ratusan node, berjalan pada ratusan komputer fisik di satu atau lebih pusat data fisik. Hal ini dapat menyebabkan beban operasional bagi pengembang aplikasi yang perlu menyediakan, menambal, dan mengelola server selain menginstal, memelihara, dan mengoperasikan perangkat lunak.

Dengan Amazon Keyspaces (untuk Apache Cassandra), Anda tidak perlu menyediakan, menambal, atau mengelola server, sehingga Anda dapat fokus membangun aplikasi yang lebih baik. Amazon Keyspaces menawarkan dua mode kapasitas throughput untuk membaca dan menulis: sesuai permintaan dan disediakan. Anda dapat memilih mode kapasitas throughput tabel Anda untuk mengoptimalkan harga pembacaan dan penulisan berdasarkan prediktabilitas dan variabilitas beban kerja Anda.

Dengan mode on-demand, Anda hanya membayar untuk membaca dan menulis bahwa aplikasi Anda benar-benar bekerja. Anda tidak perlu menentukan kapasitas throughput tabel Anda terlebih dahulu. Amazon Keyspaces mengakomodasi lalu lintas aplikasi Anda hampir secara instan saat naik atau turun, menjadikannya pilihan yang baik untuk aplikasi dengan lalu lintas yang tidak terduga.

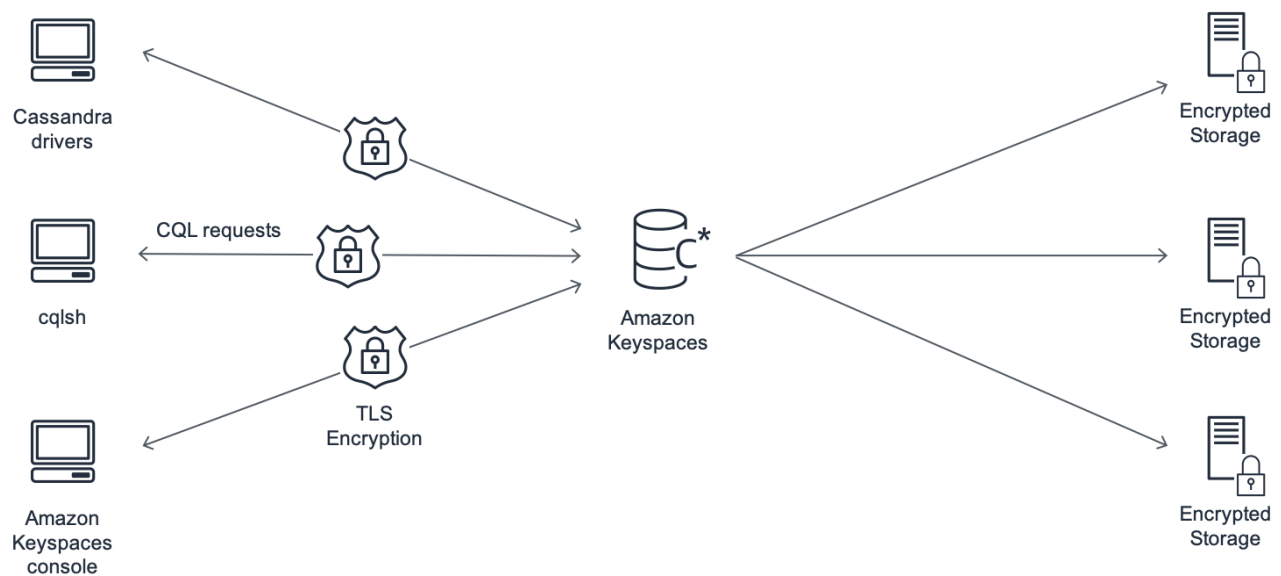
Mode kapasitas yang disediakan membantu Anda mengoptimalkan harga throughput jika Anda memiliki lalu lintas aplikasi yang dapat diprediksi dan dapat memperkirakan persyaratan kapasitas tabel Anda sebelumnya. Dengan mode kapasitas yang disediakan, Anda menentukan jumlah

pembacaan dan penulisan per detik yang Anda harapkan untuk dilakukan aplikasi Anda. [Anda dapat menambah dan mengurangi kapasitas yang disediakan untuk tabel Anda secara otomatis dengan mengaktifkan penskalaan otomatis.](#)

Anda dapat mengubah mode kapasitas tabel Anda sekali per hari saat Anda mempelajari lebih lanjut tentang pola lalu lintas beban kerja Anda, atau jika Anda berharap memiliki ledakan lalu lintas yang besar, seperti dari peristiwa besar yang Anda antisipasi akan mendorong banyak lalu lintas tabel. Untuk informasi selengkapnya tentang penyediaan kapasitas baca dan tulis, lihat. [the section called “Konfigurasi mode read/write kapasitas”](#)

Amazon Keyspaces (untuk Apache Cassandra) menyimpan tiga salinan data Anda di beberapa [Availability Zone untuk daya tahan dan ketersediaan tinggi](#). Selain itu, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan. Enkripsi saat istirahat diaktifkan secara otomatis saat Anda membuat tabel Amazon Keyspaces baru dan semua koneksi klien memerlukan Transport Layer Security (TLS). Fitur AWS keamanan tambahan termasuk [pemantauan AWS Identity and Access Management](#), dan titik akhir [virtual private cloud \(VPC\)](#). Untuk ikhtisar semua fitur keamanan yang tersedia, lihat [Keamanan](#).

Diagram berikut menunjukkan arsitektur Amazon Keyspaces.



Program klien mengakses Amazon Keyspaces dengan menghubungkan ke titik akhir yang telah ditentukan (nama host dan nomor port) dan mengeluarkan pernyataan CQL. Untuk daftar titik akhir yang tersedia, lihat [the section called “Titik akhir layanan”](#).

Model data Cassandra

Cara Anda memodelkan data untuk kasus bisnis Anda sangat penting untuk mencapai kinerja optimal dari Amazon Keyspaces. Model data yang buruk dapat menurunkan kinerja secara signifikan.

Meskipun CQL terlihat mirip dengan SQL, backend Cassandra dan database relasional sangat berbeda dan harus didekati secara berbeda. Berikut ini adalah beberapa masalah yang lebih signifikan untuk dipertimbangkan:

Penyimpanan

Anda dapat memvisualisasikan data Cassandra Anda dalam tabel, dengan setiap baris mewakili catatan dan setiap kolom bidang dalam catatan itu.

Desain tabel: Kueri terlebih dahulu

Tidak ada JOIN s di CQL. Oleh karena itu, Anda harus mendesain tabel Anda dengan bentuk data Anda dan bagaimana Anda perlu mengaksesnya untuk kasus penggunaan bisnis Anda. Ini dapat mengakibatkan de-normalisasi dengan data duplikat. Anda harus mendesain setiap tabel Anda secara khusus untuk pola akses tertentu.

Partisi

Data Anda disimpan dalam partisi pada disk. Jumlah partisi data Anda disimpan dan bagaimana itu didistribusikan di seluruh partisi ditentukan oleh kunci partisi Anda. Bagaimana Anda mendefinisikan kunci partisi Anda dapat memiliki dampak yang signifikan pada kinerja kueri Anda. Untuk praktik terbaik, lihat [the section called “Desain kunci partisi”](#).

Kunci utama

Di Cassandra, data disimpan sebagai pasangan kunci-nilai. Setiap tabel Cassandra harus memiliki kunci utama, yang merupakan kunci unik untuk setiap baris dalam tabel. Kunci utama adalah gabungan dari kunci partisi yang diperlukan dan kolom pengelompokan opsional. Data yang terdiri dari kunci utama harus unik di semua catatan dalam tabel.

- Kunci partisi — Bagian kunci partisi dari kunci utama diperlukan dan menentukan partisi mana dari cluster Anda tempat data disimpan. Kunci partisi dapat berupa kolom tunggal, atau dapat berupa nilai majemuk yang terdiri dari dua atau lebih kolom. Anda akan menggunakan kunci partisi majemuk jika kunci partisi kolom tunggal akan menghasilkan partisi tunggal atau sangat sedikit partisi yang memiliki sebagian besar data dan dengan demikian menanggung sebagian besar I/O operasi disk.

- Kolom pengelompokan - Bagian kolom pengelompokan opsional dari kunci utama Anda menentukan bagaimana data dikelompokkan dan diurutkan dalam setiap partisi. Jika Anda menyertakan kolom pengelompokan di kunci utama Anda, kolom pengelompokan dapat memiliki satu atau beberapa kolom. Jika ada beberapa kolom di kolom pengelompokan, urutan penyortiran ditentukan oleh urutan kolom yang tercantum di kolom pengelompokan, dari kiri ke kanan.

Untuk informasi selengkapnya tentang desain NoSQL dan Amazon Keyspaces, lihat [the section called “Desain NoSQL”](#) Untuk informasi selengkapnya tentang Amazon Keyspaces dan pemodelan data, lihat [the section called “Pemodelan data”](#)

Mengakses Amazon Keyspaces dari aplikasi

Amazon Keyspaces (untuk Apache Cassandra) mengimplementasikan API Apache Cassandra Query Language (CQL), sehingga Anda dapat menggunakan driver CQL dan Cassandra yang sudah Anda gunakan. Memperbarui aplikasi Anda semudah memperbarui driver atau `cqlsh` konfigurasi Cassandra Anda untuk menunjuk ke titik akhir layanan Amazon Keyspaces. Untuk informasi selengkapnya tentang kredensial yang diperlukan, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#)

Note

Untuk membantu Anda memulai, Anda dapat menemukan contoh kode ujung ke ujung untuk menghubungkan ke Amazon Keyspaces dengan menggunakan berbagai driver klien Cassandra di repositori contoh kode Amazon Keyspaces. [GitHub](#)

Pertimbangkan program Python berikut, yang menghubungkan ke cluster Cassandra dan menanyakan tabel.

```
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)
```

Untuk menjalankan program yang sama terhadap Amazon Keyspaces, Anda perlu:

- Tambahkan titik akhir dan port cluster: Misalnya, host dapat diganti dengan titik akhir layanan, seperti `cassandra.us-east-1.amazonaws.com` dan nomor port dengan: `9142`
- Tambahkan TLS/SSL konfigurasi: Untuk informasi selengkapnya tentang menambahkan TLS/SSL konfigurasi untuk terhubung ke Amazon Keyspaces dengan menggunakan driver Python klien Cassandra, lihat [Menggunakan driver klien Cassandra Python untuk mengakses Amazon Keyspaces secara terprogram](#)

Kasus penggunaan Amazon Keyspaces

Berikut ini adalah beberapa cara di mana Anda dapat menggunakan Amazon Keyspaces:

- Buat aplikasi yang membutuhkan latensi rendah — Memproses data dengan kecepatan tinggi untuk aplikasi yang memerlukan latensi milidetik satu digit, seperti pemeliharaan peralatan industri, pemantauan perdagangan, manajemen armada, dan optimalisasi rute.
- Membangun aplikasi menggunakan teknologi open-source — Bangun aplikasi AWS menggunakan API Cassandra open-source dan driver yang tersedia untuk berbagai bahasa pemrograman, seperti Java, Python, Ruby, Microsoft .NET, PHP, C ++, Node.js Perl, dan Go. Untuk contoh kode, lihat [Perpustakaan dan alat-alat](#).
- Pindahkan beban kerja Cassandra Anda ke cloud — Mengelola tabel Cassandra sendiri memakan waktu dan mahal. Dengan Amazon Keyspaces, Anda dapat mengatur, mengamankan, dan menskalakan tabel Cassandra tanpa mengelola infrastruktur. AWS Cloud Untuk informasi selengkapnya, lihat [Mengelola sumber daya tanpa server](#).

Apa itu Cassandra Query Language (CQL)?

Cassandra Query Language (CQL) adalah bahasa utama untuk berkomunikasi dengan Apache Cassandra. Amazon Keyspaces (untuk Apache Cassandra) kompatibel dengan CQL 3.x API (kompatibel dengan versi 2.x).

Di CQL, data disimpan dalam tabel, kolom, dan baris. Dalam pengertian ini CQL mirip dengan Structured Query Language (SQL). Ini adalah konsep kunci dalam CQL.

- Elemen CQL — Elemen dasar CQL adalah pengidentifikasi, konstanta, istilah, dan tipe data.
- Data Definition Language (DDL) — Pernyataan DDL digunakan untuk mengelola struktur data seperti keyspaces dan tabel, yang merupakan sumber daya AWS di Amazon Keyspaces. Pernyataan DDL adalah operasi pesawat kontrol di AWS.
- Data Manipulation Language (DML) — Pernyataan DML digunakan untuk mengelola data dalam tabel. Pernyataan DML digunakan untuk memilih, menyisipkan, memperbarui, dan menghapus data. Ini adalah operasi pesawat data di AWS.
- Built-in fungsi - Amazon Keyspaces mendukung berbagai fungsi skalar bawaan yang dapat Anda gunakan dalam pernyataan CQL.

Untuk informasi lebih lanjut tentang CQL, lihat. [Referensi bahasa CQL untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#) Untuk perbedaan fungsional dengan Apache Cassandra, lihat. [the section called “Perbedaan fungsional dengan Apache Cassandra”](#)

Untuk menjalankan kueri CQL, Anda dapat melakukan salah satu hal berikut:

- Gunakan editor CQL di. Konsol Manajemen AWS
- Gunakan AWS CloudShell dan ekspansi [cqlsh-](#).
- Gunakan `cqlsh` klien.
- Gunakan driver klien Cassandra berlisensi Apache 2.0.

Selain CQL, Anda dapat melakukan operasi Data Definition Language (DDL) di Amazon Keyspaces menggunakan SDK dan file. AWS AWS Command Line Interface

Untuk informasi selengkapnya tentang menggunakan metode ini untuk mengakses Amazon Keyspaces, lihat. [Mengakses Amazon Keyspaces \(untuk Apache Cassandra\)](#)

Bagaimana Amazon Keyspaces (untuk Apache Cassandra) dibandingkan Apache Cassandra?

[Untuk membuat sambungan ke Amazon Keyspaces, Anda dapat menggunakan titik akhir AWS layanan publik atau titik akhir pribadi menggunakan titik akhir VPC Antarmuka \(\) di Amazon AWS PrivateLink Virtual Private Cloud.](#) Bergantung pada titik akhir yang digunakan, Amazon Keyspaces dapat muncul ke klien dengan salah satu cara berikut.

AWS koneksi titik akhir layanan

Ini adalah koneksi yang dibuat di atas [titik akhir publik](#) mana pun. Dalam hal ini, Amazon Keyspaces muncul sebagai cluster Apache Cassandra 3.11.2 sembilan simpul ke klien.

Antarmuka koneksi titik akhir VPC

Ini adalah koneksi pribadi yang dibuat menggunakan titik akhir [VPC antarmuka](#). Dalam hal ini, Amazon Keyspaces muncul sebagai cluster Apache Cassandra 3.11.2 tiga simpul ke klien.

Terlepas dari jenis koneksi dan jumlah node yang terlihat oleh klien, Amazon Keyspaces menyediakan throughput dan penyimpanan yang hampir tak terbatas. Untuk melakukan ini, Amazon Keyspaces memetakan node untuk memuat penyeimbang yang merutekan kueri Anda ke salah satu dari banyak partisi penyimpanan yang mendasarinya. Untuk informasi selengkapnya tentang koneksi, lihat [the section called “Bagaimana mereka bekerja”](#).

Amazon Keyspaces menyimpan data dalam partisi. Partisi adalah alokasi penyimpanan untuk tabel, didukung oleh solid state drive (SSD). Amazon Keyspaces secara otomatis mereplikasi data Anda di beberapa [Availability Zone Wilayah AWS untuk daya tahan dan ketersediaan](#) tinggi. Seiring kebutuhan throughput atau penyimpanan Anda bertambah, Amazon Keyspaces menangani manajemen partisi untuk Anda dan secara otomatis menyediakan partisi tambahan yang diperlukan.

Amazon Keyspaces mendukung semua operasi bidang data Cassandra yang umum digunakan, seperti membuat ruang kunci dan tabel, membaca data, dan menulis data. Amazon Keyspaces [tanpa server](#), jadi Anda tidak perlu menyediakan, menambal, atau mengelola server. Anda juga tidak perlu menginstal, memelihara, atau mengoperasikan perangkat lunak. Akibatnya, di Amazon Keyspaces Anda tidak perlu menggunakan operasi API bidang kontrol Cassandra untuk mengelola pengaturan cluster dan node.

Amazon Keyspaces secara otomatis mengonfigurasi pengaturan seperti faktor replikasi dan tingkat konsistensi untuk memberi Anda ketersediaan tinggi, daya tahan, dan kinerja milidetik satu digit.

[Untuk ketahanan yang lebih tinggi dan pembacaan lokal latensi rendah, Amazon Keyspaces menawarkan replikasi Multi-wilayah.](#)

Topik

- [Perbedaan fungsional: Amazon Keyspaces vs Apache Cassandra](#)
- [API Cassandra yang didukung, operasi, fungsi, dan tipe data](#)
- [Mendukung Apache Cassandra membaca dan menulis tingkat konsistensi dan biaya terkait](#)

Perbedaan fungsional: Amazon Keyspaces vs Apache Cassandra

Berikut ini adalah perbedaan fungsional antara Amazon Keyspaces dan Apache Cassandra.

Topik

- [Apache Cassandra API, operasi, dan tipe data](#)
- [Pembuatan asinkron dan penghapusan ruang kunci dan tabel](#)
- [Autentikasi dan otorisasi](#)
- [Batch](#)
- [Change Data Capture \(CDC\)](#)
- [Konfigurasi kluster](#)
- [Koneksi](#)
- [INkata kunci](#)
- [FROZENkoleksi](#)
- [Transaksi ringan](#)
- [Penyeimbangan beban](#)
- [Paginasi](#)
- [Partisi](#)
- [Pernyataan yang disiapkan](#)
- [Hapus rentang](#)
- [Tabel sistem](#)
- [Stempel waktu](#)

- [User-defined jenis \(UDT\)](#)

Apache Cassandra API, operasi, dan tipe data

Amazon Keyspaces mendukung semua operasi bidang data Cassandra yang umum digunakan, seperti membuat ruang kunci dan tabel, membaca data, dan menulis data. Untuk melihat apa yang saat ini didukung, lihat [API Cassandra yang didukung, operasi, fungsi, dan tipe data](#).

Pembuatan asinkron dan penghapusan ruang kunci dan tabel

Amazon Keyspaces melakukan operasi data definition language (DDL), seperti membuat dan menghapus keyspaces, tabel, dan tipe secara asinkron. Untuk mempelajari cara memantau status pembuatan sumber daya, lihat [the section called “Periksa status pembuatan keyspace”](#) dan [the section called “Periksa status pembuatan tabel”](#). Untuk daftar pernyataan DDL dalam referensi bahasa CQL, lihat [the section called “Pernyataan DDL”](#)

Autentikasi dan otorisasi

Amazon Keyspaces (untuk Apache Cassandra) menggunakan AWS Identity and Access Management (IAM) untuk otentikasi dan otorisasi pengguna, dan mendukung kebijakan otorisasi yang setara dengan Apache Cassandra. Dengan demikian, Amazon Keyspaces tidak mendukung perintah konfigurasi keamanan Apache Cassandra.

Batch

Amazon Keyspaces mendukung perintah batch berikut:

- Batch log - mendukung hingga 100 perintah per batch.
- Batch yang tidak tercatat — mendukung hingga 30 perintah per batch.

Hanya INSERT, UPDATE, atau DELETE perintah yang diizinkan dalam batch.

Change Data Capture (CDC)

Di Apache Cassandra, ubah data capture (CDC) menyalin segmen log komit yang menangkap pembaruan tabel ke direktori tertentu pada node. Setiap catatan perubahan dalam log CDC dapat menyertakan pembaruan di beberapa ruang kunci dan tabel. Ketika baris diperbarui atau dihapus, log

Apache Cassandra CDC hanya menampilkan kolom yang dimodifikasi. Log tidak menyertakan status data sebelumnya sebelum perubahan. Untuk memahami detail lengkap dari modifikasi data, Anda perlu deserialisasi log ini menggunakan alat Cassandra tertentu.

Dengan Amazon Keyspaces CDC stream, Anda dapat memilih jenis informasi yang dikumpulkan oleh aliran CDC untuk setiap baris. Secara default, aliran CDC Amazon Keyspaces menangkap versi baris sebelum dan sesudah perubahan terjadi. Amazon Keyspaces CDC menyediakan catatan perubahan yang tidak digandakan dan diurutkan pada tingkat tabel.

[Di Amazon Keyspaces, setiap aliran CDC adalah AWS sumber daya dengan Nama Sumber Daya Amazon \(ARN\) dan aplikasi Anda dapat menggunakan aliran CDC Amazon Keyspaces menggunakan API CDC Amazon Keyspaces menggunakan salah satu titik akhir yang tersedia.](#)

Untuk informasi selengkapnya tentang Amazon Keyspaces CDC, lihat. [the section called “Bekerja dengan aliran CDC”](#)

Konfigurasi klaster

Amazon Keyspaces tanpa server, jadi tidak ada cluster, host, atau mesin virtual Java (JVM) untuk dikonfigurasi. Pengaturan Cassandra untuk pemadatan, kompresi, caching, pengumpulan sampah, dan penyaringan mekar tidak berlaku untuk Amazon Keyspaces dan diabaikan jika ditentukan.

Koneksi

Anda dapat menggunakan driver Cassandra yang ada untuk berkomunikasi dengan Amazon Keyspaces, tetapi Anda perlu mengonfigurasi driver secara berbeda. Amazon Keyspaces mendukung hingga 3.000 kueri CQL per koneksi TCP per detik, tetapi tidak ada batasan jumlah koneksi yang dapat dibuat oleh driver.

Sebagian besar driver Cassandra open-source membuat kumpulan koneksi ke Cassandra dan memuat kueri keseimbangan melalui kumpulan koneksi tersebut. Amazon Keyspaces mengekspos 9 alamat IP peer ke driver, dan perilaku default sebagian besar driver adalah membuat koneksi tunggal ke setiap alamat IP peer. Oleh karena itu, throughput kueri CQL maksimum dari driver yang menggunakan pengaturan default adalah 27.000 kueri CQL per detik.

Untuk meningkatkan jumlah ini, kami sarankan untuk meningkatkan jumlah koneksi per alamat IP yang dipertahankan driver Anda di kumpulan koneksinya. Misalnya, mengatur koneksi maksimum per alamat IP ke 2 menggandakan throughput maksimum driver Anda menjadi 54.000 kueri CQL per detik.

Sebagai praktik terbaik, kami merekomendasikan untuk mengonfigurasi driver untuk menggunakan 500 kueri CQL per detik per koneksi untuk memungkinkan overhead dan meningkatkan distribusi. Dalam skenario ini, perencanaan untuk 18.000 kueri CQL per detik membutuhkan 36 koneksi. Mengonfigurasi driver untuk 4 koneksi di 9 titik akhir menyediakan 36 koneksi yang melakukan 500 permintaan per detik. Untuk informasi selengkapnya tentang praktik terbaik untuk koneksi, lihat [the section called “Koneksi”](#).

Saat menghubungkan dengan titik akhir VPC, mungkin ada lebih sedikit titik akhir yang tersedia. Ini berarti Anda harus meningkatkan jumlah koneksi dalam konfigurasi driver. Untuk informasi selengkapnya tentang praktik terbaik untuk koneksi VPC, lihat [the section called “Koneksi titik akhir VPC”](#).

IN kata kunci

Amazon Keyspaces mendukung IN kata kunci dalam pernyataan. `SELECT IN` tidak didukung dengan `UPDATE` dan `DELETE`. Saat menggunakan IN kata kunci dalam `SELECT` pernyataan, hasil kueri dikembalikan dalam urutan bagaimana kunci disajikan dalam `SELECT` pernyataan. Di Cassandra, hasilnya diurutkan secara leksikografis.

Saat menggunakan `ORDER BY`, pemesanan ulang penuh dengan pagination dinonaktifkan tidak didukung dan hasilnya diurutkan dalam halaman. Kueri irisan tidak didukung dengan IN kata kunci. `TOKENS` tidak didukung dengan IN kata kunci. Amazon Keyspaces memproses kueri dengan IN kata kunci dengan membuat subkueri. Setiap subquery dihitung sebagai koneksi menuju 3.000 kueri CQL per koneksi TCP per batas detik. Untuk informasi selengkapnya, lihat [the section called “Gunakan IN SELECT”](#).

FROZEN koleksi

`FROZEN` kata kunci di Cassandra membuat serial beberapa komponen dari tipe data koleksi menjadi satu nilai abadi yang diperlakukan seperti file. `BLOB INSERT` dan `UPDATE` pernyataan menimpa seluruh koleksi.

Amazon Keyspaces mendukung hingga 8 tingkat bersarang untuk koleksi beku secara default. Untuk informasi selengkapnya, lihat [the section called “Kuota layanan Amazon Keyspaces”](#).

Amazon Keyspaces tidak mendukung perbandingan ketidaksetaraan yang menggunakan seluruh koleksi beku dalam kondisional atau pernyataan. `UPDATE SELECT` Perilaku untuk koleksi dan koleksi beku sama di Amazon Keyspaces.

Saat Anda menggunakan koleksi beku dengan stempel waktu sisi klien, jika stempel waktu operasi tulis sama dengan stempel waktu kolom yang ada yang tidak kedaluwarsa atau di-tombstoned, Amazon Keyspaces tidak melakukan perbandingan. Sebaliknya, ini memungkinkan server menentukan penulis terbaru, dan penulis terbaru menang.

Untuk informasi lebih lanjut tentang koleksi beku, lihat [the section called “Jenis koleksi”](#).

Transaksi ringan

Amazon Keyspaces (untuk Apache Cassandra) sepenuhnya mendukung membandingkan dan mengatur fungsionalitas pada `INSERT`, dan `DELETE` perintah `UPDATE`, yang dikenal sebagai transaksi ringan (LWT) di Apache Cassandra. Sebagai penawaran tanpa server, Amazon Keyspaces (untuk Apache Cassandra) memberikan kinerja yang konsisten pada skala apa pun, termasuk untuk transaksi ringan. Dengan Amazon Keyspaces, tidak ada penalti kinerja untuk menggunakan transaksi ringan.

Penyeimbangan beban

Entri `system.peers` tabel sesuai dengan penyeimbang beban Amazon Keyspaces. Untuk hasil terbaik, sebaiknya gunakan kebijakan penyeimbangan beban round robin dan menyetel jumlah koneksi per IP agar sesuai dengan kebutuhan aplikasi Anda.

Paginasi

Amazon Keyspaces mem-paginasi hasil berdasarkan jumlah baris yang dibaca untuk memproses permintaan, bukan jumlah baris yang dikembalikan dalam kumpulan hasil. Akibatnya, beberapa halaman mungkin berisi lebih sedikit baris daripada yang Anda tentukan dalam `UKURAN HALAMAN` untuk kueri yang difilter. Selain itu, Amazon Keyspaces melakukan paginasi hasil secara otomatis setelah membaca 1 MB data untuk memberikan kinerja pembacaan milidetik satu digit yang konsisten kepada pelanggan. Untuk informasi selengkapnya, lihat [the section called “Hasil paginate”](#).

Dalam tabel dengan kolom statis, Apache Cassandra dan Amazon Keyspaces menetapkan nilai kolom statis partisi di awal setiap halaman dalam kueri multi-halaman. Ketika tabel memiliki baris data yang besar, sebagai akibat dari perilaku pagination Amazon Keyspaces, kemungkinan lebih tinggi bahwa hasil operasi pembacaan rentang dapat mengembalikan lebih banyak halaman untuk Amazon Keyspaces daripada untuk Apache Cassandra. Akibatnya, ada kemungkinan lebih tinggi di Amazon Keyspaces bahwa pembaruan bersamaan ke kolom statis dapat mengakibatkan nilai kolom statis berbeda di halaman yang berbeda dari kumpulan hasil baca rentang.

Partisi

Partisi default di Amazon Keyspaces adalah `Cassandra-compatible Murmur3Partitioner`. Selain itu, Anda memiliki pilihan untuk menggunakan `Amazon Keyspaces DefaultPartitioner` atau `Cassandra-compatible RandomPartitioner`.

Dengan Amazon Keyspaces, Anda dapat dengan aman mengubah partisi untuk akun Anda tanpa harus memuat ulang data Amazon Keyspaces Anda. Setelah perubahan konfigurasi selesai, yang memakan waktu sekitar 10 menit, klien akan melihat pengaturan partisi baru secara otomatis saat berikutnya mereka terhubung. Untuk informasi selengkapnya, lihat [the section called “Bekerja dengan partisi”](#).

Pernyataan yang disiapkan

Amazon Keyspaces mendukung penggunaan pernyataan yang disiapkan untuk operasi bahasa manipulasi data (DHTML), seperti membaca dan menulis data. Amazon Keyspaces saat ini tidak mendukung penggunaan pernyataan siap untuk operasi bahasa definisi data (DDL), seperti membuat tabel dan ruang kunci. Operasi DDL harus dijalankan di luar pernyataan yang disiapkan.

Hapus rentang

Important

Mulai 1 Juli 2026, operasi penghapusan rentang akan bertransisi dari pemrosesan sinkron ke pemrosesan asinkron. Setelah perubahan ini, respons yang berhasil mengonfirmasi bahwa permintaan penghapusan Anda telah diterima, tetapi penghapusan mungkin masih berlangsung. Untuk memverifikasi bahwa penghapusan selesai, kueri item dalam rentang yang ditentukan. Ketika query tidak mengembalikan hasil, penghapusan selesai.

Amazon Keyspaces mendukung penghapusan baris dalam jangkauan. Rentang adalah serangkaian baris yang berdekatan di dalam partisi. Anda menentukan rentang dalam operasi `DELETE` menggunakan klausa `WHERE`. Anda dapat menentukan rentang menjadi seluruh partisi.

Selanjutnya, Anda dapat menentukan rentang untuk menjadi bagian dari baris yang berdekatan dalam partisi dengan menggunakan operator relasional (misalnya, `'>'`, `'<'`), atau dengan menyertakan kunci partisi dan menghilangkan satu atau lebih kolom pengelompokan. Dengan Amazon Keyspaces, Anda dapat menghapus hingga 1.000 baris dalam rentang dalam satu operasi.

Penghapusan rentang tidak terisolasi. Penghapusan baris individual dapat dilihat oleh operasi lain saat penghapusan rentang sedang dalam proses.

Tabel sistem

Amazon Keyspaces mengisi tabel sistem yang diperlukan oleh driver Cassandra open-source Apache 2.0. Tabel sistem yang terlihat oleh klien berisi informasi yang unik untuk pengguna yang diautentikasi. Tabel sistem sepenuhnya dikendalikan oleh Amazon Keyspaces dan hanya-baca. Untuk informasi selengkapnya, lihat [the section called “Ruang kunci sistem”](#).

Read-only akses ke tabel sistem diperlukan, dan Anda dapat mengontrolnya dengan kebijakan akses IAM. Untuk informasi selengkapnya, lihat [the section called “Mengelola akses menggunakan kebijakan”](#). Anda harus menentukan kebijakan kontrol akses berbasis tag untuk tabel sistem secara berbeda tergantung pada apakah Anda menggunakan panggilan API AWS SDK atau Cassandra Query Language (CQL) melalui driver Cassandra dan alat pengembang. Untuk mempelajari lebih lanjut tentang kontrol akses berbasis tag untuk tabel sistem, lihat [the section called “Akses sumber daya Amazon Keyspaces berdasarkan tag”](#).

Jika Anda mengakses Amazon Keyspaces menggunakan [titik akhir Amazon VPC](#), Anda akan melihat [entri dalam tabel untuk `system.peers` setiap titik akhir](#) Amazon VPC yang memiliki izin untuk dilihat Amazon Keyspaces. Akibatnya, driver Cassandra Anda mungkin mengeluarkan [pesan peringatan](#) tentang node kontrol itu sendiri di tabel `system.peers`. Anda dapat dengan aman mengabaikan peringatan ini.

Stempel waktu

Di Amazon Keyspaces, stempel waktu tingkat sel yang kompatibel dengan stempel waktu default di Apache Cassandra adalah fitur opt-in.

`USING TIMESTAMP` klausa dan `WRITETIME` fungsi hanya tersedia ketika stempel waktu sisi klien dihidupkan untuk sebuah tabel. Untuk mempelajari selengkapnya tentang stempel waktu sisi klien di Amazon Keyspaces, lihat [the section called “Client-side stempel waktu”](#).

User-defined jenis (UDT)

Operator ketidaksetaraan tidak didukung untuk UDT di Amazon Keyspaces.

Untuk mempelajari cara bekerja dengan UDT di Amazon Keyspaces, lihat [the section called “User-defined jenis \(UDT\)”](#).

Untuk meninjau berapa banyak UDT yang didukung per ruang kunci, tingkat penyarangan yang didukung, dan nilai dan kuota default lainnya yang terkait dengan UDT, lihat. [the section called “Kuota dan nilai default untuk tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces”](#)

API Cassandra yang didukung, operasi, fungsi, dan tipe data

Amazon Keyspaces (untuk Apache Cassandra) kompatibel dengan Cassandra Query Language (CQL) 3.11 API (kompatibel dengan versi 2.x).

Amazon Keyspaces mendukung semua operasi bidang data Cassandra yang umum digunakan, seperti membuat ruang kunci dan tabel, membaca data, dan menulis data.

Bagian berikut mencantumkan fungsionalitas yang didukung.

Topik

- [Dukungan Cassandra API](#)
- [Dukungan API bidang kontrol Cassandra](#)
- [Dukungan API pesawat data Cassandra](#)
- [Dukungan fungsi Cassandra](#)
- [Dukungan tipe data Cassandra](#)

Dukungan Cassandra API

Operasi API	Didukung
CREATE KEYSPACE	Ya
ALTER KEYSPACE	Ya
DROP KEYSPACE	Ya
CREATE TABLE	Ya
ALTER TABLE	Ya
DROP TABLE	Ya
CREATE INDEX	Tidak

Operasi API	Didukung
DROP INDEX	Tidak
UNLOGGED BATCH	Ya
LOGGED BATCH	Ya
SELECT	Ya
INSERT	Ya
DELETE	Ya
UPDATE	Ya
USE	Ya
CREATE TYPE	Ya
ALTER TYPE	Tidak
DROP TYPE	Ya
CREATE TRIGGER	Tidak
DROP TRIGGER	Tidak
CREATE FUNCTION	Tidak
DROP FUNCTION	Tidak
CREATE AGGREGATE	Tidak
DROP AGGREGATE	Tidak
CREATE MATERIALIZED VIEW	Tidak
ALTER MATERIALIZED VIEW	Tidak
DROP MATERIALIZED VIEW	Tidak

Operasi API	Didukung
TRUNCATE	Tidak

Dukungan API bidang kontrol Cassandra

Karena Amazon Keyspaces dikelola, operasi API bidang kontrol Cassandra untuk mengelola pengaturan cluster dan node tidak diperlukan. Akibatnya, fitur Cassandra berikut tidak berlaku.

Fitur	Alasan
Toggle tulis tahan lama	Semua tulisan tahan lama
Baca pengaturan perbaikan	Tidak berlaku
Detik rahmat GC	Tidak berlaku
Pengaturan filter Bloom	Tidak berlaku
Pengaturan pemadatan	Tidak berlaku
Pengaturan kompresi	Tidak berlaku
Pengaturan caching	Tidak berlaku
Pengaturan keamanan	Digantikan oleh IAM

Dukungan API pesawat data Cassandra

Fitur	Didukung
Dukungan JSON untuk pernyataan SELECT dan INSERT	Ya
Kolom statis	Ya
Waktu untuk Hidup (TTL)	Ya

Dukungan fungsi Cassandra

Untuk informasi selengkapnya tentang fungsi yang didukung, lihat [the section called “Built-in fungsi”](#).

Fungsi	Didukung
Fungsi Aggregate	Tidak
Blobkonversi	Ya
Cast	Ya
Fungsi Datetime	Ya
Fungsi Timeconversion	Ya
Fungsi TimeUuid	Ya
Token	Ya
User defined functions (UDF)	Tidak
Uuid	Ya

Dukungan tipe data Cassandra

Tabel berikut mencantumkan tipe data Apache Cassandra yang didukung di Amazon Keyspaces. Untuk informasi selengkapnya tentang tipe data di Amazon Keyspaces, lihat [the section called “Jenis Data”](#)

Jenis data	Didukung	
ascii	Ya	
bigint	Ya	
blob	Ya	
boolean	Ya	

Jenis data	Didukung	
counter	Ya	
date	Ya	
decimal	Ya	
double	Ya	
float	Ya	
frozen	Ya	
inet	Ya	
int	Ya	
list	Ya	
map	Ya	
set	Ya	
smallint	Ya	
text	Ya	
time	Ya	
timestamp	Ya	
timeuuid	Ya	
tinyint	Ya	
tuple	Ya	
user-defined types (UDTs)	Ya	
uuid	Ya	

Jenis data	Didukung	
varchar	Ya	
varint	Ya	

Mendukung Apache Cassandra membaca dan menulis tingkat konsistensi dan biaya terkait

Topik di bagian ini menjelaskan tingkat konsistensi Apache Cassandra mana yang didukung untuk operasi baca dan tulis di Amazon Keyspaces (untuk Apache Cassandra).

Topik

- [Tulis tingkat konsistensi](#)
- [Baca tingkat konsistensi](#)
- [Tingkat konsistensi yang tidak didukung](#)

Tulis tingkat konsistensi

Amazon Keyspaces mereplikasi semua operasi penulisan tiga kali di beberapa Availability Zone untuk daya tahan dan ketersediaan tinggi. Tulisan disimpan dengan tahan lama sebelum diakui menggunakan tingkat LOCAL_QUORUM konsistensi. Untuk setiap penulisan 1 KB, Anda ditagih 1 unit kapasitas tulis (WCU) untuk tabel menggunakan mode kapasitas yang disediakan atau 1 unit permintaan tulis (WRU) untuk tabel menggunakan mode sesuai permintaan.

Anda dapat menggunakan `cqlsh` untuk mengatur konsistensi untuk semua kueri dalam sesi saat ini untuk LOCAL_QUORUM menggunakan kode berikut.

```
CONSISTENCY LOCAL_QUORUM;
```

Untuk mengkonfigurasi tingkat konsistensi secara terprogram, Anda dapat mengatur konsistensi dengan driver klien Cassandra yang sesuai. Misalnya, driver Java versi 4.x memungkinkan Anda untuk mengatur tingkat konsistensi dalam `app.config` file seperti yang ditunjukkan di bawah ini.

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

Jika Anda menggunakan driver Java Cassandra versi 3.x, Anda dapat menentukan tingkat konsistensi untuk sesi dengan menambahkan `.withQueryOptions(new QueryOptions().setConsistencyLevel(ConsistencyLevel.LOCAL_QUORUM))` seperti yang ditunjukkan pada contoh kode berikut.

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

Untuk mengkonfigurasi tingkat konsistensi untuk operasi penulisan tertentu, Anda dapat menentukan konsistensi ketika Anda memanggil `QueryBuilder.insertInto` dengan `setConsistencyLevel` argumen ketika Anda menggunakan driver Java.

Baca tingkat konsistensi

Amazon Keyspaces mendukung tiga tingkat konsistensi baca: `ONE`, `LOCAL_ONE`, dan `LOCAL_QUORUM`. Selama `LOCAL_QUORUM` membaca, Amazon Keyspaces mengembalikan respons yang mencerminkan pembaruan terbaru dari semua operasi penulisan yang berhasil sebelumnya. Menggunakan tingkat konsistensi `ONE` atau `LOCAL_ONE` dapat meningkatkan kinerja dan ketersediaan permintaan baca Anda, tetapi respons mungkin tidak mencerminkan hasil penulisan yang baru saja selesai.

Untuk setiap 4 KB membaca menggunakan `ONE` atau `LOCAL_ONE` konsistensi, Anda ditagih 0,5 unit kapasitas baca (RCU) untuk tabel menggunakan mode kapasitas yang disediakan atau 0,5 unit permintaan baca (RRU) untuk tabel menggunakan mode sesuai permintaan. Untuk setiap pembacaan 4 KB menggunakan `LOCAL_QUORUM` konsistensi, Anda ditagih 1 unit kapasitas baca (RCU) untuk tabel menggunakan mode kapasitas yang disediakan atau 1 unit permintaan baca (RRU) untuk tabel menggunakan mode sesuai permintaan.

Penagihan berdasarkan konsistensi baca dan mode throughput kapasitas baca per tabel untuk setiap 4 KB pembacaan

Tingkat konsistensi	Disediakan	On-demand
ONE	0,5 RCU	0,5 RRU
LOCAL_ONE	0,5 RCU	0,5 RRU
LOCAL_QUORUM	1 RCU	1 RRU

Untuk menentukan konsistensi yang berbeda untuk operasi baca, panggil `QueryBuilder.select` dengan `setConsistencyLevel` argumen saat Anda menggunakan driver Java.

Tingkat konsistensi yang tidak didukung

Tingkat konsistensi berikut tidak didukung oleh Amazon Keyspaces dan akan menghasilkan pengecualian.

Tingkat konsistensi yang tidak didukung

Apache Cassandra	Amazon Keyspaces
EACH_QUORUM	Tidak didukung
QUORUM	Tidak didukung
ALL	Tidak didukung
TWO	Tidak didukung
THREE	Tidak didukung
ANY	Tidak didukung
SERIAL	Tidak didukung
LOCAL_SERIAL	Tidak didukung

Bermigrasi ke Amazon Keyspaces (untuk Apache Cassandra)

Migrasi ke Amazon Keyspaces (untuk Apache Cassandra) menghadirkan berbagai manfaat menarik bagi bisnis dan organisasi. Berikut adalah beberapa keuntungan utama yang menjadikan Amazon Keyspaces pilihan yang menarik untuk migrasi.

- **Skalabilitas** — Amazon Keyspaces dirancang untuk menangani beban kerja yang besar dan menskalakan dengan mulus untuk mengakomodasi volume dan lalu lintas data yang terus bertambah. Dengan Cassandra tradisional, penskalaan tidak dilakukan sesuai permintaan dan membutuhkan perencanaan untuk puncak masa depan. Dengan Amazon Keyspaces, Anda dapat dengan mudah menskalakan tabel Anda ke atas atau ke bawah berdasarkan permintaan, memastikan bahwa aplikasi Anda dapat menangani lonjakan lalu lintas yang tiba-tiba tanpa mengorbankan kinerja.
- **Kinerja** - Amazon Keyspaces menawarkan akses data latensi rendah, memungkinkan aplikasi untuk mengambil dan memproses data dengan kecepatan luar biasa. Arsitektur terdistribusinya memastikan operasi baca dan tulis didistribusikan di beberapa simpul sehingga memberikan waktu respons milidetik satu digit yang konsisten bahkan pada tingkat permintaan tinggi.
- **Dikelola sepenuhnya** - Amazon Keyspaces adalah layanan yang dikelola sepenuhnya yang disediakan oleh AWS. Ini berarti AWS menangani aspek operasional manajemen basis data, termasuk penyediaan, konfigurasi, penambalan, pencadangan, dan penskalaan. Ini memungkinkan Anda untuk lebih fokus pada pengembangan aplikasi dan tidak terlalu fokus pada tugas administrasi basis data.
- **Arsitektur tanpa server** - Amazon Keyspaces tanpa server. Anda hanya membayar untuk kapasitas yang dikonsumsi tanpa memerlukan penyediaan kapasitas di muka. Anda tidak memiliki server untuk dikelola atau instance untuk dipilih. Model bayar per permintaan ini menawarkan efisiensi biaya dan overhead operasional minimal, karena Anda hanya membayar sumber daya yang Anda gunakan tanpa perlu menyediakan dan memantau kapasitas.
- **Fleksibilitas NoSQL dengan skema** - Amazon Keyspaces mengikuti model data NoSQL, memberikan fleksibilitas dalam desain skema. Dengan Amazon Keyspaces, Anda dapat menyimpan data terstruktur, semi-terstruktur, dan tidak terstruktur, sehingga cocok untuk menangani beragam tipe data yang berkembang. Selain itu, Amazon Keyspaces melakukan validasi skema pada penulisan yang memungkinkan evolusi terpusat dari model data. Fleksibilitas ini memungkinkan siklus pengembangan yang lebih cepat dan adaptasi yang lebih mudah terhadap perubahan kebutuhan bisnis.

- Ketersediaan dan daya tahan tinggi — Amazon Keyspaces mereplikasi data di beberapa [Availability Zone](#) dalam satu Wilayah AWS, memastikan ketersediaan tinggi dan daya tahan data. Replikasi, failover, dan pemulihan direplikasi secara otomatis, sehingga meminimalkan risiko kehilangan data atau gangguan layanan. Amazon Keyspaces menyediakan ketersediaan SLA hingga 99,999%. [Untuk ketahanan yang lebih tinggi dan pembacaan lokal latensi rendah, Amazon Keyspaces menawarkan replikasi Multi-wilayah.](#)
- Keamanan dan kepatuhan — Amazon Keyspaces terintegrasi dengan AWS Identity and Access Management kontrol akses berbutir halus. Ini menyediakan enkripsi saat istirahat dan dalam perjalanan, membantu meningkatkan keamanan data Anda. Amazon Keyspaces telah dinilai oleh auditor pihak ketiga untuk keamanan dan kepatuhan terhadap program tertentu, termasuk HIPAA, PCI DSS, dan SOC, yang memungkinkan Anda memenuhi persyaratan peraturan. Untuk informasi selengkapnya, lihat [the section called “Validasi kepatuhan”](#).
- Integrasi dengan AWS Ekosistem — Sebagai bagian dari AWS ekosistem, Amazon Keyspaces terintegrasi dengan mulus dengan yang lain Layanan AWS, misalnya, AWS CloudFormation Amazon, dan. CloudWatch AWS CloudTrail Integrasi ini memungkinkan Anda membangun arsitektur nirserver, memanfaatkan infrastruktur sebagai kode, dan membuat aplikasi berbasis data waktu nyata. Lihat informasi yang lebih lengkap di [Memantau Amazon Keyspaces](#).

Topik

- [Buat rencana migrasi untuk migrasi dari Apache Cassandra ke Amazon Keyspaces](#)
- [Cara memilih alat yang tepat untuk mengunggah massal atau memigrasi data ke Amazon Keyspaces](#)

Buat rencana migrasi untuk migrasi dari Apache Cassandra ke Amazon Keyspaces

Agar migrasi berhasil dari Apache Cassandra ke Amazon Keyspaces, kami merekomendasikan peninjauan konsep migrasi yang berlaku dan praktik terbaik serta perbandingan opsi yang tersedia.

Topik ini menguraikan cara kerja proses migrasi dengan memperkenalkan beberapa konsep kunci dan alat serta teknik yang tersedia untuk Anda. Anda dapat mengevaluasi berbagai strategi migrasi untuk memilih salah satu yang paling sesuai dengan kebutuhan Anda.

Topik

- [Kompatibilitas fungsional](#)

- [Perkirakan harga Amazon Keyspaces](#)
- [Pilih strategi migrasi](#)
- [Migrasi online ke Amazon Keyspaces: strategi dan praktik terbaik](#)
- [Proses migrasi offline: Apache Cassandra ke Amazon Keyspaces](#)
- [Menggunakan solusi migrasi hibrida: Apache Cassandra ke Amazon Keyspaces](#)

Kompatibilitas fungsional

Pertimbangkan perbedaan fungsional antara Apache Cassandra dan Amazon Keyspaces dengan hati-hati sebelum migrasi. Amazon Keyspaces mendukung semua operasi bidang data Cassandra yang umum digunakan, seperti membuat ruang kunci dan tabel, membaca data, dan menulis data.

Namun ada beberapa API Cassandra yang tidak didukung Amazon Keyspaces. Untuk informasi selengkapnya tentang API yang didukung, lihat [the section called “API Cassandra yang didukung, operasi, fungsi, dan tipe data”](#). Untuk gambaran umum tentang semua perbedaan fungsional antara Amazon Keyspaces dan Apache Cassandra, lihat [the section called “Perbedaan fungsional dengan Apache Cassandra”](#)

Untuk membandingkan API dan skema Cassandra yang Anda gunakan dengan fungsionalitas yang didukung di Amazon Keyspaces, Anda dapat menjalankan skrip kompatibilitas yang tersedia di toolkit Amazon Keyspaces. [GitHub](#)

Cara menggunakan skrip kompatibilitas

1. Unduh skrip Python kompatibilitas dari [GitHub](#) dan pindahkan ke lokasi yang memiliki akses ke cluster Apache Cassandra Anda yang ada.
2. Skrip kompatibilitas menggunakan parameter yang sama seperti CQLSH. Untuk `--host` dan `--port` masukkan alamat IP dan port yang Anda gunakan untuk menghubungkan dan menjalankan kueri ke salah satu node Cassandra di cluster Anda.

Jika cluster Cassandra Anda menggunakan otentikasi, Anda juga perlu menyediakan `-username` dan `-password`. Untuk menjalankan skrip kompatibilitas, Anda dapat menggunakan perintah berikut.

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

Perkirakan harga Amazon Keyspaces

Bagian ini memberikan gambaran umum tentang informasi yang perlu Anda kumpulkan dari tabel Apache Cassandra Anda untuk menghitung perkiraan biaya Amazon Keyspaces. Setiap tabel Anda memerlukan tipe data yang berbeda, perlu mendukung kueri CQL yang berbeda, dan mempertahankan lalu lintas yang berbeda. read/write

[Memikirkan kebutuhan Anda berdasarkan tabel sejalan dengan isolasi read/write sumber daya tingkat tabel Amazon Keyspaces dan mode kapasitas throughput](#). Dengan Amazon Keyspaces, Anda dapat menentukan read/write kapasitas dan [kebijakan penskalaan otomatis](#) untuk tabel secara independen.

Memahami persyaratan tabel membantu Anda memprioritaskan tabel untuk migrasi berdasarkan fungsionalitas, biaya, dan upaya migrasi.

Kumpulkan metrik tabel Cassandra berikut sebelum migrasi. Informasi ini membantu memperkirakan biaya beban kerja Anda di Amazon Keyspaces.

- Nama tabel - Nama ruang kunci dan nama tabel yang sepenuhnya memenuhi syarat.
- Deskripsi — Deskripsi tabel, misalnya bagaimana itu digunakan, atau jenis data apa yang disimpan di dalamnya.
- Pembacaan rata-rata per detik — Jumlah rata-rata pembacaan tingkat koordinat terhadap tabel selama interval waktu yang besar.
- Rata-rata menulis per detik — Jumlah rata-rata tingkat koordinat menulis terhadap tabel selama interval waktu yang besar.
- Ukuran baris rata-rata dalam byte - Ukuran baris rata-rata dalam byte.
- Ukuran penyimpanan dalam GB — Ukuran penyimpanan mentah untuk sebuah meja.
- Rincian konsistensi baca — Persentase pembacaan yang menggunakan konsistensi akhirnya (LOCAL_ONE atau ONE) vs. konsistensi kuat (LOCAL_QUORUM).

Tabel ini menunjukkan contoh informasi tentang tabel yang perlu Anda kumpulkan saat merencanakan migrasi.

Nama tabel	Deskripsi	Rata-rata membaca per detik	Rata-rata menulis per detik	Ukuran baris rata-rata dalam byte	Ukuran penyimpanan dalam GB	Baca rincian konsistensi
mykeyspace.mytable	Digunakan untuk menyimpan sejarah keranjang belanja	10.000	5.000	2.200	2.000	100% LOCAL_ONE
mykeyspace.mytable2	Digunakan untuk menyimpan informasi profil terbaru	20.000	1.000	850	1.000	25% LOCAL_QUORUM 75% LOCAL_ONE

Cara mengumpulkan metrik tabel

Bagian ini memberikan petunjuk langkah demi langkah tentang cara mengumpulkan metrik tabel yang diperlukan dari cluster Cassandra Anda yang ada. Metrik ini mencakup ukuran baris, ukuran tabel, dan read/write permintaan per detik (RPS). Mereka memungkinkan Anda menilai persyaratan kapasitas throughput untuk tabel Amazon Keyspaces dan memperkirakan harga.

Cara mengumpulkan metrik tabel di tabel sumber Cassandra

1. Tentukan ukuran baris

Ukuran baris penting untuk menentukan kapasitas baca dan pemanfaatan kapasitas tulis di Amazon Keyspaces. Diagram berikut menunjukkan distribusi data tipikal pada rentang token Cassandra.



Anda dapat menggunakan skrip sampler ukuran baris yang tersedia [GitHub](#) untuk mengumpulkan metrik ukuran baris untuk setiap tabel di cluster Cassandra Anda.

Skrip mengeksport data tabel dari Apache Cassandra dengan menggunakan `cqlsh` dan `awk` menghitung min, maks, rata-rata, dan standar deviasi ukuran baris atas kumpulan sampel data tabel yang dapat dikonfigurasi. Sampler ukuran baris meneruskan argumen `kecqlsh`, sehingga parameter yang sama dapat digunakan untuk menghubungkan dan membaca dari cluster Cassandra Anda.

Pernyataan berikut adalah contoh dari ini.

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

Untuk informasi selengkapnya tentang cara menghitung ukuran baris di Amazon Keyspaces, lihat [the section called “Perkiraan ukuran baris”](#)

2. Tentukan ukuran tabel

Dengan Amazon Keyspaces, Anda tidak perlu menyediakan penyimpanan terlebih dahulu. Amazon Keyspaces memantau ukuran tabel yang dapat ditagih secara terus menerus untuk menentukan biaya penyimpanan Anda. Penyimpanan ditagih per. GB-month Ukuran tabel Amazon Keyspaces didasarkan pada ukuran mentah (tidak terkompresi) dari satu replika.

Untuk memantau ukuran tabel di Amazon Keyspaces, Anda dapat menggunakan metrik `BillableTableSizeInBytes`, yang ditampilkan untuk setiap tabel di. Konsol Manajemen AWS

Untuk memperkirakan ukuran tabel Amazon Keyspaces yang dapat ditagih, Anda dapat menggunakan salah satu dari dua metode ini:

- Gunakan ukuran baris rata-rata dan kalikan dengan angka atau baris.

Anda dapat memperkirakan ukuran tabel Amazon Keyspaces dengan mengalikan ukuran baris rata-rata dengan jumlah baris dari tabel sumber Cassandra Anda. Gunakan skrip sampel ukuran baris dari bagian sebelumnya untuk menangkap ukuran baris rata-rata. Untuk menangkap jumlah baris, Anda dapat menggunakan alat seperti `dsbulk count` untuk menentukan jumlah total baris di tabel sumber Anda.

- Gunakan metadata tabel `nodetool` untuk mengumpulkan.

`Nodetool` adalah alat administrasi yang disediakan dalam distribusi Apache Cassandra yang memberikan wawasan tentang keadaan proses Cassandra dan mengembalikan metadata tabel. Anda dapat menggunakan `nodetool` untuk mengambil sampel metadata tentang ukuran tabel dan dengan itu mengekstrapolasi ukuran tabel di Amazon Keyspaces.

Perintah yang harus digunakan adalah `nodetool tablestats`. `Tablestats` mengembalikan ukuran tabel dan rasio kompresi. Ukuran tabel disimpan sebagai `tablelivespace` untuk tabel dan Anda dapat membaginya dengan `compression ratio`. Kemudian gandakan nilai ukuran ini dengan jumlah node. Akhirnya bagi dengan faktor replikasi (biasanya tiga).

Ini adalah rumus lengkap untuk perhitungan yang dapat Anda gunakan untuk menilai ukuran tabel.

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

Mari kita asumsikan bahwa cluster Cassandra Anda memiliki 12 node. Menjalankan `nodetool tablestats` perintah mengembalikan 200 GB dan `compression ratio` 0,5. `tablelivespace` Ruang kunci memiliki faktor replikasi tiga.

Seperti inilah perhitungan untuk contoh ini.

```
(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. Menangkap jumlah membaca dan menulis

Untuk menentukan persyaratan kapasitas dan penskalaan untuk tabel Amazon Keyspaces Anda, tangkap tingkat permintaan baca dan tulis tabel Cassandra Anda sebelum migrasi.

Amazon Keyspaces tanpa server dan Anda hanya membayar untuk apa yang Anda gunakan. Secara umum, harga read/write throughput di Amazon Keyspaces didasarkan pada jumlah dan ukuran permintaan.

Ada dua mode kapasitas di Amazon Keyspaces:

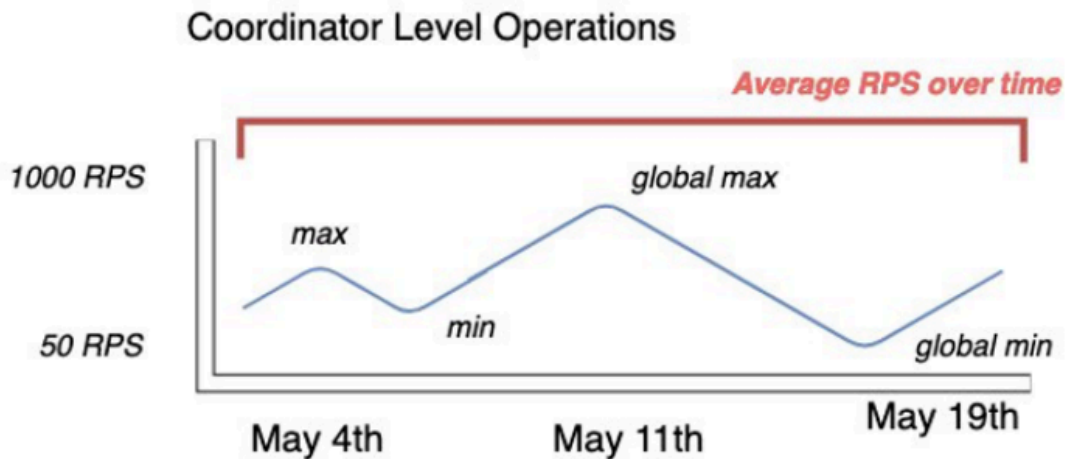
- [On-demand](#)— Ini adalah opsi penagihan fleksibel yang mampu melayani ribuan permintaan per detik tanpa perlu perencanaan kapasitas. Ini menawarkan harga bayar per permintaan untuk permintaan baca dan tulis sehingga Anda hanya membayar untuk apa yang Anda gunakan.
- [Disediakan](#) — Jika Anda memilih mode kapasitas throughput yang disediakan, Anda menentukan jumlah pembacaan dan penulisan per detik yang diperlukan untuk aplikasi Anda. Ini membantu Anda mengelola penggunaan Amazon Keyspaces agar tetap pada atau di bawah tingkat permintaan yang ditentukan untuk mempertahankan prediktabilitas.

Mode yang disediakan menawarkan [penskalaan otomatis](#) untuk secara otomatis menyesuaikan tarif yang disediakan untuk meningkatkan atau menurunkan skala guna meningkatkan efisiensi operasional. Untuk informasi selengkapnya tentang manajemen sumber daya tanpa server, lihat. [Mengelola sumber daya tanpa server](#)

Karena Anda menyediakan kapasitas throughput baca dan tulis di Amazon Keyspaces secara terpisah, Anda perlu mengukur tingkat permintaan untuk membaca dan menulis di tabel yang ada secara independen.

Untuk mengumpulkan metrik pemanfaatan paling akurat dari cluster Cassandra yang ada, tangkap permintaan rata-rata per detik (RPS) untuk operasi baca dan tulis tingkat koordinator selama periode waktu yang lama untuk tabel yang dikumpulkan di semua node dalam satu pusat data.

Menangkap RPS rata-rata selama periode setidaknya beberapa minggu menangkap puncak dan lembah dalam pola lalu lintas Anda, seperti yang ditunjukkan pada diagram berikut.



Anda memiliki dua opsi untuk menentukan tingkat permintaan baca dan tulis dari tabel Cassandra Anda.

- Gunakan pemantauan Cassandra yang ada

Anda dapat menggunakan metrik yang ditampilkan dalam tabel berikut untuk mengamati permintaan baca dan tulis. Perhatikan bahwa nama metrik dapat berubah berdasarkan alat pemantauan yang Anda gunakan.

Dimensi	Cassandra JMX metrik
Menulis	<code>org.apache.cassandra.metrics:type=ClientRequest,scope=Write,name=Latency#Count</code>
Membaca	<code>org.apache.cassandra.metrics:type=ClientRequest,scope=Read,name=Latency#Count</code>

- Gunakan `nodetool`

Gunakan `nodetool tablestats` dan `nodetool info` untuk menangkap rata-rata operasi baca dan tulis dari tabel. `tablestats` mengembalikan total jumlah baca dan tulis dari waktu node telah dimulai. `nodetool info` menyediakan up-time untuk node dalam hitungan detik.

Untuk menerima rata-rata per detik membaca dan menulis, bagilah jumlah baca dan tulis dengan waktu aktif node dalam hitungan detik. Kemudian, untuk pembacaan Anda membagi dengan iklan tingkat konsistensi untuk penulisan yang Anda bagi dengan faktor replikasi. Perhitungan ini dinyatakan dalam rumus berikut.

Rumus untuk pembacaan rata-rata per detik:

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

Rumus untuk penulisan rata-rata per detik:

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

Mari kita asumsikan kita memiliki 12 node cluster yang telah naik selama 4 minggu. `nodetool infomengembalikan` 2.419.200 detik up-time dan `nodetool tablestats` mengembalikan 1 miliar penulisan dan 2 miliar pembacaan. Contoh ini akan menghasilkan perhitungan berikut.

```
((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. Tentukan pemanfaatan kapasitas tabel

Untuk memperkirakan pemanfaatan kapasitas rata-rata, mulailah dengan tingkat permintaan rata-rata dan ukuran baris rata-rata tabel sumber Cassandra Anda.

Amazon Keyspaces menggunakan unit kapasitas baca (RCU) dan unit kapasitas tulis (WCU) untuk mengukur kapasitas throughput yang disediakan untuk membaca dan menulis untuk tabel. Untuk perkiraan ini, kami menggunakan unit ini untuk menghitung kebutuhan kapasitas baca dan tulis dari tabel Amazon Keyspaces baru setelah migrasi.

Nanti dalam topik ini kita akan membahas bagaimana pilihan antara mode kapasitas yang disediakan dan sesuai permintaan memengaruhi penagihan. Tetapi untuk perkiraan pemanfaatan kapasitas dalam contoh ini, kami berasumsi bahwa tabel dalam mode yang disediakan.

- Membaca - Satu RCU mewakili satu permintaan LOCAL_QUORUM baca, atau dua permintaan LOCAL_ONE baca, untuk satu baris hingga 4 KB. Jika Anda perlu membaca baris yang lebih besar dari 4 KB, operasi baca menggunakan RCU tambahan. Jumlah total RCU yang diperlukan tergantung pada ukuran baris, dan apakah Anda ingin menggunakan LOCAL_QUORUM atau LOCAL_ONE membaca konsistensi.

Misalnya, membaca baris 8 KB membutuhkan 2 RCU menggunakan konsistensi LOCAL_QUORUM baca, dan 1 RCU jika Anda memilih konsistensi LOCAL_ONE baca.

- Menulis — Satu WCU mewakili satu tulis untuk satu baris hingga 1 KB dalam ukuran. Semua penulisan menggunakan LOCAL_QUORUM konsistensi, dan tidak ada biaya tambahan untuk menggunakan transaksi ringan (LWT).

Jumlah total WCU yang dibutuhkan tergantung pada ukuran baris. Jika Anda perlu menulis baris yang lebih besar dari 1 KB, operasi tulis menggunakan WCU tambahan. Misalnya, jika ukuran baris Anda adalah 2 KB, Anda memerlukan 2 WCU untuk melakukan satu permintaan tulis.

Rumus berikut dapat digunakan untuk memperkirakan RCU dan WCU yang diperlukan.

- Kapasitas baca dalam RCU dapat ditentukan dengan mengalikan pembacaan per detik dengan jumlah baris yang dibaca per bacaan dikalikan dengan ukuran baris rata-rata dibagi dengan 4KB dan dibulatkan ke atas ke bilangan bulat terdekat.
- Kapasitas tulis di WCU dapat ditentukan dengan mengalikan jumlah permintaan dengan ukuran baris rata-rata dibagi dengan 1KB dan dibulatkan ke seluruh nomor terdekat.

Ini dinyatakan dalam rumus berikut.

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

$$\text{Write requests per second} * \text{ROUNDUP}(\text{Average Row Size}/1024 \text{ per unit}) = \text{WCUs per second}$$

Misalnya, jika Anda melakukan 4.960 permintaan baca dengan ukuran baris 2.5KB pada tabel Cassandra Anda, Anda memerlukan 4.960 RCU di Amazon Keyspaces. Jika saat ini Anda melakukan 1.653 permintaan tulis per detik dengan ukuran baris 2,5KB di tabel Cassandra Anda, Anda memerlukan 4.959 WCU per detik di Amazon Keyspaces.

Contoh ini dinyatakan dalam rumus berikut.

$$\begin{aligned} &4,960 \text{ read requests per second} * \text{ROUNDUP}(2.5\text{KB} / 4\text{KB bytes per unit}) \\ &= 4,960 \text{ read requests per second} * 1 \text{ RCU} \\ &= 4,960 \text{ RCUs} \end{aligned}$$

$$\begin{aligned} &1,653 \text{ write requests per second} * \text{ROUNDUP}(2.5\text{KB}/1\text{KB per unit}) \\ &= 1,653 \text{ requests per second} * 3 \text{ WCUs} \\ &= 4,959 \text{ WCUs} \end{aligned}$$

Menggunakan eventual consistency memungkinkan Anda menghemat hingga setengah dari kapasitas throughput pada setiap permintaan baca. Setiap pembacaan yang konsisten pada akhirnya dapat mengkonsumsi hingga 8KB. Anda dapat menghitung pembacaan konsisten akhirnya dengan mengalikan perhitungannya sebelumnya dengan 0,5 seperti yang ditunjukkan pada rumus berikut.

$$\begin{aligned} &4,960 \text{ read requests per second} * \text{ROUNDUP}(2.5\text{KB} / 4\text{KB per unit}) * .5 \\ &= 2,480 \text{ read request per second} * 1 \text{ RCU} \\ &= 2,480 \text{ RCUs} \end{aligned}$$

5. Hitung estimasi harga bulanan untuk Amazon Keyspaces

Untuk memperkirakan penagihan bulanan untuk tabel berdasarkan throughput read/write kapasitas, Anda dapat menghitung harga untuk on-demand dan untuk mode yang disediakan menggunakan rumus yang berbeda dan membandingkan opsi untuk tabel Anda.

Mode yang disediakan - Konsumsi kapasitas baca dan tulis ditagih pada tarif per jam berdasarkan unit kapasitas per detik. Pertama, bagi tingkat itu dengan 0,7 untuk mewakili pemanfaatan target autoscaling default sebesar 70%. Kemudian beberapa kali dengan 30 hari kalender, 24 jam per hari, dan harga tarif regional.

Perhitungan ini dirangkum dalam rumus berikut.


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

On-demand mode - Kapasitas baca dan tulis ditagih berdasarkan tarif per permintaan. Pertama, kalikan tingkat permintaan dengan 30 hari kalender, dan 24 jam per hari. Kemudian bagi dengan satu juta unit permintaan. Akhirnya, kalikan dengan tarif regional.

Perhitungan ini dirangkum dalam rumus berikut.

```
((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
```

Pilih strategi migrasi

Anda dapat memilih di antara strategi migrasi berikut saat bermigrasi dari Apache Cassandra ke Amazon Keyspaces:

- **Online** — Ini adalah migrasi langsung menggunakan penulisan ganda untuk mulai menulis data baru ke Amazon Keyspaces dan cluster Cassandra secara bersamaan. Jenis migrasi ini direkomendasikan untuk aplikasi yang tidak memerlukan waktu henti selama migrasi dan konsistensi baca setelah penulisan.

Untuk informasi selengkapnya tentang cara merencanakan dan menerapkan strategi migrasi online, lihat [the section called “Migrasi online”](#).

- **Offline** — Teknik migrasi ini melibatkan penyalinan kumpulan data dari Cassandra ke Amazon Keyspaces selama jendela downtime. Migrasi offline dapat menyederhanakan proses migrasi, karena tidak memerlukan perubahan pada aplikasi Anda atau resolusi konflik antara data historis dan penulisan baru.

Untuk informasi selengkapnya tentang cara merencanakan migrasi offline, lihat [the section called “Migrasi offline”](#).

- **Hybrid** — Teknik migrasi ini memungkinkan perubahan direplikasi ke Amazon Keyspaces dalam waktu dekat, tetapi tanpa konsistensi baca demi tulis.

Untuk informasi selengkapnya tentang cara merencanakan migrasi hibrida, lihat [the section called “Migrasi hibrida”](#).

Setelah meninjau teknik migrasi dan praktik terbaik yang dibahas dalam topik ini, Anda dapat menempatkan opsi yang tersedia di pohon keputusan untuk merancang strategi migrasi berdasarkan kebutuhan dan sumber daya yang tersedia.

Migrasi online ke Amazon Keyspaces: strategi dan praktik terbaik

Jika Anda perlu menjaga ketersediaan aplikasi selama migrasi dari Apache Cassandra ke Amazon Keyspaces, Anda dapat menyiapkan strategi migrasi online khusus dengan menerapkan komponen utama yang dibahas dalam topik ini. Dengan mengikuti praktik terbaik untuk migrasi online ini, Anda dapat memastikan bahwa ketersediaan aplikasi dan konsistensi baca-setelah-tulis dipertahankan selama seluruh proses migrasi, sehingga meminimalkan dampak pada pengguna Anda.

Saat merancang strategi migrasi online dari Apache Cassandra ke Amazon Keyspaces, Anda perlu mempertimbangkan langkah-langkah kunci berikut.

1. Menulis data baru

- ZDM Dual Write Proxy untuk Migrasi Keyspaces Amazon — Gunakan ZDM Dual Write Proxy yang tersedia di [Github](#) untuk melakukan migrasi zero-downtime dari Apache Cassandra ke Amazon Keyspaces. Proxy ZDM melakukan penulisan ganda tanpa perlu memfaktorkan ulang aplikasi yang ada dan melakukan pembacaan ganda untuk validasi kueri.
- Application dual-write: Anda dapat menerapkan penulisan ganda dalam aplikasi Anda menggunakan pustaka dan driver klien Cassandra yang ada. Tentukan satu database sebagai pemimpin dan yang lainnya sebagai pengikut. Kegagalan menulis ke database pengikut dicatat dalam [antrian huruf mati \(DLQ\)](#) untuk analisis.
- Tulis ganda tingkat pesan: Atau, Anda dapat mengonfigurasi platform perpesanan yang ada untuk mengirim tulisan ke Cassandra dan Amazon Keyspaces menggunakan konsumen tambahan. Ini pada akhirnya menciptakan tampilan yang konsisten di kedua database.

2. Migrasi data historis

- Salin data historis: Anda dapat memigrasikan data historis dari Cassandra ke Amazon Keyspaces menggunakan AWS Glue atau kustom ekstrak, transformasi, dan muat (ETL) skrip. Menangani resolusi konflik antara penulisan ganda dan beban massal menggunakan teknik seperti transaksi ringan atau stempel waktu.

- Gunakan Time-To-Live (TTL): Untuk periode retensi data yang lebih pendek, Anda dapat menggunakan TTL di Cassandra dan Amazon Keyspaces untuk menghindari mengunggah data historis yang tidak perlu. Karena data lama kedaluwarsa di Cassandra dan data baru ditulis melalui penulisan ganda, Amazon Keyspaces akhirnya menyusul.

3. Memvalidasi data

- Pembacaan ganda: Menerapkan pembacaan ganda dari database Cassandra (primer) dan Amazon Keyspaces (sekunder), membandingkan hasil secara asinkron. Perbedaan dicatat atau dikirim ke DLQ.
- Sampel dibaca: Gunakan fungsi Λ untuk mengambil sampel dan membandingkan data secara berkala di kedua sistem, mencatat perbedaan apa pun ke DLQ.

4. Migrasi aplikasi

- Blue-green strategi: Alihkan aplikasi Anda untuk memperlakukan Amazon Keyspaces sebagai primer dan Cassandra sebagai penyimpanan data sekunder dalam satu langkah. Pantau kinerja dan putar kembali jika masalah muncul.
- Penerapan Canary: Secara bertahap meluncurkan migrasi ke subset pengguna terlebih dahulu, secara bertahap meningkatkan lalu lintas ke Amazon Keyspaces sebagai primer hingga sepenuhnya dimigrasikan.

5. Penonaktifan Cassandra

Setelah aplikasi Anda sepenuhnya dimigrasikan ke Amazon Keyspaces dan konsistensi data divalidasi, Anda dapat merencanakan untuk menonaktifkan klaster Cassandra Anda berdasarkan kebijakan penyimpanan data.

Dengan merencanakan strategi migrasi online dengan komponen-komponen ini, Anda dapat bertransisi dengan lancar ke layanan Amazon Keyspaces yang dikelola sepenuhnya dengan waktu henti atau gangguan minimal. Bagian berikut masuk ke setiap komponen secara lebih rinci.

Topik

- [Menulis data baru selama migrasi online](#)
- [Mengunggah data historis selama migrasi online](#)
- [Memvalidasi konsistensi data selama migrasi online](#)
- [Migrasi aplikasi selama migrasi online](#)
- [Menonaktifkan Cassandra setelah migrasi online](#)

Menulis data baru selama migrasi online

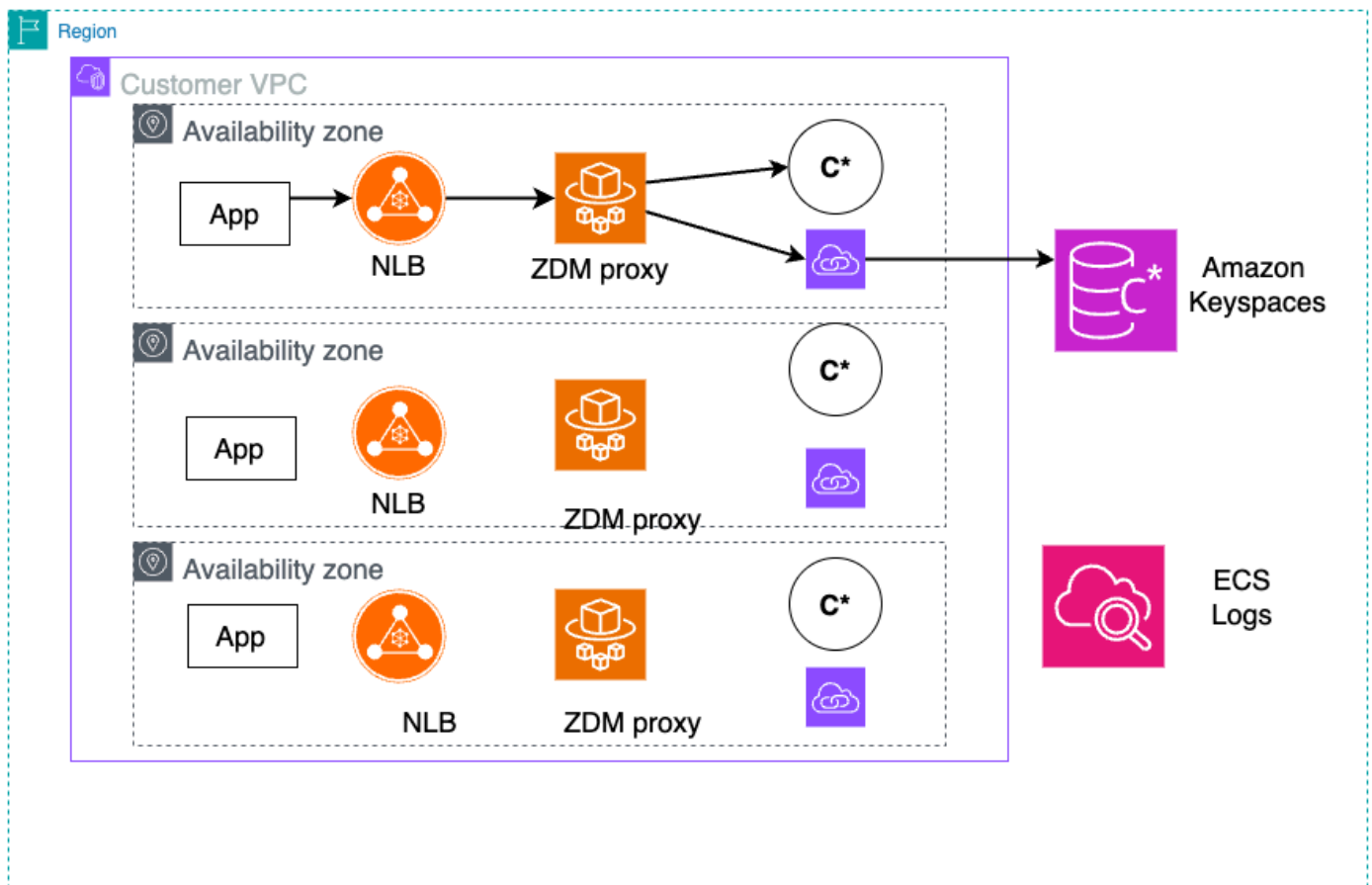
Langkah pertama dalam rencana migrasi online adalah memastikan bahwa setiap data baru yang ditulis oleh aplikasi disimpan di kedua database, cluster Cassandra yang ada, dan Amazon Keyspaces. Tujuannya adalah untuk memberikan pandangan yang konsisten di kedua penyimpanan data. Anda dapat melakukan ini dengan menerapkan semua penulisan baru ke kedua database. Untuk menerapkan penulisan ganda, pertimbangkan salah satu dari tiga opsi berikut.

- ZDM Dual Write Proxy untuk Migrasi Keyspaces Amazon — Menggunakan Proxy ZDM untuk Amazon Keyspaces yang tersedia [di Github](#), Anda dapat memigrasikan beban kerja Apache Cassandra ke Amazon Keyspaces tanpa downtime aplikasi. Solusi yang disempurnakan ini menerapkan praktik AWS terbaik dan memperluas kemampuan Proxy ZDM resmi.
- Lakukan migrasi online antara Apache Cassandra dan Amazon Keyspaces.
- Tulis data ke tabel sumber dan target secara bersamaan tanpa aplikasi refactoring.
- Validasi kueri melalui operasi dual-read.

Solusinya menawarkan penyempurnaan berikut untuk bekerja dengan dan AWS Amazon Keyspaces.

- Penyebaran kontainer — Gunakan image Docker yang telah dikonfigurasi sebelumnya dari Amazon Elastic Container Registry (Amazon ECR) Registry (Amazon ECR) untuk penerapan. VPC-accessible
- Infrastruktur sebagai kode - Terapkan menggunakan AWS CloudFormation templat untuk penyiapan otomatis aktif. AWS Fargate
- Kompatibilitas Amazon Keyspaces — Akses tabel sistem dengan adaptasi khusus untuk Amazon Keyspaces.

Solusinya berjalan di Amazon ECS dengan Fargate, menyediakan skalabilitas tanpa server berdasarkan tuntutan beban kerja Anda. Penyeimbang beban jaringan mendistribusikan lalu lintas aplikasi yang masuk di beberapa tugas Amazon ECS untuk ketersediaan tinggi.



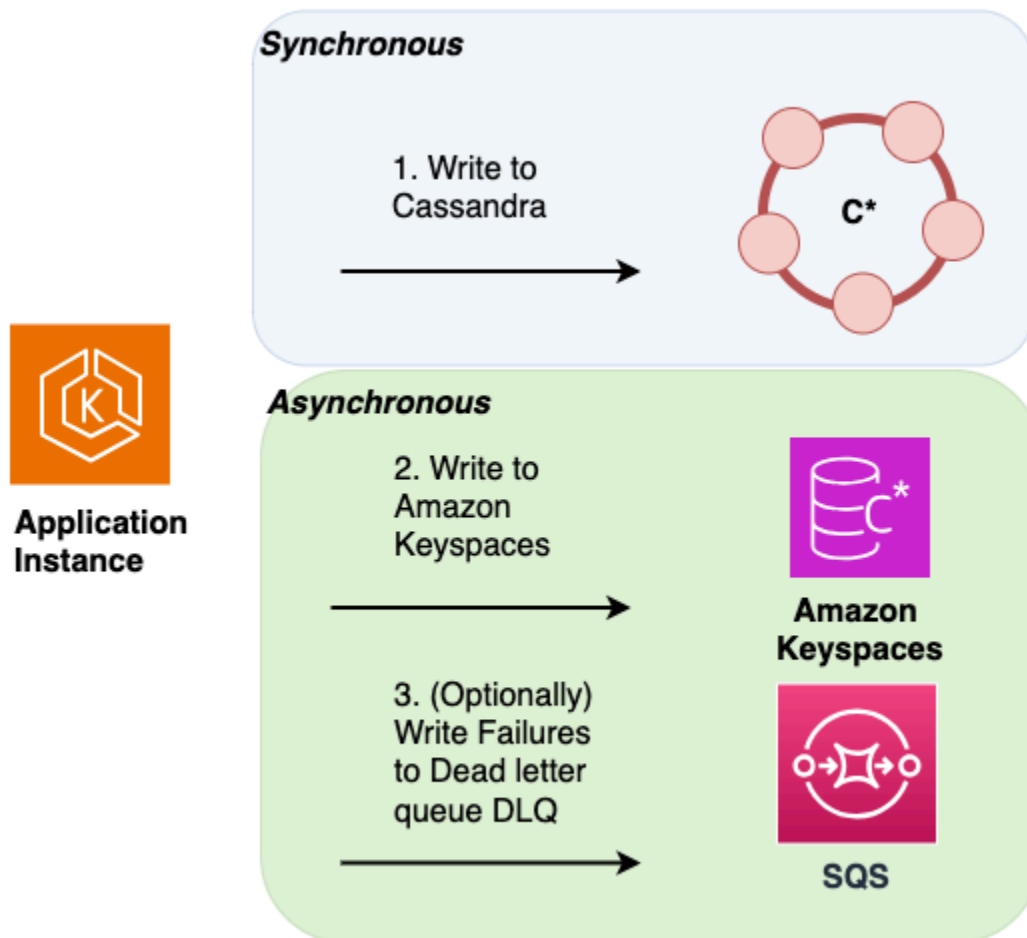
- **Application dual write** — Anda dapat menerapkan penulisan ganda dengan perubahan minimal pada kode aplikasi Anda dengan memanfaatkan pustaka dan driver klien Cassandra yang ada. Anda dapat menerapkan penulisan ganda dalam aplikasi yang ada, atau membuat layer baru dalam arsitektur untuk menangani penulisan ganda. Untuk informasi lebih lanjut dan studi kasus pelanggan yang menunjukkan bagaimana penulisan ganda diimplementasikan dalam aplikasi yang ada, lihat Studi [kasus migrasi Cassandra](#).

Saat menerapkan penulisan ganda, Anda dapat menunjuk satu database sebagai pemimpin dan database lainnya sebagai pengikut. Ini memungkinkan Anda untuk terus menulis ke sumber asli Anda, atau database pemimpin tanpa membiarkan kegagalan menulis ke pengikut, atau database tujuan mengganggu jalur kritis aplikasi Anda.

Alih-alih mencoba menulis ulang gagal ke pengikut, Anda dapat menggunakan Amazon Simple Queue Service untuk merekam penulisan gagal dalam [antrian surat mati](#) (DLQ). DLQ memungkinkan Anda menganalisis penulisan yang gagal ke pengikut dan menentukan mengapa pemrosesan tidak berhasil dalam database tujuan.

Untuk implementasi penulisan ganda yang lebih canggih, Anda dapat mengikuti praktik AWS terbaik untuk merancang urutan transaksi lokal menggunakan [pola saga](#). Pola saga memastikan bahwa jika transaksi gagal, saga menjalankan transaksi kompensasi untuk mengembalikan perubahan database yang dibuat oleh transaksi sebelumnya.

Saat menggunakan dual-write untuk migrasi online, Anda dapat mengonfigurasi penulisan ganda mengikuti pola saga sehingga setiap penulisan adalah transaksi lokal untuk memastikan operasi atom di seluruh database heterogen. Untuk informasi selengkapnya tentang merancang aplikasi terdistribusi menggunakan pola desain yang direkomendasikan AWS Cloud, lihat [Pola, arsitektur, dan implementasi desain Cloud](#).



- Messaging tier dual write — Alih-alih menerapkan penulisan ganda di lapisan aplikasi, Anda dapat menggunakan tingkat pesan yang ada untuk melakukan penulisan ganda ke Cassandra dan Amazon Keyspaces.

Untuk melakukan ini, Anda dapat mengonfigurasi konsumen tambahan ke platform perpesanan Anda untuk mengirim tulisan ke kedua penyimpanan data. Pendekatan ini menyediakan strategi kode rendah sederhana menggunakan tingkat pesan untuk membuat dua tampilan di kedua database yang pada akhirnya konsisten.

Mengunggah data historis selama migrasi online

Setelah menerapkan penulisan ganda untuk memastikan bahwa data baru ditulis ke kedua penyimpanan data secara real time, langkah selanjutnya dalam rencana migrasi adalah mengevaluasi berapa banyak data historis yang perlu Anda salin atau unggah massal dari Cassandra ke Amazon Keyspaces. Ini memastikan bahwa data baru dan data historis akan tersedia di database Amazon Keyspaces baru sebelum Anda memigrasikan aplikasi.

Bergantung pada persyaratan penyimpanan data Anda, misalnya berapa banyak data historis yang perlu Anda pertahankan berdasarkan kebijakan organisasi Anda, Anda dapat mempertimbangkan salah satu dari dua opsi berikut.

- Pengunggahan massal data historis — Migrasi data historis dari penyebaran Cassandra Anda yang ada ke Amazon Keyspaces dapat dicapai melalui berbagai teknik, misalnya menggunakan AWS Glue atau skrip khusus untuk mengekstrak, mengubah, dan memuat (ETL) data. Untuk informasi selengkapnya tentang penggunaan AWS Glue untuk mengunggah data historis, lihat [the section called “Migrasi offline”](#).

Saat merencanakan unggahan massal data historis, Anda perlu mempertimbangkan cara menyelesaikan konflik yang dapat terjadi ketika penulisan baru mencoba memperbarui data yang sama yang sedang dalam proses diunggah. Upload massal diharapkan pada akhirnya konsisten, yang berarti data akan mencapai semua node pada akhirnya.

Jika pembaruan data yang sama terjadi pada saat yang sama karena penulisan baru, Anda ingin memastikan bahwa itu tidak akan ditimpa oleh unggahan data historis. Untuk memastikan bahwa Anda mempertahankan pembaruan terbaru pada data Anda bahkan selama impor massal, Anda harus menambahkan resolusi konflik baik ke dalam skrip unggahan massal atau ke dalam logika aplikasi untuk penulisan ganda.

Misalnya, Anda dapat menggunakan [the section called “Transaksi ringan”](#) (LWT) untuk membandingkan dan mengatur operasi. Untuk melakukan ini, Anda dapat menambahkan bidang tambahan ke model data Anda yang mewakili waktu modifikasi atau status.

Selain itu, Amazon Keyspaces mendukung fungsi stempel waktu Cassandra `WRITETIME`. Anda dapat menggunakan stempel waktu sisi klien Amazon Keyspaces untuk mempertahankan stempel waktu database sumber dan menerapkan resolusi konflik last-writer-win. Untuk informasi selengkapnya, lihat [the section called “Client-side stempel waktu”](#).

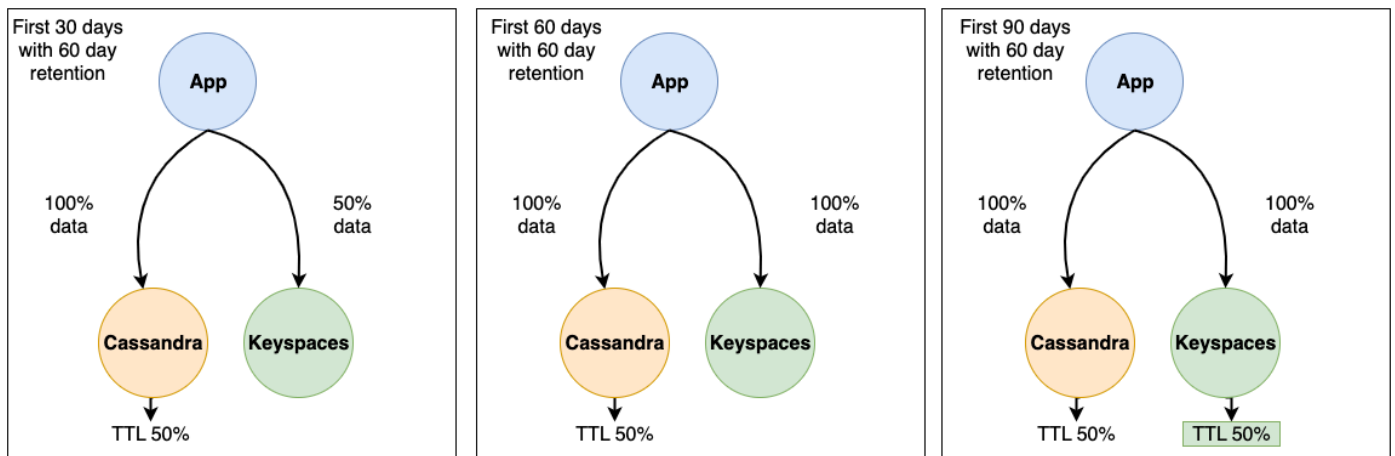
- Menggunakan Time-to-Live (TTL) — Untuk periode retensi data yang lebih pendek dari 30, 60, atau 90 hari, Anda dapat menggunakan TTL di Cassandra dan Amazon Keyspaces selama migrasi untuk menghindari mengunggah data historis yang tidak perlu ke Amazon Keyspaces. TTL memungkinkan Anda untuk mengatur periode waktu setelah data dihapus secara otomatis dari database.

Selama fase migrasi, alih-alih menyalin data historis ke Amazon Keyspaces, Anda dapat mengonfigurasi pengaturan TTL agar data historis kedaluwarsa secara otomatis di sistem lama (Cassandra) sambil hanya menerapkan penulisan baru ke Amazon Keyspaces menggunakan metode penulisan ganda. Seiring waktu dan dengan data lama yang terus kedaluwarsa di cluster Cassandra dan data baru yang ditulis menggunakan metode dual-write, Amazon Keyspaces secara otomatis menangkap data yang sama dengan Cassandra.

Pendekatan ini dapat secara signifikan mengurangi jumlah data yang akan dimigrasi, menghasilkan proses migrasi yang lebih efisien dan efisien. Anda dapat mempertimbangkan pendekatan ini ketika berhadapan dengan kumpulan data besar dengan berbagai persyaratan retensi data. Untuk informasi lebih lanjut tentang TTL, lihat [the section called “Data kedaluwarsa dengan Time to Live”](#).

Pertimbangkan contoh migrasi dari Cassandra ke Amazon Keyspaces berikut menggunakan kedaluwarsa data TTL. Dalam contoh ini kami menetapkan TTL untuk kedua database ke 60 hari dan menunjukkan bagaimana proses migrasi berlangsung selama periode 90 hari. Kedua database menerima data yang baru ditulis yang sama selama periode ini menggunakan metode penulisan ganda. Kita akan melihat tiga fase migrasi yang berbeda, setiap fase adalah 30 hari.

Cara kerja proses migrasi untuk setiap fase ditunjukkan pada gambar berikut.



1. Setelah 30 hari pertama, cluster Cassandra dan Amazon Keyspaces telah menerima penulisan baru. Cluster Cassandra juga berisi data historis yang belum mencapai retensi 60 hari, yang merupakan 50% dari data dalam cluster.

Data yang lebih tua dari 60 hari secara otomatis dihapus di cluster Cassandra menggunakan TTL. Pada titik ini Amazon Keyspaces berisi 50% data yang disimpan di cluster Cassandra, yang terdiri dari penulisan baru dikurangi data historis.

2. Setelah 60 hari, cluster Cassandra dan Amazon Keyspaces berisi data yang sama yang ditulis dalam 60 hari terakhir.
3. Dalam 90 hari, Cassandra dan Amazon Keyspaces berisi data yang sama dan data kedaluwarsa dengan kecepatan yang sama.

Contoh ini menggambarkan cara menghindari langkah mengunggah data historis dengan menggunakan TTL dengan tanggal kedaluwarsa yang ditetapkan ke 60 hari.

Memvalidasi konsistensi data selama migrasi online

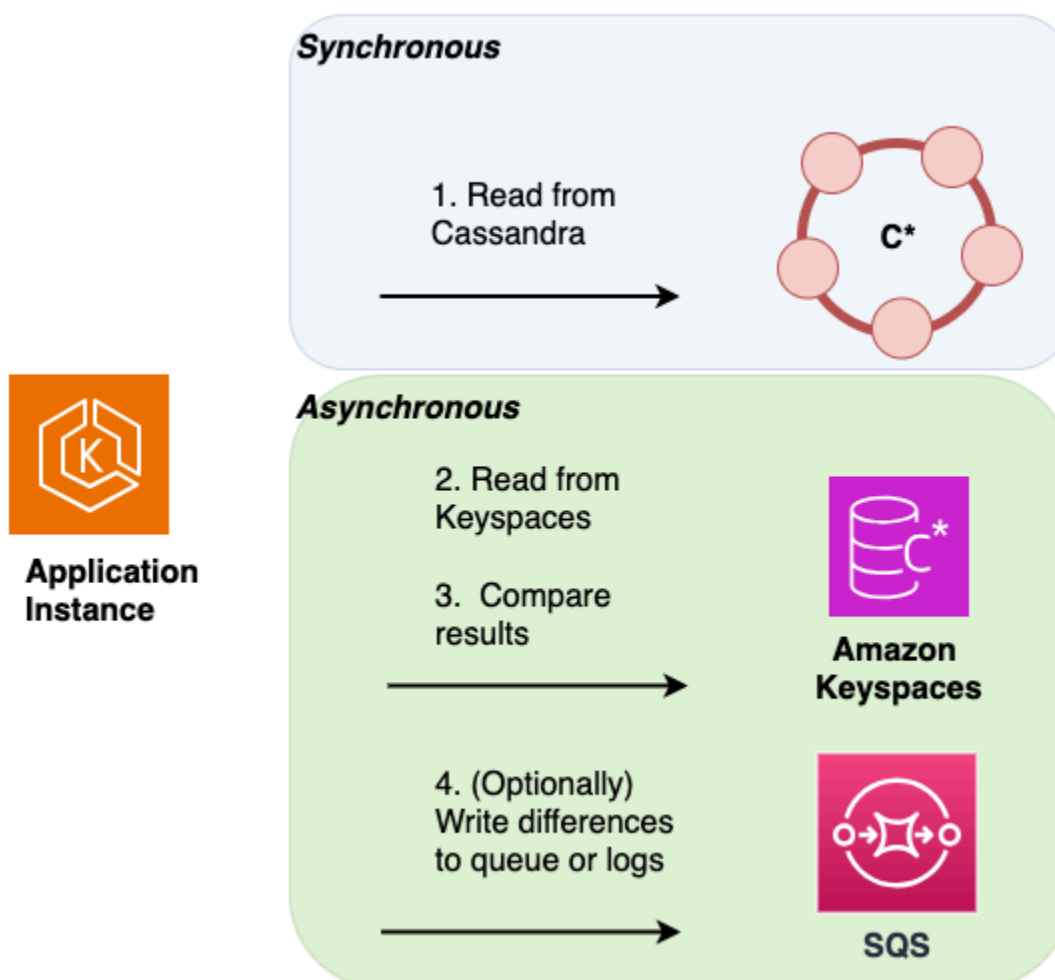
Langkah selanjutnya dalam proses migrasi online adalah validasi data. Penulisan ganda menambahkan data baru ke database Amazon Keyspaces Anda dan Anda telah menyelesaikan migrasi data historis baik menggunakan unggahan massal atau kedaluwarsa data dengan TTL.

Sekarang Anda dapat menggunakan fase validasi untuk mengonfirmasi bahwa kedua penyimpanan data sebenarnya berisi data yang sama dan mengembalikan hasil baca yang sama. Anda dapat memilih dari salah satu dari dua opsi berikut untuk memvalidasi bahwa kedua database Anda berisi data yang identik.

- **Bacaan ganda** — Untuk memvalidasi bahwa keduanya, sumber dan database tujuan berisi kumpulan data yang baru ditulis dan historis yang sama, Anda dapat menerapkan pembacaan ganda. Untuk melakukannya, Anda membaca dari Cassandra utama dan database Amazon Keyspaces sekunder Anda mirip dengan metode penulisan ganda dan membandingkan hasilnya secara asinkron.

Hasil dari database utama dikembalikan ke klien, dan hasil dari database sekunder digunakan untuk memvalidasi terhadap resultset utama. Perbedaan yang ditemukan dapat dicatat atau dikirim ke [antrian surat mati \(DLQ\)](#) untuk rekonsiliasi nanti.

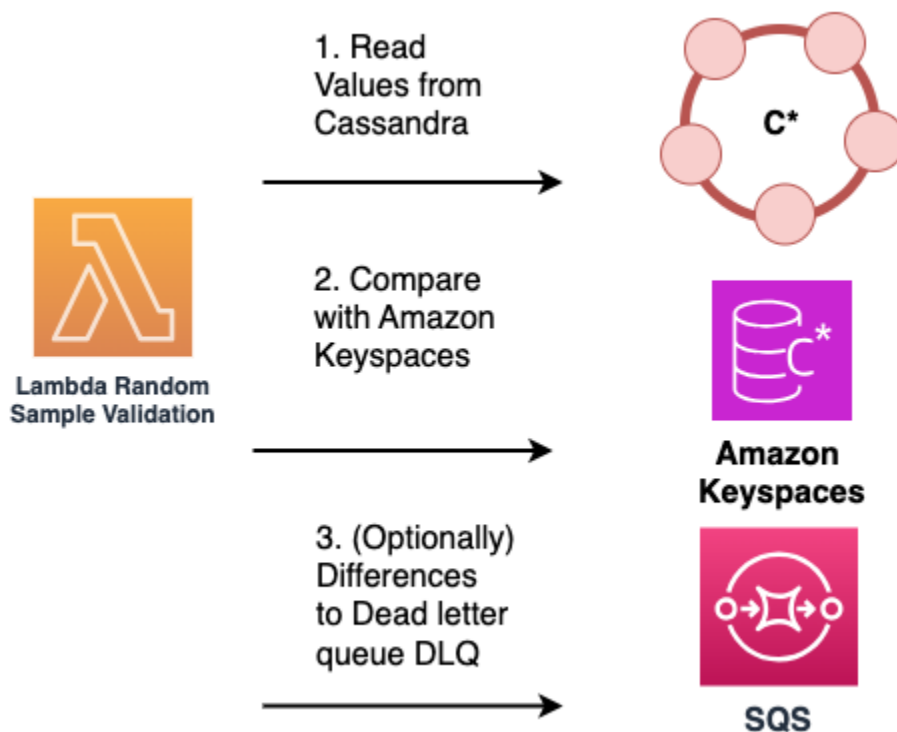
Dalam diagram berikut, aplikasi melakukan pembacaan sinkron dari Cassandra, yang merupakan penyimpanan data utama) dan pembacaan asinkron dari Amazon Keyspaces, yang merupakan penyimpanan data sekunder.



- **Pembacaan sampel** — Solusi alternatif yang tidak memerlukan perubahan kode aplikasi adalah dengan menggunakan AWS Lambda fungsi untuk mengambil sampel data secara berkala dan acak dari cluster Cassandra sumber dan database Amazon Keyspaces tujuan.

Fungsi Lambda ini dapat dikonfigurasi untuk berjalan secara berkala. Fungsi Lambda mengambil subset data acak dari sistem sumber dan tujuan, dan kemudian melakukan perbandingan data sampel. Setiap perbedaan atau ketidakcocokan antara dua kumpulan data dapat direkam dan dikirim ke [antrian surat mati khusus](#) (DLQ) untuk rekonsiliasi nanti.

Proses ini diilustrasikan dalam diagram berikut.



Migrasi aplikasi selama migrasi online

Pada fase keempat migrasi online, Anda memigrasikan aplikasi dan beralih ke Amazon Keyspaces sebagai penyimpanan data utama. Ini berarti Anda mengalihkan aplikasi Anda untuk membaca dan menulis langsung dari dan ke Amazon Keyspaces. Untuk memastikan gangguan minimal pada pengguna Anda, ini harus menjadi proses yang terencana dan terkoordinasi dengan baik.

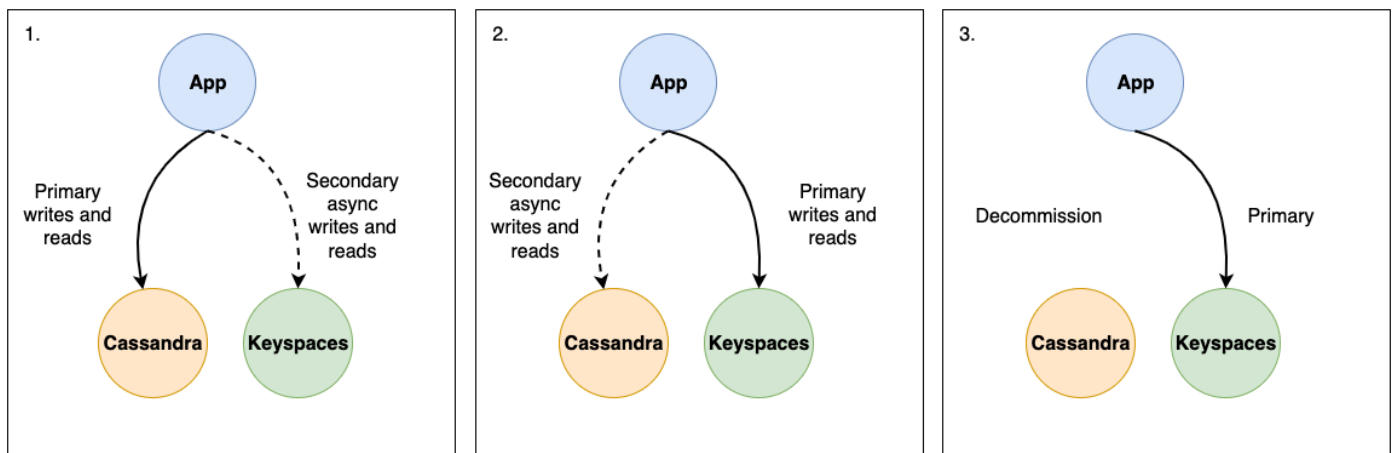
Tersedia dua solusi berbeda yang direkomendasikan untuk migrasi aplikasi, strategi pemotongan hijau biru dan strategi pemotongan kenari. Bagian berikut menguraikan strategi ini secara lebih rinci.

- Strategi hijau biru — Dengan menggunakan pendekatan ini, Anda mengalihkan aplikasi Anda untuk memperlakukan Amazon Keyspaces sebagai penyimpanan data utama dan Cassandra sebagai penyimpanan data sekunder dalam satu langkah. Anda dapat melakukan ini menggunakan flag AWS AppConfig fitur untuk mengontrol pemilihan penyimpanan data primer dan sekunder di seluruh instance aplikasi. Untuk informasi selengkapnya tentang flag fitur, lihat [Membuat profil konfigurasi flag fitur di AWS AppConfig](#).

Setelah menjadikan Amazon Keyspaces sebagai penyimpanan data utama, Anda memantau perilaku dan kinerja aplikasi, memastikan bahwa Amazon Keyspaces memenuhi persyaratan Anda dan migrasi berhasil.

Misalnya, jika Anda menerapkan pembacaan ganda untuk aplikasi Anda, selama fase migrasi aplikasi Anda mentransisikan pembacaan utama dari Cassandra ke Amazon Keyspaces dan pembacaan sekunder dari Amazon Keyspaces ke Cassandra. Setelah transisi, Anda terus memantau dan membandingkan hasil seperti yang dijelaskan di bagian [validasi data](#) untuk memastikan konsistensi di kedua database sebelum menonaktifkan Cassandra.

Jika Anda mendeteksi masalah apa pun, Anda dapat dengan cepat memutar kembali ke keadaan sebelumnya dengan kembali ke Cassandra sebagai penyimpanan data utama. Anda hanya melanjutkan ke fase penonaktifan migrasi jika Amazon Keyspaces memenuhi semua kebutuhan Anda sebagai penyimpanan data utama.

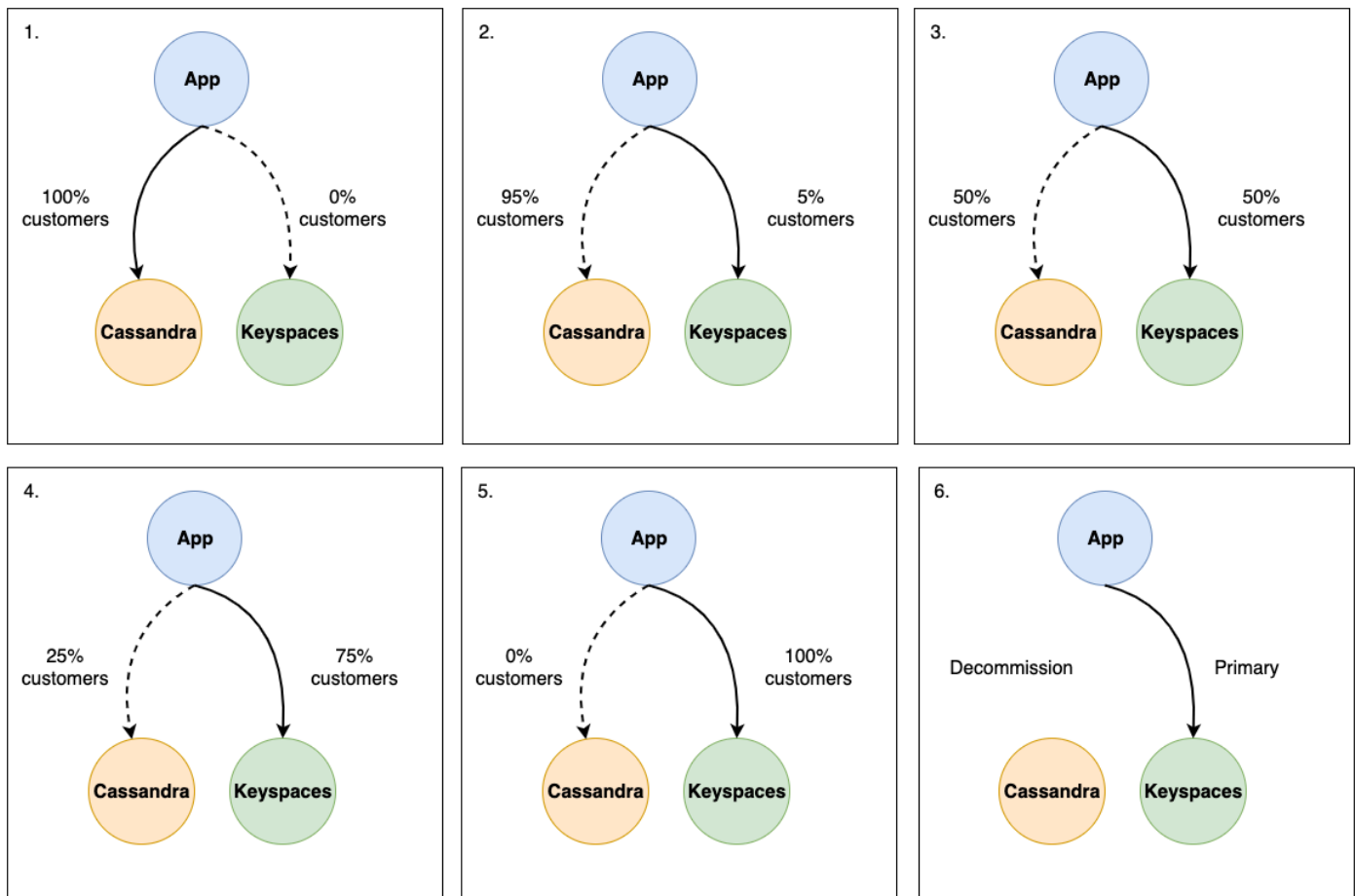


- Strategi kenari — Dalam pendekatan ini, Anda secara bertahap meluncurkan migrasi ke subset pengguna atau lalu lintas Anda. Awalnya, sebagian kecil lalu lintas aplikasi Anda, misalnya 5% dari semua lalu lintas dirutekan ke versi menggunakan Amazon Keyspaces sebagai penyimpanan data utama, sementara sisa lalu lintas terus menggunakan Cassandra sebagai penyimpanan data utama.

Ini memungkinkan Anda untuk menguji versi migrasi secara menyeluruh dengan lalu lintas dunia nyata dan memantau kinerja, stabilitas, dan menyelidiki potensi masalah. Jika Anda tidak mendeteksi masalah apa pun, Anda dapat secara bertahap meningkatkan persentase lalu lintas yang dirutekan ke Amazon Keyspaces hingga menjadi penyimpanan data utama untuk semua pengguna dan lalu lintas.

Peluncuran bertahap ini meminimalkan risiko gangguan layanan yang meluas dan memungkinkan proses migrasi yang lebih terkontrol. Jika ada masalah kritis yang muncul selama penyebaran kenari, Anda dapat dengan cepat memutar kembali ke versi sebelumnya menggunakan Cassandra sebagai penyimpanan data utama untuk segmen lalu lintas yang terpengaruh. Anda hanya melanjutkan ke fase penonaktifan migrasi setelah Anda memvalidasi bahwa Amazon Keyspaces memproses 100% pengguna dan lalu lintas Anda seperti yang diharapkan.

Diagram berikut menggambarkan langkah-langkah individu dari strategi kenari.



Menonaktifkan Cassandra setelah migrasi online

Setelah migrasi aplikasi selesai dengan aplikasi Anda sepenuhnya berjalan di Amazon Keyspaces dan Anda telah memvalidasi konsistensi data selama periode waktu tertentu, Anda dapat merencanakan untuk menonaktifkan cluster Cassandra Anda. Selama fase ini, Anda dapat mengevaluasi apakah data yang tersisa di cluster Cassandra Anda perlu diarsipkan atau dapat dihapus. Ini tergantung pada kebijakan organisasi Anda untuk penanganan dan penyimpanan data.

Dengan mengikuti strategi ini dan mempertimbangkan praktik terbaik yang direkomendasikan yang dijelaskan dalam topik ini saat merencanakan migrasi online Anda dari Cassandra ke Amazon Keyspaces, Anda dapat memastikan transisi yang mulus ke Amazon Keyspaces sambil mempertahankan konsistensi baca-setelah-tulis dan ketersediaan aplikasi Anda.

Bermigrasi dari Apache Cassandra ke Amazon Keyspaces dapat memberikan banyak manfaat, termasuk pengurangan overhead operasional, penskalaan otomatis, keamanan yang ditingkatkan, dan kerangka kerja yang membantu Anda mencapai tujuan kepatuhan Anda. Dengan merencanakan strategi migrasi online dengan penulisan ganda, pengunggahan data historis, validasi data, dan peluncuran bertahap, Anda dapat memastikan transisi yang mulus dengan gangguan minimal pada aplikasi Anda dan penggunaannya.

Menerapkan strategi migrasi online yang dibahas dalam topik ini memungkinkan Anda memvalidasi hasil migrasi, mengidentifikasi dan mengatasi masalah apa pun, dan pada akhirnya menonaktifkan penerapan Cassandra yang ada demi layanan Amazon Keyspaces yang dikelola sepenuhnya.

Proses migrasi offline: Apache Cassandra ke Amazon Keyspaces

Migrasi offline cocok bila Anda mampu melakukan downtime untuk melakukan migrasi. Sudah umum di antara perusahaan untuk memiliki jendela pemeliharaan untuk patching, rilis besar, atau downtime untuk peningkatan perangkat keras atau peningkatan besar. Migrasi offline dapat menggunakan jendela ini untuk menyalin data dan mengalihkan lalu lintas aplikasi dari Apache Cassandra ke Amazon Keyspaces.

Migrasi offline mengurangi modifikasi pada aplikasi karena tidak memerlukan komunikasi ke Cassandra dan Amazon Keyspaces secara bersamaan. Selain itu, dengan aliran data dijeda, status yang tepat dapat disalin tanpa mempertahankan mutasi.

Dalam contoh ini, kami menggunakan Amazon Simple Storage Service (Amazon S3) sebagai area pementasan data selama migrasi offline untuk meminimalkan waktu henti. Anda dapat secara otomatis mengimpor data yang Anda simpan dalam format Parquet di Amazon S3 ke dalam tabel

Amazon Keyspaces menggunakan konektor Spark Cassandra dan. AWS Glue Bagian berikut akan menunjukkan ikhtisar tingkat tinggi dari proses tersebut. Anda dapat menemukan contoh kode untuk proses ini di [Github](#).

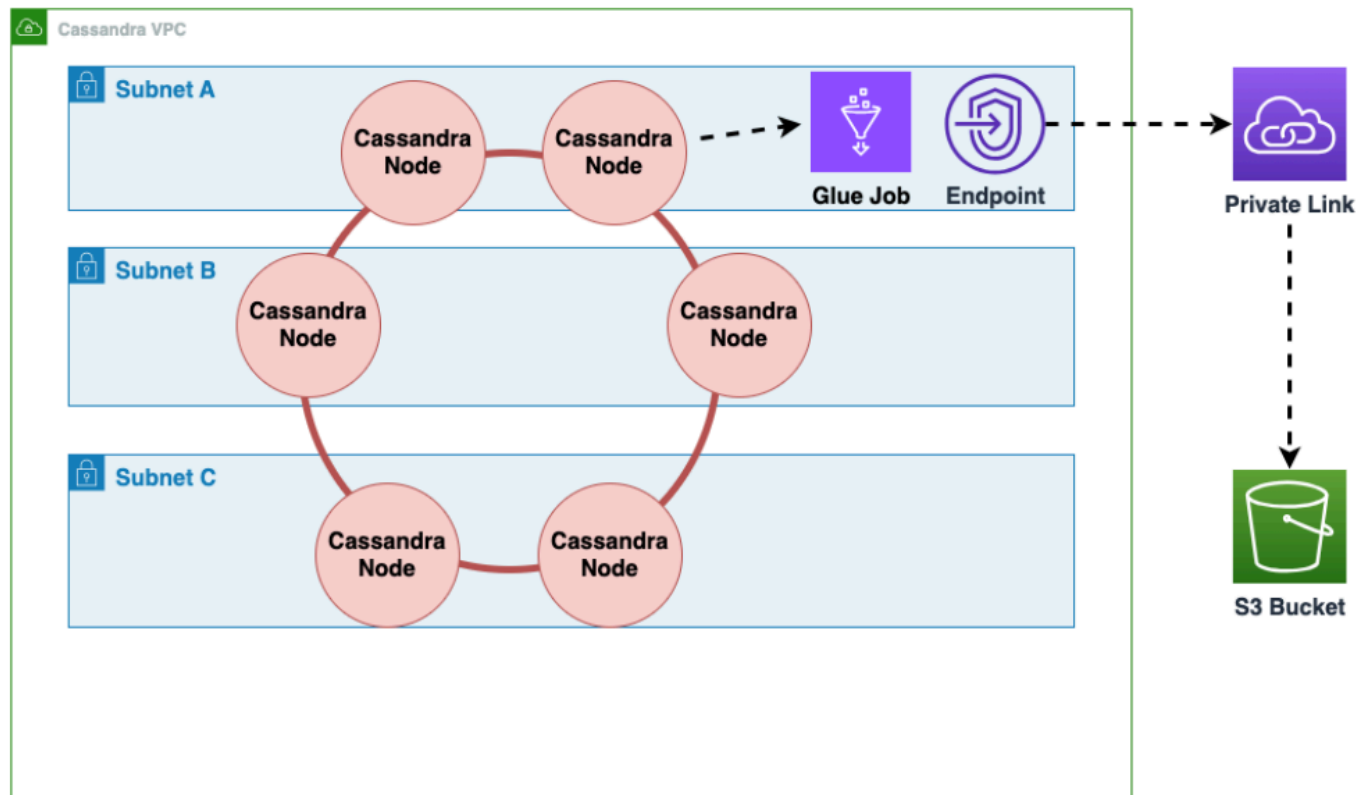
Proses migrasi offline dari Apache Cassandra ke Amazon Keyspaces menggunakan Amazon S3 dan memerlukan pekerjaan berikut. AWS Glue AWS Glue

1. Pekerjaan ETL yang mengekstrak dan mengubah data CQL dan menyimpannya di bucket Amazon S3.
2. Pekerjaan kedua yang mengimpor data dari bucket ke Amazon Keyspaces.
3. Pekerjaan ketiga untuk mengimpor data tambahan.

Cara melakukan migrasi offline ke Amazon Keyspaces dari Cassandra yang berjalan di Amazon EC2 di Amazon Virtual Private Cloud

1. Pertama Anda gunakan AWS Glue untuk mengekspor data tabel dari Cassandra dalam format Parquet dan menyimpannya ke ember Amazon S3. Anda perlu menjalankan AWS Glue pekerjaan menggunakan AWS Glue konektor ke VPC tempat instans Amazon EC2 yang menjalankan Cassandra berada. Kemudian, menggunakan titik akhir pribadi Amazon S3, Anda dapat menyimpan data ke bucket Amazon S3.

Diagram berikut menggambarkan langkah-langkah ini.



2. Kocokkan data di bucket Amazon S3 untuk meningkatkan pengacakan data. Data yang diimpor secara merata memungkinkan lalu lintas yang lebih terdistribusi di tabel target.

Langkah ini diperlukan saat mengekspor data dari Cassandra dengan partisi besar (partisi dengan lebih dari 1000 baris) untuk menghindari pola tombol pintas saat memasukkan data ke Amazon Keyspaces. Masalah kunci panas terjadi `WriteThrottleEvents` di Amazon Keyspaces dan mengakibatkan peningkatan waktu muat.



3. Gunakan AWS Glue pekerjaan lain untuk mengimpor data dari bucket Amazon S3 ke Amazon Keyspaces. Data yang diacak di bucket Amazon S3 disimpan dalam format Parquet.



Untuk informasi selengkapnya tentang proses migrasi offline, lihat lokakarya [Amazon Keyspaces with AWS Glue](#)

Menggunakan solusi migrasi hibrida: Apache Cassandra ke Amazon Keyspaces

Solusi migrasi berikut dapat dianggap sebagai hibrida antara migrasi online dan offline. Dengan pendekatan hybrid ini, data ditulis ke database tujuan dalam waktu dekat tanpa memberikan konsistensi baca demi tulis. Ini berarti bahwa data yang baru ditulis tidak akan segera tersedia dan penundaan diharapkan terjadi. Jika Anda perlu membaca setelah menulis konsistensi, lihat [the section called “Migrasi online”](#).

Untuk migrasi waktu nyata dari Apache Cassandra ke Amazon Keyspaces, Anda dapat memilih di antara dua metode yang tersedia.

- **CQLReplicator** - (Disarankan) CQLReplicator adalah utilitas open source yang tersedia di [Github](#) yang membantu Anda memigrasikan data dari Apache Cassandra ke Amazon Keyspaces dalam waktu dekat.

Untuk menentukan penulisan dan pembaruan untuk disebarkan ke database tujuan, CQLReplicator memindai rentang token Apache Cassandra dan menggunakan AWS Glue pekerjaan untuk menghapus peristiwa duplikat dan menerapkan penulisan dan pembaruan langsung ke Amazon Keyspaces.

- **Ubah pengambilan data (CDC)** - Jika Anda terbiasa dengan Cassandra CDC, fitur CDC bawaan Apache Cassandra yang memungkinkan pengambilan perubahan dengan menyalin log komit ke direktori CDC terpisah adalah opsi lain untuk menerapkan migrasi hibrida.

Anda dapat melakukannya dengan mereplikasi perubahan data ke Amazon Keyspaces, menjadikan CDC sebagai opsi alternatif untuk skenario migrasi data.

Jika Anda tidak memerlukan konsistensi baca setelah menulis, Anda dapat menggunakan CQLReplicator atau saluran CDC untuk memigrasikan data dari Apache Cassandra ke Amazon Keyspaces berdasarkan preferensi dan keakraban Anda dengan alat dan digunakan dalam setiap solusi. Layanan AWS Menggunakan metode ini untuk memigrasikan data dalam waktu dekat dapat dianggap sebagai pendekatan hibrida untuk migrasi yang menawarkan alternatif untuk migrasi online.

Strategi ini dianggap sebagai pendekatan hibrida, karena selain opsi yang diuraikan dalam topik ini, Anda harus menerapkan beberapa langkah kemajuan migrasi online, misalnya salinan data historis dan strategi migrasi aplikasi yang dibahas dalam topik [migrasi online](#).

Bagian berikut membahas opsi migrasi hibrida secara lebih rinci.

Topik

- [Migrasi data menggunakan CQLReplicator](#)
- [Migrasi data menggunakan change data capture \(CDC\)](#)

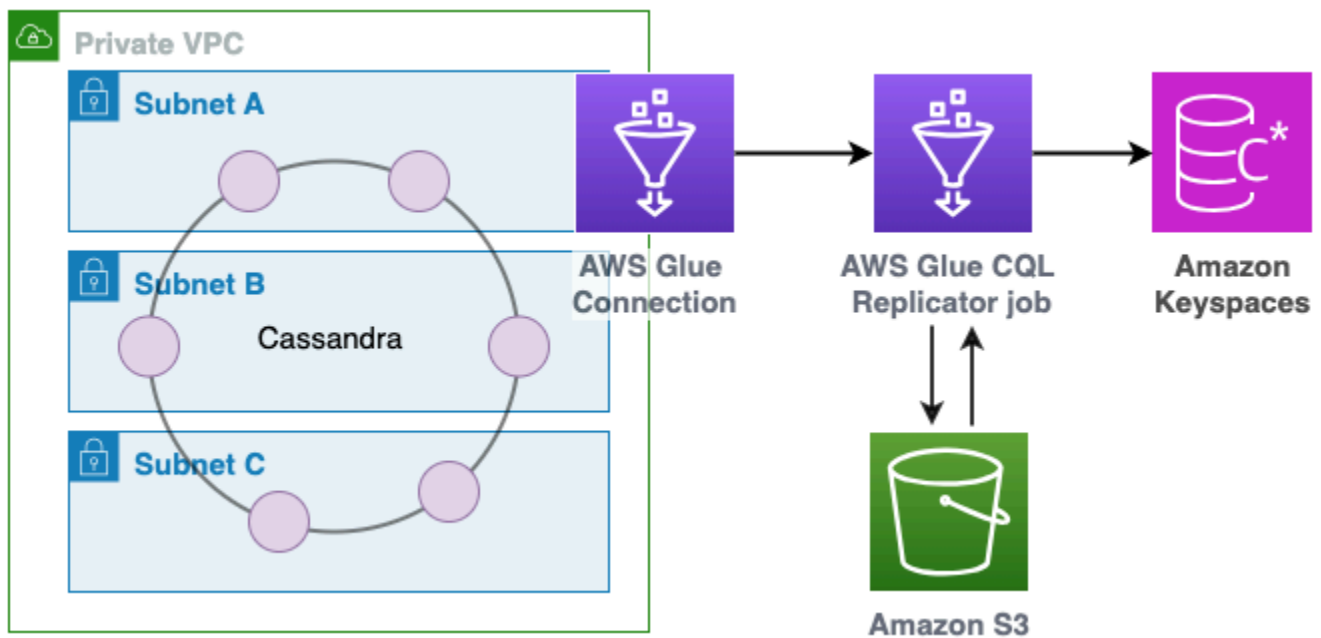
Migrasi data menggunakan CQLReplicator

Dengan [CQLReplicator](#), Anda dapat membaca data dari Apache Cassandra dalam waktu dekat dengan memindai cincin token Cassandra secara cerdas menggunakan kueri CQL. CQLReplicator tidak menggunakan Cassandra CDC dan sebagai gantinya menerapkan strategi caching untuk mengurangi hukuman kinerja pemindaian penuh.

Untuk mengurangi jumlah penulisan ke tujuan, CQLReplicator secara otomatis menghapus peristiwa replikasi duplikat. Dengan CQLReplicator, Anda dapat menyetel replikasi perubahan dari database sumber ke database tujuan, memungkinkan migrasi data secara real time dari Apache Cassandra ke Amazon Keyspaces.

Diagram berikut menunjukkan arsitektur khas dari pekerjaan CQLReplicator menggunakan. AWS Glue

1. Untuk memungkinkan akses ke Apache Cassandra berjalan di VPC pribadi, konfigurasi AWS Glue koneksi dengan jenis koneksi Jaringan.
2. Untuk menghapus duplikat dan mengaktifkan caching kunci dengan tugas CQLReplicator, konfigurasi Amazon Simple Storage Service (Amazon S3).
3. Aliran pekerjaan CQLReplicator mengubah database sumber terverifikasi langsung ke Amazon Keyspaces.



Untuk informasi selengkapnya tentang proses migrasi menggunakan CQLReplicator, lihat [postingan berikut di blog AWS Database Migrasikan beban kerja Cassandra ke Amazon Keyspaces menggunakan CQLReplicator dan panduan preskriptif Memigrasikan beban kerja Apache Cassandra ke Amazon Keyspaces AWS dengan menggunakan AWS Glue](#)

Migrasi data menggunakan change data capture (CDC)

Jika Anda sudah terbiasa mengonfigurasi pipeline change data capture (CDC) dengan [Debezium](#), Anda dapat menggunakan opsi ini untuk memigrasikan data ke Amazon Keyspaces sebagai alternatif untuk menggunakan CQLReplicator. Debezium adalah platform terdistribusi open-source untuk CDC, yang dirancang untuk memantau database dan menangkap perubahan tingkat baris dengan andal.

[Konektor Debezium untuk Apache Cassandra mengunggah](#) perubahan ke Amazon Managed Streaming for Apache Kafka (Amazon MSK) sehingga dapat dikonsumsi dan diproses oleh konsumen hilir yang pada gilirannya menulis data ke Amazon Keyspaces. Untuk informasi selengkapnya, lihat [Panduan migrasi data berkelanjutan dari Apache Cassandra ke Amazon Keyspaces](#).

Untuk mengatasi masalah konsistensi data potensial, Anda dapat menerapkan proses dengan Amazon MSK di mana konsumen membandingkan kunci atau partisi di Cassandra dengan yang ada di Amazon Keyspaces.

Untuk mengimplementasikan solusi ini dengan sukses, kami sarankan untuk mempertimbangkan hal berikut.

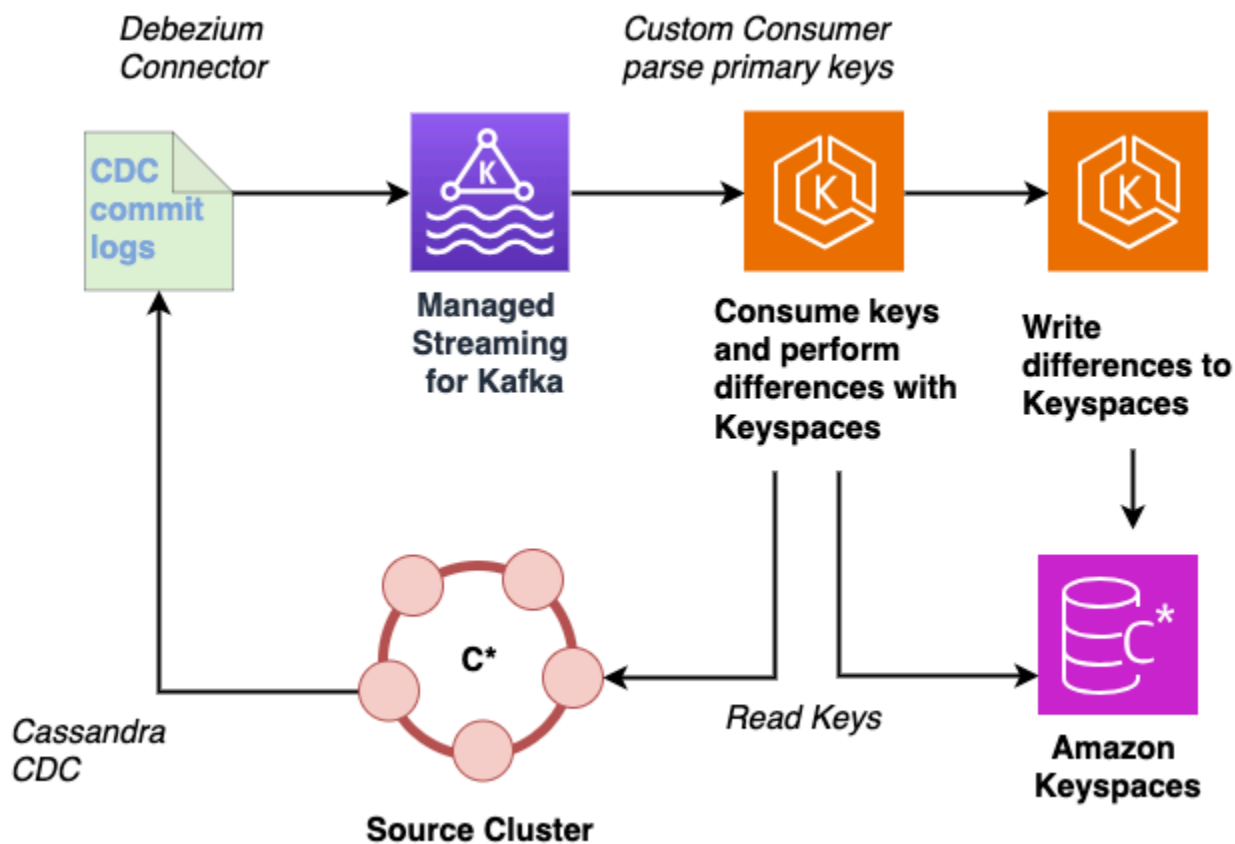
- Cara mengurai log komit CDC, misalnya cara menghapus peristiwa duplikat.
- Cara memelihara direktori CDC, misalnya cara menghapus log lama.
- Cara menangani kegagalan sebagian di Apache Cassandra, misalnya jika penulisan hanya berhasil dalam satu dari tiga replika.
- Cara menangani alokasi sumber daya, misalnya meningkatkan ukuran instance untuk memperhitungkan persyaratan CPU, memori, DISK, dan IO tambahan untuk proses CDC yang terjadi pada node.

Pola ini memperlakukan perubahan dari Cassandra sebagai “petunjuk” bahwa kunci mungkin telah berubah dari keadaan sebelumnya. Untuk menentukan apakah ada perubahan untuk disebarkan ke database tujuan, Anda harus terlebih dahulu membaca dari cluster Cassandra sumber menggunakan LOCAL_QUORUM operasi untuk menerima catatan terbaru dan kemudian menuliskannya ke Amazon Keyspaces.

Dalam kasus penghapusan rentang atau pembaruan rentang, Anda mungkin perlu melakukan perbandingan terhadap seluruh partisi untuk menentukan peristiwa penulisan atau pembaruan mana yang perlu ditulis ke database tujuan Anda.

Dalam kasus di mana penulisan tidak idempoten, Anda juga perlu membandingkan tulisan Anda dengan apa yang sudah ada di database tujuan sebelum menulis ke Amazon Keyspaces.

Diagram berikut menunjukkan arsitektur khas pipa CDC menggunakan Debezium dan Amazon MSK.



Cara memilih alat yang tepat untuk mengunggah massal atau memigrasi data ke Amazon Keyspaces

Di bagian ini, Anda dapat meninjau berbagai alat yang dapat Anda gunakan untuk mengunggah atau memigrasi data secara massal ke Amazon Keyspaces, dan mempelajari cara memilih alat yang benar berdasarkan kebutuhan Anda. Selain itu, bagian ini memberikan ikhtisar dan kasus penggunaan untuk tutorial langkah demi langkah yang tersedia yang menunjukkan cara mengimpor data ke Amazon Keyspaces.

Untuk meninjau strategi yang tersedia untuk memigrasikan beban kerja dari Apache Cassandra ke Amazon Keyspaces, lihat [the section called “Migrasi dari Cassandra”](#)

- Alat migrasi
 - Dengan [kalkulator harga untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#) yang tersedia di Github, Anda dapat memperkirakan biaya bulanan Anda untuk Amazon Keyspaces berdasarkan beban kerja Apache Cassandra yang ada. Masukkan metrik dari keluaran status nodetool Cassandra Anda dan konfigurasi tanpa server yang dimaksudkan untuk Amazon

Keyspaces untuk membandingkan biaya langsung antara kedua solusi. Perhatikan bahwa kalkulator ini hanya berfokus pada biaya operasional Amazon Keyspaces dibandingkan dengan penerapan Cassandra yang ada. Ini tidak termasuk faktor biaya kepemilikan total (TCO) seperti pemeliharaan infrastruktur, overhead operasional, atau biaya dukungan untuk Cassandra.

- ZDM Dual Write Proxy untuk Migrasi Keyspaces Amazon - ZDM Dual Write Proxy tersedia di Github mendukung migrasi [zero-downtime dari Apache Cassandra](#) ke Amazon Keyspaces.
- CQLReplicator - CQLReplicator adalah utilitas open source yang tersedia di [Github](#) yang membantu Anda memigrasikan data dari Apache Cassandra ke Amazon Keyspaces dalam waktu dekat.

Untuk informasi selengkapnya, lihat [the section called “Replikator CQL”](#).

- Untuk mempelajari selengkapnya tentang cara menggunakan Amazon Managed Streaming for Apache Kafka guna menerapkan proses migrasi [online](#) dengan penulisan ganda, [lihat Panduan migrasi data berkelanjutan dari Apache Cassandra](#) ke Amazon Keyspaces.
- Untuk migrasi besar, pertimbangkan untuk menggunakan alat ekstrak, transformasi, dan muat (ETL). Anda dapat menggunakannya AWS Glue untuk melakukan migrasi transformasi data dengan cepat dan efektif. Untuk informasi selengkapnya, lihat [the section called “Migrasi offline”](#).
- Untuk mempelajari cara menggunakan konektor Apache Cassandra Spark untuk menulis data ke Amazon Keyspaces, lihat [the section called “Integrasi dengan Apache Spark”](#)
- Mulailah dengan cepat dengan memuat data ke Amazon Keyspaces dengan menggunakan COPY FROM perintah cqlsh. cqlsh disertakan dengan Apache Cassandra dan paling cocok untuk memuat kumpulan data kecil atau data uji. Untuk instruksi langkah demi langkah, lihat [the section called “Memuat data menggunakan cqlsh”](#).
- Anda juga dapat menggunakan DataStax Bulk Loader untuk Apache Cassandra untuk memuat data ke Amazon Keyspaces menggunakan perintah. dsbulk [DSBulk menyediakan kemampuan impor yang lebih kuat daripada cqlsh dan tersedia dari repositori. GitHub](#) Untuk instruksi langkah demi langkah, lihat [the section called “Memuat data menggunakan DSBulk”](#).

Pertimbangan umum untuk upload data ke Amazon Keyspaces

- Pecah unggahan data menjadi komponen yang lebih kecil.

Pertimbangkan unit migrasi berikut dan jejak potensialnya dalam hal ukuran data mentah.

Mengunggah data dalam jumlah yang lebih kecil dalam satu atau beberapa fase dapat membantu menyederhanakan migrasi Anda.

- Berdasarkan cluster — Migrasikan semua data Cassandra Anda sekaligus. Pendekatan ini mungkin baik-baik saja untuk kelompok yang lebih kecil.
- Berdasarkan ruang kunci atau tabel — Pecah migrasi Anda ke dalam grup ruang kunci atau tabel. Pendekatan ini dapat membantu Anda memigrasikan data secara bertahap berdasarkan kebutuhan Anda untuk setiap beban kerja.
- Berdasarkan data — Pertimbangkan untuk memigrasikan data untuk grup pengguna atau produk tertentu, untuk menurunkan ukuran data lebih banyak lagi.
- Prioritaskan data apa yang akan diunggah terlebih dahulu berdasarkan kesederhanaan.

Pertimbangkan jika Anda memiliki data yang dapat dimigrasikan terlebih dahulu dengan lebih mudah—misalnya, data yang tidak berubah selama waktu tertentu, data dari pekerjaan batch malam hari, data yang tidak digunakan selama jam offline, atau data dari aplikasi internal.

Topik

- [Tutorial: Memuat data ke Amazon Keyspaces menggunakan cqlsh](#)
- [Tutorial: Memuat data ke Amazon Keyspaces menggunakan DSBulk](#)

Tutorial: Memuat data ke Amazon Keyspaces menggunakan cqlsh

Tutorial ini memandu Anda melalui proses migrasi data dari Apache Cassandra ke Amazon Keyspaces menggunakan perintah `cqlsh COPY FROM`. Perintah ini berguna untuk mengunggah kumpulan data kecil dengan cepat dan mudah ke Amazon Keyspaces untuk tujuan akademis atau pengujian. Untuk informasi selengkapnya tentang cara memigrasikan beban kerja produksi, lihat [the section called “Migrasi offline”](#). Dalam tutorial ini, Anda akan menyelesaikan langkah-langkah berikut:

Prasyarat - Siapkan AWS akun dengan kredensial, buat file penyimpanan kepercayaan JKS untuk sertifikat, dan konfigurasi untuk terhubung ke Amazon Keyspaces. `cqlsh`

1. Buat CSV sumber dan tabel target — Siapkan file CSV sebagai data sumber dan buat ruang kunci dan tabel target di Amazon Keyspaces.
2. Siapkan data — Acak data dalam file CSV dan analisis untuk menentukan ukuran baris rata-rata dan maksimum.
3. Atur kapasitas throughput — Hitung unit kapasitas tulis (WCU) yang diperlukan berdasarkan ukuran data dan waktu muat yang diinginkan, dan konfigurasi kapasitas yang disediakan tabel.

4. Konfigurasi parameter `cqlsh COPY FROM` — Tentukan nilai optimal untuk `cqlsh COPY FROM` parameter seperti `INGESTRATE`, `NUMPROCESSES`, `MAXBATCHSIZE`, dan `CHUNKSIZE` untuk mendistribusikan beban kerja secara merata.
5. Jalankan **`cqlsh COPY FROM`** perintah — Jalankan `cqlsh COPY FROM` perintah untuk mengunggah data dari file CSV ke tabel Amazon Keyspaces, dan pantau progres.

Pemecahan masalah — Mengatasi masalah umum seperti permintaan tidak valid, kesalahan parser, kesalahan kapasitas, dan kesalahan `cqlsh` selama proses pengunggahan data.

Topik

- [Prasyarat: Langkah-langkah yang harus diselesaikan sebelum Anda dapat mengunggah data menggunakan `cqlsh COPY FROM`](#)
- [Langkah 1: Buat file CSV sumber dan tabel target untuk unggahan data](#)
- [Langkah 2: Siapkan data sumber untuk mengunggah data yang berhasil](#)
- [Langkah 3: Atur kapasitas throughput untuk tabel](#)
- [Langkah 4: Konfigurasi `cqlsh COPY FROM` pengaturan](#)
- [Langkah 5: Jalankan `cqlsh COPY FROM` perintah untuk mengunggah data dari file CSV ke tabel target](#)
- [Pemecahan masalah](#)

Prasyarat: Langkah-langkah yang harus diselesaikan sebelum Anda dapat mengunggah data menggunakan **`cqlsh COPY FROM`**

Anda harus menyelesaikan tugas-tugas berikut sebelum Anda dapat memulai tutorial ini.

1. Jika Anda belum melakukannya, daftar Akun AWS dengan mengikuti langkah-langkah di [the section called “Penyiapan AWS Identity and Access Management”](#).
2. Buat kredensial khusus layanan dengan mengikuti langkah-langkah di [the section called “Buat kredensial khusus layanan”](#)
3. Siapkan koneksi shell Cassandra Query Language (`cqlsh`) dan konfirmasi bahwa Anda dapat terhubung ke Amazon Keyspaces dengan mengikuti langkah-langkah di [the section called “Menggunakan `cqlsh`”](#)

Langkah 1: Buat file CSV sumber dan tabel target untuk unggahan data

Untuk tutorial ini, kita menggunakan file nilai dipisahkan koma (CSV) dengan nama `keyspaces_sample_table.csv` sebagai file sumber untuk migrasi data. File sampel yang disediakan berisi beberapa baris data untuk tabel dengan nama `book_awards`.

1. Buat file sumber. Anda dapat memilih salah satu opsi berikut:

- Download contoh file CSV (`keyspaces_sample_table.csv`) yang terkandung dalam file arsip berikut [samplemigration.zip](#). Buka zip arsip dan catat jalur ke `keyspaces_sample_table.csv`.
- Untuk mengisi file CSV dengan data Anda sendiri yang disimpan dalam database Apache Cassandra, Anda dapat mengisi file CSV sumber dengan menggunakan `cqlsh COPY TO` pernyataan seperti yang ditunjukkan pada contoh berikut.

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

Pastikan file CSV yang Anda buat memenuhi persyaratan berikut:

- Baris pertama berisi nama kolom.
- Nama kolom dalam file CSV sumber cocok dengan nama kolom di tabel target.
- Data dibatasi dengan koma.
- Semua nilai data adalah tipe data Amazon Keyspaces yang valid. Lihat [the section called "Jenis Data"](#).

2. Buat keyspace target dan tabel di Amazon Keyspaces.

- a. Connect ke Amazon Keyspaces menggunakan `cqlsh`, mengganti endpoint layanan, nama pengguna, dan kata sandi dalam contoh berikut dengan nilai Anda sendiri.

```
cqlsh cassandra.us-east-1.amazonaws.com 9142 -u "111122223333" -p  
"wJa1rXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" --ssl
```

- b. Buat keyspace baru dengan nama `catalog` seperti yang ditunjukkan pada contoh berikut.

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

- c. Ketika keyspace baru tersedia, gunakan kode berikut untuk membuat tabel `book_awards` target.

```
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)  
);
```

Jika Apache Cassandra adalah sumber data asli Anda, cara sederhana untuk membuat tabel target Amazon Keyspaces dengan header yang cocok adalah dengan menghasilkan CREATE TABLE pernyataan dari tabel sumber, seperti yang ditunjukkan dalam pernyataan berikut.

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

Kemudian buat tabel target di Amazon Keyspaces dengan nama kolom dan tipe data yang cocok dengan deskripsi dari tabel sumber Cassandra.

Langkah 2: Siapkan data sumber untuk mengunggah data yang berhasil

Mempersiapkan data sumber untuk transfer yang efisien adalah proses dua langkah. Pertama, Anda mengacak data. Pada langkah kedua, Anda menganalisis data untuk menentukan nilai `cqlsh COPY` parameter yang sesuai dan pengaturan tabel yang diperlukan untuk memastikan bahwa unggahan data berhasil.

Acak data

`cqlsh COPY FROM` perintah membaca dan menulis data dalam urutan yang sama seperti yang muncul di file CSV. Jika Anda menggunakan `cqlsh COPY TO` perintah untuk membuat file sumber, data ditulis dalam urutan kunci yang diurutkan di CSV. Secara internal, Amazon Keyspaces mempartisi data menggunakan tombol partisi. Meskipun Amazon Keyspaces memiliki logika bawaan untuk membantu memuat permintaan keseimbangan untuk kunci partisi yang sama, memuat data lebih cepat dan lebih efisien jika Anda mengacak urutan. Ini karena Anda dapat memanfaatkan

penyeimbangan beban bawaan yang terjadi saat Amazon Keyspaces menulis ke partisi yang berbeda.

Untuk menyebarkan tulisan di seluruh partisi secara merata, Anda harus mengacak data dalam file sumber. Anda dapat menulis aplikasi untuk melakukan ini atau menggunakan alat sumber terbuka, seperti [Shuf](#). Shuf tersedia secara bebas di distribusi Linux, di macOS (dengan instalasi [coreutils](#) di [homebrew](#)), dan di Windows (dengan menggunakan Windows Subsystem for Linux (WSL)). Satu langkah tambahan diperlukan untuk mencegah baris header dengan nama kolom diacak pada langkah ini.

Untuk mengacak file sumber sambil mempertahankan header, masukkan kode berikut.

```
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 menulis ulang data ke file CSV baru yang disebut `keyspace.table.csv`. Anda sekarang dapat menghapus `keyspaces_sample_table.csv` file — Anda tidak lagi membutuhkannya.

Menganalisis data

Tentukan ukuran baris rata-rata dan maksimum dengan menganalisis data.

Anda melakukan ini karena alasan berikut:

- Ukuran baris rata-rata membantu memperkirakan jumlah total data yang akan ditransfer.
- Anda memerlukan ukuran baris rata-rata untuk menyediakan kapasitas tulis yang diperlukan untuk unggahan data.
- Anda dapat memastikan bahwa setiap baris berukuran kurang dari 1 MB, yang merupakan ukuran baris maksimum di Amazon Keyspaces.

Note

Kuota ini mengacu pada ukuran baris, bukan ukuran partisi. Tidak seperti partisi Apache Cassandra, partisi Amazon Keyspaces hampir tidak terikat ukurannya. Kunci partisi dan kolom pengelompokan memerlukan penyimpanan tambahan untuk metadata, yang harus Anda tambahkan ke ukuran baris mentah. Untuk informasi selengkapnya, lihat [the section called “Perkiraan ukuran baris”](#).

Kode berikut menggunakan [AWK](#) untuk menganalisis file CSV dan mencetak ukuran baris rata-rata dan maksimum.

```
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
```

Menjalankan kode ini menghasilkan output berikut.

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

Anda menggunakan ukuran baris rata-rata pada langkah berikutnya dari tutorial ini untuk menyediakan kapasitas tulis untuk tabel.

Langkah 3: Atur kapasitas throughput untuk tabel

Tutorial ini menunjukkan cara menyetel `cqlsh` untuk memuat data dalam rentang waktu yang ditetapkan. Karena Anda tahu berapa banyak membaca dan menulis yang Anda lakukan sebelumnya, gunakan mode kapasitas yang disediakan. Setelah Anda menyelesaikan transfer data, Anda harus mengatur mode kapasitas tabel agar sesuai dengan pola lalu lintas aplikasi Anda. Untuk mempelajari lebih lanjut tentang manajemen kapasitas, lihat [Mengelola sumber daya tanpa server](#).

Dengan mode kapasitas yang disediakan, Anda menentukan berapa banyak kapasitas baca dan tulis yang ingin Anda berikan ke tabel Anda sebelumnya. Kapasitas tulis ditagih per jam dan diukur dalam unit kapasitas tulis (WCU). Setiap WCU memiliki kapasitas tulis yang cukup untuk mendukung penulisan 1 KB data per detik. Saat Anda memuat data, laju penulisan harus berada di bawah WCU maks (parameter: `write_capacity_units`) yang ditetapkan pada tabel target.

Secara default, Anda dapat menyediakan hingga 40.000 WCU ke tabel dan 80.000 WCU di semua tabel di akun Anda. Jika Anda membutuhkan kapasitas tambahan, Anda dapat meminta peningkatan kuota di konsol [Service Quotas](#). Untuk informasi lebih lanjut tentang kuota, lihat [Kuota](#).

Hitung jumlah rata-rata WCU yang diperlukan untuk sisipan

Memasukkan 1 KB data per detik membutuhkan 1 WCU. Jika file CSV Anda memiliki 360.000 baris dan Anda ingin memuat semua data dalam 1 jam, Anda harus menulis 100 baris per detik (360.000 baris / 60 menit / 60 detik = 100 baris per detik). Jika setiap baris memiliki data hingga 1 KB, untuk menyisipkan 100 baris per detik, Anda harus menyediakan 100 WCU ke tabel Anda. Jika setiap baris

memiliki 1,5 KB data, Anda memerlukan dua WCU untuk memasukkan satu baris per detik. Oleh karena itu, untuk memasukkan 100 baris per detik, Anda harus menyediakan 200 WCU.

Untuk menentukan berapa banyak WCU yang perlu Anda masukkan satu baris per detik, bagi ukuran baris rata-rata dalam byte dengan 1024 dan bulatkan ke bilangan bulat terdekat.

Misalnya, jika ukuran baris rata-rata adalah 3000 byte, Anda memerlukan tiga WCU untuk menyisipkan satu baris per detik.

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

Hitung waktu dan kapasitas pemuatan data

Sekarang setelah Anda mengetahui ukuran rata-rata dan jumlah baris dalam file CSV Anda, Anda dapat menghitung berapa banyak WCU yang Anda butuhkan untuk memuat data dalam jumlah waktu tertentu, dan perkiraan waktu yang diperlukan untuk memuat semua data dalam file CSV Anda menggunakan pengaturan WCU yang berbeda.

Misalnya, jika setiap baris dalam file Anda adalah 1 KB dan Anda memiliki 1.000.000 baris dalam file CSV Anda, untuk memuat data dalam 1 jam, Anda perlu menyediakan setidaknya 278 WCU ke tabel Anda selama jam itu.

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

Konfigurasi pengaturan kapasitas yang disediakan

Anda dapat mengatur pengaturan kapasitas tulis tabel saat Anda membuat tabel atau dengan menggunakan perintah ALTER TABLE CQL. Berikut ini adalah sintaks untuk mengubah pengaturan kapasitas disediakan tabel dengan pernyataan CQL. ALTER TABLE

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

Untuk referensi bahasa selengkapnya, lihat [the section called “ALTER TABLE”](#).

Langkah 4: Konfigurasi **cqlsh COPY FROM** pengaturan

Bagian ini menguraikan cara menentukan nilai parameter untuk **cqlsh COPY FROM**. **cqlsh COPY FROM** Perintah membaca file CSV yang Anda siapkan sebelumnya dan menyisipkan data ke Amazon

Keyspaces menggunakan CQL. Perintah membagi baris dan mendistribusikan INSERT operasi di antara satu set pekerja. Setiap pekerja membuat koneksi dengan Amazon Keyspaces dan INSERT mengirimkan permintaan di sepanjang saluran ini.

`cqlsh COPY` Perintah tidak memiliki logika internal untuk mendistribusikan pekerjaan secara merata di antara para pekerjanya. Namun, Anda dapat mengonfigurasinya secara manual untuk memastikan bahwa pekerjaan didistribusikan secara merata. Mulailah dengan meninjau parameter `cqlsh` kunci ini:

- **DELIMITER** — Jika Anda menggunakan pembatas selain koma, Anda dapat mengatur parameter ini, yang defaultnya koma.
- **INGESTRATE** — Jumlah target baris yang `cqlsh COPY FROM` mencoba memproses per detik. Jika tidak disetel, defaultnya menjadi 100.000.
- **NUMPROCESSES** — Jumlah proses pekerja anak yang dibuat `cqlsh` untuk tugas. `COPY FROM` Maksimum untuk pengaturan ini adalah 16, defaultnya adalah `num_cores - 1`, di mana `num_cores` jumlah inti pemrosesan pada host yang menjalankan `cqlsh`.
- **MAXBATCHSIZE** — Ukuran batch menentukan jumlah maksimal baris yang dimasukkan ke dalam tabel tujuan dalam satu batch. Jika tidak disetel, `cqlsh` menggunakan batch 20 baris yang disisipkan.
- **CHUNKSIZE** — Ukuran unit kerja yang diteruskan ke pekerja anak. Secara default, ini diatur ke 5.000.
- **MAKSIMAL** — Jumlah maksimum kali untuk mencoba kembali potongan pekerja yang gagal. Setelah upaya maksimum tercapai, catatan yang gagal ditulis ke file CSV baru yang dapat Anda jalankan lagi nanti setelah menyelidiki kegagalan.

Tetapkan **INGESTRATE** berdasarkan jumlah WCU yang Anda berikan ke tabel tujuan target.

`INGESTRATE cqlsh COPY FROM` Perintah bukanlah batas—ini adalah rata-rata target. Ini berarti dapat (dan sering) meledak di atas angka yang Anda tetapkan. Untuk memungkinkan ledakan dan memastikan bahwa kapasitas yang cukup tersedia untuk menangani permintaan pemuatan data, atur **INGESTRATE** ke 90% dari kapasitas tulis tabel.

$$\text{INGESTRATE} = \text{WCUs} * .90$$

Selanjutnya, atur **NUMPROCESSES** parameter menjadi sama dengan satu kurang dari jumlah core pada sistem Anda. Untuk mengetahui berapa jumlah core sistem Anda, Anda dapat menjalankan kode berikut.

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

Untuk tutorial ini, kami menggunakan nilai berikut.

```
NUMPROCESSES = 4
```

Setiap proses membuat pekerja, dan setiap pekerja membuat koneksi ke Amazon Keyspaces. Amazon Keyspaces dapat mendukung hingga 3.000 permintaan CQL per detik pada setiap koneksi. Ini berarti Anda harus memastikan bahwa setiap pekerja memproses kurang dari 3.000 permintaan per detik.

Seperti halnya `INGESTRATE`, pekerja sering meledak di atas angka yang Anda tetapkan dan tidak dibatasi oleh detik jam. Oleh karena itu, untuk memperhitungkan semburan, atur parameter `cqlsh` Anda untuk menargetkan setiap pekerja untuk memproses 2.500 permintaan per detik. Untuk menghitung jumlah pekerjaan yang didistribusikan kepada pekerja, gunakan pedoman berikut.

- Bagilah `INGESTRATE` dengan `NUMPROCESSES`.
- Jika $INGESTRATE / NUMPROCESSES > 2.500$, turunkan `INGESTRATE` untuk membuat rumus ini benar.

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

Sebelum Anda mengonfigurasi pengaturan untuk mengoptimalkan unggahan data sampel kami, mari tinjau pengaturan `cqlsh` default dan lihat bagaimana penggunaannya memengaruhi proses pengunggahan data. Karena `cqlsh COPY FROM` menggunakan `CHUNKSIZE` untuk membuat potongan pekerjaan (`INSERT` pernyataan) untuk didistribusikan kepada pekerja, pekerjaan tidak secara otomatis didistribusikan secara merata. Beberapa pekerja mungkin duduk diam, tergantung pada `INGESTRATE` pengaturannya.

Untuk mendistribusikan pekerjaan secara merata di antara para pekerja dan menjaga setiap pekerja pada tingkat optimal 2.500 permintaan per detik, Anda harus mengatur `CHUNKSIZE`, `MAXBATCHSIZE`, dan `INGESTRATE` dengan mengubah parameter input. Untuk mengoptimalkan pemanfaatan lalu lintas jaringan selama pemuatan data, pilih nilai `MAXBATCHSIZE` yang mendekati nilai maksimum 30. Dengan mengubah `CHUNKSIZE` ke 100 dan `MAXBATCHSIZE` ke 25, 10.000 baris tersebar merata di antara empat pekerja ($10.000 / 2500 = 4$).

Contoh kode berikut menggambarkan hal ini.

```
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
```

Untuk meringkas, gunakan rumus berikut saat mengatur `cqlsh COPY FROM` parameter:

- `INGESTRATE` = `write_capacity_units` * .90
- `NUMPROCESSES` = `num_cores` -1 (default)
- `INGESTRATE/NUMPROCESSES` = 2,500 (Ini harus menjadi pernyataan yang benar.)
- `MAXBATCHSIZE` = 30 (Default ke 20. Amazon Keyspaces menerima batch hingga 30.)
- `CHUNKSIZE` = (`MENELAN/NUMPROCESSES`)/`MAXBATCHSIZE`

Sekarang Anda telah menghitung `NUMPROCESSES`, `INGESTRATE`, dan `CHUNKSIZE`, Anda siap untuk memuat data Anda.

Langkah 5: Jalankan **cqlsh COPY FROM** perintah untuk mengunggah data dari file CSV ke tabel target

Untuk menjalankan `cqlsh COPY FROM` perintah, selesaikan langkah-langkah berikut.

1. Connect ke Amazon Keyspaces menggunakan `cqlsh`.
2. Pilih ruang kunci Anda dengan kode berikut.

```
USE catalog;
```

3. Tetapkan konsistensi tulis ke `LOCAL_QUORUM`. Untuk memastikan daya tahan data, Amazon Keyspaces tidak mengizinkan pengaturan konsistensi tulis lainnya. Lihat kode berikut.

```
CONSISTENCY LOCAL_QUORUM;
```

4. Siapkan `cqlsh COPY FROM` sintaks Anda menggunakan contoh kode berikut.

```
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. Jalankan pernyataan yang disiapkan pada langkah sebelumnya. cqlsh menggemakan kembali semua pengaturan yang telah Anda konfigurasi.

- a. Pastikan pengaturan sesuai dengan input Anda. Lihat contoh berikut ini.

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

- b. Tinjau jumlah baris yang ditransfer dan tingkat rata-rata saat ini, seperti yang ditunjukkan pada contoh berikut.

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

- c. Ketika cqlsh selesai mengunggah data, tinjau ringkasan statistik pemuatan data (jumlah file yang dibaca, runtime, dan baris yang dilewati) seperti yang ditunjukkan pada contoh berikut.

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

Pada langkah terakhir tutorial ini, Anda telah mengunggah data ke Amazon Keyspaces.

Important

Sekarang setelah Anda mentransfer data Anda, sesuaikan pengaturan mode kapasitas tabel target Anda agar sesuai dengan pola lalu lintas reguler aplikasi Anda. Anda dikenakan biaya pada tarif per jam untuk kapasitas yang Anda berikan sampai Anda mengubahnya.

Pemecahan masalah

Setelah pengunggahan data selesai, periksa untuk melihat apakah baris dilewati. Untuk melakukannya, navigasikan ke direktori sumber file CSV sumber dan cari file dengan nama berikut.

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

cqlsh menulis setiap baris data yang dilewati ke dalam file dengan nama itu. Jika file ada di direktori sumber Anda dan memiliki data di dalamnya, baris ini tidak diunggah ke Amazon Keyspaces. Untuk mencoba lagi baris ini, pertama-tama periksa kesalahan yang ditemui selama pengunggahan dan sesuaikan data yang sesuai. Untuk mencoba lagi baris ini, Anda dapat menjalankan kembali prosesnya.

Kesalahan umum

Alasan paling umum mengapa baris tidak dimuat adalah kesalahan kapasitas dan kesalahan penguraian.

Kesalahan permintaan tidak valid saat mengunggah data ke Amazon Keyspaces

Dalam contoh berikut, tabel sumber berisi kolom penghitung, yang menghasilkan panggilan batch yang dicatat dari perintah cqlshCOPY. Panggilan batch yang dicatat tidak didukung oleh Amazon Keyspaces.

```
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
```

Untuk mengatasi kesalahan ini, gunakan DSBulk untuk memigrasikan data. Untuk informasi selengkapnya, lihat [the section called “Memuat data menggunakan DSBulk”](#).

Kesalahan parser saat mengunggah data ke Amazon Keyspaces

Contoh berikut menunjukkan baris yang dilewati karena aParseError.

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

Untuk mengatasi kesalahan ini, Anda perlu memastikan bahwa data yang akan diimpor cocok dengan skema tabel di Amazon Keyspaces. Tinjau file impor untuk kesalahan penguraian. Anda dapat mencoba menggunakan satu baris data menggunakan INSERT pernyataan untuk mengisolasi kesalahan.

Kesalahan kapasitas saat mengunggah data ke 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 menggunakan WriteTimeout pengecualian ReadTimeout dan untuk menunjukkan kapan permintaan tulis gagal karena kapasitas throughput yang tidak mencukupi. Untuk membantu mendiagnosis pengecualian kapasitas yang tidak mencukupi, Amazon Keyspaces WriteThrottleEvents menerbitkan dan ReadThrottledEvents metrik di Amazon CloudWatch. Untuk informasi selengkapnya, lihat [the section called “Pemantauan CloudWatch dengan”](#).

kesalahan cqlsh saat mengunggah data ke Amazon Keyspaces

Untuk membantu memecahkan masalah kesalahan cqlsh, jalankan kembali perintah yang gagal dengan bendera. --debug

Saat menggunakan versi cqlsh yang tidak kompatibel, Anda melihat kesalahan berikut.

```
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
```

Konfirmasikan bahwa versi cqlsh yang benar diinstal dengan menjalankan perintah berikut.

```
cqlsh --version
```

Anda akan melihat sesuatu seperti berikut untuk output.

```
cqlsh 5.0.1
```

Jika Anda menggunakan Windows, ganti semua instance cqlsh dengan cqlsh.bat. Misalnya, untuk memeriksa versi cqlsh di Windows, jalankan perintah berikut.

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

Koneksi ke Amazon Keyspaces gagal setelah klien cqlsh menerima tiga kesalahan berturut-turut dari jenis apa pun dari server. Klien cqlsh gagal dengan pesan berikut.

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

Untuk mengatasi kesalahan ini, Anda perlu memastikan bahwa data yang akan diimpor cocok dengan skema tabel di Amazon Keyspaces. Tinjau file impor untuk kesalahan penguraian. Anda dapat mencoba menggunakan satu baris data dengan menggunakan pernyataan INSERT untuk mengisolasi kesalahan.

Klien secara otomatis mencoba membangun kembali koneksi.

Tutorial: Memuat data ke Amazon Keyspaces menggunakan DSBulk

Tutorial langkah demi langkah ini memandu Anda melalui migrasi data dari Apache Cassandra ke Amazon Keyspaces menggunakan Bulk Loader (DSBulk) DataStax yang tersedia di [GitHub](#). Menggunakan DSBulk berguna untuk mengunggah kumpulan data ke Amazon Keyspaces untuk tujuan akademik atau pengujian. Untuk informasi selengkapnya tentang cara memigrasikan beban kerja produksi, lihat [the section called “Migrasi offline”](#). Dalam tutorial ini, Anda menyelesaikan langkah-langkah berikut.

Prasyarat - Siapkan AWS akun dengan kredensial, buat file penyimpanan kepercayaan JKS untuk sertifikat, konfigurasi, unduh dan instal DSBulk, dan konfigurasi `cqlsh` file. `application.conf`

1. Buat CSV sumber dan tabel target — Siapkan file CSV sebagai data sumber dan buat ruang kunci dan tabel target di Amazon Keyspaces.
2. Siapkan data — Acak data dalam file CSV dan analisis untuk menentukan ukuran baris rata-rata dan maksimum.
3. Atur kapasitas throughput — Hitung unit kapasitas tulis (WCU) yang diperlukan berdasarkan ukuran data dan waktu muat yang diinginkan, dan konfigurasi kapasitas yang disediakan tabel.
4. Konfigurasi pengaturan DSBulk - Buat file konfigurasi DSBulk dengan pengaturan seperti otentikasi, SSL/TLS, tingkat konsistensi, dan ukuran kumpulan koneksi.
5. Jalankan perintah pemuatan DSBulk - Jalankan perintah pemuatan DSBulk untuk mengunggah data dari file CSV ke tabel Amazon Keyspaces, dan pantau kemajuannya.

Topik

- [Prasyarat: Langkah-langkah yang harus Anda selesaikan sebelum Anda dapat mengunggah data dengan DSBulk](#)
- [Langkah 1: Buat file CSV sumber dan tabel target untuk unggahan data menggunakan DSBulk](#)
- [Langkah 2: Siapkan data untuk diunggah menggunakan DSBulk](#)

- [Langkah 3: Atur kapasitas throughput untuk tabel target](#)
- [Langkah 4: Konfigurasi DSBulk pengaturan untuk mengunggah data dari file CSV ke tabel target](#)
- [Langkah 5: Jalankan load perintah DSBulk untuk mengunggah data dari file CSV ke tabel target](#)

Prasyarat: Langkah-langkah yang harus Anda selesaikan sebelum Anda dapat mengunggah data dengan DSBulk

Anda harus menyelesaikan tugas-tugas berikut sebelum Anda dapat memulai tutorial ini.

1. Jika Anda belum melakukannya, daftar AWS akun dengan mengikuti langkah-langkah di [the section called “Penyiapan AWS Identity and Access Management”](#).
2. Buat kredensi dengan mengikuti langkah-langkah di [the section called “Buat kredensi IAM untuk otentikasi AWS”](#)
3. Buat file penyimpanan kepercayaan JKS.
 - a. Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.

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

 Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang

dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

- b. Ubah sertifikat digital menjadi file TrustStore dan tambahkan ke keystore.

```
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
```

Pada langkah terakhir, Anda perlu membuat kata sandi untuk keystore dan mempercayai setiap sertifikat. Perintah interaktif terlihat seperti ini.

```
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. Siapkan koneksi shell Cassandra Query Language (cqlsh) dan konfirmasikan bahwa Anda dapat terhubung ke Amazon Keyspaces dengan mengikuti langkah-langkah di [the section called “Menggunakan cqlsh”](#)
5. Unduh dan instal DSBulk.

Note

Versi yang ditampilkan dalam tutorial ini mungkin bukan versi terbaru yang tersedia. Sebelum Anda mengunduh DSBulk, periksa [halaman unduhan DataStax Bulk Loader](#) untuk versi terbaru, dan perbarui nomor versi dalam perintah berikut.

- a. Untuk mengunduh DSBulk, Anda dapat menggunakan kode berikut.

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

- b. Kemudian buka paket file tar dan tambahkan DSBulk ke Anda PATH seperti yang ditunjukkan pada contoh berikut.

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

- c. Buat `application.conf` file untuk menyimpan pengaturan yang akan digunakan oleh DSBulk. Anda dapat menyimpan contoh berikut sebagai `./dsbulk_keyspaces.conf`. Ganti `localhost` dengan titik kontak cluster Cassandra lokal Anda jika Anda tidak berada

di node lokal, misalnya nama DNS atau alamat IP. Catat nama file dan jalur, karena Anda akan perlu menentukan ini nanti dalam dsbulk load perintah.

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

- d. Untuk mengaktifkan dukungan SigV4, unduh jar file yang diarsir dari [GitHub](#) dan letakkan di lib folder DSBulk seperti yang ditunjukkan pada contoh berikut.

```
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
```

Langkah 1: Buat file CSV sumber dan tabel target untuk unggahan data menggunakan DSBulk

Untuk tutorial ini, kita menggunakan file nilai dipisahkan koma (CSV) dengan nama `keyspaces_sample_table.csv` sebagai file sumber untuk migrasi data. File sampel yang disediakan berisi beberapa baris data untuk tabel dengan nama `book_awards`.

1. Buat file sumber. Anda dapat memilih salah satu opsi berikut:
 - Download contoh file CSV (`keyspaces_sample_table.csv`) yang terkandung dalam file arsip berikut [samplemigration.zip](#). Buka zip arsip dan catat jalur ke `keyspaces_sample_table.csv`.
 - Untuk mengisi file CSV dengan data Anda sendiri yang disimpan dalam database Apache Cassandra, Anda dapat mengisi file CSV sumber dengan menggunakan dsbulk unload seperti yang ditunjukkan pada contoh berikut.

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

Pastikan file CSV yang Anda buat memenuhi persyaratan berikut:

- Baris pertama berisi nama kolom.
- Nama kolom dalam file CSV sumber cocok dengan nama kolom di tabel target.
- Data dibatasi dengan koma.
- Semua nilai data adalah tipe data Amazon Keyspaces yang valid. Lihat [the section called “Jenis Data”](#).

2. Buat keyspace target dan tabel di Amazon Keyspaces.

- a. Connect ke Amazon Keyspaces menggunakan `cqlsh`, mengganti endpoint layanan, nama pengguna, dan kata sandi dalam contoh berikut dengan nilai Anda sendiri.

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

- b. Buat keyspace baru dengan nama `catalog` seperti yang ditunjukkan pada contoh berikut.

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

- c. Setelah keyspace baru memiliki status yang tersedia, gunakan kode berikut untuk membuat tabel `book_awards` target. Untuk mempelajari lebih lanjut tentang pembuatan sumber daya asinkron dan cara memeriksa apakah sumber daya tersedia, lihat [the section called “Periksa status pembuatan keyspace”](#)

```
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)  
);
```

Jika Apache Cassandra adalah sumber data asli Anda, cara sederhana untuk membuat tabel target Amazon Keyspaces dengan header yang cocok adalah dengan menghasilkan `CREATE TABLE` pernyataan dari tabel sumber seperti yang ditunjukkan dalam pernyataan berikut.

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

Kemudian buat tabel target di Amazon Keyspaces dengan nama kolom dan tipe data yang cocok dengan deskripsi dari tabel sumber Cassandra.

Langkah 2: Siapkan data untuk diunggah menggunakan DSBulk

Mempersiapkan data sumber untuk transfer yang efisien adalah proses dua langkah. Pertama, Anda mengacak data. Pada langkah kedua, Anda menganalisis data untuk menentukan nilai `dsbulk` parameter yang sesuai dan pengaturan tabel yang diperlukan.

Mengacak data

`dsbulk`Perintah membaca dan menulis data dalam urutan yang sama seperti yang muncul di file CSV. Jika Anda menggunakan `dsbulk` perintah untuk membuat file sumber, data ditulis dalam urutan kunci yang diurutkan di CSV. Secara internal, Amazon Keyspaces mempartisi data menggunakan tombol partisi. Meskipun Amazon Keyspaces memiliki logika bawaan untuk membantu memuat permintaan keseimbangan untuk kunci partisi yang sama, memuat data lebih cepat dan lebih efisien jika Anda mengacak urutan. Ini karena Anda dapat memanfaatkan penyeimbangan beban bawaan yang terjadi saat Amazon Keyspaces menulis ke partisi yang berbeda.

Untuk menyebarkan tulisan di seluruh partisi secara merata, Anda harus mengacak data dalam file sumber. Anda dapat menulis aplikasi untuk melakukan ini atau menggunakan alat sumber terbuka, seperti [Shuf](#). `Shuf` tersedia secara bebas di distribusi Linux, di macOS (dengan menginstal `coreutils` di [homebrew](#)), dan di Windows (dengan menggunakan Windows Subsystem for Linux (WSL)). Satu langkah tambahan diperlukan untuk mencegah baris header dengan nama kolom diacak pada langkah ini.

Untuk mengacak file sumber sambil mempertahankan header, masukkan kode berikut.

```
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` menulis ulang data ke file CSV baru yang disebut `keyspace.table.csv` Anda sekarang dapat menghapus `keyspaces_sample_table.csv` file — Anda tidak lagi membutuhkannya.

Menganalisis data

Tentukan ukuran baris rata-rata dan maksimum dengan menganalisis data.

Anda melakukan ini karena alasan berikut:

- Ukuran baris rata-rata membantu memperkirakan jumlah total data yang akan ditransfer.
- Anda memerlukan ukuran baris rata-rata untuk menyediakan kapasitas tulis yang diperlukan untuk unggahan data.
- Anda dapat memastikan bahwa setiap baris berukuran kurang dari 1 MB, yang merupakan ukuran baris maksimum di Amazon Keyspaces.

Note

Kuota ini mengacu pada ukuran baris, bukan ukuran partisi. Tidak seperti partisi Apache Cassandra, partisi Amazon Keyspaces hampir tidak terikat ukurannya. Kunci partisi dan kolom pengelompokan memerlukan penyimpanan tambahan untuk metadata, yang harus Anda tambahkan ke ukuran baris mentah. Untuk informasi selengkapnya, lihat [the section called “Perkiraan ukuran baris”](#).

Kode berikut menggunakan [AWK](#) untuk menganalisis file CSV dan mencetak ukuran baris rata-rata dan maksimum.

```
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
```

Menjalankan kode ini menghasilkan output berikut.

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

Pastikan ukuran baris maksimum Anda tidak melebihi 1 MB. Jika ya, Anda harus memecah baris atau mengompres data untuk membawa ukuran baris di bawah 1 MB. Pada langkah berikutnya dari tutorial ini, Anda menggunakan ukuran baris rata-rata untuk menyediakan kapasitas tulis untuk tabel.

Langkah 3: Atur kapasitas throughput untuk tabel target

Tutorial ini menunjukkan cara menyetel DSBulk untuk memuat data dalam rentang waktu yang ditetapkan. Karena Anda tahu berapa banyak membaca dan menulis yang Anda lakukan sebelumnya, gunakan mode kapasitas yang disediakan. Setelah Anda menyelesaikan transfer data, Anda harus mengatur mode kapasitas tabel agar sesuai dengan pola lalu lintas aplikasi Anda. Untuk mempelajari lebih lanjut tentang manajemen kapasitas, lihat [Mengelola sumber daya tanpa server](#).

Dengan mode kapasitas yang disediakan, Anda menentukan berapa banyak kapasitas baca dan tulis yang ingin Anda berikan ke tabel Anda sebelumnya. Kapasitas tulis ditagih per jam dan diukur dalam unit kapasitas tulis (WCU). Setiap WCU memiliki kapasitas tulis yang cukup untuk mendukung penulisan 1 KB data per detik. Saat Anda memuat data, laju penulisan harus berada di bawah WCU maks (parameter: `write_capacity_units`) yang ditetapkan pada tabel target.

Secara default, Anda dapat menyediakan hingga 40.000 WCU ke tabel dan 80.000 WCU di semua tabel di akun Anda. Jika Anda membutuhkan kapasitas tambahan, Anda dapat meminta peningkatan kuota di konsol [Service](#) Quotas. Untuk informasi lebih lanjut tentang kuota, lihat [Kuota](#).

Hitung jumlah rata-rata WCU yang diperlukan untuk sisipan

Memasukkan 1 KB data per detik membutuhkan 1 WCU. Jika file CSV Anda memiliki 360.000 baris dan Anda ingin memuat semua data dalam 1 jam, Anda harus menulis 100 baris per detik (360.000 baris / 60 menit / 60 detik = 100 baris per detik). Jika setiap baris memiliki data hingga 1 KB, untuk menyisipkan 100 baris per detik, Anda harus menyediakan 100 WCU ke tabel Anda. Jika setiap baris memiliki 1,5 KB data, Anda memerlukan dua WCU untuk memasukkan satu baris per detik. Oleh karena itu, untuk memasukkan 100 baris per detik, Anda harus menyediakan 200 WCU.

Untuk menentukan berapa banyak WCU yang perlu Anda masukkan satu baris per detik, bagi ukuran baris rata-rata dalam byte dengan 1024 dan bulatkan ke seluruh nomor terdekat.

Misalnya, jika ukuran baris rata-rata adalah 3000 byte, Anda memerlukan tiga WCU untuk menyisipkan satu baris per detik.

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

Hitung waktu dan kapasitas pemuatan data

Sekarang setelah Anda mengetahui ukuran rata-rata dan jumlah baris dalam file CSV Anda, Anda dapat menghitung berapa banyak WCU yang Anda butuhkan untuk memuat data dalam jumlah

waktu tertentu, dan perkiraan waktu yang diperlukan untuk memuat semua data dalam file CSV Anda menggunakan pengaturan WCU yang berbeda.

Misalnya, jika setiap baris dalam file Anda adalah 1 KB dan Anda memiliki 1.000.000 baris dalam file CSV Anda, untuk memuat data dalam 1 jam, Anda perlu menyediakan setidaknya 278 WCU ke tabel Anda selama jam itu.

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

Konfigurasi pengaturan kapasitas yang disediakan

Anda dapat mengatur pengaturan kapasitas tulis tabel saat Anda membuat tabel atau dengan menggunakan ALTER TABLE perintah. Berikut ini adalah sintaks untuk mengubah pengaturan kapasitas disediakan tabel dengan perintah. ALTER TABLE

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

Untuk referensi bahasa lengkap, lihat [the section called “CREATE TABLE”](#) dan [the section called “ALTER TABLE”](#).

Langkah 4: Konfigurasi **DSBulk** pengaturan untuk mengunggah data dari file CSV ke tabel target

Bagian ini menguraikan langkah-langkah yang diperlukan untuk mengonfigurasi DSBulk untuk mengunggah data ke Amazon Keyspaces. Anda mengkonfigurasi DSBulk dengan menggunakan file konfigurasi. Anda menentukan file konfigurasi langsung dari baris perintah.

1. Buat file konfigurasi DSBulk untuk migrasi ke Amazon Keyspaces, dalam contoh ini kami menggunakan nama file. `dsbulk_keyspaces.conf` Tentukan pengaturan berikut di file konfigurasi DSBulk.
 - a. *PlainTextAuthProvider*— Buat penyedia otentikasi dengan `PlainTextAuthProvider` kelas. `ServiceUserName` dan `ServicePassword` harus cocok dengan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkah di [the section called “Buat kredensial akses terprogram”](#)

- b. *local-datacenter*— Tetapkan nilai *local-datacenter* untuk Wilayah AWS yang Anda sambungkan. Misalnya, jika aplikasi terhubung ke `cassandra.us-east-1.amazonaws.com`, maka atur pusat data lokal ke `us-east-1`. Untuk semua yang tersedia Wilayah AWS, lihat [the section called “Titik akhir layanan”](#). Untuk menghindari replika, atur `slow-replica-avoidance` ke `false`.
- c. *SSLConnectionFactory*— Untuk mengkonfigurasi SSL/TLS, inisialisasi *SSLConnectionFactory* dengan menambahkan bagian dalam file konfigurasi dengan satu baris yang menentukan kelas dengan `class = DefaultSslConnectionFactory`. Berikan jalur ke `cassandra_truststore.jks` dan kata sandi yang Anda buat sebelumnya.
- d. *consistency*— Tetapkan tingkat konsistensi ke `LOCAL_QUORUM`. Tingkat konsistensi penulisan lainnya tidak didukung, untuk informasi lebih lanjut lihat [the section called “Tingkat konsistensi Cassandra yang didukung”](#).
- e. Jumlah koneksi per pool dapat dikonfigurasi di driver Java. Untuk contoh ini, atur `advanced.connection.pool.local.size` ke 3.

Berikut ini adalah file konfigurasi sampel lengkap.

```
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 = DefaultSslConnectionFactory
    truststore-path = "./cassandra_truststore.jks"
    truststore-password = "my_password"
    hostname-validation = false
  }
}
```

```
}  
advanced.connection.pool.local.size = 3  
}
```

2. Tinjau parameter untuk load perintah DSBulk.

- a. *executor.maxPerSecond*— Jumlah maksimum baris yang coba diproses oleh perintah load secara bersamaan per detik. Jika tidak disetel, pengaturan ini dinonaktifkan dengan -1.

Tetapkan *executor.maxPerSecond* berdasarkan jumlah WCU yang Anda berikan ke tabel tujuan target. *executor.maxPerSecondloadPerintah* bukanlah batas — ini adalah rata-rata target. Ini berarti dapat (dan sering) meledak di atas angka yang Anda tetapkan. Untuk memungkinkan ledakan dan memastikan bahwa kapasitas yang cukup tersedia untuk menangani permintaan pemuatan data, atur *executor.maxPerSecond* ke 90% dari kapasitas tulis tabel.

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

Dalam tutorial ini, kita mengatur *executor.maxPerSecond* ke 5.

Note

Jika Anda menggunakan DSBulk 1.6.0 atau lebih tinggi, Anda dapat menggunakannya sebagai gantinya. *dsbulk.engine.maxConcurrentQueries*

- b. Konfigurasi parameter tambahan ini untuk load perintah DSBulk.
- *batch-mode*— Parameter ini memberitahu sistem untuk mengelompokkan operasi dengan kunci partisi. Kami merekomendasikan untuk menonaktifkan mode batch, karena dapat menghasilkan skenario dan penyebab hot *keyWriteThrottleEvents*.
 - *driver.advanced.retry-policy-max-retries*— Ini menentukan berapa kali untuk mencoba lagi kueri yang gagal. Jika tidak disetel, defaultnya adalah 10. Anda dapat menyesuaikan nilai ini sesuai kebutuhan.
 - *driver.basic.request.timeout*— Waktu dalam hitungan menit sistem menunggu kueri kembali. Jika tidak disetel, defaultnya adalah “5 menit”. Anda dapat menyesuaikan nilai ini sesuai kebutuhan.

Langkah 5: Jalankan **load** perintah DSBulk untuk mengunggah data dari file CSV ke tabel target

Pada langkah terakhir tutorial ini, Anda mengunggah data ke Amazon Keyspaces.

Untuk menjalankan load perintah DSBulk, selesaikan langkah-langkah berikut.

1. Jalankan kode berikut untuk mengunggah data dari file csv Anda ke tabel Amazon Keyspaces Anda. Pastikan untuk memperbarui jalur ke file konfigurasi aplikasi yang Anda buat sebelumnya.

```
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. Outputnya mencakup lokasi file log yang merinci operasi yang berhasil dan tidak berhasil. File disimpan di direktori berikut.

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

3. Entri file log akan menyertakan metrik, seperti pada contoh berikut. Periksa untuk memastikan bahwa jumlah baris konsisten dengan jumlah baris dalam file csv Anda.

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

Important

Sekarang setelah Anda mentransfer data Anda, sesuaikan pengaturan mode kapasitas tabel target Anda agar sesuai dengan pola lalu lintas reguler aplikasi Anda. Anda dikenakan biaya pada tarif per jam untuk kapasitas yang disediakan sampai Anda mengubahnya. Lihat informasi yang lebih lengkap di [the section called “Konfigurasikan mode read/write kapasitas”](#).

Mengakses Amazon Keyspaces (untuk Apache Cassandra)

Anda dapat mengakses Amazon Keyspaces menggunakan konsol,, secara terprogram dengan menjalankan `cqlsh` klien AWS CloudShell, AWS SDK, atau dengan menggunakan driver Cassandra berlisensi Apache 2.0. Amazon Keyspaces mendukung driver dan klien yang kompatibel dengan Apache Cassandra 3.11.2. Sebelum mengakses Amazon Keyspaces, Anda harus menyelesaikan AWS Identity and Access Management pengaturan dan kemudian memberikan izin akses identitas IAM ke Amazon Keyspaces.

Penyiapan AWS Identity and Access Management

Mendaftar untuk Akun AWS

Untuk memulai AWS, Anda membutuhkan Akun AWS. Untuk informasi tentang membuat Akun AWS, lihat [Memulai dengan Akun AWS](#) di Panduan AWS Account Management Referensi.

Menyiapkan Amazon Keyspaces

[Akses ke sumber daya Amazon Keyspaces dikelola menggunakan IAM.](#) Dengan IAM, Anda dapat melampirkan kebijakan ke pengguna IAM, peran, dan identitas gabungan yang memberikan izin baca dan tulis ke sumber daya tertentu di Amazon Keyspaces.

Untuk memulai pemberian izin ke identitas IAM, Anda dapat menggunakan salah satu kebijakan AWS terkelola untuk Amazon Keyspaces:

- [AmazonKeyspacesFullAccess](#)— kebijakan ini memberikan izin untuk mengakses semua sumber daya di Amazon Keyspaces dengan akses penuh ke semua fitur.
- [AmazonKeyspacesReadOnlyAccess_v2](#) — kebijakan ini memberikan izin hanya-baca ke Amazon Keyspaces.

Untuk penjelasan rinci tentang tindakan yang didefinisikan dalam kebijakan terkelola, lihat [the section called “AWS kebijakan terkelola”](#).

Untuk membatasi cakupan tindakan yang dapat dilakukan oleh identitas IAM atau membatasi sumber daya yang dapat diakses identitas, Anda dapat membuat kebijakan kustom yang menggunakan kebijakan `AmazonKeyspacesFullAccess` terkelola sebagai templat dan menghapus semua izin

yang tidak Anda perlukan. Anda juga dapat membatasi akses ke ruang kunci atau tabel tertentu. Untuk informasi selengkapnya tentang cara membatasi tindakan atau membatasi akses ke sumber daya tertentu di Amazon Keyspaces, lihat. [the section called “Cara Amazon Keyspaces bekerja dengan IAM”](#)

Untuk mengakses Amazon Keyspaces setelah Anda membuat Akun AWS dan membuat kebijakan yang memberikan akses identitas IAM ke Amazon Keyspaces, lanjutkan ke salah satu bagian berikut:

- [Menggunakan konsol](#)
- [Menggunakan AWS CloudShell](#)

Mengakses Amazon Keyspaces menggunakan konsol

Anda dapat mengakses konsol untuk Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home> Untuk informasi selengkapnya tentang Konsol Manajemen AWS akses, lihat [Mengontrol akses pengguna IAM ke Konsol Manajemen AWS](#) dalam Panduan Pengguna IAM.

Anda dapat menggunakan konsol untuk melakukan hal berikut di Amazon Keyspaces:

- Buat, hapus, dan kelola ruang kunci dan tabel.
- Pantau metrik tabel penting pada tab Monitor tabel:
 - Ukuran tabel yang dapat ditagih (Bytes)
 - Metrik kapasitas
- Jalankan kueri menggunakan editor CQL atau melalui [AWS CloudShell pengalaman terintegrasi](#), misalnya menyisipkan, memperbarui, dan menghapus data.
- Connect ke Amazon Keyspaces secara langsung melalui AWS CloudShell dukungan CQLSH terintegrasi.
- Ubah konfigurasi partisi akun.
- Lihat metrik kinerja dan kesalahan untuk akun di dasbor.

Connect ke Amazon Keyspaces menggunakan AWS CloudShell dari konsol

AWS CloudShell menyediakan cara yang efisien untuk terhubung ke Amazon Keyspaces langsung dari konsol. AWS CloudShell Integrasi secara otomatis menginisialisasi `cqlsh-expansion` untuk Anda dan membuat koneksi ke ruang kunci tertentu.

Memulai dengan AWS CloudShell integrasi

1. Akses AWS CloudShell integrasi — Dari konsol Amazon Keyspaces, navigasikan ke ruang kunci yang ingin Anda sambungkan dan pilih tombol Connect to Keyspaces. AWS CloudShell
2. Pre-filled perintah setup — Ketika Anda memilih AWS CloudShell tombol, sistem menyediakan perintah pra-isi yang mencakup inisialisasi yang diperlukan dan parameter koneksi:

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

3. Klik Jalankan
4. Tunggu Amazon Keyspaces membuat koneksi — Berikan waktu untuk proses penyiapan selesai dan koneksi ke ruang kunci yang Anda pilih akan dibuat.
5. Jalankan perintah — Ketika terhubung, Anda dapat menulis dan menjalankan perintah CQL langsung terhadap keyspace yang dipilih.

Perintah CQLSH yang tersedia

AWS CloudShell Integrasi ini menyediakan akses ke fungsionalitas CQLSH standar, termasuk dukungan untuk perintah di luar CQL dasar.

- Jelaskan — Daftar dan lihat deskripsi rinci dari keyspaces dan tabel.
- Salin - Ekspor output kueri ke file untuk diproses nanti.
- Sumber - Jalankan pernyataan CQL dari file.
- Navigasi riwayat perintah - Gulir melalui perintah sebelumnya menggunakan pintasan keyboard.
- Eksekusi kueri - Jalankan kueri langsung dari antarmuka keyboard.
- Penyalinan keluaran - Salin hasil sebagai teks sederhana untuk digunakan dalam aplikasi lain.

Pertimbangan fitur

Saat menggunakan AWS CloudShell integrasi, pertimbangkan trade-off berikut.

Fitur-fitur berikut tidak didukung:

- Saran sintaks CQL dan fungsionalitas pelengkapan otomatis.
- Alat editor seperti show autocomplete, show tooltip, find, find and replace, redo, toggle block comments, toggle code folding, toggle line comment, dan undo.

Fungsionalitas yang dimodifikasi:

- Unduhan CSV - Untuk mengunduh hasil sebagai format CSV, tentukan > operator untuk menyalurkan output perintah:

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

- Tampilan JSON - Anda dapat melihat item dalam format JSON tanpa memerlukan konversi data.
- Terminal-style antarmuka - Pengalaman beroperasi sebagai antarmuka terminal daripada editor kode berfitur lengkap.

AWS CloudShell Integrasi menyederhanakan proses koneksi sambil menyediakan fungsionalitas CQL penting untuk mengelola data Amazon Keyspaces Anda langsung dari file. Konsol Manajemen AWS

Untuk mempelajari cara membuat keyspace dan tabel Amazon Keyspaces dan mengaturnya dengan contoh data aplikasi, lihat. [Memulai dengan Amazon Keyspaces \(untuk Apache Cassandra\)](#)

Menggunakan AWS CloudShell untuk mengakses Amazon Keyspaces

AWS CloudShell adalah shell berbasis browser dan pra-otentikasi yang dapat Anda luncurkan langsung dari file. Konsol Manajemen AWS Anda dapat menjalankan AWS CLI perintah terhadap AWS layanan menggunakan shell pilihan Anda (Bash, PowerShell atau Z shell). AWS CloudShell dilengkapi pra-autentikasi dengan kredensial konsol Anda dan menyertakan alat umum untuk bekerja dengan AWS layanan, termasuk, AWS CLI Python, dan. Node.js

Anda [meluncurkan AWS CloudShell dari Konsol Manajemen AWS](#), dan AWS kredensial yang Anda gunakan untuk masuk ke konsol secara otomatis tersedia di sesi shell baru. Pra-otentikasi AWS CloudShell pengguna ini memungkinkan Anda untuk melewati konfigurasi kredensial saat berinteraksi dengan layanan AWS seperti Amazon Keyspaces menggunakan cqlsh atau AWS CLI versi 2 (pra-instal pada lingkungan komputasi shell).

AWS CloudShell opsi integrasi

Amazon Keyspaces menawarkan dua cara untuk menggunakan: AWS CloudShell

1. AWS CloudShell Pengalaman terintegrasi - Metode koneksi efisien yang tersedia langsung dari konsol Amazon Keyspaces yang secara otomatis menginisialisasi `cqlsh-expansion` dan menghubungkan ke ruang kunci tertentu. Untuk petunjuk lengkap tentang penggunaan pengalaman terintegrasi ini, lihat [the section called “CloudShell integrasi konsol”](#).
2. AWS CloudShell Pengaturan manual - Instal dan konfigurasi `cqlsh` secara manual AWS CloudShell. Langkah-langkah dijelaskan di bagian berikut dan memberikan kontrol lebih besar atas proses penyiapan dan mengaktifkan konfigurasi khusus.

Prasyarat

Sebelum menggunakan Amazon Keyspaces dengan AWS CloudShell, pastikan Anda memiliki:

- AWS Akun dengan akses ke Amazon Keyspaces dan AWS CloudShell
- Izin IAM yang sesuai untuk menggunakan kedua layanan
- Keakraban dasar dengan Cassandra Query Language (CQL)

Memulai dengan AWS CloudShell

Untuk mengakses AWS CloudShell:

1. Masuk ke Konsol Manajemen AWS
2. Luncurkan AWS CloudShell dengan memilih AWS CloudShell ikon di bilah navigasi di bagian atas konsol, atau dengan menavigasi ke <https://console.aws.amazon.com/cloudshell/>
3. Tunggu AWS CloudShell lingkungan untuk diinisialisasi

Memperoleh izin IAM untuk AWS CloudShell

Dengan menggunakan sumber daya manajemen akses yang disediakan oleh AWS Identity and Access Management, administrator dapat memberikan izin kepada pengguna IAM sehingga mereka dapat mengakses AWS CloudShell dan menggunakan fitur lingkungan.

Cara tercepat bagi administrator untuk memberikan akses ke pengguna adalah melalui kebijakan AWS terkelola. [Kebijakan AWS terkelola](#) adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan AWS terkelola berikut ini CloudShell dapat dilampirkan ke identitas IAM:

- **AWSCloudShellFullAccess**: Memberikan izin untuk menggunakan AWS CloudShell dengan akses penuh ke semua fitur.

Jika ingin membatasi cakupan tindakan yang dapat dilakukan oleh pengguna IAM AWS CloudShell, Anda dapat membuat kebijakan kustom yang menggunakan kebijakan **AWSCloudShellFullAccess** terkelola sebagai templat. Untuk informasi selengkapnya tentang membatasi tindakan yang tersedia bagi pengguna CloudShell, lihat [Mengelola AWS CloudShell akses dan penggunaan dengan kebijakan IAM](#) di Panduan AWS CloudShell Pengguna.

Note

Identitas IAM Anda juga memerlukan kebijakan yang memberikan izin untuk melakukan panggilan ke Amazon Keyspaces.

Anda dapat menggunakan kebijakan AWS terkelola untuk memberikan akses identitas IAM ke Amazon Keyspaces, atau mulai dengan kebijakan terkelola sebagai templat dan menghapus izin yang tidak Anda perlukan. Anda juga dapat membatasi akses ke ruang kunci dan tabel tertentu untuk membuat kebijakan khusus. Kebijakan terkelola berikut untuk Amazon Keyspaces dapat dilampirkan ke identitas IAM:

- [AmazonKeyspacesFullAccess](#)— Kebijakan ini memberikan izin untuk menggunakan Amazon Keyspaces dengan akses penuh ke semua fitur.

Untuk penjelasan rinci tentang tindakan yang didefinisikan dalam kebijakan terkelola, lihat [the section called “AWS kebijakan terkelola”](#).

Untuk informasi selengkapnya tentang cara membatasi tindakan atau membatasi akses ke sumber daya tertentu di Amazon Keyspaces, lihat [the section called “Cara Amazon Keyspaces bekerja dengan IAM”](#)

Menginstal klien cqlsh

AWS CloudShell tidak datang dengan cqlsh pra-instal. Anda perlu menginstalnya untuk berinteraksi dengan Amazon Keyspaces. Anda dapat menginstal Keyspaces-specific versi Amazoncqlsh. Untuk petunjuk cqlsh-expansion penginstalan, lihat [the section called “Menggunakan cqlsh-expansion”](#).

Mengkonfigurasi sertifikat TLS

Untuk terhubung ke Amazon Keyspaces menggunakan koneksi TLS, Anda perlu mengunduh sertifikat digital Amazon yang diperlukan. Untuk petunjuk terperinci tentang mengunduh dan mengonfigurasi sertifikat SSL, lihat. [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#)

Menghubungkan ke Amazon Keyspaces

Anda dapat terhubung ke Amazon Keyspaces menggunakan kredensial IAM (disarankan) atau kredensial khusus layanan.

Menghubungkan dengan kredensial IAM (disarankan)

AWS CloudShell sudah diautentikasi dengan kredensial IAM Anda. Untuk terhubung menggunakan kredensial ini, Anda dapat menggunakan plugin otentikasi SiGv4. Untuk petunjuk terperinci tentang penggunaan kredensial IAM dengan `cqlsh`, lihat. [the section called “Menggunakan cqlsh-expansion”](#)

Menghubungkan dengan kredensial khusus layanan

Jika Anda telah membuat kredensial khusus layanan untuk Amazon Keyspaces, Anda dapat menggunakannya untuk terhubung. Untuk informasi tentang membuat dan menggunakan kredensial khusus layanan, lihat. [the section called “Buat kredensial khusus layanan”](#)

Menggunakan CQLSH dengan Amazon Keyspaces

Saat terhubung, Anda dapat menggunakan perintah CQL standar untuk berinteraksi dengan ruang kunci dan tabel Anda. Untuk informasi rinci tentang perintah dan operasi CQL yang didukung, lihat. [the section called “Menggunakan cqlsh-expansion”](#)

Data yang bertahan di AWS CloudShell

AWS CloudShell menyediakan 1 GB penyimpanan persisten di direktori home Anda. Ini berarti skrip, sertifikat, dan file konfigurasi yang Anda buat tetap tersedia di seluruh sesi. Anda dapat membuat file konfigurasi `cqlsh` untuk menyederhanakan koneksi.

Dengan file konfigurasi, Anda dapat menjalankan `cqlsh` tanpa menentukan parameter tambahan. Untuk informasi selengkapnya tentang membuat dan menggunakan file `cqlsh` konfigurasi, lihat [the section called “Menggunakan cqlsh-expansion”](#).

Praktik terbaik

Saat menggunakan Amazon Keyspaces with AWS CloudShell, kami merekomendasikan praktik terbaik berikut:

- Buat skrip untuk tugas berulang dan simpan di direktori home persisten Anda.
- Gunakan variabel lingkungan untuk informasi sensitif daripada hardcoding mereka.
- Ingatlah bahwa AWS CloudShell sesi mungkin habis setelah tidak aktif.
- Berhati-hatilah dengan kuota dan batasan AWS CloudShell layanan.
- Pertimbangkan untuk menggunakan AWS CLI untuk operasi otomatis dengan Amazon Keyspaces.

Pemecahan masalah

Jika Anda mengalami masalah saat menyambungkan ke Amazon Keyspaces dari: AWS CloudShell

- Verifikasi izin IAM Anda mencakup tindakan Amazon Keyspaces yang diperlukan.
- Pastikan Anda menggunakan titik akhir yang benar untuk Wilayah Anda.
- Periksa apakah sertifikat SSL diunduh dan direferensikan dengan benar.
- Verifikasi bahwa kredensial khusus layanan Anda benar dan tidak kedaluwarsa.
- Jika menggunakan file `cqlshrc`, periksa format dan izinnya.

Contoh: Cara berinteraksi dengan Amazon Keyspaces menggunakan AWS CloudShell

Setelah Anda meluncurkan AWS CloudShell dari Konsol Manajemen AWS, Anda dapat segera mulai berinteraksi dengan Amazon Keyspaces menggunakan `cqlsh` atau antarmuka baris perintah. Jika Anda belum menginstal `cqlsh-expansion`, lihat [the section called “Menggunakan cqlsh-expansion”](#) langkah-langkah rinci.

Note

Saat menggunakan `cqlsh-expansion` in AWS CloudShell, Anda tidak perlu mengonfigurasi kredensial sebelum melakukan panggilan, karena Anda sudah diautentikasi di dalam shell.

Connect ke Amazon Keyspaces dan buat keyspace baru. Kemudian baca dari tabel sistem untuk mengonfirmasi bahwa ruang kunci dibuat menggunakan AWS CloudShell

1. Dari Konsol Manajemen AWS, Anda dapat meluncurkan CloudShell dengan memilih opsi berikut yang tersedia di bilah navigasi:
 - Pilih CloudShell ikonnya.
 - Mulai mengetik `cloudshell` di kotak pencarian dan kemudian pilih CloudShell opsi.
2. Anda dapat membuat koneksi ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk mengganti `cassandra.us-east-1.amazonaws.com` dengan titik akhir yang benar untuk Wilayah Anda.

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

Jika koneksi berhasil, Anda akan melihat output yang mirip dengan contoh berikut.

```
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. Buat keyspace baru dengan `namamykeyspace`. Anda dapat menggunakan perintah berikut untuk melakukan itu.

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

4. Untuk mengonfirmasi bahwa ruang kunci telah dibuat, Anda dapat membaca dari tabel sistem menggunakan perintah berikut.

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

Jika panggilan berhasil, baris perintah menampilkan respons dari layanan yang mirip dengan output berikut:

```
keyspace_name | durable_writes | replication
-----+-----
+-----
```

```
mykeyspace      |      True | {'class':  
'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}  
  
(1 rows)
```

Buat kredensi untuk akses terprogram ke Amazon Keyspaces

Untuk memberikan kredensi kepada pengguna dan aplikasi untuk akses terprogram ke sumber daya Amazon Keyspaces, Anda dapat melakukan salah satu hal berikut:

- Buat kredensi khusus layanan yang mirip dengan nama pengguna dan kata sandi tradisional yang digunakan Cassandra untuk otentikasi dan manajemen akses. AWS Kredensi khusus layanan dikaitkan dengan pengguna tertentu AWS Identity and Access Management (IAM) dan hanya dapat digunakan untuk layanan yang mereka buat. Untuk informasi selengkapnya, lihat [Menggunakan IAM dengan Amazon Keyspaces \(untuk Apache Cassandra\)](#) di Panduan Pengguna IAM.

Warning

Pengguna IAM memiliki kredensi jangka panjang, yang menghadirkan risiko keamanan. Untuk membantu mengurangi risiko ini, kami menyarankan agar Anda memberikan pengguna ini hanya izin yang mereka perlukan untuk melakukan tugas dan menghapus pengguna ini ketika mereka tidak lagi diperlukan.

- Untuk keamanan yang ditingkatkan, kami sarankan untuk membuat identitas IAM yang digunakan di semua AWS layanan dan menggunakan kredensial sementara. Plugin otentikasi Amazon Keyspaces SigV4 untuk driver klien Cassandra memungkinkan Anda untuk mengautentikasi panggilan ke Amazon Keyspaces menggunakan kunci akses IAM alih-alih nama pengguna dan kata sandi. [Untuk mempelajari lebih lanjut tentang cara plugin Amazon Keyspaces SigV4 memungkinkan pengguna IAM, peran, dan identitas gabungan untuk mengautentikasi dalam permintaan API Amazon Keyspaces, lihat Proses Sigv4 Versi Tanda Tangan 4 \(SigV4\).AWS](#)

Anda dapat mengunduh plugin SiGv4 dari lokasi berikut.

- Jawa: <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>.
- Pergi: <https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin>.

Untuk contoh kode yang menunjukkan cara membuat koneksi menggunakan plugin otentikasi SiGv4, lihat. [the section called “Menggunakan driver klien Cassandra”](#)

Topik

- [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)
- [Membuat dan mengonfigurasi AWS kredensial untuk Amazon Keyspaces](#)

Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces

Service-specific kredensialnya mirip dengan nama pengguna dan kata sandi tradisional yang digunakan Cassandra untuk otentikasi dan manajemen akses. Service-specific kredensial memungkinkan pengguna IAM untuk mengakses layanan tertentu. AWS Kredensi jangka panjang ini tidak dapat digunakan untuk mengakses layanan lain AWS. Mereka terkait dengan pengguna IAM tertentu dan tidak dapat digunakan oleh pengguna IAM lainnya.

Important

Service-specific kredensial adalah kredensial jangka panjang yang terkait dengan pengguna IAM tertentu dan hanya dapat digunakan untuk layanan yang mereka buat. Untuk memberikan peran IAM atau izin identitas gabungan untuk mengakses semua AWS sumber daya Anda menggunakan kredensi sementara, Anda harus menggunakan [AWS otentikasi dengan plugin otentikasi SiGv4 untuk Amazon Keyspaces](#).

Gunakan salah satu prosedur berikut untuk menghasilkan kredensial khusus layanan.

Console

Buat kredensial khusus layanan menggunakan konsol

1. Masuk ke Konsol Manajemen AWS dan buka AWS Identity and Access Management konsol di <https://console.aws.amazon.com/iam/home>.
2. Di panel navigasi, pilih Pengguna, lalu pilih pengguna yang Anda buat sebelumnya yang memiliki izin Amazon Keyspaces (kebijakan terlampir).

3. Pilih Kredensial Keamanan. Di bawah Kredensial untuk Amazon Keyspaces, pilih Hasilkan kredensial untuk menghasilkan kredensial khusus layanan.

Kredensial layanan khusus Anda sekarang tersedia. Ini adalah satu-satunya saat Anda dapat mengunduh atau melihat kata sandi. Anda tidak dapat memulihkannya nanti. Namun, Anda dapat mengatur ulang kata sandi Anda kapan saja. Simpan pengguna dan kata sandi di lokasi aman, karena Anda akan membutuhkannya nanti.

CLI

Buat kredensial khusus layanan menggunakan AWS CLI

Sebelum membuat kredensial khusus layanan, Anda perlu mengunduh, menginstal, dan mengonfigurasi (): AWS Command Line Interface AWS CLI

1. Unduh AWS CLI di <http://aws.amazon.com/cli>.

Note

AWS CLI Berjalan di Windows, macOS, atau Linux.

2. Ikuti petunjuk untuk [Menginstal AWS CLI](#) dan [Mengkonfigurasi AWS CLI di Panduan Pengguna](#). AWS Command Line Interface
3. Dengan menggunakan AWS CLI, jalankan perintah berikut untuk menghasilkan kredensial khusus layanan bagi pengguna `alice`, sehingga dia dapat mengakses Amazon Keyspaces.

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

Output-nya akan terlihat seperti berikut.

```
{  
  "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"  
  }  
}
```

Dalam output, perhatikan nilai untuk `ServiceUserName` dan `ServicePassword`. Simpan nilai-nilai ini di lokasi yang aman, karena Anda akan membutuhkannya nanti.

 Important

Ini adalah satu-satunya waktu yang `ServicePassword` akan tersedia untuk Anda.

Membuat dan mengonfigurasi AWS kredensial untuk Amazon Keyspaces

Untuk mengakses Amazon Keyspaces secara terprogram dengan, AWS SDK AWS CLI, atau dengan driver klien Cassandra dan plugin SiGv4, Anda memerlukan pengguna IAM dengan kunci akses. Saat Anda menggunakan Amazon Keyspaces secara terprogram, Anda memberikan kunci AWS akses sehingga AWS dapat memverifikasi identitas Anda dalam panggilan terprogram. Kunci akses Anda terdiri dari ID kunci akses (misalnya, AKIAIOSFODNN7EXAMPLE) dan kunci akses rahasia (misalnya, /b CYEXAMPLEKEY). wJalrXUtnFEMI/K7MDENG PxRfi Topik ini memandu Anda melalui langkah-langkah yang diperlukan dalam proses ini.

Praktik terbaik keamanan menyarankan Anda membuat pengguna IAM dengan izin terbatas dan sebagai gantinya mengaitkan peran IAM dengan izin yang diperlukan untuk melakukan tugas tertentu. Pengguna IAM kemudian dapat sementara mengambil peran IAM untuk melakukan tugas yang diperlukan. Misalnya, pengguna IAM di akun Anda yang menggunakan konsol Amazon Keyspaces dapat beralih ke peran untuk sementara menggunakan izin peran di konsol. Pengguna menyerahkan izin mereka dan mengambil izin yang ditetapkan untuk peran tersebut. Saat pengguna keluar dari peran, izin asli mereka dipulihkan. Kredensi yang digunakan pengguna untuk mengambil peran bersifat sementara. Sebaliknya, pengguna IAM memiliki kredensi jangka panjang, yang menghadirkan risiko keamanan jika alih-alih mengasumsikan peran, mereka memiliki izin yang langsung diberikan kepada mereka. Untuk membantu mengurangi risiko ini, kami menyarankan agar Anda memberikan pengguna ini hanya izin yang mereka perlukan untuk melakukan tugas dan menghapus pengguna ini ketika mereka tidak lagi diperlukan. Untuk informasi selengkapnya tentang peran, lihat [Skenario umum untuk peran: Pengguna, aplikasi, dan layanan](#) di Panduan Pengguna IAM.

Topik

- [Kredensial yang diperlukan oleh, AWS SDK AWS CLI, atau plugin Amazon Keyspaces SigV4 untuk driver klien Cassandra](#)
- [Buat kredensi sementara untuk terhubung ke Amazon Keyspaces menggunakan peran IAM dan plugin SiGv4](#)
- [Buat pengguna IAM untuk akses terprogram ke Amazon Keyspaces di akun Anda AWS](#)
- [Buat kunci akses baru untuk pengguna IAM](#)
- [Simpan kunci akses untuk akses terprogram](#)

Kredensial yang diperlukan oleh, AWS SDK AWS CLI, atau plugin Amazon Keyspaces SigV4 untuk driver klien Cassandra

Kredensi berikut diperlukan untuk mengautentikasi pengguna atau peran IAM:

`AWS_ACCESS_KEY_ID`

Menentukan kunci AWS akses yang terkait dengan pengguna IAM atau peran.

Kunci akses `aws_access_key_id` diperlukan untuk terhubung ke Amazon Keyspaces secara terprogram.

`AWS_SECRET_ACCESS_KEY`

Menentukan kunci rahasia yang terkait dengan kunci akses. Ini pada dasarnya adalah “kata sandi” untuk kunci akses.

`aws_secret_access_key` Diperlukan untuk terhubung ke Amazon Keyspaces secara terprogram.

`AWS_SESSION_TOKEN`- Opsional

Menentukan nilai token sesi yang diperlukan jika Anda menggunakan kredensial keamanan sementara yang Anda ambil langsung dari operasi. AWS Security Token Service Untuk informasi selengkapnya, lihat [the section called “Buat kredensi sementara untuk terhubung ke Amazon Keyspaces”](#).

Jika Anda terhubung dengan pengguna IAM, tidak `aws_session_token` diperlukan.

Buat kredensi sementara untuk terhubung ke Amazon Keyspaces menggunakan peran IAM dan plugin SiGv4

Cara yang disarankan untuk mengakses Amazon Keyspaces secara terprogram adalah dengan menggunakan [kredensi sementara untuk mengautentikasi](#) dengan plugin SiGv4. Dalam banyak skenario, Anda tidak memerlukan access key jangka panjang yang tidak pernah kedaluwarsa (seperti yang Anda lakukan dengan pengguna IAM). Sebagai gantinya, Anda dapat membuat peran IAM dan menghasilkan kredensial keamanan sementara. Kredensial keamanan sementara terdiri dari access key ID dan secret access key, tetapi mereka juga menyertakan token keamanan yang menunjukkan kapan kredensial kedaluwarsa. Untuk mempelajari lebih lanjut tentang cara menggunakan peran IAM alih-alih kunci akses jangka panjang, lihat [Beralih ke peran IAM \(AWS API\)](#).

Untuk memulai dengan kredensi sementara, Anda harus terlebih dahulu membuat peran IAM.

Buat peran IAM yang memberikan akses hanya-baca ke Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Peran, lalu Buat peran.
3. Pada halaman Buat peran, di bawah Pilih jenis entitas tepercaya, pilih AWS layanan. Di bawah Pilih kasus penggunaan, pilih Amazon EC2, lalu pilih Berikutnya.
4. Pada halaman Tambahkan izin, di bawah Kebijakan izin, pilih Amazon Keyspaces Baca Hanya Akses dari daftar kebijakan, lalu pilih Berikutnya.
5. Pada halaman Nama, tinjau, dan buat, masukkan nama untuk peran tersebut, dan tinjau bagian Pilih entitas tepercaya dan Tambahkan izin. Anda juga dapat menambahkan tag opsional untuk peran di halaman ini. Setelah selesai, pilih Buat peran. Ingat nama ini karena Anda akan membutuhkannya saat meluncurkan instans Amazon EC2 Anda.

Untuk menggunakan kredensi keamanan sementara dalam kode, Anda secara terprogram memanggil AWS Security Token Service API seperti `AssumeRole` dan mengekstrak kredensial dan token sesi yang dihasilkan dari peran IAM yang Anda buat pada langkah sebelumnya. Anda kemudian menggunakan nilai-nilai tersebut sebagai kredensial untuk panggilan berikutnya ke AWS. Contoh berikut menunjukkan pseudocode untuk cara menggunakan kredensial keamanan sementara:

```
assumeRoleResult = AssumeRole(role-arn);  
tempCredentials = new SessionAWSCredentials(  
    assumeRoleResult.getAccessKeyId(),  
    assumeRoleResult.getSecretAccessKey(),  
    assumeRoleResult.getSessionToken(),  
    assumeRoleResult.getExpiration()
```

```
assumeRoleResult.AccessKeyId,  
assumeRoleResult.SecretAccessKey,  
assumeRoleResult.SessionToken);  
cassandraRequest = CreateAmazoncassandraClient(tempCredentials);
```

Untuk contoh yang mengimplementasikan kredensi sementara menggunakan driver Python untuk mengakses Amazon Keyspaces, lihat. [???](#)

Untuk detail tentang cara memanggil AssumeRoleGetFederationToken, dan operasi API lainnya, lihat [Referensi AWS Security Token Service API](#). Untuk informasi tentang mendapatkan kredensial keamanan sementara dan token sesi dari hasilnya, lihat dokumentasi untuk SDK yang sedang Anda kerjakan. Anda dapat menemukan dokumentasi untuk semua AWS SDK di [halaman AWS dokumentasi](#) utama, di bagian SDK dan Toolkit.

Buat pengguna IAM untuk akses terprogram ke Amazon Keyspaces di akun Anda AWS

Untuk mendapatkan kredensial akses terprogram ke Amazon Keyspaces dengan plugin, AWS SDK AWS CLI, atau SiGv4, Anda harus terlebih dahulu membuat pengguna atau peran IAM. Proses membuat pengguna IAM dan mengonfigurasi pengguna IAM agar memiliki akses terprogram ke Amazon Keyspaces ditampilkan dalam langkah-langkah berikut:

1. Buat pengguna di Konsol Manajemen AWS AWS CLI, Alat untuk Windows PowerShell, atau menggunakan operasi AWS API. Jika Anda membuat pengguna di Konsol Manajemen AWS, maka kredensialnya dibuat secara otomatis.
2. Jika Anda membuat pengguna secara terprogram, maka Anda harus membuat kunci akses (ID kunci akses dan kunci akses rahasia) untuk pengguna tersebut dalam langkah tambahan.
3. Berikan izin pengguna untuk mengakses Amazon Keyspaces.

Untuk informasi tentang izin yang Anda perlukan untuk membuat pengguna IAM, lihat [izin yang diperlukan untuk mengakses sumber daya IAM](#).

Console

Buat pengguna IAM dengan akses terprogram (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.

2. Di panel navigasi, pilih Pengguna, lalu pilih Tambahkan pengguna.
3. Masukkan nama pengguna untuk pengguna baru. Ini adalah nama masuk untuk AWS.

 Note

Nama pengguna dapat berupa kombinasi hingga 64 huruf, digit, dan karakter ini: plus (+), sama dengan (=), koma (,), titik (.), pada a keong (@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat dua pengguna yang diberi nama TESTUSER dan testuser.

4. Pilih Kunci akses - Akses terprogram untuk membuat kunci akses untuk pengguna baru. Anda dapat melihat atau mengunduh tombol akses ketika Anda sampai ke halaman Final.

Pilih Berikutnya: Izin.

5. Pada halaman Setel izin, pilih Lampirkan kebijakan yang ada secara langsung untuk menetapkan izin ke pengguna baru.

Opsi ini menampilkan daftar kebijakan AWS terkelola dan terkelola pelanggan yang tersedia di akun Anda. Anda dapat masuk keyspaces ke kolom pencarian untuk hanya menampilkan kebijakan yang terkait dengan Amazon Keyspaces.

Untuk Amazon Keyspaces, kebijakan terkelola yang tersedia adalah `AmazonKeyspacesFullAccess` dan `AmazonKeyspacesReadOnlyAccess`. Untuk informasi selengkapnya tentang setiap kebijakan, lihat [the section called “AWS kebijakan terkelola”](#).

Untuk tujuan pengujian dan untuk mengikuti tutorial koneksi, pilih `AmazonKeyspacesReadOnlyAccess` kebijakan untuk pengguna IAM baru. Catatan: Sebagai praktik terbaik, kami menyarankan Anda mengikuti prinsip hak istimewa paling sedikit dan membuat kebijakan khusus yang membatasi akses ke sumber daya tertentu dan hanya mengizinkan tindakan yang diperlukan. Untuk informasi selengkapnya tentang kebijakan IAM dan untuk melihat contoh kebijakan untuk Amazon Keyspaces, lihat [the section called “Kebijakan berbasis identitas Amazon Keyspaces”](#). Setelah Anda membuat kebijakan izin khusus, lampirkan kebijakan Anda ke peran, lalu biarkan pengguna mengambil peran yang sesuai untuk sementara.

Pilih Berikutnya: Tanda.

6. Pada halaman Tambahkan tag (opsional) Anda dapat menambahkan tag untuk pengguna, atau memilih Berikutnya: Tinjau.
7. Pada halaman Review Anda dapat melihat semua pilihan yang Anda buat sampai saat ini. Saat Anda siap untuk melanjutkan, pilih Buat pengguna.
8. Untuk melihat access key pengguna (access key ID dan secret access key), pilih Tampilkan di samping setiap kata sandi dan kunci akses rahasia. Untuk menyimpan kunci akses tersebut, pilih Download .csv (Unduh .csv) lalu simpan file ke lokasi yang aman.

 Important

Ini adalah satu-satunya kesempatan Anda untuk melihat atau mengunduh kunci akses rahasia, dan Anda memerlukan informasi ini sebelum mereka dapat menggunakan plugin SiGv4. Simpan access key ID baru pengguna dan secret access key di tempat yang aman dan terlindungi. Anda tidak akan memiliki akses ke kunci rahasia kembali setelah langkah ini.

CLI

Buat pengguna IAM dengan akses terprogram (AWS CLI)

1. Buat pengguna dengan AWS CLI kode berikut.
 - [aws iam create-user](#)
2. Berikan akses terprogram kepada pengguna. Ini membutuhkan kunci akses, yang dapat dihasilkan dengan cara berikut.
 - AWS CLI: [aws iam create-access-key](#)
 - Tools for Windows PowerShell: [New-IAMAccessKey](#)
 - API IAM: [CreateAccessKey](#)

 Important

Ini adalah satu-satunya kesempatan Anda untuk melihat atau mengunduh kunci akses rahasia, dan Anda memerlukan informasi ini sebelum mereka dapat menggunakan plugin SiGv4. Simpan access key ID baru pengguna dan secret

access key di tempat yang aman dan terlindungi. Anda tidak akan memiliki akses ke kunci rahasia kembali setelah langkah ini.

3. Lampirkan `AmazonKeyspacesReadOnlyAccess` kebijakan ke pengguna yang menentukan izin pengguna. Catatan: Sebagai praktik terbaik, sebaiknya Anda mengelola izin pengguna dengan menambahkan pengguna ke grup dan melampirkan kebijakan ke grup, bukan melampirkan langsung ke pengguna.

- AWS CLI: [aws iam attach-user-policy](#)

Buat kunci akses baru untuk pengguna IAM

Jika Anda sudah memiliki pengguna IAM, Anda dapat membuat kunci akses baru kapan saja. Untuk informasi selengkapnya tentang manajemen kunci, misalnya cara memperbarui kunci akses, lihat [Mengelola kunci akses untuk pengguna IAM](#).

Untuk membuat kunci akses untuk pengguna IAM (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna yang kunci aksesnya ingin Anda buat.
4. Pada halaman Ringkasan pengguna, pilih tab Security credentials.
5. Di bagian Kunci akses di bawah Praktik & alternatif terbaik kunci akses, pilih kasus penggunaan Lainnya. Klik Berikutnya, masukkan informasi opsional sesuai kebutuhan, dan pilih Buat kunci akses.

Untuk melihat pasangan access key baru, pilih Show (Tampilkan). Kredensial Anda akan terlihat seperti ini:

- Access key ID: AKIAIOSFODNN7EXAMPLE
- Kunci akses rahasia: wJalrXUtnFEMI/K7MDENG /b PxRfi CYEXAMPLEKEY

 Note

Anda tidak akan memiliki akses ke secret access key lagi setelah menutup kotak dialog ini.

Pertimbangkan praktik terbaik berikut untuk key pair yang telah Anda buat.

- Jangan pernah menyimpan kunci akses Anda dalam teks biasa, dalam repositori kode, atau dalam kode.
 - Nonaktifkan atau hapus kunci akses saat tidak lagi diperlukan.
 - Aktifkan izin hak istimewa paling sedikit.
 - Putar tombol akses secara teratur.
6. Untuk mengunduh pasangan kunci tersebut, pilih Unduh file .csv. Simpan kunci di lokasi yang aman.
 7. Setelah mengunduh file .csv, pilih Tutup.

Saat Anda membuat access key, key pair akan aktif secara default, dan Anda dapat langsung menggunakan pasangan tersebut.

Simpan kunci akses untuk akses terprogram

Sebagai praktik terbaik, kami menyarankan Anda untuk tidak menyematkan kunci akses langsung ke kode. AWS SDK dan AWS Command Line Tools memungkinkan Anda untuk menempatkan kunci akses di lokasi yang diketahui sehingga Anda tidak harus menyimpannya dalam kode. Letakkan access key di salah satu lokasi berikut:

- Variabel lingkungan — Pada sistem multitenant, pilih variabel lingkungan pengguna, bukan variabel lingkungan sistem.
- File kredensial CLI — **config** File `credentials` dan diperbarui saat Anda menjalankan perintah. `aws configure credentials` File ini terletak `~/.aws/credentials` di Linux, macOS, atau Unix, atau di `C:\Users\USERNAME\.aws\credentials` Windows. File ini dapat berisi detail kredensi untuk default profil dan profil bernama apa pun.
- File konfigurasi CLI — **config** File `credentials` dan diperbarui saat Anda menjalankan perintah. `aws configure config` File ini terletak `~/.aws/config` di Linux, macOS, atau Unix,

atau di `C:\Users\USERNAME\.aws\config` Windows. File ini berisi pengaturan konfigurasi untuk profil default dan profil bernama apa pun.

Menyimpan kunci akses sebagai variabel lingkungan merupakan prasyarat untuk [the section called “Plugin otentikasi untuk Java 4.x”](#). Perhatikan bahwa ini termasuk default Wilayah AWS. Klien mencari kredensi menggunakan rantai penyedia kredensial default, dan kunci akses yang disimpan sebagai variabel lingkungan lebih diutamakan daripada semua lokasi lain, misalnya file konfigurasi. Untuk informasi selengkapnya, lihat [Pengaturan konfigurasi dan prioritas](#).

Contoh berikut menunjukkan bagaimana Anda dapat mengkonfigurasi variabel lingkungan untuk pengguna default.

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFc1YEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of security token>
$ export AWS_DEFAULT_REGION=us-east-1
```

Menyetel variabel lingkungan mengubah nilai yang digunakan hingga akhir sesi shell Anda, atau sampai Anda menyetel variabel ke nilai yang berbeda. Anda dapat membuat variabel persisten di seluruh sesi masa depan dengan menyetelnya di skrip startup shell Anda.

Windows Command Prompt

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxrFc1YEXAMPLEKEY
C:\> setx AWS_SESSION_TOKEN AQoDYXdzEJr...<remainder of security token>
C:\> setx AWS_DEFAULT_REGION us-east-1
```

Menggunakan [set](#) untuk mengatur variabel lingkungan mengubah nilai yang digunakan sampai akhir sesi prompt perintah saat ini, atau sampai Anda mengatur variabel ke nilai yang berbeda. Menggunakan [setx](#) untuk mengatur variabel lingkungan mengubah nilai yang digunakan dalam sesi prompt perintah saat ini dan semua sesi prompt perintah yang Anda buat setelah menjalankan perintah. Itu tidak mempengaruhi shell perintah lain yang sudah berjalan pada saat Anda menjalankan perintah.

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
```

```
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJa1rXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY"
PS C:\> $Env:AWS_SESSION_TOKEN="AQoDYXdzEJr...<remainder of security token>"
PS C:\> $Env:AWS_DEFAULT_REGION="us-east-1"
```

Jika Anda menetapkan variabel lingkungan pada PowerShell prompt seperti yang ditunjukkan pada contoh sebelumnya, itu menyimpan nilai hanya untuk durasi sesi saat ini. Untuk membuat pengaturan variabel lingkungan persisten di semua sesi PowerShell Command Prompt, simpan dengan menggunakan aplikasi Sistem di Control Panel. Atau, Anda dapat mengatur variabel untuk semua PowerShell sesi future dengan menambahkannya ke PowerShell profil Anda. Lihat [PowerShell dokumentasi](#) untuk informasi selengkapnya tentang menyimpan variabel lingkungan atau mempertahankannya di seluruh sesi.

Titik akhir layanan untuk Amazon Keyspaces

Topik

- [Port dan protokol](#)
- [Titik akhir global](#)
- [AWS GovCloud \(US\) Region Titik akhir FIPS](#)
- [Titik akhir Wilayah Tiongkok](#)
- [Titik akhir aliran](#)
- [Menghubungkan ke titik akhir dual-stack](#)

Port dan protokol

Anda dapat mengakses Amazon Keyspaces secara terprogram dengan menjalankan `cqlsh` klien, dengan driver Cassandra berlisensi Apache 2.0, atau dengan menggunakan dan SDK. AWS CLI AWS

Tabel berikut menunjukkan port dan protokol untuk mekanisme akses yang berbeda.

Akses Terprogram	Port	Protokol
CQLSH	9142	TLS
Pengemudi Cassandra	9142	TLS

Akses Terprogram	Port	Protokol
AWS CLI	443	HTTPS
AWS SDK	443	HTTPS

Untuk koneksi TLS, Amazon Keyspaces menggunakan sertifikat yang dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1-4) untuk mengautentikasi terhadap server. Untuk informasi lebih lanjut, lihat [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#) atau bagian [Sebelum Anda mulai](#) dari pengemudi Anda di bagian [the section called “Menggunakan driver klien Cassandra”](#) ini.

Titik akhir global

Amazon Keyspaces mendukung titik akhir publik IPv4 dan IPv6. Anda dapat memilih antara titik akhir IPv4 dan titik akhir dual-stack. Titik akhir menggunakan konvensi penamaan berikut, di mana Anda dapat mengganti *us-east-1* dengan yang lain yang tersedia Wilayah AWS dari tabel.

- Titik akhir IPv4 — `cassandra.us-east-1.amazonaws.com`
- Dual-stack titik akhir — `cassandra.us-east-1.api.aws`

Untuk informasi selengkapnya tentang titik akhir dual-stack dan cara mengonfigurasi koneksi, lihat [the section called “Menghubungkan ke titik akhir dual-stack”](#)

Amazon Keyspaces tersedia di Wilayah berikut.

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
AS Timur (Ohio)	us-east-2	<code>cassandra.us-east-2.amazonaws.com</code>	HTTPS dan TLS	
		<code>cassandra.us-east-2.api.aws</code>	HTTPS dan TLS	
AS Timur (Virginia Utara)	us-east-1	<code>cassandra.us-east-1.amazonaws.com</code> <code>cassandra-fips.us-east-1.api.aws</code>	HTTPS dan TLS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
		cassandra-fips.us-east-1.amazonaws.com cassandra.us-east-1.api.aws	HTTPS dan TLS HTTPS dan TLS HTTPS dan TLS	
US West (N. California)	us-west-1	cassandra.us-west-1.amazonaws.com cassandra.us-west-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
AS Barat (Oregon)	us-west-2	cassandra.us-west-2.amazonaws.com cassandra-fips.us-west-2.api.aws cassandra-fips.us-west-2.amazonaws.com cassandra.us-west-2.api.aws	HTTPS dan TLS HTTPS dan TLS HTTPS dan TLS HTTPS dan TLS	
Afrika (Cape Town)	af-south-1	cassandra.af-south-1.amazonaws.com cassandra.af-south-1.api.aws	HTTPS dan TLS HTTPS dan TLS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Asia Pasifik (Hong Kong)	ap-east-1	cassandra.ap-east-1.amazonaws.com cassandra.ap-east-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Asia Pasifik (Malaysia)	ap-southeast-5	cassandra.ap-southeast-5.amazonaws.com cassandra.ap-southeast-5.api.aws	HTTPS dan TLS HTTPS dan TLS	
Asia Pasifik (Mumbai)	ap-south-1	cassandra.ap-south-1.amazonaws.com cassandra.ap-south-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Asia Pasifik (Seoul)	ap-northeast-2	cassandra.ap-northeast-2.amazonaws.com cassandra.ap-northeast-2.api.aws	HTTPS dan TLS HTTPS dan TLS	
Asia Pasifik (Singapura)	ap-southeast-1	cassandra.ap-southeast-1.amazonaws.com cassandra.ap-southeast-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Asia Pasifik (Sydney)	ap-southeast-2	cassandra.ap-southeast-2.amazonaws.com cassandra.ap-southeast-2.api.aws	HTTPS dan TLS HTTPS dan TLS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Asia Pasifik (Thailand)	ap-tenggara-7	cassandra.ap-southeast-7.amazonaws.com cassandra.ap-southeast-7.api.aws	HTTPS dan TLS HTTPS dan TLS	
Asia Pacific (Tokyo)	ap-northeast-1	cassandra.ap-northeast-1.amazonaws.com cassandra.ap-northeast-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Kanada (Pusat)	ca-central-1	cassandra.ca-central-1.amazonaws.com cassandra.ca-central-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Eropa (Frankfurt)	eu-central-1	cassandra.eu-central-1.amazonaws.com cassandra.eu-central-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Eropa (Irlandia)	eu-west-1	cassandra.eu-west-1.amazonaws.com cassandra.eu-west-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Eropa (London)	eu-west-2	cassandra.eu-west-2.amazonaws.com cassandra.eu-west-2.api.aws	HTTPS dan TLS HTTPS dan TLS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Eropa (Paris)	eu-west-3	cassandra.eu-west-3.amazonaws.com cassandra.eu-west-3.api.aws	HTTPS dan TLS HTTPS dan TLS	
Eropa (Stockholm)	eu-north-1	cassandra.eu-north-1.amazonaws.com cassandra.eu-north-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Timur Tengah (Bahrain)	me-south-1	cassandra.me-south-1.amazonaws.com cassandra.me-south-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Timur Tengah (UAE)	me-central-1	cassandra.me-central-1.amazonaws.com cassandra.me-central-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
Amerika Selatan (Sao Paulo)	sa-east-1	cassandra.sa-east-1.amazonaws.com cassandra.sa-east-1.api.aws	HTTPS dan TLS HTTPS dan TLS	
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com cassandra.us-gov-east-1.api.aws	HTTPS dan TLS HTTPS dan TLS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com cassandra.us-gov-west-1.api.aws	HTTPS dan TLS HTTPS dan TLS	

AWS GovCloud (US) Region Titik akhir FIPS

Titik akhir FIPS yang tersedia di AWS GovCloud (US) Region Amazon Keyspaces mendukung titik akhir IPv4 dan IPv6 FIPS. Anda dapat memilih antara titik akhir IPv4 dan titik akhir dual-stack. Untuk informasi selengkapnya, lihat [Amazon Keyspaces di AWS GovCloud \(US\) Panduan Pengguna](#).

Nama wilayah	Region	Titik akhir FIPS IPv4	Tumpukan ganda titik akhir FIPS	Protokol
AWS GovCloud (US-East)	us-gov-east-1	cassandra.us-gov-east-1.amazonaws.com	cassandra.us-gov-east-1.api.aws	HTTPS dan TLS
AWS GovCloud (US-West)	us-gov-west-1	cassandra.us-gov-west-1.amazonaws.com	cassandra.us-gov-west-1.api.aws	HTTPS dan TLS

Titik akhir Wilayah Tiongkok

Amazon Keyspaces mendukung titik akhir IPv4 di Wilayah Tiongkok. AWS

Untuk mengakses titik akhir ini, Anda harus mendaftar untuk kumpulan kredensial akun terpisah yang unik untuk Wilayah Tiongkok. Untuk informasi selengkapnya, lihat [Pendaftaran, Akun, dan Kredensial Tiongkok](#).

Nama wilayah	Wilayah	Titik Akhir	Protokol
Tiongkok (Beijing)	cn-north-1	cassandra.cn-north-1.amazonaws.com.cn	HTTPS dan TLS
Tiongkok (Ningxia)	cn-northwest-1	cassandra.cn-northwest-1.amazonaws.com.cn	HTTPS dan TLS

Titik akhir aliran

Amazon Keyspaces CDC streams tersedia di berikut ini Wilayah AWS. Tabel ini menunjukkan endpoint layanan dual-stack yang tersedia untuk setiap Wilayah. Untuk informasi lebih lanjut tentang Amazon Keyspaces CDC streams, lihat [the section called “Akses titik akhir aliran CDC”](#).

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
AS Timur (Ohio)	us-east-2	cassandra-streams.us-east-2.api.aws	HTTPS	
AS Timur (Virginia Utara)	us-east-1	cassandra-streams.us-east-1.api.aws	HTTPS	
US West (N. California)	us-west-1	cassandra-streams.us-west-1.api.aws	HTTPS	
AS Barat (Oregon)	us-west-2	cassandra-streams.us-west-2.api.aws	HTTPS	
Afrika (Cape Town)	af-south-1	cassandra-streams.af-south-1.api.aws	HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Asia Pasifik (Hong Kong)	ap-east-1	cassandra-streams.ap-east-1.api.aws	HTTPS	
Asia Pasifik (Malaysia)	ap-southeast-5	cassandra-streams.ap-southeast-5.api.aws	HTTPS	
Asia Pasifik (Mumbai)	ap-south-1	cassandra-streams.ap-south-1.api.aws	HTTPS	
Asia Pasifik (Seoul)	ap-northeast-2	cassandra-streams.ap-northeast-2.api.aws	HTTPS	
Asia Pasifik (Singapura)	ap-southeast-1	cassandra-streams.ap-southeast-1.api.aws	HTTPS	
Asia Pasifik (Sydney)	ap-southeast-2	cassandra-streams.ap-southeast-2.api.aws	HTTPS	
Asia Pasifik (Thailand)	ap-southeast-7	cassandra-streams.ap-southeast-7.api.aws	HTTPS	
Asia Pacific (Tokyo)	ap-northeast-1	cassandra-streams.ap-northeast-1.api.aws	HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Kanada (Pusat)	ca-central-1	cassandra-streams.ca-central-1.api.aws	HTTPS	
Eropa (Frankfurt)	eu-central-1	cassandra-streams.eu-central-1.api.aws	HTTPS	
Eropa (Irlandia)	eu-west-1	cassandra-streams.eu-west-1.api.aws	HTTPS	
Eropa (London)	eu-west-2	cassandra-streams.eu-west-2.api.aws	HTTPS	
Eropa (Paris)	eu-west-3	cassandra-streams.eu-west-3.api.aws	HTTPS	
Eropa (Stockholm)	eu-north-1	cassandra-streams.eu-north-1.api.aws	HTTPS	
Timur Tengah (Bahrain)	me-south-1	cassandra-streams.me-south-1.api.aws	HTTPS	
Timur Tengah (UAE)	me-central-1	cassandra-streams.me-central-1.api.aws	HTTPS	
Amerika Selatan (Sao Paulo)	sa-east-1	cassandra-streams.sa-east-1.api.aws	HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
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	

Menghubungkan ke titik akhir dual-stack

Titik akhir global Amazon Keyspaces adalah titik akhir tumpukan ganda yang menerima permintaan IPv4 dan IPv6.

Saat menyambungkan ke Amazon Keyspaces menggunakan IPv6, layanan secara otomatis menyesuaikan respons tabel sistem agar sesuai dengan protokol koneksi Anda. Ini memastikan bahwa aplikasi Anda menerima informasi alamat jaringan yang konsisten sesuai dengan jenis koneksi mereka. Ini memberikan informasi topologi jaringan yang akurat kepada klien sambil mempertahankan kompatibilitas mundur untuk aplikasi CQL yang ada.

Amazon Keyspaces mendeteksi protokol jaringan (IPv4 atau IPv6) yang digunakan oleh koneksi klien Anda secara otomatis dan menyesuaikan respons tabel sistem yang sesuai. Deteksi ini terjadi secara transparan selama jabat tangan koneksi awal, tidak memerlukan konfigurasi tambahan dari aplikasi klien.

Amazon Keyspaces mengembalikan alamat IP berdasarkan protokol koneksi Anda. Misalnya, permintaan dari jaringan IPv4 mengembalikan respons berikut.

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

Koneksi dari jaringan IPv6 ke titik akhir dual-stack, misalnya `cassandra.us-east-1.api.aws`, mengembalikan respons berikut.

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

Untuk informasi selengkapnya tentang dukungan IPv6 di Amazon Keyspaces, lihat [the section called “Dukungan IPv6 di Amazon Keyspaces”](#)

Dukungan IPv6 di Amazon Keyspaces

Dukungan IPv6 di Amazon Keyspaces memungkinkan aplikasi untuk membuat koneksi menggunakan Internet Protocol versi 6, protokol internet generasi berikutnya yang menyediakan ruang alamat yang jauh lebih luas dibandingkan dengan IPv4. Implementasinya menggunakan endpoint dual-stack yang mendukung IPv4 dan IPv6 secara bersamaan, memastikan kompatibilitas mundur sekaligus mengaktifkan konektivitas yang siap di masa depan. Untuk daftar titik akhir, lihat [the section called “Titik akhir global”](#).

Amazon Keyspaces mengimplementasikan dukungan IPv6 melalui arsitektur dual-stack yang mempertahankan kompatibilitas mundur lengkap sekaligus mengaktifkan konektivitas IPv6.

Resolusi DNS di Amazon Keyspaces

Ketika aplikasi terhubung ke titik akhir dual-stack, proses resolusi DNS mengembalikan kedua jenis alamat:

Sebuah Catatan (IPv4)

Alamat IPv4 tradisional untuk kompatibilitas mundur

Catatan AAAA (IPv6)

Alamat IPv6 baru untuk konektivitas modern

Sistem operasi klien dan tumpukan jaringan secara otomatis memilih protokol yang paling tepat berdasarkan konfigurasi lokal, ketersediaan jaringan, dan preferensi sistem.

Protokol Cassandra Query Language (CQL) dengan mulus mendukung konektivitas IPv6 tanpa memerlukan perubahan kode aplikasi.

Pemilihan protokol otomatis

- Aplikasi menentukan titik akhir dual-stack

- Network stack memilih IPv4 atau IPv6 berdasarkan ketersediaan
- Tidak ada modifikasi kode yang diperlukan untuk aplikasi CQL yang ada

Kompatibilitas driver

- Semua driver CQL utama mendukung IPv6 secara transparan
- DataStax driver menangani alamat IPv6 secara native
- Open-source driver bekerja tanpa modifikasi

Konsistensi koneksi

- Tabel sistem mencerminkan protokol koneksi yang digunakan
- Koneksi IPv6 menunjukkan alamat IPv6 di `system.peers`
- Koneksi IPv4 terus menampilkan alamat IPv4

Menggunakan **cqlsh** untuk terhubung ke Amazon Keyspaces

Untuk terhubung ke Amazon Keyspaces menggunakan `cqlsh`, Anda dapat menggunakan file. `cqlsh-expansion` Ini adalah toolkit yang berisi alat Apache Cassandra umum seperti `cqlsh` dan pembantu yang telah dikonfigurasi sebelumnya untuk Amazon Keyspaces sambil mempertahankan kompatibilitas penuh dengan Apache Cassandra. `cqlsh-expansion` Ini mengintegrasikan plugin otentikasi `SiGv4` dan memungkinkan Anda untuk terhubung menggunakan kunci akses IAM alih-alih nama pengguna dan kata sandi. Anda hanya perlu menginstal `cqlsh` skrip untuk membuat koneksi dan bukan distribusi Apache Cassandra penuh, karena Amazon Keyspaces tanpa server. Paket instalasi ringan ini mencakup `cqlsh-expansion` dan `cqlsh` skrip klasik yang dapat Anda instal di platform apa pun yang mendukung Python.

Note

`Murmur3Partitioner` adalah partisi yang direkomendasikan untuk Amazon Keyspaces dan. `cqlsh-expansion` `cqlsh-expansion` Itu tidak mendukung Amazon `KeyspacesDefaultPartitioner`. Untuk informasi selengkapnya, lihat [the section called “Bekerja dengan partisi”](#).

Untuk informasi umum tentang `cqlsh`, lihat [cqlsh: shell CQL](#).

Topik

- [Menggunakan untuk terhubung cqlsh-expansion ke Amazon Keyspaces](#)

- [Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS](#)

Menggunakan untuk terhubung **cqlsh-expansion** ke Amazon Keyspaces

Menginstal dan mengkonfigurasi **cqlsh-expansion**

1. Untuk menginstal paket **cqlsh-expansion** Python, Anda dapat menjalankan perintah. `pip` Ini menginstal **cqlsh-expansion** skrip pada mesin Anda menggunakan `pip install` bersama dengan file yang berisi daftar dependensi. `--user` flag Memberitahu `pip` untuk menggunakan direktori instalasi pengguna Python untuk platform Anda. Pada sistem berbasis Unix, itu harus menjadi `~/.local/` direktori.

Anda memerlukan Python 3 untuk menginstal **cqlsh-expansion**, untuk mengetahui versi Python Anda, gunakan. `Python --version` Untuk menginstal, Anda dapat menjalankan perintah berikut.

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

Outputnya akan terlihat mirip dengan ini.

```
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

```

Jika direktori install tidak ada diPATH, Anda perlu menambahkannya mengikuti instruksi sistem operasi Anda. Di bawah ini adalah salah satu contoh untuk Ubuntu Linux.

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

Untuk mengonfirmasi bahwa paket diinstal, Anda dapat menjalankan perintah berikut.

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

Outputnya akan terlihat seperti ini.

```
cqlsh 6.1.0
```

2. Untuk mengkonfigurasi `cqlsh-expansion`, Anda dapat menjalankan skrip pasca-instal untuk secara otomatis menyelesaikan langkah-langkah berikut:

1. Buat `.cassandra` direktori di direktori home pengguna jika belum ada.
2. Salin file konfigurasi yang telah `cqlshrc` dikonfigurasi sebelumnya ke `.cassandra` direktori.
3. Salin file sertifikat gabungan ke dalam `.cassandra` direktori. Amazon Keyspaces menggunakan sertifikat ini untuk mengonfigurasi koneksi aman dengan Transport Layer Security (TLS). Enkripsi dalam perjalanan menyediakan lapisan perlindungan data tambahan dengan mengenkripsi data Anda saat melakukan perjalanan ke dan dari Amazon Keyspaces. Untuk informasi selengkapnya tentang sertifikat, lihat [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#).

Untuk meninjau skrip terlebih dahulu, Anda dapat mengaksesnya di repo Github di [post_install.py](#)

Untuk menggunakan skrip, Anda dapat menjalankan perintah berikut.

```
cqlsh-expansion.init
```

 Note

Direktori dan file yang dibuat oleh skrip pasca-instal tidak dihapus ketika Anda menghapus instalasi `cqlsh-expansion` dengan `pip uninstall`, dan harus dihapus secara manual.

Menghubungkan ke Amazon Keyspaces menggunakan **`cqlsh-expansion`**

1. Konfigurasi Anda Wilayah AWS dan tambahkan sebagai variabel lingkungan pengguna.

Untuk menambahkan Region default Anda sebagai variabel lingkungan pada sistem berbasis Unix, Anda dapat menjalankan perintah berikut. Untuk contoh ini, kami menggunakan `us-east-1`.

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

Untuk informasi selengkapnya tentang cara mengatur variabel lingkungan, termasuk untuk platform lain, lihat [Cara mengatur variabel lingkungan](#).

2. Temukan titik akhir layanan Anda.

Pilih titik akhir layanan yang sesuai untuk Wilayah Anda. Untuk meninjau titik akhir yang tersedia untuk Amazon Keyspaces, lihat [the section called “Titik akhir layanan”](#). Untuk contoh ini, kita menggunakan `endpointcassandra.us-east-1.amazonaws.com`.

3. Konfigurasi metode otentikasi.

Menghubungkan dengan kunci akses IAM (pengguna IAM, peran, dan identitas federasi) adalah metode yang direkomendasikan untuk meningkatkan keamanan.

Sebelum Anda dapat terhubung dengan kunci akses IAM, Anda harus menyelesaikan langkah-langkah berikut:

- a. Buat pengguna IAM, atau ikuti praktik terbaik dan buat peran IAM yang dapat diasumsikan oleh pengguna IAM. Untuk informasi selengkapnya tentang cara membuat kunci akses IAM, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).
- b. Buat kebijakan IAM yang memberikan peran (atau pengguna IAM) setidaknya akses hanya-baca ke Amazon Keyspaces. Untuk informasi selengkapnya tentang izin yang diperlukan bagi pengguna IAM atau peran untuk terhubung ke Amazon Keyspaces, lihat [the section called “Mengakses tabel Amazon Keyspaces”](#).
- c. Tambahkan kunci akses pengguna IAM ke variabel lingkungan pengguna seperti yang ditunjukkan pada contoh berikut.

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

Untuk informasi selengkapnya tentang cara mengatur variabel lingkungan, termasuk untuk platform lain, lihat [Cara mengatur variabel lingkungan](#).

Note

Jika Anda terhubung dari instans Amazon EC2, Anda juga perlu mengonfigurasi aturan keluar di grup keamanan yang mengizinkan lalu lintas dari instans ke Amazon Keyspaces. Untuk informasi selengkapnya tentang cara melihat dan mengedit aturan keluar EC2, lihat [Menambahkan aturan ke grup keamanan di Panduan Pengguna Amazon EC2](#).

4. Connect ke Amazon Keyspaces menggunakan autentikasi `cqlsh-expansion` dan `SiGv4`.

Untuk terhubung ke Amazon Keyspaces dengan `cqlsh-expansion`, Anda dapat menggunakan perintah berikut. Pastikan untuk mengganti titik akhir layanan dengan titik akhir yang benar untuk Wilayah Anda.

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

Jika koneksi berhasil, Anda akan melihat output yang mirip dengan contoh berikut.

```
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>
```

Jika Anda mengalami kesalahan koneksi, lihat [the section called “Kesalahan koneksi Cqlsh”](#) untuk informasi pemecahan masalah.

- Connect ke Amazon Keyspaces dengan kredensial khusus layanan.

Untuk terhubung dengan kombinasi nama pengguna dan kata sandi tradisional yang digunakan Cassandra untuk otentikasi, Anda harus terlebih dahulu membuat kredensial khusus layanan untuk Amazon Keyspaces seperti yang dijelaskan dalam [the section called “Buat kredensial khusus layanan”](#). Anda juga harus memberikan izin kepada pengguna tersebut untuk mengakses Amazon Keyspaces, untuk informasi selengkapnya lihat [the section called “Mengakses tabel Amazon Keyspaces”](#).

Setelah Anda membuat kredensi dan izin khusus layanan untuk pengguna, Anda harus memperbarui `cqlshrc` file, biasanya ditemukan di jalur direktori pengguna. `~/.cassandra/` Dalam `cqlshrc` file, pergi ke `[authentication]` bagian Cassandra dan komentar modul `SiGv4` dan kelas di bawah `[auth_provider]` menggunakan karakter “;” seperti yang ditunjukkan pada contoh berikut.

```
[auth_provider]

; module = cassandra_sigv4.auth

; classname = SigV4AuthProvider
```

Setelah memperbarui `cqlshrc` file, Anda dapat terhubung ke Amazon Keyspaces dengan kredensial khusus layanan menggunakan perintah berikut.

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

Pembersihan

- Untuk menghapus `cqlsh-expansion` paket Anda dapat menggunakan `pip uninstall` perintah.

```
pip3 uninstall cqlsh-expansion
```

`pip3 uninstall` Perintah tidak menghapus direktori dan file terkait yang dibuat oleh skrip pasca-instal. Untuk menghapus folder dan file yang dibuat oleh skrip pasca-instal, Anda dapat menghapus `.cassandra` direktori.

Cara mengkonfigurasi **cqlsh** koneksi secara manual untuk TLS

Amazon Keyspaces hanya menerima koneksi aman menggunakan Transport Layer Security (TLS). Anda dapat menggunakan `cqlsh-expansion` utilitas yang secara otomatis mengunduh sertifikat untuk Anda dan menginstal file konfigurasi yang telah `cqlshrc` dikonfigurasi sebelumnya. Untuk informasi lebih lanjut, lihat [the section called “Menggunakan cqlsh-expansion”](#) di halaman ini.

Jika Anda ingin mengunduh sertifikat dan mengonfigurasi koneksi secara manual, Anda dapat melakukannya menggunakan langkah-langkah berikut.

1. Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.
 - a. AmazonRootCA1
 - b. AmazonRootCA2
 - c. AmazonRootCA3
 - d. AmazonRootCA4
 - e. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.

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

 Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

2. Gabungkan semua sertifikat yang diunduh menjadi satu pem file dengan nama *keyspaces-bundle.pem* dalam contoh kami. Anda dapat melakukan ini dengan menjalankan perintah berikut. Catat jalur ke file, Anda perlu ini nanti.

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

3. Buka file `cqlshrc` konfigurasi di direktori home Cassandra, misalnya `${HOME}/.cassandra/cqlshrc` dan tambahkan baris berikut.

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

[ssl]
validate = true
certfile = path_to_file/keyspaces-bundle.pem
```

Menggunakan untuk terhubung AWS CLI ke Amazon Keyspaces

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengontrol beberapa AWS layanan dari baris perintah dan mengotomatiskannya melalui skrip. Dengan Amazon Keyspaces Anda dapat menggunakan operasi AWS CLI for data definition language (DDL), seperti membuat tabel. Selain itu, Anda dapat menggunakan layanan infrastruktur sebagai kode (IaC) dan alat seperti AWS CloudFormation dan Terraform.

Sebelum Anda dapat menggunakan AWS CLI dengan Amazon Keyspaces, Anda harus mendapatkan ID kunci akses dan kunci akses rahasia. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Untuk daftar lengkap semua perintah yang tersedia untuk Amazon Keyspaces di AWS CLI, lihat Referensi [AWS CLI Perintah](#).

Topik

- [Mengunduh dan Mengkonfigurasi AWS CLI](#)
- [Menggunakan AWS CLI Keyspaces dengan Amazon](#)

Mengunduh dan Mengkonfigurasi AWS CLI

AWS CLI Tersedia di <https://aws.amazon.com/cli>. Alat ini berjalan di Windows, macOS, atau Linux. Setelah mengunduh AWS CLI, ikuti langkah-langkah ini untuk menginstal dan mengkonfigurasinya:

1. Pergi ke [Panduan AWS Command Line Interface Pengguna](#)
2. Ikuti petunjuk untuk [Menginstal AWS CLI](#) dan [Mengkonfigurasi AWS CLI](#)

Menggunakan AWS CLI Keyspaces dengan Amazon

Format baris perintah terdiri dari nama operasi Amazon Keyspaces diikuti oleh parameter untuk operasi itu. AWS CLI Mendukung sintaks singkatan untuk nilai parameter, serta JSON. Contoh Amazon Keyspaces berikut menggunakan sintaks AWS CLI singkatan. Untuk informasi selengkapnya, lihat [Menggunakan sintaks singkatan dengan CLI. AWS](#)

Perintah berikut membuat keyspace dengan katalog nama.

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

Perintah mengembalikan sumber daya Amazon Resource Name (ARN) dalam output.

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

Untuk mengonfirmasi bahwa katalog keyspace ada, Anda dapat menggunakan perintah berikut.

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

Output dari perintah mengembalikan nilai-nilai berikut.

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

Perintah berikut membuat tabel dengan nama `book_awards`. Kunci partisi tabel terdiri dari kolom `year` dan `award` dan kunci pengelompokan terdiri dari kolom `category` dan `rank`, kedua kolom pengelompokan menggunakan urutan urutan menaik. (Agar mudah dibaca, perintah panjang di bagian ini dipecah menjadi baris terpisah.)

```
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}]'
```

Perintah ini menghasilkan output berikut.

```
{
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/table/
book_awards"
}
```

Untuk mengonfirmasi metadata dan properti tabel, Anda dapat menggunakan perintah berikut.

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

Perintah ini mengembalikan output berikut.

```
{
  "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": ""
}
}

```

Saat membuat tabel dengan skema kompleks, akan sangat membantu untuk memuat definisi skema tabel dari file JSON. Berikut ini adalah contohnya. Unduh contoh definisi skema file JSON dari [schema_definition.zip](#) dan ekstrak `schema_definition.json`, perhatikan jalur ke file. Dalam contoh ini, file JSON definisi skema terletak di direktori saat ini. Untuk opsi jalur file yang berbeda, lihat [Cara memuat parameter dari file](#).

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

```
--table-name 'book_awards' --schema-definition 'file://
schema_definition.json'
```

Contoh berikut menunjukkan cara membuat tabel sederhana dengan nama MyTable dengan opsi tambahan. Perhatikan bahwa perintah dipecah menjadi baris terpisah untuk meningkatkan keterbacaan. Perintah ini menunjukkan cara membuat tabel dan:

- mengatur mode kapasitas tabel
- aktifkan Point-in-time pemulihan untuk tabel
- mengatur nilai Time to Live (TTL) default untuk tabel menjadi satu tahun
- tambahkan dua tag untuk tabel

```
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'
```

Contoh ini menunjukkan cara membuat tabel baru yang menggunakan kunci terkelola pelanggan untuk enkripsi dan mengaktifkan TTL untuk memungkinkan Anda mengatur tanggal kedaluwarsa kolom dan baris. Untuk menjalankan sampel ini, Anda harus mengganti ARN sumber daya untuk kunci yang dikelola pelanggan dengan AWS KMS kunci Anda sendiri dan memastikan Amazon Keyspaces memiliki akses ke sana.

```
aws keyspaces create-table --keyspace-name 'catalog' --table-name 'myTable'
    --schema-definition 'allColumns=[{name=id,type=int},{name=name,type=text},
{name=date,type=timestamp}],partitionKeys=[{name=id}]'
    --encryption-specification
    'type=CUSTOMER_MANAGED_KMS_KEY,kmsKeyIdentifier=arn:aws:kms:us-
east-1:111122223333:key/11111111-2222-3333-4444-555555555555'
    --ttl 'status=ENABLED'
```

Menggunakan API untuk terhubung ke Amazon Keyspaces

Anda dapat menggunakan AWS SDK dan AWS Command Line Interface (AWS CLI) untuk bekerja secara interaktif dengan Amazon Keyspaces. Anda dapat menggunakan API untuk operasi definisi bahasa data (DDL), seperti membuat keyspace atau tabel. Selain itu, Anda dapat menggunakan layanan infrastruktur sebagai kode (IaC) dan alat seperti AWS CloudFormation dan Terraform.

Sebelum Anda dapat menggunakan AWS CLI dengan Amazon Keyspaces, Anda harus mendapatkan ID kunci akses dan kunci akses rahasia. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Untuk daftar lengkap semua operasi yang tersedia untuk Amazon Keyspaces di API, lihat Referensi API Amazon [Keyspaces](#).

Menggunakan driver klien Cassandra untuk mengakses Amazon Keyspaces secara terprogram

Anda dapat menggunakan banyak driver Cassandra sumber terbuka pihak ketiga untuk terhubung ke Amazon Keyspaces. Amazon Keyspaces kompatibel dengan driver Cassandra yang mendukung Apache Cassandra versi 3.11.2. Ini adalah driver dan versi terbaru yang telah kami uji dan rekomendasikan untuk digunakan dengan 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

Untuk informasi lebih lanjut tentang driver Cassandra, lihat driver [Apache Cassandra](#) Client.

Note

Untuk membantu Anda memulai, Anda dapat melihat dan mengunduh contoh kode ujung ke ujung yang membuat koneksi ke Amazon Keyspaces dengan driver populer. Lihat [contoh Amazon Keyspaces di GitHub](#).

Tutorial dalam Bab ini mencakup kueri CQL sederhana untuk mengonfirmasi bahwa koneksi ke Amazon Keyspaces telah berhasil dibuat. Untuk mempelajari cara bekerja dengan ruang kunci dan tabel setelah Anda terhubung ke titik akhir Amazon Keyspaces, lihat. [Referensi bahasa CQL](#) Untuk tutorial langkah demi langkah yang menunjukkan cara menyambung ke Amazon Keyspaces dari titik akhir Amazon VPC, lihat. [the section called “Menghubungkan dengan titik akhir VPC”](#)

Topik

- [Menggunakan driver klien Cassandra Java untuk mengakses Amazon Keyspaces secara terprogram](#)
- [Menggunakan driver klien Cassandra Python untuk mengakses Amazon Keyspaces secara terprogram](#)
- [Menggunakan driver Node.js klien Cassandra untuk mengakses Amazon Keyspaces secara terprogram](#)
- [Menggunakan driver klien Cassandra .NET Core untuk mengakses Amazon Keyspaces secara terprogram](#)
- [Menggunakan driver klien Cassandra Go untuk mengakses Amazon Keyspaces secara terprogram](#)
- [Menggunakan driver klien Cassandra Perl untuk mengakses Amazon Keyspaces secara terprogram](#)

Menggunakan driver klien Cassandra Java untuk mengakses Amazon Keyspaces secara terprogram

Bagian ini menunjukkan kepada Anda cara terhubung ke Amazon Keyspaces dengan menggunakan driver klien Java.

Note

Java 17 dan DataStax Java Driver 4.17 saat ini hanya dalam dukungan Beta. Untuk informasi selengkapnya, lihat https://docs.datastax.com/en/developer/java-driver/4.17/upgrade_guide/.

Untuk memberikan kredensial kepada pengguna dan aplikasi untuk akses terprogram ke sumber daya Amazon Keyspaces, Anda dapat melakukan salah satu hal berikut:

- Buat kredensial khusus layanan yang terkait dengan pengguna AWS Identity and Access Management (IAM) tertentu.

- Untuk keamanan yang ditingkatkan, kami sarankan untuk membuat kunci akses IAM untuk identitas IAM yang digunakan di semua layanan. AWS Plugin otentikasi Amazon Keyspaces SigV4 untuk driver klien Cassandra memungkinkan Anda untuk mengautentikasi panggilan ke Amazon Keyspaces menggunakan kunci akses IAM alih-alih nama pengguna dan kata sandi. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Note

Untuk contoh cara menggunakan Amazon Keyspaces dengan Spring Boot, lihat. <https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring>

Topik

- [Sebelum Anda mulai](#)
- [Step-by-step tutorial untuk terhubung ke Amazon Keyspaces menggunakan driver DataStax Java untuk Apache Cassandra menggunakan kredensial khusus layanan](#)
- [Step-by-step tutorial untuk terhubung ke Amazon Keyspaces menggunakan driver DataStax Java 4.x untuk Apache Cassandra dan plugin otentikasi SiGv4](#)
- [Connect ke Amazon Keyspaces menggunakan driver DataStax Java 3.x untuk Apache Cassandra dan plugin otentikasi SiGv4](#)

Sebelum Anda mulai

Untuk terhubung ke Amazon Keyspaces, Anda harus menyelesaikan tugas-tugas berikut sebelum dapat memulai.

1. Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien.
 - a. Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.
 1. AmazonRootCA1
 2. AmazonRootCA2
 3. AmazonRootCA3
 4. AmazonRootCA4
 5. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.

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

 Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

- b. Ubah sertifikat digital menjadi file TrustStore dan tambahkan ke keystore.

```
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
```

Pada langkah terakhir, Anda perlu membuat kata sandi untuk keystore dan mempercayai setiap sertifikat. Perintah interaktif terlihat seperti ini.

```

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. Lampirkan file TrustStore dalam argumen JVM:

```

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

```

Step-by-step tutorial untuk terhubung ke Amazon Keyspaces menggunakan driver DataStax Java untuk Apache Cassandra menggunakan kredensial khusus layanan

Tutorial langkah demi langkah berikut memandu Anda melalui koneksi ke Amazon Keyspaces menggunakan driver Java untuk Cassandra menggunakan kredensial khusus layanan. Secara khusus, Anda akan menggunakan versi 4.0 dari driver DataStax Java untuk Apache Cassandra.

Topik

- [Langkah 1: Prasyarat](#)
- [Langkah 2: Konfigurasi driver](#)
- [Langkah 3: Jalankan aplikasi sampel](#)

Langkah 1: Prasyarat

Untuk mengikuti tutorial ini, Anda perlu menghasilkan kredensial khusus layanan dan menambahkan driver DataStax Java untuk Apache Cassandra ke proyek Java Anda.

- Hasilkan kredensial khusus layanan untuk pengguna IAM Amazon Keyspaces Anda dengan menyelesaikan langkah-langkahnya. [the section called “Buat kredensial khusus layanan”](#) Jika Anda lebih suka menggunakan kunci akses IAM untuk otentikasi, lihat. [the section called “Plugin otentikasi untuk Java 4.x”](#)
- Tambahkan driver DataStax Java untuk Apache Cassandra ke proyek Java Anda. Pastikan Anda menggunakan versi driver yang mendukung Apache Cassandra 3.11.2. Untuk informasi lebih lanjut, lihat [driver DataStax Java untuk dokumentasi Apache Cassandra](#).

Langkah 2: Konfigurasi driver

Anda dapat menentukan pengaturan untuk driver DataStax Java Cassandra dengan membuat file konfigurasi untuk aplikasi Anda. File konfigurasi ini mengganti pengaturan default dan memberi tahu driver untuk terhubung ke titik akhir layanan Amazon Keyspaces menggunakan port 9142. Untuk daftar titik akhir layanan yang tersedia, lihat [the section called “Titik akhir layanan”](#).

Buat file konfigurasi dan simpan file di folder sumber daya aplikasi—misalnya, `src/main/resources/application.conf` Buka `application.conf` dan tambahkan pengaturan konfigurasi berikut.

1. Penyedia otentikasi — Buat penyedia otentikasi dengan kelas. `PlainTextAuthProvider` *ServiceUserName* dan *ServicePassword* harus cocok dengan nama pengguna dan kata

sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkah di [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)

 Note

Anda dapat menggunakan kredensial jangka pendek dengan menggunakan plugin otentikasi untuk driver DataStax Java untuk Apache Cassandra alih-alih kredensial hardcoding di file konfigurasi driver Anda. Untuk mempelajari lebih lanjut, ikuti instruksi untuk [the section called "Plugin otentikasi untuk Java 4.x"](#).

2. Pusat data lokal — Tetapkan nilai `local-datacenter` untuk Wilayah yang Anda sambungkan. Misalnya, jika aplikasi terhubung `keus-east-2.amazonaws.com`, maka atur pusat data lokal ke `us-east-2`. Untuk semua yang tersedia Wilayah AWS, lihat [???](#). Atur `slow-replica-avoidance = false` untuk memuat keseimbangan terhadap lebih sedikit node.
3. SSL/TLS— Inisialisasi `SslEngineFactory` dengan menambahkan bagian dalam file konfigurasi dengan satu baris yang menentukan kelas dengan `class = DefaultSslEngineFactory`. Berikan jalur ke file `TrustStore` dan kata sandi yang Anda buat sebelumnya. Amazon Keyspaces tidak mendukung `hostname-validation` peer, jadi setel opsi ini ke `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

Alih-alih menambahkan jalur ke TrustStore di file konfigurasi, Anda juga dapat menambahkan jalur TrustStore langsung di kode aplikasi atau Anda dapat menambahkan jalur ke TrustStore ke argumen JVM Anda.

Langkah 3: Jalankan aplikasi sampel

Contoh kode ini menunjukkan aplikasi baris perintah sederhana yang membuat kumpulan koneksi ke Amazon Keyspaces dengan menggunakan file konfigurasi yang kita buat sebelumnya. Ini menegaskan bahwa koneksi dibuat dengan menjalankan kueri sederhana.

```
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

Gunakan `try` blok untuk membuat koneksi untuk memastikan bahwa itu selalu tertutup. Jika Anda tidak menggunakan `try` blok, ingatlah untuk menutup koneksi Anda untuk menghindari kebocoran sumber daya.

Step-by-step tutorial untuk terhubung ke Amazon Keyspaces menggunakan driver DataStax Java 4.x untuk Apache Cassandra dan plugin otentikasi SiGv4

Bagian berikut menjelaskan cara menggunakan plugin otentikasi SiGv4 untuk driver DataStax Java 4.x open-source untuk Apache Cassandra untuk mengakses Amazon Keyspaces (untuk Apache Cassandra). Plugin tersedia dari [GitHub repositori](#).

Plugin otentikasi SiGv4 memungkinkan Anda menggunakan kredensial IAM untuk pengguna atau peran saat menghubungkan ke Amazon Keyspaces. Alih-alih memerlukan nama pengguna dan kata sandi, plugin ini menandatangani permintaan API menggunakan kunci akses. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Langkah 1: Prasyarat

Untuk mengikuti tutorial ini, Anda harus menyelesaikan tugas-tugas berikut.

- Jika Anda belum melakukannya, buat kredensial untuk pengguna IAM Anda atau peran mengikuti langkah-langkah di [the section called “Buat kredensi IAM untuk otentikasi AWS”](#). Tutorial ini mengasumsikan bahwa kunci akses disimpan sebagai variabel lingkungan. Untuk informasi selengkapnya, lihat [the section called “Kelola kunci akses”](#).
- Tambahkan driver DataStax Java untuk Apache Cassandra ke proyek Java Anda. Pastikan Anda menggunakan versi driver yang mendukung Apache Cassandra 3.11.2. Untuk informasi selengkapnya, lihat [Driver DataStax Java untuk dokumentasi Apache Cassandra](#).
- Tambahkan plugin otentikasi ke aplikasi Anda. Plugin otentikasi mendukung versi 4.x dari driver DataStax Java untuk Apache Cassandra. Jika Anda menggunakan Apache Maven, atau sistem build yang dapat menggunakan dependensi Maven, tambahkan dependensi berikut ke file Anda. `pom.xml`

 Important

Ganti versi plugin dengan versi terbaru seperti yang ditunjukkan di [GitHub repositori](#).

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

Langkah 2: Konfigurasi driver

Anda dapat menentukan pengaturan untuk driver DataStax Java Cassandra dengan membuat file konfigurasi untuk aplikasi Anda. File konfigurasi ini mengganti pengaturan default dan memberi tahu driver untuk terhubung ke titik akhir layanan Amazon Keyspaces menggunakan port 9142. Untuk daftar titik akhir layanan yang tersedia, lihat [the section called “Titik akhir layanan”](#).

Buat file konfigurasi dan simpan file di folder sumber daya aplikasi—misalnya, `src/main/resources/application.conf`. Buka `application.conf` dan tambahkan pengaturan konfigurasi berikut.

1. Penyedia otentikasi - Atur `advanced.auth-provider.class` ke instance baru. `software.aws.mcs.auth.SigV4AuthProvider` `SigV4AuthProvider` Ini adalah penanganan otentikasi yang disediakan oleh plugin untuk melakukan otentikasi SiGv4.
2. Pusat data lokal — Tetapkan nilai `local-datacenter` untuk Wilayah yang Anda sambungkan. Misalnya, jika aplikasi terhubung ke `cassandra.us-east-2.amazonaws.com`, maka atur pusat data lokal ke `us-east-2`. Untuk semua yang tersedia Wilayah AWS, lihat [???](#). Atur `slow-replica-avoidance = false` untuk memuat keseimbangan terhadap semua node yang tersedia.
3. Idempotence - Mengatur default idempotence untuk aplikasi untuk mengkonfigurasi driver `true` untuk selalu mencoba kembali gagal /permintaan. `read/write prepare/execute` Ini adalah praktik terbaik untuk aplikasi terdistribusi yang membantu menangani kegagalan sementara dengan mencoba kembali permintaan yang gagal.
4. SSL/TLS— Inisialisasi `SSLConnectionFactory` dengan menambahkan bagian dalam file konfigurasi dengan satu baris yang menentukan kelas dengan `class = DefaultSslConnectionFactory`. Berikan jalur ke file `TrustStore` dan kata sandi yang Anda buat sebelumnya. Amazon Keyspaces tidak mendukung `hostname-validation peer`, jadi setel opsi ini ke `false`.
5. Koneksi — Buat setidaknya 3 koneksi lokal per titik akhir dengan pengaturan `local.size = 3`. Ini adalah praktik terbaik yang membantu aplikasi Anda menangani ledakan overhead dan lalu

lintas. Untuk informasi selengkapnya tentang cara menghitung berapa banyak koneksi lokal per titik akhir yang dibutuhkan aplikasi Anda berdasarkan pola lalu lintas yang diharapkan, lihat [the section called “Cara mengkonfigurasi koneksi”](#).

6. Kebijakan coba lagi — Terapkan `AmazonKeyspacesExponentialRetryPolicy` kebijakan coba ulang Amazon Keyspaces alih-alih yang disertakan dengan driver `DefaultRetryPolicy` Cassandra. Ini memungkinkan Anda untuk mengonfigurasi jumlah upaya coba lagi untuk `AmazonKeyspacesExponentialRetryPolicy` yang memenuhi kebutuhan Anda. Secara default, jumlah percobaan ulang untuk `AmazonKeyspacesExponentialRetryPolicy` diatur ke 3. Untuk informasi selengkapnya, lihat [the section called “Cara mengonfigurasi kebijakan coba lagi”](#).
7. Pernyataan yang disiapkan - Setel `prepare-on-all-nodes` ke `false` untuk mengoptimalkan penggunaan jaringan.

```
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

Alih-alih menambahkan jalur ke TrustStore di file konfigurasi, Anda juga dapat menambahkan jalur TrustStore langsung di kode aplikasi atau Anda dapat menambahkan jalur ke TrustStore ke argumen JVM Anda.

Langkah 3: Jalankan aplikasi

Contoh kode ini menunjukkan aplikasi baris perintah sederhana yang membuat kumpulan koneksi ke Amazon Keyspaces dengan menggunakan file konfigurasi yang kita buat sebelumnya. Ini menegaskan bahwa koneksi dibuat dengan menjalankan kueri sederhana.

```
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

Gunakan try blok untuk membuat koneksi untuk memastikan bahwa itu selalu tertutup. Jika Anda tidak menggunakan try blok, ingatlah untuk menutup koneksi Anda untuk menghindari kebocoran sumber daya.

Connect ke Amazon Keyspaces menggunakan driver DataStax Java 3.x untuk Apache Cassandra dan plugin otentikasi SiGv4

Bagian berikut menjelaskan cara menggunakan plugin otentikasi SiGv4 untuk driver DataStax Java sumber terbuka 3.x untuk Apache Cassandra untuk mengakses Amazon Keyspaces. Plugin tersedia dari [GitHub repositori](#).

Plugin otentikasi SiGv4 memungkinkan Anda menggunakan kredensial IAM untuk pengguna dan peran saat menghubungkan ke Amazon Keyspaces. Alih-alih memerlukan nama pengguna dan kata sandi, plugin ini menandatangani permintaan API menggunakan kunci akses. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Langkah 1: Prasyarat

Untuk menjalankan contoh kode ini, pertama-tama Anda harus menyelesaikan tugas-tugas berikut.

- Buat kredensial untuk pengguna atau peran IAM Anda mengikuti langkah-langkah di [the section called “Buat kredensi IAM untuk otentikasi AWS”](#) Tutorial ini mengasumsikan bahwa kunci akses

disimpan sebagai variabel lingkungan. Untuk informasi selengkapnya, lihat [the section called “Kelola kunci akses”](#).

- Ikuti langkah-langkah di [the section called “Sebelum Anda mulai”](#) untuk mengunduh sertifikat digital, mengonversinya menjadi file TrustStore, dan melampirkan keystore dalam argumen JVM ke aplikasi Anda.
- Tambahkan driver DataStax Java untuk Apache Cassandra ke proyek Java Anda. Pastikan Anda menggunakan versi driver yang mendukung Apache Cassandra 3.11.2. Untuk informasi selengkapnya, lihat [Driver DataStax Java untuk dokumentasi Apache Cassandra](#).
- Tambahkan plugin otentikasi ke aplikasi Anda. Plugin otentikasi mendukung versi 3.x dari driver DataStax Java untuk Apache Cassandra. Jika Anda menggunakan Apache Maven, atau sistem build yang dapat menggunakan dependensi Maven, tambahkan dependensi berikut ke file Anda. `pom.xml` Ganti versi plugin dengan versi terbaru seperti yang ditunjukkan di [GitHub repositori](#).

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

Langkah 2: Jalankan aplikasi

Contoh kode ini menunjukkan aplikasi baris perintah sederhana yang membuat kumpulan koneksi ke Amazon Keyspaces. Ini menegaskan bahwa koneksi dibuat dengan menjalankan kueri sederhana.

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

Catatan penggunaan:

Untuk daftar titik akhir yang tersedia, lihat [the section called “Titik akhir layanan”](#).

Lihat repositori berikut untuk kebijakan driver Java yang bermanfaat, contoh, dan praktik terbaik saat menggunakan Driver Java dengan Amazon Keyspaces: <https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers>

Menggunakan driver klien Cassandra Python untuk mengakses Amazon Keyspaces secara terprogram

Di bagian ini, kami menunjukkan cara terhubung ke Amazon Keyspaces menggunakan driver klien Python. Untuk memberikan kredensial kepada pengguna dan aplikasi untuk akses terprogram ke sumber daya Amazon Keyspaces, Anda dapat melakukan salah satu hal berikut:

- Buat kredensial khusus layanan yang terkait dengan pengguna AWS Identity and Access Management (IAM) tertentu.
- Untuk keamanan yang ditingkatkan, kami sarankan untuk membuat kunci akses IAM untuk pengguna IAM atau peran yang digunakan di semua AWS layanan. Plugin otentikasi Amazon Keyspaces SigV4 untuk driver klien Cassandra memungkinkan Anda untuk mengautentikasi panggilan ke Amazon Keyspaces menggunakan kunci akses IAM alih-alih nama pengguna dan kata sandi. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Topik

- [Sebelum kamu memulai](#)
- [Connect ke Amazon Keyspaces menggunakan driver Python untuk Apache Cassandra dan kredensial khusus layanan](#)
- [Connect ke Amazon Keyspaces menggunakan driver DataStax Python untuk Apache Cassandra dan plugin otentikasi SiGv4](#)

Sebelum kamu memulai

Anda harus menyelesaikan tugas berikut sebelum Anda dapat memulai.

Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien. Untuk terhubung ke Amazon Keyspaces menggunakan TLS, Anda perlu mengunduh sertifikat digital Amazon dan mengonfigurasi driver Python untuk menggunakan TLS.

Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.

```
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.secdserver.net/repository/sf-class2-root.crt
```

Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang dikeluarkan di bawah

Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

Gabungkan semua sertifikat yang diunduh menjadi satu pem file dengan nama *keyspaces-bundle.pem* dalam contoh kami. Anda dapat melakukan ini dengan menjalankan perintah berikut. Catat jalur ke file, Anda perlu ini nanti.

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

Connect ke Amazon Keyspaces menggunakan driver Python untuk Apache Cassandra dan kredensial khusus layanan

Contoh kode berikut menunjukkan cara menyambung ke Amazon Keyspaces dengan driver klien Python dan kredensial khusus layanan.

```
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)
```

Catatan penggunaan:

1. Ganti "*path_to_file/keyspaces-bundle.pem*" dengan jalur ke file sertifikat gabungan yang disimpan di langkah pertama.
2. Pastikan bahwa *ServiceUserName* dan *ServicePassword* mencocokkan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkahnya. [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)
3. Untuk daftar titik akhir yang tersedia, lihat [the section called "Titik akhir layanan"](#).

Connect ke Amazon Keyspaces menggunakan driver DataStax Python untuk Apache Cassandra dan plugin otentikasi SiGv4

Bagian berikut menunjukkan cara menggunakan plugin otentikasi SiGv4 untuk driver DataStax Python open-source untuk Apache Cassandra untuk mengakses Amazon Keyspaces (untuk Apache Cassandra).

Jika Anda belum melakukannya, mulailah dengan membuat kredensial untuk peran IAM Anda mengikuti langkah-langkah di [the section called "Buat kredensi IAM untuk otentikasi AWS"](#) Tutorial ini menggunakan kredensi sementara, yang membutuhkan peran IAM. Untuk informasi selengkapnya tentang kredensial sementara, lihat [the section called "Buat kredensi sementara untuk terhubung ke Amazon Keyspaces"](#)

[Kemudian, tambahkan plugin otentikasi Python SiGv4 ke lingkungan Anda dari repositori. GitHub](#)

```
pip install cassandra-sigv4
```

Contoh kode berikut menunjukkan cara terhubung ke Amazon Keyspaces dengan menggunakan driver DataStax Python open-source untuk Cassandra dan plugin otentikasi SiGv4. Plugin tergantung pada AWS SDK untuk Python (Boto3). Ini digunakan `boto3.session` untuk mendapatkan kredensial sementara.

```
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)

```

Catatan penggunaan:

1. Ganti "*path_to_file/keyspaces-bundle.pem*" dengan jalur ke sertifikat yang disimpan di langkah pertama.
2. Pastikan bahwa *aws_access_key_id*, *aws_secret_access_key*, dan *aws_session_token* cocok Access Key, Secret Access Key, dan Session Token Anda dapatkan menggunakan `boto3.session`. Untuk informasi selengkapnya, lihat [Kredensyal](#) di AWS SDK untuk Python (Boto3)
3. Untuk daftar titik akhir yang tersedia, lihat [the section called "Titik akhir layanan"](#).

Menggunakan driver Node.js klien Cassandra untuk mengakses Amazon Keyspaces secara terprogram

Bagian ini menunjukkan kepada Anda cara terhubung ke Amazon Keyspaces dengan menggunakan driver Node.js klien. Untuk memberikan kredensyal kepada pengguna dan aplikasi untuk akses terprogram ke sumber daya Amazon Keyspaces, Anda dapat melakukan salah satu hal berikut:

- Buat kredensyal khusus layanan yang terkait dengan pengguna AWS Identity and Access Management (IAM) tertentu.

- Untuk keamanan yang ditingkatkan, kami sarankan untuk membuat kunci akses IAM untuk pengguna IAM atau peran yang digunakan di semua AWS layanan. Plugin otentikasi Amazon Keyspaces SigV4 untuk driver klien Cassandra memungkinkan Anda untuk mengautentikasi panggilan ke Amazon Keyspaces menggunakan kunci akses IAM alih-alih nama pengguna dan kata sandi. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Topik

- [Sebelum kamu memulai](#)
- [Connect ke Amazon Keyspaces menggunakan Node.js DataStax driver untuk Apache Cassandra dan kredensial khusus layanan](#)
- [Connect ke Amazon Keyspaces menggunakan DataStax Node.js driver untuk Apache Cassandra dan plugin otentikasi SiGv4](#)

Sebelum kamu memulai

Anda harus menyelesaikan tugas berikut sebelum Anda dapat memulai.

Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien. Untuk terhubung ke Amazon Keyspaces menggunakan TLS, Anda perlu mengunduh sertifikat digital Amazon dan mengonfigurasi driver Python untuk menggunakan TLS.

Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.

```
curl -O https://www.amazontrust.com/repository/AmazonRootCA1.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA2.pem
curl -O https://www.amazontrust.com/repository/AmazonRootCA3.pem
```

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

Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

Gabungkan semua sertifikat yang diunduh menjadi satu pem file dengan nama *keyspaces-bundle.pem* dalam contoh kami. Anda dapat melakukan ini dengan menjalankan perintah berikut. Catat jalur ke file, Anda perlu ini nanti.

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

Connect ke Amazon Keyspaces menggunakan Node.js DataStax driver untuk Apache Cassandra dan kredensial khusus layanan

Konfigurasi driver Anda untuk menggunakan file sertifikat gabungan *keyspaces-bundle.pem* untuk TLS dan autentikasi menggunakan kredensial khusus layanan. Contoh:

```
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}`));
```

Catatan penggunaan:

1. Ganti "*path_to_file/keyspaces-bundle.pem*" dengan jalur ke file sertifikat gabungan yang disimpan di langkah pertama.
2. Pastikan bahwa *ServiceUserName* dan *ServicePassword* mencocokkan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkahnya. [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)
3. Untuk daftar titik akhir yang tersedia, lihat [the section called "Titik akhir layanan"](#).

Connect ke Amazon Keyspaces menggunakan DataStax Node.js driver untuk Apache Cassandra dan plugin otentikasi SiGv4

Bagian berikut menunjukkan cara menggunakan plugin otentikasi SiGv4 untuk DataStax Node.js driver open-source untuk Apache Cassandra untuk mengakses Amazon Keyspaces (untuk Apache Cassandra).

Jika Anda belum melakukannya, buat kredensial untuk pengguna IAM Anda atau peran mengikuti langkah-langkah di [the section called "Buat kredensi IAM untuk otentikasi AWS "](#)

[Tambahkan plugin otentikasi Node.js SiGv4 ke aplikasi Anda dari repositori. GitHub](#) Plugin mendukung versi 4.x DataStax Node.js driver untuk Cassandra dan tergantung pada SDK untuk AWS Node.js Ini digunakan `AWSCredentialsProvider` untuk mendapatkan kredensi.

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

Contoh kode ini menunjukkan cara mengatur Region-specific instance SigV4AuthProvider sebagai penyedia otentikasi.

```
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}`));
```

Catatan penggunaan:

1. Ganti "*path_to_file/keyspaces-bundle.pem*" dengan jalur ke sertifikat yang disimpan di langkah pertama.
2. Pastikan bahwa *accessKeyId* dan *secretAccessKey* cocok dengan Kunci Akses dan Kunci Akses Rahasia yang Anda peroleh menggunakan `AWSCredentialsProvider`. Untuk informasi selengkapnya, lihat [Menyetel Kredensial Node.js di AWS](#) SDK untuk masuk. JavaScript Node.js

3. Untuk menyimpan kunci akses di luar kode, lihat praktik terbaik di [the section called “Kelola kunci akses”](#).
4. Untuk daftar titik akhir yang tersedia, lihat [the section called “Titik akhir layanan”](#).

Menggunakan driver klien Cassandra .NET Core untuk mengakses Amazon Keyspaces secara terprogram

Bagian ini menunjukkan kepada Anda cara terhubung ke Amazon Keyspaces dengan menggunakan driver klien .NET Core. Langkah-langkah pengaturan akan bervariasi tergantung pada lingkungan dan sistem operasi Anda, Anda mungkin harus memodifikasinya sesuai. Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien. Untuk terhubung ke Amazon Keyspaces menggunakan TLS, konfigurasi driver Anda untuk menggunakan penyimpanan kepercayaan sistem, yang mencakup Amazon Root CA 1-4.

1. Instal CassandraCSharpDriver melalui nuget, menggunakan konsol nuget.

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

2. Contoh berikut menggunakan proyek konsol .NET Core C # untuk terhubung ke Amazon Keyspaces dan menjalankan kueri.

```
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);
}
}
}
}

```

Catatan penggunaan:

1. Pastikan Anda menggunakan toko kepercayaan sistem default, yang mencakup Amazon Root CA 1-4.
2. Pastikan bahwa *ServiceUserName* dan *ServicePassword* mencocokkan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkahnya. [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)
3. Untuk daftar titik akhir yang tersedia, lihat [the section called "Titik akhir layanan"](#).

Menggunakan driver klien Cassandra Go untuk mengakses Amazon Keyspaces secara terprogram

Bagian ini menunjukkan kepada Anda cara terhubung ke Amazon Keyspaces dengan menggunakan driver klien Go Cassandra. Untuk memberikan kredensial kepada pengguna dan aplikasi untuk

akses terprogram ke sumber daya Amazon Keyspaces, Anda dapat melakukan salah satu hal berikut:

- Buat kredensial khusus layanan yang terkait dengan pengguna AWS Identity and Access Management (IAM) tertentu.
- Untuk keamanan yang ditingkatkan, kami sarankan untuk membuat kunci akses IAM untuk prinsipal IAM yang digunakan di semua layanan. AWS Plugin otentikasi Amazon Keyspaces SigV4 untuk driver klien Cassandra memungkinkan Anda untuk mengautentikasi panggilan ke Amazon Keyspaces menggunakan kunci akses IAM alih-alih nama pengguna dan kata sandi. Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Topik

- [Sebelum kamu memulai](#)
- [Connect ke Amazon Keyspaces menggunakan driver Gocql untuk Apache Cassandra dan kredensial khusus layanan](#)
- [Connect ke Amazon Keyspaces menggunakan driver Go untuk Apache Cassandra dan plugin otentikasi SiGv4](#)

Sebelum kamu memulai

Anda harus menyelesaikan tugas berikut sebelum Anda dapat memulai.

Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien. Untuk terhubung ke Amazon Keyspaces menggunakan TLS, Anda perlu mengunduh sertifikat digital Amazon dan mengonfigurasi driver Go untuk menggunakan TLS.

Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.

1. AmazonRootCA1
2. AmazonRootCA2
3. AmazonRootCA3
4. AmazonRootCA4
5. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.


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

Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

Gabungkan semua sertifikat yang diunduh menjadi satu pem file dengan nama *keyspaces-bundle.pem* dalam contoh kami. Anda dapat melakukan ini dengan menjalankan perintah berikut. Catat jalur ke file, Anda perlu ini nanti.

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

Connect ke Amazon Keyspaces menggunakan driver Gocql untuk Apache Cassandra dan kredensial khusus layanan

1. Buat direktori untuk aplikasi Anda.

```
mkdir ./gocqlexample
```

2. Arahkan ke direktori baru.

```
cd gocqlexample
```

3. Buat file untuk aplikasi Anda.

```
touch cqlapp.go
```

4. Unduh driver Go.

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

5. Tambahkan kode contoh berikut ke file cqlapp.go.

```
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()
}
```

Catatan penggunaan:

- a. Ganti "*path_to_file/keyspaces-bundle.pem*" dengan jalur ke file sertifikat gabungan yang disimpan di langkah pertama.
 - b. Pastikan bahwa *ServiceUserName* dan *ServicePassword* mencocokkan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkahnya. [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)
 - c. Untuk daftar titik akhir yang tersedia, lihat [the section called “Titik akhir layanan”](#).
6. Bangun programnya.

```
go build cqlapp.go
```

7. Jalankan program.

```
./cqlapp
```

Connect ke Amazon Keyspaces menggunakan driver Go untuk Apache Cassandra dan plugin otentikasi SiGv4

Contoh kode berikut menunjukkan cara menggunakan plugin otentikasi SiGv4 untuk driver Go open-source untuk mengakses Amazon Keyspaces (untuk Apache Cassandra).

Jika Anda belum melakukannya, buat kredensial untuk prinsipal IAM Anda mengikuti langkah-langkah di [the section called “Buat kredensi IAM untuk otentikasi AWS”](#). Jika aplikasi berjalan di Lambda atau instans Amazon EC2, aplikasi Anda secara otomatis menggunakan kredensial instans.

Untuk menjalankan tutorial ini secara lokal, Anda dapat menyimpan kredensialnya sebagai variabel lingkungan lokal.

[Tambahkan plugin otentikasi Go SigV4 ke aplikasi Anda dari repositori. GitHub](#) Plugin ini mendukung versi 1.2.x dari driver Go open-source untuk Cassandra dan bergantung pada SDK for Go. AWS

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

Dalam contoh kode ini, titik akhir Amazon Keyspaces diwakili oleh kelas. Cluster Ini menggunakan properti `AwsAuthenticator` for authenticator cluster untuk mendapatkan kredensial.

```
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)
}
}
```

Catatan penggunaan:

1. Ganti "*path_to_file/keyspaces-bundle.pem*" dengan jalur ke file sertifikat yang disimpan di langkah pertama.
2. Untuk contoh ini berjalan secara lokal, Anda perlu mendefinisikan variabel berikut sebagai variabel lingkungan:
 - AWS_ACCESS_KEY_ID
 - AWS_SECRET_ACCESS_KEY
 - AWS_DEFAULT_REGION
3. Untuk menyimpan kunci akses di luar kode, lihat praktik terbaik di [the section called “Kelola kunci akses”](#).
4. Untuk daftar titik akhir yang tersedia, lihat [the section called “Titik akhir layanan”](#).

Menggunakan driver klien Cassandra Perl untuk mengakses Amazon Keyspaces secara terprogram

Bagian ini menunjukkan cara menghubungkan ke Amazon Keyspaces dengan menggunakan driver klien Perl. Untuk contoh kode ini, kami menggunakan Perl 5. Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien.

Important

Amazon Keyspaces certificate sedang bertransisi ke hirarki Amazon Trust Services (ATS). Pastikan lingkungan Anda mempercayai Amazon Root CaS 1-4 untuk menghindari kesalahan koneksi selama rotasi ini. Driver Perl tidak memvalidasi sertifikat SSL Amazon

server, yang berarti Anda tidak dapat mengonfirmasi bahwa Anda terhubung ke Amazon Keyspaces. Langkah kedua, untuk mengkonfigurasi driver untuk menggunakan TLS saat menghubungkan ke Amazon Keyspaces masih diperlukan, dan memastikan bahwa data yang ditransfer antara klien dan server dienkripsi.

1. Unduh driver Cassandra DBI dari <https://metacpan.org/pod/DBD::Cassandra> dan instal driver ke lingkungan Perl Anda. Langkah-langkah yang tepat tergantung pada lingkungan. Berikut ini adalah contoh umum.

```
cpanm DBD::Cassandra
```

2. Buat file untuk aplikasi Anda.

```
touch cqlapp.pl
```

3. Tambahkan kode contoh berikut ke file cqlapp.pl.

```
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

Pastikan bahwa *ServiceUserName* dan *ServicePassword* mencocokkan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial khusus layanan dengan mengikuti langkah-langkahnya. [Buat kredensial khusus layanan untuk akses terprogram ke Amazon Keyspaces](#)

 Note

Untuk daftar titik akhir yang tersedia, lihat [the section called “Titik akhir layanan”](#).

4. Jalankan aplikasi.

```
perl cqlapp.pl
```

Konfigurasi akses lintas akun ke Amazon Keyspaces dengan titik akhir VPC

Anda dapat membuat dan menggunakan terpisah Akun AWS untuk mengisolasi sumber daya dan untuk digunakan di lingkungan yang berbeda, misalnya pengembangan dan produksi. Topik ini memandu Anda melalui akses lintas akun untuk Amazon Keyspaces menggunakan titik akhir VPC antarmuka dalam file. Amazon Virtual Private Cloud Untuk informasi selengkapnya tentang konfigurasi akses lintas akun IAM, lihat [Contoh skenario menggunakan akun pengembangan dan produksi terpisah](#) di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang Amazon Keyspaces dan titik akhir VPC pribadi, lihat. [the section called “Menggunakan VPC endpoint antarmuka ”](#)

Topik

- [Konfigurasi akses lintas akun ke Amazon Keyspaces menggunakan titik akhir VPC di VPC bersama](#)
- [Mengonfigurasi akses lintas akun ke Amazon Keyspaces tanpa VPC bersama](#)

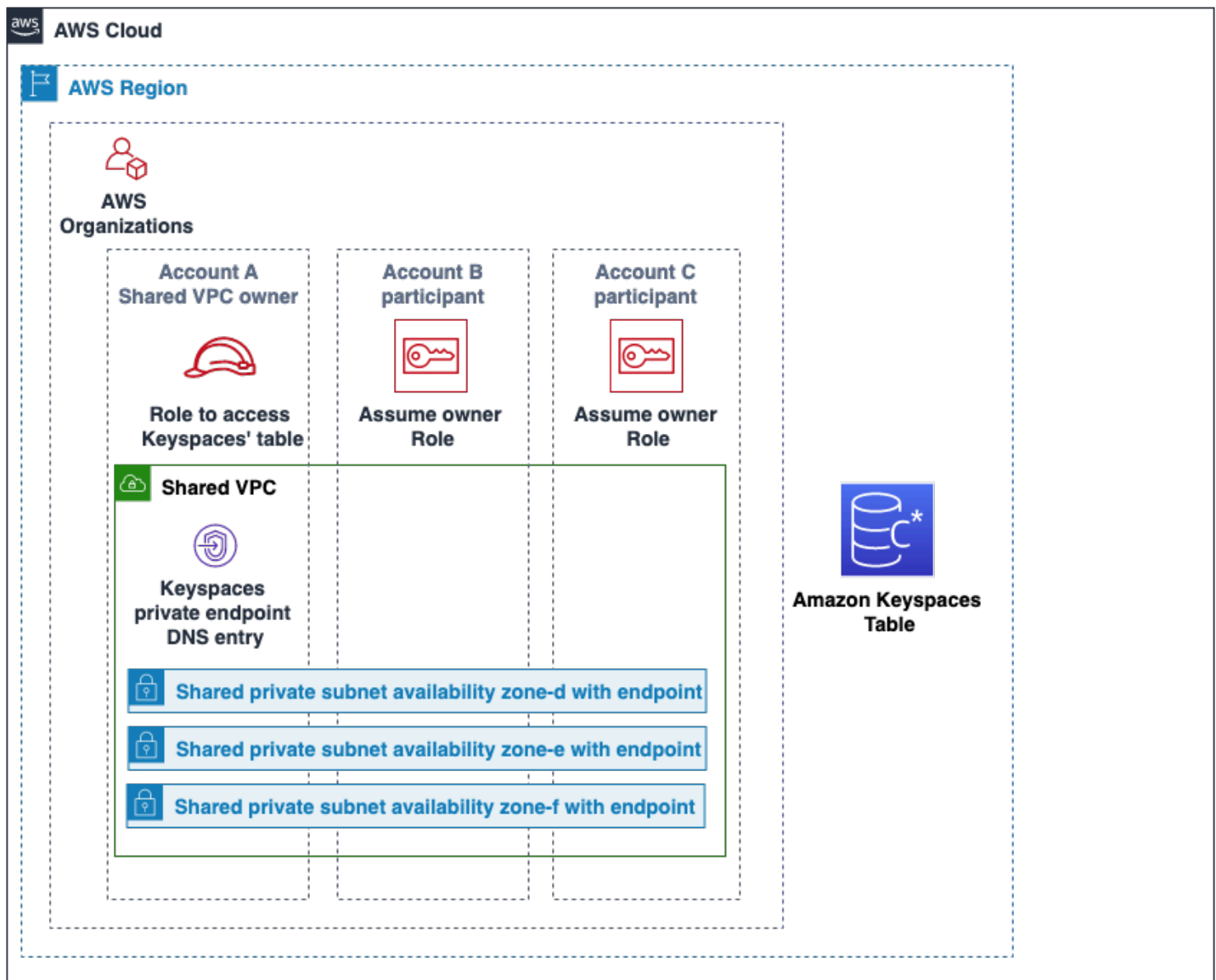
Konfigurasi akses lintas akun ke Amazon Keyspaces menggunakan titik akhir VPC di VPC bersama

Anda dapat membuat sumber daya yang berbeda Akun AWS untuk memisahkan dari aplikasi. Misalnya, Anda dapat membuat satu akun untuk tabel Amazon Keyspaces, akun berbeda untuk aplikasi di lingkungan pengembangan, dan akun lain untuk aplikasi di lingkungan produksi. Topik ini memandu Anda melalui langkah-langkah konfigurasi yang diperlukan untuk menyiapkan akses lintas akun untuk Amazon Keyspaces menggunakan titik akhir VPC antarmuka dalam VPC bersama.

Untuk langkah-langkah mendetail cara mengonfigurasi titik akhir VPC untuk Amazon Keyspaces, lihat [the section called “Langkah 3: Buat titik akhir VPC untuk Amazon Keyspaces”](#)

Dalam contoh ini kami menggunakan tiga akun berikut dalam VPC bersama:

- Account A:111111111111— Akun ini berisi infrastruktur, termasuk titik akhir VPC, subnet VPC, dan tabel Amazon Keyspaces.
- Account B:222222222222— Akun ini berisi aplikasi di lingkungan pengembangan yang perlu terhubung ke tabel Amazon Keyspaces di Account A:111111111111
- Account C:333333333333— Akun ini berisi aplikasi di lingkungan produksi yang perlu terhubung ke tabel Amazon Keyspaces di Account A:111111111111



Account A:111111111111 adalah akun yang berisi sumber daya (tabel Amazon Keyspaces) yang Account B:222222222222 dan Account C:333333333333 perlu diakses, begitu juga akun Account A:111111111111 yang dipercaya. Account B:222222222222 dan Account C:333333333333 merupakan akun dengan prinsipal yang memerlukan akses ke sumber daya (tabel Amazon Keyspaces) di Account A:111111111111, jadi Account B:222222222222 dan Account C:333333333333 merupakan akun tepercaya. Akun tepercaya memberikan izin ke akun tepercaya dengan membagikan peran IAM. Prosedur berikut menguraikan langkah-langkah konfigurasi yang diperlukan dalam Account A:111111111111.

Konfigurasi untuk Account A:111111111111

1. Gunakan AWS Resource Access Manager untuk membuat berbagi sumber daya untuk subnet dan berbagi subnet pribadi dengan Account B:222222222222 dan Account C:333333333333

Account B:222222222222 dan sekarang Account C:333333333333 dapat melihat dan membuat sumber daya di subnet yang telah dibagikan dengan mereka.

2. Buat titik akhir VPC pribadi Amazon Keyspaces yang didukung oleh AWS PrivateLink. Ini membuat beberapa titik akhir di seluruh subnet bersama dan entri DNS untuk titik akhir layanan Amazon Keyspaces.
3. Buat ruang kunci dan tabel Amazon Keyspaces.
4. Buat peran IAM Account A:111111111111 yang memiliki akses penuh ke tabel Amazon Keyspaces, baca akses ke tabel sistem Amazon Keyspaces, dan mampu mendeskripsikan resource VPC Amazon EC2 seperti yang ditunjukkan pada contoh kebijakan berikut.

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

```
}
```

5. Konfigurasi kebijakan kepercayaan untuk peran IAM Account A:111111111111 sehingga Account B:222222222222 dan Account C:333333333333 dapat mengambil peran sebagai akun tepercaya. Seperti yang ditunjukkan dalam contoh berikut.

```
{
  "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": {}
    }
  ]
}
```

Untuk informasi selengkapnya tentang kebijakan IAM lintas akun, lihat [Cross-account kebijakan](#) di Panduan Pengguna IAM.

Konfigurasi di **Account B:222222222222** dan **Account C:333333333333**

1. Di Account B:222222222222 dan Account C:333333333333, buat peran baru dan lampirkan kebijakan berikut yang memungkinkan prinsipal untuk mengambil peran bersama yang dibuat Account A:111111111111.

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

```
}
]
}
```

Mengizinkan prinsipal untuk mengambil peran bersama diimplementasikan menggunakan AssumeRole API dari AWS Security Token Service (AWS STS). Untuk informasi selengkapnya, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.

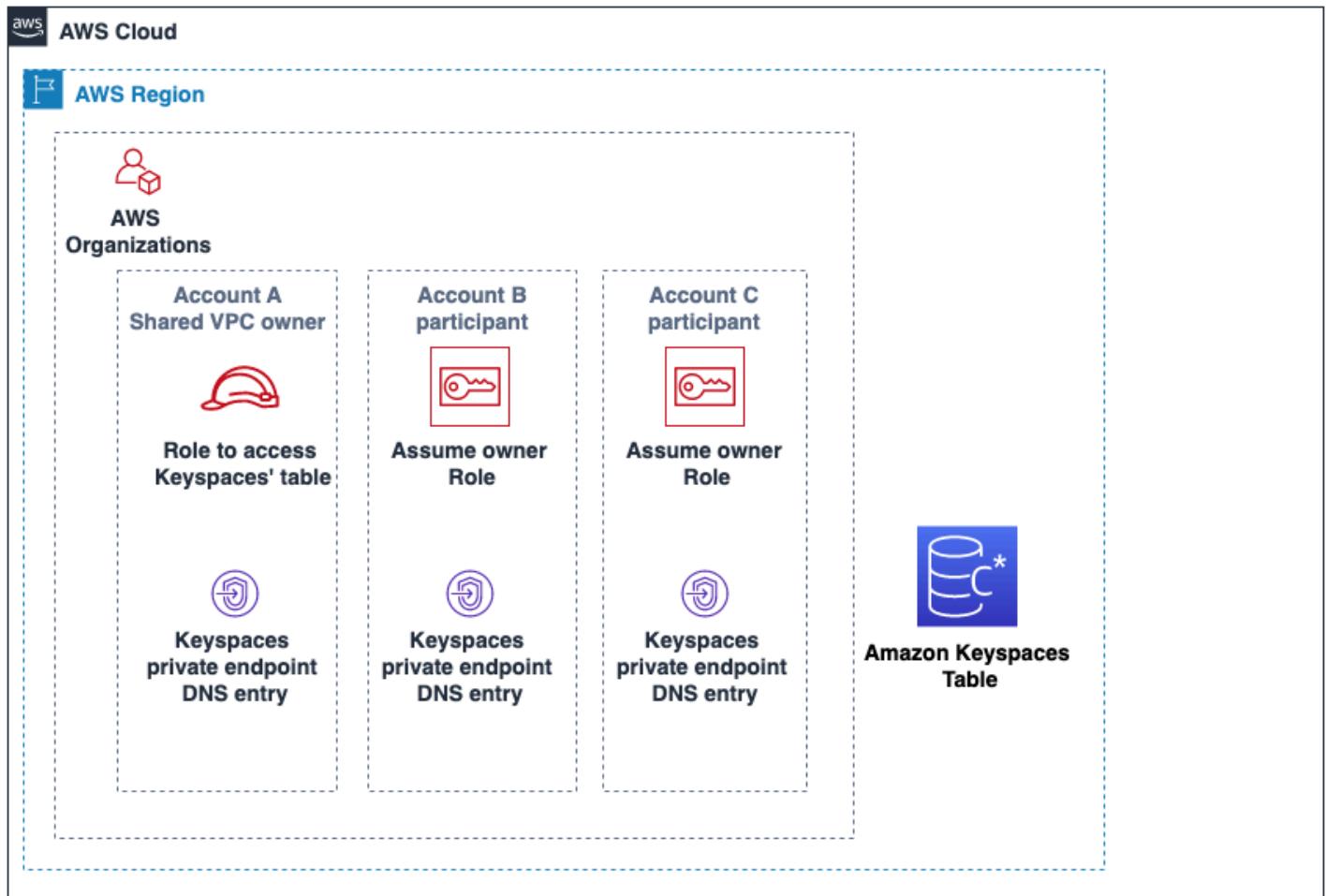
2. Di Account B:222222222222 dan Account C:333333333333, Anda dapat membuat aplikasi yang menggunakan plugin otentikasi SIGV4, yang memungkinkan aplikasi untuk mengambil peran bersama untuk terhubung ke tabel Amazon Keyspaces yang terletak di melalui titik akhir VPC di Account A:111111111111 VPC bersama. Untuk informasi selengkapnya tentang plugin otentikasi SIGV4, lihat [the section called “Buat kredensial akses terprogram”](#) Untuk informasi selengkapnya tentang cara mengonfigurasi aplikasi agar berperan di AWS akun lain, lihat [Otentikasi dan akses di Panduan](#) Referensi AWS SDK dan Alat.

Mengonfigurasi akses lintas akun ke Amazon Keyspaces tanpa VPC bersama

Jika tabel Amazon Keyspaces dan titik akhir VPC pribadi dimiliki oleh akun yang berbeda tetapi tidak berbagi VPC, aplikasi masih dapat menghubungkan lintas akun menggunakan titik akhir VPC. Karena akun tidak berbagi titik akhir VPC, Account A:111111111111 Account B:222222222222, dan Account C:333333333333 memerlukan titik akhir VPC mereka sendiri. Untuk driver klien Cassandra, Amazon Keyspaces muncul seperti satu node, bukan cluster multi-node. Setelah koneksi, driver klien mencapai server DNS yang mengembalikan salah satu titik akhir yang tersedia di VPC akun.

Anda juga dapat mengakses tabel Amazon Keyspaces di berbagai akun tanpa titik akhir VPC bersama dengan menggunakan titik akhir publik atau menerapkan titik akhir VPC pribadi di setiap akun. Saat tidak menggunakan VPC bersama, setiap akun memerlukan titik akhir VPC-nya sendiri. Dalam contoh ini Account A:111111111111, Account B:222222222222, dan Account C:333333333333 memerlukan titik akhir VPC mereka sendiri untuk mengakses tabel di Account A:111111111111 Saat menggunakan titik akhir VPC dalam konfigurasi ini, Amazon Keyspaces muncul sebagai cluster node tunggal ke driver klien Cassandra, bukan cluster multi-node. Setelah koneksi, driver klien mencapai server DNS yang mengembalikan salah satu titik akhir yang tersedia di VPC akun. Tetapi driver klien tidak dapat mengakses `system.peers` tabel untuk menemukan

titik akhir tambahan. Karena ada lebih sedikit host yang tersedia, pengemudi membuat lebih sedikit koneksi. Untuk menyesuaikan ini, tingkatkan pengaturan kumpulan koneksi driver dengan faktor tiga.



Account A:111111111111 adalah akun yang berisi sumber daya (tabel Amazon Keyspaces) yang Account B:222222222222 dan Account C:333333333333 perlu diakses, begitu juga akun Account A:111111111111 yang dipercaya. Account B:222222222222 dan Account C:333333333333 merupakan akun dengan prinsipal yang memerlukan akses ke sumber daya (tabel Amazon Keyspaces) di Account A:111111111111, jadi Account B:222222222222 dan Account C:333333333333 merupakan akun tepercaya. Akun tepercaya memberikan izin ke akun tepercaya dengan membagikan peran IAM. Prosedur berikut menguraikan langkah-langkah konfigurasi yang diperlukan dalam Account A:111111111111.

Konfigurasi untuk **Account A:111111111111**

1. Buat keyspace Amazon Keyspaces dan tabel di Account A:111111111111

2. Buat peran IAM Account A:111111111111 yang memiliki akses penuh ke tabel Amazon Keyspaces dan baca akses ke tabel sistem 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. Konfigurasi kebijakan kepercayaan untuk peran IAM Account A:111111111111 sehingga prinsipal masuk Account B:222222222222 dan Account C:333333333333 dapat menganggap peran tersebut sebagai akun tepercaya. Seperti yang ditunjukkan dalam contoh berikut.

```
{
  "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": {}
    }
  ]
}
```

Untuk informasi selengkapnya tentang kebijakan IAM lintas akun, lihat [Cross-account kebijakan](#) di Panduan Pengguna IAM.

4. Konfigurasi titik akhir VPC Account A:111111111111 dan lampirkan izin ke titik akhir yang memungkinkan peran dari Account B:222222222222 dan Account C:333333333333 untuk mengambil peran dalam menggunakan Account A titik akhir VPC. Izin ini berlaku untuk titik akhir VPC tempat mereka dilampirkan. Untuk informasi selengkapnya tentang kebijakan titik akhir VPC, lihat [the section called “Mengontrol akses ke titik akhir VPC antarmuka untuk 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"
          ]
        }
      }
    }
  ]
}
```

Konfigurasi di **Account B:222222222222** dan **Account C:333333333333**

1. Di Account B:222222222222 dan Account C:333333333333, buat peran baru dan lampirkan kebijakan berikut yang memungkinkan prinsipal untuk mengambil peran bersama yang dibuat Account A:111111111111.

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

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

Mengizinkan prinsipal untuk mengambil peran bersama diimplementasikan menggunakan AssumeRole API dari AWS Security Token Service (AWS STS). Untuk informasi selengkapnya, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.

2. Di dalam Account B:222222222222 dan Account C:333333333333, Anda dapat membuat aplikasi yang menggunakan plugin otentikasi SIGV4, yang memungkinkan aplikasi untuk mengambil peran bersama untuk terhubung ke tabel Amazon Keyspaces yang terletak di Account A:111111111111 Untuk informasi selengkapnya tentang plugin otentikasi SIGV4, lihat [the section called "Buat kredensial akses terprogram"](#) Untuk informasi selengkapnya tentang cara mengonfigurasi aplikasi agar berperan di AWS akun lain, lihat [Otentikasi dan akses di Panduan](#) Referensi AWS SDK dan Alat.

Memulai dengan Amazon Keyspaces (untuk Apache Cassandra)

Jika Anda baru mengenal Apache Cassandra dan Amazon Keyspaces, tutorial ini memandu Anda untuk menginstal program dan alat yang diperlukan untuk menggunakan Amazon Keyspaces dengan sukses. Anda akan belajar cara membuat keyspace dan tabel menggunakan Cassandra Query Language (CQL), the Konsol Manajemen AWS, atau the (). AWS Command Line Interface AWS CLI Anda kemudian menggunakan Cassandra Query Language (CQL) untuk melakukan operasi membuat, membaca, memperbarui, dan menghapus (CRUD) pada data di tabel Amazon Keyspaces Anda.

Tutorial ini mencakup langkah-langkah berikut.

- Prasyarat - Sebelum memulai tutorial, ikuti petunjuk AWS penyiapan untuk mendaftar AWS dan membuat pengguna IAM dengan akses ke Amazon Keyspaces. Kemudian Anda mengatur `cqlsh-expansion` dan AWS CloudShell. Atau Anda dapat menggunakan AWS CLI untuk membuat sumber daya di Amazon Keyspaces.
- Langkah 1: Buat keyspace dan table - Di bagian ini, Anda akan membuat keyspace bernama “catalog” dan tabel bernama “book_awards” di dalamnya. Anda akan menentukan kolom tabel, tipe data, kunci partisi, dan kolom pengelompokan menggunakan Konsol Manajemen AWS, CQL, atau kolom. AWS CLI
- Langkah 2: Lakukan operasi CRUD - Di sini, Anda akan menggunakan `cqlsh-expansion` in CloudShell untuk menyisipkan, membaca, memperbarui, dan menghapus data di tabel “book_awards”. Anda akan belajar cara menggunakan berbagai pernyataan CQL seperti SELECT, INSERT, UPDATE, dan DELETE, dan berlatih memfilter dan memodifikasi data.
- Langkah 3: Bersihkan sumber daya — Untuk menghindari biaya untuk sumber daya yang tidak digunakan, bagian ini memandu Anda menghapus tabel “book_awards” dan keyspace “katalog” menggunakan konsol, CQL, atau. AWS CLI

Untuk tutorial untuk terhubung secara terprogram ke Amazon Keyspaces menggunakan driver klien Apache Cassandra yang berbeda, lihat [the section called “Menggunakan driver klien Cassandra”](#). Untuk contoh kode yang menggunakan AWS SDK berbeda, lihat [Contoh kode untuk Amazon Keyspaces AWS](#) menggunakan SDK.

Topik

- [Prasyarat dan pertimbangan tutorial](#)
- [Buat ruang kunci di Amazon Keyspaces](#)
- [Periksa status pembuatan keyspace di Amazon Keyspaces](#)
- [Buat tabel di Amazon Keyspaces](#)
- [Periksa status pembuatan tabel di Amazon Keyspaces](#)
- [Membuat, membaca, memperbarui, dan menghapus data \(CRUD\) menggunakan CQL di Amazon Keyspaces](#)
- [Hapus tabel di Amazon Keyspaces](#)
- [Hapus ruang kunci di Amazon Keyspaces](#)

Prasyarat dan pertimbangan tutorial

Sebelum Anda dapat memulai dengan Amazon Keyspaces, ikuti petunjuk AWS penyiapan di. [Mengakses Amazon Keyspaces \(untuk Apache Cassandra\)](#) Langkah-langkah ini termasuk mendaftar AWS dan membuat pengguna AWS Identity and Access Management (IAM) dengan akses ke Amazon Keyspaces.

Untuk menyelesaikan semua langkah tutorial, Anda perlu menginstal `cqlsh`. Anda dapat mengikuti instruksi pengaturan di [Menggunakan `cqlsh` untuk terhubung ke Amazon Keyspaces](#).

Untuk mengakses Amazon Keyspaces menggunakan `cqlsh` atau AWS CLI, sebaiknya gunakan AWS CloudShell. CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. Konsol Manajemen AWS Anda dapat menjalankan perintah AWS Command Line Interface (AWS CLI) terhadap Amazon Keyspaces menggunakan shell pilihan Anda (Bash, PowerShell atau Z shell). Untuk menggunakannya `cqlsh`, Anda harus menginstal `cqlsh-expansion`. Untuk petunjuk `cqlsh-expansion` penginstalan, lihat [the section called “Menggunakan `cqlsh-expansion`”](#). Untuk informasi lebih lanjut tentang CloudShell lihat [the section called “Menggunakan AWS CloudShell”](#).

Untuk menggunakan AWS CLI cara membuat, melihat, dan menghapus sumber daya di Amazon Keyspaces, ikuti petunjuk penyiapan di. [the section called “Mengunduh dan Mengkonfigurasi AWS CLI”](#)

Setelah menyelesaikan langkah-langkah prasyarat, lanjutkan ke. [Buat ruang kunci di Amazon Keyspaces](#)

Buat ruang kunci di Amazon Keyspaces

Di bagian ini, Anda membuat ruang kunci menggunakan konsol, `cqlsh`, atau AWS CLI

Note

Sebelum Anda mulai, pastikan bahwa Anda telah mengkonfigurasi semua [prasyarat tutorial](#).

Sebuah keyspace mengelompokkan tabel terkait yang relevan untuk satu atau beberapa aplikasi. Sebuah keyspace berisi satu atau lebih tabel dan mendefinisikan strategi replikasi untuk semua tabel yang dikandungnya. Untuk informasi selengkapnya tentang ruang kunci, lihat topik berikut:

- Pernyataan bahasa definisi data (DDL) dalam referensi bahasa CQL: [Keyspaces](#)
- [Kuota untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#)

Dalam tutorial ini kita membuat ruang kunci Single-region, dan strategi replikasi keyspace adalah `SingleRegionStrategy` Menggunakan `SingleRegionStrategy`, Amazon Keyspaces mereplikasi data di tiga [Availability Zone](#) menjadi satu. Wilayah AWS Untuk mempelajari cara membuat ruang kunci Multi-wilayah, lihat [the section called “Buat ruang kunci Multi-wilayah”](#)

Menggunakan konsol

Untuk membuat ruang kunci menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces.
3. Pilih Buat ruang kunci.
4. Di kotak nama Keyspace, masukkan **catalog** sebagai nama untuk ruang kunci Anda.

Kendala nama:

- Nama tidak bisa kosong.
- Karakter yang diizinkan: karakter alfanumerik dan garis bawah (`_`).
- Panjang maksimum adalah 48 karakter.

5. Di bawah Wilayah AWS, konfirmasikan bahwa Single-Region replikasi adalah strategi replikasi untuk keyspace.
6. Untuk membuat keyspace, pilih Create keyspace.
7. Verifikasi bahwa ruang kunci catalog dibuat dengan melakukan hal berikut:
 - a. Di panel navigasi, pilih Keyspaces.
 - b. Temukan ruang kunci Anda catalog dalam daftar ruang kunci.

Menggunakan CQL

Prosedur berikut membuat keyspace menggunakan CQL.

Untuk membuat keyspace menggunakan CQL

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui *us-east-1* dengan Wilayah Anda sendiri.

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

Output dari perintah itu akan terlihat seperti ini.

```
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. Buat keyspace Anda menggunakan perintah CQL berikut.

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

SingleRegionStrategy menggunakan faktor replikasi tiga dan mereplikasi data di tiga AWS Availability Zone di Wilayahnya.

Note

Amazon Keyspaces default semua input ke huruf kecil kecuali Anda melampirkannya dalam tanda kutip.

3. Verifikasi bahwa ruang kunci Anda telah dibuat.

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

Output dari perintah ini akan terlihat mirip dengan ini.

```
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)
```

Menggunakan AWS CLI

Prosedur berikut membuat keyspace menggunakan file. AWS CLI

Untuk membuat keyspace menggunakan AWS CLI

1. Untuk mengonfirmasi bahwa lingkungan Anda sudah diatur, Anda dapat menjalankan perintah berikut di CloudShell.

```
aws keyspaces help
```

2. Buat ruang kunci Anda menggunakan AWS CLI pernyataan berikut.

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

3. Verifikasi bahwa ruang kunci Anda dibuat dengan pernyataan berikut AWS CLI

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

Output dari perintah ini akan terlihat mirip dengan contoh ini.

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

Periksa status pembuatan keyspace di Amazon Keyspaces

Amazon Keyspaces melakukan operasi data definition language (DDL), seperti membuat dan menghapus keyspaces, secara asinkron.

Anda dapat memantau status pembuatan ruang kunci baru di Konsol Manajemen AWS, yang menunjukkan kapan ruang kunci tertunda atau aktif. Anda juga dapat memantau status pembuatan keyspace baru secara terprogram dengan menggunakan `keyspace.system_schema_mcs`. Sebuah keyspace menjadi terlihat dalam `system_schema_mcs` keyspaces tabel ketika sudah siap untuk digunakan.

Pola desain yang disarankan untuk memeriksa kapan ruang kunci baru siap digunakan adalah dengan melakukan polling tabel Amazon `system_schema_mcs` keyspaces Keyspaces (`system_schema_mcs.*`). Untuk daftar pernyataan DDL untuk ruang kunci, lihat [the section called “Keyspaces”](#) bagian dalam referensi bahasa CQL.

Kueri berikut menunjukkan apakah keyspace telah berhasil dibuat.

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

Untuk keyspace yang telah berhasil dibuat, output dari query terlihat seperti berikut.

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

Buat tabel di Amazon Keyspaces

Di bagian ini, Anda membuat tabel menggunakan konsol, `cqlsh`, atau AWS CLI.

Tabel adalah tempat data Anda diatur dan disimpan. Kunci utama tabel Anda menentukan bagaimana data dipartisi dalam tabel Anda. Kunci utama terdiri dari kunci partisi yang diperlukan dan satu atau lebih kolom pengelompokan opsional. Nilai gabungan yang menyusun kunci utama harus unik di semua data tabel. Untuk informasi selengkapnya tentang tabel, lihat topik berikut:

- Desain kunci partisi: [the section called “Desain kunci partisi”](#)
- Bekerja dengan tabel: [the section called “Periksa status pembuatan tabel”](#)
- Pernyataan DDL dalam referensi bahasa CQL: [Tabel](#)
- Manajemen sumber daya tabel: [Mengelola sumber daya tanpa server](#)
- Pemantauan pemanfaatan sumber daya tabel: [the section called “Pemantauan CloudWatch dengan”](#)
- [Kuota untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#)

Saat Anda membuat tabel, Anda menentukan yang berikut:

- Nama tabel.
- Nama dan tipe data dari setiap kolom dalam tabel.
- Kunci utama untuk tabel.
 - Kunci partisi - Diperlukan
 - Kolom pengelompokan - Opsional

Gunakan prosedur berikut untuk membuat tabel dengan kolom tertentu, tipe data, kunci partisi, dan kolom pengelompokan.

Menggunakan konsol

Prosedur berikut membuat tabel `book_awards` dengan kolom dan tipe data ini.

<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

Untuk membuat tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces.
3. Pilih catalog sebagai ruang kunci tempat Anda ingin membuat tabel ini.
4. Pilih Buat tabel.
5. Di kotak Nama tabel, masukkan **book_awards** sebagai nama untuk tabel Anda.

Kendala nama:

- Nama tidak bisa kosong.
 - Karakter yang diizinkan: karakter alfanumerik dan garis bawah (. _)
 - Panjang maksimum adalah 48 karakter.
6. Di bagian Kolom, ulangi langkah-langkah berikut untuk setiap kolom yang ingin Anda tambahkan ke tabel ini.

Tambahkan kolom dan tipe data berikut.

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

- a. Nama — Masukkan nama untuk kolom.

Kendala nama:

- Nama tidak bisa kosong.
- Karakter yang diizinkan: karakter alfanumerik dan garis bawah (. _)
- Panjang maksimum adalah 48 karakter.

- b. Jenis - Dalam daftar tipe data, pilih tipe data untuk kolom ini.

- c. Untuk menambahkan kolom lain, pilih Tambahkan kolom.
7. Pilih `award` dan `year` sebagai tombol partisi di bawah Partition Key. Kunci partisi diperlukan untuk setiap tabel. Kunci partisi dapat dibuat dari satu atau lebih kolom.
8. Tambahkan `category` dan `rank` sebagai kolom Clustering. Kolom pengelompokan bersifat opsional dan menentukan urutan pengurutan dalam setiap partisi.
 - a. Untuk menambahkan kolom pengelompokan, pilih Tambahkan kolom pengelompokan.
 - b. Dalam daftar Kolom, pilih kategori. Dalam daftar Urutan, pilih ASC untuk mengurutkan dalam urutan menaik pada nilai di kolom ini. (Pilih DESC untuk urutan menurun.)
 - c. Kemudian pilih Tambahkan kolom pengelompokan dan pilih peringkat.
9. Di bagian Pengaturan tabel, pilih Pengaturan default.
10. Pilih Buat tabel.
11. Verifikasi bahwa tabel Anda telah dibuat.
 - a. Di panel navigasi, pilih Tabel.
 - b. Konfirmasikan bahwa tabel Anda ada dalam daftar tabel.
 - c. Pilih nama meja Anda.
 - d. Konfirmasikan bahwa semua kolom dan tipe data Anda sudah benar.

 Note

Kolom mungkin tidak tercantum dalam urutan yang sama dengan yang Anda tambahkan ke tabel.

Menggunakan CQL

Prosedur ini membuat tabel dengan kolom dan tipe data berikut menggunakan CQL. `award` dan `year` adalah kunci partisi dengan `category` dan `rank` sebagai kolom pengelompokan, bersama-sama mereka membentuk kunci utama tabel.

<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

Untuk membuat tabel menggunakan CQL

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui *us-east-1* dengan Wilayah Anda sendiri.

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

Output dari perintah itu akan terlihat seperti ini.

```
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. Pada prompt keyspace (`cqlsh:keyspace_name>`), buat tabel Anda dengan memasukkan kode berikut ke jendela perintah Anda.

```
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 adalah urutan pengelompokan default. Anda juga dapat menentukan DESC urutan menurun.

Perhatikan bahwa `year` dan `award` adalah kolom kunci partisi. Kemudian, `category` dan `rank` adalah kolom pengelompokan yang diurutkan berdasarkan urutan naik ()ASC. Bersama-sama, kolom-kolom ini membentuk kunci utama tabel.

3. Verifikasi bahwa tabel Anda telah dibuat.

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

Outputnya akan terlihat mirip dengan ini.

```
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. Verifikasi struktur tabel Anda.

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

Output dari pernyataan ini akan terlihat mirip dengan contoh ini.

```
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 |
0x63617465676f7229 | clustering | 0 | text
catalog | book_awards | rank | asc |
0x72616e6b | clustering | 1 | int
catalog | book_awards | author | none |
0x617574686f72 | regular | -1 | text
catalog | book_awards | book_title | none |
0x6266666b5f7469746c65 | regular | -1 | text
```

```

catalog | book_awards | publisher | none |
0x7075626c6973686572 | regular | -1 | text

```

(7 rows)

Konfirmasikan bahwa semua kolom dan tipe data seperti yang Anda harapkan. Urutan kolom mungkin berbeda dari pada CREATE pernyataan.

Menggunakan AWS CLI

Prosedur ini membuat tabel dengan kolom dan tipe data berikut menggunakan AWS CLI.

awardKolom year dan membentuk kunci partisi dengan category dan rank sebagai kolom pengelompokan.

```

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

```

Untuk membuat tabel menggunakan AWS CLI

Perintah berikut membuat tabel dengan nama book_awards. Kunci partisi tabel terdiri dari kolom year dan award dan kunci pengelompokan terdiri dari kolom category dan rank, kedua kolom pengelompokan menggunakan urutan urutan menaik. (Agar mudah dibaca, perintah schema-definition tabel buat di bagian ini dipecah menjadi baris terpisah.)

1. Anda dapat membuat tabel menggunakan pernyataan berikut.

```

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}] '

```

Perintah ini menghasilkan output berikut.

```
{
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards"
}
```

2. Untuk mengonfirmasi metadata dan properti tabel, Anda dapat menggunakan perintah berikut.

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

Perintah ini mengembalikan output berikut.

```
{
  "keyspaceName": "catalog",
  "tableName": "book_awards",
  "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards",
  "creationTimestamp": "2024-07-11T15:12:55.571000+00:00",
  "status": "ACTIVE",
  "schemaDefinition": {
    "allColumns": [
      {
        "name": "year",
        "type": "int"
      },
      {
        "name": "award",
        "type": "text"
      },
      {
        "name": "category",
        "type": "text"
      },
      {
        "name": "rank",
        "type": "int"
      },
      {
        "name": "author",
        "type": "text"
      }
    ]
  }
}
```

```

        {
            "name": "book_title",
            "type": "text"
        },
        {
            "name": "publisher",
            "type": "text"
        }
    ],
    "partitionKeys": [
        {
            "name": "year"
        },
        {
            "name": "award"
        }
    ],
    "clusteringKeys": [
        {
            "name": "category",
            "orderBy": "ASC"
        },
        {
            "name": "rank",
            "orderBy": "ASC"
        }
    ],
    "staticColumns": [],
},
"capacitySpecification": {
    "throughputMode": "PAY_PER_REQUEST",
    "lastUpdateToPayPerRequestTimestamp": "2024-07-11T15:12:55.571000+00:00"
},
"encryptionSpecification": {
    "type": "AWS_OWNED_KMS_KEY"
},
"pointInTimeRecovery": {
    "status": "DISABLED"
},
"defaultTimeToLive": 0,
"comment": {
    "message": ""
},
"replicaSpecifications": []

```

}

Untuk melakukan operasi CRUD (membuat, membaca, memperbarui, dan menghapus) pada data dalam tabel Anda, lanjutkan ke [the section called “Operasi CRUD”](#).

Periksa status pembuatan tabel di Amazon Keyspaces

Amazon Keyspaces melakukan operasi data definition language (DDL), seperti membuat dan menghapus tabel, secara asinkron. Anda dapat memantau status pembuatan tabel baru di Konsol Manajemen AWS, yang menunjukkan kapan tabel tertunda atau aktif. Anda juga dapat memantau status pembuatan tabel baru secara terprogram dengan menggunakan tabel skema sistem.

Sebuah tabel menunjukkan sebagai aktif dalam skema sistem ketika siap untuk digunakan. Pola desain yang disarankan untuk memeriksa kapan tabel baru siap digunakan adalah dengan melakukan polling tabel skema sistem Amazon Keyspaces (`system_schema_mcs`). * Untuk daftar pernyataan DDL untuk tabel, lihat [the section called “Tabel”](#) bagian dalam referensi bahasa CQL.

Query berikut menunjukkan status tabel.

```
SELECT keyspace_name, table_name, status FROM system_schema_mcs.tables WHERE
keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Untuk tabel yang masih dibuat dan tertunda, output kueri terlihat seperti ini.

```
keyspace_name | table_name | status
-----+-----+-----
mykeyspace   | mytable   | CREATING
```

Untuk tabel yang telah berhasil dibuat dan aktif, output dari query terlihat seperti berikut.

```
keyspace_name | table_name | status
-----+-----+-----
mykeyspace   | mytable   | ACTIVE
```

Membuat, membaca, memperbarui, dan menghapus data (CRUD) menggunakan CQL di Amazon Keyspaces

Pada langkah tutorial ini, Anda akan belajar cara menyisipkan, membaca, memperbarui, dan menghapus data dalam tabel Amazon Keyspaces menggunakan pernyataan CQL data manipulation language (DHTML). Di Amazon Keyspaces, Anda hanya dapat membuat pernyataan DHTML dalam bahasa CQL. Dalam tutorial ini, Anda akan berlatih menjalankan pernyataan DHTML menggunakan `cqlsh-expansion` with [AWS CloudShell](#) di Konsol Manajemen AWS

- Menyisipkan data - Bagian ini mencakup penyisipan catatan tunggal dan beberapa ke dalam tabel menggunakan pernyataan. `INSERT` Anda akan mempelajari cara mengunggah data dari file CSV dan memverifikasi sisipan yang berhasil menggunakan `SELECT` kueri.
- Membaca data — Di sini, Anda akan menjelajahi berbagai variasi `SELECT` pernyataan untuk mengambil data dari tabel. Topik termasuk memilih semua data, memilih kolom tertentu, memfilter baris berdasarkan kondisi menggunakan `WHERE` klausa, dan memahami kondisi sederhana dan gabungan.
- Memperbarui data — Di bagian ini, Anda akan mempelajari cara memodifikasi data yang ada dalam tabel menggunakan `UPDATE` pernyataan. Anda akan berlatih memperbarui kolom tunggal dan beberapa sambil memahami batasan seputar memperbarui kolom kunci utama.
- Menghapus data — Bagian terakhir mencakup penghapusan data dari tabel menggunakan pernyataan. `DELETE` Anda akan belajar cara menghapus sel tertentu, seluruh baris, dan implikasi menghapus data versus menghapus seluruh tabel atau ruang kunci.

Sepanjang tutorial, Anda akan menemukan contoh, tips, dan kesempatan untuk berlatih menulis kueri CQL Anda sendiri untuk berbagai skenario.

Topik

- [Memasukkan dan memuat data ke dalam tabel Amazon Keyspaces](#)
- [Membaca data dari tabel menggunakan `SELECT` pernyataan CQL di Amazon Keyspaces](#)
- [Perbarui data dalam tabel Amazon Keyspaces menggunakan CQL](#)
- [Hapus data dari tabel menggunakan pernyataan `DELETE` CQL](#)

Memasukkan dan memuat data ke dalam tabel Amazon Keyspaces

Untuk membuat data dalam `book_awards` tabel Anda, gunakan `INSERT` pernyataan untuk menambahkan satu baris.

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui `us-east-1` dengan Wilayah Anda sendiri.

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

Output dari perintah itu akan terlihat seperti ini.

```
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. Sebelum Anda dapat menulis data ke tabel Amazon Keyspaces menggunakan `cqlsh`, Anda harus mengatur konsistensi tulis untuk sesi `cqlsh` saat ini. `LOCAL_QUORUM` Untuk informasi selengkapnya tentang tingkat konsistensi yang didukung, lihat [the section called “Tulis tingkat konsistensi”](#). Perhatikan bahwa langkah ini tidak diperlukan jika Anda menggunakan editor CQL di Konsol Manajemen AWS

```
CONSISTENCY LOCAL_QUORUM;
```

3. Untuk menyisipkan satu catatan, jalankan perintah berikut di editor CQL.

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

4. Verifikasi bahwa data telah ditambahkan dengan benar ke tabel Anda dengan menjalankan perintah berikut.

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

Output dari pernyataan akan terlihat seperti ini.

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



```
2023 | Wolf | Fiction | 3 | Shirley Rodriguez | Mountain | AnyPublisher

(1 rows)
```

Untuk menyisipkan beberapa catatan dari file menggunakan `cqlsh`

1. Unduh contoh file CSV (`keyspaces_sample_table.csv`) yang terkandung dalam file arsip [samplemigration.zip](#). Buka zip arsip dan catat jalur ke `keyspaces_sample_table.csv`.

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

2. Buka AWS CloudShell di Konsol Manajemen AWS dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui `us-east-1` dengan Wilayah Anda sendiri.

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

3. Pada `cqlsh` prompt (`cqlsh>`), tentukan ruang kunci.

```
USE catalog ;
```

4. Tetapkan konsistensi tulis ke `LOCAL_QUORUM`. Untuk informasi selengkapnya tentang tingkat konsistensi yang didukung, lihat [the section called "Tulis tingkat konsistensi"](#).

```
CONSISTENCY LOCAL_QUORUM;
```

5. Dalam AWS CloudShell pilih Tindakan di sisi kanan atas layar dan kemudian pilih Unggah file untuk mengunggah file csv yang diunduh sebelumnya. Catat jalur ke file.
6. Pada prompt keyspace (`cqlsh:catalog>`), jalankan pernyataan berikut.

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

Output dari pernyataan harus terlihat mirip dengan ini.

```
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. Verifikasi bahwa data telah ditambahkan dengan benar ke tabel Anda dengan menjalankan kueri berikut.

```
SELECT * FROM book_awards ;
```

Anda akan melihat output berikut.

year	award publisher	category	rank	author	book_title
2020	Wolf Example Books	Non-Fiction	1	Wang Xiulan	History of Ideas
2020	Wolf SomePublisher	Non-Fiction	2	Ana Carolina Silva	Science Today
2020	Wolf AnyPublisher	Non-Fiction	3	Shirley Rodriguez	The Future of Sea Ice
2020	Kwesi Manu Prize SomePublisher	Fiction	1	Akua Mansa	Where did you go?

```

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)

Untuk mempelajari lebih lanjut tentang cara `cqlsh COPY` mengunggah data dari file csv ke tabel Amazon Keyspaces, lihat. [the section called “Memuat data menggunakan cqlsh”](#)

Membaca data dari tabel menggunakan **SELECT** pernyataan CQL di Amazon Keyspaces

Di [Memasukkan dan memuat data ke dalam tabel Amazon Keyspaces](#) bagian ini, Anda menggunakan **SELECT** pernyataan untuk memverifikasi bahwa Anda telah berhasil menambahkan data ke tabel Anda. Di bagian ini, Anda menyempurnakan penggunaan **SELECT** untuk menampilkan kolom tertentu, dan hanya baris yang memenuhi kriteria tertentu.

Bentuk umum **SELECT** pernyataan tersebut adalah sebagai berikut.

```
SELECT column_list FROM table_name [WHERE condition [ALLOW FILTERING]] ;
```

Topik

- [Pilih semua data di tabel Anda](#)
- [Pilih subset kolom](#)
- [Pilih subset baris](#)

Pilih semua data di tabel Anda

Bentuk paling sederhana dari **SELECT** pernyataan mengembalikan semua data dalam tabel Anda.

⚠ Important

Dalam lingkungan produksi, biasanya bukan praktik terbaik untuk menjalankan perintah ini, karena ia mengembalikan semua data dalam tabel Anda.

Untuk memilih semua data tabel

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui `us-east-1` dengan Wilayah Anda sendiri.

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

2. Jalankan query berikut.

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

Menggunakan karakter wild-card (*) untuk `column_list` memilih semua kolom. Output dari pernyataan terlihat seperti contoh berikut.

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

2020	Richard Roe	Fiction	3	Mateo Jackson	Inside
the Whale	AnyPublisher				

Pilih subset kolom

Untuk menanyakan subset kolom

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui *us-east-1* dengan Wilayah Anda sendiri.

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

2. Untuk mengambil hanya *award*, *category*, dan *year* kolom, jalankan query berikut.

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

Output hanya berisi kolom yang ditentukan dalam urutan yang tercantum dalam SELECT pernyataan.

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

Pilih subset baris

Saat menanyakan kumpulan data besar, Anda mungkin hanya menginginkan catatan yang memenuhi kriteria tertentu. Untuk melakukan ini, Anda dapat menambahkan WHERE klausa ke akhir pernyataan kami `SELECT`.

Untuk menanyakan subset baris

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui *us-east-1* dengan Wilayah Anda sendiri.

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

2. Untuk mengambil hanya catatan untuk penghargaan tahun tertentu, jalankan kueri berikut.

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

SELECT Pernyataan sebelumnya mengembalikan output berikut.

```

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

```

Memahami **WHERE** klausa

WHERE Klausul ini digunakan untuk memfilter data dan hanya mengembalikan data yang memenuhi kriteria yang ditentukan. Kriteria yang ditentukan dapat berupa kondisi sederhana atau kondisi majemuk.

Cara menggunakan kondisi dalam **WHERE** klausa

- Kondisi sederhana - Satu kolom.

```
WHERE column_name=value
```

Anda dapat menggunakan kondisi sederhana dalam WHERE klausa jika salah satu dari kondisi berikut terpenuhi:

- Kolom adalah satu-satunya kolom kunci partisi dari tabel.

- Anda menambahkan `ALLOW FILTERING` setelah kondisi dalam `WHERE` klausa.

Ketahui bahwa penggunaan `ALLOW FILTERING` dapat menghasilkan kinerja yang tidak konsisten, terutama dengan tabel besar dan multi-partisi.

- Kondisi majemuk — Beberapa kondisi sederhana yang dihubungkan oleh `AND`.

```
WHERE column_name1=value1 AND column_name2=value2 AND column_name3=value3...
```

Anda dapat menggunakan kondisi majemuk dalam `WHERE` klausa jika salah satu dari kondisi berikut terpenuhi:

- Kolom yang dapat Anda gunakan dalam `WHERE` klausa harus menyertakan semua atau subset kolom dalam kunci partisi tabel. Jika Anda ingin menggunakan hanya subset kolom dalam `WHERE` klausa, Anda harus menyertakan satu set kolom kunci partisi yang berdekatan dari kiri ke kanan, dimulai dengan kolom utama kunci partisi. Misalnya, jika kolom kunci partisi adalah `year`, `month`, dan `award` kemudian Anda dapat menggunakan kolom berikut dalam `WHERE` klausa:
 - `year`
 - `year` `DAN` `month`
 - `year` `month` `DAN` `award`
- Anda menambahkan `ALLOW FILTERING` setelah kondisi majemuk dalam `WHERE` klausa, seperti pada contoh berikut.

```
SELECT * FROM my_table WHERE col1=5 AND col2='Bob' ALLOW FILTERING ;
```

Ketahui bahwa penggunaan `ALLOW FILTERING` dapat menghasilkan kinerja yang tidak konsisten, terutama dengan tabel besar dan multi-partisi.

Cobalah

Buat kueri CQL Anda sendiri untuk menemukan yang berikut dari tabel Anda: `book_awards`

- Temukan pemenang penghargaan Wolf 2020 dan tampilkan judul buku dan penulis, diurutkan berdasarkan peringkat.
- Tunjukkan pemenang hadiah pertama untuk semua penghargaan pada tahun 2020 dan tampilkan judul buku dan nama penghargaan.

Perbarui data dalam tabel Amazon Keyspaces menggunakan CQL

Untuk memperbarui data dalam `book_awards` tabel Anda, gunakan `UPDATE` pernyataan.

Bentuk umum `UPDATE` pernyataan tersebut adalah sebagai berikut.

```
UPDATE table_name SET column_name=new_value WHERE primary_key=value ;
```

Tip

- Anda dapat memperbarui beberapa kolom dengan menggunakan daftar `column_names` dan nilai yang dipisahkan koma, seperti pada contoh berikut.

```
UPDATE my_table SET col1='new_value_1', col2='new_value2' WHERE col3='1' ;
```

- Jika kunci primer terdiri dari beberapa kolom, semua kolom kunci primer dan nilainya harus disertakan dalam `WHERE` klausa.
- Anda tidak dapat memperbarui kolom apa pun di kunci utama karena itu akan mengubah kunci utama untuk catatan.

Untuk memperbarui satu sel

Menggunakan `book_awards` meja Anda, ubah nama penerbit untuk pemenang penghargaan Wolf non-fiksi pada tahun 2020.

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

Verifikasi bahwa penerbit sekarang `new Books`.

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

Pernyataan harus mengembalikan output berikut.

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

Cobalah

Lanjutan: Pemenang fiksi 2020 “Kwezi Manu Prize” telah mengubah nama mereka. Perbarui catatan ini untuk mengubah nama menjadi 'Akua Mansa-House'.

Hapus data dari tabel menggunakan pernyataan **DELETE** CQL

Untuk menghapus data dalam `book_awards` tabel Anda, gunakan `DELETE` pernyataan.

Anda dapat menghapus data dari baris atau dari partisi. Hati-hati saat menghapus data, karena penghapusan tidak dapat diubah.

Menghapus satu atau semua baris dari tabel tidak menghapus tabel. Dengan demikian Anda dapat mengisinya kembali dengan data. Menghapus tabel akan menghapus tabel dan semua data di dalamnya. Untuk menggunakan tabel lagi, Anda harus membuatnya kembali dan menambahkan data ke dalamnya. Menghapus keyspace akan menghapus keyspace dan semua tabel di dalamnya. Untuk menggunakan keyspace dan tabel, Anda harus membuatnya kembali, dan kemudian mengisinya dengan data. Anda dapat menggunakan pemulihan Amazon Keyspaces Point-in-time (PITR) untuk membantu memulihkan tabel yang dihapus, untuk mempelajari selengkapnya lihat [the section called “Backup dan restore dengan pemulihan point-in-time”](#) Untuk mempelajari cara mengembalikan tabel yang dihapus dengan PITR diaktifkan, lihat [the section called “Kembalikan tabel yang dihapus”](#).

Hapus sel

Menghapus kolom dari baris menghapus data dari sel yang ditentukan. Saat Anda menampilkan kolom itu menggunakan `SELECT` pernyataan, data ditampilkan sebagai *null*, meskipun nilai null tidak disimpan di lokasi tersebut.

Sintaks umum untuk menghapus satu atau lebih kolom spesifik adalah sebagai berikut.

```
DELETE column_name1[, column_name2...] FROM table_name WHERE condition ;
```

Di `book_awards` tabel Anda, Anda dapat melihat bahwa judul buku yang memenangkan harga pertama dari harga “Richard Roe” 2020 adalah “Long Summer”. Bayangkan judul ini telah ditarik kembali, dan Anda perlu menghapus data dari sel ini.

Untuk menghapus sel tertentu

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui *us-east-1* dengan Wilayah Anda sendiri.

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

2. Jalankan DELETE query berikut.

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

3. Verifikasi bahwa permintaan penghapusan dibuat seperti yang diharapkan.

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

Output dari pernyataan ini terlihat seperti ini.

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

Hapus baris

Mungkin ada saat ketika Anda perlu menghapus seluruh baris, misalnya untuk memenuhi permintaan penghapusan data. Sintaks umum untuk menghapus baris adalah sebagai berikut.

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

Untuk menghapus baris

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui *us-east-1* dengan Wilayah Anda sendiri.

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

2. Jalankan DELETE query berikut.

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

3. Verifikasi bahwa penghapusan dilakukan seperti yang diharapkan.

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

Output dari pernyataan ini terlihat seperti ini setelah baris telah dihapus.

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

Anda dapat menghapus data kedaluwarsa secara otomatis dari tabel menggunakan Amazon Keyspaces Time to Live, untuk informasi selengkapnya, lihat. [the section called “Data kedaluwarsa dengan Time to Live”](#)

Hapus tabel di Amazon Keyspaces

Untuk menghindari biaya untuk tabel dan data yang tidak Anda butuhkan, hapus semua tabel yang tidak Anda gunakan. Saat Anda menghapus tabel, tabel dan datanya akan dihapus dan Anda berhenti mengeluarkan biaya untuk mereka. Namun, ruang kunci tetap ada. Saat Anda menghapus ruang kunci, ruang kunci dan semua tabelnya dihapus dan Anda berhenti menambah biaya untuk mereka.

Anda dapat menghapus tabel menggunakan konsol, CQL, atau file. AWS CLI Saat Anda menghapus tabel, tabel dan semua datanya akan dihapus.

Menggunakan konsol

Prosedur berikut menghapus tabel dan semua datanya menggunakan file. Konsol Manajemen AWS

Untuk menghapus tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel.
3. Pilih kotak di sebelah kiri nama setiap tabel yang ingin Anda hapus.

4. Pilih Hapus.
5. Pada layar Hapus tabel, masukkan **Delete** di dalam kotak. Kemudian, pilih Hapus tabel.
6. Untuk memverifikasi bahwa tabel telah dihapus, pilih Tabel di panel navigasi, dan konfirmasikan bahwa `book_awards` tabel tidak lagi terdaftar.

Menggunakan CQL

Prosedur berikut menghapus tabel dan semua datanya menggunakan CQL.

Untuk menghapus tabel menggunakan CQL

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui `us-east-1` dengan Wilayah Anda sendiri.

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

2. Hapus tabel Anda dengan memasukkan pernyataan berikut.

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

3. Verifikasi bahwa tabel Anda telah dihapus.

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

Outputnya akan terlihat seperti ini. Perhatikan bahwa ini mungkin memakan waktu, jadi jalankan kembali pernyataan setelah satu menit jika Anda tidak melihat hasil ini.

```
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)
```

Menggunakan AWS CLI

Prosedur berikut menghapus tabel dan semua datanya menggunakan file. AWS CLI

Untuk menghapus tabel menggunakan AWS CLI

1. Buka CloudShell
2. Hapus tabel Anda dengan pernyataan berikut.

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

3. Untuk memverifikasi bahwa tabel Anda telah dihapus, Anda dapat mencantumkan semua tabel di ruang kunci.

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

Anda akan melihat output berikut. Perhatikan bahwa operasi asinkron ini dapat memakan waktu. Re-run perintah lagi setelah beberapa saat untuk mengkonfirmasi bahwa tabel telah dihapus.

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

Hapus ruang kunci di Amazon Keyspaces

Agar tidak dikenakan biaya untuk ruang kunci, hapus semua ruang kunci yang tidak Anda gunakan. Saat Anda menghapus ruang kunci, ruang kunci dan semua tabelnya dihapus dan Anda berhenti menambah biaya untuk mereka.

Anda dapat menghapus ruang kunci menggunakan konsol, CQL, atau. AWS CLI

Menggunakan konsol

Prosedur berikut menghapus keyspace dan semua tabel dan datanya menggunakan konsol.

Untuk menghapus ruang kunci menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>

2. Di panel navigasi, pilih Keyspaces.
3. Pilih kotak di sebelah kiri nama setiap ruang tombol yang ingin Anda hapus.
4. Pilih Hapus.
5. Pada layar Delete keyspace, masukkan **Delete** di dalam kotak. Kemudian, pilih Hapus ruang kunci.
6. Untuk memverifikasi bahwa ruang kunci `catalog` telah dihapus, pilih Keyspaces di panel navigasi dan konfirmasikan bahwa itu tidak lagi terdaftar. Karena Anda menghapus ruang kuncinya, `book_awards` tabel di bawah Tabel juga tidak boleh terdaftar.

Menggunakan CQL

Prosedur berikut menghapus keyspace dan semua tabel dan data menggunakan CQL.

Untuk menghapus keyspace menggunakan CQL

1. Buka AWS CloudShell dan sambungkan ke Amazon Keyspaces menggunakan perintah berikut. Pastikan untuk memperbarui `us-east-1` dengan Wilayah Anda sendiri.

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

2. Hapus ruang kunci Anda dengan memasukkan pernyataan berikut.

```
DROP KEYSPACE IF EXISTS catalog ;
```

3. Verifikasi bahwa ruang kunci Anda telah dihapus.

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

Ruang kunci Anda tidak boleh terdaftar. Perhatikan bahwa karena ini adalah operasi asinkron, mungkin ada penundaan hingga ruang kunci dihapus. Setelah keyspace dihapus, output dari pernyataan akan terlihat seperti ini.

```
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)

Menggunakan AWS CLI

Prosedur berikut menghapus keyspace dan semua tabel dan datanya menggunakan file. AWS CLI

Untuk menghapus ruang kunci menggunakan AWS CLI

1. Buka AWS CloudShell
2. Hapus ruang kunci Anda dengan memasukkan pernyataan berikut.

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

3. Verifikasi bahwa ruang kunci Anda telah dihapus.

```
aws keyspaces list-keyspaces
```

Output dari pernyataan ini akan terlihat mirip dengan ini, dan hanya daftar keyspaces sistem. Perhatikan bahwa karena ini adalah operasi asinkron, mungkin ada penundaan hingga ruang kunci Anda dihapus.

```

{
  "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"
    }
]
}
```


Tutorial dan solusi untuk Amazon Keyspaces

Topik ini mencakup berbagai tutorial yang berjalan melalui langkah-langkah terperinci yang diperlukan untuk terhubung ke Amazon Keyspaces secara terprogram, atau menggunakan Amazon Keyspaces dengan layanan lain untuk membuat solusi. AWS Tutorial yang tersedia mencakup berbagai skenario akses lintas layanan populer dan menunjukkan cara mengintegrasikan Amazon Keyspaces dengan Amazon Virtual Private Cloud, Amazon Elastic Kubernetes Service, Amazon Simple AWS Glue Storage Service, dan, atau dengan teknologi open source seperti Apache Spark.

Untuk tutorial langkah demi langkah yang menunjukkan cara terhubung ke Amazon Keyspaces menggunakan driver Apache Cassandra, lihat [the section called “Menggunakan driver klien Cassandra”](#)

Topik

- [Tutorial: Connect ke Amazon Keyspaces menggunakan antarmuka VPC endpoint](#)
- [Tutorial: Integrasikan dengan Apache Spark untuk mengimpor atau mengeksport data](#)
- [Tutorial: Connect dari aplikasi container yang di-host di Amazon Elastic Kubernetes Service](#)
- [Tutorial: Ekspor tabel Amazon Keyspaces ke Amazon S3 menggunakan AWS Glue](#)

Tutorial: Connect ke Amazon Keyspaces menggunakan antarmuka VPC endpoint

Tutorial ini memandu Anda melalui pengaturan dan menggunakan antarmuka VPC endpoint untuk Amazon Keyspaces.

Endpoint VPC antarmuka memungkinkan komunikasi pribadi antara virtual private cloud (VPC) Anda yang berjalan di Amazon VPC dan Amazon Keyspaces. Endpoint VPC antarmuka didukung oleh AWS PrivateLink, yang merupakan AWS layanan yang memungkinkan komunikasi pribadi antara VPC dan layanan. AWS Untuk informasi selengkapnya, lihat [the section called “Menggunakan VPC endpoint antarmuka”](#).

Topik

- [Prasyarat dan pertimbangan tutorial](#)
- [Langkah 1: Luncurkan instans Amazon EC2](#)

- [Langkah 2: Konfigurasi instans Amazon EC2 Anda](#)
- [Langkah 3: Buat titik akhir VPC untuk Amazon Keyspaces](#)
- [Langkah 4: Konfigurasi izin untuk koneksi titik akhir VPC](#)
- [Langkah 5: Konfigurasi pemantauan dengan CloudWatch](#)
- [Langkah 6: \(Opsional\) Praktik terbaik untuk mengonfigurasi ukuran kolam koneksi untuk aplikasi Anda](#)
- [Langkah 7: \(Opsional\) Bersihkan](#)

Prasyarat dan pertimbangan tutorial

Sebelum Anda memulai tutorial ini, ikuti instruksi AWS pengaturan di [Mengakses Amazon Keyspaces \(untuk Apache Cassandra\)](#). Langkah-langkah ini termasuk mendaftar AWS dan membuat prinsipal AWS Identity and Access Management (IAM) dengan akses ke Amazon Keyspaces. Perhatikan nama pengguna IAM dan kunci akses karena Anda akan membutuhkannya nanti dalam tutorial ini.

Buat keyspace dengan nama myKeyspace dan setidaknya satu tabel untuk menguji koneksi menggunakan titik akhir VPC nanti dalam tutorial ini. Anda dapat menemukan instruksi terperinci di [Memulai](#).

Setelah menyelesaikan langkah-langkah prasyarat, lanjutkan ke [Langkah 1: Luncurkan instans Amazon EC2](#)

Langkah 1: Luncurkan instans Amazon EC2

Pada langkah ini, Anda meluncurkan instans Amazon EC2 di Amazon VPC default Anda. Anda kemudian dapat membuat dan menggunakan titik akhir VPC untuk Amazon Keyspaces.

Untuk meluncurkan instans Amazon EC2

1. Buka konsol Amazon EC2 di <https://console.aws.amazon.com/ec2/>
2. Pilih Luncurkan Instans, dan lakukan hal berikut:

Dari dasbor konsol EC2, di kotak Launch instance, pilih Launch instance, lalu pilih Launch instance dari opsi yang muncul.

Di bawah Nama dan tag, untuk Nama, masukkan nama deskriptif untuk instance Anda.

Di bawah Gambar Aplikasi dan OS (Gambar Mesin Amazon):

- Pilih Quick Start, lalu pilih Ubuntu. Ini adalah sistem operasi (OS) untuk instans Anda.
- Di bawah Amazon Machine Image (AMI), Anda dapat menggunakan gambar default yang ditandai sebagai Tingkat gratis yang memenuhi syarat. Amazon Machine Image (AMI) adalah konfigurasi dasar yang berfungsi sebagai templat untuk instans Anda.

Di bawah Jenis Instance:

- Dari daftar tipe Instance, pilih tipe instans t2.micro, yang dipilih secara default.

Di bawah Pasangan kunci (login), untuk nama pasangan Kunci, pilih salah satu opsi berikut untuk tutorial ini:

- Jika Anda tidak memiliki pasangan kunci Amazon EC2, pilih Buat pasangan kunci baru dan ikuti petunjuknya. Anda akan diminta untuk mengunduh file kunci pribadi (file.pem). Anda akan memerlukan file ini nanti ketika Anda masuk ke instans Amazon EC2 Anda, jadi perhatikan jalur file.
- Jika Anda sudah memiliki pasangan kunci Amazon EC2 yang ada, buka Pilih pasangan kunci dan pilih pasangan kunci Anda dari daftar. Anda harus sudah memiliki file kunci privat (file .pem) yang tersedia untuk masuk ke instans Amazon EC2 Anda.

Di bawah Pengaturan Jaringan:

- Pilih Edit.
- Pilih Pilih grup keamanan yang ada.
- Dalam daftar grup keamanan, pilih default. Ini adalah grup keamanan default untuk VPC Anda.

Lanjutkan ke Ringkasan.

- Tinjau ringkasan konfigurasi instans Anda di panel Summary. Saat Anda siap, pilih Launch instance.
3. Pada layar penyelesaian untuk instans Amazon EC2 baru, pilih ubin Connect to instance. Layar berikutnya menampilkan informasi yang diperlukan dan langkah-langkah yang diperlukan untuk terhubung ke instans baru Anda. Perhatikan informasi berikut:
- Perintah sampel untuk melindungi file kunci

- String koneksi
- Nama DNS IPv4 Publik

Setelah mencatat informasi di halaman ini, Anda dapat melanjutkan ke langkah berikutnya dalam tutorial ini ([Langkah 2: Konfigurasi instans Amazon EC2 Anda](#)).

Note

Diperlukan waktu beberapa menit agar instans Amazon EC2 Anda tersedia. Sebelum Anda melanjutkan ke langkah berikutnya, pastikan Status Instans dalam kondisi `running` dan semua Pemeriksaan Status telah lulus.

Langkah 2: Konfigurasi instans Amazon EC2 Anda

Saat instans Amazon EC2 Anda tersedia, Anda dapat masuk ke dalamnya dan menyiapkannya untuk penggunaan pertama.

Note

Langkah-langkah berikut mengasumsikan bahwa Anda terhubung ke instans Amazon EC2 dari komputer yang menjalankan Linux. Untuk cara lain untuk terhubung, lihat [Connect to Linux Anda](#) di Panduan Pengguna Amazon EC2.

Untuk mengonfigurasi instans Amazon EC2

1. Anda perlu mengotorisasi lalu lintas SSH masuk ke instans Amazon EC2 Anda. Untuk melakukan ini, buat grup keamanan EC2 baru, lalu tetapkan grup keamanan ke instans EC2 Anda.
 - a. Di panel navigasi, pilih Security Groups (Grup Keamanan).
 - b. Pilih Buat Grup Keamanan. Di jendela Buat Grup Keamanan, lakukan hal berikut:
 - Nama grup keamanan — Masukkan nama untuk grup keamanan Anda. Misalnya: `my-ssh-access`
 - Deskripsi — Masukkan deskripsi singkat untuk grup keamanan.

- VPC — Pilih VPC default Anda.
- Di bagian Aturan masuk, pilih Tambahkan Aturan dan lakukan hal berikut:
 - Jenis - Pilih SSH.
 - Sumber - Pilih IP Saya.
 - Pilih Tambahkan aturan.

Di bagian bawah halaman, konfirmasi pengaturan konfigurasi dan pilih Buat Grup Keamanan.

- c. Di panel navigasi, pilih Instans.
 - d. Pilih instans Amazon EC2 yang diluncurkan di [Langkah 1: Luncurkan instans Amazon EC2](#).
 - e. Pilih Tindakan, pilih Keamanan, lalu pilih Ubah Grup Keamanan.
 - f. Di Ubah Grup Keamanan, buka Grup keamanan terkait dan masukkan grup keamanan yang Anda buat sebelumnya dalam prosedur ini (misalnya, my-ssh-access). Saat ini grup keamanan default juga harus dipilih. Konfirmasi pengaturan konfigurasi dan pilih Simpan.
2. Gunakan perintah berikut untuk melindungi file kunci pribadi Anda dari akses. Jika Anda melewati langkah ini, koneksi gagal.

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

3. Gunakan perintah ssh untuk masuk ke instans Amazon EC2 Anda, seperti dalam contoh berikut.

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

Anda perlu menentukan file kunci pribadi Anda (file.pem) dan nama DNS publik instance Anda. (Lihat [Langkah 1: Luncurkan instans Amazon EC2](#)).

ID login adalah ubuntu. Tidak diperlukan kata sandi.

Untuk informasi selengkapnya tentang mengizinkan koneksi ke instans Amazon EC2 Anda dan untuk AWS CLI petunjuk, lihat [Mengotorisasi lalu lintas masuk untuk instans Linux Anda di Panduan Pengguna Amazon EC2](#).

4. Unduh dan instal versi terbaru dari file AWS Command Line Interface.
 - a. Instal unzip.

```
sudo apt install unzip
```

- b. Unduh zip file dengan file AWS CLI.

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

- c. Buka filenya.

```
unzip awscliv2.zip
```

- d. Instal AWS CLI.

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

- e. Konfirmasikan versi AWS CLI instalasi.

```
aws --version
```

Outputnya akan terlihat seperti ini:

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

5. Konfigurasi AWS kredensial Anda, seperti yang ditunjukkan pada contoh berikut. Masukkan ID kunci AWS akses, kunci rahasia, dan nama Wilayah default Anda saat diminta.

```
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. Tambahkan aturan ke grup keamanan VPC Anda yang memungkinkan akses HTTP, HTTPS, dan SSH masuk dari alamat IPv6.
7. Untuk mengonfirmasi bahwa titik akhir VPC Anda telah dikonfigurasi dengan benar, Anda harus menggunakan koneksi `cqlsh` ke Amazon Keyspaces. Jika Anda menggunakan lingkungan lokal atau editor CQL Amazon Keyspaces di, koneksi secara otomatis melewati titik akhir publik

Konsol Manajemen AWS, bukan titik akhir VPC Anda. Untuk digunakan `cqlsh` untuk menguji koneksi titik akhir VPC Anda dalam tutorial ini, selesaikan instruksi penyiapan di [Menggunakan `cqlsh` untuk terhubung ke Amazon Keyspaces](#)

Anda sekarang siap untuk membuat titik akhir VPC dual-stack untuk Amazon Keyspaces.

Langkah 3: Buat titik akhir VPC untuk Amazon Keyspaces

Pada langkah ini, Anda membuat titik akhir VPC dual-stack untuk Amazon Keyspaces menggunakan file. AWS CLI Untuk membuat titik akhir VPC [menggunakan konsol VPC, Anda dapat mengikuti instruksi Buat titik akhir VPC](#) di Panduan.AWS PrivateLink Saat memfilter untuk nama Layanan, masukkan **Cassandra**.

Untuk membuat titik akhir VPC menggunakan AWS CLI

1. Sebelum memulai, verifikasi bahwa Anda dapat berkomunikasi dengan Amazon Keyspaces menggunakan titik akhir publiknya.

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

Output menampilkan daftar tabel Amazon Keyspaces yang terdapat dalam keyspace yang ditentukan. Jika Anda tidak memiliki tabel apa pun, daftarnya kosong.

```
{
  "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. Verifikasi bahwa Amazon Keyspaces adalah layanan yang tersedia untuk membuat titik akhir VPC di Wilayah saat ini. AWS (Perintah ditampilkan dalam teks tebal, diikuti dengan contoh output.)

```
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"
  ]
}
```

Jika Amazon Keyspaces adalah salah satu layanan yang tersedia dalam output perintah, Anda dapat melanjutkan dengan membuat titik akhir VPC.

3. Untuk terhubung ke Amazon Keyspaces menggunakan titik akhir dual-stack yang diaktifkan IPv6, konfirmasikan bahwa VPC Anda mendukung IPv6 dan konfigurasi subnet dengan dukungan IPv6. Untuk menambahkan dukungan IPv6 ke VPC yang ada yang saat ini hanya mendukung IPv4, lihat dukungan IPv6 [untuk VPC Anda di Panduan Pengguna Amazon VPC](#);
4. Tentukan pengidentifikasi VPC Anda.

```
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"
  }
]
}

```

Dalam contoh output, ID VPC adalah `vpc-a1234bcd`.

- Gunakan filter untuk mengumpulkan detail tentang subnet 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"
          }
        },
        {
          "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": 4096,
  "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
    }
]
}

```

Dalam contoh output, ada dua ID subnet yang tersedia: subnet-70b24b16 dan subnet-c63ffbe7.

6. Buat titik akhir VPC. Untuk parameter `--vpc-id`, tentukan ID VPC dari langkah sebelumnya. Untuk `--subnet-ids` parameter, tentukan ID subnet dari langkah sebelumnya. Gunakan `--vpc-endpoint-type` parameter untuk mendefinisikan titik akhir sebagai antarmuka. Untuk membuat titik akhir dual-stack, gunakan `--ip-address-type dualstack`. Untuk informasi selengkapnya tentang perintah, lihat [create-vpc-endpoint](#) di Referensi AWS CLI Perintah.

```

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-0000000abc111d2ef3",
    "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"
  }
}
```

```
}
```

Langkah 4: Konfigurasikan izin untuk koneksi titik akhir VPC

Prosedur dalam langkah ini menunjukkan cara mengonfigurasi aturan dan izin untuk menggunakan titik akhir VPC dengan Amazon Keyspaces.

Untuk mengonfigurasi aturan masuk untuk titik akhir baru untuk mengizinkan lalu lintas masuk TCP

1. Di konsol VPC Amazon, di panel sisi kiri, pilih Endpoints dan pilih endpoint yang Anda buat di langkah sebelumnya.
2. Pilih Kelola grup keamanan, lalu pilih grup keamanan yang terkait dengan titik akhir ini.
3. Pilih Aturan masuk dan kemudian pilih Edit aturan masuk.
4. Tambahkan aturan masuk dengan Type as CQLSH/CASSANDRA. Ini mengatur rentang Port, secara otomatis ke 9142.
5. Untuk menyimpan aturan masuk baru, pilih Simpan aturan.

Untuk mengonfigurasi izin pengguna IAM

1. Konfirmasikan bahwa pengguna IAM yang digunakan untuk terhubung ke Amazon Keyspaces memiliki izin yang sesuai. Di AWS Identity and Access Management (IAM), Anda dapat menggunakan kebijakan AWS terkelola AmazonKeyspacesReadOnlyAccess untuk memberikan akses baca pengguna IAM ke Amazon Keyspaces.
 - a. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
 - b. Pada dasbor konsol IAM, pilih Pengguna, lalu pilih pengguna IAM Anda dari daftar.
 - c. Di halaman Ringkasan, pilih Tambahkan izin.
 - d. Pilih Lampirkan kebijakan yang sudah ada secara langsung.
 - e. Dari daftar kebijakan, pilih AmazonKeyspacesReadOnlyAccess, lalu pilih Berikutnya: Tinjau.
 - f. Pilih Tambahkan izin.
2. Verifikasi bahwa Anda dapat mengakses Amazon Keyspaces melalui titik akhir VPC.

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

Jika mau, Anda dapat mencoba beberapa AWS CLI perintah lain untuk Amazon Keyspaces. Untuk informasi selengkapnya, lihat [Referensi Perintah AWS CLI](#).

 Note

Izin minimum yang diperlukan untuk pengguna IAM atau peran untuk mengakses Amazon Keyspaces adalah izin baca ke tabel sistem, seperti yang ditunjukkan dalam kebijakan berikut. Untuk informasi selengkapnya tentang izin berbasis kebijakan, lihat [the section called “Identity-based contoh kebijakan”](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cassandra:Select"
      ],
      "Resource": [
        "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
      ]
    }
  ]
}
```

3. Berikan akses baca pengguna IAM ke instans Amazon EC2 dengan VPC.

Saat menggunakan Amazon Keyspaces dengan titik akhir VPC, Anda harus memberikan pengguna IAM atau peran yang mengakses izin hanya-baca Amazon Keyspaces ke instans Amazon EC2 dan VPC untuk mengumpulkan data endpoint dan antarmuka jaringan. Amazon Keyspaces menyimpan informasi ini dalam `system.peers` tabel dan menggunakannya untuk mengelola koneksi.

 Note

Kebijakan terkelola `AmazonKeyspacesReadOnlyAccess_v2` dan `AmazonKeyspacesFullAccess` menyertakan izin yang diperlukan agar Amazon

Keyspaces mengakses instans Amazon EC2 untuk membaca informasi tentang titik akhir VPC antarmuka yang tersedia.

- a. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
- b. Di dasbor konsol IAM, pilih Kebijakan.
- c. Pilih Buat kebijakan, lalu pilih tab JSON.
- d. Salin kebijakan berikut dan pilih Berikutnya: Tag.

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

- e. Pilih Berikutnya: Tinjau, masukkan nama keyspacesVPCendpoint kebijakan, dan pilih Buat kebijakan.
 - f. Pada dasbor konsol IAM, pilih Pengguna, lalu pilih pengguna IAM Anda dari daftar.
 - g. Di halaman Ringkasan, pilih Tambahkan izin.
 - h. Pilih Lampirkan kebijakan yang sudah ada secara langsung.
 - i. Dari daftar kebijakan, pilih KeyspaceSVPcendPoint, lalu pilih Berikutnya: Tinjau.
 - j. Pilih Tambahkan izin.
4. Untuk memverifikasi bahwa `system.peers` tabel Amazon Keyspaces diperbarui dengan informasi VPC, jalankan kueri berikut dari instans Amazon EC2 Anda menggunakan `cqlsh`. Jika Anda belum menginstal `cqlsh` instans Amazon EC2 di langkah 2, ikuti petunjuk di [the section called "Menggunakan cqlsh-expansion"](#)

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

Output mengembalikan node dengan alamat IP IPv6 pribadi, tergantung pada VPC dan pengaturan subnet Anda di Wilayah Anda. 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

Anda harus menggunakan `cqlsh` koneksi ke Amazon Keyspaces untuk mengonfirmasi bahwa titik akhir VPC Anda telah dikonfigurasi dengan benar. Jika Anda menggunakan lingkungan lokal atau editor CQL Amazon Keyspaces di, koneksi secara otomatis melewati titik akhir publik Konsol Manajemen AWS, bukan titik akhir VPC Anda. Jika Anda melihat sembilan alamat IP, ini adalah entri Amazon Keyspaces yang secara otomatis menulis ke tabel untuk koneksi `system.peers` titik akhir publik.

Langkah 5: Konfigurasi pemantauan dengan CloudWatch

Langkah ini menunjukkan cara menggunakan Amazon CloudWatch untuk memantau koneksi titik akhir VPC ke Amazon Keyspaces.

AWS PrivateLink menerbitkan titik data ke CloudWatch tentang titik akhir antarmuka Anda. Anda dapat menggunakan metrik untuk memverifikasi bahwa sistem Anda bekerja sesuai harapan. AWS/PrivateLinkEndpointsNamespace CloudWatch termasuk metrik untuk titik akhir antarmuka. Untuk informasi selengkapnya, lihat [CloudWatch metrik AWS PrivateLink](#) di AWS PrivateLink Panduan.

Untuk membuat CloudWatch dasbor dengan metrik titik akhir VPC

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, silakan pilih Dasbor. Kemudian pilih Buat dasbor. Masukkan nama untuk dasbor dan pilih Buat.
3. Di bawah Tambah widget, pilih Nomor.
4. Di bawah Metrik, pilih AWS/PrivateLinkEndpoints.
5. Pilih Jenis Titik Akhir, Nama Layanan, ID Titik Akhir VPC, ID VPC.
6. Pilih metrik `ActiveConnections` dan `NewConnections`, dan pilih Buat Widget.
7. Simpan dasbor.

`ActiveConnections` Metrik didefinisikan sebagai jumlah koneksi aktif bersamaan yang diterima titik akhir selama periode satu menit terakhir. `NewConnections` Metrik didefinisikan sebagai jumlah koneksi baru yang dibuat melalui titik akhir selama periode satu menit terakhir.

Untuk informasi selengkapnya tentang membuat dasbor, lihat [Membuat dasbor](#) di Panduan CloudWatch Pengguna.

Langkah 6: (Opsional) Praktik terbaik untuk mengonfigurasi ukuran kolam koneksi untuk aplikasi Anda

Pada bagian ini, kami menguraikan cara menentukan ukuran kumpulan koneksi yang ideal berdasarkan persyaratan throughput kueri aplikasi Anda.

Amazon Keyspaces memungkinkan maksimum 3.000 kueri CQL per detik per koneksi TCP. Jadi hampir tidak ada batasan jumlah koneksi yang dapat dibuat driver dengan Amazon Keyspaces. Namun, kami menyarankan agar Anda mencocokkan ukuran kumpulan koneksi dengan persyaratan aplikasi Anda dan mempertimbangkan titik akhir yang tersedia saat Anda menggunakan Amazon Keyspaces dengan koneksi titik akhir VPC.

Anda mengonfigurasi ukuran kolam koneksi di driver klien. Misalnya, berdasarkan ukuran kolam lokal 2 dan titik akhir antarmuka VPC yang dibuat di 3 Availability Zones, driver membuat 6 koneksi untuk kueri (total 7, yang mencakup koneksi kontrol). Dengan menggunakan 6 koneksi ini, Anda dapat mendukung maksimal 18.000 kueri CQL per detik.

Jika aplikasi Anda perlu mendukung 40.000 kueri CQL per detik, kerjakan mundur dari jumlah kueri yang diperlukan untuk menentukan ukuran kumpulan koneksi yang diperlukan. Untuk mendukung 40.000 kueri CQL per detik, Anda perlu mengonfigurasi ukuran kolam lokal menjadi minimal 5, yang mendukung minimal 45.000 kueri CQL per detik.

Anda dapat memantau jika Anda melebihi kuota untuk jumlah maksimum operasi per detik, per koneksi dengan menggunakan `PerConnectionRequestRateExceeded` CloudWatch metrik di `AWS/Cassandra` namespace. `PerConnectionRequestRateExceeded` metrik menunjukkan jumlah permintaan ke Amazon Keyspaces yang melebihi kuota untuk tingkat permintaan per koneksi.

Contoh kode dalam langkah ini menunjukkan cara memperkirakan dan mengonfigurasi penyatuan koneksi saat Anda menggunakan titik akhir VPC antarmuka.

Java

Anda dapat mengonfigurasi jumlah koneksi per kumpulan di driver Java. Untuk contoh lengkap koneksi driver klien Java, lihat [the section called “Menggunakan driver klien Cassandra Java”](#).

Ketika driver klien dimulai, pertama koneksi kontrol dibuat untuk tugas-tugas administratif, seperti untuk skema dan perubahan topologi. Kemudian koneksi tambahan dibuat.

Dalam contoh berikut, konfigurasi driver ukuran kolam lokal ditentukan sebagai 2. Jika titik akhir VPC dibuat di 3 subnet dalam VPC, ini menghasilkan 7 in CloudWatch untuk titik akhir antarmuka, seperti yang ditunjukkan `NewConnections` pada rumus berikut.

```
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
    }
}

```

Jika jumlah koneksi aktif tidak sesuai dengan ukuran kumpulan yang dikonfigurasi (agregasi di seluruh subnet) +1 koneksi kontrol, ada sesuatu yang mencegah koneksi dibuat.

Node.js

Anda dapat mengonfigurasi jumlah koneksi per kumpulan di Node.js driver. Untuk contoh lengkap koneksi driver Node.js klien, lihat [the section called “Menggunakan driver klien Cassandra Node.js”](#).

Contoh kode di bawah ini menggunakan *keyspaces-bundle.pem* file untuk sertifikat digital. Untuk informasi selengkapnya, lihat [the section called “Sebelum kamu memulai”](#).

Untuk contoh kode berikut, konfigurasi driver ukuran kolam lokal ditentukan sebagai 1. Jika titik akhir VPC dibuat di 4 subnet dalam VPC, ini menghasilkan 5 in CloudWatch untuk titik akhir antarmuka, seperti yang ditunjukkan NewConnections pada rumus berikut.

```

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 }
});

```

Langkah 7: (Opsional) Bersihkan

Jika Anda ingin menghapus sumber daya yang telah Anda buat dalam tutorial ini, ikuti prosedur ini.

Untuk menghapus titik akhir VPC Anda untuk Amazon Keyspaces

1. Masuk ke instans Amazon EC2 Anda.
2. Tentukan ID titik akhir VPC yang digunakan untuk Amazon Keyspaces. Jika Anda menghilangkan grep parameter, informasi titik akhir VPC ditampilkan untuk semua layanan.

```

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"
  }
}

```

Dalam contoh output, ID titik akhir VPC adalah `vpce-1a23b4c5`. Pastikan untuk mengganti nilai ini dengan nilai Anda sendiri.

3. Hapus titik akhir VPC.

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-1a23b4c5

{
  "Unsuccessful": []
}
```

Array kosong `[]` menunjukkan keberhasilan (tidak ada permintaan yang gagal).

Untuk menghentikan instans Amazon EC2 Anda

1. Buka konsol Amazon EC2 di. <https://console.aws.amazon.com/ec2/>
2. Di panel navigasi, pilih Instans.
3. Pilih instans Amazon EC2 Anda.
4. Pilih Tindakan, pilih Status Instance, lalu pilih Terminate.
5. Dalam jendela konfirmasi, pilih Ya, Hentikan.

Tutorial: Integrasikan dengan Apache Spark untuk mengimpor atau mengekspor data

Apache Spark adalah mesin open-source untuk analisis data skala besar. Apache Spark memungkinkan Anda melakukan analitik pada data yang disimpan di Amazon Keyspaces dengan lebih efisien. Anda juga dapat menggunakan Amazon Keyspaces untuk menyediakan aplikasi dengan akses baca milidetik satu digit yang konsisten ke data analitik dari Spark. Konektor Spark Cassandra sumber terbuka menyederhanakan membaca dan menulis data antara Amazon Keyspaces dan Spark.

Dukungan Amazon Keyspaces untuk Spark Cassandra Connector menyederhanakan menjalankan beban kerja Cassandra di pipeline Spark-based analitik dengan menggunakan layanan database yang dikelola sepenuhnya dan tanpa server. Dengan Amazon Keyspaces, Anda tidak perlu khawatir tentang Spark bersaing untuk sumber daya infrastruktur dasar yang sama dengan tabel Anda. Tabel

Amazon Keyspaces menskalakan naik dan turun secara otomatis berdasarkan lalu lintas aplikasi Anda.

Tutorial berikut memandu Anda melalui langkah-langkah dan praktik terbaik yang diperlukan untuk membaca dan menulis data ke Amazon Keyspaces menggunakan Spark Cassandra Connector. Tutorial ini menunjukkan cara memigrasikan data ke Amazon Keyspaces dengan memuat data dari file dengan Spark Cassandra Connector dan menulisnya ke tabel Amazon Keyspaces. Kemudian, tutorial menunjukkan cara membaca data kembali dari Amazon Keyspaces menggunakan Spark Cassandra Connector. Anda akan melakukan ini untuk menjalankan beban kerja Cassandra di Spark-based pipeline analitik.

Topik

- [Prasyarat untuk membuat koneksi ke Amazon Keyspaces dengan Spark Cassandra Connector](#)
- [Langkah 1: Konfigurasi Amazon Keyspaces untuk integrasi dengan Apache Cassandra Spark Connector](#)
- [Langkah 2: Konfigurasi Konektor Spark Apache Cassandra](#)
- [Langkah 3: Buat file konfigurasi aplikasi](#)
- [Langkah 4: Siapkan data sumber dan tabel target di Amazon Keyspaces](#)
- [Langkah 5: Tulis dan baca data Amazon Keyspaces menggunakan Apache Cassandra Spark Connector](#)
- [Memecahkan masalah kesalahan umum saat menggunakan Konektor Spark Cassandra dengan Amazon Keyspaces](#)

Prasyarat untuk membuat koneksi ke Amazon Keyspaces dengan Spark Cassandra Connector

Sebelum Anda terhubung ke Amazon Keyspaces dengan Spark Cassandra Connector, Anda perlu memastikan bahwa Anda telah menginstal yang berikut ini. Kompatibilitas Amazon Keyspaces dengan Spark Cassandra Connector telah diuji dengan versi yang direkomendasikan berikut:

- Java versi 8
- Scala 2.12
- Percikan 3.4
- Konektor Cassandra 2.5 dan lebih tinggi
- Pengemudi Cassandra 4.12

1. Untuk menginstal Scala, ikuti instruksi di <https://www.scala-lang.org/download/scala2.html>.
2. Untuk menginstal Spark 3.4.1, ikuti contoh ini.

```
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
```
```

## Langkah 1: Konfigurasikan Amazon Keyspaces untuk integrasi dengan Apache Cassandra Spark Connector

Pada langkah ini, Anda mengonfirmasi bahwa partisi untuk akun Anda kompatibel dengan Apache Spark Connector dan mengatur izin IAM yang diperlukan. Praktik terbaik berikut membantu Anda menyediakan read/write kapasitas yang cukup untuk tabel.

1. Konfirmasikan bahwa `Murmur3Partitioner` partisi adalah partisi default untuk akun Anda. Partisi ini kompatibel dengan Konektor Spark Cassandra. Untuk informasi lebih lanjut tentang partisi dan cara mengubahnya, lihat [the section called “Bekerja dengan partisi”](#)
2. Siapkan izin IAM Anda untuk Amazon Keyspaces, menggunakan titik akhir VPC antarmuka, dengan Apache Spark.
  - Tetapkan read/write akses ke tabel pengguna dan baca akses ke tabel sistem seperti yang ditunjukkan dalam contoh kebijakan IAM yang tercantum di bawah ini.
  - [Mengisi tabel `system.peers` dengan titik akhir VPC antarmuka yang tersedia diperlukan untuk klien yang mengakses Amazon Keyspaces dengan Spark over VPC endpoint.](#)

```
{
 "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. Pertimbangkan praktik terbaik berikut untuk mengonfigurasi kapasitas read/write throughput yang memadai untuk tabel Amazon Keyspaces Anda guna mendukung lalu lintas dari Spark Cassandra Connector.
  - Mulai gunakan kapasitas sesuai permintaan untuk membantu Anda menguji skenario.
  - Untuk mengoptimalkan biaya throughput tabel untuk lingkungan produksi, gunakan pembatas laju untuk lalu lintas dari konektor, dan konfigurasi tabel Anda untuk menggunakan kapasitas yang disediakan dengan penskalaan otomatis. Untuk informasi selengkapnya, lihat [the section called “Kelola kapasitas throughput dengan penskalaan otomatis”](#).
  - Anda dapat menggunakan pembatas tarif tetap yang disertakan dengan driver Cassandra. [Ada beberapa pembatas tarif yang disesuaikan dengan Amazon Keyspaces di repo sampel.AWS](#)
  - Untuk informasi selengkapnya tentang manajemen kapasitas, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).

## Langkah 2: Konfigurasi Konektor Spark Apache Cassandra

Apache Spark adalah platform komputasi tujuan umum yang dapat Anda konfigurasi dengan berbagai cara. Untuk mengonfigurasi Spark dan Spark Cassandra Connector untuk integrasi dengan



Amazon Keyspaces, kami sarankan Anda memulai dengan pengaturan konfigurasi minimum yang dijelaskan di bagian berikut, dan kemudian meningkatkannya nanti sesuai dengan beban kerja Anda.

- Buat ukuran partisi Spark lebih kecil dari 8 MB.

Dalam Spark, partisi mewakili potongan atom data yang dapat dijalankan secara paralel. Saat Anda menulis data ke Amazon Keyspaces dengan Spark Cassandra Connector, semakin kecil partisi Spark, semakin kecil jumlah catatan yang akan ditulis tugas. Jika tugas Spark mengalami beberapa kesalahan, tugas gagal setelah jumlah percobaan ulang yang ditentukan habis. Untuk menghindari memutar ulang tugas besar dan memproses ulang banyak data, jaga agar ukuran partisi Spark tetap kecil.

- Gunakan jumlah penulisan bersamaan yang rendah per eksekutor dengan sejumlah besar percobaan ulang.

Amazon Keyspaces mengembalikan kesalahan kapasitas yang tidak mencukupi kembali ke driver Cassandra sebagai batas waktu operasi. Anda tidak dapat mengatasi batas waktu yang disebabkan oleh kapasitas yang tidak mencukupi dengan mengubah durasi batas waktu yang dikonfigurasi karena Konektor Spark Cassandra mencoba ulang permintaan secara transparan menggunakan `MultipleRetryPolicy` Untuk memastikan bahwa percobaan ulang tidak membebani kumpulan koneksi pengemudi, gunakan jumlah penulisan bersamaan yang rendah per eksekutor dengan sejumlah besar percobaan ulang. Cuplikan kode berikut adalah contoh dari ini.

```
spark.cassandra.query.retry.count = 500
spark.cassandra.output.concurrent.writes = 3
```

- Memecah total throughput dan mendistribusikannya di beberapa sesi Cassandra.
  - Konektor Cassandra Spark membuat satu sesi untuk setiap pelaksana Spark. Pikirkan sesi ini sebagai unit skala untuk menentukan throughput yang diperlukan dan jumlah koneksi yang diperlukan.
  - Saat menentukan jumlah core per eksekutor dan jumlah core per tugas, mulailah rendah dan tingkatkan sesuai kebutuhan.
  - Atur kegagalan tugas Spark untuk memungkinkan pemrosesan jika terjadi kesalahan sementara. Setelah Anda terbiasa dengan karakteristik dan persyaratan lalu lintas aplikasi Anda, kami sarankan pengaturan `spark.task.maxFailures` ke nilai terbatas.
  - Misalnya, konfigurasi berikut dapat menangani dua tugas bersamaan per pelaksana, per sesi:

```
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
```

- Matikan batching.
- Kami menyarankan Anda mematikan batching untuk meningkatkan pola akses acak. Cuplikan kode berikut adalah contoh dari ini.

```
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)
```

- Setel **SPARK\_LOCAL\_DIRS** ke disk lokal yang cepat dengan ruang yang cukup.
- Secara default, Spark menyimpan file keluaran peta dan kumpulan data terdistribusi tangguh (RDD) ke folder. /tmp Bergantung pada konfigurasi host Spark Anda, ini dapat mengakibatkan tidak ada ruang tersisa pada kesalahan gaya perangkat.
- Untuk mengatur variabel SPARK\_LOCAL\_DIRS lingkungan ke direktori yang disebut/example/spark-dir, Anda dapat menggunakan perintah berikut.

```
export SPARK_LOCAL_DIRS=/example/spark-dir
```

## Langkah 3: Buat file konfigurasi aplikasi

Untuk menggunakan Konektor Spark Cassandra open-source dengan Amazon Keyspaces, Anda perlu menyediakan file konfigurasi aplikasi yang berisi pengaturan yang diperlukan untuk terhubung dengan driver Java. DataStax Anda dapat menggunakan kredensi khusus layanan atau plugin SiGv4 untuk terhubung.

Jika Anda belum melakukannya, Anda perlu mengonversi sertifikat digital yang digunakan untuk membuat koneksi TLS menjadi file TrustStore. Anda dapat mengikuti langkah-langkah rinci di [the section called “Sebelum Anda mulai”](#) dari tutorial koneksi driver Java. Catat jalur file dan kata sandi TrustStore karena Anda memerlukan informasi ini saat membuat file konfigurasi aplikasi.

### Connect dengan otentikasi SiGv4

Bagian ini menunjukkan contoh `application.conf` file yang dapat Anda gunakan saat menghubungkan dengan AWS kredensyal dan plugin SigV4. Jika Anda belum melakukannya, Anda perlu membuat kunci akses IAM Anda (ID kunci akses dan kunci akses rahasia) dan menyimpannya

di file AWS konfigurasi Anda atau sebagai variabel lingkungan. Untuk petunjuk mendetail, lihat [the section called “Kredensi yang diperlukan untuk otentikasi AWS”](#).

Dalam contoh berikut, ganti path file ke file TrustStore Anda, dan ganti kata sandi.

```
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
}
```

Perbarui dan simpan file konfigurasi ini sebagai `/home/user1/application.conf`. Contoh berikut menggunakan jalur ini.

## Connect dengan kredensi khusus layanan

Bagian ini menunjukkan contoh `application.conf` file yang dapat Anda gunakan saat menghubungkan dengan kredensi khusus layanan. Jika Anda belum melakukannya, Anda perlu membuat kredensial khusus layanan untuk Amazon Keyspaces. Untuk petunjuk mendetail, lihat [the section called “Buat kredensial khusus layanan”](#).

Dalam contoh berikut, ganti `username` dan `password` dengan kredensial Anda sendiri. Juga, ganti jalur file ke file TrustStore Anda, dan ganti kata sandi.

```

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
 }
 }
 }
}

```

Perbarui dan simpan file konfigurasi ini `/home/user1/application.conf` untuk digunakan dengan contoh kode.

## Connect dengan tarif tetap

Untuk memaksa laju tetap per pelaksana Spark, Anda dapat menentukan throttler permintaan. Throttler permintaan ini membatasi tingkat permintaan per detik. Konektor Spark Cassandra menyebarkan sesi Cassandra per eksekutor. Menggunakan rumus berikut dapat membantu Anda mencapai throughput yang konsisten terhadap tabel.

```
max-request-per-second * numberOfExecutors = total throughput against a table
```

Anda dapat menambahkan contoh ini ke file konfigurasi aplikasi yang Anda buat sebelumnya.

```
datastax-java-driver {
 advanced.throttler {
 class = RateLimitingRequestThrottler

 max-requests-per-second = 3000
 max-queue-size = 30000
 drain-interval = 1 millisecond
 }
}
```

## Langkah 4: Siapkan data sumber dan tabel target di Amazon Keyspaces

Pada langkah ini, Anda membuat file sumber dengan data sampel dan tabel Amazon Keyspaces.

1. Buat file sumber. Anda dapat memilih salah satu opsi berikut:

- Untuk tutorial ini, Anda menggunakan file nilai dipisahkan koma (CSV) dengan nama `keyspaces_sample_table.csv` sebagai file sumber untuk migrasi data. File sampel yang disediakan berisi beberapa baris data untuk tabel dengan nama `book_awards`.
- Download contoh file CSV (`keyspaces_sample_table.csv`) yang terkandung dalam file arsip berikut [samplemigration.zip](#). Buka zip arsip dan catat jalur `keyspaces_sample_table.csv`.
- Jika Anda ingin mengikuti file CSV Anda sendiri untuk menulis data ke Amazon Keyspaces, pastikan datanya diacak. Data yang dibaca langsung dari database atau diekspor ke file datar biasanya diurutkan oleh partisi dan kunci primer. Mengimpor data yang dipesan ke Amazon Keyspaces dapat menyebabkannya ditulis ke segmen yang lebih kecil dari partisi Amazon Keyspaces, yang menghasilkan distribusi lalu lintas yang tidak merata. Hal ini dapat menyebabkan kinerja lebih lambat dan tingkat kesalahan yang lebih tinggi.

Sebaliknya, pengacakan data membantu memanfaatkan kemampuan penyeimbangan beban bawaan Amazon Keyspaces dengan mendistribusikan lalu lintas di seluruh partisi secara lebih merata. Ada berbagai alat yang dapat Anda gunakan untuk mengacak data. Untuk contoh yang menggunakan alat sumber terbuka [Shuf](#), lihat [the section called “Langkah 2: Siapkan data”](#) di tutorial migrasi data. Berikut ini adalah contoh yang menunjukkan cara mengacak data sebagai DataFrame file.

```
import org.apache.spark.sql.functions.randval
```

```
shuffledDF = dataframe.orderBy(rand())
```

## 2. Buat keyspace target dan tabel di Amazon Keyspaces.

- a. Connect ke Amazon Keyspaces menggunakan file. `cqlsh-expansion` Untuk petunjuk `cqlsh-expansion` penginstalan, lihat [the section called “Menggunakan cqlsh-expansion”](#).

Ganti titik akhir layanan dalam contoh berikut dengan nilai Anda sendiri.

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

- b. Buat keyspace baru dengan nama `catalog` seperti yang ditunjukkan pada contoh berikut.

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

- c. Setelah keyspace baru memiliki status yang tersedia, gunakan kode berikut untuk membuat tabel `book_awards` target. Untuk mempelajari lebih lanjut tentang pembuatan sumber daya asinkron dan cara memeriksa apakah sumber daya tersedia, lihat [the section called “Periksa status pembuatan keyspace”](#)

```
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)
);
```

## Langkah 5: Tulis dan baca data Amazon Keyspaces menggunakan Apache Cassandra Spark Connector

Pada langkah ini, Anda mulai dengan memuat data dari file sampel ke dalam `DataFrame` dengan `Spark Cassandra Connector`. Selanjutnya, Anda menulis data dari tabel `DataFrame` Amazon Keyspaces Anda. Anda juga dapat menggunakan bagian ini secara independen, misalnya, untuk memigrasikan data ke tabel Amazon Keyspaces. Akhirnya, Anda membaca data dari tabel Anda

ke dalam DataFrame menggunakan Spark Cassandra Connector. Anda juga dapat menggunakan bagian ini secara independen, misalnya, untuk membaca data dari tabel Amazon Keyspaces untuk melakukan analisis data dengan Apache Spark.

1. Mulai Spark Shell seperti yang ditunjukkan pada contoh berikut. Perhatikan bahwa contoh ini menggunakan otentikasi 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. Impor Konektor Spark Cassandra dengan kode berikut.

```
import org.apache.spark.sql.cassandra._
```

3. Untuk membaca data dari file CSV dan menyimpannya di aDataFrame, Anda dapat menggunakan contoh kode berikut.

```
var df =
 spark.read.option("header","true").option("inferSchema","true").csv("keyspaces_sample_table.csv")
```

Anda dapat menampilkan hasilnya dengan perintah berikut.

```
scala> df.show();
```

Outputnya akan terlihat mirip dengan ini.

```
+-----+-----+-----+-----+-----+-----+
+-----+
| 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|
+-----+-----+-----+-----+-----+-----+
+-----+

```

Anda dapat mengkonfirmasi skema data dalam DataFrame seperti yang ditunjukkan pada contoh berikut.

```
scala> df.printSchema
```

Outputnya akan terlihat seperti ini.

```

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)

```

- Gunakan perintah berikut untuk menulis data di DataFrame tabel Amazon Keyspaces.

```
df.write.cassandraFormat("book_awards", "catalog").mode("APPEND").save()
```

- Untuk mengonfirmasi bahwa data telah disimpan, Anda dapat membacanya kembali ke kerangka data, seperti yang ditunjukkan pada contoh berikut.

```
var newDf = spark.read.cassandraFormat("book_awards", "catalog").load()
```

Kemudian Anda dapat menampilkan data yang sekarang terkandung dalam kerangka data.



```
scala> newDf.show()
```

Output dari perintah itu akan terlihat seperti ini.

```
+-----+-----+-----+-----+
+----+----+
| 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|
+-----+-----+-----+-----+
+----+----+
```

## Memecahkan masalah kesalahan umum saat menggunakan Konektor Spark Cassandra dengan Amazon Keyspaces

Jika Anda menggunakan Amazon Virtual Private Cloud dan Anda terhubung ke Amazon Keyspaces, kesalahan paling umum yang dialami saat menggunakan konektor Spark disebabkan oleh masalah konfigurasi berikut.

- Pengguna atau peran IAM yang digunakan dalam VPC tidak memiliki izin yang diperlukan untuk mengakses tabel `system.peers` di Amazon Keyspaces. Untuk informasi selengkapnya, lihat [the section called “Mengisi entri `system.peers` tabel dengan informasi titik akhir VPC antarmuka](#)”.
- Pengguna atau peran IAM tidak memiliki read/write izin yang diperlukan untuk tabel pengguna dan membaca akses ke tabel sistem di Amazon Keyspaces. Untuk informasi selengkapnya, lihat [the section called “Langkah 1: Konfigurasi Amazon Keyspaces](#)”.
- Konfigurasi driver Java tidak menonaktifkan verifikasi nama host saat membuat SSL/TLS koneksi. Sebagai contoh, lihat [the section called “Langkah 2: Konfigurasi driver](#)”.

Untuk langkah-langkah pemecahan masalah koneksi yang mendetail, lihat. [the section called “Kesalahan koneksi titik akhir VPC”](#)

Selain itu, Anda dapat menggunakan CloudWatch metrik Amazon untuk membantu Anda memecahkan masalah dengan konfigurasi Spark Cassandra Connector di Amazon Keyspaces. Untuk mempelajari selengkapnya tentang menggunakan Amazon Keyspaces dengan CloudWatch, lihat. [the section called “Pemantauan CloudWatch dengan”](#)

Bagian berikut menjelaskan metrik yang paling berguna untuk diamati saat Anda menggunakan Konektor Spark Cassandra.

#### PerConnectionRequestRateExceeded

Amazon Keyspaces memiliki kuota 3.000 permintaan per detik, per koneksi. Setiap pelaksana Spark membuat koneksi dengan Amazon Keyspaces. Menjalankan beberapa percobaan ulang dapat menghabiskan kuota tingkat permintaan per koneksi Anda. Jika Anda melebihi kuota ini, Amazon Keyspaces memancarkan `PerConnectionRequestRateExceeded` metrik dalam CloudWatch

Jika Anda melihat `PerConnectionRequestRateExceeded` peristiwa hadir bersama dengan kesalahan sistem atau pengguna lainnya, kemungkinan Spark menjalankan beberapa percobaan ulang di luar jumlah permintaan yang dialokasikan per koneksi.

Jika Anda melihat `PerConnectionRequestRateExceeded` peristiwa tanpa kesalahan lain, maka Anda mungkin perlu meningkatkan jumlah koneksi di pengaturan driver Anda untuk memungkinkan lebih banyak throughput, atau Anda mungkin perlu meningkatkan jumlah pelaksana dalam pekerjaan Spark Anda.

## StoragePartitionThroughputCapacityExceeded

Amazon Keyspaces memiliki kuota 1.000 WCU atau WRU per second/3.000 RCU atau RRU per detik, per partisi. Jika Anda melihat `StoragePartitionThroughputCapacityExceeded` CloudWatch peristiwa, itu bisa menunjukkan bahwa data tidak diacak saat dimuat. Untuk contoh cara mengacak data, lihat [the section called “Langkah 4: Siapkan data sumber dan tabel target”](#).

## Kesalahan dan peringatan umum

Jika Anda menggunakan Amazon Virtual Private Cloud dan Anda terhubung ke Amazon Keyspaces, driver Cassandra mungkin mengeluarkan pesan peringatan tentang node kontrol itu sendiri di `system.peers`. Untuk informasi selengkapnya, lihat [the section called “Kesalahan dan peringatan umum”](#). Anda dapat dengan aman mengabaikan peringatan ini.

## Tutorial: Connect dari aplikasi container yang di-host di Amazon Elastic Kubernetes Service

Tutorial ini memandu Anda melalui langkah-langkah yang diperlukan untuk menyiapkan cluster Amazon Elastic Kubernetes Service (Amazon EKS) untuk meng-host aplikasi container yang terhubung ke Amazon Keyspaces menggunakan otentikasi `SiGv4`.

Amazon EKS adalah layanan terkelola yang menghilangkan kebutuhan untuk menginstal, mengoperasikan, dan memelihara pesawat kontrol Kubernetes Anda sendiri. [Kubernetes](#) adalah sistem sumber terbuka untuk melakukan otomatisasi terhadap deployment, penskalaan, dan pengelolaan aplikasi dalam kontainer.

Tutorial ini menyediakan panduan langkah demi langkah untuk mengonfigurasi, membangun, dan menyebarkan aplikasi Java kontainer ke Amazon EKS. Pada langkah terakhir Anda menjalankan aplikasi untuk menulis data ke tabel Amazon Keyspaces.

### Topik

- [Prasyarat untuk menghubungkan dari Amazon EKS ke Amazon Keyspaces](#)
- [Langkah 1: Konfigurasi cluster Amazon EKS dan setel izin IAM](#)
- [Langkah 2: Konfigurasi aplikasi](#)
- [Langkah 3: Buat gambar aplikasi dan unggah file Docker ke repositori Amazon ECR Anda](#)
- [Langkah 4: Menyebar aplikasi ke Amazon EKS dan menulis data ke tabel Anda](#)
- [Langkah 5: Pembersihan \(Opsional\)](#)

## Prasyarat untuk menghubungkan dari Amazon EKS ke Amazon Keyspaces

Buat AWS sumber daya berikut sebelum Anda dapat memulai dengan tutorial

1. Sebelum Anda memulai tutorial ini, ikuti instruksi AWS pengaturan di [Mengakses Amazon Keyspaces \(untuk Apache Cassandra\)](#). Langkah-langkah ini termasuk mendaftar AWS dan membuat prinsipal AWS Identity and Access Management (IAM) dengan akses ke Amazon Keyspaces.
2. Buat keyspace Amazon Keyspaces dengan nama `aws` dan tabel dengan nama yang dapat Anda tulis dari aplikasi container user yang berjalan di Amazon EKS nanti dalam tutorial ini. Anda dapat melakukan ini baik dengan AWS CLI atau menggunakan `cqlsh`.

### AWS CLI

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

Untuk mengonfirmasi bahwa ruang kunci telah dibuat, Anda dapat menggunakan perintah berikut.

```
aws keyspaces list-keyspaces
```

Untuk membuat tabel, Anda dapat menggunakan perintah berikut.

```
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}]'
```

Untuk mengonfirmasi bahwa tabel Anda telah dibuat, Anda dapat menggunakan perintah berikut.

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

Untuk informasi selengkapnya, lihat [membuat keyspace](#) dan [membuat tabel](#) di AWS CLI Command Reference.

## 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
);
```

Untuk memverifikasi bahwa tabel Anda telah dibuat, Anda dapat menggunakan pernyataan berikut.

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

Tabel Anda harus tercantum dalam output pernyataan ini. Perhatikan bahwa mungkin ada penundaan hingga tabel dibuat. Untuk informasi selengkapnya, lihat [the section called “CREATE TABLE”](#).

3. Buat cluster Amazon EKS dengan tipe simpul Fargate - Linux. Fargate adalah mesin komputasi tanpa server yang memungkinkan Anda menerapkan Pod Kubernetes tanpa mengelola instans Amazon Amazon EC2. Untuk mengikuti tutorial ini tanpa harus memperbarui nama cluster di semua perintah contoh, buat cluster dengan nama `my-eks-cluster` mengikuti petunjuk di [Memulai Amazon EKS — eksctl](#) di Panduan Pengguna Amazon EKS. Saat klaster Anda dibuat, verifikasi bahwa node Anda dan dua Pod default berjalan dan sehat. Anda dapat melakukannya dengan perintah berikut.

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

Anda akan melihat sesuatu yang mirip dengan output ini.

| 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. Pasang Docker. Untuk petunjuk tentang cara menginstal Docker di instans Amazon EC2, [lihat Menginstal Docker](#) di Panduan Pengguna Amazon Elastic Container Registry.

Docker tersedia untuk banyak sistem operasi yang berbeda, termasuk sebagian besar distribusi Linux modern, seperti Ubuntu, dan bahkan macOS dan Windows. Untuk informasi lebih lanjut tentang cara menginstal Docker pada sistem operasi tertentu Anda, kunjungi situs web [panduan penginstalan Docker](#).

5. Buat repositori Amazon ECR. Amazon ECR adalah layanan registri gambar kontainer AWS terkelola yang dapat Anda gunakan dengan CLI pilihan Anda untuk mendorong, menarik, dan mengelola gambar Docker. Untuk informasi selengkapnya tentang repositori Amazon ECR, lihat Panduan Pengguna [Amazon Elastic Container Registry](#). Anda dapat menggunakan perintah berikut untuk membuat repositori dengan nama `my-ecr-repository`

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

Setelah menyelesaikan langkah-langkah prasyarat, lanjutkan ke [the section called “Langkah 1: Konfigurasi cluster Amazon EKS”](#)

## Langkah 1: Konfigurasi cluster Amazon EKS dan setel izin IAM

Konfigurasi kluster Amazon EKS dan buat resource IAM yang diperlukan agar akun layanan Amazon EKS dapat terhubung ke tabel Amazon Keyspaces

1. Buat penyedia Open ID Connect (OIDC) untuk kluster Amazon EKS. Ini diperlukan untuk menggunakan peran IAM untuk akun layanan. Untuk informasi selengkapnya tentang penyedia OIDC dan cara membuatnya, lihat [Membuat penyedia IAM OIDC untuk kluster Anda di](#) Panduan Pengguna Amazon EKS.
  - a. Buat penyedia identitas IAM OIDC untuk kluster Anda dengan perintah berikut. Contoh ini mengasumsikan bahwa nama cluster Anda adalah `my-eks-cluster`. Jika Anda memiliki cluster dengan nama yang berbeda, ingatlah untuk memperbarui nama di semua perintah future.

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

- b. Konfirmasikan bahwa penyedia identitas OIDC telah terdaftar dengan IAM dengan perintah berikut.

```
aws iam list-open-id-connect-providers --region us-east-1
```

Outputnya akan terlihat mirip dengan ini. Perhatikan Nama Sumber Daya Amazon (ARN) OIDC, Anda memerlukannya di langkah berikutnya saat Anda membuat kebijakan kepercayaan untuk akun layanan.

```
{
 "OpenIDConnectProviderList": [
 ..
 {
 "Arn": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
 }
]
}
```

2. Buat akun layanan untuk kluster Amazon EKS. Akun layanan memberikan identitas untuk proses yang berjalan di Pod. Pod adalah objek Kubernetes terkecil dan paling sederhana yang dapat Anda gunakan untuk menyebarkan aplikasi dalam kontainer. Selanjutnya, buat peran IAM yang dapat diasumsikan oleh akun layanan untuk mendapatkan izin ke sumber daya. Anda dapat mengakses AWS layanan apa pun dari Pod yang telah dikonfigurasi untuk menggunakan akun layanan yang dapat mengambil peran IAM dengan izin akses ke layanan tersebut.
  - a. Buat namespace baru untuk akun layanan. Namespace membantu mengisolasi sumber daya cluster yang dibuat untuk tutorial ini. Anda dapat membuat namespace baru menggunakan perintah berikut.

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

- b. Untuk menggunakan namespace khusus, Anda harus mengaitkannya dengan profil Fargate. Kode berikut adalah contoh dari ini.

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

```
--labels *=*
```

- c. Buat akun layanan dengan nama `my-eks-serviceaccount` di namespace `my-eks-namespace` untuk kluster Amazon EKS Anda dengan menggunakan perintah berikut.

```
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. Jalankan perintah berikut untuk membuat file kebijakan kepercayaan yang menginstruksikan peran IAM untuk mempercayai akun layanan Anda. Hubungan kepercayaan ini diperlukan sebelum kepala sekolah dapat mengambil peran. Anda perlu melakukan pengeditan berikut pada file:

- Untuk `Principal`, masukkan ARN yang IAM kembali ke perintah. `list-open-id-connect-providers` ARN berisi nomor akun dan Wilayah Anda.
- Dalam `condition` pernyataan itu, ganti Wilayah AWS dan id OIDC.
- Konfirmasikan bahwa nama akun layanan dan namespace sudah benar.

Anda perlu melampirkan file kebijakan kepercayaan di langkah berikutnya saat Anda membuat peran 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

```

Opsional: Anda juga dapat menambahkan beberapa entri dalam `StringLike` kondisi `StringEquals` atau untuk memungkinkan beberapa akun layanan atau ruang nama untuk mengambil peran. Untuk mengizinkan akun layanan Anda mengambil peran IAM di AWS akun lain, lihat [Cross-account Izin IAM](#) di Panduan Pengguna Amazon EKS.

3. Buat peran IAM dengan nama `my-iam-role` untuk akun layanan Amazon EKS untuk diasumsikan. Lampirkan file kebijakan kepercayaan yang dibuat pada langkah terakhir ke peran. Kebijakan kepercayaan menentukan akun layanan dan penyedia OIDC yang dapat dipercaya oleh peran IAM.

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

4. Tetapkan izin peran IAM ke Amazon Keyspaces dengan melampirkan kebijakan akses.
  - a. Lampirkan kebijakan akses untuk menentukan tindakan yang dapat dilakukan peran IAM pada resource Amazon Keyspaces tertentu. Untuk tutorial ini kami menggunakan kebijakan AWS `terkelolaAmazonKeyspacesFullAccess`, karena aplikasi kami akan menulis data ke tabel Amazon Keyspaces Anda. Namun, sebagai praktik terbaik, disarankan untuk membuat kebijakan akses khusus yang menerapkan prinsip hak istimewa paling sedikit. Untuk informasi selengkapnya, lihat [the section called "Cara Amazon Keyspaces bekerja dengan IAM"](#).

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

Konfirmasikan bahwa kebijakan telah berhasil dilampirkan ke peran IAM dengan pernyataan berikut.

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

Outputnya akan terlihat seperti ini.

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

- b. Beri anotasi akun layanan dengan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dapat diasumsikan. Pastikan untuk memperbarui peran ARN dengan ID akun Anda.

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

5. Konfirmasikan bahwa peran IAM dan akun layanan dikonfigurasi dengan benar.

- a. Konfirmasikan bahwa kebijakan kepercayaan peran IAM telah dikonfigurasi dengan benar dengan pernyataan berikut.

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

Outputnya akan terlihat mirip dengan ini.

```
{
 "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. Konfirmasikan bahwa akun layanan Amazon EKS dianotasi dengan peran IAM.

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

Outputnya akan terlihat mirip dengan ini.

```

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>
[...]

```

Setelah Anda membuat akun layanan Amazon EKS, peran IAM, dan mengonfigurasi hubungan dan izin yang diperlukan, lanjutkan ke [the section called “Langkah 2: Konfigurasikan aplikasi”](#)

## Langkah 2: Konfigurasikan aplikasi

Pada langkah ini Anda membangun aplikasi Anda yang terhubung ke Amazon Keyspaces menggunakan plugin SiGv4. [Anda dapat melihat dan mengunduh contoh aplikasi Java dari repo kode contoh Amazon Keyspaces di Github.](#) Atau Anda dapat mengikuti bersama menggunakan aplikasi Anda sendiri, memastikan untuk menyelesaikan semua langkah konfigurasi.

Konfigurasi aplikasi Anda dan tambahkan dependensi yang diperlukan.

1. Anda dapat mengunduh contoh aplikasi Java dengan mengkloning repositori Github menggunakan perintah berikut.

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

2. Setelah mengunduh repo Github, unzip file yang diunduh dan arahkan ke `resources` direktori ke file `application.conf`

- a. Konfigurasi aplikasi

Pada langkah ini Anda mengonfigurasi plugin otentikasi SiGv4. Anda dapat menggunakan contoh berikut dalam aplikasi Anda. Jika Anda belum melakukannya, Anda perlu membuat kunci akses IAM Anda (ID kunci akses dan kunci akses rahasia) dan menyimpannya di file AWS konfigurasi Anda atau sebagai variabel lingkungan. Untuk petunjuk mendetail, lihat [the section called “Kredensi yang diperlukan untuk otentikasi AWS”](#). Perbarui AWS Wilayah dan titik akhir layanan untuk Amazon Keyspaces sesuai kebutuhan. Untuk titik akhir layanan lainnya, lihat [the section called “Titik akhir layanan”](#). Ganti lokasi truststore, nama truststore, dan kata sandi truststore dengan milik Anda.

```
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. Tambahkan ketergantungan modul STS.

Ini menambahkan kemampuan untuk menggunakan `WebIdentityTokenCredentialsProvider` yang mengembalikan AWS kredensi yang perlu disediakan aplikasi sehingga akun layanan dapat mengambil peran IAM. Anda dapat melakukan ini berdasarkan contoh berikut.

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

c. Tambahkan dependensi SigV4.

Paket ini mengimplementasikan plugin otentikasi SiGv4 yang diperlukan untuk mengautentikasi ke Amazon Keyspaces

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

3. Tambahkan ketergantungan logging.

Tanpa log, pemecahan masalah koneksi tidak mungkin dilakukan. Dalam tutorial ini, kita gunakan `slf4j` sebagai kerangka logging, dan gunakan `logback.xml` untuk menyimpan output log. Kami mengatur level logging debug untuk membuat koneksi. Anda dapat menggunakan contoh berikut untuk menambahkan ketergantungan.

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

Anda dapat menggunakan cuplikan kode berikut untuk mengonfigurasi logging.

```
<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

debugLevel diperlukan untuk menyelidiki kegagalan koneksi. Setelah Anda berhasil terhubung ke Amazon Keyspaces dari aplikasi Anda, Anda dapat mengubah level logging ke info atau warning sesuai kebutuhan.

## Langkah 3: Buat gambar aplikasi dan unggah file Docker ke repositori Amazon ECR Anda

Pada langkah ini, Anda mengkompilasi aplikasi contoh, membangun image Docker, dan mendorong gambar ke repositori Amazon ECR Anda.

Buat aplikasi Anda, buat image Docker, dan kirimkan ke Amazon Elastic Container Registry

1. Tetapkan variabel lingkungan untuk build yang menentukan Anda Wilayah AWS. Ganti Wilayah dalam contoh dengan milik Anda sendiri.

```
export CASSANDRA_HOST=cassandra.us-east-1.amazonaws.com:9142
export CASSANDRA_DC=us-east-1
```

2. Kompilasi aplikasi Anda dengan Apache Maven versi 3.6.3 atau lebih tinggi menggunakan perintah berikut.

```
mvn clean install
```

Ini membuat JAR file dengan semua dependensi yang disertakan dalam direktori. target

3. Ambil URI repositori ECR Anda yang diperlukan untuk langkah berikutnya dengan perintah berikut. Pastikan untuk memperbarui Wilayah ke wilayah yang telah Anda gunakan.

```
aws ecr describe-repositories --region us-east-1
```

Outputnya akan terlihat seperti pada contoh berikut.

```
"repositories": [
 {
 "repositoryArn": "arn:aws:ecr:us-east-1:111122223333:repository/my-ecr-
repository",
 "registryId": "111122223333",
 "repositoryName": "my-ecr-repository",
 "repositoryUri": "111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository",
 "createdAt": "2023-11-02T03:46:34+00:00",
 "imageTagMutability": "MUTABLE",
 "imageScanningConfiguration": {
 "scanOnPush": false
 },
 "encryptionConfiguration": {
 "encryptionType": "AES256"
 }
 },
]
```

4. Dari direktori root aplikasi, buat image Docker menggunakan URI repositori dari langkah terakhir. Ubah file Docker sesuai kebutuhan. Dalam perintah build, pastikan untuk mengganti ID akun Anda dan atur Wilayah AWS ke Wilayah tempat repositori my-ecr-repository Amazon ECR berada.

```
docker build -t 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-
repository:latest .
```

5. Ambil token otentikasi untuk mendorong gambar Docker ke Amazon ECR. Anda dapat melakukannya dengan perintah berikut.

```
aws ecr get-login-password --region us-east-1 | docker login --username AWS --
password-stdin 111122223333.dkr.ecr.us-east-1.amazonaws.com
```

6. Pertama, periksa gambar yang ada di repositori Amazon ECR Anda. Anda dapat menggunakan perintah berikut.

```
aws ecr describe-images --repository-name my-ecr-repository --region us-east-1
```

Kemudian, dorong image Docker ke repo. Anda dapat menggunakan perintah berikut.

```
docker push 111122223333.dkr.ecr.us-east-1.amazonaws.com/my-ecr-repository:latest
```

## Langkah 4: Menyebar aplikasi ke Amazon EKS dan menulis data ke tabel Anda

Pada langkah tutorial ini, Anda mengonfigurasi penyebaran Amazon EKS untuk aplikasi Anda, dan mengonfirmasi bahwa aplikasi sedang berjalan dan dapat terhubung ke Amazon Keyspaces.

Untuk menyebar aplikasi ke Amazon EKS, Anda perlu mengonfigurasi semua pengaturan yang relevan dalam file bernama `deployment.yaml`. File ini kemudian digunakan oleh Amazon EKS untuk menyebar aplikasi. Metadata dalam file harus berisi informasi berikut:

- Nama aplikasi nama aplikasi. Untuk tutorial ini, kita gunakan `my-keyspaces-app`.
- Kubernetes namespace adalah namespace dari kluster Amazon EKS. Untuk tutorial ini, kita gunakan `my-eks-namespace`.
- Nama akun layanan Amazon EKS nama akun layanan Amazon EKS. Untuk tutorial ini, kita gunakan `my-eks-serviceaccount`.
- nama gambar nama gambar aplikasi. Untuk tutorial ini, kita gunakan `my-keyspaces-app`.
- URI gambar URI gambar Docker dari Amazon ECR.
- AWS ID akun ID AWS akun Anda.
- Peran IAM ARN ARN dari peran IAM yang dibuat untuk diasumsikan oleh akun layanan. Untuk tutorial ini, kita gunakan `my-iam-role`.
- Wilayah AWS dari kluster Amazon EKS tempat Wilayah AWS Anda membuat cluster Amazon EKS Anda.

Pada langkah ini, Anda menyebar dan menjalankan aplikasi yang terhubung ke Amazon Keyspaces dan menulis data ke tabel.

1. Konfigurasi file `deployment.yaml`. Anda perlu mengganti nilai-nilai berikut:

- `name`
- `namespace`
- `serviceName`
- `image`



- AWS\_ROLE\_ARN value
- Wilayah AWS Di CASSANDRA\_HOST
- AWS\_REGION

Anda dapat menggunakan file berikut sebagai contoh.

```
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. Menyebarkan deployment.yaml.

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

Outputnya akan terlihat seperti ini.

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

## 3. Periksa status Pod di namespace Anda di kluster Amazon EKS.

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

Outputnya akan terlihat mirip dengan contoh ini.

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

Untuk lebih jelasnya, Anda dapat menggunakan perintah berikut.

```
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

```

- Periksa log Pod untuk mengonfirmasi bahwa aplikasi Anda sedang berjalan dan dapat terhubung ke tabel Amazon Keyspaces Anda. Anda dapat melakukannya dengan perintah berikut. Pastikan untuk mengganti nama penerapan Anda.

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

Anda harus dapat melihat entri log aplikasi yang mengonfirmasi koneksi ke Amazon Keyspaces seperti pada contoh di bawah ini.

```

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.c.core.context.EventBus - [s0] Registering
com.datastax.oss.driver.internal.core.metadata.LoadBalancingPolicyWrapper$$Lambda

```

```
$812/0x0000000801105e88@769afb95 for class
com.datastax.oss.driver.internal.core.metadata.NodeStateEvent
22:47:20.566 [s0-admin-1] DEBUG c.d.o.d.i.c.c.ControlConnection -
[s0] Trying to establish a connection to Node(endPoint=cassandra.us-
east-1.amazonaws.com/1.222.333.44:9142, hostId=null, hashCode=e750d92)
```

5. Jalankan kueri CQL berikut di tabel Amazon Keyspaces Anda untuk mengonfirmasi bahwa satu baris data telah ditulis ke tabel Anda:

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

Anda akan melihat output berikut:

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

## Langkah 5: Pembersihan (Opsional)

Ikuti langkah-langkah ini untuk menghapus semua sumber daya yang dibuat dalam tutorial ini.

Hapus sumber daya yang dibuat dalam tutorial ini

1. Hapus penerapan Anda. Anda dapat menggunakan perintah berikut untuk melakukannya.

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

2. Hapus cluster Amazon EKS dan semua Pod yang ada di dalamnya. Ini juga menghapus sumber daya terkait seperti akun layanan dan penyedia identitas OIDC. Anda dapat menggunakan perintah berikut untuk melakukannya.

```
eksctl delete cluster --name my-eks-cluster --region us-east-1
```

3. Hapus peran IAM yang digunakan untuk akun layanan Amazon EKS dengan izin akses ke Amazon Keyspaces. Pertama, Anda harus menghapus kebijakan terkelola yang dilampirkan pada peran.

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

Kemudian Anda dapat menghapus peran menggunakan perintah berikut.

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

Untuk informasi selengkapnya, lihat [Menghapus peran IAM \(AWS CLI\)](#) di Panduan Pengguna IAM.

4. Hapus repositori Amazon ECR termasuk semua gambar yang tersimpan di dalamnya. Anda dapat melakukannya dengan menggunakan perintah berikut.

```
aws ecr delete-repository \
 --repository-name my-ecr-repository \
 --force \
 --region us-east-1
```

Perhatikan bahwa `force` bendera diperlukan untuk menghapus repositori yang berisi gambar. Untuk menghapus gambar Anda terlebih dahulu, Anda dapat melakukannya menggunakan perintah berikut.

```
aws ecr batch-delete-image \
 --repository-name my-ecr-repository \
 --image-ids imageTag=latest \
 --region us-east-1
```

Untuk informasi selengkapnya, lihat [Menghapus gambar](#) di Panduan Pengguna Amazon Elastic Container Registry.

5. Hapus ruang kunci dan tabel Amazon Keyspaces. Menghapus keyspace secara otomatis menghapus semua tabel di keyspace itu. Anda dapat menggunakan salah satu opsi berikut untuk melakukannya.

## AWS CLI

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

Untuk mengonfirmasi bahwa ruang kunci telah dihapus, Anda dapat menggunakan perintah berikut.

```
aws keyspaces list-keyspaces
```

Untuk menghapus tabel terlebih dahulu, Anda dapat menggunakan perintah berikut.

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

Untuk mengonfirmasi bahwa tabel Anda telah dihapus, Anda dapat menggunakan perintah berikut.

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

Untuk informasi selengkapnya, lihat [menghapus keyspace](#) dan [menghapus tabel](#) di Referensi AWS CLI Perintah.

cqlsh

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

Untuk memverifikasi bahwa ruang kunci Anda telah dihapus, Anda dapat menggunakan pernyataan berikut.

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

Ruang kunci Anda tidak boleh tercantum dalam output pernyataan ini. Perhatikan bahwa mungkin ada penundaan hingga ruang kunci dihapus. Untuk informasi selengkapnya, lihat [the section called “JATUHKAN RUANG KUNCI”](#).

Untuk menghapus tabel terlebih dahulu, Anda dapat menggunakan perintah berikut.

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

Untuk mengonfirmasi bahwa tabel Anda telah dihapus, Anda dapat menggunakan perintah berikut.

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

Tabel Anda tidak boleh tercantum dalam output pernyataan ini. Perhatikan bahwa mungkin ada penundaan hingga tabel dihapus. Lihat informasi yang lebih lengkap di [the section called “MEJA DROP”](#).

# Tutorial: Ekspor tabel Amazon Keyspaces ke Amazon S3 menggunakan AWS Glue

Tutorial ini menunjukkan cara mengekspor tabel Amazon Keyspaces ke bucket Amazon S3 menggunakan AWS Glue. Untuk tutorial ini, banyak langkah manual otomatis menggunakan skrip shell yang tersedia di repo Amazon [Keyspaces Github](#). Dengan menggunakan proses ini, Anda dapat mengekspor data Amazon Keyspaces ke Amazon S3 tanpa harus menyiapkan cluster Spark.

## Topik

- [Prasyarat untuk mengekspor data dari Amazon Keyspaces ke Amazon S3](#)
- [Langkah 1: Buat bucket Amazon S3, unduh alat yang diperlukan, dan konfigurasi lingkungan](#)
- [Langkah 2: Konfigurasi AWS Glue pekerjaan yang mengekspor tabel Amazon Keyspaces](#)
- [Langkah 3: Jalankan AWS Glue pekerjaan untuk mengekspor tabel Amazon Keyspaces ke bucket Amazon S3 dari AWS CLI](#)
- [Langkah 4: \(Opsional\) Buat pemicu untuk menjadwalkan pekerjaan ekspor](#)
- [Langkah 5: Pembersihan \(Opsional\)](#)

## Prasyarat untuk mengekspor data dari Amazon Keyspaces ke Amazon S3

Konfirmasikan prasyarat berikut dan buat sumber daya Amazon Keyspaces sebelum Anda mulai dengan tutorial

1. Sebelum Anda memulai tutorial ini, ikuti instruksi AWS pengaturan di [Mengakses Amazon Keyspaces \(untuk Apache Cassandra\)](#). Langkah-langkah ini termasuk mendaftar AWS dan membuat prinsipal AWS Identity and Access Management (IAM) dengan akses ke Amazon Keyspaces.
2. Skrip dalam tutorial ini menggunakan kredensial Anda dan default Wilayah AWS disimpan di lokasi yang diketahui. Untuk informasi selengkapnya, lihat [the section called “Kelola kunci akses”](#).

Contoh berikut menunjukkan bagaimana untuk menyimpan nilai-nilai yang diperlukan sebagai variabel lingkungan untuk pengguna default.

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
```



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

3. Untuk menjalankan skrip dalam tutorial ini, Anda memerlukan perangkat lunak dan alat berikut yang diinstal pada mesin Anda:

- [Java](#)
- [Apache Maven](#)
- [Git](#)
- [AWS CLI](#)

Tutorial ini diuji dengan AWS CLI 2, Java 17.0.13, dan Apache Maven 3.8.7.

4. Anda memerlukan tabel Amazon Keyspaces dengan data sampel untuk diekspor nanti dalam tutorial ini. Anda dapat menggunakan tabel Amazon Keyspaces Anda sendiri atau membuat tabel sampel mengikuti langkah-langkah dalam tutorial. [Memulai](#)
- Untuk menginstal `cqlsh-expansion`, ikuti langkah-langkah di [the section called “Menggunakan cqlsh-expansion”](#).
  - Konfirmasikan bahwa `Murmur3Partitioner` partisi adalah partisi default untuk akun Anda. Partisi ini kompatibel dengan Apache Spark Cassandra Connector dan dengan AWS Glue Untuk informasi lebih lanjut tentang partisi, lihat. [the section called “Bekerja dengan partisi”](#)

Untuk mengubah partisi akun Anda, Anda dapat menggunakan pernyataan berikut.

```
SELECT partitioner FROM system.local;

UPDATE system.local set
 partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

- Untuk membuat keyspace Amazon Keyspaces, ikuti langkah-langkah di [the section called “Buat ruang kunci”](#)
- Untuk membuat tabel Amazon Keyspaces, ikuti langkah-langkah di [the section called “Membuat tabel”](#)
- Untuk memuat data sampel ke dalam tabel untuk mengekspor ke Amazon S3, ikuti langkah-langkah di [the section called “Buat”](#)

Setelah menyelesaikan langkah-langkah prasyarat, lanjutkan ke [the section called “Langkah 1: Buat bucket Amazon S3, unduh alat, dan konfigurasi lingkungan”](#)

## Langkah 1: Buat bucket Amazon S3, unduh alat yang diperlukan, dan konfigurasi lingkungan

Pada langkah ini, Anda mengunduh alat eksternal dan membuat serta mengonfigurasi AWS sumber daya yang diperlukan untuk solusi ekspor data otomatis tabel Amazon Keyspaces ke bucket Amazon S3 menggunakan pekerjaan. AWS Glue Untuk melakukan semua tugas ini dengan cara yang efisien, kami menjalankan skrip shell dengan nama yang `setup-connector.sh` tersedia di [Github](#).

Script `setup-connector.sh` mengotomatiskan langkah-langkah berikut.

1. Membuat bucket Amazon S3 menggunakan CloudFormation Bucket ini menyimpan jar yang diunduh dan file konfigurasi, serta data tabel yang diekspor.
2. Membuat peran IAM menggunakan CloudFormation. AWS Glue pekerjaan menggunakan peran ini untuk mengakses Amazon Keyspaces dan Amazon S3.
3. Unduh [Apache Spark Cassandra Connector](#) dan unggah ke bucket Amazon S3.
4. Unduh [plugin Otentikasi SiGv4](#) dan unggah ke bucket Amazon S3.
5. Unduh [Apache Spark Extensions](#) dan unggah ke bucket Amazon S3.
6. Mengunduh [Kebijakan Coba Kembali Keyspaces](#) dari Github, mengkompilasi kode menggunakan Maven, dan mengunggah output ke bucket Amazon S3.
7. Mengunggah **keyspaces-application.conf** ke bucket Amazon S3.

Gunakan skrip **setup-connector.sh** shell untuk mengotomatiskan langkah-langkah pengaturan dan konfigurasi.

1. Salin file dari repositori [aws-glue](#) di Github ke mesin lokal Anda. Direktori ini berisi skrip shell serta file lain yang diperlukan.
2. Jalankan skrip shell `setup-connector.sh`. Anda dapat menentukan tiga parameter opsional berikut.
  1. **SETUP\_STACKNAME**— Ini adalah nama CloudFormation tumpukan yang digunakan untuk membuat AWS sumber daya.
  2. **S3\_BUCKET\_NAME**- Ini adalah nama ember Amazon S3.

3. `GLUE_SERVICE_ROLE_NAME`— Ini adalah nama peran layanan IAM yang AWS Glue digunakan untuk menjalankan pekerjaan yang terhubung ke Amazon Keyspaces dan Amazon S3.

Anda dapat menggunakan perintah berikut untuk menjalankan skrip shell, memberikan tiga parameter dengan nama-nama berikut.

```
./setup-connector.sh cfn-setup s3-keyspaces iam-export-role
```

Untuk mengonfirmasi bahwa bucket Anda telah dibuat, Anda dapat menggunakan AWS CLI perintah berikut.

```
aws s3 ls s3://s3-keyspaces
```

Output dari perintah akan terlihat seperti ini.

```
PRE conf/
PRE jars/
```

Untuk mengonfirmasi bahwa peran IAM telah dibuat dan untuk meninjau detailnya, Anda dapat menggunakan AWS CLI pernyataan berikut.

```
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"
 }
}

```

Jika proses CloudFormation tumpukan gagal, Anda dapat meninjau informasi kesalahan terperinci tentang tumpukan yang gagal di CloudFormation konsol.

Setelah bucket Amazon S3 yang berisi semua skrip dan alat telah dibuat dan peran IAM dikonfigurasi, lanjutkan ke [the section called “Langkah 2: Konfigurasi AWS Glue pekerjaan”](#)

## Langkah 2: Konfigurasi AWS Glue pekerjaan yang mengeksport tabel Amazon Keyspaces

Pada langkah kedua tutorial Anda menggunakan skrip yang `setup-export.sh` tersedia di [Github](#) untuk membuat dan mengonfigurasi AWS Glue pekerjaan yang terhubung ke Amazon Keyspaces menggunakan plugin SiGv4 dan kemudian mengeksport tabel yang ditentukan ke bucket Amazon S3 Anda yang dibuat pada langkah sebelumnya. Menggunakan skrip memungkinkan Anda untuk mengeksport data dari Amazon Keyspaces tanpa menyiapkan cluster Apache Spark.

Buat AWS Glue pekerjaan untuk mengeksport tabel Amazon Keyspaces ke bucket Amazon S3.

- Pada langkah ini, Anda menjalankan skrip `setup-export.sh` shell yang terletak di `export-to-s3/` direktori yang akan digunakan CloudFormation untuk membuat dan mengonfigurasi pekerjaan AWS Glue ekspor. Script mengambil parameter berikut.

```
PARENT_STACK_NAME, EXPORT_STACK_NAME, KEYSPACE_NAME, TABLE_NAME, S3_URI, FORMAT
```

- **PARENT\_STACK\_NAME**— Nama CloudFormation tumpukan yang dibuat pada langkah sebelumnya.
- **EXPORT\_STACK\_NAME**— Nama CloudFormation tumpukan yang menciptakan pekerjaan AWS Glue ekspor.
- **KEYSPACE\_NAME** dan **TABLE\_NAME** — Nama keyspace dan tabel yang sepenuhnya memenuhi syarat untuk diekspor. Untuk tutorial ini, kami menggunakan `catalog.book_awards`, tetapi Anda dapat menggantinya dengan nama tabel Anda sendiri yang sepenuhnya memenuhi syarat.
- **S3URI**— URI opsional dari bucket Amazon S3. Defaultnya adalah bucket Amazon S3 dari tumpukan induk.
- **FORMAT**— Format data opsional. Nilai default-nya adalah `parquet`. Untuk tutorial ini, untuk membuat pemuatan data dan transformasi lebih mudah, kami menggunakan default.

Anda dapat menggunakan perintah berikut sebagai contoh.

```
setup-export.sh cfn-setup cfn-glue catalog book_awards
```

Untuk mengonfirmasi bahwa pekerjaan telah dibuat, Anda dapat menggunakan pernyataan berikut.

```
aws glue list-jobs
```

Output dari pernyataan harus terlihat mirip dengan ini.

```
{
 "JobNames": [
 "AmazonKeyspacesExportToS3-cfn-setup-cfn-glue"
]
}
```

Untuk melihat detail pekerjaan, Anda dapat menggunakan perintah berikut.

```
aws glue get-job --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue
```

Output dari perintah menunjukkan semua detail pekerjaan. Ini termasuk argumen default yang dapat Anda timpa saat menjalankan pekerjaan.

```
{
 "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"
}
}
```

Jika proses CloudFormation tumpukan gagal, Anda dapat meninjau kesalahan untuk tumpukan yang gagal di CloudFormation konsol. Anda dapat meninjau detail pekerjaan ekspor di AWS Glue konsol dengan memilih pekerjaan ETL di menu sisi kiri.

Setelah Anda mengonfirmasi detail pekerjaan AWS Glue ekspor, lanjutkan [the section called “Langkah 3: Jalankan AWS Glue pekerjaan ekspor dari AWS CLI”](#) untuk menjalankan pekerjaan untuk mengekspor data dari tabel Amazon Keyspaces Anda.

## Langkah 3: Jalankan AWS Glue pekerjaan untuk mengekspor tabel Amazon Keyspaces ke bucket Amazon S3 dari AWS CLI

Pada langkah ini, Anda menggunakan AWS CLI untuk menjalankan AWS Glue pekerjaan yang dibuat pada langkah sebelumnya untuk mengekspor tabel Amazon Keyspaces ke bucket Anda di Amazon S3.

Jalankan pekerjaan ekspor dari AWS CLI

1. Dalam contoh berikut, AWS CLI perintah menjalankan pekerjaan yang dibuat pada langkah sebelumnya.

```
aws glue start-job-run --job-name AmazonKeyspacesExportToS3-cfn-setup-cfn-glue
```

- Anda dapat mengganti salah satu parameter AWS Glue pekerjaan termasuk argumen default dalam AWS CLI perintah. Untuk mengganti argumen default dari pekerjaan, misalnya keyspace atau nama tabel, Anda dapat meneruskannya sebagai argumen. Untuk daftar lengkap argumen, lihat [start-job-run](#) di Command Line Reference. AWS Glue

Perintah berikut menjalankan pekerjaan AWS Glue ekspor, tetapi mengganti jumlah pekerja, jenis AWS Glue pekerja, dan nama tabel.

```
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. Konfirmasikan bahwa tabel Anda telah diekspor ke bucket Amazon S3 Anda. Berdasarkan ukuran tabel, ini bisa memakan waktu. Ketika pekerjaan ekspor selesai, Anda dapat melihat folder berikut di bucket menggunakan perintah example.

```
aws s3 ls s3://s3-keyspaces
```

Outputnya menunjukkan struktur berikut di bucket Anda.

```
PRE conf/
PRE export/
PRE jars/
PRE scripts/
PRE spark-logs/
```

File Anda akan berada di struktur folder berikut di bawah `export`, data/time nilai akan menunjukkan nilai Anda sendiri.

```
\----- export
 \----- keyspace_name
 \----- table_name
 \----- snapshot
 \----- year=2025
 \----- month=01
 \----- day=02
 \----- hour=09
 \----- minute=22
 \--- YOUR DATA HERE
```

Untuk menjadwalkan AWS Glue pekerjaan yang baru saja Anda jalankan secara manual, lanjutkan ke [the section called “Langkah 4: \(Opsional\) Jadwalkan pekerjaan ekspor”](#).



## Langkah 4: (Opsional) Buat pemacu untuk menjadwalkan pekerjaan ekspor

Untuk menjalankan pekerjaan ekspor yang dibuat pada langkah sebelumnya secara teratur, Anda dapat membuat pemacu terjadwal. Untuk informasi selengkapnya, lihat [AWS Glue pemacu](#) di Panduan AWS Glue Pengembang.

### Jadwalkan AWS Glue pekerjaan

1. AWS CLI Perintah berikut adalah contoh pemacu sederhana dengan nama `KeyspacesExportWeeklyTrigger` yang menjalankan AWS Glue pekerjaan dengan nama seminggu `AmazonKeyspacesExportToS3-cfn-setup-cfn-glue` sekali pada hari Senin pukul 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"
]'
```

- Untuk mengganti salah satu pengaturan default dari pekerjaan terjadwal, Anda dapat meneruskannya sebagai argumen. Dalam contoh ini kita mengganti nama keyspace, nama tabel, jumlah pekerja, dan tipe pekerja dengan meneruskannya sebagai argumen. Perintah berikut adalah contohnya.

```
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. Untuk mengonfirmasi bahwa pemicu telah dibuat, gunakan perintah berikut.

```
aws glue list-triggers
```

Output dari perintah akan terlihat mirip dengan ini.

```
{
 "TriggerNames": [
 "KeyspacesExportWeeklyTrigger"
]
}
```

Untuk membersihkan AWS sumber daya yang dibuat dalam tutorial ini, lanjutkan ke [the section called “Langkah 5: Pembersihan \(Opsional\)”](#).

## Langkah 5: Pembersihan (Opsional)

Ikuti langkah-langkah ini untuk menghapus semua AWS sumber daya yang dibuat dalam tutorial ini.

Hapus sumber daya yang dibuat dalam tutorial ini

1. Hapus CloudFormation tumpukan kedua yang dibuat dalam tutorial ini. Ini menghapus AWS Glue pekerjaan dan pemicu yang dibuat dalam tutorial ini. Anda dapat menggunakan perintah berikut untuk melakukan ini.

```
aws cloudformation delete-stack --stack-name cfn-glue
```

2. Hapus bucket Amazon S3 bersama dengan semua data yang tersimpan di dalamnya. Anda dapat menggunakan perintah berikut untuk melakukannya.

```
aws s3 rm s3://s3-keyspaces --recursive
```

3. Hapus tumpukan pertama yang dibuat dalam tutorial ini. Ini menghapus peran IAM dan izin terkait yang dibuat dalam tutorial ini. Anda dapat menggunakan perintah berikut sebagai contoh.

```
aws cloudformation delete-stack --stack-name cfn-setup
```

4. Hapus ruang kunci dan tabel Amazon Keyspaces. Menghapus keyspace secara otomatis menghapus semua tabel di keyspace itu. Anda dapat menggunakan salah satu opsi berikut untuk melakukannya.

#### AWS CLI

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

Untuk mengonfirmasi bahwa ruang kunci telah dihapus, Anda dapat menggunakan perintah berikut.

```
aws keyspaces list-keyspaces
```

Untuk menghapus tabel terlebih dahulu, Anda dapat menggunakan perintah berikut.

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

Untuk mengonfirmasi bahwa tabel Anda telah dihapus, Anda dapat menggunakan perintah berikut.

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

Untuk informasi selengkapnya, lihat [menghapus keyspace](#) dan [menghapus tabel](#) di Referensi AWS CLI Perintah.

#### cqlsh

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

Untuk memverifikasi bahwa ruang kunci Anda telah dihapus, Anda dapat menggunakan pernyataan berikut.

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

Ruang kunci Anda tidak boleh tercantum dalam output pernyataan ini. Perhatikan bahwa mungkin ada penundaan hingga ruang kunci dihapus. Untuk informasi selengkapnya, lihat [the section called “JATUHKAN RUANG KUNCI”](#).

Untuk menghapus tabel terlebih dahulu, Anda dapat menggunakan perintah berikut.

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

Untuk mengonfirmasi bahwa tabel Anda telah dihapus, Anda dapat menggunakan perintah berikut.

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

Tabel Anda tidak boleh tercantum dalam output pernyataan ini. Perhatikan bahwa mungkin ada penundaan hingga tabel dihapus. Lihat informasi yang lebih lengkap di [the section called “MEJA DROP”](#).

# Mengelola sumber daya tanpa server di Amazon Keyspaces (untuk Apache Cassandra)

Amazon Keyspaces (untuk Apache Cassandra) bersifat nirserver. Alih-alih menerapkan, mengelola, dan memelihara penyimpanan dan menghitung sumber daya untuk beban kerja Anda melalui node dalam kluster, Amazon Keyspaces mengalokasikan sumber daya penyimpanan dan throughput langsung ke tabel. `read/write`

Amazon Keyspaces menyediakan penyimpanan secara otomatis berdasarkan data yang disimpan dalam tabel Anda. Ini menskalakan penyimpanan naik dan turun saat Anda menulis, memperbarui, dan menghapus data, dan Anda hanya membayar untuk penyimpanan yang Anda gunakan. Data direplikasi di beberapa [Availability Zone](#) untuk ketersediaan tinggi. Amazon Keyspaces memonitor ukuran tabel Anda secara terus menerus untuk menentukan biaya penyimpanan Anda. Untuk informasi selengkapnya tentang cara Amazon Keyspaces menghitung ukuran data yang dapat ditagih, lihat [the section called “Perkiraan ukuran baris”](#)

Bab ini mencakup aspek-aspek kunci manajemen sumber daya di Amazon Keyspaces.

- **Perkiraan ukuran baris** — Untuk memperkirakan ukuran baris yang dikodekan di Amazon Keyspaces, pertimbangkan faktor-faktor seperti metadata kunci partisi, metadata kolom pengelompokan, pengidentifikasi kolom, tipe data, dan metadata baris. Ukuran baris yang dikodekan ini digunakan untuk penagihan, manajemen kuota, dan perencanaan kapasitas throughput yang disediakan.
- **Perkiraan konsumsi kapasitas** — Bagian ini mencakup contoh cara memperkirakan konsumsi kapasitas baca dan tulis untuk skenario umum seperti kueri rentang, kueri batas, pemindaian tabel, transaksi ringan, kolom statis, dan tabel Multi-wilayah. Anda dapat menggunakan Amazon CloudWatch untuk memantau pemanfaatan kapasitas aktual. Untuk informasi lebih lanjut tentang pemantauan dengan CloudWatch, lihat [the section called “Pemantauan CloudWatch dengan”](#).
- **Mengkonfigurasi mode `read/write` kapasitas** - Anda dapat memilih antara dua mode kapasitas untuk memproses membaca dan menulis di tabel Anda:
  - **On-demand mode (default)** - Bayar per permintaan untuk throughput baca dan tulis. Amazon Keyspaces dapat secara instan meningkatkan kapasitas hingga tingkat lalu lintas yang dicapai sebelumnya.
  - **Mode yang disediakan** - Tentukan jumlah unit kapasitas baca dan tulis yang diperlukan sebelumnya. Mode ini membantu mempertahankan kinerja throughput yang dapat diprediksi.

- Kelola kapasitas throughput dengan penskalaan otomatis — Untuk tabel yang disediakan, Anda dapat mengaktifkan penskalaan otomatis untuk menyesuaikan kapasitas throughput secara otomatis berdasarkan lalu lintas aplikasi aktual. Amazon Keyspaces menggunakan pelacakan target untuk menambah atau mengurangi kapasitas yang disediakan, menjaga pemanfaatan pada target yang Anda tentukan.
- Gunakan kapasitas burst secara efektif — Amazon Keyspaces menyediakan kapasitas burst dengan memesan sebagian throughput yang tidak digunakan untuk menangani lonjakan lalu lintas. Fleksibilitas ini memungkinkan ledakan aktivitas sesekali di luar throughput yang Anda berikan.

Untuk memecahkan masalah kesalahan kapasitas, lihat. [the section called “Kesalahan kapasitas tanpa server”](#)

#### Topik

- [Perkirakan ukuran baris di Amazon Keyspaces](#)
- [Perkirakan konsumsi kapasitas throughput baca dan tulis di Amazon Keyspaces](#)
- [Konfigurasi mode read/write kapasitas di Amazon Keyspaces](#)
- [Kelola kapasitas throughput secara otomatis dengan penskalaan otomatis Amazon Keyspaces](#)
- [Gunakan kapasitas burst secara efektif di Amazon Keyspaces](#)

## Perkirakan ukuran baris di Amazon Keyspaces

Amazon Keyspaces menyediakan penyimpanan terkelola penuh yang menawarkan kinerja baca dan tulis milidetik satu digit dan menyimpan data secara tahan lama di beberapa Availability Zone. AWS Amazon Keyspaces melampirkan metadata ke semua baris dan kolom kunci utama untuk mendukung akses data yang efisien dan ketersediaan tinggi.

Topik ini memberikan detail tentang cara memperkirakan ukuran baris yang dikodekan di Amazon Keyspaces. Ukuran baris yang dikodekan digunakan saat menghitung tagihan dan penggunaan kuota Anda. Anda juga dapat menggunakan ukuran baris yang dikodekan saat memperkirakan persyaratan kapasitas throughput yang disediakan untuk tabel.

Untuk menghitung ukuran baris yang dikodekan di Amazon Keyspaces, Anda dapat menggunakan panduan berikut.

#### Topik

- [Perkirakan ukuran kolom yang dikodekan](#)

- [Perkirakan ukuran nilai data yang dikodekan berdasarkan tipe data](#)
- [Pertimbangkan dampak fitur Amazon Keyspaces pada ukuran baris](#)
- [Pilih rumus yang tepat untuk menghitung ukuran baris yang dikodekan](#)
- [Contoh perhitungan ukuran baris](#)

## Perkirakan ukuran kolom yang dikodekan

Bagian ini menunjukkan cara memperkirakan ukuran kolom yang dikodekan di Amazon Keyspaces.

- Kolom reguler — Untuk kolom reguler, yang merupakan kolom yang bukan kunci utama, kolom pengelompokan, atau STATIC kolom, gunakan ukuran mentah data sel berdasarkan [tipe data](#) dan tambahkan metadata yang diperlukan. Tipe data dan beberapa perbedaan utama dalam cara Amazon Keyspaces menyimpan nilai tipe data dan metadata tercantum di bagian berikutnya.
- Kolom kunci partisi - Tombol partisi dapat berisi hingga 2048 byte data. Setiap kolom kunci dalam kunci partisi membutuhkan hingga 3 byte metadata. Saat menghitung ukuran baris Anda, Anda harus menganggap setiap kolom kunci partisi menggunakan metadata 3 byte penuh.
- Kolom pengelompokan - Kolom pengelompokan dapat menyimpan hingga 850 byte data. Selain ukuran nilai data, setiap kolom pengelompokan membutuhkan hingga 20% dari ukuran nilai data untuk metadata. Saat menghitung ukuran baris Anda, Anda harus menambahkan 1 byte metadata untuk setiap 5 byte nilai data kolom pengelompokan.

### Note

Untuk mendukung kueri yang efisien dan pengindeksan bawaan, Amazon Keyspaces menyimpan nilai data dari setiap kunci partisi dan kolom kunci pengelompokan dua kali.

- Nama kolom - Ruang yang diperlukan untuk setiap nama kolom disimpan menggunakan pengidentifikasi kolom dan ditambahkan ke setiap nilai data yang disimpan di kolom. Nilai penyimpanan pengenalan kolom tergantung pada jumlah keseluruhan kolom dalam tabel Anda:
  - 1—62 kolom: 1 byte
  - 63—124 kolom: 2 byte
  - 125—186 kolom: 3 byte

Untuk setiap tambahan 62 kolom tambahkan 1 byte. Perhatikan bahwa di Amazon Keyspaces, hingga 225 kolom reguler dapat dimodifikasi dengan satu INSERT atau pernyataan. UPDATE Untuk informasi selengkapnya, lihat [the section called “Kuota layanan Amazon Keyspaces”](#).

## Perkirakan ukuran nilai data yang dikodekan berdasarkan tipe data

Bagian ini menunjukkan cara memperkirakan ukuran yang dikodekan dari berbagai tipe data di Amazon Keyspaces.

- Jenis string — Cassandra `ASCII`, `TEXT`, dan tipe data `VARCHAR` string semuanya disimpan di Amazon Keyspaces menggunakan Unicode dengan pengkodean biner. UTF-8 Ukuran string di Amazon Keyspaces sama dengan jumlah byte yang dikodekan. UTF-8
- Tipe numerik — Cassandra `INT`, `BIGINT`, `SMALLINT`, `TINYINT`, dan tipe `VARINT` data disimpan di Amazon Keyspaces sebagai nilai data dengan panjang variabel, dengan hingga 38 digit signifikan. Angka nol di depan dan di belakang dipangkas. Ukuran salah satu tipe data ini kira-kira 1 byte per dua digit signifikan+1 byte.
- Tipe gumpalan — A `BLOB` di Amazon Keyspaces disimpan dengan panjang byte mentah nilai.
- Jenis Boolean — Ukuran nilai atau Boolean `Null` nilai adalah 1 byte.
- Tipe koleksi — Kolom yang menyimpan tipe data koleksi seperti `LIST` atau `MAP` membutuhkan 3 byte metadata, terlepas dari isinya. Ukuran a `LIST` or `MAP` is (id kolom) + jumlah (ukuran elemen bersarang) + (3 byte). Ukuran kosong `LIST` atau `MAP` adalah (kolom id) + (3 byte). Setiap individu `LIST` atau `MAP` elemen juga membutuhkan 1 byte metadata.
- User-defined tipe — [Tipe yang ditentukan pengguna \(UDT\)](#) membutuhkan 3 byte untuk metadata, terlepas dari isinya. Untuk setiap elemen UDT, Amazon Keyspaces memerlukan tambahan 1 byte metadata.

Untuk menghitung ukuran UDT yang dikodekan, mulailah dengan `field name` dan `field value` untuk bidang UDT:

- nama bidang - Setiap nama bidang UDT tingkat atas disimpan menggunakan pengenal. Nilai penyimpanan pengenal tergantung pada jumlah keseluruhan bidang di UDT tingkat atas Anda, dan dapat bervariasi antara 1 dan 3 byte:
  - 1—62 bidang: 1 byte
  - 63—124 bidang: 2 byte
  - 125— bidang maks: 3 byte
- nilai bidang - Byte yang diperlukan untuk menyimpan nilai bidang UDT tingkat atas bergantung pada tipe data yang disimpan:
  - Jenis data skalar — Byte yang diperlukan untuk penyimpanan sama dengan tipe data yang sama yang disimpan dalam kolom biasa.



- UDT Beku — Untuk setiap UDT bersarang beku, UDT bersarang memiliki ukuran yang sama seperti pada protokol biner CQL. Untuk UDT bersarang, 4 byte disimpan untuk setiap bidang (termasuk bidang kosong) dan nilai bidang yang disimpan adalah format serialisasi protokol biner CQL dari nilai bidang.
- Koleksi beku:
  - LIST dan SET — Untuk frozen bersarang LIST atau SET, 4 byte disimpan untuk setiap elemen koleksi ditambah format serialisasi protokol biner CQL dari nilai koleksi.
  - MAP - Untuk beku bersarang MAP, setiap pasangan nilai kunci memiliki persyaratan penyimpanan berikut:
    - Untuk setiap kunci mengalokasikan 4 byte, lalu tambahkan format serialisasi protokol biner CQL dari kunci.
    - Untuk setiap nilai mengalokasikan 4 byte, kemudian tambahkan format serialisasi protokol biner CQL dari nilai.
- Kata kunci FROZEN — Untuk koleksi beku yang bersarang di dalam koleksi beku, Amazon Keyspaces tidak memerlukan byte tambahan untuk meta data.
- Kata kunci STATIS - data STATIC kolom tidak dihitung terhadap ukuran baris maksimum 1 MB. Untuk menghitung ukuran data kolom statis, lihat [the section called “Hitung ukuran kolom statis per partisi logis”](#).

## Pertimbangkan dampak fitur Amazon Keyspaces pada ukuran baris

Bagian ini menunjukkan bagaimana fitur di Amazon Keyspaces memengaruhi ukuran baris yang dikodekan.

- Client-side stempel waktu — Client-side stempel waktu disimpan untuk setiap kolom di setiap baris saat fitur dihidupkan. Stempel waktu ini memakan waktu sekitar 20-40 byte (tergantung pada data Anda), dan berkontribusi pada biaya penyimpanan dan throughput untuk baris tersebut. Untuk informasi selengkapnya tentang stempel waktu sisi klien, lihat. [the section called “Client-side stempel waktu”](#)
- Time to Live (TTL) — Metadata TTL memakan waktu sekitar 8 byte untuk satu baris saat fitur dihidupkan. Selain itu, metadata TTL disimpan untuk setiap kolom dari setiap baris. Metadata TTL membutuhkan sekitar 8 byte untuk setiap kolom yang menyimpan tipe data skalar atau koleksi beku. Jika kolom menyimpan tipe data koleksi yang tidak dibekukan, untuk setiap elemen koleksi TTL memerlukan sekitar 8 byte tambahan untuk metadata. Untuk kolom yang menyimpan tipe data koleksi saat TTL diaktifkan, Anda dapat menggunakan rumus berikut.

```
total encoded size of column = (column id) + sum (nested elements + collection
metadata (1 byte) + TTL metadata (8 bytes)) + collection column metadata (3 bytes)
```

Metadata TTL berkontribusi pada biaya penyimpanan dan throughput untuk baris. Untuk informasi lebih lanjut tentang TTL, lihat [the section called “Data kedaluwarsa dengan Time to Live”](#).

## Pilih rumus yang tepat untuk menghitung ukuran baris yang dikodekan

Bagian ini menunjukkan rumus berbeda yang dapat Anda gunakan untuk memperkirakan penyimpanan atau persyaratan throughput kapasitas untuk deretan data di Amazon Keyspaces.

Ukuran total deretan data yang dikodekan dapat diperkirakan berdasarkan salah satu rumus berikut, berdasarkan tujuan Anda:

- Kapasitas throughput — Untuk memperkirakan ukuran baris yang dikodekan untuk menilai unit read/write permintaan yang diperlukan (RRUs/WRUs) atau unit read/write kapasitas (): RCUs/WCUs

```
total encoded size of row = partition key columns + clustering columns + regular
columns
```

- Ukuran penyimpanan — Untuk memperkirakan ukuran baris yang dikodekan untuk memprediksi `BillableTableSizeInBytes`, tambahkan metadata yang diperlukan untuk penyimpanan baris:

```
total encoded size of row = partition key columns + clustering columns + regular
columns + row metadata (100 bytes)
```

### Important

Semua metadata kolom, misalnya id kolom, metadata kunci partisi, metadata kolom pengelompokan, serta stempel waktu sisi klien, TTL, dan metadata baris dihitung menuju ukuran baris maksimum 1 MB.

## Contoh perhitungan ukuran baris

Perhatikan contoh berikut dari tabel di mana semua kolom adalah tipe integer. Tabel ini memiliki dua kolom kunci partisi, dua kolom pengelompokan, dan satu kolom reguler. Karena tabel ini memiliki lima kolom, ruang yang diperlukan untuk pengenalan nama kolom adalah 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));
```

Dalam contoh ini, kita menghitung ukuran data ketika kita menulis baris ke tabel seperti yang ditunjukkan dalam pernyataan berikut:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1)
values(1,2,3,4,5);
```

Untuk memperkirakan total byte yang diperlukan oleh operasi penulisan ini, Anda dapat menggunakan langkah-langkah berikut.

1. Hitung ukuran kolom kunci partisi dengan menambahkan byte untuk tipe data yang disimpan di kolom dan byte metadata. Ulangi ini untuk semua kolom kunci partisi.

- a. Hitung ukuran kolom pertama dari kunci partisi (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. Hitung ukuran kolom kedua dari kunci partisi (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. Tambahkan kedua kolom untuk mendapatkan ukuran estimasi total kolom kunci partisi:

```
8 bytes + 8 bytes = 16 bytes for the partition key columns
```

2. Hitung ukuran kolom pengelompokan dengan menambahkan byte untuk tipe data yang disimpan di kolom dan byte metadata. Ulangi ini untuk semua kolom pengelompokan.

- a. Hitung ukuran kolom pertama kolom pengelompokan (ck\_col1):

$(2 \text{ bytes for the integer data type}) \times 2 + 20\% \text{ of the data value (2 bytes) for clustering column metadata} + 1 \text{ byte for the column id} = 6 \text{ bytes}$

- b. Hitung ukuran kolom kedua dari kolom pengelompokan (ck\_col2):

$(2 \text{ bytes for the integer data type}) \times 2 + 20\% \text{ of the data value (2 bytes) for clustering column metadata} + 1 \text{ byte for the column id} = 6 \text{ bytes}$

- c. Tambahkan kedua kolom untuk mendapatkan perkiraan ukuran total kolom pengelompokan:

$6 \text{ bytes} + 6 \text{ bytes} = 12 \text{ bytes for the clustering columns}$

3. Tambahkan ukuran kolom reguler. Dalam contoh ini kita hanya memiliki satu kolom yang menyimpan integer digit tunggal, yang membutuhkan 2 byte dengan 1 byte untuk id kolom.
4. Akhirnya, untuk mendapatkan total ukuran baris yang dikodekan, tambahkan byte untuk semua kolom. Untuk memperkirakan ukuran penyimpanan yang dapat ditagih, tambahkan 100 byte tambahan untuk metadata baris:

$16 \text{ bytes for the partition key columns} + 12 \text{ bytes for clustering columns} + 3 \text{ bytes for the regular column} + 100 \text{ bytes for row metadata} = 131 \text{ bytes.}$

Untuk mempelajari cara memantau sumber daya tanpa server dengan Amazon CloudWatch, lihat [the section called “Pemantauan CloudWatch dengan”](#)

## Perkirakan konsumsi kapasitas throughput baca dan tulis di Amazon Keyspaces

Saat Anda membaca atau menulis data di Amazon Keyspaces, jumlah unit read/write permintaan (RRUs/WRUs) atau unit read/write kapasitas ( ) yang dikonsumsi kueri Anda bergantung pada jumlah total data yang harus diproses Amazon Keyspaces untuk menjalankan kueri. RCUs/WCUs Dalam beberapa kasus, data yang dikembalikan ke klien dapat menjadi bagian dari data yang harus dibaca Amazon Keyspaces untuk memproses kueri. Untuk penulisan bersyarat, Amazon Keyspaces menggunakan kapasitas tulis meskipun pemeriksaan bersyarat gagal.

Untuk memperkirakan jumlah total data yang sedang diproses untuk permintaan, Anda harus mempertimbangkan ukuran baris yang dikodekan dan jumlah total baris. Topik ini mencakup beberapa contoh skenario umum dan pola akses untuk menunjukkan bagaimana Amazon Keyspaces

memproses kueri dan bagaimana hal itu memengaruhi konsumsi kapasitas. Anda dapat mengikuti contoh untuk memperkirakan persyaratan kapasitas tabel Anda dan menggunakan Amazon CloudWatch untuk mengamati konsumsi kapasitas baca dan tulis untuk kasus penggunaan ini.

Untuk informasi tentang cara menghitung ukuran baris yang dikodekan di Amazon Keyspaces, lihat [the section called “Perkiraan ukuran baris”](#)

## Topik

- [Perkiraan konsumsi kapasitas kueri rentang di Amazon Keyspaces](#)
- [Perkiraan konsumsi kapasitas baca kueri batas](#)
- [Perkiraan konsumsi kapasitas baca pemindaian tabel](#)
- [Perkiraan konsumsi kapasitas transaksi ringan di Amazon Keyspaces](#)
- [Perkiraan konsumsi kapasitas untuk kolom statis di Amazon Keyspaces](#)
- [Memperkirakan dan menyediakan kapasitas untuk tabel Multi-wilayah di Amazon Keyspaces](#)
- [Perkiraan konsumsi kapasitas baca dan tulis dengan Amazon CloudWatch di Amazon Keyspaces](#)

## Perkiraan konsumsi kapasitas kueri rentang di Amazon Keyspaces

Untuk melihat konsumsi kapasitas baca dari kueri rentang, kami menggunakan tabel contoh berikut yang menggunakan mode kapasitas sesuai permintaan.

```
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>
```

Sekarang jalankan query berikut pada tabel ini.

```
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;
```

Anda menerima set hasil berikut dari kueri dan operasi baca yang dilakukan oleh Amazon Keyspaces menggunakan 2 RRU dalam mode konsistensi. LOCAL\_QUORUM

```
pk1 | pk2 | pk3 | ck1 | ck2 | ck3 | value
-----+-----+-----+-----+-----+-----+-----
```

```
a | b | 1 | a | b | 60 | value_1
```

Amazon Keyspaces menggunakan 2 RRU untuk mengevaluasi baris dengan nilai `ck3=60` dan `ck3=70` untuk memproses kueri. Namun, Amazon Keyspaces hanya mengembalikan baris di mana `WHERE` kondisi yang ditentukan dalam kueri adalah `true`, yang merupakan baris dengan nilai. `ck3=60` Untuk mengevaluasi rentang yang ditentukan dalam kueri, Amazon Keyspaces membaca baris yang cocok dengan batas atas rentang, dalam hal ini `ck3 = 70`, tetapi tidak mengembalikan baris itu dalam hasilnya. Konsumsi kapasitas baca didasarkan pada data yang dibaca saat memproses kueri, bukan pada data yang dikembalikan.

## Perkirakan konsumsi kapasitas baca kueri batas

Saat memproses kueri yang menggunakan `LIMIT` klausa, Amazon Keyspaces membaca baris hingga ukuran halaman maksimum saat mencoba mencocokkan kondisi yang ditentukan dalam kueri. Jika Amazon Keyspaces tidak dapat menemukan data pencocokan yang memadai yang memenuhi `LIMIT` nilai pada halaman pertama, satu atau beberapa panggilan berhalaman mungkin diperlukan. Untuk melanjutkan pembacaan di halaman berikutnya, Anda dapat menggunakan token pagination. Ukuran halaman default adalah 1MB. Untuk mengurangi kapasitas baca saat menggunakan `LIMIT` klausa, Anda dapat mengurangi ukuran halaman. Untuk informasi lebih lanjut tentang pagination, lihat [the section called “Hasil paginate”](#).

Sebagai contoh, mari kita lihat query berikut.

```
SELECT * FROM my_table WHERE partition_key=1234 LIMIT 1;
```

Jika Anda tidak mengatur ukuran halaman, Amazon Keyspaces membaca 1MB data meskipun hanya mengembalikan 1 baris kepada Anda. Untuk hanya memiliki Amazon Keyspaces membaca satu baris, Anda dapat mengatur ukuran halaman ke 1 untuk kueri ini. Dalam hal ini, Amazon Keyspaces hanya akan membaca satu baris asalkan Anda tidak memiliki baris kedaluwarsa berdasarkan Time-to-live pengaturan atau stempel waktu sisi klien.

`PAGE_SIZE` Parameter menentukan berapa banyak baris Amazon Keyspaces memindai dari disk untuk setiap permintaan, bukan berapa banyak baris Amazon Keyspaces kembali ke klien. Amazon Keyspaces menerapkan filter yang Anda berikan, misalnya ketidaksetaraan pada kolom non-kunci atau `LIMIT` setelah memindai data pada disk. Jika Anda tidak secara eksplisit menyetel, `PAGE_SIZE` Amazon Keyspaces membaca hingga 1MB data sebelum menerapkan filter. Misalnya, jika Anda menggunakan `LIMIT 1` tanpa menentukan `PAGE_SIZE`, Amazon Keyspaces dapat membaca ribuan baris dari disk sebelum menerapkan klausa batas dan hanya mengembalikan satu baris.

Untuk menghindari pembacaan berlebihan, kurangi PAGE SIZE yang mengurangi jumlah baris pindaian Amazon Keyspaces untuk setiap pengambilan. Misalnya, jika Anda menentukan LIMIT 5 dalam kueri Anda, setel PAGE SIZE ke nilai antara 5 - 10 sehingga Amazon Keyspaces hanya memindai 5 - 10 baris pada setiap panggilan paginasi. Anda dapat memodifikasi nomor ini untuk mengurangi jumlah pengambilan. Untuk batas yang lebih besar dari ukuran halaman, Amazon Keyspaces mempertahankan jumlah hasil total dengan status pagination. Dalam kasus 10.000 baris, Amazon Keyspaces dapat mengambil hasil ini dalam dua halaman masing-masing 5.000 baris. LIMIT Batas 1MB adalah batas atas untuk setiap set ukuran halaman.

## Perkirakan konsumsi kapasitas baca pemindaian tabel

Kueri yang menghasilkan pemindaian tabel lengkap, misalnya kueri menggunakan ALLOW FILTERING opsi, adalah contoh lain dari kueri yang memproses lebih banyak pembacaan daripada apa yang mereka kembalikan sebagai hasil. Dan konsumsi kapasitas baca didasarkan pada data yang dibaca, bukan data yang dikembalikan.

Untuk contoh pemindaian tabel kami menggunakan tabel contoh berikut dalam mode kapasitas sesuai permintaan.

```
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 membuat tabel dalam mode kapasitas sesuai permintaan dengan empat partisi secara default. Dalam tabel contoh ini, semua data disimpan dalam satu partisi dan tiga partisi sisanya kosong.

Sekarang jalankan query berikut di atas meja.

```
SELECT * from amazon_keyspaces.example_table_2;
```

Kueri ini menghasilkan operasi pemindaian tabel di mana Amazon Keyspaces memindai keempat partisi tabel dan mengkonsumsi 6 RRU dalam mode konsistensi. LOCAL\_QUORUM Pertama, Amazon Keyspaces mengkonsumsi 3 RRU untuk membaca tiga baris dengan. pk= 'pk' Kemudian, Amazon Keyspaces menggunakan 3 RRU tambahan untuk memindai tiga partisi kosong tabel. Karena kueri ini menghasilkan pemindaian tabel, Amazon Keyspaces memindai semua partisi dalam tabel, termasuk partisi tanpa data.

## Perkirakan konsumsi kapasitas transaksi ringan di Amazon Keyspaces

Transaksi ringan (LWT) memungkinkan Anda melakukan operasi penulisan bersyarat terhadap data tabel Anda. Operasi pembaruan bersyarat berguna saat memasukkan, memperbarui, dan menghapus catatan berdasarkan kondisi yang mengevaluasi keadaan saat ini.

Di Amazon Keyspaces, semua operasi penulisan memerlukan konsistensi `LOCAL_QUORUM` dan tidak ada biaya tambahan untuk menggunakan LWT. Perbedaan untuk LWT adalah bahwa ketika pemeriksaan kondisi LWT menghasilkan, `FALSE` Amazon Keyspaces menggunakan unit kapasitas tulis (WCU) atau unit permintaan tulis (WRU). Jumlah yang WCUs/WRUs dikonsumsi tergantung pada ukuran baris.

Misalnya, jika ukuran baris adalah 2 KB, penulisan bersyarat yang gagal mengkonsumsi dua WCUs/WRUs. Jika baris saat ini tidak ada dalam tabel, operasi akan menggunakan satu WCUs/WRUs.

Untuk menentukan jumlah permintaan yang mengakibatkan kegagalan pemeriksaan kondisi, Anda dapat memantau `ConditionalCheckFailed` metrik di CloudWatch.

## Perkirakan biaya LWT untuk tabel dengan Time to Live (TTL)

LWT dapat memerlukan unit kapasitas baca tambahan (RCU) atau unit permintaan baca (RRU) untuk tabel yang dikonfigurasi dengan TTL yang tidak menggunakan stempel waktu sisi klien. Saat menggunakan `IF EXISTS` atau `IF NOT EXISTS` kata kunci hasil pemeriksaan kondisi `FALSE`, unit kapasitas berikut dikonsumsi:

- RCU/RRUs — Jika baris ada, yang RCU/RRUs dikonsumsi didasarkan pada ukuran baris yang ada.
- RCU/RRUs — Jika baris tidak ada, satu pun RCU/RRU dikonsumsi.

Jika kondisi yang dievaluasi menghasilkan operasi penulisan WCUs/WRUs yang sukses, dikonsumsi berdasarkan ukuran baris baru.

## Perkirakan konsumsi kapasitas untuk kolom statis di Amazon Keyspaces

Dalam tabel Amazon Keyspaces dengan kolom pengelompokan, Anda dapat menggunakan `STATIC` kata kunci untuk membuat kolom statis. Nilai yang disimpan dalam kolom statis dibagi antara semua baris dalam partisi logis. Saat Anda memperbarui nilai kolom ini, Amazon Keyspaces menerapkan perubahan secara otomatis ke semua baris di partisi.



Bagian ini menjelaskan cara menghitung ukuran data yang dikodekan saat Anda menulis ke kolom statis. Proses ini ditangani secara terpisah dari proses yang menulis data ke kolom nonstatis baris. Selain kuota ukuran untuk data statis, operasi baca dan tulis pada kolom statis juga memengaruhi pengukuran dan kapasitas throughput untuk tabel secara independen. Untuk perbedaan fungsional dengan Apache Cassandra saat menggunakan kolom statis dan hasil baca rentang paginasi, lihat [the section called “Paginasi”](#)

## Topik

- [Hitung ukuran kolom statis per partisi logis di Amazon Keyspaces](#)
- [Perkirakan persyaratan throughput kapasitas untuk read/write operasi pada data statis di Amazon Keyspaces](#)

## Hitung ukuran kolom statis per partisi logis di Amazon Keyspaces

Bagian ini memberikan detail tentang cara memperkirakan ukuran kolom statis yang dikodekan di Amazon Keyspaces. Ukuran yang dikodekan digunakan saat Anda menghitung tagihan dan penggunaan kuota Anda. Anda juga harus menggunakan ukuran yang dikodekan saat menghitung persyaratan kapasitas throughput yang disediakan untuk tabel. Untuk menghitung ukuran kolom statis yang dikodekan di Amazon Keyspaces, Anda dapat menggunakan panduan berikut.

- Kunci partisi dapat berisi hingga 2048 byte data. Setiap kolom kunci dalam kunci partisi membutuhkan hingga 3 byte metadata. Byte metadata ini dihitung terhadap kuota ukuran data statis Anda sebesar 1 MB per partisi. Saat menghitung ukuran data statis Anda, Anda harus berasumsi bahwa setiap kolom kunci partisi menggunakan metadata 3 byte penuh.
- Gunakan ukuran mentah dari nilai data kolom statis berdasarkan tipe data. Untuk informasi selengkapnya tentang jenis data, lihat [the section called “Jenis Data”](#).
- Tambahkan 104 byte ke ukuran data statis untuk metadata.
- Kolom pengelompokan dan kolom kunci nonprimer reguler tidak dihitung terhadap ukuran data statis. Untuk mempelajari cara memperkirakan ukuran data nonstatis dalam baris, lihat [the section called “Perkiraan ukuran baris”](#).

Ukuran total dikodekan dari kolom statis didasarkan pada rumus berikut:

```
partition key columns + static columns + metadata = total encoded size of static data
```

Perhatikan contoh berikut dari tabel di mana semua kolom adalah tipe integer. Tabel ini memiliki dua kolom kunci partisi, dua kolom pengelompokan, satu kolom reguler, dan satu kolom statis.

```
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));
```

Dalam contoh ini, kami menghitung ukuran data statis dari pernyataan berikut:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, static_col1) values(1,2,6);
```

Untuk memperkirakan total byte yang diperlukan oleh operasi penulisan ini, Anda dapat menggunakan langkah-langkah berikut.

1. Hitung ukuran kolom kunci partisi dengan menambahkan byte untuk tipe data yang disimpan di kolom dan byte metadata. Ulangi ini untuk semua kolom kunci partisi.

- a. Hitung ukuran kolom pertama dari kunci partisi (pk\_col1):

```
4 bytes for the integer data type + 3 bytes for partition key metadata = 7
bytes
```

- b. Hitung ukuran kolom kedua dari kunci partisi (pk\_col2):

```
4 bytes for the integer data type + 3 bytes for partition key metadata = 7
bytes
```

- c. Tambahkan kedua kolom untuk mendapatkan ukuran estimasi total kolom kunci partisi:

```
7 bytes + 7 bytes = 14 bytes for the partition key columns
```

2. Tambahkan ukuran kolom statis. Dalam contoh ini, kita hanya memiliki satu kolom statis yang menyimpan integer (yang membutuhkan 4 byte).
3. Terakhir, untuk mendapatkan ukuran total yang dikodekan dari data kolom statis, tambahkan byte untuk kolom kunci utama dan kolom statis, dan tambahkan 104 byte tambahan untuk metadata:

```
14 bytes for the partition key columns + 4 bytes for the static column + 104 bytes
for metadata = 122 bytes.
```

Anda juga dapat memperbarui data statis dan nonstatis dengan pernyataan yang sama. Untuk memperkirakan ukuran total operasi tulis, Anda harus terlebih dahulu menghitung ukuran pembaruan data nonstatis. Kemudian hitung ukuran pembaruan baris seperti yang ditunjukkan pada contoh [di the section called “Perkiraan ukuran baris”](#), dan tambahkan hasilnya.

Dalam hal ini, Anda dapat menulis total 2 MB—1 MB adalah kuota ukuran baris maksimum, dan 1 MB adalah kuota untuk ukuran data statis maksimum per partisi logis.

Untuk menghitung ukuran total pembaruan data statis dan nonstatis dalam pernyataan yang sama, Anda dapat menggunakan rumus berikut:

```
(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
```

Perhatikan contoh berikut dari tabel di mana semua kolom adalah tipe integer. Tabel ini memiliki dua kolom kunci partisi, dua kolom pengelompokan, satu kolom reguler, dan satu kolom statis.

```
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));
```

Dalam contoh ini, kita menghitung ukuran data ketika kita menulis baris ke tabel, seperti yang ditunjukkan dalam pernyataan berikut:

```
INSERT INTO mykeyspace.mytable (pk_col1, pk_col2, ck_col1, ck_col2, reg_col1,
static_col1) values(2,3,4,5,6,7);
```

Untuk memperkirakan total byte yang diperlukan oleh operasi penulisan ini, Anda dapat menggunakan langkah-langkah berikut.

1. Hitung total ukuran data statis yang dikodekan seperti yang ditunjukkan sebelumnya. Dalam contoh ini, itu 122 byte.
2. Tambahkan ukuran total ukuran baris yang dikodekan berdasarkan pembaruan data nonstatis, mengikuti langkah-langkah di [the section called “Perkiraan ukuran baris”](#). Dalam contoh ini, ukuran total pembaruan baris adalah 134 byte.

```
122 bytes for static data + 134 bytes for nonstatic data = 256 bytes.
```

## Perkirakan persyaratan throughput kapasitas untuk read/write operasi pada data statis di Amazon Keyspaces

Data statis dikaitkan dengan partisi logis di Cassandra, bukan baris individual. Partisi logis di Amazon Keyspaces dapat hampir tidak terikat ukurannya dengan menjangkau beberapa partisi penyimpanan fisik. Akibatnya, Amazon Keyspaces meteran menulis operasi pada data statis dan nonstatis secara terpisah. Selanjutnya, menulis yang mencakup data statis dan nonstatis memerlukan operasi tambahan yang mendasari untuk memberikan konsistensi data.

Jika Anda melakukan operasi penulisan campuran dari data statis dan nonstatis, ini menghasilkan dua operasi tulis terpisah—satu untuk nonstatis dan satu untuk data statis. Ini berlaku untuk mode kapasitas sesuai permintaan dan yang disediakan. read/write

Contoh berikut memberikan detail tentang cara memperkirakan unit kapasitas baca (RCU) yang diperlukan dan unit kapasitas tulis (WCU) saat Anda menghitung persyaratan kapasitas throughput yang disediakan untuk tabel di Amazon Keyspaces yang memiliki kolom statis. Anda dapat memperkirakan berapa banyak kapasitas yang dibutuhkan tabel Anda untuk memproses penulisan yang mencakup data statis dan nonstatis dengan menggunakan rumus berikut:

```
2 x WCUs required for nonstatic data + 2 x WCUs required for static data
```

Misalnya, jika aplikasi Anda menulis 27 KB data per detik dan setiap penulisan mencakup 25,5 KB data nonstatis dan 1,5 KB data statis, maka tabel Anda memerlukan 56 WCU ( $2 \times 26 \text{ WCU} + 2 \times 2 \text{ WCU}$ ).

Amazon Keyspaces mengukur pembacaan data statis dan nonstatis sama dengan pembacaan beberapa baris. Akibatnya, harga pembacaan data statis dan nonstatis dalam operasi yang sama didasarkan pada ukuran agregat data yang diproses untuk melakukan pembacaan.

Untuk mempelajari cara memantau sumber daya tanpa server dengan Amazon CloudWatch, lihat [the section called “Pemantauan CloudWatch dengan”](#)

## Memperkirakan dan menyediakan kapasitas untuk tabel Multi-wilayah di Amazon Keyspaces

Anda dapat mengonfigurasi kapasitas throughput tabel Multi-wilayah dengan salah satu dari dua cara:

- On-demand mode kapasitas, diukur dalam unit permintaan tulis (WRU)

- Mode kapasitas yang disediakan dengan penskalaan otomatis, diukur dalam unit kapasitas tulis (WCU)

Anda dapat menggunakan mode kapasitas yang disediakan dengan penskalaan otomatis atau mode kapasitas sesuai permintaan untuk membantu memastikan bahwa tabel Multi-wilayah memiliki kapasitas yang cukup untuk melakukan penulisan yang direplikasi ke semua. Wilayah AWS

 Note

Mengubah mode kapasitas tabel di salah satu Wilayah mengubah mode kapasitas untuk semua replika.

Secara default, Amazon Keyspaces menggunakan mode sesuai permintaan untuk tabel Multi-wilayah. Dengan mode on-demand, Anda tidak perlu menentukan berapa banyak throughput baca dan tulis yang Anda harapkan untuk dilakukan aplikasi Anda. Amazon Keyspaces langsung mengakomodasi beban kerja Anda saat mereka naik atau turun ke tingkat lalu lintas yang dicapai sebelumnya. Jika tingkat lalu lintas beban kerja mencapai puncak baru, Amazon Keyspaces beradaptasi dengan cepat untuk mengakomodasi beban kerja.

Jika Anda memilih mode kapasitas yang disediakan untuk sebuah tabel, Anda harus mengonfigurasi jumlah unit kapasitas baca (RCU) dan unit kapasitas tulis (WCU) per detik yang dibutuhkan aplikasi Anda.

Untuk merencanakan kebutuhan kapasitas throughput tabel Multi-wilayah, Anda harus terlebih dahulu memperkirakan jumlah WCU per detik yang dibutuhkan untuk setiap Wilayah. Kemudian Anda menambahkan penulisan dari semua Wilayah tempat tabel Anda direplikasi, dan menggunakan jumlah untuk menyediakan kapasitas untuk setiap Wilayah. Hal ini diperlukan karena setiap penulisan yang dilakukan di satu Wilayah juga harus diulang di setiap replika Region.

Jika tabel tidak memiliki kapasitas yang cukup untuk menangani penulisan dari semua Wilayah, pengecualian kapasitas akan terjadi. Selain itu, waktu tunggu replikasi antar-regional akan meningkat.

Misalnya, jika Anda memiliki tabel Multi-wilayah di mana Anda mengharapkan 5 penulisan per detik di AS Timur (Virginia N.), 10 menulis per detik di AS Timur (Ohio), dan 5 menulis per detik di Eropa (Irlandia), Anda harus mengharapkan tabel untuk mengkonsumsi 20 WCU di setiap Wilayah: AS Timur (Virginia N.), AS Timur (Ohio), dan Eropa (Irlandia). Itu berarti bahwa dalam contoh ini, Anda perlu menyediakan 20 WCU untuk setiap replika tabel. Anda dapat memantau konsumsi kapasitas

tabel Anda menggunakan Amazon CloudWatch. Untuk informasi selengkapnya, lihat [the section called “Pemantauan CloudWatch dengan”](#).

Setiap penulisan ditagih sebagai 1 WCU, jadi Anda akan melihat total 60 WCU ditagih dalam contoh ini. Untuk informasi selengkapnya tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

Untuk informasi selengkapnya tentang kapasitas yang disediakan dengan penskalaan otomatis Amazon Keyspaces, lihat. [the section called “Kelola kapasitas throughput dengan penskalaan otomatis”](#)

#### Note

Jika tabel berjalan dalam mode kapasitas yang disediakan dengan penskalaan otomatis, kapasitas tulis yang disediakan diizinkan untuk mengambang dalam pengaturan penskalaan otomatis tersebut untuk setiap Wilayah.

## Perkirakan konsumsi kapasitas baca dan tulis dengan Amazon CloudWatch di Amazon Keyspaces

Untuk memperkirakan dan memantau konsumsi kapasitas baca dan tulis, Anda dapat menggunakan CloudWatch dasbor. Untuk informasi selengkapnya tentang metrik yang tersedia untuk Amazon Keyspaces, lihat. [the section called “Metrik dan dimensi”](#)

Untuk memantau unit kapasitas baca dan tulis yang dikonsumsi oleh pernyataan tertentu dengan CloudWatch, Anda dapat mengikuti langkah-langkah ini.

1. Buat tabel baru dengan data sampel
2. Konfigurasi CloudWatch dasbor Amazon Keyspaces untuk tabel. Untuk memulai, Anda dapat menggunakan template dasbor yang tersedia di [Github](#).
3. Jalankan pernyataan CQL, misalnya menggunakan `ALLOW FILTERING` opsi, dan periksa unit kapasitas baca yang dikonsumsi untuk pemindaian tabel lengkap di dasbor.

## Konfigurasi mode read/write kapasitas di Amazon Keyspaces

Amazon Keyspaces memiliki dua mode read/write kapasitas untuk memproses membaca dan menulis di tabel Anda:

- On-demand (default)
- Disediakan

Mode read/write kapasitas yang Anda pilih mengontrol bagaimana Anda dikenakan biaya untuk throughput baca dan tulis dan bagaimana kapasitas throughput tabel dikelola.

### Topik

- [Konfigurasi mode kapasitas sesuai permintaan](#)
- [Konfigurasi mode kapasitas yang disediakan](#)
- [Melihat mode kapasitas tabel di Amazon Keyspaces](#)
- [Ubah mode kapasitas tabel di Amazon Keyspaces](#)
- [Mengonfigurasi prapenghangatan untuk tabel di Amazon Keyspaces](#)

## Konfigurasi mode kapasitas sesuai permintaan

Amazon Keyspaces (untuk Apache Cassandra) mode kapasitas sesuai permintaan adalah opsi penagihan fleksibel yang mampu melayani ribuan permintaan per detik tanpa perencanaan kapasitas. Opsi ini menawarkan harga bayar per permintaan untuk permintaan baca dan tulis sehingga Anda hanya membayar untuk apa yang Anda gunakan.

Saat Anda memilih mode sesuai permintaan, Amazon Keyspaces dapat menskalakan kapasitas throughput untuk tabel Anda hingga tingkat lalu lintas yang sebelumnya dicapai secara instan, dan kemudian mundur saat lalu lintas aplikasi menurun. Jika tingkat lalu lintas beban kerja mencapai puncak baru, layanan beradaptasi dengan cepat untuk meningkatkan kapasitas throughput untuk meja Anda. Anda dapat mengaktifkan mode kapasitas sesuai permintaan untuk tabel baru dan yang sudah ada.

On-demand mode adalah pilihan yang baik jika salah satu dari berikut ini benar:

- Anda membuat tabel baru dengan beban kerja yang tidak diketahui.
- Anda memiliki lalu lintas aplikasi yang tidak dapat diprediksi.
- Anda lebih menyukai kemudahan membayar hanya sesuai penggunaan.

Untuk memulai mode on-demand, Anda dapat membuat tabel baru atau memperbarui tabel yang ada untuk menggunakan mode kapasitas sesuai permintaan menggunakan konsol atau dengan beberapa

baris kode Cassandra Query Language (CQL). Untuk informasi selengkapnya, lihat [the section called “Tabel”](#).

## Topik

- [Unit permintaan baca dan unit permintaan tulis](#)
- [Properti lalu lintas puncak dan penskalaan](#)
- [Throughput awal untuk mode kapasitas sesuai permintaan](#)

## Unit permintaan baca dan unit permintaan tulis

Dengan tabel mode kapasitas sesuai permintaan, Anda tidak perlu menentukan berapa banyak throughput baca dan tulis yang Anda harapkan untuk digunakan aplikasi Anda sebelumnya. Amazon Keyspaces menagih Anda untuk membaca dan menulis yang Anda lakukan di tabel Anda dalam hal unit permintaan baca (RRU) dan unit permintaan tulis (WRU).

- Satu RRU mewakili satu permintaan LOCAL\_QUORUM baca, atau dua permintaan LOCAL\_ONE baca, untuk satu baris hingga 4 KB. Jika Anda perlu membaca baris yang lebih besar dari 4 KB, operasi baca menggunakan RRU tambahan. Jumlah total RRU yang diperlukan tergantung pada ukuran baris, dan apakah Anda ingin menggunakan LOCAL\_QUORUM atau LOCAL\_ONE membaca konsistensi. Misalnya, membaca baris 8 KB membutuhkan 2 RRU menggunakan konsistensi LOCAL\_QUORUM baca, dan 1 RRU jika Anda memilih konsistensi LOCAL\_ONE baca.
- Satu WRU mewakili satu tulis untuk satu baris dengan ukuran hingga 1 KB. Semua penulisan menggunakan LOCAL\_QUORUM konsistensi, dan tidak ada biaya tambahan untuk menggunakan transaksi ringan (LWT). Jika Anda perlu menulis baris yang lebih besar dari 1 KB, operasi tulis menggunakan WRU tambahan. Jumlah total WRU yang dibutuhkan tergantung pada ukuran baris. Misalnya, jika ukuran baris Anda adalah 2 KB, Anda memerlukan 2 WRU untuk melakukan satu permintaan tulis.

Untuk informasi tentang tingkat konsistensi yang didukung, lihat [the section called “Tingkat konsistensi Cassandra yang didukung”](#).

## Properti lalu lintas puncak dan penskalaan

Tabel Amazon Keyspaces yang menggunakan mode kapasitas sesuai permintaan secara otomatis beradaptasi dengan volume lalu lintas aplikasi Anda. On-demand mode kapasitas langsung mengakomodasi hingga dua kali lipat lalu lintas puncak sebelumnya di atas meja. Misalnya, pola lalu



lintas aplikasi Anda mungkin bervariasi antara 5.000 dan 10.000 LOCAL\_QUORUM pembacaan per detik, di mana 10.000 pembacaan per detik adalah puncak lalu lintas sebelumnya.

Dengan pola ini, mode kapasitas sesuai permintaan langsung mengakomodasi lalu lintas berkelanjutan hingga 20.000 pembacaan per detik. Jika aplikasi Anda mempertahankan lalu lintas 20.000 pembacaan per detik, puncak itu menjadi puncak baru Anda sebelumnya, memungkinkan lalu lintas berikutnya mencapai hingga 40.000 pembacaan per detik.

Jika Anda membutuhkan lebih dari dua kali lipat puncak sebelumnya di atas meja, Amazon Keyspaces secara otomatis mengalokasikan lebih banyak kapasitas saat volume lalu lintas Anda meningkat. Ini membantu memastikan bahwa tabel Anda memiliki kapasitas throughput yang cukup untuk memproses permintaan tambahan. Namun, Anda mungkin mengamati kesalahan kapasitas throughput yang tidak mencukupi jika Anda melebihi dua kali lipat puncak sebelumnya dalam 30 menit.

Misalnya, misalkan pola lalu lintas aplikasi Anda bervariasi antara 5.000 dan 10.000 pembacaan yang sangat konsisten per detik, di mana 20.000 pembacaan per detik adalah puncak lalu lintas yang dicapai sebelumnya. Dalam hal ini, layanan merekomendasikan agar Anda menempatkan pertumbuhan lalu lintas Anda setidaknya selama 30 menit sebelum mengemudi hingga 40.000 pembacaan per detik.

Untuk mempelajari cara memperkirakan konsumsi kapasitas baca dan tulis tabel, lihat [the section called “Perkiraan konsumsi kapasitas”](#).

Untuk mempelajari lebih lanjut tentang kuota default untuk akun Anda dan cara meningkatkannya, lihat [Kuota](#).

## Throughput awal untuk mode kapasitas sesuai permintaan

Jika Anda membuat tabel baru dengan mode kapasitas sesuai permintaan diaktifkan atau mengalihkan tabel yang ada ke mode kapasitas sesuai permintaan untuk pertama kalinya, tabel memiliki pengaturan puncak sebelumnya berikut, meskipun sebelumnya tidak melayani lalu lintas menggunakan mode kapasitas sesuai permintaan:

- Tabel yang baru dibuat dengan mode kapasitas sesuai permintaan: Puncak sebelumnya adalah 2.000 WRU dan 6.000 RRU. Anda dapat berkendara hingga menggandakan puncak sebelumnya dengan segera. Melakukan hal ini memungkinkan tabel sesuai permintaan yang baru dibuat untuk melayani hingga 4.000 WRU dan 12.000 RRU.

- Tabel yang ada beralih ke mode kapasitas sesuai permintaan: Puncak sebelumnya adalah setengah dari WCU dan RCU sebelumnya yang disediakan untuk tabel atau pengaturan untuk tabel yang baru dibuat dengan mode kapasitas sesuai permintaan, mana yang lebih tinggi.

## Konfigurasi mode kapasitas yang disediakan

Jika Anda memilih mode kapasitas throughput yang disediakan, Anda menentukan jumlah pembacaan dan penulisan per detik yang diperlukan untuk aplikasi Anda. Ini membantu Anda mengelola penggunaan Amazon Keyspaces agar tetap pada atau di bawah tingkat permintaan yang ditentukan untuk mempertahankan prediktabilitas. Untuk mempelajari lebih lanjut tentang penskalaan otomatis untuk throughput yang disediakan, lihat [the section called “Kelola kapasitas throughput dengan penskalaan otomatis”](#)

Mode kapasitas throughput yang disediakan adalah pilihan yang baik jika salah satu dari berikut ini benar:

- Anda memiliki lalu lintas aplikasi yang dapat diprediksi.
- Anda menjalankan aplikasi yang lalu lintasnya konsisten atau naik secara bertahap.
- Anda dapat memperkirakan persyaratan kapasitas.

### Unit kapasitas baca dan unit kapasitas tulis

Untuk tabel mode kapasitas throughput yang disediakan, Anda menentukan kapasitas throughput dalam hal unit kapasitas baca (RCU) dan unit kapasitas tulis (WCU):

- Satu RCU mewakili satu LOCAL\_QUORUM pembacaan per detik, atau dua LOCAL\_ONE pembacaan per detik, untuk satu baris hingga 4 KB. Jika Anda perlu membaca baris yang lebih besar dari 4 KB, operasi baca menggunakan RCU tambahan.

Jumlah total RCU yang diperlukan tergantung pada ukuran baris, dan apakah Anda ingin LOCAL\_QUORUM atau LOCAL\_ONE membaca. Misalnya, jika ukuran baris Anda adalah 8 KB, Anda memerlukan 2 RCU untuk mempertahankan satu LOCAL\_QUORUM pembacaan per detik, dan 1 RCU jika Anda memilih pembacaan. LOCAL\_ONE

- Satu WCU mewakili satu tulis per detik untuk satu baris dengan ukuran hingga 1 KB. Semua penulisan menggunakan LOCAL\_QUORUM konsistensi, dan tidak ada biaya tambahan untuk menggunakan transaksi ringan (LWT). Jika Anda perlu menulis baris yang lebih besar dari 1 KB, operasi tulis menggunakan WCU tambahan.

Jumlah total WCU yang dibutuhkan tergantung pada ukuran baris. Misalnya, jika ukuran baris Anda adalah 2 KB, Anda memerlukan 2 WCU untuk mempertahankan satu permintaan tulis per detik. Untuk informasi lebih lanjut tentang cara memperkirakan konsumsi kapasitas baca dan tulis tabel, lihat [the section called “Perkiraan konsumsi kapasitas”](#).

Jika aplikasi Anda membaca atau menulis baris yang lebih besar (hingga ukuran baris maksimum Amazon Keyspaces sebesar 1 MB), aplikasi akan menghabiskan lebih banyak unit kapasitas. Untuk mempelajari lebih lanjut tentang cara memperkirakan ukuran baris, lihat [the section called “Perkiraan ukuran baris”](#). Misalnya, Anda membuat tabel yang disediakan dengan 6 RCU dan 6 WCU. Dengan pengaturan ini, aplikasi Anda dapat melakukan hal berikut:

- Lakukan LOCAL\_QUORUM pembacaan hingga 24 KB per detik ( $4 \text{ KB} \times 6 \text{ RCU}$ ).
- Lakukan LOCAL\_ONE pembacaan hingga 48 KB per detik (throughput baca dua kali lebih banyak).
- Tulis hingga 6 KB per detik ( $1 \text{ KB} \times 6 \text{ WCU}$ ).

Throughput yang disediakan adalah jumlah maksimum kapasitas throughput yang dapat dikonsumsi aplikasi dari sebuah tabel. Jika aplikasi Anda melebihi kapasitas throughput yang disediakan, Anda mungkin melihat kesalahan kapasitas yang tidak mencukupi.

Misalnya, permintaan baca yang tidak memiliki kapasitas throughput yang cukup gagal dengan `Read_Timeout` pengecualian dan diposting ke `ReadThrottleEvents` metrik. Permintaan tulis yang tidak memiliki kapasitas throughput yang cukup gagal dengan `Write_Timeout` pengecualian dan diposting ke `WriteThrottleEvents` metrik.

Anda dapat menggunakan Amazon CloudWatch untuk memantau metrik throughput yang disediakan dan aktual serta peristiwa kapasitas yang tidak mencukupi. Untuk informasi selengkapnya tentang metrik ini, lihat [the section called “Metrik dan dimensi”](#).

#### Note

Kesalahan berulang karena kapasitas yang tidak mencukupi dapat menyebabkan pengecualian khusus driver sisi klien, misalnya driver DataStax Java gagal dengan file. `NoHostAvailableException`

Untuk mengubah pengaturan kapasitas throughput untuk tabel, Anda dapat menggunakan Konsol Manajemen AWS atau ALTER TABLE pernyataan menggunakan CQL, untuk informasi selengkapnya lihat [the section called “ALTER TABLE”](#)

Untuk mempelajari lebih lanjut tentang kuota default untuk akun Anda dan cara meningkatkannya, lihat [Kuota](#).

## Melihat mode kapasitas tabel di Amazon Keyspaces

Anda dapat menanyakan tabel sistem di ruang kunci sistem Amazon Keyspaces untuk meninjau informasi mode kapasitas tentang tabel. Anda juga dapat melihat apakah tabel menggunakan mode kapasitas throughput sesuai permintaan atau disediakan. Jika tabel dikonfigurasi dengan mode kapasitas throughput yang disediakan, Anda dapat melihat kapasitas throughput yang disediakan untuk tabel.

Anda juga dapat menggunakan AWS CLI untuk melihat mode kapasitas tabel.

Untuk mengubah throughput tabel yang disediakan, lihat [the section called “Ubah mode kapasitas”](#)

### Cassandra Query Language (CQL)

#### Contoh

```
SELECT * from system_schema_mcs.tables where keyspace_name = 'mykeyspace' and
table_name = 'mytable';
```

Tabel yang dikonfigurasi dengan mode kapasitas sesuai permintaan mengembalikan yang berikut ini.

```
{
 "capacity_mode":{
 "last_update_to_pay_per_request_timestamp":"1579551547603",
 "throughput_mode":"PAY_PER_REQUEST"
 }
}
```

Tabel yang dikonfigurasi dengan mode kapasitas throughput yang disediakan mengembalikan yang berikut ini.

```
{
 "capacity_mode":{
```

```
"last_update_to_pay_per_request_timestamp": "1579048006000",
"read_capacity_units": "5000",
"throughput_mode": "PROVISIONED",
"write_capacity_units": "6000"
}
}
```

`last_update_to_pay_per_request_timestamp` Nilai diukur dalam milidetik.

## CLI

Melihat mode kapasitas throughput tabel menggunakan AWS CLI

```
aws keyspaces get-table --keyspace-name myKeyspace --table-name myTable
```

Output dari perintah dapat terlihat mirip dengan ini untuk tabel dalam mode kapasitas yang disediakan.

```
"capacitySpecification": {
 "throughputMode": "PROVISIONED",
 "readCapacityUnits": 4000,
 "writeCapacityUnits": 2000
}
```

Output untuk tabel dalam mode on-demand terlihat seperti ini.

```
"capacitySpecification": {
 "throughputMode": "PAY_PER_REQUEST",
 "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
}
```

## Ubah mode kapasitas tabel di Amazon Keyspaces

Saat Anda mengalihkan tabel dari mode kapasitas yang disediakan ke mode kapasitas sesuai permintaan, Amazon Keyspaces membuat beberapa perubahan pada struktur tabel dan partisi Anda. Proses ini dapat memakan waktu beberapa menit. Selama periode switching, tabel Anda memberikan throughput yang konsisten dengan jumlah WCU dan RCU yang telah disediakan sebelumnya.

Saat Anda beralih dari mode kapasitas sesuai permintaan kembali ke mode kapasitas yang disediakan, tabel Anda memberikan throughput yang konsisten dengan puncak sebelumnya yang dicapai saat tabel diatur ke mode kapasitas sesuai permintaan.

Periode tunggu berikut berlaku saat Anda beralih mode kapasitas:

- Anda dapat mengganti tabel yang baru dibuat dalam mode sesuai permintaan ke mode kapasitas yang disediakan kapan saja. Namun, Anda hanya dapat mengubahnya kembali ke mode sesuai permintaan 24 jam setelah stempel waktu pembuatan tabel.
- Anda dapat mengganti tabel yang ada dalam mode sesuai permintaan ke mode kapasitas yang disediakan kapan saja. Namun, Anda dapat mengubah mode kapasitas dari yang disediakan ke sesuai permintaan hanya sekali dalam periode 24 jam.

## Cassandra Query Language (CQL)

Ubah mode kapasitas throughput tabel menggunakan CQL

1. Untuk mengubah mode kapasitas tabel untuk PROVISIONED Anda harus mengkonfigurasi kapasitas baca dan menulis unit kapasitas berdasarkan beban kerja Anda nilai puncak yang diharapkan. pernyataan berikut adalah contoh dari ini. Anda juga dapat menjalankan pernyataan ini untuk menyesuaikan kapasitas baca atau unit kapasitas tulis tabel.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 6000,
'write_capacity_units': 3000}};
```

Untuk mengonfigurasi mode kapasitas yang disediakan dengan auto-scaling, lihat. [the section called “Konfigurasi penskalaan otomatis pada tabel yang ada”](#)

2. Untuk mengubah mode kapasitas tabel ke mode on-demand, atur mode throughput ke. PAY\_PER\_REQUEST Pernyataan berikut adalah contohnya.

```
ALTER TABLE catalog.book_awards WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PAY_PER_REQUEST'}};
```

3. Anda dapat menggunakan pernyataan berikut untuk mengonfirmasi mode kapasitas tabel.

```
SELECT * from system_schema.tables where keyspace_name = 'catalog' and
table_name = 'book_awards';
```

Tabel yang dikonfigurasi dengan mode kapasitas sesuai permintaan mengembalikan yang berikut ini.

```
{
 "capacity_mode":{
 "last_update_to_pay_per_request_timestamp":"1727952499092",
 "throughput_mode":"PAY_PER_REQUEST"
 }
}
```

`last_update_to_pay_per_request_timestamp` Nilai diukur dalam milidetik.

## CLI

Ubah mode kapasitas throughput tabel menggunakan AWS CLI

1. Untuk mengubah mode kapasitas tabel, PROVISIONED Anda harus mengonfigurasi kapasitas baca dan unit kapasitas tulis berdasarkan nilai puncak yang diharapkan dari beban kerja Anda. Perintah berikut adalah contohnya. Anda juga dapat menjalankan perintah ini untuk menyesuaikan kapasitas baca atau unit kapasitas tulis tabel.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards
 \--capacity-specification
 throughputMode=PROVISIONED,readCapacityUnits=6000,writeCapacityUnits=3000
```

Untuk mengonfigurasi mode kapasitas yang disediakan dengan auto-scaling, lihat. [the section called “Konfigurasi penskalaan otomatis pada tabel yang ada”](#)

2. Untuk mengubah mode kapasitas tabel ke mode on-demand, Anda mengatur mode throughput ke. PAY\_PER\_REQUEST Pernyataan berikut adalah contohnya.

```
aws keyspaces update-table --keyspace-name catalog --table-name book_awards
 \--capacity-specification
 throughputMode=PAY_PER_REQUEST
```

3. Anda dapat menggunakan perintah berikut untuk meninjau mode kapasitas yang dikonfigurasi untuk tabel.

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

Output untuk tabel dalam mode on-demand terlihat seperti ini.

```
"capacitySpecification": {
 "throughputMode": "PAY_PER_REQUEST",
 "lastUpdateToPayPerRequestTimestamp": "2024-10-03T10:48:19.092000+00:00"
}
```

## Mengonfigurasi prapenghangatan untuk tabel di Amazon Keyspaces

Amazon Keyspaces secara otomatis menskalakan partisi penyimpanan berdasarkan throughput sesuai permintaan atau yang disediakan, tetapi untuk tabel baru atau puncak throughput mendadak, diperlukan waktu lebih lama untuk mengalokasikan partisi penyimpanan yang diperlukan. Untuk memastikan bahwa tabel baru atau yang sudah ada memiliki kapasitas yang cukup untuk mendukung throughput puncak yang diantisipasi, Anda dapat secara manual mengatur nilai throughput hangat tertentu untuk pra-pemanasan tabel Anda.

Throughput hangat mengacu pada jumlah operasi baca dan tulis yang dapat didukung oleh tabel Amazon Keyspaces Anda secara instan. Nilai-nilai ini tersedia secara default untuk semua tabel baru dan yang sudah ada. Jika Anda menggunakan mode sesuai permintaan, atau jika Anda memperbarui throughput yang disediakan, Amazon Keyspaces memastikan bahwa aplikasi Anda dapat mengeluarkan permintaan hingga nilai tersebut secara instan.

Amazon Keyspaces secara otomatis menyesuaikan nilai throughput hangat saat penggunaan Anda meningkat. Untuk menyesuaikan kapasitas throughput untuk peristiwa puncak yang akan datang, misalnya saat Anda memigrasikan data dari database lain, yang mungkin memerlukan pemuatan terabyte data dalam waktu singkat, Anda dapat meningkatkan nilai throughput hangat tabel secara manual. Ini berguna untuk peristiwa puncak yang direncanakan di mana tingkat permintaan mungkin meningkat 10x, 100x, atau lebih. Pertama, menilai apakah throughput hangat saat ini cukup untuk menangani lalu lintas yang diharapkan. [Kemudian, jika Anda perlu melakukan pra-pemanasan tabel untuk beban kerja puncak yang direncanakan, Anda dapat meningkatkan nilai throughput hangat secara manual tanpa mengubah pengaturan throughput atau mode kapasitas Anda.](#)

Anda dapat melakukan pra-pemanasan tabel untuk operasi baca, operasi tulis, atau keduanya. Anda dapat meningkatkan nilai ini untuk tabel Single-region baru dan yang sudah ada serta tabel Multi-region dan pengaturan throughput hangat yang Anda tetapkan berlaku secara otomatis ke semua replika tabel Multi-wilayah. Tidak ada batasan jumlah tabel Amazon Keyspaces yang dapat Anda hangatkan kapan saja. Waktu untuk menyelesaikan pra-pemanasan tergantung pada nilai yang Anda tetapkan dan ukuran tabel. Anda dapat mengirimkan permintaan pra-hangat secara simultan dan permintaan ini tidak mengganggu operasi tabel apa pun. Anda dapat melakukan pra-pemanasan



tabel hingga batas kuota tabel untuk akun Anda di Wilayah tersebut. Gunakan [konsol Service Quotas untuk memeriksa kuota](#) Anda saat ini dan meningkatkannya jika diperlukan.

Nilai throughput hangat yang disesuaikan Amazon Keyspaces berdasarkan penggunaan sesuai permintaan atau kapasitas yang disediakan tersedia secara default untuk semua tabel tanpa biaya tambahan. Namun, jika Anda secara manual meningkatkan nilai throughput hangat default ke tabel pra-hangat untuk peristiwa lalu lintas puncak, biaya tambahan akan berlaku. Untuk informasi selengkapnya, lihat [harga Amazon Keyspaces](#).

Berikut adalah beberapa skenario dan praktik terbaik yang mungkin Anda pertimbangkan saat melakukan pra-pemanasan tabel Amazon Keyspaces.

### Throughput hangat dan pola akses yang tidak merata

Sebuah tabel mungkin memiliki throughput hangat 30.000 unit baca per detik dan 10.000 unit tulis per detik, tetapi Anda masih dapat mengalami kapasitas melebihi peristiwa saat membaca atau menulis sebelum mencapai nilai tersebut. Ini kemungkinan karena partisi panas. Meskipun Amazon Keyspaces dapat terus melakukan penskalaan untuk mendukung throughput yang hampir tidak terbatas, setiap partisi individu dibatasi hingga 1.000 unit tulis per detik dan 3.000 unit baca per detik. Jika aplikasi Anda mengarahkan terlalu banyak lalu lintas ke sebagian kecil partisi tabel, kapasitas melebihi peristiwa dapat terjadi bahkan sebelum Anda mencapai nilai throughput hangat tabel. Sebaiknya ikuti [praktik terbaik Amazon Keyspaces](#) untuk memastikan skalabilitas yang mulus dan menghindari partisi panas.

### Throughput hangat untuk tabel yang disediakan

Pertimbangkan tabel yang disediakan yang memiliki throughput hangat 30.000 unit baca per detik dan 10.000 unit tulis per detik tetapi saat ini memiliki throughput yang disediakan sebesar 4.000 RCU dan 8.000 WCU. Anda dapat langsung menskalakan throughput yang disediakan tabel hingga 30.000 RCU atau 10.000 WCU dengan memperbarui pengaturan throughput yang disediakan. Saat Anda meningkatkan throughput yang disediakan di luar nilai-nilai ini, throughput hangat menyesuaikan secara otomatis ke nilai baru yang lebih tinggi, karena Anda telah menetapkan throughput puncak baru. Misalnya, jika Anda menetapkan throughput yang disediakan ke 50.000 RCU, throughput hangat meningkat menjadi 50.000 unit baca per detik.

```
"ProvisionedThroughput":
 {
 "ReadCapacityUnits": 4000,
 "WriteCapacityUnits": 8000
```

```
}
"WarmThroughput":
{
 "ReadUnitsPerSecond": 30000,
 "WriteUnitsPerSecond": 10000
}
```

## Throughput hangat untuk meja sesuai permintaan

Tabel on-demand baru dimulai dengan throughput hangat 12.000 unit baca per detik dan 4.000 unit tulis per detik. Meja Anda dapat langsung mengakomodasi lalu lintas berkelanjutan hingga level ini. Jika permintaan Anda melebihi 12.000 unit baca per detik atau 4.000 unit tulis per detik, throughput hangat menyesuaikan secara otomatis ke nilai yang lebih tinggi.

```
"WarmThroughput":
{
 "ReadUnitsPerSecond": 12000,
 "WriteUnitsPerSecond": 4000
}
```

## Praktik terbaik untuk tabel Amazon Keyspaces pra-pemanasan

Ikuti praktik terbaik ini saat menerapkan pra-pemanasan untuk tabel Amazon Keyspaces Anda:

### Memperkirakan kapasitas yang dibutuhkan secara akurat

Karena pra-pemanasan menimbulkan biaya satu kali, hitung dengan cermat throughput yang dibutuhkan berdasarkan beban kerja yang diharapkan untuk menghindari penyediaan berlebih.

### Pertimbangkan skema tabel

Tabel dengan baris yang lebih besar mungkin memerlukan lebih banyak partisi untuk throughput yang sama. Faktor ukuran baris rata-rata Anda saat memperkirakan persyaratan pra-pemanasan.

### Pantau kinerja meja

Setelah pemanasan awal, gunakan CloudWatch metrik untuk memverifikasi bahwa tabel Anda menangani beban seperti yang diharapkan. Untuk informasi selengkapnya, lihat [the section called "Pantau pra-pemanasan dengan CloudWatch"](#).

## Kelola kuota

Jika aplikasi Anda memerlukan throughput yang lebih tinggi daripada kuota default yang diizinkan (40.000 RCUs/WCUs atau 2.000 partisi), kuota permintaan meningkat jauh sebelum peristiwa lalu lintas tinggi Anda. Untuk meminta peningkatan kuota, gunakan konsol [Service Quotas](#).

## Optimalkan biaya

Untuk peristiwa lalu lintas tinggi sementara, pertimbangkan untuk menggunakan pra-pemanasan alih-alih beralih ke mode yang disediakan dengan kapasitas tinggi, karena mungkin lebih hemat biaya untuk acara berdurasi pendek. Untuk informasi selengkapnya tentang harga, lihat harga [Amazon Keyspaces](#).

### Note

Pantau metrik kinerja aplikasi Anda selama fase pengujian untuk memvalidasi bahwa konfigurasi pra-pemanasan mendukung persyaratan beban kerja Anda secara memadai.

## Topik

- [Membuat tabel Amazon Keyspaces baru dengan throughput hangat yang lebih tinggi](#)
- [Meningkatkan throughput hangat tabel Amazon Keyspaces yang ada](#)
- [Lihat throughput hangat dari tabel Amazon Keyspaces](#)
- [Pantau kinerja tabel pra-pemanasan menggunakan Amazon CloudWatch](#)

## Membuat tabel Amazon Keyspaces baru dengan throughput hangat yang lebih tinggi

Anda dapat menyesuaikan nilai throughput hangat saat membuat tabel Amazon Keyspaces menggunakan konsol, CQL, atau tabel. AWS CLI

## Console

Cara membuat tabel baru dengan pengaturan throughput hangat

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih Buat tabel.

3. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel baru.
4. Di bagian Kolom, buat skema untuk tabel Anda.
5. Di bagian kunci Primer, tentukan kunci utama tabel dan pilih kolom pengelompokan opsional.
6. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
7. Lanjutkan ke pengaturan Read/write kapasitas.
8. Untuk mode Kapasitas, Anda dapat memilih salah satu On-demand atau Disediakan.
9. Di bagian Pre-warming untuk tabel, Anda dapat meningkatkan nilai untuk unit Baca per detik dan Tulis unit per detik sesuai kebutuhan untuk mempersiapkan tabel Anda untuk menangani peristiwa puncak yang direncanakan.

Nilai throughput hangat yang disesuaikan Amazon Keyspaces berdasarkan penggunaan sesuai permintaan atau kapasitas yang disediakan tersedia secara default untuk semua tabel tanpa biaya tambahan. Perhatikan bahwa jika Anda secara manual meningkatkan nilai throughput hangat default untuk menghangatkan tabel untuk peristiwa lalu lintas puncak, biaya tambahan akan berlaku.

10. Konfigurasi fitur tabel opsional lainnya sesuai kebutuhan. Kemudian pilih Buat tabel.

## Cassandra Query Language (CQL)

- Buat tabel dengan throughput hangat menggunakan salah satu metode berikut:
  - Untuk mode yang disediakan, buat tabel dan tentukan kapasitas puncak yang diharapkan untuk membaca dan menulis menggunakan sintaks CQL berikut:

```
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
 }
};
```

- Untuk mode sesuai permintaan, buat tabel dan tentukan kapasitas puncak yang diharapkan untuk membaca dan menulis menggunakan sintaks CQL berikut:

```
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
 }
};
```

Untuk mengonfirmasi pengaturan kapasitas tabel, lihat [the section called “Lihat throughput hangat meja Anda”](#).

## CLI

1. Buat tabel dengan throughput hangat menggunakan salah satu metode berikut menggunakan AWS CLI
  - Buat tabel baru dalam mode yang disediakan dan tentukan nilai kapasitas puncak yang diharapkan untuk membaca dan menulis untuk tabel baru. Pernyataan berikut adalah contohnya.

```
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
```

- Buat tabel baru dalam mode sesuai permintaan dan tentukan nilai kapasitas puncak yang diharapkan untuk membaca dan menulis untuk tabel baru. Pernyataan berikut adalah contohnya.

```
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. Output dari perintah mengembalikan ARN dari tabel seperti yang ditunjukkan pada contoh berikut.

```
{
 "resourceArn": "arn:aws::cassandra:us-east-1:111122223333:/keyspace/catalog/
table/book_awards>"
}
```

Untuk mengonfirmasi pengaturan kapasitas tabel, lihat [the section called “Lihat throughput hangat meja Anda”](#).

## Java

Buat tabel baru menggunakan SDK for Java.

- Buat tabel baru dalam mode yang disediakan dan tentukan nilai kapasitas puncak yang diharapkan untuk membaca dan menulis untuk tabel baru. Contoh kode berikut adalah contoh dari ini.

```
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());
 }
}

```

## Meningkatkan throughput hangat tabel Amazon Keyspaces yang ada

Anda dapat meningkatkan nilai throughput hangat tabel Amazon Keyspaces saat ini menggunakan konsol, CQL, atau AWS CLI

### Console

Cara meningkatkan pengaturan pra-hangat tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih tabel yang ingin Anda perbarui.
3. Pada tab Kapasitas tabel, lanjutkan ke Pre-warming untuk tabel.
4. Di bagian Pre-warming untuk tabel, pilih Edit.



5. Pada halaman Edit pra-pemanasan untuk tabel, Anda dapat memperbarui nilai untuk unit Baca per detik dan untuk unit Tulis per detik.
6. Pilih Simpan perubahan. Tabel Anda diperbarui dengan pengaturan pra-pemanasan yang ditentukan.

## Cassandra Query Language (CQL)

Tingkatkan pengaturan throughput hangat dari tabel menggunakan CQL

- Gunakan ALTER TABLE pernyataan untuk meningkatkan throughput hangat dari sebuah tabel. Pernyataan berikut adalah contohnya.

```
ALTER TABLE catalog.book_awards
WITH CUSTOM_PROPERTIES = {
 'warm_throughput': {
 'read_units_per_second': 60000,
 'write_units_per_second': 30000
 }
};
```

Untuk mengonfirmasi pengaturan kapasitas tabel yang diperbarui, lihat [the section called “Lihat throughput hangat meja Anda”](#).

## CLI

Tingkatkan pengaturan pra-pemanasan tabel menggunakan AWS CLI

- Untuk meningkatkan throughput hangat tabel, Anda dapat menggunakan perintah. update-table Pernyataan berikut adalah contohnya.

```
aws keyspaces update-table \
--keyspace-name 'catalog' \
--table-name 'book_awards' \
--warmThroughputSpecification readUnitsPerSecond=60000,writeUnitsPerSecond=30000
```

Untuk mengonfirmasi pengaturan kapasitas tabel yang diperbarui, lihat [the section called “Lihat throughput hangat meja Anda”](#).

## Java

Perbarui pengaturan pra-pemanasan tabel menggunakan SDK for Java.

- Perbarui pengaturan throughput hangat untuk tabel. Contoh kode berikut adalah contoh dari ini.

```
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());
 }
}
```

## Lihat throughput hangat dari tabel Amazon Keyspaces

Anda dapat melihat nilai throughput hangat tabel Amazon Keyspaces saat ini menggunakan konsol, CQL, atau AWS CLI

## Console

Cara melihat pengaturan pra-pemanasan tabel Anda menggunakan konsol.

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih tabel yang ingin Anda tinjau.
3. Pada tab Kapasitas tabel, lanjutkan ke Pre-warming untuk tabel.

## Cassandra Query Language (CQL)

Melihat pengaturan throughput hangat dari tabel menggunakan CQL

- Untuk melihat pengaturan hangat-throughput tabel, Anda dapat menggunakan pernyataan CQL berikut.

```
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

Melihat pengaturan throughput hangat dari tabel menggunakan AWS CLI

- Anda dapat melihat pengaturan hangat-throughput tabel menggunakan `get-table` perintah seperti yang ditunjukkan pada contoh berikut.

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

Berikut ini menunjukkan contoh output dari `get-table` perintah untuk tabel Single-region dalam mode yang disediakan.

```
{
 "keyspaceName": "catalog",
 "tableName": "book_awards",
 ... Existing Fields ...,
 "capacitySpecificationSummary": {
 "throughputMode": "PROVISIONED",
 "readCapacityUnits": 20000,
 "writeCapacityUnits": 10000
 },
 "warmThroughputSpecificationSummary": {
 "readUnitsPerSecond": 40000,
 "writeUnitsPerSecond": 20000,
 "status": "AVAILABLE"
 }
}
```

Berikut ini menunjukkan contoh output untuk tabel Single-region dalam mode on-demand.

```
{
 "keyspaceName": "catalog",
 "tableName": "book_awards_ondemand",
 ... Existing Fields ...,
 "capacitySpecification": {
 "throughputMode": "PAY_PER_REQUEST"
 },
 "warmThroughputSpecificationSummary": {
 "readUnitsPerSecond": 40000,
 "writeUnitsPerSecond": 20000,
 "status": "AVAILABLE"
 }
}
```

## Java

Baca pengaturan pra-pemanasan tabel menggunakan SDK for Java.

- Baca nilai throughput hangat dari tabel menggunakan `get-table` Contoh kode berikut adalah contoh dari ini.

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

```
}
}
}
```

## Pantau kinerja tabel pra-pemanasan menggunakan Amazon CloudWatch

Amazon Keyspaces pra-pemanasan tidak memperkenalkan CloudWatch metrik baru, tetapi Anda dapat memantau kinerja tabel pra-pemanasan menggunakan metrik Amazon Keyspaces yang ada:

### SuccessfulRequestLatency

Pantau metrik ini untuk memverifikasi bahwa tabel pra-pemanasan menangani permintaan dengan latensi yang diharapkan.

### WriteThrottleEvents dan ReadThrottleEvents

Metrik ini harus tetap rendah untuk tabel pra-pemanasan yang benar. Jika Anda melihat kesalahan kapasitas yang tidak mencukupi meskipun pra-pemanasan, Anda mungkin perlu menyesuaikan nilai throughput hangat Anda.

### ConsumedReadCapacityUnits dan ConsumedWriteCapacityUnits

Metrik ini menunjukkan konsumsi kapasitas aktual, yang dapat membantu memvalidasi jika konfigurasi pra-pemanasan Anda sesuai.

### ProvisionedReadCapacityUnits dan ProvisionedWriteCapacityUnits

Untuk tabel yang disediakan, metrik ini menunjukkan kapasitas yang saat ini dialokasikan.

Metrik ini dapat dilihat di CloudWatch konsol atau ditanyakan menggunakan API. CloudWatch Lihat informasi yang lebih lengkap di [the section called “Pemantauan CloudWatch dengan”](#).

## Kelola kapasitas throughput secara otomatis dengan penskalaan otomatis Amazon Keyspaces

Banyak beban kerja basis data yang memiliki sifat berhubungan dengan siklus atau sulit untuk diprediksi. Misalnya, pertimbangkan aplikasi jaringan sosial yang sebagian besar penggunaanya aktif selama waktu siang hari. Basis data harus mampu menangani aktivitas siang hari, tetapi tidak perlu untuk tingkat throughput yang sama pada malam hari.

Contoh lainnya mungkin berupa aplikasi game seluler baru yang mengalami adopsi cepat. Jika permainan menjadi sangat populer, itu bisa melebihi sumber daya database yang tersedia, yang akan menghasilkan kinerja yang lambat dan pelanggan yang tidak bahagia. Beban kerja semacam ini kerap perlu dinaikkan atau dikurangi skala sumber daya basis datanya secara manual guna merespons berbagai tingkat penggunaan.

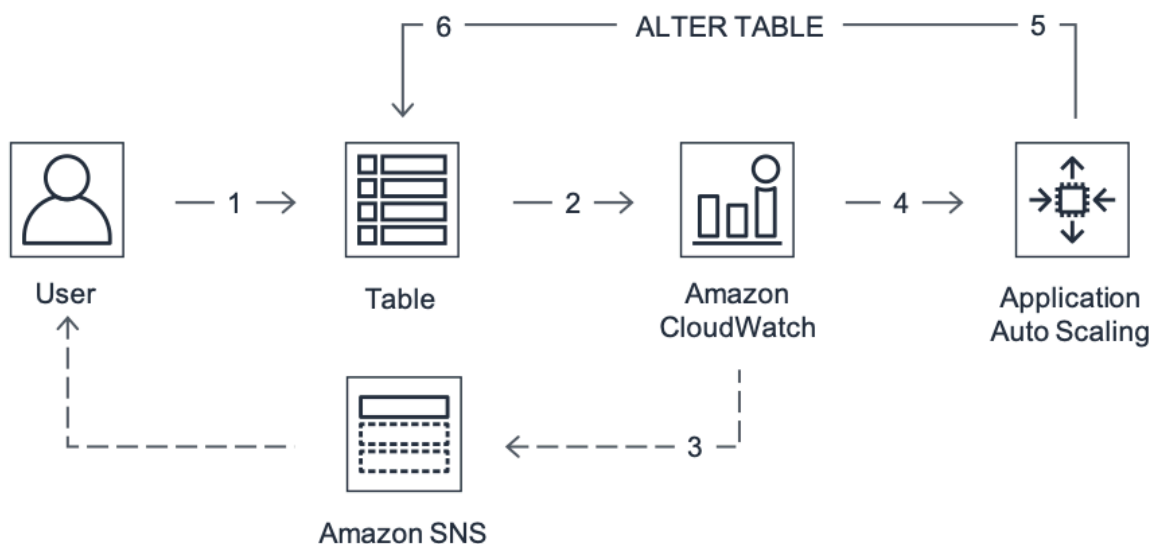
Amazon Keyspaces (untuk Apache Cassandra) membantu Anda menyediakan kapasitas throughput secara efisien untuk beban kerja variabel dengan menyesuaikan kapasitas throughput secara otomatis sebagai respons terhadap lalu lintas aplikasi yang sebenarnya. Amazon Keyspaces menggunakan layanan Application Auto Scaling untuk menambah dan mengurangi kapasitas baca dan tulis tabel atas nama Anda. Untuk informasi selengkapnya tentang Application Auto Scaling, lihat Panduan Pengguna [Application Auto Scaling](#).

#### Note

Untuk memulai penskalaan otomatis Amazon Keyspaces dengan cepat, lihat [the section called “Konfigurasi dan perbarui kebijakan penskalaan otomatis”](#)

## Cara kerja penskalaan otomatis Amazon Keyspaces

Diagram berikut memberikan ikhtisar tingkat tinggi tentang cara penskalaan otomatis Amazon Keyspaces mengelola kapasitas throughput untuk sebuah tabel.



Untuk mengaktifkan penskalaan otomatis untuk tabel, Anda membuat kebijakan penskalaan. Kebijakan penskalaan menentukan apakah Anda ingin menskalakan kapasitas baca atau kapasitas tulis (atau keduanya), dan pengaturan unit kapasitas minimum dan maksimum yang disediakan untuk tabel.

Kebijakan penskalaan juga mendefinisikan pemanfaatan target. Pemanfaatan target adalah rasio unit kapasitas yang dikonsumsi terhadap unit kapasitas yang disediakan pada suatu titik waktu, dinyatakan sebagai persentase. Penskalaan otomatis menggunakan algoritme pelacakan target untuk menyesuaikan throughput tabel yang disediakan ke atas atau ke bawah sebagai respons terhadap beban kerja aktual. Hal ini dilakukan agar pemanfaatan kapasitas aktual tetap pada atau dekat target pemanfaatan Anda.

Anda dapat mengatur nilai pemanfaatan target penskalaan otomatis antara 20 dan 90 persen untuk kapasitas baca dan tulis Anda. Tingkat pemanfaatan target default adalah 70 persen. Anda dapat menetapkan target pemanfaatan menjadi persentase yang lebih rendah jika lalu lintas Anda berubah dengan cepat dan Anda ingin kapasitas untuk mulai meningkatkan lebih cepat. Anda juga dapat menetapkan tingkat pemanfaatan target ke tingkat yang lebih tinggi jika lalu lintas aplikasi Anda berubah lebih lambat dan Anda ingin mengurangi biaya throughput.

Untuk informasi selengkapnya tentang kebijakan penskalaan, lihat [Kebijakan penskalaan pelacakan target untuk Application Auto Scaling](#) di Panduan Pengguna [Application Auto Scaling](#).

Saat Anda membuat kebijakan penskalaan, Amazon Keyspaces membuat dua pasang alarm CloudWatch Amazon atas nama Anda. Setiap pasangan mewakili batas atas dan bawah Anda untuk pengaturan throughput yang disediakan dan dikonsumsi. CloudWatch Alarm ini dipicu ketika penggunaan tabel yang sebenarnya menyimpang dari penggunaan target Anda untuk jangka waktu yang berkelanjutan. Untuk mempelajari lebih lanjut tentang Amazon CloudWatch, lihat [Panduan CloudWatch Pengguna Amazon](#).

Ketika salah satu CloudWatch alarm dipicu, Amazon Simple Notification Service (Amazon SNS) mengirimkan pemberitahuan kepada Anda (jika Anda telah mengaktifkannya). CloudWatch Alarm kemudian memanggil Application Auto Scaling untuk mengevaluasi kebijakan penskalaan Anda. Hal ini pada gilirannya mengeluarkan permintaan Alter Table ke Amazon Keyspaces untuk menyesuaikan kapasitas yang disediakan tabel ke atas atau ke bawah sebagaimana mestinya. Untuk mempelajari lebih lanjut tentang notifikasi Amazon SNS, lihat [Menyiapkan notifikasi Amazon SNS](#).

Amazon Keyspaces memproses permintaan Alter Table dengan meningkatkan (atau mengurangi) kapasitas throughput yang disediakan tabel sehingga mendekati pemanfaatan target Anda.



 Note

Penskalaan otomatis Amazon Keyspaces memodifikasi setelan throughput yang disediakan hanya jika beban kerja sebenarnya tetap meningkat (atau tertekan) selama beberapa menit. Algoritma pelacakan target berusaha untuk menjaga pemanfaatan target pada atau mendekati nilai yang Anda pilih dalam jangka panjang. Lonjakan aktivitas yang mendadak dan berdurasi singkat diakomodasi oleh kapasitas lonjakan bawaan tabel.

## Cara kerja penskalaan otomatis untuk tabel Multi-wilayah

Untuk memastikan bahwa selalu ada kapasitas baca dan tulis yang cukup untuk semua replika tabel di semua Wilayah AWS tabel Multi-wilayah dalam mode kapasitas yang disediakan, sebaiknya Anda mengonfigurasi penskalaan otomatis Amazon Keyspaces.

Saat Anda menggunakan tabel Multi-wilayah dalam mode yang disediakan dengan penskalaan otomatis, Anda tidak dapat menonaktifkan penskalaan otomatis untuk satu replika tabel. Tetapi Anda dapat menyesuaikan pengaturan penskalaan otomatis baca tabel untuk Wilayah yang berbeda. Misalnya, Anda dapat menentukan kapasitas baca yang berbeda dan membaca pengaturan penskalaan otomatis untuk setiap Wilayah tempat tabel direplikasi.

Pengaturan penskalaan otomatis baca yang Anda konfigurasikan untuk replika tabel di Wilayah tertentu menimpa pengaturan penskalaan otomatis umum tabel. Kapasitas tulis, bagaimanapun, harus tetap disinkronkan di semua replika tabel untuk memastikan bahwa ada kapasitas yang cukup untuk mereplikasi penulisan di semua Wilayah.

Penskalaan otomatis Amazon Keyspaces secara independen memperbarui kapasitas tabel yang disediakan di masing-masing Wilayah AWS berdasarkan penggunaan di Wilayah tersebut. Akibatnya, kapasitas yang disediakan di setiap Wilayah untuk tabel Multi-wilayah mungkin berbeda saat penskalaan otomatis aktif.

Anda dapat mengonfigurasi pengaturan penskalaan otomatis tabel Multi-wilayah dan replika menggunakan konsol Amazon Keyspaces, API, atau CQL. AWS CLI Untuk informasi selengkapnya tentang cara membuat dan memperbarui pengaturan penskalaan otomatis untuk tabel Multi-wilayah, lihat. [the section called “Memperbarui pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”](#)

 Note

Jika Anda menggunakan penskalaan otomatis untuk tabel Multi-wilayah, Anda harus selalu menggunakan operasi Amazon Keyspaces API untuk mengonfigurasi pengaturan penskalaan otomatis. Jika Anda menggunakan operasi Application Auto Scaling API secara langsung untuk mengonfigurasi pengaturan penskalaan otomatis, Anda tidak memiliki kemampuan untuk menentukan tabel Wilayah AWS Multi-wilayah. Hal ini dapat mengakibatkan konfigurasi yang tidak didukung.

## Catatan penggunaan

Sebelum Anda mulai menggunakan penskalaan otomatis Amazon Keyspaces, Anda harus mengetahui hal berikut:

- Penskalaan otomatis Amazon Keyspaces tidak tersedia di Wilayah Timur Tengah (UEA).
- Penskalaan otomatis Amazon Keyspaces dapat meningkatkan kapasitas baca atau kapasitas tulis sesering yang diperlukan, sesuai dengan kebijakan penskalaan Anda. Semua kuota Amazon Keyspaces tetap berlaku, seperti yang dijelaskan dalam [Kuota](#)
- Penskalaan otomatis Amazon Keyspaces tidak mencegah Anda memodifikasi setelan throughput yang disediakan secara manual. Penyesuaian manual ini tidak memengaruhi CloudWatch alarm yang ada yang melekat pada kebijakan penskalaan.
- Jika Anda menggunakan konsol untuk membuat tabel dengan kapasitas throughput yang disediakan, penskalaan otomatis Amazon Keyspaces diaktifkan secara default. Anda dapat mengubah pengaturan penskalaan otomatis Anda kapan saja. Untuk informasi selengkapnya, lihat [the section called “Matikan penskalaan otomatis Amazon Keyspaces untuk sebuah tabel”](#).
- Jika Anda menggunakan CloudFormation untuk membuat kebijakan penskalaan, Anda harus mengelola kebijakan penskalaan CloudFormation agar tumpukan disinkronkan dengan templat tumpukan. Jika Anda mengubah kebijakan penskalaan dari Amazon Keyspaces, kebijakan tersebut akan ditimpa dengan nilai asli dari CloudFormation templat tumpukan saat tumpukan disetel ulang.
- Jika Anda menggunakannya CloudTrail untuk memantau penskalaan otomatis Amazon Keyspaces, Anda mungkin melihat peringatan untuk panggilan yang dilakukan oleh Application Auto Scaling sebagai bagian dari proses validasi konfigurasinya. Anda dapat memfilter peringatan ini dengan menggunakan `invokedBy` bidang, yang berisi pemeriksaan `application-autoscaling.amazonaws.com` validasi ini.

# Konfigurasi dan perbarui kebijakan penskalaan otomatis Amazon Keyspaces

Anda dapat menggunakan konsol, CQL, atau AWS Command Line Interface (AWS CLI) untuk mengonfigurasi penskalaan otomatis Amazon Keyspaces untuk tabel baru dan yang sudah ada. Anda juga dapat memodifikasi pengaturan penskalaan otomatis atau menonaktifkan penskalaan otomatis.

Untuk fitur yang lebih canggih seperti menyetel waktu cooldown scale-in dan scale-out, sebaiknya gunakan CQL atau untuk AWS CLI mengelola kebijakan penskalaan Amazon Keyspaces.

## Topik

- [Konfigurasi izin untuk penskalaan otomatis Amazon Keyspaces](#)
- [Buat tabel baru dengan penskalaan otomatis](#)
- [Konfigurasi penskalaan otomatis pada tabel yang ada](#)
- [Lihat konfigurasi penskalaan otomatis Amazon Keyspaces tabel Anda](#)
- [Matikan penskalaan otomatis Amazon Keyspaces untuk sebuah tabel](#)
- [Melihat aktivitas penskalaan otomatis untuk tabel Amazon Keyspaces di Amazon CloudWatch](#)

## Konfigurasi izin untuk penskalaan otomatis Amazon Keyspaces

Untuk memulai, konfirmasi bahwa prinsipal memiliki izin yang sesuai untuk membuat dan mengelola pengaturan penskalaan otomatis. Di AWS Identity and Access Management (IAM), kebijakan AWS terkelola `AmazonKeyspacesFullAccess` diperlukan untuk mengelola kebijakan penskalaan Amazon Keyspaces.

### Important

`application-autoscaling`: \*izin diperlukan untuk menonaktifkan penskalaan otomatis di atas meja. Anda harus mematikan penskalaan otomatis untuk tabel sebelum Anda dapat menghapusnya.

Untuk menyiapkan pengguna atau peran IAM untuk akses konsol Amazon Keyspaces dan penskalaan otomatis Amazon Keyspaces, tambahkan kebijakan berikut.

## Untuk melampirkan **AmazonKeyspacesFullAccess** kebijakan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di dasbor konsol IAM, pilih Pengguna, lalu pilih pengguna atau peran IAM Anda dari daftar.
3. Di halaman Ringkasan, pilih Tambahkan izin.
4. Pilih Lampirkan kebijakan yang sudah ada secara langsung.
5. Dari daftar kebijakan, pilih AmazonKeyspacesFullAccess, lalu pilih Berikutnya: Tinjau.
6. Pilih Tambahkan izin.

## Buat tabel baru dengan penskalaan otomatis

Saat membuat tabel Amazon Keyspaces baru, Anda dapat secara otomatis mengaktifkan penskalaan otomatis untuk kapasitas tulis atau baca tabel. Hal ini memungkinkan Amazon Keyspaces menghubungi Application Auto Scaling atas nama Anda untuk mendaftarkan tabel sebagai target yang dapat diskalakan dan menyesuaikan kapasitas tulis atau baca yang disediakan.

Untuk informasi selengkapnya tentang cara membuat tabel Multi-wilayah dan mengonfigurasi pengaturan penskalaan otomatis yang berbeda untuk replika tabel, lihat [the section called “Buat tabel Multi-wilayah dalam mode yang disediakan”](#)

### Note

Penskalaan otomatis Amazon Keyspaces memerlukan keberadaan peran terkait layanan (AWSServiceRoleForApplicationAutoScaling\_CassandraTable) yang melakukan tindakan penskalaan otomatis atas nama Anda. Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Menggunakan Peran Terkait Layanan”](#).

## Console

Buat tabel baru dengan penskalaan otomatis diaktifkan menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih Buat tabel.

3. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel baru.
4. Di bagian Kolom, buat skema untuk tabel Anda.
5. Di bagian kunci Primer, tentukan kunci utama tabel dan pilih kolom pengelompokan opsional.
6. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
7. Lanjutkan ke pengaturan Read/write kapasitas.
8. Untuk mode Kapasitas, pilih Disediakan.
9. Di bagian Baca kapasitas, konfirmasi bahwa Skala dipilih secara otomatis.

Pada langkah ini, Anda memilih unit kapasitas baca minimum dan maksimum untuk tabel, serta pemanfaatan target.

- Unit kapasitas minimum — Masukkan nilai untuk tingkat throughput minimum yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimum per detik untuk akun Anda (40.000 secara default).
- Unit kapasitas maksimum — Masukkan jumlah maksimum throughput yang ingin Anda sediakan untuk tabel. Nilai harus antara 1 dan kuota throughput maksimum per detik untuk akun Anda (40.000 secara default).
- Target pemanfaatan — Masukkan tingkat pemanfaatan target antara 20% dan 90%. Ketika lalu lintas melebihi tingkat pemanfaatan target yang ditentukan, kapasitas secara otomatis ditingkatkan. Ketika lalu lintas jatuh di bawah target yang ditentukan, secara otomatis diperkecil lagi.

 Note

Untuk mempelajari lebih lanjut tentang kuota default untuk akun Anda dan cara meningkatkannya, lihat [Kuota](#).

10. Di bagian Tulis kapasitas, pilih pengaturan yang sama seperti yang ditentukan pada langkah sebelumnya untuk kapasitas baca, atau konfigurasi nilai kapasitas secara manual.
11. Pilih Buat tabel. Tabel Anda dibuat dengan parameter penskalaan otomatis yang ditentukan.

## Cassandra Query Language (CQL)

Buat tabel baru dengan penskalaan otomatis Amazon Keyspaces menggunakan CQL

Untuk mengonfigurasi pengaturan penskalaan otomatis untuk tabel secara terprogram, Anda menggunakan `AUTOSCALING_SETTINGS` pernyataan yang berisi parameter untuk penskalaan otomatis Amazon Keyspaces. Parameter menentukan kondisi yang mengarahkan Amazon Keyspaces untuk menyesuaikan throughput yang disediakan tabel Anda, dan tindakan opsional tambahan apa yang harus diambil. Dalam contoh ini, Anda menentukan pengaturan penskalaan otomatis untuk `mytable`.

Kebijakan tersebut berisi elemen berikut:

- `AUTOSCALING_SETTINGS`— Menentukan apakah Amazon Keyspaces diizinkan untuk menyesuaikan kapasitas throughput atas nama Anda. Nilai-nilai berikut diperlukan:
  - `provisioned_write_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `provisioned_read_capacity_autoscaling_update`:
    - `minimum_units`
    - `maximum_units`
  - `scaling_policy`— Amazon Keyspaces mendukung kebijakan pelacakan target. Untuk menentukan kebijakan pelacakan target, Anda mengonfigurasi parameter berikut.
    - `target_value`— Penskalaan otomatis Amazon Keyspaces memastikan bahwa rasio kapasitas yang dikonsumsi terhadap kapasitas yang disediakan tetap pada atau mendekati nilai ini. Anda mendefinisikan `target_value` sebagai persentase.
    - `disableScaleIn`: (Opsional) A boolean yang menentukan `scale-in` apakah dinonaktifkan atau diaktifkan untuk tabel. Parameter ini dinonaktifkan secara default. Untuk menghidupkan `scale-in`, atur boolean nilainya ke `FALSE`. Ini berarti bahwa kapasitas secara otomatis diperkecil untuk tabel atas nama Anda.
    - `scale_out_cooldown`— Aktivitas `scale-out` meningkatkan throughput yang disediakan dari tabel Anda. Untuk menambahkan periode cooldown untuk aktivitas `scale-out`, tentukan nilai, dalam detik, untuk `scale_out_cooldown`. Jika Anda tidak menentukan nilai, nilai defaultnya adalah 0. Untuk informasi selengkapnya tentang pelacakan target dan periode cooldown, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).
    - `scale_in_cooldown`— Aktivitas `scale-in` mengurangi throughput yang disediakan dari tabel Anda. Untuk menambahkan periode cooldown untuk aktivitas penskalaan, tentukan nilai, dalam detik, untuk `scale_in_cooldown`. Jika Anda tidak menentukan nilai, nilai

defaultnya adalah 0. Untuk informasi selengkapnya tentang pelacakan target dan periode cooldown, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).

 Note

Untuk lebih memahami cara kerja `target_value`, misalkan Anda memiliki sebuah tabel dengan pengaturan throughput yang disediakan sebanyak 200 unit kapasitas tulis. Anda memutuskan untuk membuat kebijakan penskalaan untuk tabel ini, dengan `target_value` sebesar 70 persen.

Sekarang anggaplah Anda mulai mendorong lalu lintas tulis ke tabel sehingga throughput tulis aktual adalah sebesar 150 unit kapasitas. Rasio konsumsi terhadap penyediaan saat ini ( $150/200$ ), atau 75 persen. Rasio ini melebihi target Anda, jadi penskalaan otomatis meningkatkan kapasitas tulis yang disediakan menjadi 215 sehingga rasionya ( $150/215$ ), atau 69,77 persen—sedekat mungkin dengan Anda, tetapi tidak melebihi itu. `target_value`

Untuk `mytable`, Anda mengatur `TargetValue` kapasitas baca dan tulis hingga 50 persen. Penskalaan otomatis Amazon Keyspaces menyesuaikan throughput yang disediakan tabel dalam kisaran 5-10 unit kapasitas sehingga rasio konsumsi-ke-penyediaan tetap pada atau mendekati 50 persen. Untuk kapasitas baca, Anda mengatur nilai untuk `ScaleOutCooldown` dan `ScaleInCooldown` ke 60 detik.

Anda dapat menggunakan pernyataan berikut untuk membuat tabel Amazon Keyspaces baru dengan penskalaan otomatis diaktifkan.

```
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

Buat tabel baru dengan penskalaan otomatis Amazon Keyspaces menggunakan AWS CLI

Untuk mengonfigurasi pengaturan penskalaan otomatis untuk tabel secara terprogram, Anda menggunakan `autoScalingSpecification` tindakan yang menentukan parameter untuk penskalaan otomatis Amazon Keyspaces. Parameter menentukan kondisi yang mengarahkan Amazon Keyspaces untuk menyesuaikan throughput yang disediakan tabel Anda, dan tindakan opsional tambahan apa yang harus diambil. Dalam contoh ini, Anda menentukan pengaturan penskalaan otomatis untuk mytable.

Kebijakan tersebut berisi elemen berikut:

- **autoScalingSpecification**— Menentukan apakah Amazon Keyspaces diizinkan untuk menyesuaikan throughput kapasitas atas nama Anda. Anda dapat mengaktifkan penskalaan otomatis untuk membaca dan untuk kapasitas tulis secara terpisah. Maka Anda harus menentukan parameter berikut untuk `autoScalingSpecification`:
  - **writeCapacityAutoScaling**— Unit kapasitas tulis maksimum dan minimum.
  - **readCapacityAutoScaling**— Unit kapasitas baca maksimum dan minimum.
  - **scalingPolicy**— Amazon Keyspaces mendukung kebijakan pelacakan target. Untuk menentukan kebijakan pelacakan target, Anda mengonfigurasi parameter berikut.



- **targetValue**— Penskalaan otomatis Amazon Keyspaces memastikan bahwa rasio kapasitas yang dikonsumsi terhadap kapasitas yang disediakan tetap pada atau mendekati nilai ini. Anda mendefinisikan **targetValue** sebagai persentase.
- **disableScaleIn**: (Opsional) A boolean yang menentukan **scale-in** apakah dinonaktifkan atau diaktifkan untuk tabel. Parameter ini dinonaktifkan secara default. Untuk menghidupkan **scale-in**, atur boolean nilainya ke **FALSE**. Ini berarti bahwa kapasitas secara otomatis diperkecil untuk tabel atas nama Anda.
- **scaleOutCooldown**— Aktivitas **scale-out** meningkatkan throughput yang disediakan dari tabel Anda. Untuk menambahkan periode **cooldown** untuk aktivitas **scale-out**, tentukan nilai, dalam detik, untuk **ScaleOutCooldown**. Nilai default-nya adalah 0. Untuk informasi selengkapnya tentang pelacakan target dan periode **cooldown**, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).
- **scaleInCooldown**— Aktivitas **scale-in** mengurangi throughput yang disediakan dari tabel Anda. Untuk menambahkan periode **cooldown** untuk aktivitas penskalaan, tentukan nilai, dalam detik, untuk **ScaleInCooldown**. Nilai default-nya adalah 0. Untuk informasi selengkapnya tentang pelacakan target dan periode **cooldown**, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).

#### Note

Untuk lebih memahami cara kerja **TargetValue**, misalkan Anda memiliki sebuah tabel dengan pengaturan throughput yang disediakan sebanyak 200 unit kapasitas tulis. Anda memutuskan untuk membuat kebijakan penskalaan untuk tabel ini, dengan **TargetValue** sebesar 70 persen.

Sekarang anggaplah Anda mulai mendorong lalu lintas tulis ke tabel sehingga throughput tulis aktual adalah sebesar 150 unit kapasitas. Rasio konsumsi terhadap penyediaan saat ini ( $150/200$ ), atau 75 persen. Rasio ini melebihi target Anda, jadi penskalaan otomatis meningkatkan kapasitas tulis yang disediakan menjadi 215 sehingga rasionya ( $150/215$ ), atau 69,77 persen—sedekat mungkin dengan Anda, tetapi tidak melebihi itu. **TargetValue**

Untuk **mytable**, Anda mengatur **TargetValue** kapasitas baca dan tulis hingga 50 persen. Penskalaan otomatis Amazon Keyspaces menyesuaikan throughput yang disediakan tabel dalam kisaran 5-10 unit kapasitas sehingga rasio konsumsi-ke-penyediaan tetap pada atau

mendekati 50 persen. Untuk kapasitas baca, Anda mengatur nilai untuk `ScaleOutCooldown` dan `ScaleInCooldown` ke 60 detik.

Saat membuat tabel dengan pengaturan penskalaan otomatis yang kompleks, akan sangat membantu untuk memuat pengaturan penskalaan otomatis dari file JSON. Untuk contoh berikut, Anda dapat men-download contoh file JSON dari [auto-scaling.zip](#) dan ekstrak `auto-scaling.json`, mencatat path ke file. Dalam contoh ini, file JSON terletak di direktori saat ini. Untuk opsi jalur file yang berbeda, lihat [Cara memuat parameter dari file](#).

```
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
```

## Konfigurasi penskalaan otomatis pada tabel yang ada

Anda dapat memperbarui tabel Amazon Keyspaces yang ada untuk mengaktifkan penskalaan otomatis untuk kapasitas tulis atau baca tabel. Jika Anda memperbarui tabel yang saat ini dalam mode kapasitas sesuai permintaan, Anda harus terlebih dahulu mengubah mode kapasitas tabel ke mode kapasitas yang disediakan.

Untuk informasi selengkapnya tentang cara memperbarui pengaturan penskalaan otomatis untuk tabel Multi-wilayah, lihat [the section called “Memperbarui pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”](#)

Penskalaan otomatis Amazon Keyspaces memerlukan keberadaan peran terkait layanan (`AWSServiceRoleForApplicationAutoScaling_CassandraTable`) yang melakukan tindakan penskalaan otomatis atas nama Anda. Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Menggunakan Peran Terkait Layanan”](#).

## Console

Konfigurasi penskalaan otomatis Amazon Keyspaces untuk tabel yang ada

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Pilih tabel yang ingin Anda kerjakan, dan buka tab Kapasitas.

3. Di bagian Pengaturan kapasitas, pilih Edit.
4. Di bawah mode Kapasitas, pastikan tabel menggunakan mode kapasitas yang disediakan.
5. Pilih Skala secara otomatis dan lihat langkah 6 [the section called “Buat tabel baru dengan penskalaan otomatis”](#) untuk mengedit kapasitas baca dan tulis.
6. Saat pengaturan penskalaan otomatis ditentukan, pilih Simpan.

## Cassandra Query Language (CQL)

Konfigurasi tabel yang ada dengan penskalaan otomatis Amazon Keyspaces menggunakan CQL

Anda dapat menggunakan ALTER TABLE pernyataan untuk tabel Amazon Keyspaces yang ada untuk mengonfigurasi penskalaan otomatis untuk kapasitas tulis atau baca tabel. Jika Anda memperbarui tabel yang saat ini dalam mode kapasitas sesuai permintaan, Anda harus menyetel `capacity_mode` ke `provisioned`. Jika tabel Anda sudah dalam mode kapasitas yang disediakan, bidang ini dapat dihilangkan.

Dalam contoh berikut, pernyataan memperbarui tabel `mytable`, yang dalam mode kapasitas sesuai permintaan. Pernyataan tersebut mengubah mode kapasitas tabel menjadi mode yang disediakan dengan penskalaan otomatis diaktifkan.

Kapasitas tulis dikonfigurasi dalam kisaran 5-10 unit kapasitas dengan nilai target 50%. Kapasitas baca juga dikonfigurasi dalam kisaran 5-10 unit kapasitas dengan nilai target 50%. Untuk kapasitas baca, Anda mengatur nilai untuk `scale_out_cooldown` dan `scale_in_cooldown` ke 60 detik.

```
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

Konfigurasi tabel yang ada dengan penskalaan otomatis Amazon Keyspaces menggunakan AWS CLI

Untuk tabel Amazon Keyspaces yang ada, Anda dapat mengaktifkan penskalaan otomatis untuk kapasitas tulis atau baca tabel menggunakan operasi. `UpdateTable`

Anda dapat menggunakan perintah berikut untuk mengaktifkan penskalaan otomatis Amazon Keyspaces untuk tabel yang ada. Pengaturan penskalaan otomatis untuk tabel dimuat dari file JSON. Untuk contoh berikut, Anda dapat men-download contoh file JSON dari [auto-scaling.zip](#) dan ekstrak `auto-scaling.json`, mencatat path ke file. Dalam contoh ini, file JSON terletak di direktori saat ini. Untuk opsi jalur file yang berbeda, lihat [Cara memuat parameter dari file](#).

Untuk informasi selengkapnya tentang pengaturan penskalaan otomatis yang digunakan dalam contoh berikut, lihat [the section called “Buat tabel baru dengan penskalaan otomatis”](#).

```

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

```

## Lihat konfigurasi penskalaan otomatis Amazon Keyspaces tabel Anda

Anda dapat menggunakan konsol, CQL, atau AWS CLI untuk melihat dan memperbarui pengaturan penskalaan otomatis Amazon Keyspaces dari sebuah tabel.

### Console

Lihat pengaturan penskalaan otomatis menggunakan konsol

1. Pilih tabel yang ingin Anda lihat dan buka tab Kapasitas.
2. Di bagian Pengaturan kapasitas, pilih Edit. Anda sekarang dapat memodifikasi pengaturan di bagian Kapasitas Baca atau Kapasitas Tulis. Untuk informasi selengkapnya tentang pengaturan ini, lihat [the section called “Buat tabel baru dengan penskalaan otomatis”](#).

### Cassandra Query Language (CQL)

Melihat kebijakan penskalaan otomatis Amazon Keyspaces tabel Anda menggunakan CQL

Untuk melihat detail konfigurasi penskalaan otomatis tabel, gunakan perintah berikut.

```
SELECT * FROM system_schema_mcs.autoscaling WHERE keyspace_name = 'mykeyspace' AND
table_name = 'mytable';
```

Output untuk perintah ini terlihat seperti ini.

```
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

Melihat kebijakan penskalaan otomatis Amazon Keyspaces tabel Anda menggunakan AWS CLI

Untuk melihat konfigurasi penskalaan otomatis tabel, Anda dapat menggunakan `get-table-auto-scaling-settings` operasi. Perintah CLI berikut adalah contoh dari ini.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --table-name mytable
```

Output untuk perintah ini terlihat seperti ini.

```
{
 "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
 }
 }
 }
 }
}
```

```
}
 }
}
```

## Matikan penskalaan otomatis Amazon Keyspaces untuk sebuah tabel

Anda dapat menonaktifkan penskalaan otomatis Amazon Keyspaces untuk tabel Anda kapan saja. Jika Anda tidak perlu lagi menskalakan kapasitas baca atau tulis tabel Anda, Anda harus mempertimbangkan untuk mematikan penskalaan otomatis sehingga Amazon Keyspaces tidak terus memodifikasi pengaturan kapasitas baca atau tulis tabel Anda. Anda dapat memperbarui tabel menggunakan konsol, CQL, atau file. AWS CLI

Mematikan penskalaan otomatis juga menghapus CloudWatch alarm yang dibuat atas nama Anda.

Untuk menghapus peran terkait layanan yang digunakan oleh Application Auto Scaling untuk mengakses tabel Amazon Keyspaces, ikuti langkah-langkahnya. [the section called “Menghapus peran terkait layanan untuk Amazon Keyspaces”](#)

### Note

Untuk menghapus peran terkait layanan yang digunakan Application Auto Scaling, Anda harus menonaktifkan penskalaan otomatis pada semua tabel di akun. Wilayah AWS

## Console

Matikan penskalaan otomatis Amazon Keyspaces untuk tabel Anda menggunakan konsol

Menggunakan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pilih tabel yang ingin Anda perbarui dan buka tab Kapasitas.
3. Di bagian Pengaturan kapasitas, pilih Edit.
4. Untuk menonaktifkan penskalaan otomatis Amazon Keyspaces, kosongkan kotak centang Skala secara otomatis. Menonaktifkan penskalaan otomatis membatalkan pendaftaran tabel sebagai target yang dapat diskalakan dengan Application Auto Scaling.

## Cassandra Query Language (CQL)

Matikan penskalaan otomatis Amazon Keyspaces untuk tabel Anda menggunakan CQL

Pernyataan berikut mematikan penskalaan otomatis untuk kapasitas tulis tabel mytable.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
 'provisioned_write_capacity_autoscaling_update': {
 'autoscaling_disabled': true
 }
};
```

## CLI

Matikan penskalaan otomatis Amazon Keyspaces untuk tabel Anda menggunakan AWS CLI

Perintah berikut mematikan penskalaan otomatis untuk kapasitas baca tabel. Ini juga menghapus CloudWatch alarm yang dibuat atas nama Anda.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
 \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

## Melihat aktivitas penskalaan otomatis untuk tabel Amazon Keyspaces di Amazon CloudWatch

Anda dapat memantau cara penskalaan otomatis Amazon Keyspaces menggunakan sumber daya dengan menggunakan Amazon CloudWatch, yang menghasilkan metrik tentang penggunaan dan kinerja Anda. Ikuti langkah-langkah di [Panduan Application Auto Scaling Pengguna](#) untuk membuat CloudWatch dasbor.

## Gunakan kapasitas burst secara efektif di Amazon Keyspaces

Amazon Keyspaces memberikan beberapa fleksibilitas dalam penyediaan throughput per partisi Anda dengan menyediakan kapasitas burst. Kapan pun Anda tidak sepenuhnya menggunakan throughput partisi, Amazon Keyspaces menyimpan sebagian dari kapasitas yang tidak terpakai itu untuk semburan throughput nanti guna menangani lonjakan penggunaan.



Amazon Keyspaces saat ini mempertahankan hingga 5 menit (300 detik) kapasitas baca dan tulis yang tidak digunakan. Selama ledakan aktivitas baca atau tulis sesekali, unit kapasitas ekstra ini dapat dikonsumsi dengan cepat—bahkan lebih cepat daripada kapasitas throughput yang disediakan per detik yang telah Anda tetapkan untuk tabel Anda.

Amazon Keyspaces juga dapat menggunakan kapasitas burst untuk pemeliharaan latar belakang dan tugas lainnya tanpa pemberitahuan sebelumnya.

Perhatikan bahwa detail kapasitas lonjakan mungkin berubah di masa mendatang.

# Bekerja dengan fitur Amazon Keyspaces (untuk Apache Cassandra)

Bab ini memberikan rincian tentang bekerja dengan Amazon Keyspaces dan berbagai fitur database, misalnya backup dan restore, Time to Live, dan Multi-region replikasi.

- Waktu untuk Hidup - Amazon Keyspaces kedaluwarsa data dari tabel secara otomatis berdasarkan nilai Time to Live yang Anda tetapkan. Pelajari cara mengkonfigurasi TTL dan cara menggunakannya di tabel Anda.
- PITR — Lindungi tabel Amazon Keyspaces Anda dari operasi tulis atau hapus yang tidak disengaja dengan membuat cadangan data tabel Anda secara berkelanjutan. Pelajari cara mengkonfigurasi PITR pada tabel Anda dan cara mengembalikan tabel ke titik waktu tertentu atau cara mengembalikan tabel yang telah dihapus secara tidak sengaja.
- Bekerja dengan tabel Multi-wilayah — Multi-Region tabel di Amazon Keyspaces harus memiliki kapasitas throughput tulis yang dikonfigurasi dalam mode kapasitas sesuai permintaan atau yang disediakan dengan penskalaan otomatis. Rencanakan kebutuhan kapasitas throughput dengan memperkirakan unit kapasitas tulis (WCU) yang diperlukan untuk setiap Wilayah, dan berikan jumlah penulisan dari semua Wilayah untuk memastikan kapasitas yang cukup untuk penulisan yang direplikasi.
- Amazon Keyspaces change data capture (CDC) — Amazon Keyspaces CDC stream merekam peristiwa perubahan tingkat baris dari tabel Anda dalam waktu hampir real time. Pelajari cara menggunakan Kinesis Client Library (KCL) untuk mengkonsumsi dan memproses data dari Amazon Keyspaces CDC stream.
- Kueri dan pagination — Amazon Keyspaces mendukung kemampuan kueri tingkat lanjut seperti menggunakan IN operator dengan SELECT pernyataan, mengurutkan hasil dengan ORDER BY, dan pagination otomatis dari kumpulan hasil besar. Bagian ini menjelaskan bagaimana Amazon Keyspaces memproses kueri ini dan memberikan contoh.
- Partisi - Amazon Keyspaces menyediakan tiga partisi: `Murmur3Partitioner` (default),, dan `RandomPartitioner`. `DefaultPartitioner` Anda dapat mengubah partisi per Wilayah di tingkat akun menggunakan Konsol Manajemen AWS atau Cassandra Query Language (CQL).
- Client-side stempel waktu — Client-side stempel waktu adalah stempel waktu Cassandra-compatible yang disimpan Amazon Keyspaces untuk setiap sel di tabel Anda. Gunakan stempel waktu sisi klien untuk resolusi konflik dan biarkan aplikasi klien Anda menentukan urutan penulisan.

- **User-defined type (UDT)** - Dengan UDT Anda dapat menentukan struktur data dalam aplikasi Anda yang mewakili hierarki data dunia nyata.
- **Sumber daya penandaan** — Anda dapat memberi label sumber daya Amazon Keyspaces seperti keyspaces dan tabel menggunakan tag. Tag membantu mengkategorikan sumber daya, mengaktifkan pelacakan biaya, dan memungkinkan Anda mengonfigurasi kontrol akses berdasarkan tag. Bagian ini mencakup pembatasan penandaan, operasi, dan praktik terbaik untuk Amazon Keyspaces.
- **CloudFormation template** — CloudFormation membantu Anda memodelkan dan mengatur ruang kunci dan tabel Amazon Keyspaces sehingga Anda dapat menghabiskan lebih sedikit waktu untuk membuat dan mengelola sumber daya dan infrastruktur Anda.

## Topik

- [Ruang kunci sistem di Amazon Keyspaces](#)
- [User-defined jenis \(UDT\) di Amazon Keyspaces](#)
- [Bekerja dengan kueri CQL di Amazon Keyspaces](#)
- [Bekerja dengan aliran change data capture \(CDC\) di Amazon Keyspaces](#)
- [Bekerja dengan partisi di Amazon Keyspaces](#)
- [Client-side stempel waktu di Amazon Keyspaces](#)
- [Multi-Region replikasi untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#)
- [Cadangkan dan pulihkan data dengan pemulihan point-in-time untuk Amazon Keyspaces](#)
- [Data kedaluwarsa dengan Time to Live \(TTL\) untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#)
- [Menggunakan layanan ini dengan AWS SDK](#)
- [Bekerja dengan tag dan label untuk sumber daya Amazon Keyspaces](#)
- [Buat sumber daya Amazon Keyspaces dengan AWS CloudFormation](#)
- [Menggunakan NoSQL Workbench dengan Amazon Keyspaces \(untuk Apache Cassandra\)](#)

## Ruang kunci sistem di Amazon Keyspaces

Ruang kunci sistem dan tabel sistem di Amazon Keyspaces (untuk Apache Cassandra) adalah sumber daya hanya-baca yang menyimpan metadata tentang sumber daya Amazon Keyspaces

Anda. Ruang kunci sistem hadir di setiap Akun AWS, terlepas dari apakah Anda telah membuat ruang kunci atau tabel apa pun. Mereka adalah fitur kompatibilitas dengan Apache Cassandra, dan mereka disediakan tanpa biaya tambahan.

Anda tidak dapat memodifikasi atau menghapus ruang kunci sistem. Konsol Amazon Keyspaces hanya menampilkan ruang kunci yang dibuat pengguna. Ruang kunci sistem dapat diakses secara terprogram melalui CQL dan muncul di layanan seperti dan. AWS CloudFormation AWS Config

Amazon Keyspaces menggunakan empat ruang kunci sistem:

- `system`
- `system_schema`
- `system_schema_mcs`
- `system_multiregion_info`

Bagian berikut memberikan rincian tentang ruang kunci sistem dan tabel sistem yang didukung di Amazon Keyspaces.

## system

Ini adalah ruang kunci Cassandra. Amazon Keyspaces menggunakan tabel berikut.

| Nama tabel         | Nama kolom                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Komentar                          |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| <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>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_address</code> , <code>schema_version</code> , | Informasi tentang keyspace lokal. |

| Nama tabel          | Nama kolom                                                                                                             | Komentar                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------|------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     | thrift_version,<br>tokens, truncated_at                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| peers               | peer, data_center,<br>host_id, preferred<br>_ip, rack, release_v<br>ersion, rpc_addre<br>ss, schema_version,<br>tokens | Kueri tabel ini untuk melihat titik akhir yang tersedia. Misalnya, jika Anda terhubung melalui titik akhir publik, Anda melihat daftar sembilan alamat IP yang tersedia. Jika Anda terhubung melalui titik akhir FIPS, Anda melihat daftar tiga alamat IP. Jika Anda terhubung melalui titik akhir AWS PrivateLink VPC, Anda melihat daftar alamat IP yang telah Anda konfigurasi. Untuk informasi selengkapnya, lihat <a href="#">the section called “Mengisi entri system.peers tabel dengan informasi titik akhir VPC antarmuka”</a> . |
| size_estimates      | keyspace_name,<br>table_name, range_sta<br>rt, range_end,<br>mean_partition_size,<br>partitions_count                  | Tabel ini mendefinisikan ukuran total dan jumlah partisi untuk setiap rentang token untuk setiap tabel. Ini diperlukan untuk Apache Cassandra Spark Connector, yang menggunakan perkiraan ukuran partisi untuk mendistribusikan pekerjaan.                                                                                                                                                                                                                                                                                                |
| prepared_statements | prepared_id,<br>logged_keyspace,<br>query_string                                                                       | Tabel ini berisi informasi tentang kueri yang disimpan.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

## system\_schema

Ini adalah ruang kunci Cassandra. Amazon Keyspaces menggunakan tabel berikut.

| Nama tabel | Nama kolom                                                                                                                                                                                                                                                                                                                                       | Komentar                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| keyspaces  | keyspace_name,<br>durable_writes,<br>replication                                                                                                                                                                                                                                                                                                 | Informasi tentang keyspace tertentu.                            |
| tables     | keyspace_name,<br>table_name, bloom_filter_fp_chance,<br>caching, comment,<br>compaction, compression, crc_check_chance,<br>dclocal_read_repair_chance,<br>default_time_to_live, extensions,<br>flags, gc_grace_seconds, id,<br>max_index_interval, memtable_flush_period_in_ms,<br>min_index_interval, read_repair_chance,<br>speculative_retry | Informasi tentang tabel tertentu.                               |
| types      | keyspace_name,<br>type_name, field_names, field_types                                                                                                                                                                                                                                                                                            | Informasi tentang tipe yang ditentukan pengguna (UDT) tertentu. |
| columns    | keyspace_name,<br>table_name, column_name, clustering_order, column_name                                                                                                                                                                                                                                                                         | Informasi tentang kolom tertentu.                               |

| Nama tabel | Nama kolom                     | Komentar |
|------------|--------------------------------|----------|
|            | me_bytes, kind, position, type |          |

## system\_schema\_mcs

Ini adalah ruang kunci Amazon Keyspaces yang menyimpan informasi tentang atau AWS setelan khusus Amazon Keyspaces.

| Nama tabel | Nama kolom                                                                                                                                                                                                                                                                                             | Komentar                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| keyspaces  | keyspace_name, durable_writes, replication                                                                                                                                                                                                                                                             | Kueri tabel ini untuk mengetahui secara terprogram apakah ruang kunci telah dibuat. Untuk informasi selengkapnya, lihat <a href="#">the section called “Periksa status pembuatan keyspace”</a> .                                                                                                                                                                                                                                          |
| tables     | keyspace_name, creation_time, speculative_retry, cdc, gc_grace_seconds, crc_check_chance, min_index_interval, bloom_filter_fp_chance, flags, custom_properties, dclocal_read_repair_chance, table_name, caching, default_time_to_live, read_repair_chance, max_index_interval, extensions, compaction, | <p>Kueri tabel ini untuk mengetahui status tabel tertentu. Untuk informasi selengkapnya, lihat <a href="#">the section called “Periksa status pembuatan tabel”</a>.</p> <p>Anda juga dapat menanyakan tabel ini untuk mencantumkan pengaturan yang khusus untuk Amazon Keyspaces dan disimpan sebagai custom_properties</p> <p>Contoh:</p> <ul style="list-style-type: none"> <li>• capacity_mode</li> <li>• cdc_specification</li> </ul> |

| Nama tabel     | Nama kolom                                                                                         | Komentar                                                                                                                                                        |
|----------------|----------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                | comment, id, compression, memtable_flush_period_in_ms, cdc_specification, latest_stream_arn status | <ul style="list-style-type: none"> <li>• client_side_timestamps</li> <li>• encryption_specification</li> <li>• point_in_time_recovery</li> <li>• ttl</li> </ul> |
| tables_history | keyspace_name, table_name, event_time, creation_time, custom_properties, event                     | Kueri tabel ini untuk mempelajari tentang perubahan skema untuk tabel tertentu.                                                                                 |
| columns        | keyspace_name, table_name, column_name, clustering_order, column_name_bytes, kind, position, type  | Tabel ini identik dengan tabel Cassandra di keyspace.<br>system_schema                                                                                          |
| tags           | resource_id, keyspace_name, resource_name, resource_type, tags                                     | Kueri tabel ini untuk mengetahui apakah ruang kunci memiliki tag. Untuk informasi selengkapnya, lihat <a href="#">the section called “Lihat tag tabel”</a> .    |



| Nama tabel  | Nama kolom                                                                                                                                           | Komentar                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| types       | keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types | Kueri tabel ini untuk mengetahui informasi tentang tipe yang ditentukan pengguna (UDT). Misalnya Anda dapat menanyakan tabel ini untuk mencantumkan semua UDT untuk ruang kunci tertentu. Untuk informasi selengkapnya, lihat <a href="#">the section called “User-defined jenis (UDT)”</a> .                                                                                                                        |
| autoscaling | keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update                               | Kueri tabel ini untuk mendapatkan pengaturan penskalaan otomatis dari tabel yang disediakan. Perhatikan bahwa pengaturan ini tidak akan tersedia sampai tabel aktif. Untuk menanyakan tabel ini, Anda harus menentukan keyspace_name dan table_name dalam WHERE klausa. Untuk informasi selengkapnya, lihat <a href="#">the section called “Lihat konfigurasi penskalaan otomatis Amazon Keyspaces tabel Anda”</a> . |

## system\_multiregion\_info

Ini adalah ruang kunci Amazon Keyspaces yang menyimpan informasi tentang replikasi Multi-wilayah.

| Nama tabel | Nama kolom                                                           | Komentar                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------|----------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| tables     | keyspace_name,<br>table_name, region,<br>status                      | <p>Tabel ini berisi informasi tentang tabel Multi-wilayah —misalnya, tabel Wilayah AWS yang direplikasi dan status tabel. Anda juga dapat menanyakan tabel ini untuk mencantumkan pengaturan yang khusus untuk Amazon Keyspaces yang disimpan sebagai <code>custom_properties</code>. Contoh:</p> <ul style="list-style-type: none"> <li>• <code>capacity_mode</code></li> </ul> <p>Untuk menanyakan tabel ini, Anda harus menentukan <code>keyspace_name</code> dan <code>table_name</code> dalam WHERE klausa. Untuk informasi selengkapnya, lihat <a href="#">the section called “Buat ruang kunci Multi-wilayah”</a>.</p> |
| keyspaces  | keyspace_name,<br>region, status,<br>tables_replication<br>_progress | <p>Tabel ini berisi informasi tentang kemajuan ALTER KEYSPACE operasi yang menambahkan replika ke ruang kunci — misalnya, berapa banyak tabel yang telah dibuat di Wilayah baru, dan berapa banyak tabel yang masih dalam proses. Sebagai</p>                                                                                                                                                                                                                                                                                                                                                                                 |

| Nama tabel  | Nama kolom                                                                                                                                                   | Komentar                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|             |                                                                                                                                                              | contoh, lihat <a href="#">the section called “Periksa kemajuan replikasi”</a> .                                                                                                                                                                                                                                                                                                                                                                                     |
| autoscaling | keyspace_name, table_name, provisioned_read_capacity_autoscaling_update, provisioned_write_capacity_autoscaling_update, region                               | Kueri tabel ini untuk mendapatkan pengaturan penskalaan otomatis dari tabel yang disediakan Multi-wilayah. Perhatikan bahwa pengaturan ini tidak akan tersedia sampai tabel aktif. Untuk menanyakan tabel ini, Anda harus menentukan keyspace_name dan table_name dalam WHERE klausa. Untuk informasi selengkapnya, lihat <a href="#">the section called “Memperbarui pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”</a> . |
| types       | keyspace_name, type_name, field_names, field_types, max_nesting_depth, last_modified_timestamp, status, direct_referring_tables, direct_parent_types, region | Kueri tabel ini untuk mengetahui informasi tentang tipe yang ditentukan pengguna (UDT) di ruang kunci multi-wilayah. Misalnya, Anda dapat menanyakan tabel ini untuk mencantumkan semua replika tabel dan AWS Wilayah masing-masing yang menggunakan UDT untuk ruang kunci tertentu. Lihat informasi yang lebih lengkap di <a href="#">the section called “User-defined jenis (UDT)”</a> .                                                                          |

# User-defined jenis (UDT) di Amazon Keyspaces

Tipe yang ditentukan pengguna (UDT) adalah pengelompokan bidang dan tipe data yang dapat Anda gunakan untuk menentukan satu kolom di Amazon Keyspaces. Tipe data yang valid untuk UDT adalah semua tipe data Cassandra yang didukung, termasuk koleksi dan UDT lain yang telah Anda buat di ruang kunci yang sama. Untuk informasi selengkapnya tentang tipe data Cassandra yang didukung, lihat. [the section called “Dukungan tipe data Cassandra”](#)

Anda dapat menggunakan tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces untuk mengatur data dengan cara yang lebih efisien. Misalnya, Anda dapat membuat UDT dengan koleksi bersarang yang memungkinkan Anda menerapkan pemodelan data yang lebih kompleks dalam aplikasi Anda. Anda juga dapat menggunakan kata kunci beku untuk mendefinisikan UDT.

UDT terikat ke ruang kunci dan tersedia untuk semua tabel dan UDT di ruang kunci yang sama. Anda dapat membuat UDT di ruang kunci Single-region dan Multi-region.

Anda dapat membuat tabel baru atau mengubah tabel yang ada dan menambahkan kolom baru yang menggunakan UDT. Untuk membuat UDT dengan UDT bersarang, UDT bersarang harus dibekukan.

Untuk meninjau berapa banyak UDT yang didukung per ruang kunci, tingkat penyarangan yang didukung, dan nilai dan kuota default lainnya yang terkait dengan UDT, lihat. [the section called “Kuota dan nilai default untuk tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces”](#)

Untuk informasi tentang cara menghitung ukuran UDT yang dikodekan, lihat. [the section called “Perkiraan ukuran nilai data yang dikodekan berdasarkan tipe data”](#)

Untuk informasi selengkapnya tentang sintaks CQL, lihat. [the section called “Tipe”](#)

Untuk mempelajari lebih lanjut tentang UDT dan pemulihan waktu point-in, lihat. [the section called “PITR dan UDT”](#)

## Topik

- [Mengonfigurasi izin untuk bekerja dengan tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces](#)
- [Buat tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces](#)
- [Lihat tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces](#)
- [Menghapus tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces](#)

## Mengonfigurasi izin untuk bekerja dengan tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces

Seperti tabel, UDT terikat ke ruang kunci tertentu. Tetapi tidak seperti tabel, Anda tidak dapat menentukan izin secara langsung untuk UDT. UDT tidak dianggap sebagai sumber daya AWS dan mereka tidak memiliki pengidentifikasi unik dalam format Nama Sumber Daya Amazon (ARN). Sebagai gantinya, untuk memberikan izin utama IAM untuk melakukan tindakan tertentu pada UDT, Anda harus menentukan izin untuk ruang kunci yang terikat UDT. Untuk bekerja dengan UDT di ruang kunci Multi-wilayah, izin tambahan diperlukan.

Untuk dapat membuat, melihat, atau menghapus UDT, prinsipal, misalnya pengguna atau peran IAM, memerlukan izin yang sama yang diperlukan untuk melakukan tindakan yang sama pada ruang kunci yang terikat UDT.

Untuk informasi lebih lanjut tentang AWS Identity and Access Management, lihat [the section called “AWS Identity and Access Management”](#).

### Izin untuk membuat UDT

Untuk membuat UDT di ruang kunci wilayah Tunggal, prinsipal memerlukan `Create` izin untuk ruang kunci.

Kebijakan IAM berikut adalah contohnya.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "cassandra:Create",
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
]
 }
]
}
```

Untuk membuat UDT di ruang kunci Multi-wilayah, selain izin, prinsipal juga memerlukan `Create` izin untuk tindakan `CreateMultiRegionResource` untuk ruang kunci yang ditentukan.

Kebijakan IAM berikut adalah contohnya.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": ["cassandra:Create", "cassandra:CreateMultiRegionResource"],
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
]
 }
]
}
```

## Izin untuk melihat UDT

Untuk melihat atau mencantumkan UDT di ruang kunci Single-region, prinsipal memerlukan izin baca untuk ruang kunci sistem. Untuk informasi selengkapnya, lihat [the section called “system\\_schema\\_mcs”](#).

Kebijakan IAM berikut adalah contohnya.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "cassandra:Select",
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
]
 }
]
}
```

Untuk melihat atau mencantumkan UDT untuk ruang kunci Multi-wilayah, prinsipal memerlukan izin untuk tindakan SELECT dan SelectMultiRegionResource untuk ruang kunci sistem. Untuk informasi selengkapnya, lihat [the section called “system\\_multiregion\\_info”](#).

Kebijakan IAM berikut adalah contohnya.

```
{
```

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": ["cassandra:Select", "cassandra:SelectMultiRegionResource"],
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
]
 }
]
}

```

## Izin untuk menghapus UDT

Untuk menghapus UDT dari ruang kunci wilayah Tunggal, prinsipal memerlukan izin untuk Drop tindakan untuk ruang kunci yang ditentukan.

Kebijakan IAM berikut adalah contohnya.

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "cassandra:Drop",
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
]
 }
]
}

```

Untuk menghapus UDT dari ruang kunci Multi-wilayah, prinsipal memerlukan izin untuk Drop tindakan dan tindakan untuk ruang kunci yang DropMultiRegionResource ditentukan.

Kebijakan IAM berikut adalah contohnya.

```

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

```

```

 "Action": ["cassandra:Drop", "cassandra:DropMultiRegionResource"],
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/my_keyspace/"
]
 }
}

```

## Buat tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces

Untuk membuat UDT di ruang kunci Single-region, Anda dapat menggunakan `CREATE TYPE` pernyataan di CQL, `create-type` perintah dengan, atau konsol. AWS CLI

Nama UDT harus berisi 48 karakter atau kurang, harus dimulai dengan karakter alfabet, dan hanya dapat berisi karakter alfa-numerik dan garis bawah. Amazon Keyspaces mengonversi karakter huruf besar secara otomatis menjadi karakter huruf kecil.

Atau, Anda dapat mendeklarasikan nama UDT dalam tanda kutip ganda. Saat mendeklarasikan nama UDT di dalam tanda kutip ganda, Amazon Keyspaces mempertahankan casing atas dan memungkinkan karakter khusus.

Anda juga dapat menggunakan tanda kutip ganda sebagai bagian dari nama saat Anda membuat UDT, tetapi Anda harus melarikan diri dari setiap karakter kutipan ganda dengan karakter kutipan ganda tambahan.

Tabel berikut menunjukkan contoh nama UDT yang diizinkan. Kolom pertama menunjukkan cara memasukkan nama saat Anda membuat jenis, kolom kedua menunjukkan bagaimana Amazon Keyspaces memformat nama secara internal. Amazon Keyspaces mengharapkan nama yang diformat untuk operasi seperti. `GetType`

| Nama yang dimasukkan | Nama yang diformat | Catatan                                                                                                                               |
|----------------------|--------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| MY_UDT               | my_udt             | Tanpa tanda kutip ganda, Amazon Keyspaces mengonversi semua karakter huruf besar menjadi huruf kecil.                                 |
| "MY_UDT"             | MY_UDT             | Dengan tanda kutip ganda, Amazon Keyspaces menghormati karakter huruf besar, dan menghapus tanda kutip ganda dari nama yang diformat. |



| Nama yang dimasukkan      | Nama yang diformat     | Catatan                                                                                                                                |
|---------------------------|------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| "1234"                    | 1234                   | Dengan tanda kutip ganda, nama dapat dimulai dengan angka, dan Amazon Keyspaces menghapus tanda kutip ganda dari nama yang diformat.   |
| "Special_Ch@r@cter s<>!!" | Special_Ch@r@cters<>!! | Dengan tanda kutip ganda, nama dapat berisi karakter khusus, dan Amazon Keyspaces menghapus tanda kutip ganda dari nama yang diformat. |
| "nested""""quote s"       | nested""""quotes       | Amazon Keyspaces menghapus tanda kutip ganda luar dan tanda kutip ganda escape dari nama yang diformat.                                |

## Console

Buat tipe yang ditentukan pengguna (UDT) dengan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces, lalu pilih keyspace dari daftar.
3. Pilih tab UDTs.
4. Pilih Buat UDT
5. Di bawah rincian UDT, masukkan nama untuk UDT. Di bawah bidang UDT Anda menentukan skema UDT.
6. Untuk menyelesaikannya, pilih Buat UDT.

## Cassandra Query Language (CQL)

Buat tipe yang ditentukan pengguna (UDT) dengan CQL

Dalam contoh ini kita membuat versi baru dari tabel penghargaan buku yang digunakan [the section called “Membuat tabel”](#). Dalam tabel ini, kami menyimpan semua penghargaan yang diterima penulis untuk buku yang diberikan. Kami membuat dua UDT yang bersarang dan berisi informasi tentang buku yang menerima penghargaan.

1. Buat keyspace dengan nama `catalog`.

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

2. Buat tipe pertama. Jenis ini menyimpan kode BISAC, yang digunakan untuk menentukan genre buku. Kode BISAC terdiri dari kode alfa-numerik dan hingga empat bidang materi pelajaran.

```
CREATE TYPE catalog.bisac (
 bisac_code text,
 subject1 text,
 subject2 text,
 subject3 text,
 subject4 text
);
```

3. Buat tipe kedua untuk penghargaan buku yang menggunakan UDT pertama. UDT bersarang harus dibekukan.

```
CREATE TYPE catalog.book (
 award_title text,
 book_title text,
 publication_date date,
 page_count int,
 ISBN text,
 genre FROZEN <bisac>
);
```

4. Buat tabel dengan kolom untuk nama penulis dan gunakan jenis daftar untuk penghargaan buku. Perhatikan bahwa UDT yang digunakan dalam daftar harus dibekukan.

```
CREATE TABLE catalog.authors (
 author_name text PRIMARY KEY,
 awards list <FROZEN <book>>
);
```

5. Pada langkah ini kita memasukkan satu baris data ke dalam tabel baru.

```
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. Pada langkah terakhir kita membaca data dari tabel.

```
SELECT * FROM catalog.authors;
```

Output dari perintah akan terlihat seperti ini.

```
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)
```

Untuk informasi selengkapnya tentang sintaks CQL, lihat. [the section called “BUAT TIPE”](#)

## CLI

Buat tipe yang ditentukan pengguna (UDT) dengan AWS CLI

1. Untuk membuat tipe Anda dapat menggunakan sintaks berikut.

```
aws keyspaces create-type
--keyspace-name 'my_keyspace'
--type-name 'my_udt'
--field-definitions
 '['
 {"name" : "field1", "type" : "int"},
 {"name" : "field2", "type" : "text"}
 ']
```

2. Output dari perintah itu terlihat mirip dengan contoh ini. Perhatikan bahwa typeName mengembalikan nama UDT yang diformat.

```
{
 "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
 "typeName": "my_udt"
}
```

## Lihat tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces

Untuk melihat atau mencantumkan semua UDT di ruang kunci Single-region, Anda dapat melakukan kueri tabel `system_schema_mcs.types` di ruang kunci sistem menggunakan pernyataan di CQL, atau menggunakan `list-type` perintah `get-type` and dengan, atau konsol. AWS CLI

Untuk salah satu opsi, prinsipal IAM memerlukan izin baca ke ruang kunci sistem. Untuk informasi selengkapnya, lihat [the section called “Mengatur Konfigurasi Izin”](#).

## Console

Lihat tipe yang ditentukan pengguna (UDT) dengan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces, lalu pilih keyspace dari daftar.

3. Pilih tab UDT untuk meninjau daftar semua UDT di ruang kunci.
4. Untuk meninjau satu UDT secara detail, pilih UDT dari daftar.
5. Pada tab Skema Anda dapat meninjau skema. Pada tab Digunakan di Anda dapat melihat apakah UDT ini digunakan dalam tabel atau UDT lainnya. Perhatikan bahwa Anda hanya dapat menghapus UDT yang tidak digunakan oleh tabel atau UDT lainnya.

## Cassandra Query Language (CQL)

Melihat tipe yang ditentukan pengguna (UDT) dari ruang kunci wilayah tunggal dengan CQL

1. Untuk melihat jenis yang tersedia di ruang kunci tertentu, Anda dapat menggunakan pernyataan berikut.

```
SELECT type_name
FROM system_schema_mcs.types
WHERE keyspace_name = 'my_keyspace';
```

2. Untuk melihat detail tentang jenis tertentu, Anda dapat menggunakan pernyataan berikut.

```
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. Anda dapat mencantumkan semua UDT yang ada di akun menggunakan `DESC TYPE`.

```
DESC TYPES;

Keyspace my_keyspace

my_udt1 my_udt2

Keyspace my_keyspace2
```

```

my_udt1
```

- Anda dapat mencantumkan semua UDT di ruang kunci yang dipilih saat ini menggunakan.  
`DESC TYPE`

```
USE my_keyspace;
my_keyspace DESC TYPES;

my_udt1 my_udt2
```

- Untuk mencantumkan semua UDT di ruang kunci Multi-wilayah, Anda dapat melakukan kueri tabel sistem `types` di ruang kunci. `system_multiregion_info` Query berikut adalah contoh dari ini.

```
SELECT keyspace_name, type_name, region, status FROM
system_multiregion_info.types WHERE keyspace_name = 'mykeyspace' AND table_name
= 'mytable';
```

Output dari perintah ini terlihat mirip dengan ini.

| 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

Lihat tipe yang ditentukan pengguna (UDT) dengan AWS CLI

- Untuk membuat daftar jenis yang tersedia di ruang kunci, Anda dapat menggunakan `list-types` perintah.

```
aws keyspace list-types
--keyspace-name 'my_keyspace'
```

Output dari perintah itu terlihat mirip dengan contoh ini.

```
{
 "types": [
```

```
 "my_udt",
 "parent_udt"
]
}
```

2. Untuk melihat detail tentang jenis tertentu, Anda dapat menggunakan `get-type` perintah.

```
aws keyspaces get-type
--type-name 'my_udt'
--keyspace-name 'my_keyspace'
```

Output dari perintah ini terlihat mirip dengan contoh ini.

```
{
 "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/"
}
```

## Menghapus tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces

Untuk menghapus UDT di ruang kunci, Anda dapat menggunakan `DROP TYPE` pernyataan di CQL, `delete-type` perintah dengan AWS CLI, atau konsol.

## Console

Menghapus tipe yang ditentukan pengguna (UDT) dengan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces, lalu pilih keyspace dari daftar.
3. Pilih tab UDTs.
4. Pilih UDT yang ingin Anda hapus. Pada Digunakan di Anda dapat mengonfirmasi bahwa jenis yang ingin Anda hapus saat ini tidak digunakan oleh tabel atau UDT lainnya.
5. Pilih Hapus di atas Ringkasan.
6. Ketik Delete dialog yang muncul, dan pilih Hapus UDT.

## Cassandra Query Language (CQL)

Hapus tipe yang ditentukan pengguna (UDT) dengan CQL

- Untuk menghapus tipe, Anda dapat menggunakan pernyataan berikut.

```
DROP TYPE my_keyspace.my_udt;
```

Untuk informasi selengkapnya tentang sintaks CQL, lihat. [the section called “TIPE DROP”](#)

## CLI

Hapus tipe yang ditentukan pengguna (UDT) dengan AWS CLI

1. Untuk menghapus jenis, Anda dapat menggunakan perintah berikut.

```
aws keyspaces delete-type
--keyspace-name 'my_keyspace'
--type-name 'my_udt'
```

2. Output dari perintah terlihat mirip dengan contoh ini.

```
{
 "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
```



```
"typeName": "my_udt"
}
```

## Bekerja dengan kueri CQL di Amazon Keyspaces

Bagian ini memberikan pengantar untuk bekerja dengan kueri di Amazon Keyspaces (untuk Apache Cassandra). Pernyataan CQL yang tersedia untuk query, transformasi, dan mengelola data adalah `SELECT`, `INSERT`, `UPDATE`, dan `DELETE`. Topik berikut menguraikan beberapa opsi yang lebih kompleks yang tersedia saat bekerja dengan kueri. Untuk sintaks bahasa lengkap dengan contoh, lihat [the section called “Pernyataan DXML”](#).

### Topik

- [Gunakan `IN` operator dengan `SELECT` pernyataan dalam kueri di Amazon Keyspaces](#)
- [Menggunakan pernyataan batch di Amazon Keyspaces](#)
- [Pesan hasil dengan `ORDER BY` di Amazon Keyspaces](#)
- [Hasil paginasi di Amazon Keyspaces](#)

## Gunakan **IN** operator dengan **SELECT** pernyataan dalam kueri di Amazon Keyspaces

### PILIH DI

Anda dapat menanyakan data dari tabel menggunakan `SELECT` pernyataan, yang membaca satu atau beberapa kolom untuk satu atau beberapa baris dalam tabel dan mengembalikan kumpulan hasil yang berisi baris yang cocok dengan permintaan. `SELECT` pernyataan berisi `select_clause` yang menentukan kolom mana yang akan dibaca dan dikembalikan dalam kumpulan hasil. Klausula dapat berisi instruksi untuk mengubah data sebelum mengembalikannya. `WHERE` klausula opsional menentukan baris mana yang harus ditanyakan dan terdiri dari hubungan pada kolom yang merupakan bagian dari kunci utama. Amazon Keyspaces mendukung `IN` kata kunci dalam klausula. `WHERE` Bagian ini menggunakan contoh untuk menunjukkan bagaimana Amazon Keyspaces memproses `SELECT` pernyataan dengan kata kunci `IN`.

Contoh ini menunjukkan bagaimana Amazon Keyspaces `SELECT` memecah pernyataan dengan kata kunci menjadi `IN` subquery. Dalam contoh ini kita menggunakan tabel dengan nama `my_keyspace.customers`. Tabel memiliki satu kolom kunci utama `department_id`, dua

kolom pengelompokan `sales_region_id` dan `sales_representative_id`, dan satu kolom yang berisi nama pelanggan di `customer_name` kolom.

```
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             |

Dengan menggunakan tabel ini, Anda dapat menjalankan `SELECT` pernyataan berikut untuk menemukan pelanggan di departemen dan wilayah penjualan yang Anda minati dengan `IN` kata kunci dalam `WHERE` klausa. Pernyataan berikut adalah contoh dari ini.

```
SELECT * FROM my_keyspace.customers WHERE department_id IN (0, 1) AND sales_region_id IN (0, 1);
```

Amazon Keyspaces membagi pernyataan ini menjadi empat subquery seperti yang ditunjukkan pada output berikut.

```
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             |

Saat IN kata kunci digunakan, Amazon Keyspaces secara otomatis melakukan paginasi hasil dalam salah satu kasus berikut:

- Setelah setiap subquery ke-10 diproses.
- Setelah memproses 1MB IO logis.
- Jika Anda mengonfigurasi `PAGE SIZE`, Amazon Keyspaces melakukan paginasi setelah membaca jumlah kueri untuk diproses berdasarkan set. `PAGE SIZE`
- Saat Anda menggunakan `LIMIT` kata kunci untuk mengurangi jumlah baris yang dikembalikan, Amazon Keyspaces melakukan paginasi setelah membaca jumlah kueri untuk diproses berdasarkan set. `LIMIT`

Tabel berikut digunakan untuk mengilustrasikan ini dengan sebuah contoh.

Untuk informasi lebih lanjut tentang pagination, lihat [the section called “Hasil paginate”](#).

```
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 |  | 1 |

Anda dapat menjalankan pernyataan berikut pada tabel ini untuk melihat bagaimana pagination bekerja.

```
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 memproses pernyataan ini sebagai 24 subkueri, karena kardinalitas produk Cartesien dari semua istilah yang terkandung dalam kueri ini adalah 24. IN

| 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             |

Contoh ini menunjukkan bagaimana Anda dapat menggunakan ORDER BY klausa dalam SELECT pernyataan dengan IN kata kunci.

```
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                       |  | 1             |
| 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 |

Subkueri diproses dalam urutan di mana kunci partisi dan kolom kunci pengelompokan disajikan dalam kueri. Dalam contoh di bawah ini, subquery untuk nilai kunci partisi” 2 “diproses terlebih dahulu, diikuti oleh subquery untuk nilai kunci partisi” 3 “dan” 1”. Hasil subquery yang diberikan diurutkan sesuai dengan klausa pengurutan kueri, jika ada, atau urutan pengelompokan tabel yang ditentukan selama pembuatan tabel.

```
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             |

## Menggunakan pernyataan batch di Amazon Keyspaces

Anda dapat menggabungkan beberapa INSERT, UPDATE, dan DELETE operasi ke dalam BATCH pernyataan. LOGGED batch adalah default.

```
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
```

Ketika Anda menjalankan pernyataan batch, driver menggabungkan semua pernyataan dalam batch ke dalam operasi batch tunggal.

Untuk memutuskan jenis operasi batch yang akan digunakan, Anda dapat mempertimbangkan pedoman berikut.

Gunakan batch yang dicatat saat:

- Anda membutuhkan jaminan transaksi atom.
- Latensi yang sedikit lebih tinggi adalah trade-off yang dapat diterima.

Gunakan batch yang tidak tersumbat saat:

- Anda perlu mengoptimalkan operasi partisi tunggal.
- Anda ingin mengurangi overhead jaringan.
- Anda memiliki persyaratan throughput tinggi.

Untuk informasi tentang kuota pernyataan batch, lihat [Kuota](#).

## Batch yang tidak tersumbat

Dengan batch yang tidak tercatat, Amazon Keyspaces memproses beberapa operasi sebagai permintaan tunggal tanpa mempertahankan log batch. Dengan operasi batch yang tidak tercatat, ada kemungkinan beberapa tindakan berhasil sementara yang lain gagal. Batch yang tidak tersumbat berguna saat Anda ingin:

- Optimalkan operasi dalam satu partisi.
- Kurangi lalu lintas jaringan dengan mengelompokkan permintaan terkait.

Sintaks untuk batch yang tidak tercatat mirip dengan batch yang dicatat, dengan penambahan kata kunci. UNLOGGED

```
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;
```

## Batch yang dicatat

Batch yang dicatat menggabungkan beberapa tindakan penulisan ke dalam satu operasi atom. Saat Anda menjalankan batch yang dicatat:

- Semua tindakan berhasil bersama atau gagal bersama.

- Operasi ini sinkron dan idempoten.
- Anda dapat menulis ke beberapa tabel Amazon Keyspaces, selama mereka berada di AWS akun yang sama dan. Wilayah AWS

Batch yang dicatat mungkin memiliki latensi yang sedikit lebih tinggi. Untuk aplikasi throughput tinggi, pertimbangkan untuk menggunakan batch yang tidak tercatat.

Tidak ada biaya tambahan untuk menggunakan batch yang dicatat di Amazon Keyspaces. Anda hanya membayar untuk penulisan yang merupakan bagian dari operasi batch Anda. Amazon Keyspaces melakukan dua penulisan yang mendasari setiap baris dalam batch: satu untuk menyiapkan baris untuk batch dan satu untuk melakukan batch. Saat merencanakan kapasitas untuk tabel yang menggunakan batch yang dicatat, ingatlah bahwa setiap baris dalam batch membutuhkan dua kali kapasitas operasi penulisan standar. Misalnya, jika aplikasi Anda menjalankan satu batch yang dicatat per detik dengan tiga baris 1KB, Anda perlu menyediakan enam unit kapasitas tulis (WCU) dibandingkan dengan hanya tiga WCU untuk penulisan individual atau batch yang tidak tercatat.

Untuk informasi tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

Praktik terbaik untuk operasi batch

Pertimbangkan praktik yang disarankan berikut saat menggunakan operasi batch Amazon Keyspaces.

- Aktifkan penskalaan otomatis sehingga Anda memiliki kapasitas throughput yang cukup untuk tabel Anda untuk menangani operasi batch dan persyaratan throughput tambahan dari batch yang dicatat.
- Gunakan operasi individual atau batch yang tidak tercatat saat operasi dapat berjalan secara independen tanpa memengaruhi kebenaran aplikasi.
- Rancang aplikasi Anda untuk meminimalkan pembaruan bersamaan ke baris yang sama, karena operasi batch simultan dapat bertentangan dan gagal.
- Untuk konsumsi data massal throughput tinggi tanpa persyaratan atomisitas, gunakan operasi penulisan individual atau batch yang tidak tercatat.

Konsistensi dan konkurensi

Amazon Keyspaces memberlakukan aturan konsistensi dan konkurensi berikut untuk batch yang dicatat:

- Semua operasi batch menggunakan tingkat LOCAL\_QUORUM konsistensi.
- Batch bersamaan yang mempengaruhi baris yang berbeda dapat dieksekusi secara bersamaan.
- Bersamaan INSERT, UPDATE, atau DELETE operasi pada baris yang terlibat dalam batch yang sedang berlangsung gagal dengan konflik.

## Operator dan kondisi yang didukung

Operator WHERE klausa yang didukung:

- Kesetaraan (=)

Operator yang tidak didukung:

- Operator jangkauan (>, <, >=, <=)
- Operator IN
- Operator LIKE
- Operator BETWEEN

Tidak didukung dalam batch yang dicatat:

- Beberapa pernyataan yang mempengaruhi baris yang sama
- Operasi kontra
- Rentang menghapus

Kondisi kegagalan pernyataan batch yang dicatat

Operasi batch yang dicatat mungkin gagal dalam salah satu kasus berikut:

- Ekspresi kondisi (seperti IF NOT EXISTS atau IF) dievaluasi menjadi false.
- Satu atau lebih operasi berisi parameter yang tidak valid.
- Permintaan bertentangan dengan operasi batch lain yang berjalan pada baris yang sama.
- Tabel tidak memiliki kapasitas yang disediakan yang memadai.
- Baris melebihi batas ukuran maksimum.
- Format data input tidak valid.

## Pernyataan Batch dan replikasi Multi-wilayah

Dalam penerapan Multi-wilayah:



- Operasi Wilayah Sumber bersifat sinkron dan atom.
- Operasi Wilayah Tujuan tidak sinkron.
- Semua operasi batch bereplikasi ke Wilayah tujuan, tetapi mungkin tidak mempertahankan isolasi selama aplikasi.

### Pantau operasi batch

Anda dapat memantau operasi batch menggunakan CloudWatch metrik Amazon untuk melacak kinerja, kesalahan, dan pola penggunaan. Amazon Keyspaces menyediakan CloudWatch metrik berikut untuk memantau operasi batch per tabel:

- `SuccessfulRequestCount`— Lacak operasi batch yang berhasil.
- `Latency`- Ukur kinerja operasi batch.
- `ConsumedWriteCapacityUnits`— Memantau konsumsi kapasitas operasi batch.

Untuk informasi selengkapnya, lihat [the section called “Metrik Amazon Keyspaces”](#).

Selain CloudWatch metrik, Anda dapat menggunakan AWS CloudTrail untuk mencatat semua tindakan API Amazon Keyspaces. Setiap tindakan API dalam batch dicatat CloudTrail sehingga memudahkan untuk melacak dan mengaudit operasi batch di tabel Amazon Keyspaces Anda.

### Contoh operasi Batch

Berikut ini adalah contoh pernyataan batch dasar yang dicatat.

```
BEGIN BATCH
 INSERT INTO users (id, firstname, lastname) VALUES (1, 'John', 'Doe');
 INSERT INTO users (id, firstname, lastname) VALUES (2, 'Jane', 'Smith');
APPLY BATCH;
```

Ini adalah contoh dari batch yang mencakup INSERT, UPDATE, dan DELETE pernyataan.

```
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;
```

Ini adalah contoh batch yang menggunakan stempel waktu sisi klien.

```
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;
```

Ini adalah contoh dari batch bersyarat.

```
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;
```

Ini adalah contoh batch yang menggunakan 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;
```

Ini adalah contoh pernyataan batch yang memperbarui beberapa tabel.

```
BEGIN BATCH
 INSERT INTO users (id, firstname) VALUES (1, 'John');
```

```
INSERT INTO user_emails (user_id, email) VALUES (1, 'john@example.com');
APPLY BATCH;
```

Ini adalah contoh operasi batch menggunakan tipe yang ditentukan pengguna (UDT). Contohnya mengasumsikan bahwa UDT `address` ada.

```
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;
```

## Pesan hasil dengan **ORDER BY** di Amazon Keyspaces

**ORDER BY** klausa menentukan urutan dari hasil yang dikembalikan dalam sebuah **SELECT** pernyataan. Pernyataan ini mengambil daftar nama kolom sebagai argumen dan untuk setiap kolom Anda dapat menentukan urutan pengurutan untuk data. Anda hanya dapat menentukan kolom pengelompokan dalam klausa pengurutan, kolom non-pengelompokan tidak diperbolehkan.

Dua opsi urutan pengurutan yang tersedia untuk hasil yang dikembalikan adalah **ASC** untuk urutan urutan naik dan **DESC** untuk urutan menurun.

```
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    |

Jika Anda tidak menentukan urutan pengurutan dalam pernyataan kueri, urutan default kolom pengelompokan digunakan.

Urutan pengurutan yang mungkin dapat Anda gunakan dalam klausa pemesanan bergantung pada urutan pengurutan yang ditetapkan ke setiap kolom pengelompokan pada pembuatan tabel. Hasil kueri hanya dapat diurutkan dalam urutan yang ditentukan untuk semua kolom pengelompokan pada pembuatan tabel atau kebalikan dari urutan pengurutan yang ditentukan. Kombinasi lain yang mungkin tidak diperbolehkan.

Misalnya, jika tabel `CLUSTERING ORDER` adalah (col1 ASC, col2 DESC, col3 ASC), maka parameter yang valid untuk `ORDER BY` adalah (col1 ASC, col2 DESC, col3 ASC) atau (col1 DESC, col2 ASC, col3 DESC). Untuk informasi lebih lanjut tentang `CLUSTERING ORDER`, lihat `table_options` di bawah [the section called “CREATE TABLE”](#).

## Hasil paginasi di Amazon Keyspaces

Amazon Keyspaces secara otomatis melakukan paginasi hasil dari `SELECT` pernyataan saat data yang dibaca untuk memproses `SELECT` pernyataan melebihi 1 MB. Dengan pagination, hasil `SELECT` pernyataan dibagi menjadi “halaman” data yang berukuran 1 MB (atau kurang). Aplikasi bisa memproses halaman pertama hasil, lalu halaman kedua, dan seterusnya. Klien harus selalu memeriksa token pagination saat memproses `SELECT` kueri yang mengembalikan beberapa baris.

Jika klien memasok data `PAGE SIZE` yang memerlukan pembacaan lebih dari 1 MB data, Amazon Keyspaces akan memecah hasilnya secara otomatis menjadi beberapa halaman berdasarkan peningkatan pembacaan data sebesar 1 MB.

Misalnya, jika ukuran rata-rata sebuah baris adalah 100 KB dan Anda menentukan 20, Amazon Keyspaces melakukan paginasi data secara otomatis setelah membaca 10 baris (1000 KB data dibaca). `PAGE SIZE`

Karena Amazon Keyspaces mem-paginasi hasil berdasarkan jumlah baris yang dibaca untuk memproses permintaan dan bukan jumlah baris yang dikembalikan dalam kumpulan hasil, beberapa halaman mungkin tidak berisi baris apa pun jika Anda menjalankan kueri yang difilter.

Misalnya, jika Anda menyetel `PAGE SIZE` ke 10 dan Keyspaces mengevaluasi 30 baris untuk memproses kueri Anda `SELECT`, Amazon Keyspaces akan menampilkan tiga halaman. Jika hanya sebagian dari baris yang cocok dengan kueri Anda, beberapa halaman mungkin memiliki kurang dari 10 baris. Untuk contoh bagaimana `PAGE SIZE LIMIT` kueri dapat memengaruhi kapasitas baca, lihat [the section called “Perkiraan konsumsi kapasitas baca kueri batas”](#).

Untuk perbandingan dengan pagination Apache Cassandra, lihat. [the section called “Paginasi”](#)

## Bekerja dengan aliran change data capture (CDC) di Amazon Keyspaces

Amazon Keyspaces change data capture (CDC) merekam peristiwa perubahan tingkat baris dari tabel Amazon Keyspaces dalam waktu hampir real time.

Amazon Keyspaces CDC memungkinkan kasus penggunaan berbasis peristiwa seperti IoT industri dan deteksi penipuan serta kasus penggunaan pemrosesan data seperti pencarian teks lengkap dan arsip data. Peristiwa perubahan yang ditangkap Amazon Keyspaces CDC dalam aliran dapat dikonsumsi oleh aplikasi hilir yang menjalankan fungsi penting bisnis seperti analitik data, pencarian teks, ML training/inference, dan pencadangan data berkelanjutan untuk arsip. Misalnya, Anda dapat mentransfer data streaming ke layanan AWS analitik dan penyimpanan seperti Amazon OpenSearch Service, Amazon Redshift, dan Amazon S3 untuk diproses lebih lanjut.

Amazon Keyspaces CDC menawarkan catatan perubahan yang diurutkan waktu dan de-duplikasi untuk tabel, dengan penskalaan otomatis throughput data dan waktu retensi hingga 24 jam.

Aliran CDC Amazon Keyspaces sepenuhnya tanpa server, dan Anda tidak perlu mengelola infrastruktur data untuk menangkap peristiwa perubahan. Selain itu, Amazon Keyspaces CDC tidak menggunakan kapasitas tabel apa pun untuk komputasi atau penyimpanan. Untuk informasi selengkapnya, lihat [the section called “Cara kerja aliran CDC”](#).

Anda dapat menggunakan Amazon Keyspaces Streams API untuk membangun aplikasi yang menggunakan aliran CDC Amazon Keyspaces dan mengambil tindakan berdasarkan konten. Untuk titik akhir yang tersedia, lihat [the section called “Akses titik akhir aliran CDC”](#).

Untuk daftar lengkap semua operasi yang tersedia untuk Amazon Keyspaces di API Streams, lihat Referensi API Amazon [Keyspaces Streams](#).

## Topik

- [Cara kerja stream change data capture \(CDC\) di Amazon Keyspaces](#)
- [Cara menggunakan stream change data capture \(CDC\) di Amazon Keyspaces](#)

## Cara kerja stream change data capture (CDC) di Amazon Keyspaces

Bagian ini memberikan gambaran umum tentang cara kerja aliran change data capture (CDC) di Amazon Keyspaces.

Amazon Keyspaces change data capture (CDC) merekam urutan urutan modifikasi tingkat baris di tabel Amazon Keyspaces dan menyimpan informasi ini dalam log yang disebut stream hingga 24 jam. Setiap modifikasi tingkat baris menghasilkan catatan CDC baru yang menyimpan informasi kolom kunci utama serta status “sebelum” dan “setelah” baris termasuk semua kolom. Aplikasi dapat mengakses aliran dan melihat mutasi dalam waktu nyaris nyata.

Saat Anda mengaktifkan CDC di meja Anda, Amazon Keyspaces membuat aliran CDC baru dan mulai menangkap informasi tentang setiap modifikasi dalam tabel. Aliran CDC memiliki Nama Sumber Daya Amazon (ARN) dengan format berikut:

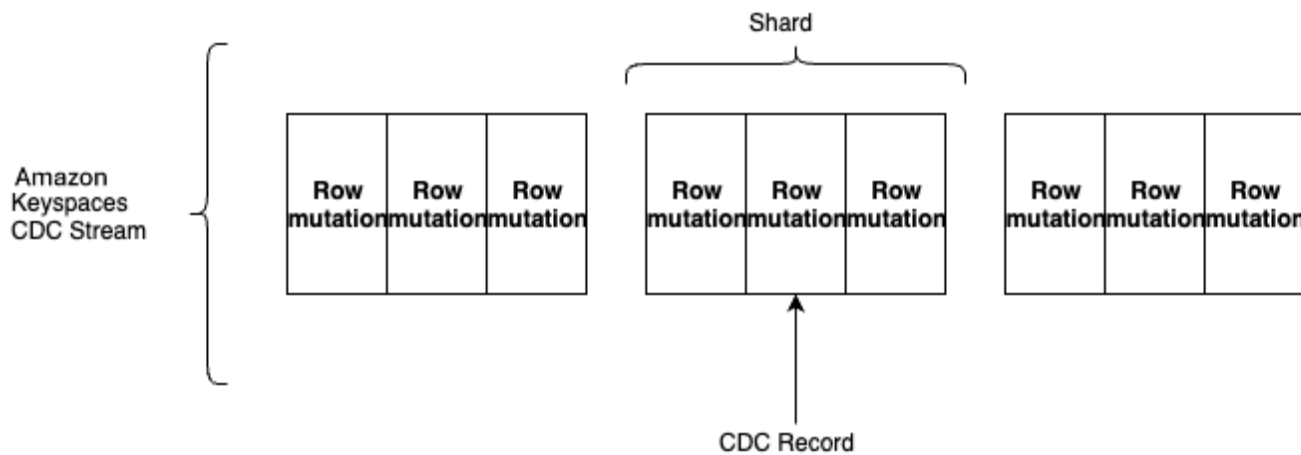
```
arn:${Partition}:cassandra:{Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

Anda dapat memilih jenis informasi atau jenis tampilan yang dikumpulkan oleh aliran CDC untuk setiap rekaman saat pertama kali mengaktifkan aliran CDC. Anda tidak dapat mengubah jenis tampilan aliran sesudahnya. Amazon Keyspaces mendukung jenis tampilan berikut:

- **NEW\_AND\_OLD\_IMAGES**— Menangkap versi baris sebelum dan sesudah mutasi. Ini adalah opsi default.
- **NEW\_IMAGE**— Menangkap versi baris setelah mutasi.
- **OLD\_IMAGE**— Menangkap versi baris sebelum mutasi.
- **KEYS\_ONLY**— Menangkap kunci partisi dan pengelompokan baris yang bermutasi.

Setiap aliran CDC terdiri dari catatan. Setiap catatan mewakili modifikasi baris tunggal dalam tabel Amazon Keyspaces. Catatan secara logis diatur ke dalam kelompok-kelompok yang dikenal sebagai

pecahan. Kelompok-kelompok ini secara logis diatur oleh rentang kunci utama (kombinasi kunci partisi, rentang kunci pengelompokan) dan merupakan konstruksi internal Amazon Keyspaces. Setiap pecahan bertindak sebagai wadah untuk beberapa catatan, dan berisi informasi yang diperlukan untuk mengakses dan mengulang melalui catatan ini.



Setiap catatan CDC diberi nomor urut, yang mencerminkan urutan catatan diterbitkan dalam pecahan. Nomor urut dijamin akan meningkat dan unik dalam setiap pecahan.

Amazon Keyspaces membuat dan menghapus pecahan secara otomatis. Berdasarkan beban lalu lintas Amazon Keyspaces juga dapat membagi atau menggabungkan pecahan dari waktu ke waktu. Misalnya, Amazon Keyspaces dapat membagi satu pecahan menjadi beberapa pecahan baru atau menggabungkan pecahan menjadi pecahan tunggal baru. Amazon Keyspaces API mempublikasikan informasi aliran shard dan CDC untuk memungkinkan aplikasi yang mengkonsumsi memproses catatan dalam urutan yang benar dengan mengakses seluruh grafik garis keturunan pecahan.

Amazon Keyspaces CDC didasarkan pada prinsip-prinsip berikut yang dapat Anda andalkan saat membuat aplikasi:

- Setiap catatan mutasi tingkat baris muncul tepat sekali dalam aliran CDC.
- Saat Anda mengkonsumsi pecahan dalam urutan garis keturunan, setiap catatan mutasi tingkat baris muncul dalam urutan yang sama dengan urutan mutasi aktual pada kunci utama.

## Topik

- [Cara kerja retensi data untuk aliran CDC di Amazon Keyspaces](#)
- [Cara kedaluwarsa data Time to Live \(TTL\) bekerja dengan aliran CDC di Amazon Keyspaces](#)
- [Cara kerja operasi batch untuk aliran CDC di Amazon Keyspaces](#)

- [Cara kerja kolom statis di aliran CDC di Amazon Keyspaces](#)
- [Cara kerja enkripsi saat istirahat untuk aliran CDC di Amazon Keyspaces](#)
- [Cara kerja replikasi multi-wilayah untuk aliran CDC di Amazon Keyspaces](#)
- [Streaming CDC dan integrasi dengan layanan AWS](#)

## Cara kerja retensi data untuk aliran CDC di Amazon Keyspaces

Amazon Keyspaces menyimpan catatan dalam aliran CDC selama 24 jam. Anda tidak dapat mengubah periode retensi. Jika Anda menonaktifkan CDC di atas meja, data dalam aliran terus dapat dibaca selama 24 jam. Setelah waktu ini, data kedaluwarsa dan catatan dihapus secara otomatis.

## Cara kedaluwarsa data Time to Live (TTL) bekerja dengan aliran CDC di Amazon Keyspaces

Amazon Keyspaces menunjukkan waktu kedaluwarsa pada column/cell tingkat serta tingkat baris dalam bidang metadata yang disebut `expirationTime` dalam catatan perubahan CDC. Saat Amazon Keyspaces TTL mendeteksi kedaluwarsa sel, CDC membuat catatan perubahan baru yang menunjukkan TTL sebagai asal perubahan. Untuk informasi lebih lanjut tentang TTL, lihat [the section called “Data kedaluwarsa dengan Time to Live”](#).

## Cara kerja operasi batch untuk aliran CDC di Amazon Keyspaces

Operasi Batch secara internal dibagi menjadi modifikasi tingkat baris individu. Amazon Keyspaces menyimpan semua catatan dalam aliran CDC pada tingkat baris, bahkan jika modifikasi terjadi dalam operasi batch. Amazon Keyspaces mempertahankan urutan catatan dalam aliran CDC dalam urutan yang sama dengan urutan mutasi yang terjadi pada tingkat baris atau pada kunci utama.

## Cara kerja kolom statis di aliran CDC di Amazon Keyspaces

Nilai kolom statis dibagi di antara semua baris dalam partisi di Cassandra. Karena perilaku ini, Amazon Keyspaces menangkap pembaruan apa pun ke kolom statis sebagai catatan terpisah dalam aliran CDC. Contoh berikut merangkum perilaku mutasi kolom statis:

- Ketika hanya kolom statis diperbarui, aliran CDC berisi modifikasi baris untuk kolom statis sebagai satu-satunya kolom di baris.
- Ketika baris diperbarui tanpa perubahan pada kolom statis, aliran CDC berisi modifikasi baris yang berisi semua kolom kecuali kolom statis.



- Ketika baris diperbarui bersama dengan kolom statis, aliran CDC berisi dua modifikasi baris terpisah, satu untuk kolom statis dan yang lainnya untuk sisa baris.

## Cara kerja enkripsi saat istirahat untuk aliran CDC di Amazon Keyspaces

Untuk mengenkripsi data saat istirahat di log yang dipesan CDC, Amazon Keyspaces menggunakan kunci enkripsi yang sama yang sudah digunakan untuk tabel. Untuk informasi selengkapnya tentang enkripsi diam, lihat [the section called “Enkripsi saat diam”](#).

## Cara kerja replikasi multi-wilayah untuk aliran CDC di Amazon Keyspaces

Anda dapat mengaktifkan dan menonaktifkan aliran CDC untuk setiap replika tabel Multi-wilayah dengan menggunakan `update-table` API atau perintah CQL. `ALTER TABLE` Karena replikasi asinkron dan resolusi konflik, aliran CDC untuk tabel Multi-wilayah tidak konsisten. Wilayah AWS Oleh karena itu, catatan yang ditangkap Amazon Keyspaces dalam aliran mungkin muncul dalam urutan yang berbeda di Wilayah yang berbeda.

Untuk informasi selengkapnya tentang replikasi Multi-wilayah, lihat. [the section called “Multi-Region replikasi”](#)

## Streaming CDC dan integrasi dengan layanan AWS

### Cara bekerja dengan titik akhir VPC untuk aliran CDC di Amazon Keyspaces

Anda dapat menggunakan titik akhir VPC untuk mengakses aliran CDC Amazon Keyspaces. Untuk informasi tentang cara membuat dan mengakses titik akhir VPC untuk aliran, lihat. [the section called “Menggunakan titik akhir VPC antarmuka dengan aliran”](#)

### Cara CloudWatch kerja pemantauan dengan streaming CDC di Amazon Keyspaces

Anda dapat menggunakan Amazon CloudWatch untuk memantau panggilan API yang dilakukan ke titik akhir CDC Amazon Keyspaces. Untuk informasi selengkapnya tentang metrik yang tersedia, lihat [the section called “Metrik untuk Amazon Keyspaces mengubah pengambilan data \(CDC\)”](#).

### Cara CloudTrail kerja logging dengan streaming CDC di Amazon Keyspaces

Amazon Keyspaces CDC terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau layanan AWS di Amazon Keyspaces. CloudTrail menangkap panggilan API Data Definition Language (DDL) dan panggilan API Data Manipulation Language (DML/Data Manipulation Language) untuk Amazon Keyspaces sebagai event. Panggilan

yang diambil termasuk panggilan dari konsol Amazon Keyspaces dan panggilan terprogram ke operasi Amazon Keyspaces API.

Untuk informasi lebih lanjut tentang peristiwa CDC yang ditangkap oleh CloudTrail, lihat [the section called “Logging dengan CloudTrail”](#).

## Cara kerja penandaan untuk aliran CDC di Amazon Keyspaces

Amazon Keyspaces CDC stream adalah sumber daya yang dapat diberi tag. Anda dapat menandai aliran saat membuat tabel secara terprogram menggunakan CQL, AWS SDK, atau file. AWS CLI Anda juga dapat menandai aliran yang ada, menghapus tag, atau melihat tag aliran. Lihat informasi yang lebih lengkap di [the section called “Tandai ruang kunci, tabel, dan aliran”](#).

## Cara menggunakan stream change data capture (CDC) di Amazon Keyspaces

### Topik

- [Konfigurasi izin untuk bekerja dengan aliran CDC di Amazon Keyspaces](#)
- [Cara mengakses titik akhir aliran CDC di Amazon Keyspaces](#)
- [Aktifkan aliran CDC saat membuat tabel baru di Amazon Keyspaces](#)
- [Aktifkan aliran CDC untuk tabel yang ada di Amazon Keyspaces](#)
- [Nonaktifkan aliran CDC di Amazon Keyspaces](#)
- [Lihat aliran CDC di Amazon Keyspaces](#)
- [Akses catatan dalam aliran CDC di Amazon Keyspaces](#)
- [Gunakan Kinesis Client Library \(KCL\) untuk memproses stream Amazon Keyspaces](#)

## Konfigurasi izin untuk bekerja dengan aliran CDC di Amazon Keyspaces

Untuk mengaktifkan aliran CDC, prinsipal, misalnya pengguna atau peran IAM, memerlukan izin berikut.

Untuk informasi lebih lanjut tentang AWS Identity and Access Management, lihat [the section called “AWS Identity and Access Management”](#).

### Izin untuk mengaktifkan aliran CDC untuk tabel

Untuk mengaktifkan aliran CDC untuk tabel Amazon Keyspaces, prinsipal pertama-tama memerlukan izin untuk membuat atau mengubah tabel dan kedua izin untuk membuat peran terkait layanan.

[AWSServiceRoleForAmazonKeyspacesCDC](#) Amazon Keyspaces menggunakan peran terkait layanan untuk mempublikasikan CloudWatch metrik ke akun Anda atas nama Anda

Kebijakan IAM berikut adalah contohnya.

```
{
 "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"
 }
 }
 }
]
}
```

Untuk menonaktifkan aliran, hanya ALTER TABLE izin yang diperlukan.

Izin untuk melihat aliran CDC

Untuk melihat atau mencantumkan aliran CDC, prinsipal memerlukan izin baca untuk ruang kunci sistem. Untuk informasi selengkapnya, lihat [the section called “system\\_schema\\_mcs”](#).

Kebijakan IAM berikut adalah contohnya.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "cassandra:Select",
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
]
 }
]
}
```

Untuk melihat atau mencantumkan aliran CDC dengan AWS CLI atau Amazon Keyspaces API, prinsipal memerlukan izin tambahan untuk tindakan `cassandra:ListStreams` dan `cassandra:GetStream`.

Kebijakan IAM berikut adalah contohnya.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "cassandra:Select",
 "cassandra:ListStreams",
 "cassandra:GetStream"
],
 "Resource": "*"
 }
]
}
```

Izin untuk membaca aliran CDC

Untuk membaca aliran CDC, prinsipal memerlukan izin berikut.

```
{
 "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"
]
}
```

Izin untuk memproses stream CDC Amazon Keyspaces dengan Kinesis Client Library (KCL)

Untuk memproses aliran CDC Amazon Keyspaces dengan KCL, prinsipal IAM memerlukan izin berikut.

- Amazon Keyspaces— Read-only akses ke aliran CDC Amazon Keyspaces yang ditentukan.
- DynamoDB— Izin untuk membuat shard lease tabel, membaca dan menulis akses ke tabel, dan akses baca ke indeks seperti yang diperlukan untuk pemrosesan aliran KCL.
- CloudWatch— Izin untuk mempublikasikan data metrik dari Amazon Keyspaces CDC mengalirkan pemrosesan dengan KCL ke namespace aplikasi klien KCL Anda di akun Anda. CloudWatch Untuk informasi selengkapnya tentang pemantauan, lihat [Memantau Perpustakaan Klien Kinesis dengan 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": "*"
 }
]
}

```

## Cara mengakses titik akhir aliran CDC di Amazon Keyspaces

Amazon Keyspaces mempertahankan [titik akhir](#) terpisah untuk keyspaces/tables dan untuk aliran CDC di masing-masing tempat Wilayah AWS Amazon Keyspaces tersedia. Untuk mengakses aliran CDC, pilih Wilayah tabel dan ganti `cassandra` awalan dengan `cassandra-streams` nama titik akhir seperti yang ditunjukkan pada contoh berikut:

| Nama wilayah                | Region    | Titik akhir untuk ruang kunci dan tabel   | Titik akhir untuk aliran                          |
|-----------------------------|-----------|-------------------------------------------|---------------------------------------------------|
| US East (Northern Virginia) | us-east-1 | <code>cassandra</code> .us-east-1.api.aws | <code>cassandra-streams</code> .us-east-1.api.aws |

Tabel berikut berisi daftar lengkap endpoint publik yang tersedia untuk Amazon Keyspaces change data capture streams. Amazon Keyspaces CDC streams mendukung IPv4 dan IPv6. Semua titik akhir publik, misalnya `cassandra-streams.us-east-1.api.aws`, adalah titik akhir tumpukan ganda yang dapat dikonfigurasi untuk IPv4 dan IPv6.

| Nama Wilayah    | Wilayah   | Titik Akhir                                      | Protokol |  |
|-----------------|-----------|--------------------------------------------------|----------|--|
| AS Timur (Ohio) | us-east-2 | <code>cassandra-streams.us-east-2.api.aws</code> | HTTPS    |  |

| Nama Wilayah              | Wilayah        | Titik Akhir                              | Protokol |  |
|---------------------------|----------------|------------------------------------------|----------|--|
| AS Timur (Virginia Utara) | us-east-1      | cassandra-streams.us-east-1.api.aws      | HTTPS    |  |
| US West (N. California)   | us-west-1      | cassandra-streams.us-west-1.api.aws      | HTTPS    |  |
| AS Barat (Oregon)         | us-west-2      | cassandra-streams.us-west-2.api.aws      | HTTPS    |  |
| Afrika (Cape Town)        | af-south-1     | cassandra-streams.af-south-1.api.aws     | HTTPS    |  |
| Asia Pasifik (Hong Kong)  | ap-east-1      | cassandra-streams.ap-east-1.api.aws      | HTTPS    |  |
| Asia Pasifik (Malaysia)   | ap-southeast-5 | cassandra-streams.ap-southeast-5.api.aws | HTTPS    |  |
| Asia Pasifik (Mumbai)     | ap-south-1     | cassandra-streams.ap-south-1.api.aws     | HTTPS    |  |
| Asia Pasifik (Seoul)      | ap-northeast-2 | cassandra-streams.ap-northeast-2.api.aws | HTTPS    |  |



| Nama Wilayah             | Wilayah        | Titik Akhir                              | Protokol |  |
|--------------------------|----------------|------------------------------------------|----------|--|
| Asia Pasifik (Singapura) | ap-southeast-1 | cassandra-streams.ap-southeast-1.api.aws | HTTPS    |  |
| Asia Pasifik (Sydney)    | ap-southeast-2 | cassandra-streams.ap-southeast-2.api.aws | HTTPS    |  |
| Asia Pasifik (Thailand)  | ap-tenggara-7  | cassandra-streams.ap-southeast-7.api.aws | HTTPS    |  |
| Asia Pacific (Tokyo)     | ap-northeast-1 | cassandra-streams.ap-northeast-1.api.aws | HTTPS    |  |
| Kanada (Pusat)           | ca-central-1   | cassandra-streams.ca-central-1.api.aws   | HTTPS    |  |
| Eropa (Frankfurt)        | eu-central-1   | cassandra-streams.eu-central-1.api.aws   | HTTPS    |  |
| Eropa (Irlandia)         | eu-west-1      | cassandra-streams.eu-west-1.api.aws      | HTTPS    |  |
| Eropa (London)           | eu-west-2      | cassandra-streams.eu-west-2.api.aws      | HTTPS    |  |
| Eropa (Paris)            | eu-west-3      | cassandra-streams.eu-west-3.api.aws      | HTTPS    |  |

| Nama Wilayah                | Wilayah       | Titik Akhir                             | Protokol |  |
|-----------------------------|---------------|-----------------------------------------|----------|--|
| Eropa (Stockholm)           | eu-north-1    | cassandra-streams.eu-north-1.api.aws    | HTTPS    |  |
| Timur Tengah (Bahrain)      | me-south-1    | cassandra-streams.me-south-1.api.aws    | HTTPS    |  |
| Timur Tengah (UAE)          | me-central-1  | cassandra-streams.me-central-1.api.aws  | HTTPS    |  |
| Amerika Selatan (Sao 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    |  |

## Aktifkan aliran CDC saat membuat tabel baru di Amazon Keyspaces

Untuk mengaktifkan aliran CDC saat Anda membuat tabel, Anda dapat menggunakan CREATE TABLE pernyataan di CQL atau `create-table` perintah dengan AWS CLI

Untuk setiap baris yang diubah dalam tabel, Amazon Keyspaces dapat menangkap perubahan berikut berdasarkan pilihan `view_type` yang `cdc_specification` Anda pilih:

- NEW\_AND\_OLD\_IMAGES— kedua versi baris, sebelum dan sesudah perubahan. Ini adalah opsi default.
- NEW\_IMAGE— versi baris setelah perubahan.
- OLD\_IMAGE— versi baris sebelum perubahan.
- KEYS\_ONLY— kunci partisi dan pengelompokan baris yang diubah.

Untuk informasi tentang cara menandai aliran, lihat [the section called “Buat aliran dengan tag dengan tabel baru”](#).

#### Note

Amazon Keyspaces CDC memerlukan keberadaan peran terkait layanan (AWSServiceRoleForAmazonKeyspacesCDC) yang menerbitkan data metrik dari aliran CDC Amazon Keyspaces ke dalam akun Anda atas nama Anda. "cloudwatch:namespace": "AWS/Cassandra" CloudWatch Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Aliran CDC”](#).

## Cassandra Query Language (CQL)

Aktifkan aliran CDC saat Anda membuat tabel dengan 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. Untuk mengonfirmasi pengaturan streaming, Anda dapat menggunakan pernyataan berikut.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

Output dari pernyataan itu akan terlihat mirip dengan ini.

```
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

## Aktifkan aliran CDC saat Anda membuat tabel dengan AWS CLI

1. Untuk membuat aliran, Anda dapat menggunakan sintaks berikut.

```
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. Output dari perintah itu menunjukkan `create-table` respons standar dan terlihat mirip dengan contoh ini.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable" }
```

## Aktifkan aliran CDC untuk tabel yang ada di Amazon Keyspaces

Untuk mengaktifkan aliran CDC untuk tabel yang ada, Anda dapat menggunakan ALTER TABLE pernyataan di CQL, update-table perintah dengan AWS CLI, atau Anda dapat menggunakan konsol.

Untuk setiap baris yang diubah dalam tabel, Amazon Keyspaces dapat menangkap perubahan berikut berdasarkan pilihan `view_type` yang `cdc_specification` Anda pilih:

- NEW\_AND\_OLD\_IMAGES— kedua versi baris, sebelum dan sesudah perubahan. Ini adalah opsi default.

- NEW\_IMAGE— versi baris setelah perubahan.
- OLD\_IMAGE— versi baris sebelum perubahan.
- KEYS\_ONLY— kunci partisi dan pengelompokan baris yang diubah.

Untuk informasi tentang cara menandai aliran, lihat [the section called “Tambahkan tag baru ke aliran”](#).

#### Note

Amazon Keyspaces CDC memerlukan keberadaan peran terkait layanan (AWSServiceRoleForAmazonKeyspacesCDC) yang menerbitkan data metrik dari aliran CDC Amazon Keyspaces ke dalam akun Anda atas nama Anda. "cloudwatch:namespace": "AWS/Cassandra" CloudWatch Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Aliran CDC”](#).

## Cassandra Query Language (CQL)

Aktifkan aliran (aliran CDC) dengan CQL

Anda dapat menggunakan ALTER TABLE untuk mengaktifkan aliran untuk tabel yang ada.

1. Contoh berikut membuat aliran yang hanya menangkap perubahan pada partisi dan kunci pengelompokan dari baris yang diubah.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = TRUE
AND CUSTOM_PROPERTIES={'cdc_specification': {'view_type': 'KEYS_ONLY'}};
```

2. Untuk memverifikasi pengaturan aliran, Anda dapat menggunakan pernyataan berikut.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

Output dari pernyataan terlihat mirip dengan ini.

```
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

### Buat aliran CDC dengan AWS CLI

1. Untuk membuat aliran untuk tabel yang ada, Anda dapat menggunakan sintaks berikut.

```
aws keyspaces update-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--cdc-specification status=ENABLED,viewType=NEW_AND_OLD_IMAGES
```

2. Output dari perintah itu menunjukkan create-table respons standar dan terlihat mirip dengan contoh ini.

```
{ "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/
table/mytable" }
```

## Console

### Aktifkan aliran CDC dengan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih tabel dari daftar.
3. Pilih tab Streams.
4. Pilih Edit untuk mengaktifkan aliran.
5. Pilih Aktifkan aliran.
6. Pilih Lihat jenis aliran. Pilihan berikut tersedia. Perhatikan bahwa Anda tidak dapat mengubah jenis tampilan aliran setelah dibuat.

- Gambar baru dan lama — Amazon Keyspaces menangkap kedua versi baris, sebelum dan sesudah perubahan. Ini adalah opsi default.
- Gambar baru - Amazon Keyspaces hanya menangkap versi baris setelah perubahan.
- Gambar lama - Amazon Keyspaces hanya menangkap versi baris sebelum perubahan.
- Hanya kunci utama — Amazon Keyspaces hanya menangkap partisi dan pengelompokan kolom kunci dari baris yang diubah.

7. Untuk menyelesaikannya, pilih Simpan perubahan.

## Nonaktifkan aliran CDC di Amazon Keyspaces

Untuk menonaktifkan aliran CDC di ruang kunci, Anda dapat menggunakan ALTER TABLE pernyataan di CQL, update-table perintah dengan AWS CLI, atau konsol.

### Cassandra Query Language (CQL)

Nonaktifkan aliran (aliran CDC) dengan CQL

1. Untuk menonaktifkan aliran, Anda dapat menggunakan pernyataan berikut.

```
ALTER TABLE mykeyspace.mytable
WITH cdc = FALSE;
```

2. Untuk mengonfirmasi bahwa aliran dinonaktifkan, Anda dapat menggunakan pernyataan berikut.

```
SELECT keyspace_name, table_name, cdc, custom_properties FROM
system_schema_mcs.tables WHERE keyspace_name = 'mykeyspace' AND table_name =
'mytable';
```

Output dari pernyataan itu terlihat mirip dengan ini.

```
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

### Nonaktifkan aliran (aliran CDC) dengan AWS CLI

1. Untuk menonaktifkan aliran, Anda dapat menggunakan perintah berikut.

```
aws keyspaces update-table \
--keyspace-name 'mykeyspace' \
--table-name 'mytable' \
--cdc-specification status=DISABLED
```

2. Output dari perintah terlihat mirip dengan contoh ini.

```
{
 "keyspaceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
my_keyspace/",
 "streamName": "my_stream"
}
```

## Console

### Nonaktifkan aliran (aliran CDC) dengan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih tabel dari daftar.
3. Pilih tab Streams.
4. Pilih Edit.
5. Batalkan pilihan Aktifkan aliran.
6. Pilih Simpan perubahan untuk menonaktifkan aliran.



## Lihat aliran CDC di Amazon Keyspaces

Untuk melihat atau mencantumkan semua aliran di ruang kunci, Anda dapat melakukan kueri tabel `system_schema_mcs.streams` di ruang kunci sistem menggunakan pernyataan di CQL, atau menggunakan `list-stream` perintah `get-stream` and `denon`, atau konsol. AWS CLI

Untuk izin yang diperlukan, lihat [the section called “Mengatur Konfigurasi Izin”](#).

### Cassandra Query Language (CQL)

#### Lihat aliran CDC dengan CQL

- Untuk memantau status CDC tabel Anda, Anda dapat menggunakan pernyataan berikut.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name='my_keyspace' and table_name='my_table';
```

Output dari perintah terlihat mirip dengan ini.

```
...
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

#### Lihat aliran CDC dengan AWS CLI

1. Contoh ini menunjukkan cara melihat informasi aliran untuk tabel.

```
aws keyspaces get-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table'
```

Output dari perintah terlihat seperti ini.

```
{
 "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. Anda dapat mencantumkan semua aliran di akun Anda dalam yang ditentukan Wilayah AWS. Perintah berikut adalah contohnya.

```
aws keyspacesstreams list-streams --region us-east-1
```

Output dari perintah bisa terlihat mirip dengan ini.

```
{
 "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",
 },
 {
 "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. Anda juga dapat mencantumkan aliran CDC untuk ruang kunci tertentu menggunakan parameter berikut.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --region us-east-1
```

Output dari perintah terlihat mirip dengan ini.

```

{
 "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. Anda juga dapat mencantumkan aliran CDC untuk tabel tertentu menggunakan parameter berikut.

```
aws keyspacesstreams list-streams --keyspace-name ks_1 --table-name t2 --
region us-east-1
```

Output dari perintah terlihat mirip dengan ini.

```

{
 "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

Lihat aliran CDC di konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih tabel dari daftar.
3. Pilih tab Streams untuk meninjau detail streaming.

## Akses catatan dalam aliran CDC di Amazon Keyspaces

Untuk mengakses rekaman dalam aliran, Anda menggunakan [Amazon Keyspaces Streams](#) API. Bagian berikut berisi contoh tentang cara mengakses catatan menggunakan file AWS CLI.

Untuk izin yang diperlukan, lihat [the section called “Mengatur Konfigurasi Izin”](#).

Mengakses catatan dalam aliran menggunakan AWS CLI

1. Anda dapat menggunakan Amazon Keyspaces Streams API untuk mengakses catatan perubahan aliran. Untuk informasi selengkapnya, lihat [Referensi API Amazon Keyspaces Streams](#). Untuk mengambil pecahan dalam aliran, Anda dapat menggunakan get-stream API seperti yang ditunjukkan pada contoh berikut.

```
aws keyspacesstreams get-stream \
--stream-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable/stream/STREAM_LABEL'
```

Berikut adalah contoh output.

```
{
 "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. Untuk mengambil catatan dari aliran, Anda mulai dengan mendapatkan iterator yang memberi Anda titik awal untuk mengakses catatan. Untuk melakukan ini, Anda dapat menggunakan pecahan dalam aliran CDC yang dikembalikan oleh API pada langkah sebelumnya. Untuk mengumpulkan iterator, Anda dapat menggunakan `get-shard-iterator` API. Untuk contoh ini, Anda menggunakan iterator tipe `TRIM_HORIZON` yang mengambil dari titik atau awal yang dipangkas terakhir) dari pecahan.

```
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'
```

Output dari perintah terlihat seperti pada contoh berikut.

```
{
 "ShardIterator": "<SHARD_ITERATOR>"
}
```

3. Untuk mengambil catatan CDC menggunakan `get-records` API, Anda dapat menggunakan iterator yang dikembalikan pada langkah terakhir. Perintah berikut adalah contohnya.

```
aws keyspacesstreams get-records \
--shard-iterator 'SHARD_ITERATOR' \
--limit 100
```

## Gunakan Kinesis Client Library (KCL) untuk memproses stream Amazon Keyspaces

Topik ini menjelaskan cara menggunakan Kinesis Client Library (KCL) untuk mengkonsumsi dan memproses data dari Amazon Keyspaces change data capture (CDC) stream.

Alih-alih bekerja langsung dengan Amazon Keyspaces Streams API, bekerja dengan Kinesis Client Library (KCL) memberikan banyak manfaat, misalnya:

- Dibangun pada pelacakan garis keturunan pecahan dan penanganan iterator.
- Penyeimbangan beban otomatis di seluruh pekerja.
- Toleransi kesalahan dan pemulihan dari kegagalan pekerja.
- Checkpointing untuk melacak kemajuan pemrosesan.
- Adaptasi terhadap perubahan kapasitas aliran.
- Komputasi terdistribusi yang disederhanakan untuk memproses catatan CDC.

Bagian berikut menguraikan mengapa dan bagaimana menggunakan Kinesis Client Library (KCL) untuk memproses stream dan memberikan contoh untuk memproses aliran CDC Amazon Keyspaces dengan KCL.

Untuk informasi tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

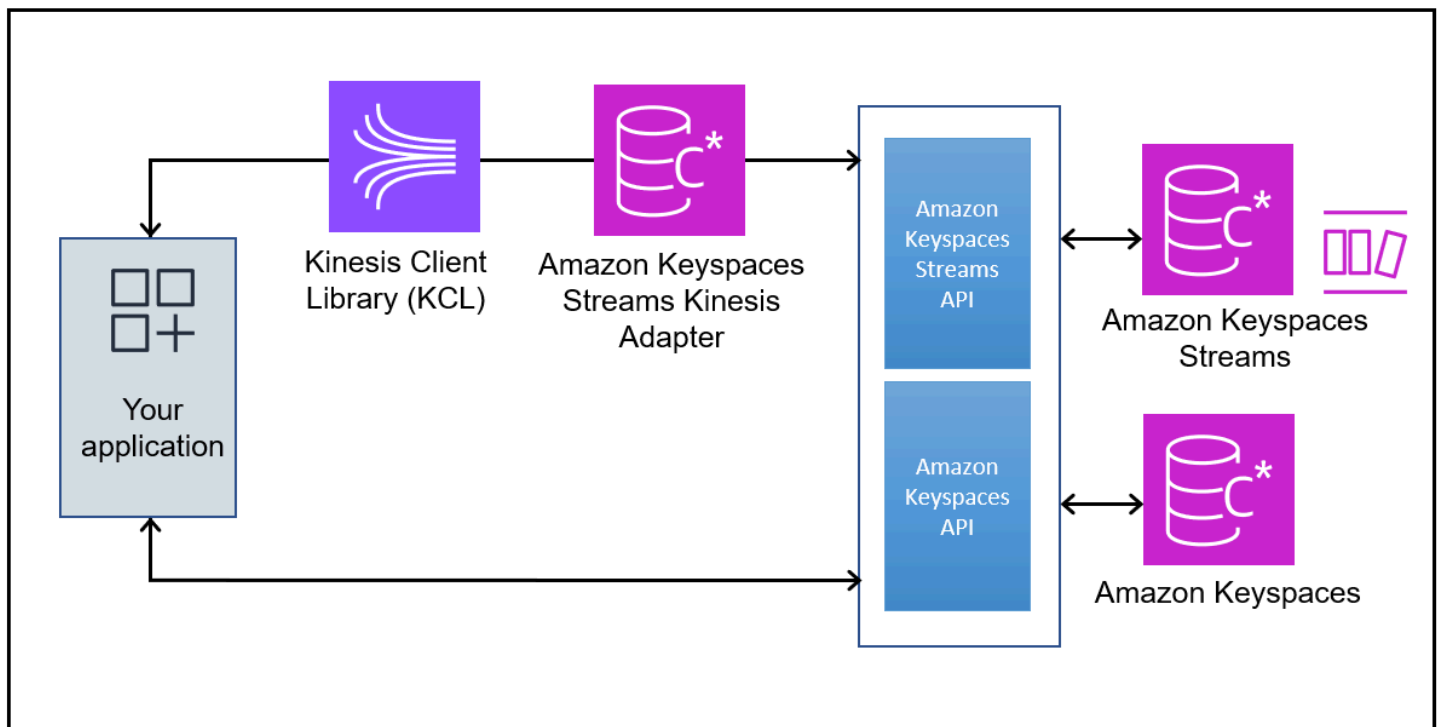
### Apa itu Perpustakaan Klien Kinesis?

Kinesis Client Library (KCL) adalah pustaka perangkat lunak Java mandiri yang dirancang untuk menyederhanakan proses konsumsi dan pemrosesan data dari aliran. KCL menangani banyak tugas kompleks yang terkait dengan komputasi terdistribusi, memungkinkan Anda fokus pada penerapan logika bisnis Anda saat memproses data aliran. KCL mengelola aktivitas seperti load balancing di beberapa pekerja, menanggapi kegagalan pekerja, memeriksa catatan yang diproses, dan menanggapi perubahan jumlah pecahan dalam aliran.

Untuk memproses aliran CDC Amazon Keyspaces, Anda dapat menggunakan pola desain yang ditemukan di KCL untuk bekerja dengan pecahan aliran dan catatan aliran. KCL menyederhanakan pengodean dengan menyediakan abstraksi yang berguna di atas Kinesis Data Streams API tingkat rendah. Untuk informasi selengkapnya tentang KCL, lihat [Mengembangkan konsumen dengan KCL](#) di Panduan Pengembang Amazon Kinesis Data Streams.

Untuk menulis aplikasi menggunakan KCL, Anda menggunakan Adaptor Kinesis Streams Amazon Keyspaces. Adaptor Kinesis mengimplementasikan antarmuka Kinesis Data Streams sehingga Anda dapat menggunakan KCL untuk mengkonsumsi dan memproses catatan dari aliran Amazon Keyspaces. Untuk petunjuk tentang cara mengatur dan menginstal adaptor Kinesis stream Amazon Keyspaces, kunjungi repositori. [GitHub](#)

Diagram berikut menunjukkan bagaimana pustaka ini berinteraksi satu sama lain.



KCL sering diperbarui untuk menggabungkan versi yang lebih baru dari pustaka yang mendasari, peningkatan keamanan, dan perbaikan bug. Kami menyarankan Anda menggunakan versi terbaru KCL untuk menghindari masalah yang diketahui dan mendapatkan manfaat dari semua peningkatan terbaru. Untuk menemukan versi KCL terbaru, lihat repositori [KCL GitHub](#).

## Konsep KCL

Sebelum Anda menerapkan aplikasi konsumen menggunakan KCL, Anda harus memahami konsep-konsep berikut:

## Aplikasi konsumen KCL

Aplikasi konsumen KCL adalah program yang memproses data dari aliran CDC Amazon Keyspaces. KCL bertindak sebagai perantara antara kode aplikasi konsumen Anda dan aliran CDC Amazon Keyspaces.

## Pekerja

Pekerja adalah unit eksekusi aplikasi konsumen KCL Anda yang memproses data dari aliran CDC Amazon Keyspaces. Aplikasi Anda dapat menjalankan beberapa pekerja yang didistribusikan di beberapa instance.

## Rekam prosesor

Pemroses rekaman adalah logika dalam aplikasi Anda yang memproses data dari pecahan di aliran CDC Amazon Keyspaces. Prosesor rekaman dipakai oleh pekerja untuk setiap pecahan yang dikelolanya.

## Sewa

Sewa merupakan tanggung jawab pemrosesan untuk pecahan. Pekerja menggunakan sewa untuk mengoordinasikan pekerja mana yang memproses pecahan mana. KCL menyimpan data sewa dalam tabel di Amazon DynamoDB.

## Pos pemeriksaan

Pos pemeriksaan adalah catatan posisi dalam pecahan di mana prosesor rekaman telah berhasil memproses catatan. Checkpointing memungkinkan aplikasi Anda untuk melanjutkan pemrosesan dari tempat yang ditinggalkan jika pekerja gagal.

Dengan adaptor Kinesis Amazon Keyspaces, Anda dapat mulai mengembangkan antarmuka KCL, dengan panggilan API diarahkan secara mulus ke titik akhir aliran Amazon Keyspaces. Untuk daftar titik akhir yang tersedia, lihat [the section called “Akses titik akhir aliran CDC”](#).

Saat aplikasi Anda dimulai, aplikasi akan memanggil KCL untuk membuat instance pekerja. Anda harus memberi pekerja informasi konfigurasi untuk aplikasi, seperti deskriptor aliran dan AWS kredensial, dan nama kelas prosesor rekaman yang Anda berikan. Saat menjalankan kode di pemroses rekaman, pekerja melakukan tugas-tugas berikut:

- Menghubungkan ke aliran
- Menghitung pecahan dalam aliran
- Mengkoordinasikan asosiasi serpihan dengan pekerja lain (jika ada)



- Membuat instance pemroses rekaman untuk setiap pecahan yang dikelolanya
- Menarik catatan dari aliran
- Mendorong rekaman ke pemroses rekaman yang sesuai
- Catatan yang diproses di pos pemeriksaan
- Menyeimbangkan asosiasi pekerja pecahan ketika jumlah instans pekerja berubah
- Menyeimbangkan asosiasi pekerja pecahan saat pecahan dipisahkan

## Menerapkan aplikasi konsumen KCL untuk aliran CDC Amazon Keyspaces

Topik ini memberikan panduan langkah demi langkah untuk menerapkan aplikasi konsumen KCL untuk memproses aliran CDC Amazon Keyspaces.

1. Prasyarat: Sebelum Anda mulai, pastikan Anda memiliki:
  - Tabel Amazon Keyspaces dengan aliran CDC
  - Izin IAM yang diperlukan untuk prinsipal IAM untuk mengakses aliran CDC Amazon Keyspaces, membuat dan mengakses tabel DynamoDB untuk pemrosesan aliran KCL, dan izin untuk mempublikasikan metrik. CloudWatch Untuk informasi selengkapnya dan contoh kebijakan, lihat [the section called “Izin untuk KCL”](#).
  - Pastikan AWS kredensial yang valid diatur dalam konfigurasi lokal Anda. Untuk informasi selengkapnya, lihat [the section called “Kelola kunci akses”](#).
  - Java Development Kit (JDK) 8 atau lebih baru
  - Persyaratan yang tercantum dalam [Readme](#) di Github.
2. Pada langkah ini, Anda menambahkan ketergantungan KCL ke proyek Anda. Untuk Maven, tambahkan yang berikut ini ke pom.xml Anda:

```
<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>
```

&lt;/dependencies&gt;

 Note

Selalu periksa versi terbaru KCL di repositori [KCL GitHub](#).

## 3. Buat kelas pabrik yang menghasilkan instance prosesor rekaman:

```
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()) {
```

```

 RecordProcessorCheckpointter 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. Buat pabrik rekaman seperti yang ditunjukkan pada contoh berikut.

```

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. Pada langkah ini Anda membuat kelas dasar untuk mengonfigurasi KClv3 dan adaptor 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. Pada langkah ini Anda menerapkan kelas prosesor rekaman untuk aplikasi Anda untuk mulai memproses peristiwa perubahan.

```

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);
}
}

```

## Praktik terbaik

Ikuti praktik terbaik ini saat menggunakan KCL dengan aliran CDC Amazon Keyspaces:

### Penanganan kesalahan

Terapkan penanganan kesalahan yang kuat di prosesor rekaman Anda untuk menangani pengecualian dengan anggun. Pertimbangkan untuk menerapkan logika coba lagi untuk kegagalan sementara.

### Frekuensi pos pemeriksaan

Seimbangkan frekuensi checkpointing untuk meminimalkan pemrosesan duplikat sekaligus memastikan pelacakan kemajuan yang wajar. Pemeriksaan yang terlalu sering dapat memengaruhi kinerja, sementara pos pemeriksaan yang terlalu jarang dapat menyebabkan pemrosesan ulang lebih banyak jika pekerja gagal.

### Penskalaan pekerja

Skala jumlah pekerja berdasarkan jumlah pecahan di aliran CDC Anda. Titik awal yang baik adalah memiliki satu pekerja per pecahan, tetapi Anda mungkin perlu menyesuaikan berdasarkan persyaratan pemrosesan Anda.

### Pemantauan

Gunakan CloudWatch metrik yang disediakan oleh KCL untuk memantau kesehatan dan kinerja aplikasi konsumen Anda. Metrik utama termasuk latensi pemrosesan, usia pos pemeriksaan, dan jumlah sewa.

### Pengujian

Uji aplikasi konsumen Anda secara menyeluruh, termasuk skenario seperti kegagalan pekerja, resharding aliran, dan berbagai kondisi beban.

## Menggunakan KCL dengan bahasa non-Java

Sementara KCL terutama perpustakaan Java, Anda dapat menggunakannya dengan bahasa pemrograman lain melalui. MultiLangDaemon MultiLangDaemon Ini adalah Java-based daemon yang mengelola interaksi antara prosesor rekaman non-Java Anda dan KCL.

KCL menyediakan dukungan untuk bahasa-bahasa berikut:

- Python

- Ruby
- Node.js
- .NET

Untuk informasi selengkapnya tentang penggunaan KCL dengan bahasa non-Java, lihat dokumentasi [MultiLangDaemon KCL](#).

## Pemecahan masalah

Bagian ini memberikan solusi untuk masalah umum yang mungkin Anda temui saat menggunakan KCL dengan aliran CDC Amazon Keyspaces.

### Pemrosesan lambat

Jika aplikasi konsumen Anda memproses catatan secara perlahan, pertimbangkan:

- Meningkatkan jumlah instans pekerja
- Mengoptimalkan logika pemrosesan rekaman Anda
- Memeriksa kemacetan dalam sistem hilir

### Pemrosesan duplikat

Jika Anda melihat pemrosesan duplikat catatan, periksa logika pos pemeriksaan Anda. Pastikan Anda melakukan checkpointing setelah berhasil memproses catatan.

### Kegagalan pekerja

Jika pekerja sering gagal, periksa:

- Kendala sumber daya (CPU, memori)
- Masalah konektivitas jaringan
- Masalah izin

### Masalah tabel sewa

Jika Anda mengalami masalah dengan tabel sewa KCL:

- Periksa apakah aplikasi Anda memiliki izin yang sesuai untuk mengakses tabel Amazon Keyspaces
- Verifikasi bahwa tabel memiliki throughput yang disediakan yang memadai

## Bekerja dengan partisi di Amazon Keyspaces

Di Apache Cassandra, partisi mengontrol data node mana yang disimpan di dalam cluster. Partisi membuat token numerik menggunakan nilai hash dari kunci partisi. Cassandra menggunakan token ini untuk mendistribusikan data di seluruh node. Klien juga dapat menggunakan token ini dalam SELECT operasi dan WHERE klausa untuk mengoptimalkan operasi baca dan tulis. Misalnya, klien dapat secara efisien melakukan query paralel pada tabel besar dengan menentukan rentang token yang berbeda untuk kueri di setiap pekerjaan paralel.

Amazon Keyspaces menyediakan tiga partisi yang berbeda.

### Murmur3Partitioner (Default)

Apache Cassandra-compatible Murmur3Partitioner. Murmur3Partitioner ini adalah partisi Cassandra default di Amazon Keyspaces dan di Cassandra 1.2 dan versi yang lebih baru.

### RandomPartitioner

Apache Cassandra-compatible RandomPartitioner. RandomPartitioner ini adalah partisi Cassandra default untuk versi lebih awal dari Cassandra 1.2.

### Partisi Default Keyspaces

DefaultPartitioner Mengembalikan hasil token fungsi yang sama seperti RandomPartitioner.

Pengaturan partisi diterapkan per Wilayah di tingkat akun. Misalnya, jika Anda mengubah partisi di US East (Virginia N.), perubahan diterapkan ke semua tabel di akun yang sama di Wilayah ini. Anda dapat dengan aman mengganti partisi Anda kapan saja. Perhatikan bahwa perubahan konfigurasi membutuhkan waktu sekitar 10 menit untuk menyelesaikannya. Anda tidak perlu memuat ulang data Amazon Keyspaces saat mengubah setelan partisi. Klien akan secara otomatis menggunakan pengaturan partisi baru saat berikutnya mereka terhubung.

## Cara mengubah partisi di Amazon Keyspaces

Anda dapat mengubah partisi dengan menggunakan Konsol Manajemen AWS atau Cassandra Query Language (CQL).

## Konsol Manajemen AWS

Untuk mengubah partisi menggunakan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pada panel navigasi, pilih Konfigurasi.
3. Pada halaman Konfigurasi, buka Edit partisi.
4. Pilih partisi yang kompatibel dengan versi Cassandra Anda. Perubahan partisi membutuhkan waktu sekitar 10 menit untuk diterapkan.

### Note

Setelah perubahan konfigurasi selesai, Anda harus memutuskan dan menyambung kembali ke Amazon Keyspaces untuk permintaan menggunakan partisi baru.

## Cassandra Query Language (CQL)

1. Untuk melihat partisi mana yang dikonfigurasi untuk akun, Anda dapat menggunakan kueri berikut.

```
SELECT partitioner from system.local;
```

Jika partisi belum diubah, kueri memiliki output sebagai berikut.

```
partitioner

com.amazonaws.cassandra.DefaultPartitioner
```

2. Untuk memperbarui partisi ke Murmur3 partisi, Anda dapat menggunakan pernyataan berikut.

```
UPDATE system.local set
partitioner='org.apache.cassandra.dht.Murmur3Partitioner' where key='local';
```

3. Perhatikan bahwa perubahan konfigurasi ini membutuhkan waktu sekitar 10 menit untuk menyelesaikannya. Untuk mengonfirmasi bahwa partisi telah disetel, Anda dapat menjalankan SELECT kueri lagi. Perhatikan bahwa karena konsistensi baca akhirnya,

respons mungkin belum mencerminkan hasil perubahan partisi yang baru saja selesai. Jika Anda mengulangi SELECT operasi lagi setelah waktu yang singkat, respons akan mengembalikan data terbaru.

```
SELECT partitioner from system.local;
```

 Note

Anda harus memutuskan dan menyambung kembali ke Amazon Keyspaces sehingga permintaan menggunakan partisi baru.

## Client-side stempel waktu di Amazon Keyspaces

Di Amazon Keyspaces, stempel waktu sisi klien adalah stempel waktu yang disimpan untuk Cassandra-compatible setiap sel di tabel Anda. Anda dapat menggunakan stempel waktu sisi klien untuk penyelesaian konflik dengan membiarkan aplikasi klien Anda menentukan urutan penulisan. Misalnya, ketika klien dari aplikasi yang didistribusikan secara global melakukan pembaruan ke data yang sama, stempel waktu sisi klien mempertahankan urutan pembaruan dilakukan pada klien. Amazon Keyspaces menggunakan stempel waktu ini untuk memproses penulisan.

Stempel waktu sisi klien Amazon Keyspaces dikelola sepenuhnya. Anda tidak perlu mengelola pengaturan sistem tingkat rendah seperti strategi pembersihan dan pemadatan.

Saat Anda menghapus data, baris ditandai untuk dihapus dengan batu nisan. Amazon Keyspaces menghapus data tombstoned secara otomatis (biasanya dalam 10 hari) tanpa memengaruhi kinerja atau ketersediaan aplikasi Anda. Data tombstoned tidak tersedia untuk pernyataan bahasa manipulasi data (DML/bahasa manipulasi data). Saat Anda terus melakukan pembacaan dan penulisan pada baris yang berisi data batu nisan, data batu nisan terus dihitung terhadap penyimpanan, unit kapasitas baca (RCU), dan unit kapasitas tulis (WCU) hingga dihapus dari penyimpanan.

Setelah stempel waktu sisi klien diaktifkan untuk sebuah tabel, Anda dapat menentukan stempel waktu dengan USING TIMESTAMP klausa dalam kueri CQL Data Manipulation Language (DHTML) Anda. Untuk informasi selengkapnya, lihat [the section called “Gunakan stempel waktu sisi klien dalam kueri”](#). Jika Anda tidak menentukan stempel waktu dalam kueri CQL Anda, Amazon Keyspaces menggunakan stempel waktu yang diteruskan oleh driver klien Anda. Jika driver klien tidak menyediakan stempel waktu, Amazon Keyspaces menetapkan stempel waktu tingkat sel

secara otomatis, karena stempel waktu tidak bisa. NULL Untuk kueri stempel waktu, Anda dapat menggunakan WRITETIME fungsi dalam pernyataan DMLmu.

Amazon Keyspaces tidak mengenakan biaya tambahan untuk mengaktifkan stempel waktu sisi klien. Namun, dengan stempel waktu sisi klien Anda menyimpan dan menulis data tambahan untuk setiap nilai di baris Anda. Hal ini dapat menyebabkan penggunaan penyimpanan tambahan dan dalam beberapa kasus penggunaan throughput tambahan. Untuk informasi selengkapnya tentang harga Amazon Keyspaces, lihat harga Amazon [Keyspaces \(untuk Apache Cassandra\)](#).

Saat stempel waktu sisi klien diaktifkan di Amazon Keyspaces, setiap kolom dari setiap baris menyimpan stempel waktu. Stempel waktu ini memakan waktu sekitar 20-40 byte (tergantung pada data Anda), dan berkontribusi pada biaya penyimpanan dan throughput untuk baris tersebut. Byte metadata ini juga dihitung terhadap kuota ukuran baris 1-MB Anda. Untuk menentukan peningkatan keseluruhan ruang penyimpanan (untuk memastikan bahwa ukuran baris tetap di bawah 1 MB), pertimbangkan jumlah kolom dalam tabel Anda dan jumlah elemen koleksi di setiap baris. Misalnya, jika tabel memiliki 20 kolom, dengan setiap kolom menyimpan 40 byte data, ukuran baris meningkat dari 800 byte menjadi 1200 byte. Untuk informasi lebih lanjut tentang cara memperkirakan ukuran baris, lihat [the section called “Perkiraan ukuran baris”](#). Selain 400 byte tambahan untuk penyimpanan, dalam contoh ini, jumlah unit kapasitas tulis (WCU) yang dikonsumsi per penulisan meningkat dari 1 WCU menjadi 2 WCU. Untuk informasi lebih lanjut tentang cara menghitung kapasitas baca dan tulis, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).

Setelah stempel waktu sisi klien dinyalakan untuk sebuah meja, Anda tidak dapat mematikannya.

Untuk mempelajari lebih lanjut tentang cara menggunakan stempel waktu sisi klien dalam kueri, lihat [the section called “Gunakan stempel waktu sisi klien dalam kueri”](#)

## Topik

- [Bagaimana stempel waktu sisi klien Amazon Keyspaces terintegrasi dengan layanan AWS](#)
- [Buat tabel baru dengan stempel waktu sisi klien di Amazon Keyspaces](#)
- [Konfigurasi stempel waktu sisi klien untuk tabel di Amazon Keyspaces](#)
- [Gunakan stempel waktu sisi klien dalam kueri di Amazon Keyspaces](#)

## Bagaimana stempel waktu sisi klien Amazon Keyspaces terintegrasi dengan layanan AWS

Metrik stempel waktu sisi klien berikut tersedia di Amazon CloudWatch untuk memungkinkan pemantauan berkelanjutan.

- `SystemReconciliationDeletes`— Jumlah operasi penghapusan yang diperlukan untuk menghapus data kuburan.

Untuk informasi selengkapnya tentang cara memantau CloudWatch metrik, lihat [the section called “Pemantauan CloudWatch dengan”](#).

Saat menggunakan CloudFormation, Anda dapat mengaktifkan stempel waktu sisi klien saat membuat tabel Amazon Keyspaces. Untuk informasi selengkapnya, silakan lihat [Panduan Pengguna AWS CloudFormation](#).

## Buat tabel baru dengan stempel waktu sisi klien di Amazon Keyspaces

Ikuti contoh berikut untuk membuat tabel Amazon Keyspaces baru dengan stempel waktu sisi klien diaktifkan menggunakan Amazon Konsol Manajemen AWS Keyspaces, Cassandra Query Language (CQL), atau AWS Command Line Interface

### Console

Buat tabel baru dengan stempel waktu sisi klien (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih Buat tabel.
3. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel baru.
4. Di bagian Skema, buat skema untuk tabel Anda.
5. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
6. Lanjutkan ke Client-side stempel waktu.

Pilih Aktifkan stempel waktu sisi klien untuk mengaktifkan stempel waktu sisi klien untuk tabel.



7. Pilih Buat tabel. Tabel Anda dibuat dengan stempel waktu sisi klien dihidupkan.

## Cassandra Query Language (CQL)

### Buat tabel baru menggunakan CQL

1. Untuk membuat tabel baru dengan stempel waktu sisi klien diaktifkan menggunakan CQL, Anda dapat menggunakan contoh berikut.

```
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. Untuk mengonfirmasi pengaturan stempel waktu sisi klien untuk tabel baru, gunakan SELECT pernyataan untuk meninjau custom\_properties seperti yang ditunjukkan pada contoh berikut.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
'my_keyspace' and table_name = 'my_table';
```

Output dari pernyataan ini menunjukkan status untuk stempel waktu sisi klien.

```
'client_side_timestamps': {'status': 'enabled'}
```

## AWS CLI

### Buat tabel baru menggunakan AWS CLI

1. Untuk membuat tabel baru dengan stempel waktu sisi klien diaktifkan, Anda dapat menggunakan contoh berikut.

```
./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. Untuk mengonfirmasi bahwa stempel waktu sisi klien diaktifkan untuk tabel baru, jalankan kode berikut.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

Outputnya akan terlihat mirip dengan contoh ini.

```
{
 "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": ""
 }
}
```

## Konfigurasi stempel waktu sisi klien untuk tabel di Amazon Keyspaces

Ikuti contoh ini untuk mengaktifkan stempel waktu sisi klien untuk tabel yang ada menggunakan Amazon Keyspaces, Cassandra Query Language (CQL) Konsol Manajemen AWS, atau. AWS Command Line Interface

### Console

Untuk mengaktifkan stempel waktu sisi klien untuk tabel yang ada (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pilih tabel yang ingin Anda perbarui, lalu pilih tab Pengaturan tambahan.
3. Pada tab Pengaturan tambahan, buka Modify client-side timestamps dan pilih Turn on client-side timestamps
4. Pilih Simpan perubahan untuk mengubah pengaturan tabel.

## Cassandra Query Language (CQL)

### Menggunakan pernyataan CQL

1. Aktifkan stempel waktu sisi klien untuk tabel yang ada dengan pernyataan CQL. ALTER TABLE

```
ALTER TABLE my_table WITH custom_properties = {'client_side_timestamps':
{'status': 'enabled'}};;
```

2. Untuk mengonfirmasi pengaturan stempel waktu sisi klien untuk tabel baru, gunakan SELECT pernyataan untuk meninjau custom\_properties seperti yang ditunjukkan pada contoh berikut.

```
SELECT custom_properties from system_schema_mcs.tables where keyspace_name =
'my_keyspace' and table_name = 'my_table';
```

Output dari pernyataan ini menunjukkan status untuk stempel waktu sisi klien.

```
'client_side_timestamps': {'status': 'enabled'}
```

## AWS CLI

### Menggunakan AWS CLI

1. Anda dapat mengaktifkan stempel waktu sisi klien untuk tabel yang ada menggunakan contoh berikut AWS CLI .

```
./aws keyspaces update-table \
--keyspace-name my_keyspace \
--table-name my_table \
--client-side-timestamps 'status=ENABLED'
```

2. Untuk mengonfirmasi bahwa stempel waktu sisi klien diaktifkan untuk tabel, jalankan kode berikut.

```
./aws keyspaces get-table \
--keyspace-name my_keyspace \
--table-name my_table
```

Outputnya akan terlihat mirip dengan contoh ini dan menyatakan status stempel waktu sisi klien sebagai. 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": ""
 }
 }
}
```

## Gunakan stempel waktu sisi klien dalam kueri di Amazon Keyspaces

Setelah Anda mengaktifkan stempel waktu sisi klien, Anda dapat meneruskan stempel waktu diUPDATE, dan pernyataan dengan INSERT klausa. DELETE USING TIMESTAMP

Nilai stempel waktu `bigint` mewakili sejumlah mikrodetik sejak waktu dasar standar yang dikenal sebagaiePOCH: 1 Januari 1970 pada 00:00:00 GMT. Stempel waktu yang disediakan oleh klien harus berada di antara kisaran 2 hari yang lalu dan 5 menit di masa depan dari waktu jam dinding saat ini.

Amazon Keyspaces menyimpan metadata stempel waktu untuk masa pakai data. Anda dapat menggunakan `WRITETIME` fungsi ini untuk mencari stempel waktu yang terjadi bertahun-tahun yang lalu. Untuk informasi selengkapnya tentang sintaks CQL, lihat [the section called “Pernyataan DXML”](#)

Pernyataan CQL berikut adalah contoh bagaimana menggunakan stempel waktu sebagai `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;
```

Jika Anda tidak menentukan stempel waktu dalam kueri CQL Anda, Amazon Keyspaces menggunakan stempel waktu yang diteruskan oleh driver klien Anda. Jika tidak ada stempel waktu yang diberikan oleh driver klien, Amazon Keyspaces menetapkan stempel waktu sisi server untuk operasi penulisan Anda.

Untuk melihat nilai stempel waktu yang disimpan untuk kolom tertentu, Anda dapat menggunakan `WRITETIME` fungsi dalam `SELECT` pernyataan seperti yang ditunjukkan pada contoh berikut.

```
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 replikasi untuk Amazon Keyspaces (untuk Apache Cassandra)

Anda dapat menggunakan replikasi Multi-wilayah Amazon Keyspaces untuk mereplikasi data Anda dengan replikasi aktif-aktif otomatis, terkelola sepenuhnya, dan di seluruh pilihan Anda. Wilayah AWS Dengan replikasi aktif-aktif, setiap Wilayah dapat melakukan membaca dan menulis secara terpisah. Anda dapat meningkatkan ketersediaan dan ketahanan dari degradasi Regional, sementara juga mendapatkan manfaat dari pembacaan dan penulisan lokal latensi rendah untuk aplikasi global.

Dengan replikasi Multi-wilayah, Amazon Keyspaces secara asinkron mereplikasi data antar Wilayah, dan data biasanya disebarkan di seluruh Wilayah dalam satu detik. Selain itu, dengan replikasi Multi-wilayah, Anda tidak lagi memiliki pekerjaan yang sulit untuk menyelesaikan konflik dan memperbaiki masalah divergensi data, sehingga Anda dapat fokus pada aplikasi Anda.

Secara default, Amazon Keyspaces mereplikasi data di tiga [Availability Zone yang sama Wilayah AWS untuk daya tahan dan ketersediaan](#) tinggi. Dengan replikasi Multi-wilayah, Anda dapat membuat ruang kunci Multi-wilayah yang mereplikasi tabel Anda dalam geografis yang berbeda pilihan Anda. Wilayah AWS

### Topik

- [Manfaat menggunakan replikasi Multi-region](#)
- [Mode kapasitas dan harga](#)
- [Cara kerja replikasi Multi-wilayah di Amazon Keyspaces](#)
- [Amazon Keyspaces Catatan penggunaan replikasi multi-wilayah](#)
- [Konfigurasi replikasi Multi-wilayah untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#)

## Manfaat menggunakan replikasi Multi-region

Multi-Region replikasi memberikan manfaat sebagai berikut.

- Global membaca dan menulis dengan latensi milidetik satu digit — Di Amazon Keyspaces, replikasi aktif aktif. Anda dapat menyajikan bacaan dan penulisan secara lokal dari Wilayah terdekat dengan pelanggan Anda dengan latensi milidetik satu digit pada skala apa pun. Anda dapat menggunakan Amazon Keyspaces Tabel Multi-region untuk aplikasi global yang membutuhkan waktu respons cepat di mana saja di dunia.
- Peningkatan kelangsungan bisnis dan perlindungan dari degradasi Single-region — Dengan replikasi Multi-region, Anda dapat memulihkan diri dari degradasi dalam satu Wilayah AWS dengan mengarahkan aplikasi Anda ke Wilayah lain di ruang kunci Multi-wilayah Anda. Karena Amazon Keyspaces menawarkan replikasi aktif-aktif, tidak ada dampak pada pembacaan dan penulisan Anda.

Amazon Keyspaces melacak penulisan apa pun yang telah dilakukan di ruang kunci Multi-wilayah Anda tetapi belum disebarkan ke semua Wilayah replika. Setelah Region kembali online, Amazon Keyspaces secara otomatis menyinkronkan perubahan yang hilang sehingga Anda dapat pulih tanpa dampak aplikasi apa pun.

- High-speed replikasi lintas Wilayah — Multi-Region replikasi menggunakan replikasi fisik data berbasis penyimpanan yang cepat di seluruh Wilayah, dengan jeda replikasi yang biasanya kurang dari 1 detik.

Replikasi di Amazon Keyspaces memiliki sedikit atau tidak ada dampak pada kueri database Anda karena tidak berbagi sumber daya komputasi dengan aplikasi Anda. Ini berarti Anda dapat mengatasi kasus penggunaan throughput penulisan tinggi atau kasus penggunaan dengan lonjakan atau semburan throughput tiba-tiba tanpa dampak aplikasi apa pun.

- Konsistensi dan resolusi konflik — Setiap perubahan yang dilakukan pada data di Wilayah mana pun direplikasi ke Wilayah lain di ruang kunci Multi-wilayah. Jika aplikasi memperbarui data yang sama di Wilayah yang berbeda secara bersamaan, konflik dapat muncul.

Untuk membantu memberikan konsistensi pada akhirnya, Amazon Keyspaces menggunakan stempel waktu tingkat sel dan penulis terakhir memenangkan rekonsiliasi antara pembaruan bersamaan. Resolusi konflik dikelola sepenuhnya dan terjadi di latar belakang tanpa dampak aplikasi apa pun.



Untuk informasi selengkapnya tentang konfigurasi dan fitur yang didukung, lihat [the section called “Catatan penggunaan”](#).

## Mode kapasitas dan harga

Untuk ruang kunci Multi-wilayah, Anda dapat menggunakan mode kapasitas sesuai permintaan atau mode kapasitas yang disediakan. Untuk informasi selengkapnya, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).

Untuk mode sesuai permintaan, Anda ditagih 1 unit permintaan tulis (WRU) untuk menulis hingga 1 KB data per baris dengan cara yang sama seperti untuk tabel Wilayah tunggal. Tetapi Anda ditagih untuk menulis di setiap Wilayah ruang kunci Multi-wilayah Anda. Misalnya, menulis baris 3 KB data di ruang kunci Multi-wilayah dengan dua Wilayah membutuhkan 6 WRU:  $3 * 2 = 6$  WRU. Selain itu, penulisan yang mencakup data statis dan non-statis memerlukan operasi penulisan tambahan.

Untuk mode yang disediakan, Anda ditagih 1 unit kapasitas tulis (WCU) untuk menulis hingga 1 KB data per baris, dengan cara yang sama seperti untuk tabel Wilayah tunggal. Tetapi Anda ditagih untuk menulis di setiap Wilayah ruang kunci Multi-wilayah Anda. Misalnya, menulis baris 3 KB data per detik di ruang kunci Multi-wilayah dengan dua Wilayah membutuhkan 6 WCU:  $3 * 2 = 6$  WCU. Selain itu, penulisan yang mencakup data statis dan non-statis memerlukan operasi penulisan tambahan.

Untuk informasi selengkapnya tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

## Cara kerja replikasi Multi-wilayah di Amazon Keyspaces

Bagian ini memberikan ikhtisar tentang cara kerja replikasi Multi-wilayah Amazon Keyspaces. Untuk informasi selengkapnya tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

### Topik

- [Cara kerja replikasi Multi-wilayah di Amazon Keyspaces](#)
- [Multi-Region resolusi konflik replikasi](#)
- [Multi-Region replikasi pemulihan bencana](#)
- [Multi-Region replikasi Wilayah AWS dinonaktifkan secara default](#)
- [Multi-Region replikasi dan integrasi dengan pemulihan point-in-time \(PITR\)](#)
- [Multi-Region replikasi dan integrasi dengan layanan AWS](#)

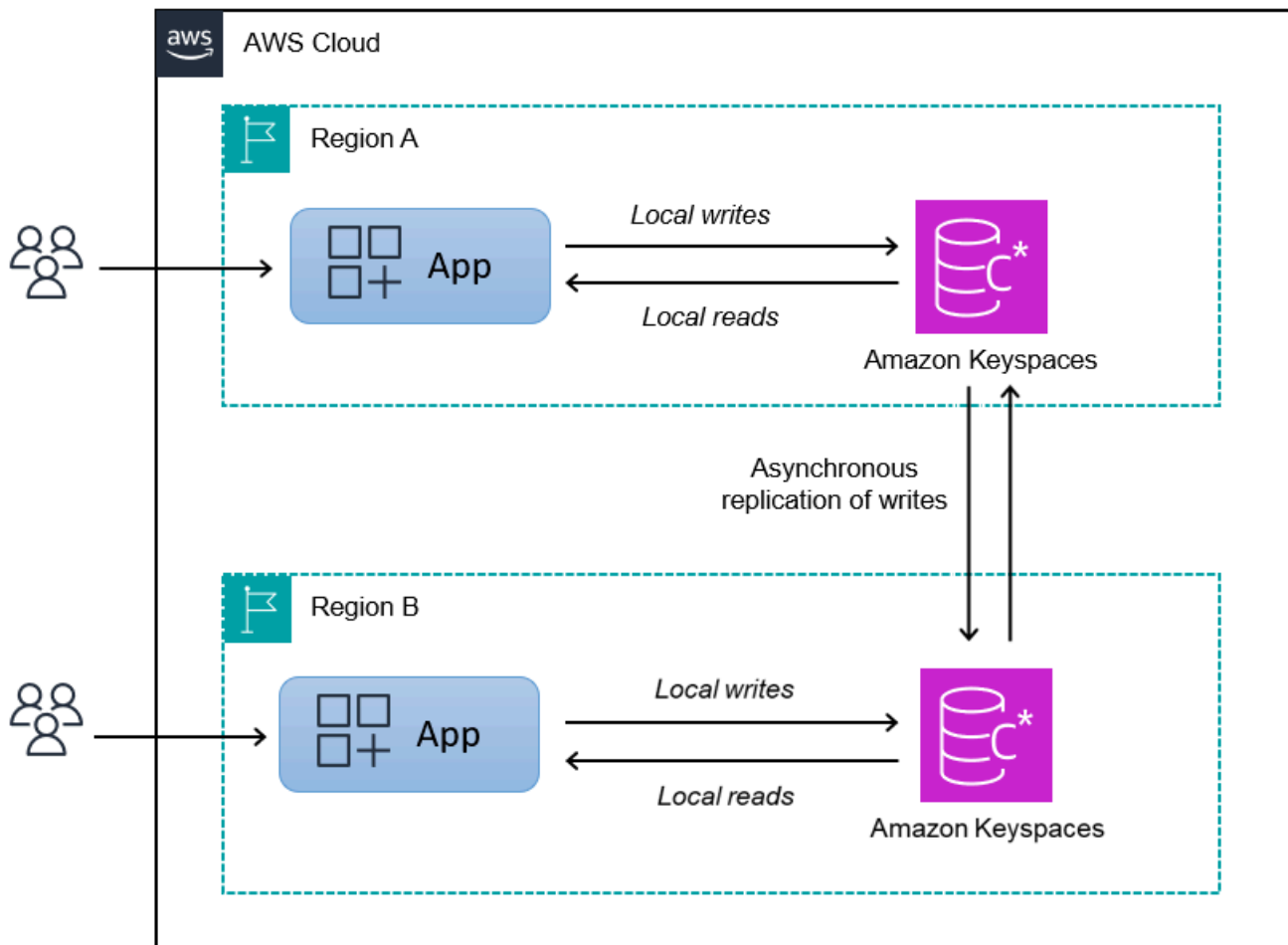
## Cara kerja replikasi Multi-wilayah di Amazon Keyspaces

Amazon Keyspaces Replikasi multi-wilayah mengimplementasikan arsitektur ketahanan data yang mendistribusikan data Anda secara independen dan terdistribusi secara geografis. Wilayah AWS Ini menggunakan replikasi aktif-aktif, yang memberikan latensi rendah lokal dengan setiap Wilayah dapat melakukan pembacaan dan penulisan secara terpisah.

Saat membuat ruang kunci Multi-wilayah Amazon Keyspaces, Anda dapat memilih Wilayah tambahan tempat data akan direplikasi. Setiap tabel yang Anda buat di ruang kunci Multi-wilayah terdiri dari beberapa tabel replika (satu per Wilayah) yang dianggap Amazon Keyspaces sebagai satu unit.

Setiap replika memiliki nama tabel yang sama dan skema kunci primer yang sama. Ketika aplikasi menulis data ke tabel lokal di satu Wilayah, data ditulis dengan tahan lama menggunakan tingkat LOCAL\_QUORUM konsistensi. Amazon Keyspaces secara otomatis mereplikasi data secara asinkron ke Wilayah replikasi lainnya. Kelambatan replikasi di seluruh Wilayah biasanya kurang dari satu detik dan tidak memengaruhi kinerja atau throughput aplikasi Anda.

Setelah data ditulis, Anda dapat membacanya dari tabel Multi-region di Region replikasi lain dengan tingkat LOCAL\_ONE/LOCAL\_QUORUM konsistensi. Untuk informasi selengkapnya tentang konfigurasi dan fitur yang didukung, lihat [the section called “Catatan penggunaan”](#).



## Multi-Region resolusi konflik replikasi

Amazon Keyspaces Replikasi multi-wilayah dikelola sepenuhnya, yang berarti Anda tidak perlu melakukan tugas replikasi seperti menjalankan operasi perbaikan secara teratur untuk membersihkan masalah sinkronisasi data. Amazon Keyspaces memantau konsistensi data antar tabel yang berbeda Wilayah AWS dengan mendeteksi dan memperbaiki konflik, dan menyinkronkan replika secara otomatis.

Amazon Keyspaces menggunakan metode last writer wins dari rekonsiliasi data. Dengan mekanisme resolusi konflik ini, semua Wilayah di ruang kunci Multi-wilayah menyetujui pembaruan terbaru dan menyatu menuju keadaan di mana mereka semua memiliki data yang identik. Proses rekonsiliasi tidak berdampak pada kinerja aplikasi. Untuk mendukung penyelesaian konflik, stempel waktu sisi klien diaktifkan secara otomatis untuk tabel Multi-wilayah dan tidak dapat dimatikan. Untuk informasi selengkapnya, lihat [the section called “Client-side stempel waktu”](#).

## Multi-Region replikasi pemulihan bencana

Dengan replikasi Multi-wilayah Amazon Keyspaces, penulisan direplikasi secara asinkron di setiap Wilayah. Jika terjadi degradasi atau kegagalan Wilayah yang jarang terjadi, replikasi Multi-wilayah membantu Anda pulih dari bencana dengan sedikit atau tanpa dampak pada aplikasi Anda. Pemulihan dari bencana biasanya diukur menggunakan nilai untuk tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO).

Tujuan waktu pemulihan — Waktu yang dibutuhkan sistem untuk kembali ke keadaan kerja setelah bencana. RTO mengukur jumlah waktu henti yang dapat ditoleransi oleh beban kerja Anda, diukur dalam waktu. Untuk rencana pemulihan bencana yang menggunakan replikasi Multi-wilayah untuk gagal ke Wilayah yang tidak terpengaruh, RTO bisa hampir nol. RTO dibatasi oleh seberapa cepat aplikasi Anda dapat mendeteksi kondisi kegagalan dan mengarahkan lalu lintas ke Wilayah lain.

Tujuan titik pemulihan — Jumlah data yang dapat hilang (diukur dalam waktu). Untuk rencana pemulihan bencana yang menggunakan replikasi Multi-wilayah untuk gagal ke Wilayah yang tidak terpengaruh, RPO biasanya satu digit detik. RPO dibatasi oleh latensi replikasi ke replika target failover.

Jika terjadi kegagalan atau degradasi Regional, Anda tidak perlu mempromosikan Wilayah sekunder atau melakukan prosedur failover database karena replikasi di Amazon Keyspaces aktif. Sebagai gantinya, Anda dapat menggunakan Amazon Route 53 untuk merutekan aplikasi Anda ke Wilayah sehat terdekat. Untuk mempelajari lebih lanjut tentang Route 53, lihat [Apa itu Amazon Route 53?](#).

Jika satu Wilayah AWS menjadi terisolasi atau terdegradasi, aplikasi Anda dapat mengarahkan lalu lintas ke Wilayah lain menggunakan Route 53 untuk melakukan pembacaan dan penulisan terhadap tabel replika yang berbeda. Anda juga dapat menerapkan logika bisnis khusus untuk menentukan kapan harus mengarahkan permintaan ke Wilayah lain. Contohnya adalah membuat aplikasi Anda sadar akan beberapa titik akhir yang tersedia.

Ketika Region kembali online, Amazon Keyspaces melanjutkan menyebarkan penulisan yang tertunda dari Wilayah tersebut ke tabel replika di Wilayah lain. DynamoDB juga melanjutkan penyebaran penulisan dari tabel replika lain ke Wilayah yang saat ini kembali online.

## Multi-Region replikasi Wilayah AWS dinonaktifkan secara default

Amazon Keyspaces Replikasi multi-wilayah didukung dalam hal berikut Wilayah AWS yang dinonaktifkan secara default:

- Wilayah Afrika (Cape Town)

- Wilayah Timur Tengah (UEA)
- Wilayah Asia Pasifik (Hong Kong)
- Wilayah Timur Tengah (Bahrain)

Sebelum Anda dapat menggunakan Wilayah yang dinonaktifkan secara default dengan replikasi Multi-region Amazon Keyspaces, Anda harus mengaktifkan Region terlebih dahulu. Untuk informasi selengkapnya, lihat [Mengaktifkan atau menonaktifkan Wilayah AWS di akun Anda](#) di [Panduan AWS Organizations Pengguna](#).

Setelah mengaktifkan Region, Anda dapat membuat resource Amazon Keyspaces baru di Region dan menambahkan Region ke ruang kunci Multi-region.

Saat Anda menonaktifkan Wilayah yang digunakan oleh replikasi Multi-wilayah Amazon Keyspaces, Amazon Keyspaces memulai masa tenggang 24 jam. Selama jendela waktu ini, Anda dapat mengharapkan perilaku berikut:

- Amazon Keyspaces terus melakukan operasi data manipulation language (DML/bahasa manipulasi data) di Wilayah yang diaktifkan.
- Amazon Keyspaces menghentikan sementara mereplikasi pembaruan data dari Wilayah yang diaktifkan ke Wilayah yang dinonaktifkan.
- Amazon Keyspaces memblokir semua permintaan bahasa definisi data (DDL) di Wilayah yang dinonaktifkan.

Jika Anda menonaktifkan Wilayah karena kesalahan, Anda dapat mengaktifkan kembali Wilayah dalam waktu 24 jam. Jika Anda mengaktifkan kembali Wilayah selama masa tenggang 24 jam, Amazon Keyspaces akan mengambil tindakan berikut:

- Secara otomatis melanjutkan semua replikasi ke Wilayah yang diaktifkan kembali.
- Replikasi setiap pembaruan data yang terjadi di Wilayah yang diaktifkan saat Wilayah dinonaktifkan untuk memastikan konsistensi data.
- Lanjutkan semua operasi replikasi Multi-wilayah tambahan secara otomatis.

Jika Region tetap dinonaktifkan setelah jendela 24 jam ditutup, Amazon Keyspaces mengambil tindakan berikut untuk menghapus Region secara permanen dari replikasi Multi-region:

- Hapus Wilayah yang dinonaktifkan dari semua ruang kunci replikasi Multi-wilayah.

- Ubah replika tabel replikasi multi-wilayah di Wilayah yang dinonaktifkan menjadi ruang kunci dan tabel wilayah tunggal.
- Amazon Keyspaces tidak menghapus sumber daya apa pun dari Wilayah yang dinonaktifkan.

Setelah Amazon Keyspaces menghapus Wilayah yang dinonaktifkan secara permanen dari ruang kunci Multi-wilayah, Anda tidak dapat menambahkan kembali Wilayah yang dinonaktifkan.

## Multi-Region replikasi dan integrasi dengan pemulihan point-in-time (PITR)

Point-in-time pemulihan didukung untuk tabel Multi-wilayah. Agar berhasil mengembalikan tabel Multi-wilayah dengan PITR, kondisi berikut harus dipenuhi.

- Sumber dan tabel target harus dikonfigurasi sebagai tabel Multi-wilayah.
- Regions replikasi untuk keyspace dari tabel sumber dan untuk keyspace dari tabel target harus sama.
- PITR harus diaktifkan pada semua replika tabel sumber.

Anda dapat menjalankan pernyataan pemulihan dari salah satu Wilayah tempat tabel sumber tersedia. Amazon Keyspaces secara otomatis mengembalikan tabel target di setiap Wilayah. Untuk informasi lebih lanjut tentang PITR, lihat [the section called “Cara kerjanya”](#).

Saat Anda membuat tabel Multi-wilayah, pengaturan PITR yang Anda tentukan selama proses pembuatan secara otomatis diterapkan ke semua tabel di semua Wilayah. Saat Anda mengubah pengaturan PITR menggunakan `ALTER TABLE`, Amazon Keyspaces menerapkan pembaruan hanya ke tabel lokal dan bukan ke replika di Wilayah lain. Untuk mengaktifkan PITR untuk tabel Multi-region yang ada, Anda harus mengulangi `ALTER TABLE` pernyataan untuk semua replika.

## Multi-Region replikasi dan integrasi dengan layanan AWS

Anda dapat memantau kinerja replikasi antar tabel dalam berbagai tabel Wilayah AWS dengan menggunakan CloudWatch metrik Amazon. Metrik berikut menyediakan pemantauan terus menerus terhadap ruang kunci Multi-wilayah.

- **ReplicationLatency**— Metrik ini mengukur waktu yang diperlukan untuk mereplikasi `updates`, `inserts`, atau `deletes` dari satu tabel replika ke tabel replika lain di ruang kunci Multi-wilayah.

Untuk informasi selengkapnya tentang cara memantau CloudWatch metrik, lihat [the section called “Pemantauan CloudWatch dengan”](#).

## Amazon Keyspaces Catatan penggunaan replikasi multi-wilayah

Pertimbangkan hal berikut saat Anda menggunakan replikasi Multi-wilayah dengan Amazon Keyspaces.

- Anda dapat memilih salah satu [publik yang tersedia](#) Wilayah AWS. Untuk informasi selengkapnya tentang hal Wilayah AWS [itu dinonaktifkan secara default](#), lihat [the section called “Wilayah dinonaktifkan secara default”](#).
- AWS GovCloud (US) Regions dan Wilayah Tiongkok tidak didukung.
- Pertimbangkan solusi berikut hingga fitur tersedia:

Konfigurasi Time to Live (TTL) saat membuat tabel Multi-region. Anda tidak akan dapat mengaktifkan dan menonaktifkan TTL, atau menyesuaikan nilai TTL nanti. Untuk informasi selengkapnya, lihat [the section called “Data kedaluwarsa dengan Time to Live”](#).

- Untuk enkripsi saat istirahat, gunakan kunci yang AWS dimiliki. Kunci terkelola pelanggan saat ini tidak didukung untuk tabel Multi-wilayah. Untuk informasi selengkapnya, silakan lihat [the section called “Cara kerjanya”](#).
- Anda dapat menggunakan ALTER KEYSPACE untuk menambahkan Wilayah ke wilayah Tunggal atau ruang kunci Multi-wilayah. Untuk informasi selengkapnya, lihat [the section called “Menambahkan Region ke keyspace”](#).
  - Sebelum menambahkan Region ke ruang kunci Single-region, pastikan tidak ada tabel di bawah ruang kunci yang dikonfigurasi dengan kunci terkelola pelanggan.
  - Setiap tag yang ada yang dikonfigurasi untuk ruang kunci atau tabel tidak direplikasi ke Wilayah baru.
- Saat Anda menggunakan manajemen kapasitas yang disediakan dengan penskalaan otomatis Amazon Keyspaces, pastikan untuk menggunakan operasi Amazon Keyspaces API untuk membuat dan mengonfigurasi tabel Multi-wilayah Anda. Operasi Application Auto Scaling API yang mendasari yang dipanggil Amazon Keyspaces atas nama Anda tidak memiliki kemampuan Multi-region.

Untuk informasi selengkapnya, lihat [the section called “Memperbarui pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”](#). Untuk informasi selengkapnya

tentang cara memperkirakan throughput kapasitas tulis dari tabel Multi-wilayah yang disediakan, lihat [the section called “Perkiraan kapasitas untuk tabel Multi-wilayah”](#)

- Meskipun data secara otomatis direplikasi di seluruh Wilayah yang dipilih dari tabel Multi-wilayah, ketika klien terhubung ke titik akhir di satu Wilayah dan menanyakan `system.peers` tabel, kueri hanya mengembalikan informasi lokal. Hasil kueri muncul seperti kluster pusat data tunggal ke klien.
- Amazon Keyspaces Replikasi multi-wilayah bersifat asinkron, dan mendukung konsistensi untuk penulisan. `LOCAL_QUORUM LOCAL_QUORUM` konsistensi mengharuskan pembaruan ke baris bertahan lama pada dua replika di Wilayah lokal sebelum mengembalikan kesuksesan ke klien. Perbanyak penulisan ke Wilayah (atau Wilayah) yang direplikasi kemudian dilakukan secara asinkron.

Amazon Keyspaces Replikasi multi-wilayah tidak mendukung replikasi atau konsistensi sinkron. `QUORUM`

- Saat Anda membuat ruang kunci atau tabel Multi-region, tag apa pun yang Anda tentukan selama proses pembuatan secara otomatis diterapkan ke semua ruang kunci dan tabel di semua Wilayah. Saat Anda mengubah tag yang ada menggunakan `ALTER KEYSPACE` or `ALTER TABLE`, pembaruan hanya diterapkan ke ruang kunci atau tabel di Wilayah tempat Anda melakukan perubahan.
- Amazon CloudWatch menyediakan `ReplicationLatency` metrik untuk setiap Wilayah yang direplikasi. Ini menghitung metrik ini dengan melacak baris yang tiba, membandingkan waktu kedatangan mereka dengan waktu tulis awal mereka, dan menghitung rata-rata. Pengaturan waktu disimpan CloudWatch di dalam Wilayah sumber. Untuk informasi selengkapnya, lihat [the section called “Pemantauan CloudWatch dengan”](#).

Ini dapat berguna untuk melihat pengaturan waktu rata-rata dan maksimum untuk menentukan kelambatan replikasi rata-rata dan terburuk. Tidak ada SLA pada latensi ini.

- Saat menggunakan tabel Multi-wilayah dalam mode sesuai permintaan, Anda dapat mengamati peningkatan latensi untuk replikasi penulisan asinkron jika replika tabel mengalami puncak lalu lintas baru. Mirip dengan cara Amazon Keyspaces secara otomatis menyesuaikan kapasitas tabel on-demand Single-region dengan lalu lintas aplikasi yang diterimanya, Amazon Keyspaces secara otomatis menyesuaikan kapasitas replika tabel on-demand Multi-region dengan lalu lintas yang diterimanya. Peningkatan latensi replikasi bersifat sementara karena Amazon Keyspaces secara otomatis mengalokasikan lebih banyak kapasitas saat volume lalu lintas Anda meningkat. Setelah semua replika disesuaikan dengan volume lalu lintas Anda, latensi replikasi akan kembali normal. Untuk informasi selengkapnya, lihat [the section called “Properti lalu lintas puncak dan penskalaan”](#).



- Saat menggunakan tabel Multi-wilayah dalam mode yang disediakan, jika aplikasi melebihi kapasitas throughput yang disediakan, Anda mungkin melihat kesalahan kapasitas yang tidak mencukupi dan peningkatan latensi replikasi. Untuk memastikan bahwa selalu ada kapasitas baca dan tulis yang cukup untuk semua replika tabel di semua Wilayah AWS tabel Multi-wilayah, sebaiknya Anda mengonfigurasi penskalaan otomatis Amazon Keyspaces. Penskalaan otomatis Amazon Keyspaces membantu Anda menyediakan kapasitas throughput secara efisien untuk beban kerja variabel dengan menyesuaikan kapasitas throughput secara otomatis sebagai respons terhadap lalu lintas aplikasi yang sebenarnya. Lihat informasi yang lebih lengkap di [the section called “Cara kerja penskalaan otomatis untuk tabel Multi-wilayah”](#).

## Konfigurasikan replikasi Multi-wilayah untuk Amazon Keyspaces (untuk Apache Cassandra)

Anda dapat menggunakan konsol, Cassandra Query Language (CQL), atau AWS Command Line Interface untuk membuat dan mengelola ruang kunci dan tabel Multi-region di Amazon Keyspaces.

Bagian ini memberikan contoh cara membuat dan mengelola ruang kunci dan tabel Multi-region. Semua tabel yang Anda buat di ruang kunci Multi-region secara otomatis mewarisi pengaturan Multi-region dari keyspace.

Untuk informasi selengkapnya tentang konfigurasi dan fitur yang didukung, lihat [the section called “Catatan penggunaan”](#).

### Topik

- [Konfigurasikan izin IAM yang diperlukan untuk membuat ruang kunci dan tabel Multi-wilayah](#)
- [Konfigurasikan izin IAM yang diperlukan untuk menambahkan ke ruang Wilayah AWS kunci](#)
- [Buat ruang kunci Multi-wilayah di Amazon Keyspaces](#)
- [Tambahkan Wilayah AWS ke ruang kunci di Amazon Keyspaces](#)
- [Periksa kemajuan replikasi saat menambahkan Wilayah baru ke ruang kunci](#)
- [Buat tabel Multi-wilayah dengan pengaturan default di Amazon Keyspaces](#)
- [Buat tabel Multi-wilayah dalam mode yang disediakan dengan penskalaan otomatis di Amazon Keyspaces](#)
- [Memperbarui kapasitas yang disediakan dan pengaturan penskalaan otomatis untuk tabel Multi-wilayah di Amazon Keyspaces](#)

- [Melihat pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah di Amazon Keyspaces](#)
- [Matikan penskalaan otomatis untuk tabel di Amazon Keyspaces](#)
- [Mengatur kapasitas yang disediakan dari tabel Multi-wilayah secara manual di Amazon Keyspaces](#)

## Konfigurasi izin IAM yang diperlukan untuk membuat ruang kunci dan tabel Multi-wilayah

Agar berhasil membuat ruang kunci dan tabel Multi-region, prinsipal IAM harus dapat membuat peran terkait layanan. Peran terkait layanan ini adalah jenis unik peran IAM yang telah ditentukan sebelumnya oleh Amazon Keyspaces. Ini mencakup semua izin yang diperlukan Amazon Keyspaces untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya tentang peran terkait layanan, lihat [the section called “Multi-Region Replikasi”](#).

Untuk membuat peran terkait layanan yang diperlukan oleh replikasi Multi-wilayah, kebijakan untuk prinsipal IAM memerlukan elemen berikut:

- `iam:CreateServiceLinkedRole`— Tindakan yang dapat dilakukan kepala sekolah.
- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`— Sumber daya tempat tindakan dapat dilakukan.
- `iam:AWSServiceName`: `"replication.cassandra.amazonaws.com"`— Satu-satunya AWS layanan yang dapat dilampirkan peran ini adalah Amazon Keyspaces.

Berikut ini adalah contoh kebijakan yang memberikan izin minimum yang diperlukan kepada prinsipal untuk membuat ruang kunci dan tabel Multi-region.

```
{
 "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"}}
}
```

Untuk izin IAM tambahan untuk ruang kunci dan tabel Multi-region, lihat [tombol Tindakan, sumber daya, dan kondisi untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#) di Referensi Otorisasi Layanan.

Konfigurasi izin IAM yang diperlukan untuk menambahkan ke ruang Wilayah AWS kunci

Untuk menambahkan Region ke ruang kunci, prinsipal IAM memerlukan izin berikut:

- `cassandra:Alter`
- `cassandra:AlterMultiRegionResource`
- `cassandra:Create`
- `cassandra:CreateMultiRegionResource`
- `cassandra:Select`
- `cassandra:SelectMultiRegionResource`
- `cassandra:Modify`
- `cassandra:ModifyMultiRegionResource`

Jika tabel dikonfigurasi dalam mode yang disediakan dengan penskalaan otomatis diaktifkan, izin tambahan berikut diperlukan.

- `application-autoscaling:RegisterScalableTarget`
- `application-autoscaling:DeregisterScalableTarget`
- `application-autoscaling:DescribeScalableTargets`
- `application-autoscaling:PutScalingPolicy`
- `application-autoscaling:DescribeScalingPolicies`

Agar berhasil menambahkan Region ke ruang kunci Single-region, prinsipal IAM juga harus dapat membuat peran terkait layanan. Peran terkait layanan ini adalah jenis unik peran IAM yang telah ditentukan sebelumnya oleh Amazon Keyspaces. Ini mencakup semua izin yang diperlukan Amazon Keyspaces untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya tentang peran terkait layanan, lihat [the section called “Multi-Region Replikasi”](#).

Untuk membuat peran terkait layanan yang diperlukan oleh replikasi Multi-wilayah, kebijakan untuk prinsipal IAM memerlukan elemen berikut:

- `iam:CreateServiceLinkedRole`— Tindakan yang dapat dilakukan kepala sekolah.
- `arn:aws:iam::*:role/aws-service-role/replication.cassandra.amazonaws.com/AWSServiceRoleForKeyspacesReplication`— Sumber daya tempat tindakan dapat dilakukan.
- `iam:AWSServiceName": "replication.cassandra.amazonaws.com`— Satu-satunya AWS layanan yang dapat dilampirkan peran ini adalah Amazon Keyspaces.

Berikut ini adalah contoh kebijakan yang memberikan izin minimum yang diperlukan kepada prinsipal untuk menambahkan Wilayah ke ruang kunci.

```
{
 "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"}}
}
```

## Buat ruang kunci Multi-wilayah di Amazon Keyspaces

Bagian ini memberikan contoh cara membuat ruang kunci Multi-wilayah. Anda dapat melakukan ini di konsol Amazon Keyspaces, menggunakan CQL atau AWS CLI Semua tabel yang Anda buat di ruang kunci Multi-region secara otomatis mewarisi pengaturan Multi-region dari keyspace.

### Note

Saat membuat ruang kunci Multi-wilayah, Amazon Keyspaces membuat peran terkait layanan dengan nama di akun Anda. `AWSServiceRoleForAmazonKeyspacesReplication` Peran ini memungkinkan Amazon Keyspaces untuk mereplikasi penulisan ke semua replika tabel Multi-wilayah atas nama Anda. Untuk mempelajari selengkapnya, lihat [the section called “Multi-Region Replikasi”](#).

## Console

### Buat ruang kunci Multi-wilayah (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces, lalu pilih Create keyspace.
3. Untuk nama Keyspace, masukkan nama untuk keyspace.
4. Di bagian Multi-Region replikasi, Anda dapat menambahkan Wilayah tambahan yang tersedia dalam daftar.
5. Untuk menyelesaikannya, pilih Create keyspace.

## Cassandra Query Language (CQL)

### Buat ruang kunci Multi-wilayah menggunakan CQL

1. Untuk membuat ruang kunci Multi-wilayah, gunakan `NetworkTopologyStrategy` untuk menentukan ruang kunci Wilayah AWS yang akan direplikasi. Anda harus menyertakan Wilayah Anda saat ini dan setidaknya satu Wilayah tambahan.

Semua tabel di keyspace mewarisi strategi replikasi dari keyspace. Anda tidak dapat mengubah strategi replikasi di tingkat tabel.

`NetworkTopologyStrategy`— Faktor replikasi untuk setiap Wilayah adalah tiga karena Amazon Keyspaces mereplikasi data di [tiga Availability Zone](#) dalam Wilayah AWS yang sama, secara default.

Pernyataan CQL berikut adalah contoh dari ini.

```
CREATE KEYSPACE mykeyspace
WITH REPLICATION = {'class':'NetworkTopologyStrategy', 'us-east-1':'3', 'ap-southeast-1':'3', 'eu-west-1':'3' };
```

2. Anda dapat menggunakan pernyataan CQL untuk menanyakan tabel di `system_multiregion_info` ruang kunci untuk secara terprogram mencantumkan Wilayah dan status tabel Multi-wilayah yang Anda tentukan. Kode berikut adalah contoh dari ini.

```
SELECT * from system_multiregion_info.tables WHERE keyspace_name = 'mykeyspace'
AND table_name = 'mytable';
```

Output dari pernyataan terlihat seperti berikut:

| 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

Buat ruang kunci Multi-wilayah baru menggunakan AWS CLI

- Untuk membuat ruang kunci Multi-wilayah, Anda dapat menggunakan pernyataan CLI berikut. Tentukan Wilayah Anda saat ini dan setidaknya satu Wilayah tambahan di `regionList`.

```
aws keyspaces create-keyspace --keyspace-name mykeyspace \
--replication-specification replicationStrategy=MULTI_REGION,regionList=us-
east-1,eu-west-1
```

Untuk membuat tabel Multi-region, lihat [the section called “Buat tabel Multi-region dengan pengaturan default”](#) dan [the section called “Buat tabel Multi-wilayah dalam mode yang disediakan”](#).

## Tambahkan Wilayah AWS ke ruang kunci di Amazon Keyspaces

Anda dapat menambahkan yang baru Wilayah AWS ke ruang kunci yang merupakan ruang kunci tunggal atau Multi-wilayah. Region replika baru diterapkan ke semua tabel di keyspace.

Untuk mengubah wilayah Tunggal menjadi ruang kunci Multi-wilayah, Anda harus mengaktifkan stempel waktu sisi klien untuk semua tabel di ruang kunci. Untuk informasi selengkapnya, lihat [the section called “Client-side stempel waktu”](#).

Jika Anda menambahkan Wilayah tambahan ke ruang kunci Multi-wilayah, Amazon Keyspaces harus mereplikasi tabel yang ada ke Wilayah baru menggunakan pemulihan Lintas wilayah satu kali untuk setiap tabel yang ada. Biaya pemulihan untuk setiap tabel ditagih per GB, untuk informasi selengkapnya lihat [Backup dan restore](#) di halaman harga Amazon Keyspaces (untuk Apache

Cassandra). Tidak ada biaya untuk transfer data di seluruh Wilayah untuk operasi pemulihan ini. Selain data, semua properti tabel dengan pengecualian tag akan direplikasi ke Wilayah baru.

Anda dapat menggunakan ALTER KEYSPACE pernyataan di CQL, update-keyspace perintah dengan AWS CLI, atau konsol untuk menambahkan Wilayah baru ke satu atau ke ruang kunci Multi-wilayah di Amazon Keyspaces. Untuk menjalankan pernyataan dengan sukses, akun yang Anda gunakan harus berada di salah satu Wilayah di mana ruang kunci sudah tersedia. Saat replika ditambahkan, Anda tidak dapat melakukan operasi bahasa definisi data (DDL) lainnya pada sumber daya yang sedang diperbarui dan direplikasi.

Untuk informasi selengkapnya tentang izin yang diperlukan untuk menambahkan Wilayah, lihat [the section called “Konfigurasi izin IAM untuk menambahkan Wilayah”](#).

#### Note

Saat menambahkan Wilayah tambahan ke ruang kunci Wilayah Tunggal, Amazon Keyspaces membuat peran terkait layanan dengan nama di akun Anda. `AWSServiceRoleForAmazonKeyspacesReplication` Peran ini memungkinkan Amazon Keyspaces untuk mereplikasi tabel ke Wilayah baru dan mereplikasi penulisan dari satu tabel ke semua replika tabel Multi-wilayah atas nama Anda. Untuk mempelajari selengkapnya, lihat [the section called “Multi-Region Replikasi”](#).

## Console

Ikuti langkah-langkah berikut untuk menambahkan Wilayah ke ruang kunci menggunakan konsol Amazon Keyspaces.

### Menambahkan Region ke keyspace (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces, lalu pilih keyspace dari daftar.
3. Pilih Wilayah AWStab.
4. Pada Wilayah AWStab, pilih Tambah Wilayah.
5. Dalam dialog Add Region, pilih Region tambahan yang ingin Anda tambahkan ke keyspace.
6. Untuk menyelesaikannya, pilih Tambah.

## Cassandra Query Language (CQL)

### Menambahkan Region ke keyspace menggunakan CQL

- Untuk menambahkan Region baru ke keyspace, Anda dapat menggunakan pernyataan berikut. Dalam contoh ini, ruang kunci sudah tersedia di Wilayah AS Timur (Virginia N.) dan Wilayah Barat (Oregon) AS, dan pernyataan CQL menambahkan Wilayah AS Barat (California N.).

```
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

### Menambahkan Region ke keyspace menggunakan AWS CLI

- Untuk menambahkan Region baru ke keyspace menggunakan CLI, Anda dapat menggunakan contoh berikut. Perhatikan bahwa nilai default untuk `client-side-timestamps` adalah `DISABLED`. Dengan `update-keyspace` perintah, Anda harus mengubah nilainya menjadi `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"}'
```

## Periksa kemajuan replikasi saat menambahkan Wilayah baru ke ruang kunci

Menambahkan Wilayah baru ke ruang kunci Amazon Keyspaces adalah operasi yang berjalan lama. Untuk melacak kemajuan, Anda dapat menggunakan kueri yang ditampilkan di bagian ini.



## Cassandra Query Language (CQL)

### Menggunakan CQL untuk memverifikasi kemajuan add Region

- Untuk memverifikasi kemajuan pembuatan replika tabel baru di ruang kunci tertentu, Anda dapat melakukan kueri `system_multiregion_info.keyspaces`. Pernyataan CQL berikut adalah contoh dari ini.

```
SELECT keyspace_name, region, status, tables_replication_progress
FROM system_multiregion_info.keyspaces
WHERE keyspace_name = 'my_keyspace';
```

Sementara operasi replikasi sedang berlangsung, status menunjukkan kemajuan pembuatan tabel di Wilayah baru. Ini adalah contoh di mana 5 dari 10 tabel telah direplikasi ke Wilayah baru.

| 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%                         |

Setelah proses replikasi berhasil diselesaikan, output akan terlihat seperti contoh ini.

| keyspace_name | region    | status |
|---------------|-----------|--------|
| my_keyspace   | us-east-1 | Active |
| my_keyspace   | us-west-2 | Active |
| my_keyspace   | eu-west-1 | Active |

## CLI

### Menggunakan AWS CLI untuk memverifikasi kemajuan menambahkan Wilayah

- Untuk mengonfirmasi status pembuatan replika tabel untuk ruang kunci tertentu, Anda dapat menggunakan contoh berikut.

```
aws keyspaces get-keyspace \
--keyspace-name my_keyspace
```

Outputnya akan terlihat mirip dengan contoh ini.

```
{
 "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%"
 }
]
}
```

## Buat tabel Multi-wilayah dengan pengaturan default di Amazon Keyspaces

Bagian ini memberikan contoh cara membuat tabel Multi-region dalam mode on-demand dengan semua pengaturan default. Anda dapat melakukan ini di konsol Amazon Keyspaces, menggunakan CQL atau. AWS CLI Semua tabel yang Anda buat di ruang kunci Multi-region secara otomatis mewarisi pengaturan Multi-region dari keyspace.

Untuk membuat ruang kunci Multi-wilayah, lihat. [the section called “Buat ruang kunci Multi-wilayah”](#)

### Console

Buat tabel Multi-region dengan pengaturan default (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pilih ruang kunci Multi-wilayah.

3. Pada tab Tabel, pilih Buat tabel.
4. Untuk nama Tabel, masukkan nama untuk tabel. Tabel ini sedang direplikasi ditampilkan di kotak info. Wilayah AWS
5. Lanjutkan dengan skema tabel.
6. Di bawah Pengaturan tabel, lanjutkan dengan opsi Pengaturan default. Perhatikan pengaturan default berikut untuk tabel Multi-region.
  - Mode kapasitas - Mode kapasitas default adalah On-demand. Untuk informasi selengkapnya tentang mengonfigurasi mode yang disediakan, lihat. [the section called “Buat tabel Multi-wilayah dalam mode yang disediakan”](#)
  - Manajemen kunci enkripsi - Hanya Kunci milik AWSops yang didukung.
  - Client-side stempel waktu - Fitur ini diperlukan untuk tabel Multi-wilayah.
- Pilih Sesuaikan pengaturan jika Anda perlu mengaktifkan Time to Live (TTL) untuk tabel dan semua replika.

 Note

Anda tidak akan dapat mengubah pengaturan TTL pada tabel Multi-wilayah yang ada.

7. Untuk menyelesaikannya, pilih Buat tabel.

## Cassandra Query Language (CQL)

Buat tabel Multi-region dalam mode on-demand dengan pengaturan default

- Untuk membuat tabel Multi-region dengan pengaturan default, Anda dapat menggunakan pernyataan CQL berikut.

```
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

### Menggunakan AWS CLI

1. Untuk membuat tabel Multi-region dengan pengaturan default, Anda hanya perlu menentukan skema. Anda dapat menggunakan contoh berikut.

```
aws keyspaces create-table --keyspace-name mykeyspace --table-name mytable \
--schema-definition 'allColumns=[{name=pk,type=int}],partitionKeys={name= pk}'
```

Output dari perintah tersebut adalah:

```
{
 "resourceArn": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
mykeyspace/table/mytable"
}
```

2. Untuk mengonfirmasi pengaturan tabel, Anda dapat menggunakan pernyataan berikut.

```
aws keyspaces get-table --keyspace-name mykeyspace --table-name mytable
```

Output menunjukkan semua pengaturan default dari tabel Multi-region.

```
{
 "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
 }
 }
]
}
```

## Buat tabel Multi-wilayah dalam mode yang disediakan dengan penskalaan otomatis di Amazon Keyspaces

Bagian ini memberikan contoh cara membuat tabel Multi-wilayah dalam mode yang disediakan dengan penskalaan otomatis. Anda dapat melakukan ini di konsol Amazon Keyspaces, menggunakan CQL atau AWS CLI

Untuk informasi selengkapnya tentang konfigurasi yang didukung dan fitur replikasi multi-wilayah, lihat [the section called “Catatan penggunaan”](#)

Untuk membuat ruang kunci Multi-wilayah, lihat [the section called “Buat ruang kunci Multi-wilayah”](#)

Saat Anda membuat tabel Multi-wilayah baru dalam mode yang disediakan dengan pengaturan penskalaan otomatis, Anda dapat menentukan pengaturan umum untuk tabel yang valid untuk semua tabel Wilayah AWS yang direplikasi. Anda kemudian dapat menimpa pengaturan kapasitas baca dan membaca pengaturan penskalaan otomatis untuk setiap replika. Kapasitas tulis, bagaimanapun, tetap disinkronkan antara semua replika untuk memastikan bahwa ada kapasitas yang cukup untuk mereplikasi penulisan di semua Wilayah.

### Note

Penskalaan otomatis Amazon Keyspaces memerlukan keberadaan peran terkait layanan (AWSServiceRoleForApplicationAutoScaling\_CassandraTable) yang melakukan tindakan penskalaan otomatis atas nama Anda. Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Menggunakan Peran Terkait Layanan”](#).

## Console

Buat tabel Multi-wilayah baru dengan penskalaan otomatis diaktifkan

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pilih ruang kunci Multi-wilayah.
3. Pada tab Tabel, pilih Buat tabel.
4. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel baru.
5. Di bagian Kolom, buat skema untuk tabel Anda.
6. Di bagian kunci Primer, tentukan kunci utama tabel dan pilih kolom pengelompokan opsional.
7. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
8. Lanjutkan ke pengaturan Read/write kapasitas.
9. Untuk mode Kapasitas, pilih Provisioned.
10. Di bagian Baca kapasitas, konfirmasi bahwa Skala dipilih secara otomatis.

Anda dapat memilih untuk mengonfigurasi unit kapasitas baca yang sama untuk semua tabel Wilayah AWS yang direplikasi. Atau, Anda dapat menghapus kotak centang dan mengonfigurasi kapasitas baca untuk setiap Wilayah secara berbeda.

Jika Anda memilih untuk mengonfigurasi setiap Wilayah secara berbeda, Anda memilih unit kapasitas baca minimum dan maksimum untuk setiap replika tabel, serta pemanfaatan target.

- Unit kapasitas minimum — Masukkan nilai untuk tingkat throughput minimum yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimum per detik untuk akun Anda (40.000 secara default).
- Unit kapasitas maksimum — Masukkan jumlah maksimum throughput yang ingin Anda berikan untuk tabel. Nilai harus antara 1 dan kuota throughput maksimum per detik untuk akun Anda (40.000 secara default).
- Target pemanfaatan — Masukkan tingkat pemanfaatan target antara 20% dan 90%. Ketika lalu lintas melebihi tingkat pemanfaatan target yang ditentukan, kapasitas secara otomatis ditingkatkan. Ketika lalu lintas jatuh di bawah target yang ditentukan, secara otomatis diperkecil lagi.

- Kosongkan kotak centang Skala secara otomatis jika Anda ingin menyediakan kapasitas baca tabel secara manual. Pengaturan ini berlaku untuk semua replika tabel.

 Note

Untuk memastikan bahwa ada kapasitas baca yang cukup untuk semua replika, kami merekomendasikan penskalaan otomatis Amazon Keyspaces untuk tabel Multi-wilayah yang disediakan.

 Note

Untuk mempelajari lebih lanjut tentang kuota default untuk akun Anda dan cara meningkatkannya, lihat [Kuota](#).

11. Di bagian Tulis kapasitas, konfirmasi bahwa Skala dipilih secara otomatis. Kemudian konfigurasi unit kapasitas untuk tabel. Unit kapasitas tulis tetap disinkronkan di seluruh wilayah Wilayah AWS untuk memastikan bahwa ada kapasitas yang cukup untuk mereplikasi peristiwa penulisan di seluruh Wilayah.
- Hapus Skala secara otomatis jika Anda ingin menyediakan kapasitas tulis tabel secara manual. Pengaturan ini berlaku untuk semua replika tabel.

 Note

Untuk memastikan bahwa ada kapasitas tulis yang cukup untuk semua replika, kami merekomendasikan penskalaan otomatis Amazon Keyspaces untuk tabel Multi-wilayah yang disediakan.

12. Pilih Buat tabel. Tabel Anda dibuat dengan parameter penskalaan otomatis yang ditentukan.

## Cassandra Query Language (CQL)

Buat tabel Multi-wilayah dengan mode kapasitas yang disediakan dan penskalaan otomatis menggunakan CQL

- Untuk membuat tabel Multi-wilayah dalam mode yang disediakan dengan penskalaan otomatis, Anda harus terlebih dahulu menentukan mode kapasitas dengan menentukan



CUSTOM\_PROPERTIES tabel. Setelah menentukan mode kapasitas yang disediakan, Anda dapat mengonfigurasi pengaturan penskalaan otomatis untuk tabel yang digunakan. AUTOSCALING\_SETTINGS

Untuk informasi mendetail tentang pengaturan penskalaan otomatis, kebijakan pelacakan target, nilai target, dan setelan opsional, lihat [the section called “Buat tabel baru dengan penskalaan otomatis”](#).

Untuk menentukan kapasitas baca replika tabel di Wilayah tertentu, Anda dapat mengonfigurasi parameter berikut sebagai bagian dari tabel: `replica_updates`

- Wilayah
- Unit kapasitas baca yang disediakan (opsional)
- Pengaturan penskalaan otomatis untuk kapasitas baca (opsional)

Contoh berikut menunjukkan CREATE TABLE pernyataan untuk tabel Multi-region dalam mode yang disediakan. Pengaturan penskalaan otomatis kapasitas tulis dan baca umum adalah sama. Namun, pengaturan penskalaan otomatis baca menentukan periode cooldown tambahan 60 detik sebelum menskalakan kapasitas baca tabel ke atas atau ke bawah. Selain itu, pengaturan penskalaan otomatis kapasitas baca untuk Wilayah AS Timur (Virginia N.) lebih tinggi daripada replika lainnya. Juga, nilai target diatur ke 70%, bukan 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

Buat tabel Multi-wilayah baru dalam mode yang disediakan dengan penskalaan otomatis menggunakan AWS CLI

- Untuk membuat tabel Multi-wilayah dalam mode yang disediakan dengan konfigurasi penskalaan otomatis, Anda dapat menggunakan tabel. AWS CLI Perhatikan bahwa Anda harus menggunakan `create-table` perintah Amazon Keyspaces CLI untuk mengonfigurasi pengaturan penskalaan otomatis Multi-wilayah. Ini karena Application Auto Scaling, layanan yang digunakan Amazon Keyspaces untuk melakukan penskalaan otomatis atas nama Anda, tidak mendukung beberapa Wilayah.

Untuk informasi selengkapnya tentang pengaturan penskalaan otomatis, kebijakan pelacakan target, nilai target, dan setelan opsional, lihat [the section called “Buat tabel baru dengan penskalaan otomatis”](#).

Untuk menentukan kapasitas baca replika tabel di Wilayah tertentu, Anda dapat mengonfigurasi parameter berikut sebagai bagian dari tabel: `replicaSpecifications`

- Wilayah
- Unit kapasitas baca yang disediakan (opsional)
- Pengaturan penskalaan otomatis untuk kapasitas baca (opsional)

Saat Anda membuat tabel Multi-wilayah yang disediakan dengan pengaturan penskalaan otomatis yang kompleks dan konfigurasi berbeda untuk replika tabel, akan sangat membantu untuk memuat pengaturan penskalaan otomatis tabel dan konfigurasi replika dari file JSON.

Untuk menggunakan contoh kode berikut, Anda dapat men-download contoh file JSON dari [auto-scaling.zip](#), dan ekstrak `auto-scaling.json` dan `replication.json`. Catat jalur ke file.

Dalam contoh ini, file JSON terletak di direktori saat ini. Untuk opsi jalur file yang berbeda, lihat [Cara memuat parameter dari file](#).

```
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
```

## Memperbarui kapasitas yang disediakan dan pengaturan penskalaan otomatis untuk tabel Multi-wilayah di Amazon Keyspaces

Bagian ini mencakup contoh cara menggunakan konsol, CQL, dan pengaturan penskalaan otomatis Amazon Keyspaces AWS CLI untuk tabel Multi-wilayah yang disediakan. Untuk informasi selengkapnya tentang opsi konfigurasi penskalaan otomatis umum dan cara kerjanya, lihat [the section called “Kelola kapasitas throughput dengan penskalaan otomatis”](#).

Perhatikan bahwa jika Anda menggunakan mode kapasitas yang disediakan untuk tabel Multi-wilayah, Anda harus selalu menggunakan panggilan API Amazon Keyspaces untuk mengonfigurasi

penskalaan otomatis. Ini karena operasi Application Auto Scaling API yang mendasarinya tidak Region-aware

Untuk informasi selengkapnya tentang cara memperkirakan throughput kapasitas tulis dari tabel Multi-wilayah yang disediakan, lihat [the section called “Perkiraan kapasitas untuk tabel Multi-wilayah”](#)

Untuk informasi selengkapnya tentang Amazon Keyspaces API, lihat Referensi API Amazon [Keyspaces](#).

Saat memperbarui mode yang disediakan atau pengaturan penskalaan otomatis dari tabel Multi-wilayah, Anda dapat memperbarui pengaturan kapasitas baca dan konfigurasi penskalaan otomatis baca untuk setiap replika tabel.

Kapasitas tulis, bagaimanapun, tetap disinkronkan antara semua replika untuk memastikan bahwa ada kapasitas yang cukup untuk mereplikasi penulisan di semua Wilayah.

## Cassandra Query Language (CQL)

Memperbarui kapasitas yang disediakan dan pengaturan penskalaan otomatis dari tabel Multi-wilayah menggunakan CQL

- Anda dapat menggunakan ALTER TABLE untuk memperbarui mode kapasitas dan pengaturan penskalaan otomatis dari tabel yang ada. Jika Anda memperbarui tabel yang saat ini dalam mode kapasitas sesuai permintaan, `capacity_mode` diperlukan. Jika tabel Anda sudah dalam mode kapasitas yang disediakan, bidang ini dapat dihilangkan.

Untuk informasi mendetail tentang pengaturan penskalaan otomatis, kebijakan pelacakan target, nilai target, dan setelan opsional, lihat [the section called “Buat tabel baru dengan penskalaan otomatis”](#).

Dalam pernyataan yang sama, Anda juga dapat memperbarui kapasitas baca dan pengaturan penskalaan otomatis replika tabel di Wilayah tertentu dengan memperbarui properti tabel. `replica_updates` Pernyataan berikut adalah contoh dari ini.

```
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

Memperbarui kapasitas yang disediakan dan pengaturan penskalaan otomatis dari tabel Multi-wilayah menggunakan AWS CLI

- Untuk memperbarui mode yang disediakan dan konfigurasi penskalaan otomatis dari tabel yang ada, Anda dapat menggunakan perintah. AWS CLI `update-table`

Perhatikan bahwa Anda harus menggunakan perintah Amazon Keyspaces CLI untuk membuat atau memodifikasi pengaturan penskalaan otomatis Multi-wilayah. Ini karena Application Auto Scaling, layanan yang digunakan Amazon Keyspaces untuk melakukan penskalaan otomatis kapasitas tabel atas nama Anda, tidak mendukung banyak. Wilayah AWS

Untuk memperbarui kapasitas baca replika tabel di Wilayah tertentu, Anda dapat mengubah salah satu parameter opsional tabel berikut: `replicaSpecifications`

- Unit kapasitas baca yang disediakan (opsional)
- Pengaturan penskalaan otomatis untuk kapasitas baca (opsional)

Saat Anda memperbarui tabel Multi-wilayah dengan pengaturan penskalaan otomatis yang kompleks dan konfigurasi berbeda untuk replika tabel, akan sangat membantu untuk memuat pengaturan penskalaan otomatis tabel dan konfigurasi replika dari file JSON.

Untuk menggunakan contoh kode berikut, Anda dapat men-download contoh file JSON dari [auto-scaling.zip](#), dan ekstrak `auto-scaling.json` dan `replication.json`. Catat jalur ke file.

Dalam contoh ini, file JSON terletak di direktori saat ini. Untuk opsi jalur file yang berbeda, lihat [Cara memuat parameter dari file](#).

```
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
```

## Melihat pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah di Amazon Keyspaces

Anda dapat melihat pengaturan kapasitas dan penskalaan otomatis yang disediakan tabel Multi-wilayah di konsol Amazon Keyspaces, menggunakan CQL, atau. AWS CLI Bagian ini memberikan contoh bagaimana melakukan ini menggunakan CQL dan. AWS CLI

### Cassandra Query Language (CQL)

Melihat kapasitas yang disediakan dan pengaturan penskalaan otomatis dari tabel Multi-wilayah menggunakan CQL

- Untuk melihat konfigurasi penskalaan otomatis dari tabel Multi-region, gunakan perintah berikut.

```
SELECT * FROM system_multiregion_info.autoscaling WHERE keyspace_name =
'mykeyspace' AND table_name = 'mytable';
```

Output untuk perintah ini terlihat seperti berikut:

```
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

Melihat kapasitas yang disediakan dan pengaturan penskalaan otomatis dari tabel Multi-wilayah menggunakan AWS CLI

- Untuk melihat konfigurasi penskalaan otomatis dari tabel Multi-wilayah, Anda dapat menggunakan operasi. `get-table-auto-scaling-settings` Perintah CLI berikut adalah contoh dari ini.

```
aws keyspaces get-table-auto-scaling-settings --keyspace-name mykeyspace --
table-name mytable
```

Anda akan melihat output berikut.

```
{
 "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
 }
 }
 }
 }
}
]
}

```

## Matikan penskalaan otomatis untuk tabel di Amazon Keyspaces

Bagian ini memberikan contoh cara mematikan penskalaan otomatis untuk tabel Multi-wilayah dalam mode kapasitas yang disediakan. Anda dapat melakukan ini di konsol Amazon Keyspaces, menggunakan CQL atau AWS CLI

### Important

Sebaiknya gunakan penskalaan otomatis untuk tabel Multi-wilayah yang menggunakan mode kapasitas yang disediakan. Untuk informasi selengkapnya, lihat [the section called “Perkiraan kapasitas untuk tabel Multi-wilayah”](#).

### Note

Untuk menghapus peran terkait layanan yang digunakan Application Auto Scaling, Anda harus menonaktifkan penskalaan otomatis pada semua tabel di akun di semua Wilayah AWS

## Console

Matikan penskalaan otomatis Amazon Keyspaces untuk tabel Multi-wilayah yang ada di konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Pilih tabel yang ingin Anda gunakan dan pilih tab Kapasitas.
3. Di bagian Pengaturan kapasitas, pilih Edit.
4. Untuk menonaktifkan penskalaan otomatis Amazon Keyspaces, kosongkan kotak centang Skala secara otomatis. Menonaktifkan penskalaan otomatis membatalkan pendaftaran tabel sebagai target yang dapat diskalakan dengan Application Auto Scaling. Untuk menghapus peran terkait layanan yang digunakan Application Auto Scaling untuk mengakses tabel Amazon Keyspaces, ikuti langkah-langkahnya. [the section called “Menghapus peran terkait layanan untuk Amazon Keyspaces”](#)
5. Saat pengaturan penskalaan otomatis ditentukan, pilih Simpan.

## Cassandra Query Language (CQL)

Matikan penskalaan otomatis untuk tabel Multi-wilayah menggunakan CQL

- Anda dapat menggunakan `ALTER TABLE` untuk mematikan auto scaling untuk tabel yang ada. Perhatikan bahwa Anda tidak dapat mematikan penskalaan otomatis untuk replika tabel individual.

Dalam contoh berikut, penskalaan otomatis dimatikan untuk kapasitas baca tabel.

```
ALTER TABLE mykeyspace.mytable
WITH AUTOSCALING_SETTINGS = {
 'provisioned_read_capacity_autoscaling_update': {
 'autoscaling_disabled': true
 }
};
```

## CLI

Matikan penskalaan otomatis untuk tabel Multi-wilayah menggunakan AWS CLI

- Anda dapat menggunakan AWS CLI `update-table` perintah untuk mematikan penskalaan otomatis untuk tabel yang ada. Perhatikan bahwa Anda tidak dapat mematikan penskalaan otomatis untuk replika tabel individual.

Dalam contoh berikut, penskalaan otomatis dimatikan untuk kapasitas baca tabel.

```
aws keyspaces update-table --keyspace-name mykeyspace --table-name mytable
 \ --auto-scaling-specification
 readCapacityAutoScaling={autoScalingDisabled=true}
```

## Mengatur kapasitas yang disediakan dari tabel Multi-wilayah secara manual di Amazon Keyspaces

Jika Anda harus mematikan penskalaan otomatis untuk tabel Multi-wilayah, Anda dapat menyediakan kapasitas baca tabel untuk tabel replika secara manual menggunakan CQL atau tabel. AWS CLI

**Note**

Sebaiknya gunakan penskalaan otomatis untuk tabel Multi-wilayah yang menggunakan mode kapasitas yang disediakan. Untuk informasi selengkapnya, lihat [the section called “Perkiraan kapasitas untuk tabel Multi-wilayah”](#).

## Cassandra Query Language (CQL)

Mengatur kapasitas yang disediakan dari tabel Multi-wilayah secara manual menggunakan CQL

- Anda dapat menggunakan ALTER TABLE untuk menyediakan kapasitas baca tabel untuk tabel replika secara manual.

```
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

Mengatur kapasitas yang disediakan dari tabel Multi-region secara manual menggunakan AWS CLI

- Jika Anda harus mematikan penskalaan otomatis untuk tabel Multi-wilayah, Anda dapat menggunakan update-table untuk menyediakan kapasitas baca tabel untuk tabel replika secara manual.

```
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
```

## Cadangkan dan pulihkan data dengan pemulihan point-in-time untuk Amazon Keyspaces

Point-in-time recovery (PITR) membantu melindungi tabel Amazon Keyspaces Anda dari operasi tulis atau penghapusan yang tidak disengaja dengan menyediakan pencadangan berkelanjutan dari data tabel Anda.

Misalnya, skrip pengujian menulis secara tidak sengaja ke tabel Amazon Keyspaces produksi. Dengan pemulihan point-in-time, Anda dapat memulihkan data tabel itu ke detik kapan saja sejak PITR diaktifkan dalam 35 hari terakhir. Jika Anda menghapus tabel dengan pemulihan point-in-time diaktifkan, Anda dapat menanyakan data tabel yang dihapus selama 35 hari (tanpa biaya tambahan), dan mengembalikannya ke keadaan tepat sebelum titik penghapusan.

Anda dapat memulihkan tabel Amazon Keyspaces ke titik waktu dengan menggunakan konsol, AWS SDK dan AWS Command Line Interface (AWS CLI), atau Cassandra Query Language (CQL). Untuk informasi selengkapnya, lihat [Gunakan pemulihan point-in-time di Amazon Keyspaces](#).

Point-in-time operasi tidak memiliki dampak kinerja atau ketersediaan pada tabel dasar, dan memulihkan tabel tidak menggunakan throughput tambahan.

Untuk informasi tentang kuota PITR, lihat [Kuota](#)

Untuk informasi tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

Topik

- [Cara kerja pemulihan point-in-time di Amazon Keyspaces](#)
- [Gunakan pemulihan point-in-time di Amazon Keyspaces](#)

## Cara kerja pemulihan point-in-time di Amazon Keyspaces

Bagian ini memberikan ikhtisar tentang cara kerja pemulihan point-in-time (PITR) Amazon Keyspaces. Untuk informasi selengkapnya tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

Topik

- [Jendela waktu untuk pencadangan berkelanjutan Pitr](#)
- [Pengaturan pemulihan Pitr](#)
- [Pitr mengembalikan tabel terenkripsi](#)
- [Pitr mengembalikan tabel Multi-wilayah](#)
- [Pitr mengembalikan tabel dengan tipe yang ditentukan pengguna \(UDT\)](#)
- [Waktu pemulihan tabel dengan Pitr](#)
- [Amazon Keyspaces Pitr dan integrasi dengan layanan AWS](#)

## Jendela waktu untuk pencadangan berkelanjutan Pitr

Amazon Keyspaces Pitr menggunakan dua stempel waktu untuk mempertahankan kerangka waktu cadangan yang dapat dipulihkan tersedia untuk sebuah tabel.

- Waktu paling awal yang dapat dipulihkan — Menandai waktu pencadangan paling awal yang dapat dipulihkan. Pencadangan paling awal yang dapat dipulihkan kembali hingga 35 hari atau ketika Pitr diaktifkan, mana yang lebih baru. Jendela cadangan maksimum 35 hari tidak dapat diubah.
- Waktu saat ini - Stempel waktu untuk cadangan terbaru yang dapat dipulihkan adalah waktu saat ini. Jika tidak ada stempel waktu yang diberikan selama pemulihan, waktu saat ini digunakan.

Ketika Pitr diaktifkan, Anda dapat mengembalikan ke titik waktu antara

`EarliestRestorableDateTime` dan `CurrentTime`. Anda hanya dapat mengembalikan data tabel ke waktu ketika Pitr diaktifkan.

Jika Anda menonaktifkan Pitr dan kemudian mengaktifkannya kembali, Anda mengatur ulang waktu mulai untuk cadangan pertama yang tersedia saat Pitr diaktifkan kembali. Ini berarti bahwa menonaktifkan Pitr menghapus riwayat cadangan Anda.

### Note

Operasi bahasa definisi data (DDL) pada tabel, seperti perubahan skema, dilakukan secara asinkron. Anda hanya dapat melihat operasi yang telah selesai dalam data tabel yang dipulihkan, tetapi Anda mungkin melihat tindakan tambahan pada tabel sumber jika sedang berlangsung pada saat pemulihan. Untuk daftar pernyataan DDL, lihat [the section called “Pernyataan DDL”](#).

Tabel tidak harus aktif untuk dipulihkan. Anda juga dapat memulihkan tabel yang dihapus jika PITR diaktifkan pada tabel yang dihapus dan penghapusan terjadi dalam jendela cadangan (atau dalam 35 hari terakhir).

#### Note

Jika tabel baru dibuat dengan nama yang memenuhi syarat yang sama (misalnya, `mykeyspace.mytable`) sebagai tabel yang dihapus sebelumnya, tabel yang dihapus tidak akan lagi dapat dipulihkan. Jika Anda mencoba melakukan ini dari konsol, peringatan akan ditampilkan.

## Pengaturan pemulihan PITR

Saat memulihkan tabel menggunakan PITR, Amazon Keyspaces mengembalikan skema dan data tabel sumber Anda ke status berdasarkan timestamp `day:hour:minute:second ()` yang dipilih ke tabel baru. PITR tidak menimpa tabel yang ada.

Selain skema dan data tabel, PITR mengembalikan `custom_properties` dari tabel sumber. Tidak seperti data tabel, yang dipulihkan berdasarkan stempel waktu yang dipilih antara waktu pemulihan paling awal dan waktu saat ini, properti kustom selalu dipulihkan berdasarkan pengaturan tabel pada waktu saat ini.

Pengaturan tabel yang dipulihkan cocok dengan pengaturan tabel sumber dengan stempel waktu saat pemulihan dimulai. Jika Anda ingin menimpa pengaturan ini selama pemulihan, Anda dapat melakukannya dengan menggunakan `WITH custom_properties`. Properti kustom termasuk pengaturan berikut.

- Read/write modus kapasitas
- Pengaturan kapasitas throughput yang disediakan
- Pengaturan PITR

Jika tabel dalam mode kapasitas yang disediakan dengan penskalaan otomatis diaktifkan, operasi pemulihan juga mengembalikan pengaturan penskalaan otomatis tabel. Anda dapat menimpa mereka menggunakan `autoscaling_settings` parameter di CQL atau dengan `CLlautoScalingSpecification`. Untuk informasi selengkapnya tentang pengaturan penskalaan otomatis, lihat [the section called “Kelola kapasitas throughput dengan penskalaan otomatis”](#).



Ketika Anda melakukan pemulihan tabel penuh, semua pengaturan tabel untuk tabel yang dipulihkan berasal dari pengaturan saat ini dari tabel sumber pada saat pemulihan.

Sebagai contoh, misalkan throughput yang disediakan tabel baru-baru ini diturunkan ke 50 unit kapasitas baca dan 50 unit kapasitas tulis. Anda kemudian mengembalikan status tabel ke tiga minggu yang lalu. Pada saat ini, throughput yang disediakan ditetapkan menjadi 100 unit kapasitas baca dan 100 unit kapasitas tulis. Dalam hal ini, Amazon Keyspaces mengembalikan data tabel Anda ke titik waktu tersebut, tetapi menggunakan pengaturan throughput yang disediakan saat ini (50 unit kapasitas baca dan 50 unit kapasitas tulis).

Pengaturan berikut tidak dipulihkan, dan Anda harus mengkonfigurasinya secara manual untuk tabel baru.

- Amazon Keyspaces mengubah aliran pengambilan data (CDC)
- AWS Identity and Access Management Kebijakan (IAM)
- CloudWatch Metrik dan alarm Amazon
- Tag (dapat ditambahkan ke RESTORE pernyataan CQL menggunakan) WITH TAGS

## PITR mengembalikan tabel terenkripsi

Saat Anda memulihkan tabel menggunakan PITR, Amazon Keyspaces mengembalikan setelan enkripsi tabel sumber Anda. Jika tabel dienkripsi dengan Kunci milik AWS (default), tabel dipulihkan dengan pengaturan yang sama secara otomatis. Jika tabel yang ingin Anda pulihkan dienkripsi menggunakan kunci yang dikelola pelanggan, kunci terkelola pelanggan yang sama harus dapat diakses ke Amazon Keyspaces untuk memulihkan data tabel.

Anda dapat mengubah pengaturan enkripsi tabel pada saat pemulihan. Untuk beralih dari kunci yang dikelola pelanggan Kunci milik AWS ke kunci yang dikelola pelanggan, Anda harus menyediakan kunci terkelola pelanggan yang valid dan dapat diakses pada saat pemulihan.

Jika Anda ingin mengubah dari kunci yang dikelola pelanggan ke kunci Kunci milik AWS, konfirmasi bahwa Amazon Keyspaces memiliki akses ke kunci terkelola pelanggan dari tabel sumber untuk memulihkan tabel dengan file. Kunci milik AWS Untuk informasi selengkapnya tentang enkripsi pada pengaturan istirahat untuk tabel, lihat [the section called “Cara kerjanya”](#).

### Note

Jika tabel dihapus karena Amazon Keyspaces kehilangan akses ke kunci yang dikelola pelanggan, Anda perlu memastikan kunci yang dikelola pelanggan dapat diakses oleh

Amazon Keyspaces sebelum mencoba memulihkan tabel. Tabel yang dienkripsi dengan kunci yang dikelola pelanggan tidak dapat dipulihkan jika Amazon Keyspaces tidak memiliki akses ke kunci tersebut. Untuk informasi selengkapnya, lihat [Memecahkan masalah akses kunci](#) di Panduan AWS Key Management Service Pengembang.

## PITR mengembalikan tabel Multi-wilayah

Anda dapat mengembalikan tabel Multi-wilayah menggunakan PITR. Agar operasi pemulihan berhasil, PITR harus diaktifkan pada semua replika tabel sumber dan tabel tujuan harus direplikasi ke yang sama. Wilayah AWS

Amazon Keyspaces mengembalikan pengaturan tabel sumber di setiap Wilayah yang direplikasi yang merupakan bagian dari ruang kunci. Anda juga dapat mengganti pengaturan selama operasi pemulihan. Untuk informasi selengkapnya tentang pengaturan yang dapat diubah selama pemulihan, lihat [the section called “Kembalikan pengaturan”](#).

Untuk informasi selengkapnya tentang replikasi Multi-wilayah, lihat [the section called “Cara kerjanya”](#)

## PITR mengembalikan tabel dengan tipe yang ditentukan pengguna (UDT)

Anda dapat mengembalikan tabel yang menggunakan UDT. Agar operasi pemulihan berhasil, UDT yang direferensikan harus ada dan valid di ruang kunci.

Jika ada UDT yang diperlukan yang hilang saat Anda mencoba memulihkan tabel, Amazon Keyspaces mencoba memulihkan skema UDT secara otomatis dan kemudian melanjutkan untuk memulihkan tabel.

Jika Anda menghapus dan membuat ulang UDT, Amazon Keyspaces mengembalikan UDT dengan skema UDT yang baru dan menolak permintaan untuk memulihkan tabel menggunakan skema UDT asli. Dalam hal ini, jika Anda ingin mengembalikan tabel dengan skema UDT lama, Anda dapat mengembalikan tabel ke ruang kunci baru. Ketika Anda menghapus dan membuat ulang UDT, bahkan jika skema UDT yang dibuat ulang sama dengan skema UDT yang dihapus, UDT yang dibuat ulang dianggap UDT baru. Dalam hal ini, Amazon Keyspaces menolak permintaan untuk mengembalikan tabel dengan skema UDT lama.

Jika UDT hilang dan Amazon Keyspaces mencoba memulihkan UDT, upaya gagal jika Anda telah mencapai jumlah maksimum UDT untuk akun di Wilayah.

Untuk informasi selengkapnya tentang kuota UDT dan nilai default, lihat [the section called “Kuota dan nilai default untuk tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces”](#) Untuk informasi selengkapnya tentang bekerja dengan UDT, lihat [the section called “User-defined jenis \(UDT\)”](#).

## Waktu pemulihan tabel dengan PITR

Waktu yang dibutuhkan untuk mengembalikan tabel didasarkan pada beberapa faktor dan tidak selalu berkorelasi langsung dengan ukuran tabel.

Berikut ini adalah beberapa pertimbangan untuk waktu pemulihan.

- Anda mengembalikan cadangan ke tabel baru. Diperlukan waktu hingga 20 menit (bahkan jika tabel kosong) untuk melakukan semua tindakan untuk membuat tabel baru dan memulai proses pemulihan.
- Waktu pemulihan untuk tabel besar dengan model data yang terdistribusi dengan baik bisa beberapa jam atau lebih lama.
- Jika tabel sumber Anda berisi data yang miring secara signifikan, waktu untuk memulihkan mungkin meningkat. Misalnya, jika kunci utama tabel Anda menggunakan bulan dalam setahun sebagai kunci partisi, dan semua data Anda berasal dari bulan Desember, Anda memiliki data miring.

Praktik terbaik saat merencanakan pemulihan bencana adalah mendokumentasikan waktu penyelesaian pemulihan rata-rata secara teratur dan menetapkan bagaimana waktu tersebut memengaruhi Sasaran Waktu Pemulihan Anda secara keseluruhan.

## Amazon Keyspaces PITR dan integrasi dengan layanan AWS

Operasi PITR berikut dicatat menggunakan AWS CloudTrail untuk memungkinkan pemantauan dan audit berkelanjutan.

- Buat tabel baru dengan PITR diaktifkan atau dinonaktifkan.
- Aktifkan atau nonaktifkan PITR pada tabel yang ada.
- Kembalikan tabel yang aktif atau dihapus.

Untuk informasi selengkapnya, lihat [Mencatat panggilan API Amazon Keyspaces dengan AWS CloudTrail](#).

Anda dapat melakukan tindakan PITR berikut menggunakan CloudFormation.

- Buat tabel baru dengan PITR diaktifkan atau dinonaktifkan.
- Aktifkan atau nonaktifkan PITR pada tabel yang ada.

Untuk informasi selengkapnya, lihat [Referensi Jenis Sumber Daya Cassandra](#) di [CloudFormation Panduan Pengguna](#).

## Gunakan pemulihan point-in-time di Amazon Keyspaces

Dengan Amazon Keyspaces (untuk Apache Cassandra), Anda dapat mengembalikan tabel ke titik waktu tertentu menggunakan Point-in-Time Restore (PITR). PITR memungkinkan Anda mengembalikan tabel ke keadaan sebelumnya dalam 35 hari terakhir, memberikan perlindungan data dan kemampuan pemulihan. Fitur ini berharga dalam kasus-kasus seperti penghapusan data yang tidak disengaja, kesalahan aplikasi, atau untuk tujuan pengujian. Anda dapat dengan cepat dan efisien memulihkan data, meminimalkan downtime dan kehilangan data. Bagian berikut memandu Anda melalui proses memulihkan tabel menggunakan PITR di Amazon Keyspaces, memastikan integritas data dan kelangsungan bisnis.

### Topik

- [Konfigurasi izin IAM tabel pemulihan untuk Amazon Keyspaces PITR](#)
- [Konfigurasi PITR untuk tabel di Amazon Keyspaces](#)
- [Matikan PITR untuk tabel Amazon Keyspaces](#)
- [Pulihkan tabel dari cadangan ke titik waktu tertentu di Amazon Keyspaces](#)
- [Kembalikan tabel yang dihapus menggunakan Amazon Keyspaces PITR](#)

## Konfigurasi izin IAM tabel pemulihan untuk Amazon Keyspaces PITR

Bagian ini merangkum cara mengonfigurasi izin untuk prinsipal AWS Identity and Access Management (IAM) untuk memulihkan tabel Amazon Keyspaces. Di IAM, kebijakan AWS terkelola `AmazonKeyspacesFullAccess` menyertakan izin untuk memulihkan tabel Amazon Keyspaces. Untuk menerapkan kebijakan khusus dengan izin minimum yang diperlukan, pertimbangkan persyaratan yang diuraikan di bagian berikutnya.

Agar berhasil memulihkan tabel, prinsipal IAM memerlukan izin minimum berikut:

- `cassandra:Restore`— Tindakan pemulihan diperlukan agar tabel target dipulihkan.
- `cassandra:Select`— Tindakan pilih diperlukan untuk membaca dari tabel sumber.

- `cassandra:TagResource`— Tindakan tag adalah opsional, dan hanya diperlukan jika operasi pemulihan menambahkan tag.

Ini adalah contoh kebijakan yang memberikan izin minimum yang diperlukan kepada pengguna untuk memulihkan tabel di ruang kunci. `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*"
]
 }
]
}
```

Izin tambahan untuk memulihkan tabel mungkin diperlukan berdasarkan fitur lain yang dipilih. Misalnya, jika tabel sumber dienkripsi saat istirahat dengan kunci yang dikelola pelanggan, Amazon Keyspaces harus memiliki izin untuk mengakses kunci terkelola pelanggan dari tabel sumber agar berhasil memulihkan tabel. Untuk informasi selengkapnya, lihat [the section called “PITR dan tabel terenkripsi”](#).

Jika Anda menggunakan kebijakan IAM dengan [kunci kondisi](#) untuk membatasi lalu lintas masuk ke sumber tertentu, Anda harus memastikan bahwa Amazon Keyspaces memiliki izin untuk melakukan operasi pemulihan atas nama kepala sekolah Anda. Anda harus menambahkan kunci `aws:ViaAWSService` kondisi ke kebijakan IAM Anda jika kebijakan Anda membatasi lalu lintas masuk ke salah satu dari berikut ini:

- Titik akhir VPC dengan `aws:SourceVpce`
- Rentang IP dengan `aws:SourceIp`
- VPC dengan `aws:SourceVpc`

Kunci `aws:ViaAWSService` kondisi memungkinkan akses ketika AWS layanan apa pun membuat permintaan menggunakan kredensi kepala sekolah. Untuk informasi selengkapnya, lihat [elemen kebijakan IAM JSON: Kunci kondisi](#) di Panduan Pengguna IAM.

Berikut ini adalah contoh kebijakan yang membatasi lalu lintas sumber ke alamat IP tertentu dan memungkinkan Amazon Keyspaces memulihkan tabel atas nama kepala sekolah.

```
{
 "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"
 }
 }
 }
]
}
```

Untuk contoh kebijakan menggunakan kunci kondisi `aws:ViaAWSService` global, lihat [the section called “Kebijakan titik akhir VPC dan pemulihan point-in-time \(PITR\) Amazon Keyspaces”](#).

## Konfigurasi Pitr untuk tabel di Amazon Keyspaces

Anda dapat mengonfigurasi tabel di Amazon Keyspaces untuk operasi pencadangan dan pemulihan menggunakan Pitr dengan konsol, CQL, dan. AWS CLI

Saat membuat tabel baru menggunakan CQL atau AWS CLI, Anda harus secara eksplisit mengaktifkan Pitr dalam pernyataan buat tabel. Saat Anda membuat tabel baru menggunakan konsol, Pitr akan diaktifkan secara default.

Untuk mempelajari cara mengembalikan tabel, lihat [the section called “Kembalikan tabel ke titik waktu”](#).

### Console

Konfigurasi Pitr untuk tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel dan pilih tabel yang ingin Anda edit.
3. Pada tab Backup, pilih Edit.
4. Di bagian Edit pengaturan pemulihan point-in-time, pilih Aktifkan pemulihan. Point-in-time
5. Pilih Simpan perubahan.

### Cassandra Query Language (CQL)

Konfigurasi Pitr untuk tabel menggunakan CQL

1. Anda dapat mengelola pengaturan Pitr untuk tabel dengan menggunakan properti `point_in_time_recovery` kustom.

Untuk mengaktifkan Pitr saat Anda membuat tabel baru, Anda harus mengatur status `point_in_time_recovery` ke `enabled`. Anda dapat menggunakan perintah CQL berikut sebagai contoh.

```
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

Jika tidak ada properti kustom pemulihan point-in-time yang ditentukan, pemulihan point-in-time dinonaktifkan secara default.

2. Untuk mengaktifkan Pitr untuk tabel yang ada menggunakan CQL, jalankan perintah CQL berikut.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'enabled'}}
```

## CLI

### Konfigurasi Pitr untuk tabel menggunakan AWS CLI

1. Anda dapat mengelola pengaturan Pitr untuk tabel dengan menggunakan UpdateTable API.

Untuk mengaktifkan Pitr saat Anda membuat tabel baru, Anda harus menyertakan point-in-time-recovery 'status=ENABLED' dalam perintah create table. Anda dapat menggunakan AWS CLI perintah berikut sebagai contoh. Perintah telah dipecah menjadi baris terpisah untuk meningkatkan keterbacaan.

```
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

Jika tidak ada nilai pemulihan point-in-time yang ditentukan, pemulihan point-in-time dinonaktifkan secara default.



2. Untuk mengonfirmasi pengaturan pemulihan point-in-time untuk sebuah tabel, Anda dapat menggunakan perintah berikut. AWS CLI

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

3. Untuk mengaktifkan Pitr untuk tabel yang ada menggunakan AWS CLI, jalankan perintah berikut.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --point-in-time-recovery 'status=ENABLED'
```

## Matikan Pitr untuk tabel Amazon Keyspaces

Anda dapat menonaktifkan Pitr untuk tabel Amazon Keyspaces kapan saja menggunakan konsol, CQL, atau. AWS CLI

### Important

Menonaktifkan Pitr segera menghapus riwayat cadangan Anda, bahkan jika Anda mengaktifkan kembali Pitr di atas meja dalam waktu 35 hari.

Untuk mempelajari cara mengembalikan tabel, lihat [the section called “Kembalikan tabel ke titik waktu”](#).

### Console

Nonaktifkan Pitr untuk tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel dan pilih tabel yang ingin Anda edit.
3. Pada tab Backup, pilih Edit.
4. Di bagian Edit pengaturan pemulihan point-in-time, kosongkan kotak centang Aktifkan Point-in-time pemulihan.
5. Pilih Simpan perubahan.

## Cassandra Query Language (CQL)

Nonaktifkan Pitr untuk tabel menggunakan CQL

- Untuk menonaktifkan Pitr untuk tabel yang ada, jalankan perintah CQL berikut.

```
ALTER TABLE mykeyspace.mytable
WITH custom_properties = {'point_in_time_recovery': {'status': 'disabled'}}
```

## CLI

Nonaktifkan Pitr untuk tabel menggunakan AWS CLI

- Untuk menonaktifkan Pitr untuk tabel yang ada, jalankan AWS CLI perintah berikut.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
--point-in-time-recovery 'status=DISABLED'
```

## Pulihkan tabel dari cadangan ke titik waktu tertentu di Amazon Keyspaces

Bagian berikut menunjukkan cara mengembalikan tabel Amazon Keyspaces yang ada ke titik waktu tertentu.

### Note

Prosedur ini mengasumsikan bahwa tabel yang Anda gunakan telah dikonfigurasi dengan pemulihan point-in-time. Untuk mengaktifkan Pitr untuk tabel, lihat [the section called “Konfigurasi Pitr”](#).

### Important

Saat pemulihan sedang berlangsung, jangan mengubah atau menghapus kebijakan AWS Identity and Access Management (IAM) yang memberikan izin utama IAM (misalnya, pengguna, grup, atau peran) untuk melakukan pemulihan. Jika tidak, perilaku tak terduga dapat terjadi. Misalnya, jika Anda menghapus izin tulis untuk tabel saat tabel tersebut dipulihkan, `RestoreTableToPointInTime` operasi yang mendasarinya tidak dapat menulis data yang dipulihkan ke tabel.

Anda dapat memodifikasi atau menghapus izin hanya setelah operasi pemulihan selesai.

## Console

Kembalikan tabel ke titik waktu tertentu menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi di sisi kiri konsol, pilih Tabel.
3. Dalam daftar tabel, pilih tabel yang ingin Anda pulihkan.
4. Pada tab Cadangan tabel, di bagian Point-in-time pemulihan, pilih Pulihkan.
5. Untuk nama tabel baru, masukkan nama baru untuk tabel yang dipulihkan, misalnya **mytable\_restored**.
6. Untuk menentukan titik waktu operasi pemulihan, Anda dapat memilih di antara dua opsi:
  - Pilih waktu paling awal yang telah dikonfigurasi sebelumnya.
  - Pilih Tentukan tanggal dan waktu dan masukkan tanggal dan waktu yang ingin Anda kembalikan tabel baru.

### Note

Anda dapat mengembalikan ke titik waktu mana pun dalam waktu paling awal dan waktu saat ini. Amazon Keyspaces mengembalikan data tabel Anda ke status berdasarkan tanggal dan waktu yang dipilih (day:hour:minute:second).

7. Pilih Pulihkan untuk memulai proses pemulihan.

Tabel yang sedang dipulihkan ditampilkan dengan status Memulihkan. Setelah proses pemulihan selesai, status tabel yang dipulihkan berubah menjadi Aktif.

## Cassandra Query Language (CQL)

Kembalikan tabel ke titik waktu menggunakan CQL

1. Anda dapat mengembalikan tabel aktif ke point-in-time antara `earliest_restorable_timestamp` dan waktu saat ini. Waktu saat ini adalah default.

Untuk mengonfirmasi bahwa pemulihan point-in-time diaktifkan untuk tabel, kueri `system_schema_mcs.tables` seperti yang ditunjukkan dalam contoh ini.

```
SELECT custom_properties
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable';
```

Point-in-time pemulihan diaktifkan seperti yang ditunjukkan pada output sampel berikut.

```
custom_properties

{
 ...,
 "point_in_time_recovery": {
 "earliest_restorable_timestamp": "2020-06-30T19:19:21.175Z"
 "status": "enabled"
 }
}
```

2. • Kembalikan tabel ke waktu saat ini. Ketika Anda menghilangkan `WITH restore_timestamp = ...` klausa, stempel waktu saat ini digunakan.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

- Anda juga dapat mengembalikan ke titik waktu tertentu, yang ditentukan oleh format `restore_timestamp` ISO 8601. Anda dapat menentukan titik waktu apa pun selama 35 hari terakhir. Sebagai contoh, perintah berikut memulihkan tabel ke `EarliestRestorableDateTime`.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable
WITH restore_timestamp = '2020-06-30T19:19:21.175Z';
```

Untuk deskripsi sintaks lengkap, lihat [the section called “MENGEMBALIKAN TABEL”](#) di referensi bahasa.

3. Untuk memverifikasi bahwa pemulihan tabel berhasil, kueri `system_schema_mcs.tables` untuk mengkonfirmasi status tabel.

```
SELECT status
FROM system_schema_mcs.tables
WHERE keyspace_name = 'mykeyspace' AND table_name = 'mytable_restored'
```

Kueri menunjukkan output berikut.

```
status

RESTORING
```

Tabel yang sedang dipulihkan ditampilkan dengan status Memulihkan. Setelah proses pemulihan selesai, status tabel berubah menjadi Aktif.

## CLI

Kembalikan tabel ke titik waktu menggunakan AWS CLI

1. Buat tabel sederhana bernama myTable yang mengaktifkan PITR. Perintah telah dipecah menjadi baris terpisah untuk keterbacaan.

```
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. Konfirmasikan properti tabel baru dan tinjau earliestRestorableTimestamp untuk PITR.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Output dari perintah ini mengembalikan berikut ini.

```
{
 "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. • Untuk mengembalikan tabel ke titik waktu, tentukan a `restore_timestamp` dalam format ISO 8601. Anda dapat memilih titik waktu apa pun selama 35 hari terakhir dalam interval satu detik. Sebagai contoh, perintah berikut memulihkan tabel ke `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"
```

Output dari perintah ini mengembalikan ARN dari tabel yang dipulihkan.

```
{
 "restoredTableARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/table/myTable_restored"
}
```

- Untuk mengembalikan tabel ke waktu saat ini, Anda dapat menghilangkan `restore-timestamp` parameter.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name 'myTable_restored1'"
```

## Kembalikan tabel yang dihapus menggunakan Amazon Keyspaces PITR

Prosedur berikut menunjukkan cara mengembalikan tabel yang dihapus dari cadangan ke saat penghapusan. Anda dapat melakukan ini menggunakan CQL atau AWS CLI

### Note

Prosedur ini mengasumsikan bahwa PITR diaktifkan pada tabel yang dihapus.

## Cassandra Query Language (CQL)

Kembalikan tabel yang dihapus menggunakan CQL

1. Untuk mengonfirmasi bahwa pemulihan point-in-time diaktifkan untuk tabel yang dihapus, kueri tabel sistem. Hanya tabel dengan pemulihan point-in-time yang diaktifkan yang ditampilkan.

```
SELECT custom_properties
FROM system_schema_mcs.tables_history
```

```
WHERE keyspace_name = 'mykeyspace' AND table_name = 'my_table';
```

Kueri menunjukkan output berikut.

```
custom_properties

{
 ...,
 "point_in_time_recovery":{
 "restorable_until_time":"2020-08-04T00:48:58.381Z",
 "status":"enabled"
 }
}
```

2. Kembalikan tabel ke waktu penghapusan dengan contoh pernyataan berikut.

```
RESTORE TABLE mykeyspace.mytable_restored
FROM TABLE mykeyspace.mytable;
```

## CLI

Kembalikan tabel yang dihapus menggunakan AWS CLI

1. Hapus tabel yang Anda buat sebelumnya yang mengaktifkan PITR. Berikut adalah contoh perintah tersebut.

```
aws keyspaces delete-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

2. Kembalikan tabel yang dihapus ke waktu penghapusan dengan perintah berikut.

```
aws keyspaces restore-table --source-keyspace-name 'myKeyspace' --source-
table-name 'myTable' --target-keyspace-name 'myKeyspace' --target-table-name
'myTable_restored2'
```

Output dari perintah ini mengembalikan ARN dari tabel yang dipulihkan.

```
{
 "restoredTableARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/
myKeyspace/table/myTable_restored2"
}
```



# Data kedaluwarsa dengan Time to Live (TTL) untuk Amazon Keyspaces (untuk Apache Cassandra)

Amazon Keyspaces (untuk Apache Cassandra) Time to Live (TTL) membantu Anda menyederhanakan logika aplikasi dan mengoptimalkan harga penyimpanan dengan kedaluwarsa data dari tabel secara otomatis. Data yang tidak lagi Anda perlukan akan dihapus secara otomatis dari tabel berdasarkan nilai Time to Live yang Anda tetapkan.

Hal ini memudahkan untuk mematuhi kebijakan penyimpanan data berdasarkan persyaratan bisnis, industri, atau peraturan yang menentukan berapa lama data perlu disimpan atau menentukan kapan data harus dihapus.

Misalnya, Anda dapat menggunakan TTL dalam AdTech aplikasi untuk menjadwalkan kapan data untuk iklan tertentu kedaluwarsa dan tidak lagi terlihat oleh klien. Anda juga dapat menggunakan TTL untuk menghentikan data lama secara otomatis dan menghemat biaya penyimpanan Anda.

Anda dapat menetapkan nilai TTL default untuk seluruh tabel, dan menimpa nilai tersebut untuk masing-masing baris dan kolom. Operasi TTL tidak memengaruhi kinerja aplikasi Anda. Selain itu, jumlah baris dan kolom yang ditandai untuk kedaluwarsa dengan TTL tidak memengaruhi ketersediaan tabel Anda.

Amazon Keyspaces secara otomatis menyaring data kedaluwarsa sehingga data kedaluwarsa tidak ditampilkan dalam hasil kueri atau tersedia untuk digunakan dalam pernyataan bahasa manipulasi data (DML/bahasa manipulasi data). Amazon Keyspaces biasanya menghapus data kedaluwarsa dari penyimpanan dalam waktu 10 hari sejak tanggal kedaluwarsa.

Dalam kasus yang jarang terjadi, Amazon Keyspaces mungkin tidak dapat menghapus data dalam waktu 10 hari jika ada aktivitas berkelanjutan pada partisi penyimpanan yang mendasarinya untuk melindungi ketersediaan. Dalam kasus ini, Amazon Keyspaces terus mencoba menghapus data yang kedaluwarsa setelah lalu lintas di partisi berkurang.

Setelah data dihapus secara permanen dari penyimpanan, Anda berhenti mengeluarkan biaya penyimpanan.

Anda dapat mengatur, memodifikasi, atau menonaktifkan pengaturan TTL default untuk tabel baru dan yang sudah ada dengan menggunakan konsol, Cassandra Query Language (CQL), atau. AWS CLI

Pada tabel dengan TTL default yang dikonfigurasi, Anda dapat menggunakan pernyataan CQL untuk mengganti pengaturan TTL default tabel dan menerapkan nilai TTL khusus ke baris dan kolom. Untuk

informasi selengkapnya, lihat [the section called “Gunakan INSERT untuk mengatur TTL kustom untuk baris baru”](#) dan [the section called “Gunakan UPDATE untuk mengatur TTL kustom untuk baris dan kolom”](#).

Harga TTL didasarkan pada ukuran baris yang dihapus atau diperbarui dengan menggunakan Time to Live. Operasi TTL diukur dalam satuan. TTL deletes Satu penghapusan TTL dikonsumsi per KB data per baris yang dihapus atau diperbarui.

Misalnya, untuk memperbarui baris yang menyimpan 2,5 KB data dan untuk menghapus satu atau lebih kolom dalam baris pada saat yang sama memerlukan tiga penghapusan TTL. Atau, untuk menghapus seluruh baris yang berisi 3,5 KB data memerlukan empat penghapusan TTL.

Satu penghapusan TTL dikonsumsi per KB data yang dihapus per baris. Untuk informasi selengkapnya tentang harga, lihat harga [Amazon Keyspaces \(untuk Apache Cassandra\)](#).

## Topik

- [Amazon Keyspaces Waktu untuk Hidup dan integrasi dengan layanan AWS](#)
- [Buat tabel baru dengan pengaturan Time to Live \(TTL\) default](#)
- [Perbarui nilai default Time to Live \(TTL\) dari sebuah tabel](#)
- [Buat tabel dengan pengaturan Time to Live \(TTL\) kustom diaktifkan](#)
- [Perbarui tabel dengan Custom Time to Live \(TTL\)](#)
- [Gunakan INSERT pernyataan untuk menyetel nilai Time to Live \(TTL\) kustom untuk baris baru](#)
- [Gunakan UPDATE pernyataan untuk mengedit pengaturan Time to Live \(TTL\) kustom untuk baris dan kolom](#)

## Amazon Keyspaces Waktu untuk Hidup dan integrasi dengan layanan AWS

Metrik TTL berikut tersedia di Amazon CloudWatch untuk memungkinkan pemantauan berkelanjutan.

- **TTLDeletes**— Unit yang digunakan untuk menghapus atau memperbarui data berturut-turut dengan menggunakan Time to Live (TTL).

Untuk informasi selengkapnya tentang cara memantau CloudWatch metrik, lihat [the section called “Pemantauan CloudWatch dengan”](#).

Saat Anda menggunakan CloudFormation, Anda dapat mengaktifkan TTL saat membuat tabel Amazon Keyspaces. Untuk informasi selengkapnya, silakan lihat [Panduan Pengguna AWS CloudFormation](#).

## Buat tabel baru dengan pengaturan Time to Live (TTL) default

Di Amazon Keyspaces, Anda dapat menetapkan nilai TTL default untuk semua baris dalam tabel saat tabel dibuat.

Nilai TTL default untuk tabel adalah nol, yang berarti bahwa data tidak kedaluwarsa secara otomatis. Jika nilai TTL default untuk tabel lebih besar dari nol, stempel waktu kedaluwarsa ditambahkan ke setiap baris.

Nilai TTL diatur dalam detik, dan nilai maksimum yang dapat dikonfigurasi adalah 630.720.000 detik, yang setara dengan 20 tahun.

Setelah pembuatan tabel, Anda dapat menimpa pengaturan TTL default tabel untuk baris atau kolom tertentu dengan pernyataan DMLCQL. Untuk informasi selengkapnya, lihat [the section called “Gunakan INSERT untuk mengatur TTL kustom untuk baris baru”](#) dan [the section called “Gunakan UPDATE untuk mengatur TTL kustom untuk baris dan kolom”](#).

Saat Anda mengaktifkan TTL di atas meja, Amazon Keyspaces mulai menyimpan metadata TTL-related tambahan untuk setiap baris. Selain itu, TTL menggunakan stempel waktu kedaluwarsa untuk melacak kapan baris atau kolom kedaluwarsa. Stempel waktu disimpan sebagai metadata baris dan berkontribusi pada biaya penyimpanan untuk baris tersebut.

Setelah fitur TTL diaktifkan, Anda tidak dapat menonaktifkannya untuk tabel. Menyetel tabel `default_time_to_live` ke 0 akan menonaktifkan waktu kedaluwarsa default untuk data baru, tetapi tidak menonaktifkan fitur TTL atau mengembalikan tabel kembali ke metadata penyimpanan Amazon Keyspaces asli atau perilaku tulis.

Contoh berikut menunjukkan cara membuat tabel baru dengan nilai TTL default.

### Console

Buat tabel baru dengan nilai default Time to Live menggunakan konsol.

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih Buat tabel.

3. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel baru.
4. Di bagian Skema, buat skema untuk tabel Anda.
5. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
6. Lanjutkan ke Time to Live (TTL).

Pada langkah ini, Anda memilih pengaturan TTL default untuk tabel.

Untuk periode TTL Default, masukkan waktu kedaluwarsa dan pilih satuan waktu yang Anda masukkan, misalnya detik, hari, atau tahun. Amazon Keyspaces akan menyimpan nilainya dalam hitungan detik.

7. Pilih Buat tabel. Tabel Anda dibuat dengan nilai TTL default yang ditentukan.

## Cassandra Query Language (CQL)

Buat tabel baru dengan nilai TTL default menggunakan CQL

1. Pernyataan berikut membuat tabel baru dengan nilai TTL default diatur ke 3.024.000 detik, yang mewakili 35 hari.

```
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. Untuk mengonfirmasi pengaturan TTL untuk tabel baru, gunakan `cqlsh DESCRIBE` pernyataan seperti yang ditunjukkan pada contoh berikut. Output menunjukkan pengaturan TTL default untuk tabel sebagai `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

Buat tabel baru dengan nilai TTL default menggunakan AWS CLI

1. Anda dapat menggunakan perintah berikut untuk membuat tabel baru dengan nilai TTL default diatur ke satu tahun.

```

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. Untuk mengkonfirmasi status TTL tabel, Anda dapat menggunakan perintah berikut.

```

aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'

```

Output dari perintah terlihat seperti pada contoh berikut

```

{
 "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": []
}
```

## Perbarui nilai default Time to Live (TTL) dari sebuah tabel

Anda dapat memperbarui tabel yang ada dengan nilai TTL default baru. Nilai TTL diatur dalam detik, dan nilai maksimum yang dapat dikonfigurasi adalah 630.720.000 detik, yang setara dengan 20 tahun.

Saat Anda mengaktifkan TTL di atas meja, Amazon Keyspaces mulai menyimpan metadata TTL-related tambahan untuk setiap baris. Selain itu, TTL menggunakan stempel waktu kedaluwarsa untuk melacak kapan baris atau kolom kedaluwarsa. Stempel waktu disimpan sebagai metadata baris dan berkontribusi pada biaya penyimpanan untuk baris tersebut.

Setelah TTL diaktifkan untuk tabel, Anda dapat menimpa pengaturan TTL default tabel untuk baris atau kolom tertentu dengan pernyataan CQL DHTML. Untuk informasi selengkapnya, lihat [the section called “Gunakan INSERT untuk mengatur TTL kustom untuk baris baru”](#) dan [the section called “Gunakan UPDATE untuk mengatur TTL kustom untuk baris dan kolom”](#).

Setelah fitur TTL diaktifkan, Anda tidak dapat menonaktifkannya untuk tabel. Menyetel tabel `default_time_to_live` ke 0 akan menonaktifkan waktu kedaluwarsa default untuk data baru, tetapi tidak menonaktifkan fitur TTL atau mengembalikan tabel kembali ke metadata penyimpanan Amazon Keyspaces asli atau perilaku tulis.

Ikuti langkah-langkah ini untuk memperbarui pengaturan Waktu ke Langsung default untuk tabel yang ada menggunakan konsol, CQL, atau AWS CLI

### Console

Perbarui nilai TTL default dari tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pilih tabel yang ingin Anda perbarui, lalu pilih tab Pengaturan tambahan.
3. Lanjutkan ke Time to Live (TTL) dan pilih Edit.
4. Untuk periode TTL Default, masukkan waktu kedaluwarsa dan pilih satuan waktu, misalnya detik, hari, atau tahun. Amazon Keyspaces akan menyimpan nilainya dalam hitungan detik. Ini tidak mengubah nilai TTL dari baris yang ada.

5. Ketika pengaturan TTL ditentukan, pilih Simpan perubahan.

## Cassandra Query Language (CQL)

Perbarui nilai TTL default dari tabel menggunakan CQL

1. Anda dapat menggunakan ALTER TABLE untuk mengedit pengaturan default Time to Live (TTL) dari sebuah tabel. Untuk memperbarui pengaturan TTL default tabel menjadi 2.592.000 detik, yang mewakili 30 hari, Anda dapat menggunakan pernyataan berikut.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

2. Untuk mengonfirmasi pengaturan TTL untuk tabel yang diperbarui, gunakan cqlsh DESCRIBE pernyataan seperti yang ditunjukkan pada contoh berikut. Output menunjukkan pengaturan TTL default untuk tabel sebagai default\_time\_to\_live.

```
DESC TABLE my_table;
```

Output dari pernyataan harus terlihat mirip dengan contoh ini.

```
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

Perbarui nilai TTL default dari tabel menggunakan AWS CLI

1. Anda dapat menggunakan `update-table` untuk mengedit nilai TTL default sebuah tabel. Untuk memperbarui pengaturan TTL default tabel menjadi 2.592.000 detik, yang mewakili 30 hari, Anda dapat menggunakan pernyataan berikut.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable'
--default-time-to-live '2592000'
```

2. Untuk mengonfirmasi nilai TTL default yang diperbarui, Anda dapat menggunakan pernyataan berikut.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Output dari pernyataan akan terlihat seperti pada contoh berikut.

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

## Buat tabel dengan pengaturan Time to Live (TTL) kustom diaktifkan

Untuk membuat tabel baru dengan pengaturan kustom Time to Live yang dapat diterapkan ke baris dan kolom tanpa mengaktifkan pengaturan default TTL untuk seluruh tabel, Anda dapat menggunakan perintah berikut.

### Note

Jika tabel dibuat dengan pengaturan `ttl` khusus diaktifkan, Anda tidak dapat menonaktifkan pengaturan nanti.

## Cassandra Query Language (CQL)

Buat tabel baru dengan pengaturan TTL khusus menggunakan CQL

- ```
CREATE TABLE my_keyspace.my_table (id int primary key) WITH  
CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

CLI

Buat tabel baru dengan pengaturan TTL khusus menggunakan AWS CLI

1. Anda dapat menggunakan perintah berikut untuk membuat tabel baru dengan TTL diaktifkan.

```
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. Untuk mengonfirmasi bahwa TTL diaktifkan untuk tabel, Anda dapat menggunakan pernyataan berikut.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Output dari pernyataan akan terlihat seperti pada contoh berikut.

```
{  
  "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": []
}

```

Perbarui tabel dengan Custom Time to Live (TTL)

Untuk mengaktifkan pengaturan kustom Time to Live untuk tabel sehingga nilai TTL dapat diterapkan ke masing-masing baris dan kolom tanpa menetapkan nilai default TTL untuk seluruh tabel, Anda dapat menggunakan perintah berikut.

Note

Setelah `t1` diaktifkan, Anda tidak dapat menonaktifkannya untuk tabel.

Cassandra Query Language (CQL)

Aktifkan pengaturan TTL khusus untuk tabel menggunakan CQL

- ```
ALTER TABLE my_table WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

## CLI

Aktifkan pengaturan TTL kustom untuk tabel menggunakan AWS CLI

- Anda dapat menggunakan perintah berikut untuk memperbarui pengaturan TTL kustom dari sebuah tabel.

```
aws keyspaces update-table --keyspace-name 'myKeyspace' --table-name 'myTable' --ttl 'status=ENABLED'
```

- Untuk mengonfirmasi bahwa TTL sekarang diaktifkan untuk tabel, Anda dapat menggunakan pernyataan berikut.

```
aws keyspaces get-table --keyspace-name 'myKeyspace' --table-name 'myTable'
```

Output dari pernyataan akan terlihat seperti pada contoh berikut.

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

## Gunakan **INSERT** pernyataan untuk menyetel nilai Time to Live (TTL) kustom untuk baris baru

### Note

Sebelum Anda dapat mengatur nilai TTL kustom untuk baris menggunakan INSERT pernyataan, Anda harus terlebih dahulu mengaktifkan TTL kustom di atas meja. Untuk informasi selengkapnya, lihat [the section called “Perbarui tabel kustom TTL”](#).

Untuk menimpa nilai TTL default tabel dengan menetapkan tanggal kedaluwarsa untuk masing-masing baris, Anda dapat menggunakan pernyataan: INSERT

- INSERT— Masukkan baris data baru dengan set nilai TTL.

Menyetel nilai TTL untuk baris baru menggunakan INSERT pernyataan lebih diutamakan daripada pengaturan TTL default tabel.

Pernyataan CQL berikut menyisipkan deretan data ke dalam tabel dan mengubah pengaturan TTL default menjadi 259.200 detik (yang setara dengan 3 hari).

```
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;
```

Untuk mengonfirmasi pengaturan TTL untuk baris yang disisipkan, gunakan pernyataan berikut.

```
SELECT TTL (subject) from my_table;
```

## Gunakan **UPDATE** pernyataan untuk mengedit pengaturan Time to Live (TTL) kustom untuk baris dan kolom

### Note

Sebelum Anda dapat mengatur nilai TTL khusus untuk baris dan kolom, Anda harus mengaktifkan TTL pada tabel terlebih dahulu. Untuk informasi selengkapnya, lihat [the section called “Perbarui tabel kustom TTL”](#).

Anda dapat menggunakan UPDATE pernyataan untuk menerima nilai TTL default tabel dengan menetapkan tanggal kedaluwarsa untuk masing-masing baris dan kolom:

- Baris - Anda dapat memperbarui baris data yang ada dengan nilai TTL khusus.
- Kolom - Anda dapat memperbarui subset kolom dalam baris yang ada dengan nilai TTL kustom.

Mengatur nilai TTL untuk baris dan kolom lebih diutamakan daripada pengaturan TTL default untuk tabel.

Untuk mengubah pengaturan TTL kolom 'subjek' yang disisipkan sebelumnya dari 259.200 detik (3 hari) menjadi 86.400 detik (satu hari), gunakan pernyataan berikut.

```
UPDATE my_table USING TTL 86400 set subject = 'Updated Message' WHERE userid =
B79CB3BA-745E-5D9A-8903-4A02327A7E09 and time = 96a29100-5e25-11ec-90d7-b5d91eceda0a;
```

Anda dapat menjalankan kueri pilih sederhana untuk melihat catatan yang diperbarui sebelum waktu kedaluwarsa.

```
SELECT * from my_table;
```

Kueri menunjukkan output berikut.

| userid                               | subject         | user            | time                                 | body  |
|--------------------------------------|-----------------|-----------------|--------------------------------------|-------|
| b79cb3ba-745e-5d9a-8903-4a02327a7e09 | Updated Message | 205.212.123.123 | 96a29100-5e25-11ec-90d7-b5d91eceda0a | Hello |



```
50554d6e-29bb-11e5-b345-feff819cdc9f | cf03fb21-59b5-11ec-b371-dff626ab9620 | Hello |
Message | 205.212.123.123
```

Untuk mengonfirmasi bahwa kedaluwarsa berhasil, jalankan kueri yang sama lagi setelah waktu kedaluwarsa yang dikonfigurasi.

```
SELECT * from my_table;
```

Kueri menunjukkan output berikut setelah kolom 'subjek' kedaluwarsa.

```
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
```

## Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

| Dokumentasi SDK                          | Contoh kode                                          |
|------------------------------------------|------------------------------------------------------|
| <a href="#">AWS SDK untuk C++</a>        | <a href="#">AWS SDK untuk C++ contoh kode</a>        |
| <a href="#">AWS CLI</a>                  | <a href="#">AWS CLI contoh kode</a>                  |
| <a href="#">AWS SDK untuk Go</a>         | <a href="#">AWS SDK untuk Go contoh kode</a>         |
| <a href="#">AWS SDK untuk Java</a>       | <a href="#">AWS SDK untuk Java contoh kode</a>       |
| <a href="#">AWS SDK untuk JavaScript</a> | <a href="#">AWS SDK untuk JavaScript contoh kode</a> |
| <a href="#">AWS SDK untuk Kotlin</a>     | <a href="#">AWS SDK untuk Kotlin contoh kode</a>     |
| <a href="#">AWS SDK untuk .NET</a>       | <a href="#">AWS SDK untuk .NET contoh kode</a>       |

| Dokumentasi SDK                              | Contoh kode                                              |
|----------------------------------------------|----------------------------------------------------------|
| <a href="#">AWS SDK untuk PHP</a>            | <a href="#">AWS SDK untuk PHP contoh kode</a>            |
| <a href="#">Alat AWS untuk PowerShell</a>    | <a href="#">Alat AWS untuk PowerShell contoh kode</a>    |
| <a href="#">AWS SDK untuk Python (Boto3)</a> | <a href="#">AWS SDK untuk Python (Boto3) contoh kode</a> |
| <a href="#">AWS SDK untuk Ruby</a>           | <a href="#">AWS SDK untuk Ruby contoh kode</a>           |
| <a href="#">AWS SDK for Rust</a>             | <a href="#">AWS SDK for Rust contoh kode</a>             |
| <a href="#">AWS SDK for SAP ABAP</a>         | <a href="#">AWS SDK for SAP ABAP contoh kode</a>         |
| <a href="#">AWS SDK for Swift</a>            | <a href="#">AWS SDK for Swift contoh kode</a>            |

#### Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

## Bekerja dengan tag dan label untuk sumber daya Amazon Keyspaces

Anda dapat memberi label sumber daya Amazon Keyspaces (untuk Apache Cassandra) menggunakan tag. Tag memungkinkan Anda mengkategorikan sumber daya Anda dengan cara yang berbeda—misalnya, berdasarkan tujuan, pemilik, lingkungan, atau kriteria lainnya. Tanda membantu Anda melakukan hal berikut:

- Identifikasi sumber daya dengan cepat berdasarkan tanda yang Anda tetapkan padanya.
- Lihat AWS tagihan yang dipecah berdasarkan tag.
- Kontrol akses ke sumber daya Amazon Keyspaces berdasarkan tag. Untuk contoh kebijakan IAM menggunakan tag, lihat [the section called “Otorisasi berdasarkan tag Amazon Keyspaces”](#).

Penandaan didukung oleh AWS layanan seperti Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Keyspaces, dan banyak lagi. Penandaan

yang efisien dapat memberikan wawasan biaya dengan memungkinkan Anda membuat laporan di seluruh layanan yang membawa tanda tertentu.

Untuk memulai penandaan, lakukan hal berikut:

1. Pahami [Pembatasan untuk menggunakan tag untuk memberi label sumber daya di Amazon Keyspaces](#).
2. Buat tanda menggunakan [Tandai ruang kunci, tabel, dan aliran di Amazon Keyspaces](#).
3. Gunakan [Membuat laporan alokasi biaya menggunakan tag untuk Amazon Keyspaces](#) untuk melacak AWS biaya Anda per tag aktif.

Terakhir, merupakan praktik yang baik untuk mengikuti strategi penandaan yang optimal. Untuk informasi selengkapnya, lihat [Strategi penandaan AWS](#).

## Pembatasan untuk menggunakan tag untuk memberi label sumber daya di Amazon Keyspaces

Setiap tanda terdiri dari kunci dan nilai, yang keduanya Anda tentukan. Pembatasan berikut berlaku:

- Setiap ruang kunci, tabel, atau aliran Amazon Keyspaces hanya dapat memiliki satu tag dengan kunci yang sama. Jika Anda mencoba menambahkan tanda yang ada (kunci yang sama), nilai tanda yang ada akan diperbarui ke nilai baru.
- Tag yang diterapkan ke ruang kunci tidak secara otomatis berlaku untuk tabel di dalam ruang kunci itu. Untuk menerapkan tag yang sama ke ruang kunci dan semua tabelnya, setiap sumber daya harus ditandai secara individual.
- Tag yang diterapkan ke tabel tidak secara otomatis berlaku untuk aliran tabel itu. Untuk menerapkan tag yang sama ke tabel dan aliran selama pembuatan tabel, Anda dapat menggunakan `PropagateTagsOnEnable` bendera saat membuat tabel. Menggunakan flag ini, Amazon Keyspaces menerapkan tag tabel ke aliran selama pembuatan stream. Saat aliran aktif, perubahan pada tag tabel tidak berlaku untuk aliran.
- Saat Anda membuat ruang kunci atau tabel Multi-region, tag apa pun yang Anda tentukan selama proses pembuatan secara otomatis diterapkan ke semua ruang kunci dan tabel di semua Wilayah. Saat Anda mengubah tag yang ada menggunakan `ALTER KEYSPACE` or `ALTER TABLE`, pembaruan hanya diterapkan ke ruang kunci atau tabel di Wilayah tempat Anda melakukan perubahan.

- Nilai bertindak sebagai deskriptor dalam kategori tanda (kunci). Di Amazon Keyspaces nilainya tidak bisa kosong atau null.
- Kunci dan nilai tanda peka huruf besar-kecil.
- Panjang kunci maksimum adalah 128 karakter Unicode.
- Panjang nilai maksimum adalah 256 karakter Unicode.
- Karakter yang diperbolehkan adalah huruf, spasi kosong, dan angka, ditambah karakter khusus berikut: + - = . \_ : /
- Jumlah maksimum tanda per sumber daya adalah 50.
- AWS-Assigned nama tag dan nilai secara otomatis diberi aws : awalan, yang tidak dapat Anda tetapkan. AWS-nama-nama tag yang ditetapkan tidak dihitung terhadap batas tag 50. User-assigned nama tag memiliki awalan user : dalam laporan alokasi biaya.
- Anda tidak dapat mengubah penerapan tanda ke versi sebelumnya.

## Tandai ruang kunci, tabel, dan aliran di Amazon Keyspaces

Anda dapat menambahkan, membuat daftar, mengedit, atau menghapus tag untuk ruang kunci, tabel, dan aliran menggunakan konsol Amazon Keyspaces, the AWS CLI, atau Cassandra Query Language (CQL). Anda kemudian dapat mengaktifkan tag yang ditentukan pengguna ini sehingga muncul di AWS Manajemen Penagihan dan Biaya konsol untuk pelacakan alokasi biaya. Untuk informasi selengkapnya, lihat [Membuat laporan alokasi biaya menggunakan tag untuk Amazon Keyspaces](#).

Untuk pengeditan massal, Anda juga dapat menggunakan Editor Tag di konsol. Untuk informasi selengkapnya, lihat [Bekerja dengan Tag Editor](#) dalam Panduan Pengguna AWS Resource Groups .

Untuk informasi selengkapnya tentang struktur tag, lihat [Pembatasan untuk menggunakan tag untuk memberi label sumber daya di Amazon Keyspaces](#).

### Topik

- [Tambahkan tag saat membuat ruang kunci baru](#)
- [Tambahkan tag ke ruang kunci](#)
- [Hapus tag dari ruang kunci](#)
- [Melihat tag dari keyspace](#)
- [Tambahkan tag saat membuat tabel baru](#)
- [Tambahkan tag ke tabel](#)

- [Hapus tag dari tabel](#)
- [Lihat tag tabel](#)
- [Tambahkan tag ke aliran baru saat membuat tabel](#)
- [Tambahkan tag ke aliran baru untuk tabel yang ada](#)
- [Tambahkan tag baru ke aliran](#)
- [Hapus tag dari aliran](#)
- [Lihat tag aliran](#)

## Tambahkan tag saat membuat ruang kunci baru

Anda dapat menggunakan konsol Amazon Keyspaces, CQL atau AWS CLI untuk menambahkan tag saat Anda membuat ruang kunci baru.

### Console

Setel tag saat membuat ruang kunci baru menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces, lalu pilih Create keyspace.
3. Pada halaman Create keyspace, berikan nama untuk keyspace.
4. Di bawah Tag pilih Tambahkan tag baru dan masukkan kunci dan nilai.
5. Pilih Buat ruang kunci.

### Cassandra Query Language (CQL)

Tetapkan tag saat membuat ruang kunci baru menggunakan CQL

- Contoh berikut membuat keyspace baru dengan tag.

```
CREATE KEYSPACE mykeyspace WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

## CLI

Tetapkan tag saat membuat ruang kunci baru menggunakan AWS CLI

- Pernyataan berikut membuat keyspace baru dengan tag.

```
aws keyspaces create-keyspace --keyspace-name 'myKeyspace' --tags
'key=key1,value=val1' 'key=key2,value=val2'
```

## Tambahkan tag ke ruang kunci

Contoh berikut menunjukkan cara menambahkan tag ke ruang kunci di Amazon Keyspaces.

### Console

Tambahkan tag ke ruang kunci yang ada menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces.
3. Pilih keyspace dari daftar. Kemudian pilih tab Tag di mana Anda dapat melihat tag ruang kunci.
4. Pilih Kelola tag untuk menambah, mengedit, atau menghapus tag.
5. Pilih Simpan perubahan.

### Cassandra Query Language (CQL)

Tambahkan tag ke ruang kunci yang ada menggunakan CQL

- ```
ALTER KEYSPACE mykeyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

CLI

Tambahkan tag ke ruang kunci yang ada menggunakan AWS CLI

- Contoh berikut menunjukkan cara menambahkan tag baru ke ruang kunci yang ada.

```
aws keyspaces tag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

Hapus tag dari ruang kunci

Console

Menghapus tag dari ruang kunci yang ada menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces.
3. Pilih keyspace dari daftar. Kemudian pilih tab Tag di mana Anda dapat melihat tag ruang kunci.
4. Pilih Kelola tag dan hapus tag yang tidak Anda butuhkan lagi.
5. Pilih Simpan perubahan.

Cassandra Query Language (CQL)

Menghapus tag dari ruang kunci yang ada menggunakan CQL

- ```
ALTER KEYSPACE mykeyspace DROP TAGS {'key1':'val1', 'key2':'val2'};
```

### CLI

Menghapus tag dari ruang kunci yang ada menggunakan AWS CLI

- Pernyataan berikut menghapus tag tertentu dari keyspace.

```
aws keyspaces untag-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/' --tags 'key=key3,value=val3' 'key=key4,value=val4'
```

## Melihat tag dari keyspace

Contoh berikut menunjukkan cara membaca tag menggunakan konsol, CQL atau. AWS CLI

### Console

Melihat tag ruang kunci menggunakan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Keyspaces.
3. Pilih keyspace dari daftar. Kemudian pilih tab Tag di mana Anda dapat melihat tag ruang kunci.

### Cassandra Query Language (CQL)

Melihat tag keyspace menggunakan CQL

Untuk membaca tag yang dilampirkan ke keyspace, gunakan pernyataan CQL berikut.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

WHERE klausul diperlukan, dan harus menggunakan salah satu format berikut:

- `keyspace_name = 'mykeyspace' AND resource_type = 'keyspace'`
- `resource_id = arn`
- Pernyataan berikut menunjukkan apakah keyspace memiliki tag.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
resource_type = 'keyspace';
```

Output dari query terlihat seperti berikut ini.

```
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

Melihat tag dari keyspace menggunakan AWS CLI

- Contoh ini menunjukkan cara membuat daftar tag dari sumber daya yang ditentukan.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-east-1:111122223333:/keyspace/myKeyspace/'
```

Output dari perintah terakhir terlihat seperti ini.

```
{
 "tags": [
 {
 "key": "key1",
 "value": "val1"
 },
 {
 "key": "key2",
 "value": "val2"
 },
 {
 "key": "key3",
 "value": "val3"
 },
 {
 "key": "key4",
 "value": "val4"
 }
]
}
```

## Tambahkan tag saat membuat tabel baru

Anda dapat menggunakan konsol Amazon Keyspaces, CQL atau AWS CLI untuk menambahkan tag ke tabel baru saat Anda membuatnya.

## Console

Tambahkan tag saat membuat tabel baru menggunakan (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih Buat tabel.
3. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel.
4. Di bagian Skema, buat skema untuk tabel Anda.
5. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
6. Lanjutkan ke tag Tabel — bagian opsional, dan pilih Tambahkan tag baru untuk membuat tag baru.
7. Pilih Buat tabel.

## Cassandra Query Language (CQL)

Tambahkan tag saat membuat tabel baru menggunakan CQL

- Contoh berikut membuat tabel baru dengan tag.

```
CREATE TABLE mytable(...) WITH TAGS = {'key1':'val1', 'key2':'val2'};
```

## CLI

Tambahkan tag saat membuat tabel baru menggunakan AWS CLI

- Contoh berikut menunjukkan cara membuat tabel baru dengan tag. Perintah membuat tabel MyTable di MyKeySpace keyspace yang sudah ada. Perhatikan bahwa perintah telah dipecah menjadi baris yang berbeda untuk membantu keterbacaan.

```
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'
```

## Tambahkan tag ke tabel

Anda dapat menambahkan tag ke tabel yang ada di Amazon Keyspaces menggunakan konsol, CQL, atau. AWS CLI

### Console

Tambahkan tag ke tabel menggunakan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel.
3. Pilih tabel dari daftar dan pilih tab Tag.
4. Pilih Kelola tag untuk menambahkan tag ke tabel.
5. Pilih Simpan perubahan.

### Cassandra Query Language (CQL)

Tambahkan tag ke tabel menggunakan CQL

- Pernyataan berikut menunjukkan cara menambahkan tag ke tabel yang ada.

```
ALTER TABLE mykeyspace.mytable ADD TAGS {'key1':'val1', 'key2':'val2'};
```

### CLI

Tambahkan tag ke tabel menggunakan AWS CLI

- Contoh berikut menunjukkan cara menambahkan tag baru ke tabel yang ada.

```
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'
```

## Hapus tag dari tabel

### Console

Hapus tag dari tabel menggunakan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel.
3. Pilih tabel dari daftar dan pilih tab Tag.
4. Pilih Kelola tag untuk menghapus tag dari tabel.
5. Pilih Simpan perubahan.

### Cassandra Query Language (CQL)

Hapus tag dari tabel menggunakan CQL

- Pernyataan berikut menunjukkan cara menghapus tag dari tabel yang ada.

```
ALTER TABLE mytable DROP TAGS {'key3':'val3', 'key4':'val4'};
```

### CLI

Tambahkan tag ke tabel menggunakan AWS CLI

- Pernyataan berikut menghapus tag tertentu dari keyspace.

```
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'
```

## Lihat tag tabel

Contoh berikut menunjukkan cara melihat tag tabel di Amazon Keyspaces menggunakan konsol, CQL, atau. AWS CLI

## Console

Melihat tag tabel menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel.
3. Pilih tabel dari daftar dan pilih tab Tag.

## Cassandra Query Language (CQL)

Lihat tag tabel menggunakan CQL

Untuk membaca tag yang dilampirkan ke tabel, gunakan pernyataan CQL berikut.

```
SELECT * FROM system_schema_mcs.tags WHERE valid_where_clause;
```

WHERE Klausul diperlukan, dan harus menggunakan salah satu format berikut:

- `keyspace_name = 'mykeyspace' AND resource_name = 'mytable'`
- `resource_id = arn`
- Query berikut mengembalikan tag dari tabel yang ditentukan.

```
SELECT * FROM system_schema_mcs.tags WHERE keyspace_name = 'mykeyspace' AND
resource_name = 'mytable';
```

Output dari kueri itu terlihat seperti berikut ini.

```
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

### Lihat tag tabel menggunakan AWS CLI

- Contoh ini menunjukkan cara membuat daftar tag dari sumber daya yang ditentukan.

```
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'
```

Output dari perintah terakhir terlihat seperti ini.

```
{
 "tags": [
 {
 "key": "key1",
 "value": "val1"
 },
 {
 "key": "key2",
 "value": "val2"
 },
 {
 "key": "key3",
 "value": "val3"
 },
 {
 "key": "key4",
 "value": "val4"
 }
]
}
```

### Tambahkan tag ke aliran baru saat membuat tabel

Anda dapat menambahkan tag ke aliran saat Anda membuat tabel baru dengan aliran menggunakan CQL atau AWS CLI untuk menandai aliran.

**Note**

Amazon Keyspaces CDC memerlukan keberadaan peran terkait layanan (AWSServiceRoleForAmazonKeyspacesCDC) yang menerbitkan data metrik dari aliran CDC Amazon Keyspaces ke dalam akun Anda atas nama Anda. "cloudwatch:namespace": "AWS/Cassandra" CloudWatch Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Aliran CDC”](#).

## Cassandra Query Language (CQL)

Tambahkan tag ke aliran saat membuat tabel baru menggunakan CQL

1. Untuk membuat tabel baru dengan aliran dan menerapkan tag tabel secara otomatis ke aliran, Anda dapat menggunakan 'propagate\_tags': 'TABLE' bendera. Pernyataan berikut adalah contoh dari ini.

```
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. Untuk menerapkan tag baru ke aliran, Anda dapat menggunakan contoh berikut.

```
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

Tambahkan tag ke aliran saat membuat tabel baru menggunakan AWS CLI

1. Untuk membuat tabel dengan aliran dan menerapkan tag tabel secara otomatis ke aliran, Anda dapat menggunakan `propagateTags=Table` bendera. Kode berikut adalah contoh dari ini.

```
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. Untuk menerapkan tag yang berbeda ke aliran, Anda dapat menggunakan contoh berikut.

```
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}]'
```

## Tambahkan tag ke aliran baru untuk tabel yang ada

Anda dapat menambahkan tag saat membuat aliran baru untuk tabel yang ada. Anda dapat menggunakan `PropagateTags` bendera untuk menerapkan tag tabel ke aliran atau menentukan tag baru untuk aliran. Anda dapat menggunakan CQL atau AWS CLI untuk menandai aliran baru.

### Note

Amazon Keyspaces CDC memerlukan keberadaan peran terkait layanan (`AWSServiceRoleForAmazonKeyspacesCDC`) yang menerbitkan data metrik dari aliran CDC Amazon Keyspaces ke dalam akun Anda atas nama Anda.



"cloudwatch:namespace": "AWS/Cassandra" CloudWatch Peran ini dibuat secara otomatis untuk Anda. Untuk informasi selengkapnya, lihat [the section called “Aliran CDC”](#).

## Console

Tambahkan tag saat membuat aliran baru menggunakan (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih tabel yang ingin Anda tambahkan alirannya.
3. Pilih tab Streams.
4. Di bagian Streaming details, pilih Edit.
5. Pilih Aktifkan aliran.
6. Pilih jenis Tampilan dan lanjutkan ke Tag untuk membuat tag untuk aliran.
7. Anda dapat memilih salah satu opsi berikut:
  - Tidak ada tag - Gunakan opsi ini jika Anda tidak ingin membuat tag apa pun untuk aliran.
  - Salin tag dari tabel - Gunakan opsi ini jika Anda ingin menyalin tag dari tabel ke aliran. Setelah menyalin tag, Anda dapat mengeditnya untuk streaming. Perhatikan bahwa opsi ini hanya tersedia jika tabel memiliki tag.
  - Tambahkan tag baru - Anda dapat menambahkan hingga 50 tag untuk aliran dengan memilih Tambahkan tag baru.
8. Pilih Simpan perubahan.

## Cassandra Query Language (CQL)

Tambahkan tag saat membuat aliran baru

1. Untuk membuat aliran baru untuk tabel yang ada dan menerapkan tag tabel ke aliran, Anda dapat menggunakan 'propagate\_tags': 'TABLE' bendera. Pernyataan berikut adalah contohnya.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':
{ 'view_type': 'NEW_IMAGE', 'propagate_tags': 'TABLE' } };
```

2. Untuk membuat aliran baru untuk tabel yang ada dan menentukan tag baru, Anda dapat menggunakan contoh berikut.

```
ALTER TABLE mytable WITH cdc = TRUE AND CUSTOM_PROPERTIES={ 'cdc_specification':
 { 'view_type': 'NEW_IMAGE', 'tags': { 'key': 'string', 'value': 'string' }} };
```

## CLI

Tambahkan tag saat membuat aliran baru menggunakan AWS CLI

1. Untuk membuat aliran baru dengan tag, Anda dapat menggunakan `propagateTags=TABLE` bendera untuk menerapkan tag tabel secara otomatis ke aliran. Kode berikut adalah contoh dari ini.

```
aws keyspaces update-table \
--keyspace-name 'my_keyspace' \
--table-name 'my_table' \
--cdc-specification propagateTags=TABLE,status=ENABLED,viewType=NEW_IMAGE
```

2. Untuk membuat aliran baru untuk tabel yang ada dan menentukan tag baru, Anda dapat menggunakan contoh berikut.

```
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}]'
```

## Tambahkan tag baru ke aliran

Anda dapat menambahkan tag baru ke aliran yang ada di Amazon Keyspaces menggunakan CQL atau file. AWS CLI Anda hanya dapat menambahkan tag ke aliran terbaru.

## Console

Tambahkan tag ke aliran yang ada (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>

2. Di panel navigasi, pilih Tabel, lalu pilih tabel dengan aliran yang ingin Anda tag.
3. Pilih tab Streams.
4. Di bagian Tag, pilih Kelola tag.
5. Pilih Tambahkan tag baru untuk menambahkan tag baru. Anda dapat membuat hingga 50 tag dengan mengulangi langkah ini.
6. Pilih Simpan perubahan.

## Cassandra Query Language (CQL)

Tambahkan tag ke aliran menggunakan CQL

- Pernyataan berikut menunjukkan cara menambahkan tag ke aliran yang ada.

```
ALTER TABLE mykeyspace.mytable ADD TAGS_FOR_CDC {'key1':'val1', 'key2':'val2'};
```

## CLI

Tambahkan tag ke aliran menggunakan AWS CLI

- Contoh berikut menunjukkan cara menambahkan tag baru ke aliran yang ada.

```
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'
```

## Hapus tag dari aliran

Untuk menghapus tag dari aliran, Anda dapat menggunakan CQL atau file. AWS CLI Anda hanya dapat menghapus tag untuk aliran terbaru.

## Console

Hapus tag dari tabel menggunakan konsol Amazon Keyspaces

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>

2. Di panel navigasi, pilih Tabel.
3. Pilih tabel dari daftar dan pilih tab Streams.
4. Di bagian Tag pilih Kelola tag untuk menghapus tag dari tabel.
5. Setelah tag yang ingin Anda hapus, pilih Hapus.
6. Pilih Simpan perubahan.

## Cassandra Query Language (CQL)

Hapus tag dari aliran menggunakan CQL

- Pernyataan berikut menunjukkan cara menghapus tag dari aliran yang ada.

```
ALTER TABLE mytable DROP TAGS_FOR_CDC {'key3':'val3', 'key4':'val4'};
```

## CLI

Hapus tag dari aliran menggunakan AWS CLI

- Pernyataan berikut menghapus tag yang ditentukan dari aliran.

```
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'
```

## Lihat tag aliran

Contoh berikut menunjukkan cara melihat tag aliran di Amazon Keyspaces menggunakan CQL atau file. AWS CLI

### Console

Melihat tag aliran menggunakan konsol

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel.

3. Pilih tabel dari daftar dan pilih tab Streams.
4. Anda dapat melihat tag aliran di bagian Tag.

## Cassandra Query Language (CQL)

## Lihat tag aliran menggunakan CQL

Untuk membaca tag yang dilampirkan ke aliran, Anda harus menentukan ARN sumber daya aliran dalam klausa. WHERE Sintaks CQL berikut adalah contoh dari ini.

```
SELECT * FROM system_schema_mcs.tags WHERE resource_id = stream_arn;
```

- Query berikut mengembalikan tag untuk aliran tertentu.

```
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';
```

Output dari kueri itu terlihat seperti berikut ini.

```

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

## Lihat tag aliran menggunakan AWS CLI

- Contoh ini menunjukkan cara membuat daftar tag untuk semua aliran di bawah ruang kunci yang ditentukan.

```
aws keyspaces list-tags-for-resource --resource-arn 'arn:aws:cassandra:us-east-1:11112223333:/keyspace/my_keyspace/table/my_table/stream/2025-05-11T21:21:33.291'
```

Output dari perintah terakhir terlihat seperti ini.

```
{
 "tags": [
 {
 "key": "key1",
 "value": "val1"
 },
 {
 "key": "key2",
 "value": "val2"
 },
 {
 "key": "key3",
 "value": "val3"
 },
 {
 "key": "key4",
 "value": "val4"
 }
]
}
```

## Membuat laporan alokasi biaya menggunakan tag untuk Amazon Keyspaces

AWS menggunakan tag untuk mengatur biaya sumber daya pada laporan alokasi biaya Anda. AWS menyediakan dua jenis tag alokasi biaya:

- Tag AWS yang dihasilkan. AWS mendefinisikan, membuat, dan menerapkan tag ini untuk Anda.
- User-defined tag. Anda menentukan, membuat, dan menerapkan tanda ini.

Anda harus mengaktifkan kedua jenis tanda secara terpisah sebelum tanda tersebut muncul di Cost Explorer atau laporan alokasi biaya.

Untuk mengaktifkan tag AWS yang dihasilkan:

1. Masuk ke Konsol Manajemen AWS dan buka konsol Billing and Cost Management <https://console.aws.amazon.com/billing/homedi> #/.
2. Di panel navigasi, pilih Tanda Alokasi Biaya.
3. Di bagian Tanda Alokasi Biaya yang Dihasilkan AWS, pilih Aktifkan.

Untuk mengaktifkan tanda yang ditentukan pengguna:

1. Masuk ke Konsol Manajemen AWS dan buka konsol Billing and Cost Management <https://console.aws.amazon.com/billing/homedi> #/.
2. Di panel navigasi, pilih Tanda Alokasi Biaya.
3. Di bawah Tag Alokasi User-Defined Biaya, pilih Aktifkan.

Setelah Anda membuat dan mengaktifkan tag, AWS buat laporan alokasi biaya dengan penggunaan dan biaya yang dikelompokkan berdasarkan tag aktif Anda. Laporan alokasi biaya mencakup semua AWS biaya Anda untuk setiap periode penagihan. Laporan ini mencakup sumber daya yang diberi tanda dan tidak diberi tanda, sehingga Anda dapat mengatur biaya untuk sumber daya dengan jelas.

#### Note

Saat ini, data apa pun yang ditransfer keluar dari Amazon Keyspaces tidak akan dipecah berdasarkan tag pada laporan alokasi biaya.

Untuk informasi selengkapnya, lihat [Menggunakan tanda alokasi biaya](#).

## Buat sumber daya Amazon Keyspaces dengan AWS CloudFormation

Amazon Keyspaces terintegrasi dengan AWS CloudFormation, layanan yang membantu Anda memodelkan dan mengatur AWS ruang kunci dan tabel sehingga Anda dapat menghabiskan lebih sedikit waktu untuk membuat dan mengelola sumber daya dan infrastruktur Anda. Anda membuat templat yang menjelaskan ruang kunci dan tabel yang Anda inginkan, serta CloudFormation menangani penyediaan dan konfigurasi sumber daya tersebut untuk Anda.

Bila Anda menggunakan CloudFormation, Anda dapat menggunakan kembali template Anda untuk mengatur sumber daya Amazon Keyspaces Anda secara konsisten dan berulang kali. Cukup jelaskan sumber daya Anda sekali, lalu sediakan sumber daya yang sama berulang-ulang di beberapa Akun AWS dan Wilayah.

## Amazon Keyspaces dan template CloudFormation

[Untuk menyediakan dan mengonfigurasi sumber daya untuk Amazon Keyspaces, Anda harus memahami CloudFormation template.](#) Templat adalah file teks dengan format JSON atau YAML. Template ini menjelaskan sumber daya yang ingin Anda sediakan di CloudFormation tumpukan Anda. Jika Anda tidak terbiasa dengan JSON atau YAMAL, Anda dapat menggunakan CloudFormation Designer untuk membantu Anda memulai dengan template. CloudFormation Untuk informasi lebih lanjut, lihat [Apa itu CloudFormation desainer?](#) dalam AWS CloudFormation User Guide.

Amazon Keyspaces mendukung pembuatan keyspaces dan tabel di. CloudFormation Untuk tabel yang Anda buat menggunakan CloudFormation templat, Anda dapat menentukan skema, read/write mode, pengaturan throughput yang disediakan, dan fitur lain yang didukung. Untuk informasi selengkapnya, termasuk contoh template JSON dan YAMAL untuk ruang kunci dan tabel, lihat referensi jenis sumber daya Amazon [Keyspaces \(untuk Apache Cassandra\)](#) di Referensi Template.AWS CloudFormation

## Pelajari lebih lanjut tentang CloudFormation

Untuk mempelajari selengkapnya CloudFormation, lihat sumber daya berikut:

- [AWS CloudFormation](#)
- [AWS CloudFormation Panduan Pengguna](#)
- [AWS CloudFormation antarmuka baris perintah Panduan Pengguna](#)

## Menggunakan NoSQL Workbench dengan Amazon Keyspaces (untuk Apache Cassandra)

NoSQL Workbench adalah aplikasi sisi klien yang membantu Anda merancang dan memvisualisasikan model data nonrelasional untuk Amazon Keyspaces dengan lebih mudah. Klien NoSQL Workbench tersedia untuk Windows, macOS, dan Linux.



## Merancang model data dan membuat sumber daya secara otomatis

NoSQL Workbench menyediakan antarmuka point-and-click untuk merancang dan membuat model data Amazon Keyspaces. Anda dapat dengan mudah membuat model data baru dari awal dengan mendefinisikan ruang kunci, tabel, dan kolom. Anda juga dapat mengimpor model data yang ada dan membuat modifikasi (seperti menambahkan, mengedit, atau menghapus kolom) untuk menyesuaikan model data untuk aplikasi baru. NoSQL Workbench kemudian memungkinkan Anda untuk mengkomit model data ke Amazon Keyspaces atau Apache Cassandra, dan membuat keyspaces dan tabel secara otomatis. Untuk mempelajari cara membuat model data, lihat [the section called “Buat model data”](#) dan [the section called “Mengedit model data”](#).

### Memvisualisasikan model data

Menggunakan NoSQL Workbench, Anda dapat memvisualisasikan model data Anda untuk membantu memastikan bahwa model data dapat mendukung kueri dan pola akses aplikasi Anda. Anda juga dapat menyimpan dan mengekspor model data Anda dalam berbagai format untuk kolaborasi, dokumentasi, dan presentasi. Untuk informasi selengkapnya, lihat [the section called “Visualisasikan model data”](#).

### Topik

- [Unduh NoSQL Workbench](#)
- [Memulai dengan NoSQL Workbench](#)
- [Visualisasikan model data dengan NoSQL Workbench](#)
- [Buat model data baru dengan NoSQL Workbench](#)
- [Edit model data yang ada dengan NoSQL Workbench](#)
- [Cara mengkomit model data ke Amazon Keyspaces dan Apache Cassandra](#)
- [Contoh model data di NoSQL Workbench](#)
- [Riwayat rilis untuk NoSQL Workbench](#)

## Unduh NoSQL Workbench

Ikuti petunjuk ini untuk mengunduh dan menginstal NoSQL Workbench.

Untuk mengunduh dan menginstal NoSQL Workbench

1. Gunakan salah satu tautan berikut untuk mengunduh NoSQL Workbench secara gratis.

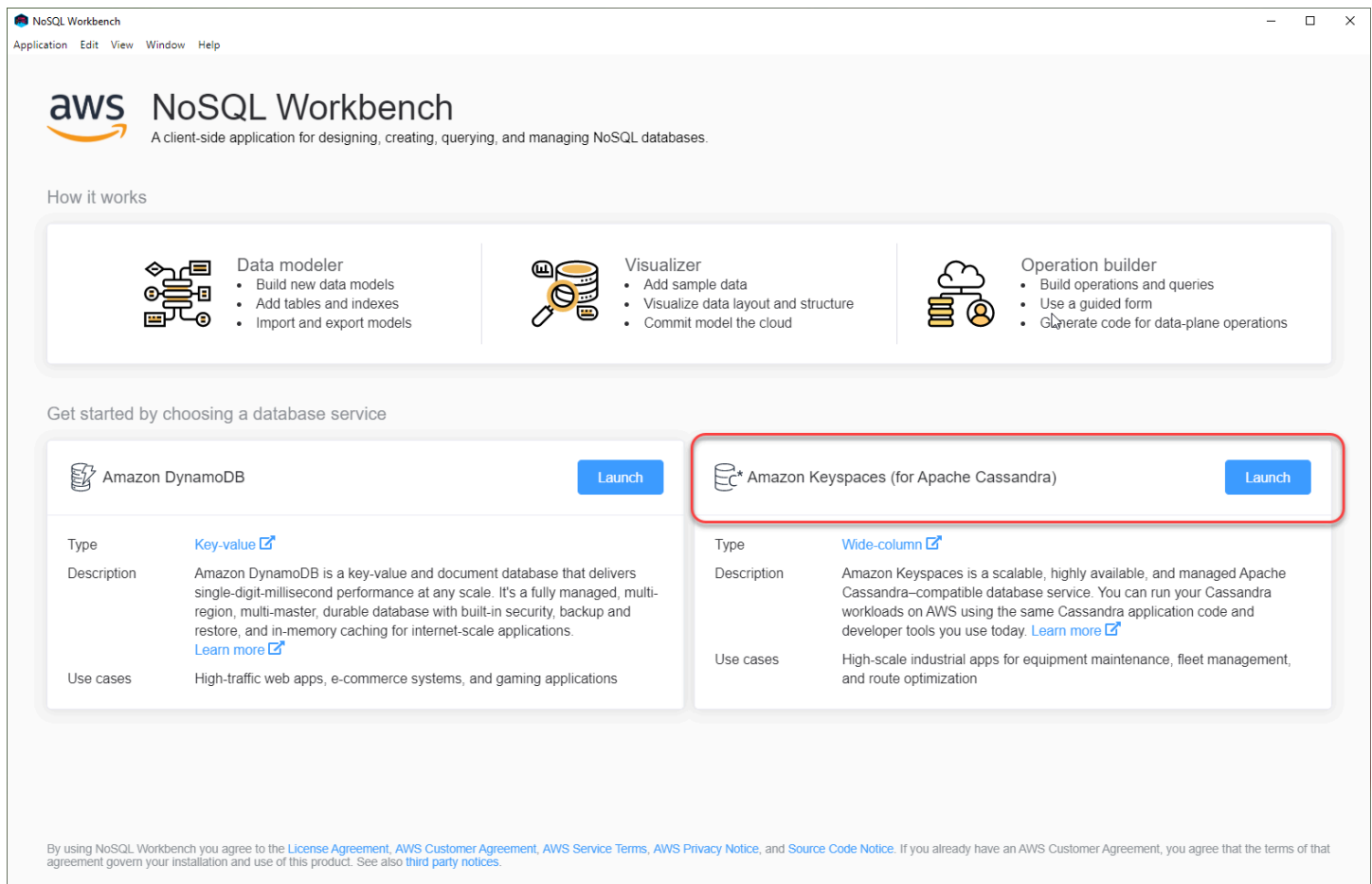
| Sistem Operasi | Tautan Unduhan                      |
|----------------|-------------------------------------|
| macOS          | <a href="#">Unduh untuk macOS</a>   |
| Linux*         | <a href="#">Unduh untuk Linux</a>   |
| Windows        | <a href="#">Unduh untuk Windows</a> |

\* NoSQL Workbench mendukung Ubuntu 12.04, Fedora 21, dan Debian 8 atau versi terbaru dari distribusi Linux ini.

- Setelah pengunduhan selesai, mulai aplikasi dan ikuti petunjuk di layar untuk menyelesaikan instalasi.

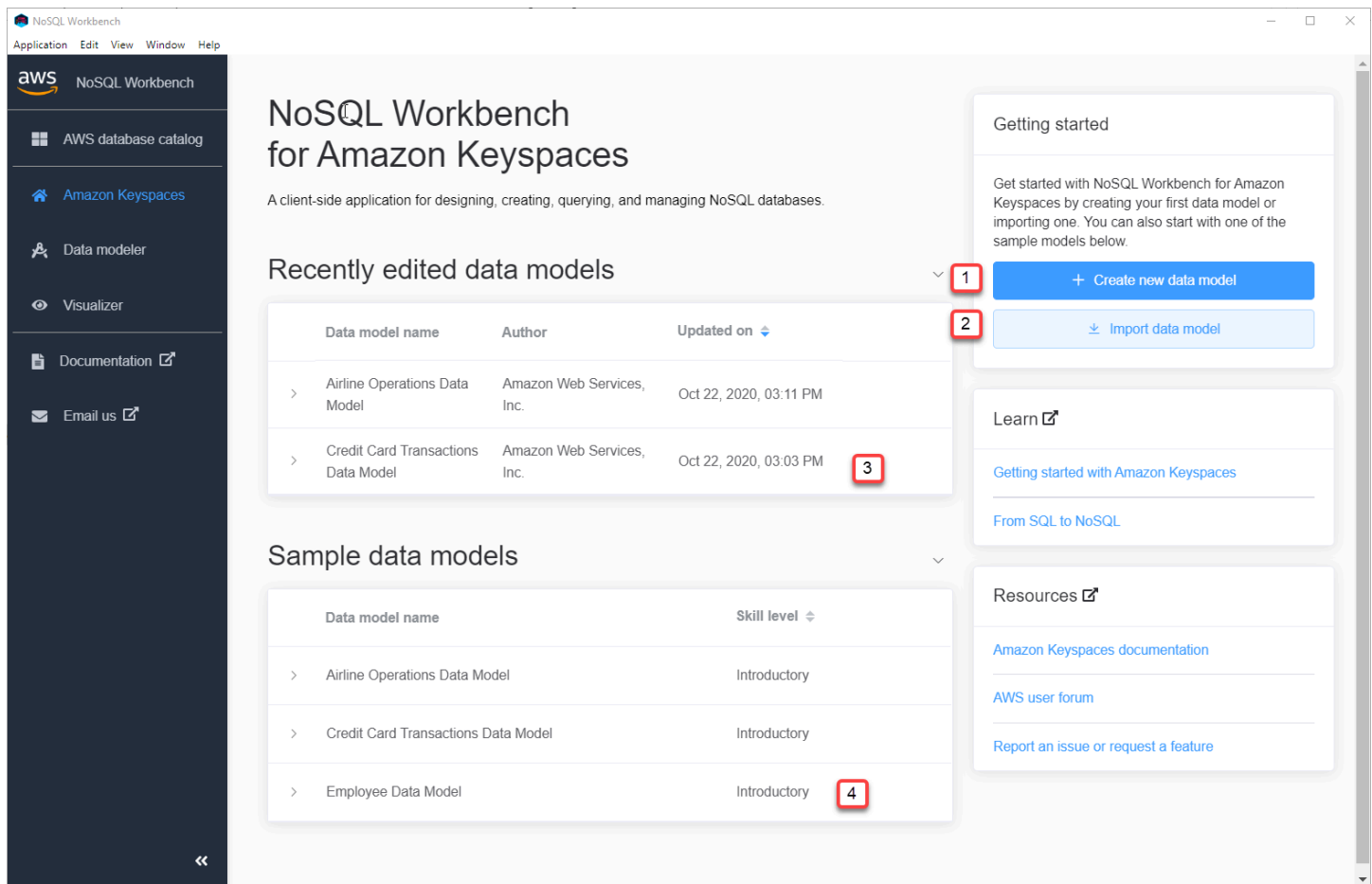
## Memulai dengan NoSQL Workbench

Untuk memulai dengan NoSQL Workbench, pada halaman Katalog Database di NoSQL Workbench, pilih Amazon Keyspaces, lalu pilih Launch.



Ini membuka halaman beranda NoSQL Workbench untuk Amazon Keyspaces di mana Anda memiliki opsi berikut untuk memulai:

1. Buat model data baru.
2. Impor model data yang ada dalam format JSON.
3. Buka model data yang baru saja diedit.
4. Buka salah satu model sampel yang tersedia.



Masing-masing opsi membuka pemodel data NoSQL Workbench. Untuk terus membuat model data baru, lihat [the section called “Buat model data”](#). Untuk mengedit model data yang ada, lihat [the section called “Mengedit model data”](#).

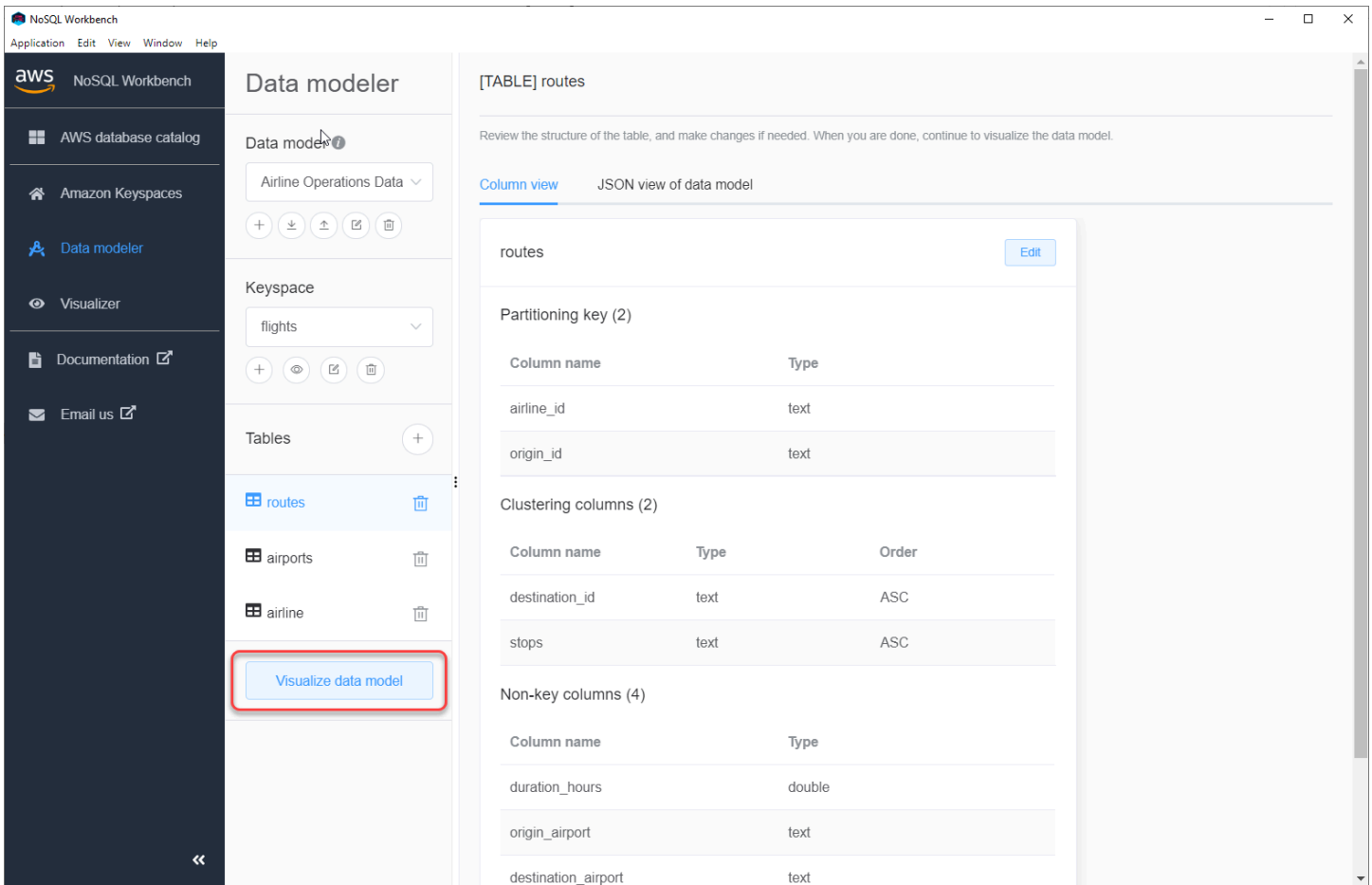
## Visualisasikan model data dengan NoSQL Workbench

Menggunakan NoSQL Workbench, Anda dapat memvisualisasikan model data Anda untuk membantu memastikan bahwa model data dapat mendukung kueri dan pola akses aplikasi Anda. Anda juga dapat menyimpan dan mengeksport model data Anda dalam berbagai format untuk kolaborasi, dokumentasi, dan presentasi.

Setelah Anda membuat model data baru atau mengedit model data yang ada, Anda dapat memvisualisasikan model tersebut.

## Memvisualisasikan model data dengan NoSQL Workbench

Ketika Anda telah menyelesaikan model data dalam pemodel data, pilih Visualisasikan model data.



The screenshot shows the AWS NoSQL Workbench interface. The left sidebar contains the following menu items: AWS database catalog, Amazon Keyspaces, Data modeler (selected), Visualizer, Documentation, and Email us. The main area is titled 'Data modeler' and shows the 'Airline Operations Data' dataset. The 'Keyspaces' dropdown is set to 'flights'. The 'Tables' list includes 'routes', 'airports', and 'airline'. The 'Visualize data model' button is highlighted with a red box. The right pane shows the 'routes' table structure with the following columns:

| Column name | Type |
|-------------|------|
| airline_id  | text |
| origin_id   | text |

Partitioning key (2)

| Column name    | Type | Order |
|----------------|------|-------|
| destination_id | text | ASC   |
| stops          | text | ASC   |

Clustering columns (2)

| Column name         | Type   |
|---------------------|--------|
| duration_hours      | double |
| origin_airport      | text   |
| destination_airport | text   |

Non-key columns (4)

Ini membawa Anda ke visualizer data di NoSQL Workbench. Visualizer data menyediakan representasi visual dari skema tabel dan memungkinkan Anda menambahkan data sampel. Untuk menambahkan data sampel ke tabel, pilih tabel dari model, lalu pilih Edit. Untuk menambahkan baris data baru, pilih Tambahkan baris baru di bagian bawah layar. Pilih Simpan ketika Anda selesai.

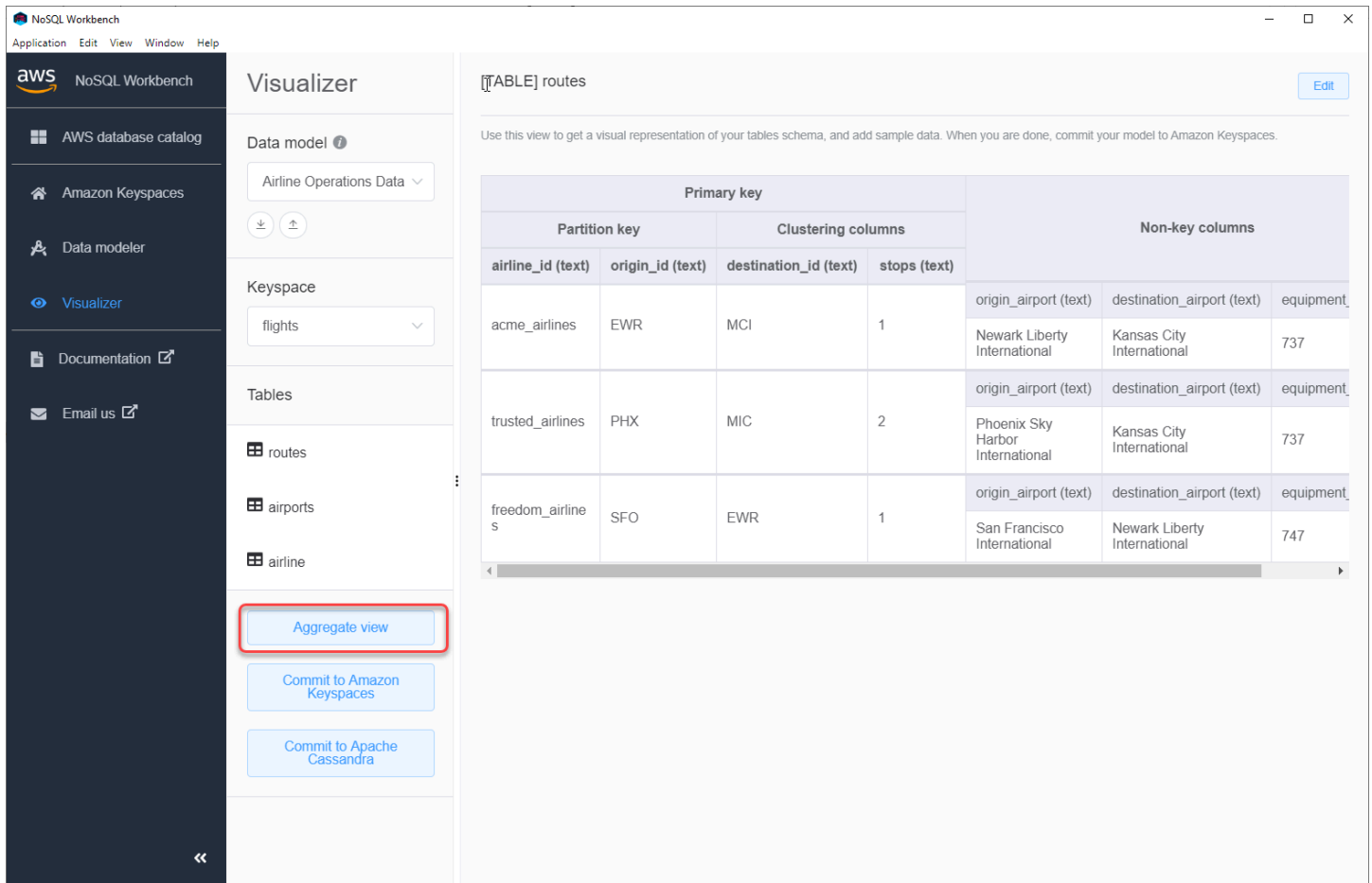
The screenshot shows the AWS NoSQL Workbench Visualizer interface. On the left is a sidebar with navigation options: AWS database catalog, Amazon Keyspaces, Data modeler, Visualizer (selected), Documentation, and Email us. The main area is titled 'Visualizer' and shows the 'Data model' for 'Airline Operations'. The 'Keyspace' is set to 'flights'. Under 'Tables', a list includes 'routes', 'airports', and 'airline'. The 'routes' table is selected, and its data is displayed in a table view. The table has 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). The data rows are:

| 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         |

At the bottom of the table, there is a button labeled '+ Add new row' which is highlighted with a red rectangle. Other buttons visible include 'Aggregate view', 'Commit to Amazon Keyspaces', and 'Commit to Apache Cassandra'.

## Tampilan agregat

Setelah Anda mengonfirmasi skema tabel, Anda dapat menggabungkan visualisasi model data.



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.

**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 |
| 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 interface also includes a sidebar with navigation options: AWS database catalog, Amazon Keyspaces, Data modeler, Visualizer (selected), Documentation, and Email us. The 'Aggregate view' button is highlighted with a red box.

Setelah Anda menggabungkan tampilan model data, Anda dapat mengekspor tampilan ke file PNG. Untuk mengekspor model data ke file JSON, pilih tanda unggah di bawah nama model data.

#### Note

Anda dapat mengekspor model data dalam format JSON kapan saja dalam proses desain.

Visualizer

Data model ⓘ

Airline Operations Data

KeySpace

flights

Tables

- routes
- airports
- airline

Aggregate view

Commit to Amazon Keyspaces

Commit to Apache Cassandra

Export to PNG

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)   | equipe |
| 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_airline s | SFO              | EWR                   | 1            | San Francisco International      | Newark Liberty International | 747    |

[TABLE] airports

| Primary key       | Non-key columns |             |                  |                          |
|-------------------|-----------------|-------------|------------------|--------------------------|
| Partition key     |                 |             |                  |                          |
| airport_id (text) |                 |             |                  |                          |
| EWR               | name (text)     | city (text) | iata_code (text) | details (map<text,text>) |

Anda memiliki opsi berikut untuk melakukan perubahan:

- Berkomitmen ke Amazon Keyspaces
- Berkomitmen pada cluster Apache Cassandra

Untuk mempelajari lebih lanjut tentang cara melakukan perubahan, lihat [the section called “Komit model data”](#).

## Buat model data baru dengan NoSQL Workbench

Anda dapat menggunakan pemodel data NoSQL Workbench untuk merancang model data baru berdasarkan pola akses data aplikasi Anda. Untuk membuat model data baru untuk Amazon Keyspaces, Anda dapat menggunakan pemodel data NoSQL Workbench untuk membuat ruang kunci, tabel, dan kolom. Ikuti langkah-langkah ini untuk membuat model data baru.

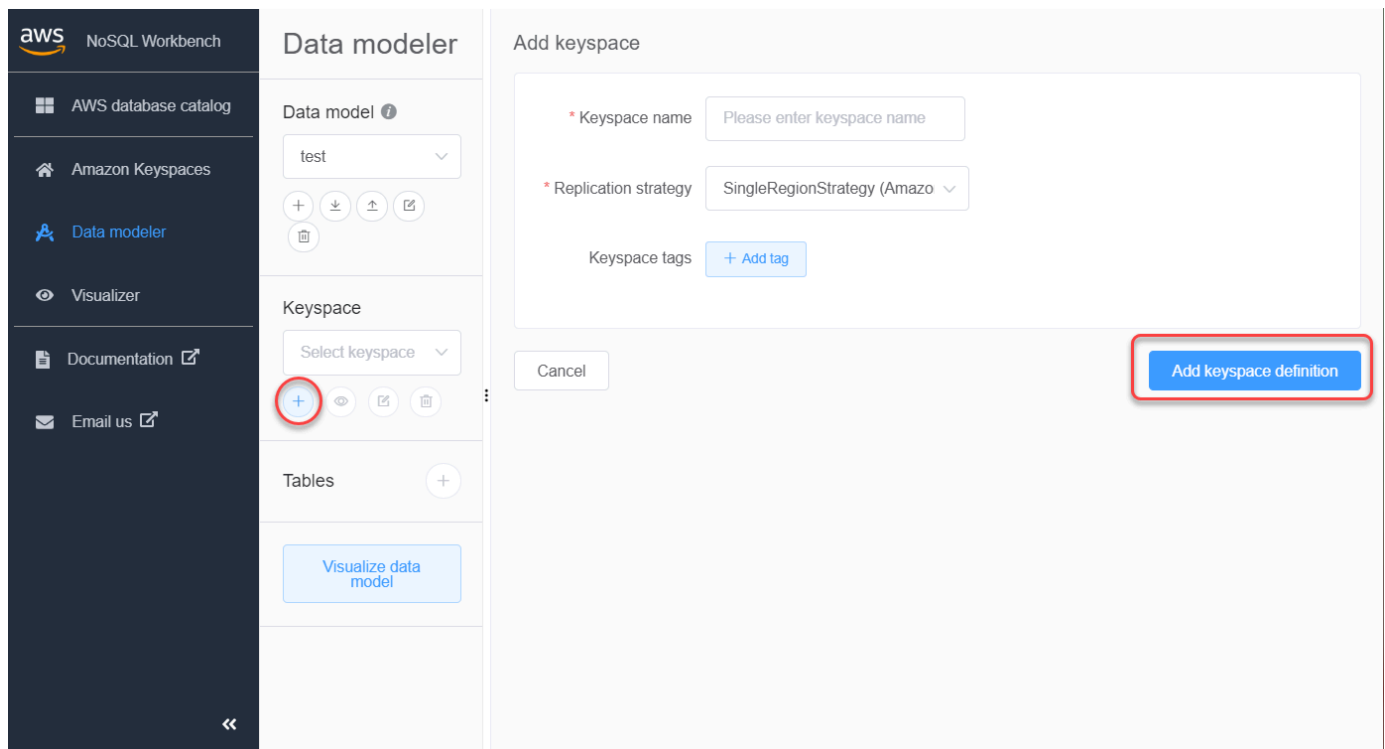
1. Untuk membuat keyspace baru, pilih tanda plus di bawah Keyspace.



Pada langkah ini, pilih properti dan pengaturan berikut.

- Nama Keyspace — Masukkan nama keyspace baru.
- Strategi replikasi — Pilih strategi replikasi untuk keyspace. Amazon Keyspaces menggunakan SingleRegionStrategy untuk mereplikasi data tiga kali secara otomatis di beberapa AWS Availability Zone. Jika Anda berencana untuk mengkomit model data ke cluster Apache Cassandra, Anda dapat memilih atau. SimpleStrategyNetworkTopologyStrategy
- Tag Keyspaces — Tag sumber daya bersifat opsional dan memungkinkan Anda mengkategorikan sumber daya Anda dengan cara yang berbeda—misalnya, berdasarkan tujuan, pemilik, lingkungan, atau kriteria lainnya. Untuk mempelajari lebih lanjut tentang tag untuk sumber daya Amazon Keyspaces, lihat. [the section called “Bekerja dengan tag”](#)

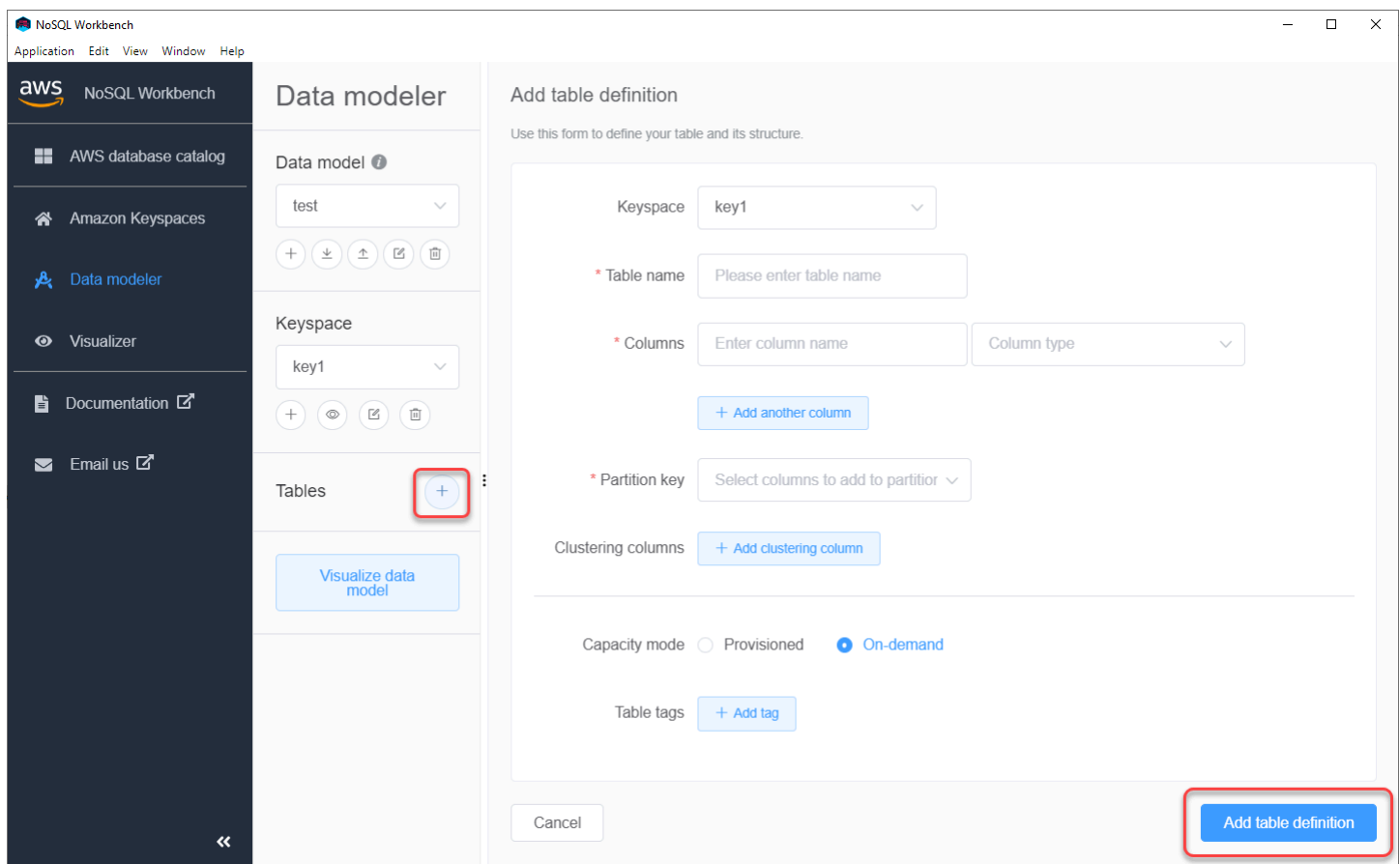
2. Pilih Add keyspace definition untuk membuat keyspace.



3. Untuk membuat tabel baru, pilih tanda plus di samping Tabel. Pada langkah ini, Anda menentukan properti dan pengaturan berikut.

- Nama tabel — Nama tabel baru.
- Kolom - Tambahkan nama kolom dan pilih tipe data. Ulangi langkah-langkah ini untuk setiap kolom dalam skema Anda.
- Tombol partisi - Pilih kolom untuk kunci partisi.

- Kolom pengelompokan - Pilih kolom pengelompokan (opsional).
  - Mode kapasitas - Pilih mode read/write kapasitas untuk tabel. Anda dapat memilih kapasitas yang disediakan atau sesuai permintaan. Untuk mempelajari lebih lanjut tentang mode kapasitas, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).
  - Tag tabel — Tag sumber daya bersifat opsional dan memungkinkan Anda mengkategorikan sumber daya Anda dengan cara yang berbeda—misalnya, berdasarkan tujuan, pemilik, lingkungan, atau kriteria lainnya. Untuk mempelajari lebih lanjut tentang tag untuk sumber daya Amazon Keyspaces, lihat [the section called “Bekerja dengan tag”](#)
4. Pilih Tambahkan definisi tabel untuk membuat tabel baru.
  5. Ulangi langkah-langkah ini untuk membuat tabel tambahan.
  6. Lanjutkan [the section called “Memvisualisasikan Model Data”](#) untuk memvisualisasikan model data yang Anda buat.



## Edit model data yang ada dengan NoSQL Workbench

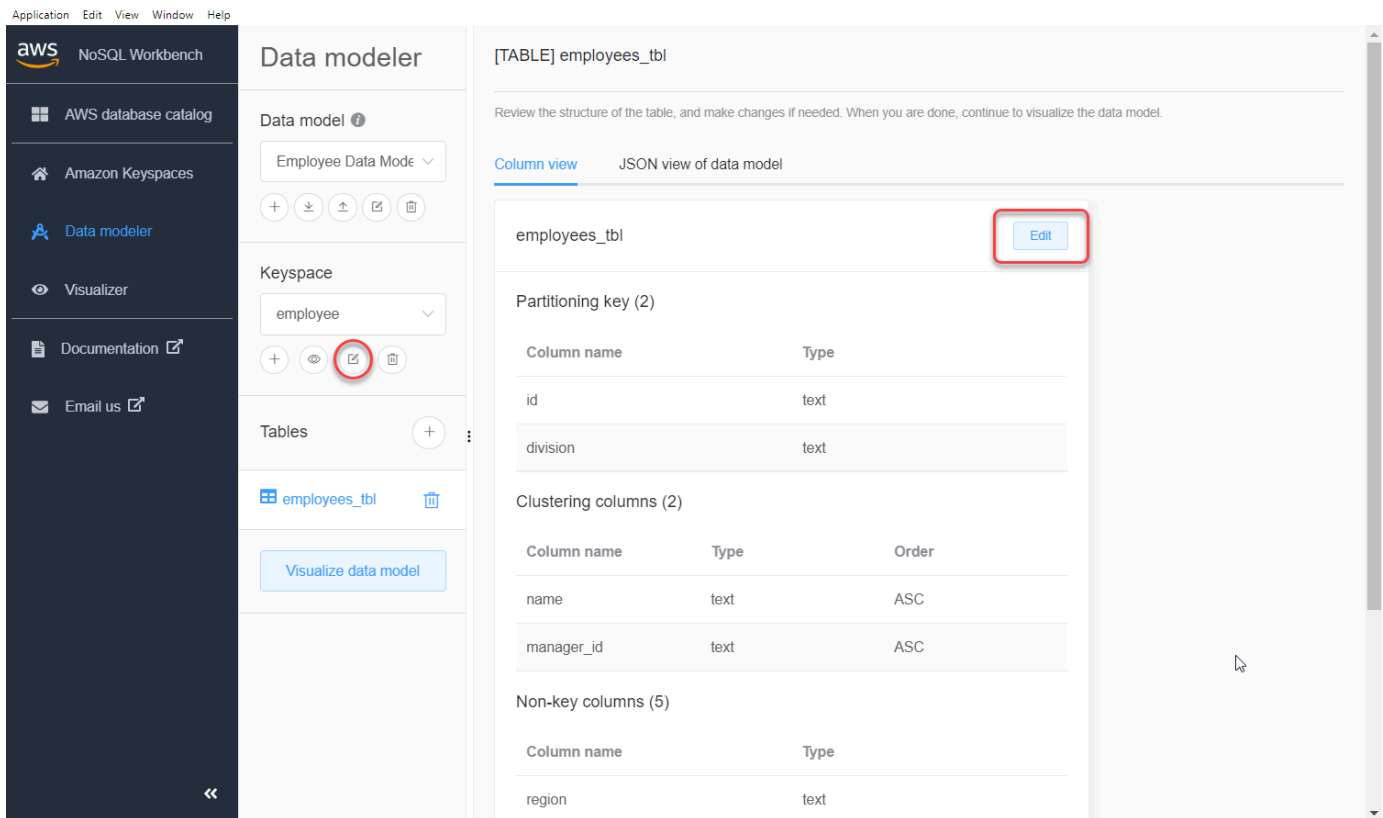
Anda dapat menggunakan pemodel data untuk mengimpor dan memodifikasi model data yang ada yang dibuat menggunakan NoSQL Workbench. Pemodel data juga menyertakan beberapa contoh model data untuk membantu Anda memulai pemodelan data. Model data yang dapat Anda edit dengan NoSQL Workbench dapat berupa model data yang diimpor dari file, model data sampel yang disediakan, atau model data yang Anda buat sebelumnya.

1. Untuk mengedit ruang kunci, pilih simbol edit di bawah Keyspace.

Pada langkah ini, Anda dapat mengedit properti dan pengaturan berikut.

- Nama Keyspace — Masukkan nama keyspace baru.
- Strategi replikasi — Pilih strategi replikasi untuk keyspace. Amazon Keyspaces menggunakan `SingleRegionStrategy` untuk mereplikasi data tiga kali secara otomatis di beberapa AWS Availability Zone. Jika Anda berencana untuk mengkomit model data ke cluster Apache Cassandra, Anda dapat memilih atau. `SimpleStrategyNetworkTopologyStrategy`
- Tag Keyspaces — Tag sumber daya bersifat opsional dan memungkinkan Anda mengkategorikan sumber daya Anda dengan cara yang berbeda—misalnya, berdasarkan tujuan, pemilik, lingkungan, atau kriteria lainnya. Untuk mempelajari lebih lanjut tentang tag untuk sumber daya Amazon Keyspaces, lihat. [the section called “Bekerja dengan tag”](#)

2. Pilih Simpan suntingan untuk memperbarui ruang kunci.



3. Untuk mengedit tabel, pilih Edit di samping nama tabel. Pada langkah ini, Anda dapat memperbarui properti dan pengaturan berikut.
  - Nama tabel — Nama tabel baru.
  - Kolom - Tambahkan nama kolom dan pilih tipe data. Ulangi langkah-langkah ini untuk setiap kolom dalam skema Anda.
  - Tombol partisi - Pilih kolom untuk kunci partisi.
  - Kolom pengelompokan - Pilih kolom pengelompokan (opsional).
  - Mode kapasitas - Pilih mode read/write kapasitas untuk tabel. Anda dapat memilih kapasitas yang disediakan atau sesuai permintaan. Untuk mempelajari lebih lanjut tentang mode kapasitas, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).
  - Tag tabel — Tag sumber daya bersifat opsional dan memungkinkan Anda mengkategorikan sumber daya Anda dengan cara yang berbeda—misalnya, berdasarkan tujuan, pemilik, lingkungan, atau kriteria lainnya. Untuk mempelajari lebih lanjut tentang tag untuk sumber daya Amazon Keyspaces, lihat. [the section called “Bekerja dengan tag”](#)
4. Pilih Simpan suntingan untuk memperbarui tabel.
5. Lanjutkan [the section called “Memvisualisasikan Model Data”](#) untuk memvisualisasikan model data yang Anda perbarui.

# Cara mengkomit model data ke Amazon Keyspaces dan Apache Cassandra

Bagian ini menunjukkan kepada Anda cara mengkomit model data yang telah selesai ke Amazon Keyspaces dan Apache Cassandra cluster. Proses ini secara otomatis membuat sumber daya sisi server untuk ruang kunci dan tabel berdasarkan pengaturan yang Anda tentukan dalam model data.

The screenshot shows the AWS NoSQL Workbench Visualizer interface. On the left sidebar, the 'Visualizer' tab is selected. The main panel displays the 'routes' table schema. The schema is organized into three main sections: 'Primary key', 'Partition 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       |

At the bottom of the interface, there are three buttons: 'Aggregate view', 'Commit to Amazon Keyspaces', and 'Commit to Apache Cassandra'. The 'Commit to Amazon Keyspaces' and 'Commit to Apache Cassandra' buttons are highlighted with a red rectangle.

## Topik

- [Sebelum Anda mulai](#)
- [Connect ke Amazon Keyspaces dengan kredensial khusus layanan](#)
- [Connect ke Amazon Keyspaces dengan kredensial AWS Identity and Access Management \(IAM\)](#)
- [Gunakan koneksi tersimpan](#)
- [Berkomitmen untuk Apache Cassandra](#)

## Sebelum Anda mulai

Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien. Untuk terhubung ke Amazon Keyspaces menggunakan TLS, Anda harus menyelesaikan tugas berikut sebelum dapat memulai.

- Unduh sertifikat digital berikut dan simpan file secara lokal atau di direktori home Anda.
  1. AmazonRootCA1
  2. AmazonRootCA2
  3. AmazonRootCA3
  4. AmazonRootCA4
  5. Starfield Class 2 Root (opsional - untuk kompatibilitas mundur)

Untuk mengunduh sertifikat, Anda dapat menggunakan perintah berikut.

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

### Note

Amazon Keyspaces sebelumnya menggunakan sertifikat TLS yang ditambahkan ke Starfield Class 2 CA. AWS memigrasikan semua Wilayah AWS ke sertifikat yang dikeluarkan di bawah Amazon Trust Services (Amazon Root CA 1—4). Selama transisi ini, konfigurasi klien untuk mempercayai Amazon Root CA 1-4 dan root Starfield untuk memastikan kompatibilitas di semua Wilayah.

Gabungkan semua sertifikat yang diunduh menjadi satu pem file dengan nama *keyspaces-bundle.pem* dalam contoh kami. Anda dapat melakukan ini dengan menjalankan perintah berikut. Catat jalur ke file, Anda perlu ini nanti.

```
cat AmazonRootCA1.pem \
 AmazonRootCA2.pem \
 AmazonRootCA3.pem \
 AmazonRootCA4.pem \
```

```
sf-class2-root.crt \
> keyspaces-bundle.pem
```

Setelah Anda menyimpan file sertifikat, Anda dapat terhubung ke Amazon Keyspaces. Salah satu opsi adalah terhubung dengan menggunakan kredensial khusus layanan. Service-specific kredensial adalah nama pengguna dan kata sandi yang terkait dengan pengguna IAM tertentu dan hanya dapat digunakan dengan layanan yang ditentukan. Opsi kedua adalah terhubung dengan kredensial IAM yang menggunakan [proses AWS Signature Version 4 \(SigV4\)](#). Untuk mempelajari lebih lanjut tentang dua opsi ini, lihat [the section called “Buat kredensial akses terprogram”](#).

Untuk terhubung dengan kredensial khusus layanan, lihat. [the section called “Connect dengan kredensial khusus layanan”](#)

Untuk terhubung dengan kredensial IAM, lihat. [the section called “Connect dengan kredensial IAM”](#)

## Connect ke Amazon Keyspaces dengan kredensial khusus layanan

Bagian ini menunjukkan cara menggunakan kredensial khusus layanan untuk mengkomit model data yang Anda buat atau edit dengan NoSQL Workbench.

1. Untuk membuat koneksi baru menggunakan kredensial khusus layanan, pilih tab Connect by using user name dan password.
  - Sebelum memulai, Anda harus membuat kredensial khusus layanan menggunakan proses yang didokumentasikan di. [the section called “Buat kredensial khusus layanan”](#)

Setelah Anda memperoleh kredensial khusus layanan, Anda dapat terus mengatur koneksi. Lanjutkan dengan salah satu dari berikut ini:

- Nama pengguna — Masukkan nama pengguna.
- Kata sandi — Masukkan kata sandi.
- Wilayah AWS— Untuk Wilayah yang tersedia, lihat [the section called “Titik akhir layanan”](#).
- Port - Amazon Keyspaces menggunakan port 9142.

Atau, Anda dapat mengimpor kredensial yang disimpan dari file.

2. Pilih Komit untuk memperbarui Amazon Keyspaces dengan model data.

## Commit to Amazon Keyspaces

 On this page, you can create server-side resources such as keyspaces and tables for the chosen data model.



< Use saved connections    Connect by using IAM credentials    Connect by using user name >



You can generate service-specific credentials to allow your users to access Amazon Keyspaces using AWS Management Console or AWS CLI.

[How to generate Amazon Keyspaces credentials](#)

\* User Name

anika

\* Password

.....



\* AWS Region

us-east-1



\* Port

9142

OR

 Import from credential file

Cancel

Reset

Commit



## Connect ke Amazon Keyspaces dengan kredensial AWS Identity and Access Management (IAM)

Bagian ini menunjukkan cara menggunakan kredensial IAM untuk melakukan model data yang dibuat atau diedit dengan NoSQL Workbench.

1. Untuk membuat koneksi baru menggunakan kredensial IAM, pilih tab Connect by using IAM credentials.
  - Sebelum Anda mulai, Anda harus membuat kredensial IAM menggunakan salah satu metode berikut.
    - Untuk akses konsol, gunakan nama pengguna dan kata sandi IAM Anda untuk masuk ke halaman masuk [Konsol Manajemen AWS](#) dari IAM. Untuk informasi tentang kredensial AWS keamanan, termasuk akses terprogram dan alternatif untuk kredensial jangka panjang, lihat kredensial [AWS keamanan](#) di Panduan Pengguna IAM. Untuk detail tentang masuk ke Akun AWS, lihat [Cara masuk AWS di](#) Panduan AWS Sign-In Pengguna.
    - Untuk akses CLI, Anda memerlukan ID kunci akses dan kunci akses rahasia. Gunakan kredensial sementara alih-alih kunci akses jangka panjang jika memungkinkan. Kredensial sementara mencakup ID kunci akses, kunci akses rahasia, dan token keamanan yang menunjukkan kapan kredensialnya kedaluwarsa. Untuk informasi selengkapnya, lihat [Menggunakan kredensial sementara dengan AWS sumber daya](#) di Panduan Pengguna IAM.
    - Untuk akses API, Anda memerlukan access key ID dan secret access key. Gunakan kunci akses pengguna IAM alih-alih kunci Pengguna root akun AWS akses. Untuk informasi selengkapnya tentang membuat kunci akses, lihat [Mengelola kunci akses untuk pengguna IAM](#) di Panduan Pengguna IAM.

Untuk informasi selengkapnya, lihat [Mengelola kunci akses untuk pengguna IAM](#).

Setelah Anda mendapatkan kredensial IAM, Anda dapat terus mengatur koneksi.

- Nama koneksi — Nama koneksi.
- Wilayah AWS— Untuk Wilayah yang tersedia, lihat [the section called “Titik akhir layanan”](#).
- ID kunci akses — Masukkan ID kunci akses.

- Kunci akses rahasia — Masukkan kunci akses rahasia.
  - Port - Amazon Keyspaces menggunakan port 9142.
  - AWS sertifikat publik — Arahkan ke AWS sertifikat yang diunduh pada langkah pertama.
  - Persisten koneksi - Pilih kotak centang ini jika Anda ingin menyimpan rahasia AWS koneksi secara lokal.
2. Pilih Komit untuk memperbarui Amazon Keyspaces dengan model data.

**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\administrator\aws\credentials

Cancel

Reset

Commit

## Gunakan koneksi tersimpan

Jika sebelumnya Anda telah menyiapkan sambungan ke Amazon Keyspaces, Anda dapat menggunakannya sebagai koneksi default untuk melakukan perubahan model data. Pilih tab Gunakan koneksi tersimpan dan lanjutkan melakukan pembaruan.

### 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

## Berkomitmen untuk Apache Cassandra

Bagian ini memandu Anda melalui membuat koneksi ke cluster Apache Cassandra untuk melakukan model data yang dibuat atau diedit dengan NoSQL Workbench.

 Note

Hanya model data yang telah dibuat dengan `SimpleStrategy` atau `NetworkTopologyStrategy` dapat berkomitmen untuk cluster Apache Cassandra. Untuk mengubah strategi replikasi, edit ruang kunci di pemodel data.

1.
  - Nama pengguna — Masukkan nama pengguna jika otentikasi diaktifkan di cluster.
  - Kata sandi — Masukkan kata sandi jika otentikasi diaktifkan di cluster.
  - Titik kontak - Masukkan titik kontak.
  - Pusat data lokal — Masukkan nama pusat data lokal.
  - Port — Koneksi menggunakan port 9042.
2. Pilih Komit untuk memperbarui cluster Apache Cassandra dengan model data.

## 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

## Contoh model data di NoSQL Workbench

Halaman beranda untuk pemodel dan visualizer menampilkan sejumlah model sampel yang dikirimkan dengan NoSQL Workbench. Bagian ini menjelaskan model tersebut dan potensi penggunaannya.

### Topik

- [Model data karyawan](#)
- [Model data transaksi kartu kredit](#)
- [Model data operasi maskapai](#)

### Model data karyawan

Model data ini mewakili skema Amazon Keyspaces untuk aplikasi database karyawan.

Aplikasi yang mengakses informasi karyawan untuk perusahaan tertentu dapat menggunakan model data ini.

Pola akses yang didukung oleh model data ini adalah:

- Pengambilan catatan karyawan dengan ID yang diberikan.
- Pengambilan catatan karyawan dengan ID dan divisi yang diberikan.
- Pengambilan catatan karyawan dengan ID dan nama yang diberikan.

### Model data transaksi kartu kredit

Model data ini mewakili skema Amazon Keyspaces untuk transaksi kartu kredit di toko ritel.

Penyimpanan transaksi kartu kredit tidak hanya membantu toko dengan pembukuan, tetapi juga membantu manajer toko menganalisis tren pembelian, yang dapat membantu mereka dengan peramalan dan perencanaan.

Pola akses yang didukung oleh model data ini adalah:

- Pengambilan transaksi dengan nomor kartu kredit, bulan dan tahun, dan tanggal.
- Pengambilan transaksi berdasarkan nomor kartu kredit, kategori, dan tanggal.
- Pengambilan transaksi berdasarkan kategori, lokasi, dan nomor kartu kredit.

- Pengambilan transaksi dengan nomor kartu kredit dan status sengketa.

## Model data operasi maskapai

Model data ini menunjukkan data tentang penerbangan pesawat, termasuk bandara, maskapai penerbangan, dan rute penerbangan.

Komponen utama pemodelan Amazon Keyspaces yang ditunjukkan adalah pasangan nilai kunci, penyimpanan data kolom lebar, kunci komposit, dan tipe data kompleks seperti peta untuk mendemonstrasikan pola akses data NoSQL yang umum.

Pola akses yang didukung oleh model data ini adalah:

- Pengambilan rute yang berasal dari maskapai penerbangan tertentu di bandara tertentu.
- Pengambilan rute dengan bandara tujuan tertentu.
- Pengambilan bandara dengan penerbangan langsung.
- Pengambilan detail bandara dan detail maskapai penerbangan.

## Riwayat rilis untuk NoSQL Workbench

Tabel berikut menjelaskan perubahan penting dalam setiap rilis aplikasi sisi klien NoSQL Workbench.

| Ubah                                         | Deskripsi                                                                                                                                                                                                                         | Date            |
|----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| NoSQL Workbench untuk Amazon Keyspaces - GA. | NoSQL Workbench untuk Amazon Keyspaces umumnya tersedia.                                                                                                                                                                          | 28 Oktober 2020 |
| Pratinjau NoSQL Workbench dirilis.           | NoSQL Workbench adalah aplikasi sisi klien yang membantu Anda merancang dan memvisualisasikan model data nonrelasional untuk Amazon Keyspaces dengan lebih mudah. Klien NoSQL Workbench tersedia untuk Windows, macOS, dan Linux. | 5 Oktober 2020  |



| Ubah | Deskripsi                                                                                  | Date |
|------|--------------------------------------------------------------------------------------------|------|
|      | Untuk informasi selengkapnya, lihat <a href="#">NoSQL Workbench for Amazon Keyspaces</a> . |      |

# Contoh kode untuk Amazon Keyspaces menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Amazon Keyspaces dengan AWS software development kit (SDK).

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

- [Contoh dasar untuk Amazon Keyspaces menggunakan SDK AWS](#)
  - [Halo Amazon Keyspaces](#)
  - [Pelajari dasar-dasar Amazon Keyspaces dengan SDK AWS](#)
  - [Tindakan untuk Amazon Keyspaces menggunakan SDK AWS](#)
    - [Gunakan CreateKeyspace dengan AWS SDK](#)
    - [Gunakan CreateTable dengan AWS SDK](#)
    - [Gunakan DeleteKeyspace dengan AWS SDK](#)
    - [Gunakan DeleteTable dengan AWS SDK](#)
    - [Gunakan GetKeyspace dengan AWS SDK](#)
    - [Gunakan GetTable dengan AWS SDK](#)
    - [Gunakan ListKeyspaces dengan AWS SDK](#)
    - [Gunakan ListTables dengan AWS SDK](#)
    - [Gunakan RestoreTable dengan AWS SDK](#)
    - [Gunakan UpdateTable dengan AWS SDK](#)

# Contoh dasar untuk Amazon Keyspaces menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar Amazon Keyspaces (untuk Apache Cassandra) dengan SDK. AWS

Contoh

- [Halo Amazon Keyspaces](#)
- [Pelajari dasar-dasar Amazon Keyspaces dengan SDK AWS](#)
- [Tindakan untuk Amazon Keyspaces menggunakan SDK AWS](#)
  - [Gunakan CreateKeyspace dengan AWS SDK](#)
  - [Gunakan CreateTable dengan AWS SDK](#)
  - [Gunakan DeleteKeyspace dengan AWS SDK](#)
  - [Gunakan DeleteTable dengan AWS SDK](#)
  - [Gunakan GetKeyspace dengan AWS SDK](#)
  - [Gunakan GetTable dengan AWS SDK](#)
  - [Gunakan ListKeyspaces dengan AWS SDK](#)
  - [Gunakan ListTables dengan AWS SDK](#)
  - [Gunakan RestoreTable dengan AWS SDK](#)
  - [Gunakan UpdateTable dengan AWS SDK](#)

## Halo Amazon Keyspaces

Contoh kode berikut menunjukkan cara memulai menggunakan Amazon Keyspaces.

.NET

SDK untuk .NET

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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();
 }
}
```

- Untuk detail API, lihat [ListKeyspaces](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}
}

```

- Untuk detail API, lihat [ListKeyspaces](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}")
 }
 }
}
}

```

- Untuk detail API, lihat [ListKeyspaces](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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"))
```

- Untuk detail API, lihat [ListKeyspaces](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Pelajari dasar-dasar Amazon Keyspaces dengan SDK AWS

Contoh-contoh kode berikut menunjukkan cara:

- Buat keyspace dan tabel. Skema tabel menyimpan data film dan mengaktifkan pemulihan point-in-time.
- Connect ke keyspace menggunakan koneksi TLS aman dengan otentikasi SiGv4.
- Kueri tabel. Tambahkan, ambil, dan perbarui data film.
- Perbarui tabel. Tambahkan kolom untuk melacak film yang ditonton.
- Kembalikan tabel ke keadaan sebelumnya dan bersihkan sumber daya.

### .NET

SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode 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!;
 }
 }
}

```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk .NET .
  - [CreateKeyspace](#)
  - [CreateTable](#)
  - [DeleteKeyspace](#)
  - [DeleteTable](#)
  - [GetKeyspace](#)

- [GetTable](#)
- [ListKeyspaces](#)
- [ListTables](#)
- [RestoreTable](#)
- [UpdateTable](#)

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode 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);
 }
 }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - [CreateKeyspace](#)
  - [CreateTable](#)
  - [DeleteKeyspace](#)
  - [DeleteTable](#)
  - [GetKeyspace](#)
  - [GetTable](#)
  - [ListKeyspaces](#)
  - [ListTables](#)
  - [RestoreTable](#)
  - [UpdateTable](#)

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode 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](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 keyspace 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 collList = ArrayList<ColumnDefinition>()
collList.add(defTitle)
collList.add(defYear)
collList.add(defReleaseDate)
collList.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 = collList
 }

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}")
 }
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK For Kotlin.
  - [CreateKeyspace](#)
  - [CreateTable](#)
  - [DeleteKeyspace](#)
  - [DeleteTable](#)
  - [GetKeyspace](#)
  - [GetTable](#)
  - [ListKeyspaces](#)
  - [ListTables](#)
  - [RestoreTable](#)
  - [UpdateTable](#)

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode AWS](#).

Jalankan skenario interaktif di penggugah/prompt perintah.

```
class KeyspaceScenario:
 """Runs an interactive scenario that shows how to get started using Amazon
 Keyspaces."""

 def __init__(self, ks_wrapper):
 """
 :param ks_wrapper: An object that wraps Amazon Keyspace actions.
 """
 self.ks_wrapper = ks_wrapper

 @demo_func
```

```

def create_keyspace(self):
 """
 1. Creates a keyspace.
 2. Lists up to 10 keyspaces in your account.
 """
 print("Let's create a keyspace.")
 ks_name = q.ask(
 "Enter a name for your new keyspace.\nThe name can contain only
letters, "
 "numbers and underscores: ",
 q.non_empty,
)
 if self.ks_wrapper.exists_keyspace(ks_name):
 print(f"A keyspace named {ks_name} exists.")
 else:
 ks_arn = self.ks_wrapper.create_keyspace(ks_name)
 ks_exists = False
 while not ks_exists:
 wait(3)
 ks_exists = self.ks_wrapper.exists_keyspace(ks_name)
 print(f"Created a new keyspace.\n\t{ks_arn}.")
 print("The first 10 keyspaces in your account are:\n")
 self.ks_wrapper.list_keyspaces(10)

@demo_func
def create_table(self):
 """
 1. Creates a table in the keyspace. The table is configured with a schema
to hold
 movie data and has point-in-time recovery enabled.
 2. Waits for the table to be in an active state.
 3. Displays schema information for the table.
 4. Lists tables in the keyspace.
 """
 print("Let's create a table for movies in your keyspace.")
 table_name = q.ask("Enter a name for your table: ", q.non_empty)
 table = self.ks_wrapper.get_table(table_name)
 if table is not None:
 print(
 f"A table named {table_name} already exists in keyspace "
 f"{self.ks_wrapper.ks_name}."
)
 else:
 table_arn = self.ks_wrapper.create_table(table_name)

```

```

 print(f"Created table {table_name}:\n\t{table_arn}")
 table = {"status": None}
 print("Waiting for your table to be ready...")
 while table["status"] != "ACTIVE":
 wait(5)
 table = self.ks_wrapper.get_table(table_name)
 print(f"Your table is {table['status']}. Its schema is:")
 pp(table["schemaDefinition"])
 print("\nThe tables in your keyspace are:\n")
 self.ks_wrapper.list_tables()

 @demo_func
 def ensure_tls_cert(self):
 """
 Ensures you have a TLS certificate available to use to secure the
 connection
 to the keyspace. This function downloads a default certificate or lets
 you
 specify your own.
 """
 print("To connect to your keyspace, you must have a TLS certificate.")
 print("Checking for TLS certificate...")
 cert_path = os.path.join(
 os.path.dirname(__file__), QueryManager.DEFAULT_CERT_FILE
)
 if not os.path.exists(cert_path):
 cert_choice = q.ask(
 f"Press enter to download a certificate from
{QueryManager.CERT_URL} "
 f"or enter the full path to the certificate you want to use: "
)
 if cert_choice:
 cert_path = cert_choice
 else:
 cert = requests.get(QueryManager.CERT_URL).text
 with open(cert_path, "w") as cert_file:
 cert_file.write(cert)
 else:
 q.ask(f"Certificate {cert_path} found. Press Enter to continue.")
 print(
 f"Certificate {cert_path} will be used to secure the connection to
your keyspace."
)
 return cert_path

```



```

@demo_func
def query_table(self, qm, movie_file):
 """
 1. Adds movies to the table from a sample movie data file.
 2. Gets a list of movies from the table and lets you select one.
 3. Displays more information about the selected movie.
 """
 qm.add_movies(self.ks_wrapper.table_name, movie_file)
 movies = qm.get_movies(self.ks_wrapper.table_name)
 print(f"Added {len(movies)} movies to the table:")
 sel = q.choose("Pick one to learn more about it: ", [m.title for m in
movies])
 movie_choice = qm.get_movie(
 self.ks_wrapper.table_name, movies[sel].title, movies[sel].year
)
 print(movie_choice.title)
 print(f"\tReleased: {movie_choice.release_date}")
 print(f"\tPlot: {movie_choice.plot}")

@demo_func
def update_and_restore_table(self, qm):
 """
 1. Updates the table by adding a column to track watched movies.
 2. Marks some of the movies as watched.
 3. Gets the list of watched movies from the table.
 4. Restores to a movies_restored table at a previous point in time.
 5. Gets the list of movies from the restored table.
 """
 print("Let's add a column to record which movies you've watched.")
 pre_update_timestamp = datetime.utcnow()
 print(
 f"Recorded the current UTC time of {pre_update_timestamp} so we can
restore the table later."
)
 self.ks_wrapper.update_table()
 print("Waiting for your table to update...")
 table = {"status": "UPDATING"}
 while table["status"] != "ACTIVE":
 wait(5)
 table = self.ks_wrapper.get_table(self.ks_wrapper.table_name)
 print("Column 'watched' added to table.")
 q.ask(

```

```

 "Let's mark some of the movies as watched. Press Enter when you're
ready.\n"
)
 movies = qm.get_movies(self.ks_wrapper.table_name)
 for movie in movies[:10]:
 qm.watched_movie(self.ks_wrapper.table_name, movie.title, movie.year)
 print(f"Marked {movie.title} as watched.")
 movies = qm.get_movies(self.ks_wrapper.table_name, watched=True)
 print("-" * 88)
 print("The watched movies in our table are:\n")
 for movie in movies:
 print(movie.title)
 print("-" * 88)
 if q.ask(
 "Do you want to restore the table to the way it was before all of
these\n"
 "updates? Keep in mind, this can take up to 20 minutes. (y/n) ",
 q.is_yesno,
):
 starting_table_name = self.ks_wrapper.table_name
 table_name_restored =
self.ks_wrapper.restore_table(pre_update_timestamp)
 table = {"status": "RESTORING"}
 while table["status"] != "ACTIVE":
 wait(10)
 table = self.ks_wrapper.get_table(table_name_restored)
 print(
 f"Restored {starting_table_name} to {table_name_restored} "
 f"at a point in time of {pre_update_timestamp}."
)
 movies = qm.get_movies(table_name_restored)
 print("Now the movies in our table are:")
 for movie in movies:
 print(movie.title)

def cleanup(self, cert_path):
 """
 1. Deletes the table and waits for it to be removed.
 2. Deletes the keyspace.

 :param cert_path: The path of the TLS certificate used in the demo. If
the
 certificate was downloaded during the demo, it is
removed.

```

```

 """
 if q.ask(
 f"Do you want to delete your {self.ks_wrapper.table_name} table and "
 f"{self.ks_wrapper.ks_name} keyspace? (y/n) ",
 q.is_yesno,
):
 table_name = self.ks_wrapper.table_name
 self.ks_wrapper.delete_table()
 table = self.ks_wrapper.get_table(table_name)
 print("Waiting for the table to be deleted.")
 while table is not None:
 wait(5)
 table = self.ks_wrapper.get_table(table_name)
 print("Table deleted.")
 self.ks_wrapper.delete_keyspace()
 print(
 "Keyspace deleted. If you chose to restore your table during the
"
 "demo, the original table is also deleted."
)
 if cert_path == os.path.join(
 os.path.dirname(__file__), QueryManager.DEFAULT_CERT_FILE
) and os.path.exists(cert_path):
 os.remove(cert_path)
 print("Removed certificate that was downloaded for this demo.")

 def run_scenario(self):
 logging.basicConfig(level=logging.INFO, format="%(levelname)s:
%(message)s")

 print("-" * 88)
 print("Welcome to the Amazon Keyspaces (for Apache Cassandra) demo.")
 print("-" * 88)

 self.create_keyspace()
 self.create_table()
 cert_file_path = self.ensure_tls_cert()
 # Use a context manager to ensure the connection to the keyspace is
 closed.
 with QueryManager(
 cert_file_path, boto3.DEFAULT_SESSION, self.ks_wrapper.ks_name
) as qm:
 self.query_table(qm, "../../resources/sample_files/movies.json")
 self.update_and_restore_table(qm)

```

```

 self.cleanup(cert_file_path)

 print("\nThanks for watching!")
 print("-" * 88)

if __name__ == "__main__":
 try:
 scenario = KeyspaceScenario(KeyspaceWrapper.from_client())
 scenario.run_scenario()
 except Exception:
 logging.exception("Something went wrong with the demo.")

```

Tentukan kelas yang membungkus tindakan keyspace dan tabel.

```

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

```

Tentukan kelas yang membuat koneksi TLS ke ruang kunci, mengautentikasi dengan SigV4, dan mengirimkan kueri CQL ke tabel di ruang kunci.

```

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],
)

```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).

- [CreateKeyspace](#)
- [CreateTable](#)
- [DeleteKeyspace](#)
- [DeleteTable](#)
- [GetKeyspace](#)
- [GetTable](#)
- [ListKeyspaces](#)
- [ListTables](#)
- [RestoreTable](#)
- [UpdateTable](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Tindakan untuk Amazon Keyspaces menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan tindakan Amazon Keyspaces individual dengan AWS SDK. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi [API Amazon Keyspaces \(untuk Apache Cassandra\)](#).

### Contoh

- [Gunakan CreateKeyspace dengan AWS SDK](#)
- [Gunakan CreateTable dengan AWS SDK](#)
- [Gunakan DeleteKeyspace dengan AWS SDK](#)
- [Gunakan DeleteTable dengan AWS SDK](#)
- [Gunakan GetKeyspace dengan AWS SDK](#)
- [Gunakan GetTable dengan AWS SDK](#)
- [Gunakan ListKeyspaces dengan AWS SDK](#)
- [Gunakan ListTables dengan AWS SDK](#)

- [Gunakan RestoreTable dengan AWS SDK](#)
- [Gunakan UpdateTable dengan AWS SDK](#)

## Gunakan **CreateKeyspace** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `CreateKeyspace`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

### .NET

SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}
```

- Untuk detail API, lihat [CreateKeyspace](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}
```

- Untuk detail API, lihat [CreateKeyspace](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}")
 }
}
```

- Untuk detail API, lihat [CreateKeyspace](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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

```

- Untuk detail API, lihat [CreateKeyspace](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.

```

- Untuk detail API, lihat [CreateKeyspace](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **CreateTable** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `CreateTable`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

## .NET

### SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
 }

```

- Untuk detail API, lihat [CreateTable](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}
```

- Untuk detail API, lihat [CreateTable](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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 collList = ArrayList<ColumnDefinition>()
 collList.add(defTitle)
 collList.add(defYear)
 collList.add(defReleaseDate)
 collList.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 = collList
 }

 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}")
 }
}

```

- Untuk detail API, lihat [CreateTable](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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"]

```

- Untuk detail API, lihat [CreateTable](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.

```

- Untuk detail API, lihat [CreateTable](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **DeleteKeyspace** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `DeleteKeyspace`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

.NET

SDK untuk .NET

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}

```

- Untuk detail API, lihat [DeleteKeyspace](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}

```

- Untuk detail API, lihat [DeleteKeyspace](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
suspend fun deleteKeyspace(keyspaceNameVal: String?) {
 val deleteKeyspaceRequest =
 DeleteKeyspaceRequest {
 keyspaceName = keyspaceNameVal
 }

 KeyspacesClient.fromEnvironment { region = "us-east-1" }.use { keyClient ->
 keyClient.deleteKeyspace(deleteKeyspaceRequest)
 }
}
```

- Untuk detail API, lihat [DeleteKeyspace](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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
```

- Untuk detail API, lihat [DeleteKeyspace](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.
```

- Untuk detail API, lihat [DeleteKeyspace](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **DeleteTable** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `DeleteTable`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

## .NET

### SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}
```

- Untuk detail API, lihat [DeleteTable](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
}
```

- Untuk detail API, lihat [DeleteTable](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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)
 }
}
```

- Untuk detail API, lihat [DeleteTable](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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
```

- Untuk detail API, lihat [DeleteTable](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.
```

- Untuk detail API, lihat [DeleteTable](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **GetKeyspace** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `GetKeyspace`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

### .NET

SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}
```

- Untuk detail API, lihat [GetKeyspace](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}
```

- Untuk detail API, lihat [GetKeyspace](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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")
 }
}
```

- Untuk detail API, lihat [GetKeyspace](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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
```

- Untuk detail API, lihat [GetKeyspace](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.

```

- Untuk detail API, lihat [GetKeyspace](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **GetTable** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `GetTable`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

## .NET

### SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
 }

```

- Untuk detail API, lihat [GetTable](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
}
}

```

- Untuk detail API, lihat [GetTable](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}")
 }
 }
}
}

```

- Untuk detail API, lihat [GetTable](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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

```

- Untuk detail API, lihat [GetTable](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.
```

- Untuk detail API, lihat [GetTable](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ListKeyspaces** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `ListKeyspaces`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

## .NET

### SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}");
 }
}
```

- Untuk detail API, lihat [ListKeyspaces](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}
```

- Untuk detail API, lihat [ListKeyspaces](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}")
 }
 }
}
```

- Untuk detail API, lihat [ListKeyspaces](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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

```

- Untuk detail API, lihat [ListKeyspaces](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.
```

- Untuk detail API, lihat [ListKeyspaces](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ListTables** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `ListTables`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

## .NET

### SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}
```

- Untuk detail API, lihat [ListTables](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}

```

- Untuk detail API, lihat [ListTables](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}")
 }
 }
}

```

- Untuk detail API, lihat [ListTables](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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

```

- Untuk detail API, lihat [ListTables](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.

```

- Untuk detail API, lihat [ListTables](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **RestoreTable** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `RestoreTable`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

.NET

SDK untuk .NET

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}

```

- Untuk detail API, lihat [RestoreTable](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}

```

- Untuk detail API, lihat [RestoreTable](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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}")
 }
}

```

- Untuk detail API, lihat [RestoreTable](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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

```

- Untuk detail API, lihat [RestoreTable](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

TRY.

```

oo_result = lo_kys->restoretable(
 iv_sourcekeyspacename = iv_source_keyspace_name
 iv_sourcetablename = iv_source_table_name
 iv_targetkeyspacename = iv_target_keyspace_name
 iv_targettablename = iv_target_table_name
 iv_restoretimestamp = iv_restore_timestamp).

```

```

 MESSAGE 'Table restore initiated successfully.' TYPE 'I'.
 CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
 DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
 MESSAGE lv_error TYPE 'E'.
 ENDTRY.

```

- Untuk detail API, lihat [RestoreTable](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **UpdateTable** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `UpdateTable`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Pelajari dasar-dasarnya](#)

## .NET

### SDK untuk .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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;
}

```

- Untuk detail API, lihat [UpdateTable](#) di Referensi AWS SDK untuk .NET API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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);
 }
}
```

- Untuk detail API, lihat [UpdateTable](#) di Referensi AWS SDK for Java 2.x API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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)
 }
}
```

- Untuk detail API, lihat [UpdateTable](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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

```

- Untuk detail API, lihat [UpdateTable](#) di AWS SDK for Python (Boto3) Referensi API.

## SAP ABAP

### SDK for SAP ABAP

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode 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.
```

- Untuk detail API, lihat [UpdateTable](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

# Amazon Keyspaces (untuk Apache Cassandra)

## perpustakaan dan alat

Bagian ini memberikan informasi tentang perpustakaan, contoh kode, dan alat Amazon Keyspaces (untuk Apache Cassandra).

Topik

- [Perpustakaan dan contoh](#)
- [Contoh yang disorot dan repo alat pengembang](#)

## Perpustakaan dan contoh

[Anda dapat menemukan pustaka sumber terbuka Amazon Keyspaces dan alat pengembang di GitHub dalam dan sampel repo AWS.AWS](#)

## Perangkat pengembang Amazon Keyspaces (untuk Apache Cassandra)

Repositori ini menyediakan gambar docker dengan alat pengembang yang bermanfaat untuk Amazon Keyspaces. Misalnya, ini termasuk file CQLSHRC dengan praktik terbaik, ekspansi AWS otentikasi opsional untuk cqlsh, dan alat pembantu untuk melakukan tugas-tugas umum. Toolkit ini dioptimalkan untuk Amazon Keyspaces, tetapi juga berfungsi dengan cluster Apache Cassandra.

<https://github.com/aws-samples/amazon-keyspaces-toolkit>.

## Amazon Keyspaces (untuk Apache Cassandra) contoh

Repo ini adalah daftar resmi kode contoh Amazon Keyspaces kami. Repo dibagi lagi menjadi beberapa bagian berdasarkan bahasa (lihat [Contoh](#)). Setiap bahasa memiliki subbagian contohnya sendiri. Contoh-contoh ini menunjukkan implementasi dan pola layanan Amazon Keyspaces umum yang dapat Anda gunakan saat membuat aplikasi.

<https://github.com/aws-samples/amazon-keyspaces-examples/>.

## AWS Plugin otentikasi Sigv4 Versi Tanda Tangan 4 (SiGv4)

Plugin memungkinkan Anda mengelola akses ke Amazon Keyspaces dengan AWS Identity and Access Management menggunakan pengguna dan peran (IAM).

Jawa: <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>.

Pergi: <https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin>.

## Contoh yang disorot dan repo alat pengembang

Di bawah ini adalah pilihan alat komunitas yang bermanfaat untuk Amazon Keyspaces (untuk Apache Cassandra).

### Buffer Protokol Amazon Keyspaces

Anda dapat menggunakan Protocol Buffer (Protobuf) dengan Amazon Keyspaces untuk memberikan alternatif Apache Cassandra User Defined Types (UDTs). Protobuf adalah format data lintas platform sumber terbuka dan gratis yang digunakan untuk membuat serial data terstruktur. Anda dapat menyimpan data Protobuf menggunakan tipe data CQL dan refactor UDT sambil mempertahankan BLOB data terstruktur di seluruh aplikasi dan bahasa pemrograman.

Repositori ini menyediakan contoh kode yang terhubung ke Amazon Keyspaces, membuat tabel baru, dan menyisipkan baris yang berisi pesan Protobuf. Kemudian baris dibaca dengan konsistensi yang kuat.

<https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/protobuf-user-defined-types>

### CloudFormation template untuk membuat CloudWatch dasbor Amazon untuk metrik Amazon Keyspaces (untuk Apache Cassandra)

Repositori ini menyediakan CloudFormation template untuk menyiapkan CloudWatch metrik dengan cepat untuk Amazon Keyspaces. Menggunakan template ini akan memungkinkan Anda untuk memulai dengan lebih mudah dengan menyediakan CloudWatch dasbor prebuilt deployable dengan metrik yang umum digunakan.

<https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates>.

## Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan AWS Lambda

Repositori berisi contoh yang menunjukkan cara menghubungkan ke Amazon Keyspaces dari Lambda. Di bawah ini adalah beberapa contoh.

C#/.NET: <https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/dotnet/datastax-v3/connection-lambda>

Jawa: <https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/connection-lambda>.

Contoh Lambda lain yang menunjukkan cara menerapkan dan menggunakan Amazon Keyspaces dari Lambda Python tersedia dari repo berikut.

<https://github.com/aws-samples/aws-keyspaces-lambda-python>

## Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan Spring

Ini adalah contoh yang menunjukkan cara menggunakan Amazon Keyspaces dengan Spring Boot.

<https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/java/datastax-v4/spring>

## Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan Scala

Ini adalah contoh yang menunjukkan cara terhubung ke Amazon Keyspaces menggunakan plugin otentikasi SiGv4 dengan Scala.

<https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/connection-sigv4>

## Menggunakan Amazon Keyspaces (untuk Apache Cassandra) dengan AWS Glue

Ini adalah contoh yang menunjukkan cara menggunakan Amazon Keyspaces dengan AWS Glue

<https://github.com/aws-samples/amazon-keyspaces-examples/tree/main/scala/datastax-v4/aws-glue>

## Amazon Keyspaces (untuk Apache Cassandra) Bahasa kueri Cassandra (CQL) ke konverter CloudFormation

Paket ini mengimplementasikan alat baris perintah untuk mengonversi skrip Apache Cassandra Query Language (CQL) ke CloudFormation templat (), yang memungkinkan skema CloudFormation Amazon Keyspaces mudah dikelola dalam tumpukan. CloudFormation

<https://github.com/aws/amazon-keyspaces-cql-to-cfn-converter>.

## Amazon Keyspaces (untuk Apache Cassandra) pembantu untuk driver Apache Cassandra untuk Java

Repositori ini berisi kebijakan driver, contoh, dan praktik terbaik saat menggunakan Driver DataStax Java dengan Amazon Keyspaces (untuk Apache Cassandra).

<https://github.com/aws-samples/amazon-keyspaces-java-driver-helpers>.

## Amazon Keyspaces (untuk Apache Cassandra) demo kompresi tajam

Repositori ini menunjukkan cara mengompres, menyimpan, dan objek read/write besar untuk kinerja yang lebih cepat dan biaya throughput dan penyimpanan yang lebih rendah.

<https://github.com/aws-samples/amazon-keyspaces-compression-example>.

## Amazon Keyspaces (untuk Apache Cassandra) dan demo codec Amazon S3

Codec Amazon S3 khusus mendukung pemetaan pointer UUID yang transparan dan dapat dikonfigurasi pengguna ke objek Amazon S3.

<https://github.com/aws-samples/amazon-keyspaces-large-object-s3-demo>.

# Praktik terbaik untuk merancang dan membuat arsitektur dengan Amazon Keyspaces (untuk Apache Cassandra)

Gunakan bagian ini untuk menemukan rekomendasi dengan cepat untuk memaksimalkan kinerja dan meminimalkan biaya throughput saat bekerja dengan Amazon Keyspaces.

## Daftar Isi

- [Perbedaan utama dan prinsip desain desain NoSQL](#)
  - [Perbedaan antara desain data relasional dan NoSQL](#)
  - [Dua konsep utama untuk desain NoSQL](#)
  - [Mendekati desain NoSQL](#)
- [Optimalkan koneksi driver klien untuk lingkungan tanpa server](#)
  - [Cara kerja koneksi di Amazon Keyspaces](#)
  - [Cara mengonfigurasi koneksi di Amazon Keyspaces](#)
  - [Cara mengonfigurasi kebijakan coba lagi untuk koneksi di Amazon Keyspaces](#)
  - [Cara mengonfigurasi koneksi melalui titik akhir VPC di Amazon Keyspaces](#)
  - [Cara memantau koneksi di Amazon Keyspaces](#)
  - [Cara menangani kesalahan koneksi di Amazon Keyspaces](#)
- [Praktik terbaik pemodelan data: rekomendasi untuk merancang model data](#)
  - [Cara menggunakan kunci partisi secara efektif di Amazon Keyspaces](#)
    - [Gunakan sharding tulis untuk mendistribusikan beban kerja secara merata di seluruh partisi](#)
      - [Sharding menggunakan kunci partisi majemuk dan nilai acak](#)
      - [Sharding menggunakan kunci partisi majemuk dan nilai yang dihitung](#)
- [Mengoptimalkan biaya tabel Amazon Keyspaces](#)
  - [Evaluasi biaya Anda di tingkat tabel](#)
    - [Cara melihat biaya satu tabel Amazon Keyspaces](#)
    - [Tampilan default Cost Explorer](#)
    - [Cara menggunakan dan menerapkan tanda tabel di Cost Explorer](#)
  - [Evaluasi mode kapasitas tabel Anda](#)
    - [Mode kapasitas tabel yang tersedia](#)
    - [Kapan harus memilih mode kapasitas sesuai permintaan](#)

- [Kapan harus mode kapasitas yang disediakan](#)
- [Faktor lain yang perlu dipertimbangkan saat memilih mode kapasitas tabel](#)
- [Evaluasi pengaturan Application Auto Scaling tabel Anda](#)
  - [Memahami pengaturan Application Auto Scaling](#)
  - [Cara mengidentifikasi tabel dengan pemanfaatan target rendah \( \$\leq 50\%\$ \)](#)
  - [Cara mengatasi beban kerja dengan varian musiman](#)
  - [Cara mengatasi lonjakan beban kerja dengan pola yang tidak diketahui](#)
  - [Cara mengatasi beban kerja dengan aplikasi tertaut](#)
- [Identifikasi sumber daya yang tidak digunakan untuk mengoptimalkan biaya di Amazon Keyspaces](#)
  - [Cara mengidentifikasi sumber daya yang tidak terpakai](#)
  - [Mengidentifikasi sumber daya tabel yang tidak terpakai](#)
  - [Membersihkan sumber daya tabel yang tidak terpakai](#)
  - [Membersihkan cadangan point-in-time recovery \(PITR\) yang tidak terpakai](#)
- [Evaluasi pola penggunaan tabel Anda untuk mengoptimalkan kinerja dan biaya](#)
  - [Lakukan lebih sedikit operasi bacaan sangat konsisten](#)
  - [Aktifkan Waktu untuk Tayang \(TTL\)](#)
- [Evaluasi kapasitas yang disediakan untuk penyediaan ukuran yang tepat](#)
  - [Cara mengambil metrik konsumsi dari tabel Amazon Keyspaces](#)
  - [Cara mengidentifikasi tabel Amazon Keyspaces yang kurang disediakan](#)
  - [Cara mengidentifikasi tabel Amazon Keyspaces yang disediakan secara berlebihan](#)

## Perbedaan utama dan prinsip desain desain NoSQL

Sistem database NoSQL seperti Amazon Keyspaces menggunakan model alternatif untuk manajemen data, seperti pasangan nilai kunci atau penyimpanan dokumen. Ketika Anda beralih dari sistem manajemen database relasional ke sistem database NoSQL seperti Amazon Keyspaces, penting untuk memahami perbedaan utama dan pendekatan desain tertentu.

### Topik

- [Perbedaan antara desain data relasional dan NoSQL](#)

- [Mendekati desain NoSQL](#)

## Perbedaan antara desain data relasional dan NoSQL

Sistem basis data relasional (RDBMS) dan basis data NoSQL memiliki keunggulan dan kelemahan yang berbeda:

- Di RDBMS, data dapat dikueri secara fleksibel, tetapi kueri relatif mahal dan tidak dapat diskalakan dengan baik dalam situasi lalu lintas tinggi (lihat [the section called “Pemodelan data”](#)).
- Dalam database NoSQL seperti Amazon Keyspaces, data dapat ditanyakan secara efisien dalam sejumlah cara, di luar mana kueri bisa mahal dan lambat.

Perbedaan ini membuat desain basis data menjadi berbeda di antara kedua sistem:

- Di RDBMS, Anda mendesain untuk fleksibilitas tanpa perlu mengkhawatirkan detail penerapan atau performa. Optimasi kueri umumnya tidak memengaruhi desain skema, tetapi normalisasi itu penting.
- Di Amazon Keyspaces, Anda mendesain skema Anda secara khusus untuk membuat kueri yang paling umum dan penting secepat dan semurah mungkin. Struktur data Anda disesuaikan dengan kebutuhan spesifik kasus penggunaan bisnis Anda.

## Dua konsep utama untuk desain NoSQL

Desain NoSQL membutuhkan pola pikir yang berbeda dari desain RDBMS. Untuk RDBMS, Anda dapat melanjutkan dan membuat model data yang dinormalisasi tanpa memikirkan pola akses. Anda kemudian dapat memperluasnya nanti ketika ada pertanyaan dan persyaratan kueri baru. Anda dapat mengatur setiap jenis data ke dalam tabelnya sendiri.

### Perbedaan desain NoSQL

- Sebaliknya, Anda tidak boleh mulai mendesain skema Anda untuk Amazon Keyspaces sampai Anda mengetahui pertanyaan yang perlu dijawab. Memahami masalah bisnis dan kasus penggunaan aplikasi di awal sangat penting.
- Anda harus memelihara tabel sesedikit mungkin dalam aplikasi Amazon Keyspaces. Memiliki lebih sedikit tabel membuat hal-hal lebih skalabel, memerlukan lebih sedikit manajemen izin, dan mengurangi biaya overhead untuk aplikasi Amazon Keyspaces Anda. Hal ini juga dapat membantu menjaga biaya pencadangan tetap rendah secara keseluruhan.



## Mendekati desain NoSQL

Langkah pertama dalam merancang aplikasi Amazon Keyspaces Anda adalah mengidentifikasi pola kueri spesifik yang harus dipenuhi oleh sistem.

Secara khusus, penting untuk memahami tiga properti dasar dari pola akses aplikasi Anda sebelum memulai:

- **Ukuran data:** Mengetahui berapa banyak data yang akan disimpan dan diminta sekaligus membantu menentukan cara paling efektif untuk mempartisi data.
- **Bentuk data:** Alih-alih membentuk kembali data saat kueri diproses (seperti yang dilakukan sistem RDBMS), basis data NoSQL mengatur data sehingga bentuknya dalam basis data tersebut sesuai dengan apa yang akan dikueri. Ini adalah faktor kunci dalam meningkatkan kecepatan dan skalabilitas.
- **Kecepatan data:** Amazon Keyspaces menskalakan dengan meningkatkan jumlah partisi fisik yang tersedia untuk memproses kueri, dan dengan mendistribusikan data secara efisien di seluruh partisi tersebut. Mengetahui sebelumnya apa beban kueri puncak mungkin membantu menentukan cara mempartisi data ke I/O kapasitas penggunaan terbaik.

Setelah mengidentifikasi persyaratan kueri tertentu, Anda bisa mengatur data menurut prinsip umum yang mengatur performa:

- **Menyimpan data terkait bersama-sama.** Penelitian tentang optimasi tabel perutean 20 tahun yang lalu menemukan bahwa "lokalitas referensi" adalah satu-satunya faktor terpenting dalam mempercepat waktu respons: menyimpan data terkait di satu tempat. Ini juga berlaku dalam sistem NoSQL saat ini, penyimpanan data terkait dalam jarak dekat memiliki dampak besar pada biaya dan performa. Alih-alih mendistribusikan item data terkait di beberapa tabel, Anda harus menyimpan item terkait di sistem NoSQL Anda sedekat mungkin.

Sebagai aturan umum, Anda harus memelihara tabel sesedikit mungkin dalam aplikasi Amazon Keyspaces.

Pengecualian adalah kasus yang melibatkan data deret waktu bervolume tinggi, atau set data yang memiliki pola akses yang sangat berbeda. Tabel tunggal dengan indeks terbalik biasanya dapat mengaktifkan kueri sederhana untuk membuat dan mengambil struktur data hierarki kompleks yang diperlukan oleh aplikasi Anda.

- Menggunakan urutan. Item terkait dapat dikelompokkan bersama dan dikueri secara efisien jika desain utamanya menyebabkan item tersebut disortir bersama. Ini adalah strategi desain NoSQL yang penting.
- Mendistribusikan kueri. Penting juga bahwa volume kueri yang tinggi tidak difokuskan pada satu bagian database, di mana mereka dapat melebihi I/O kapasitas. Sebagai gantinya, Anda harus mendesain kunci data untuk mendistribusikan lalu lintas secara merata di seluruh partisi sebanyak mungkin, menghindari "hot spot".

Prinsip-prinsip umum ini diterjemahkan ke dalam beberapa pola desain umum yang dapat Anda gunakan untuk memodelkan data secara efisien di Amazon Keyspaces.

## Optimalkan koneksi driver klien untuk lingkungan tanpa server

Untuk berkomunikasi dengan Amazon Keyspaces, Anda dapat menggunakan driver klien Apache Cassandra yang ada pilihan Anda. Karena Amazon Keyspaces adalah layanan tanpa server, kami menyarankan Anda mengoptimalkan konfigurasi koneksi driver klien Anda untuk kebutuhan throughput aplikasi Anda. Topik ini memperkenalkan praktik terbaik termasuk cara menghitung berapa banyak koneksi yang dibutuhkan aplikasi Anda, serta pemantauan dan penanganan kesalahan koneksi.

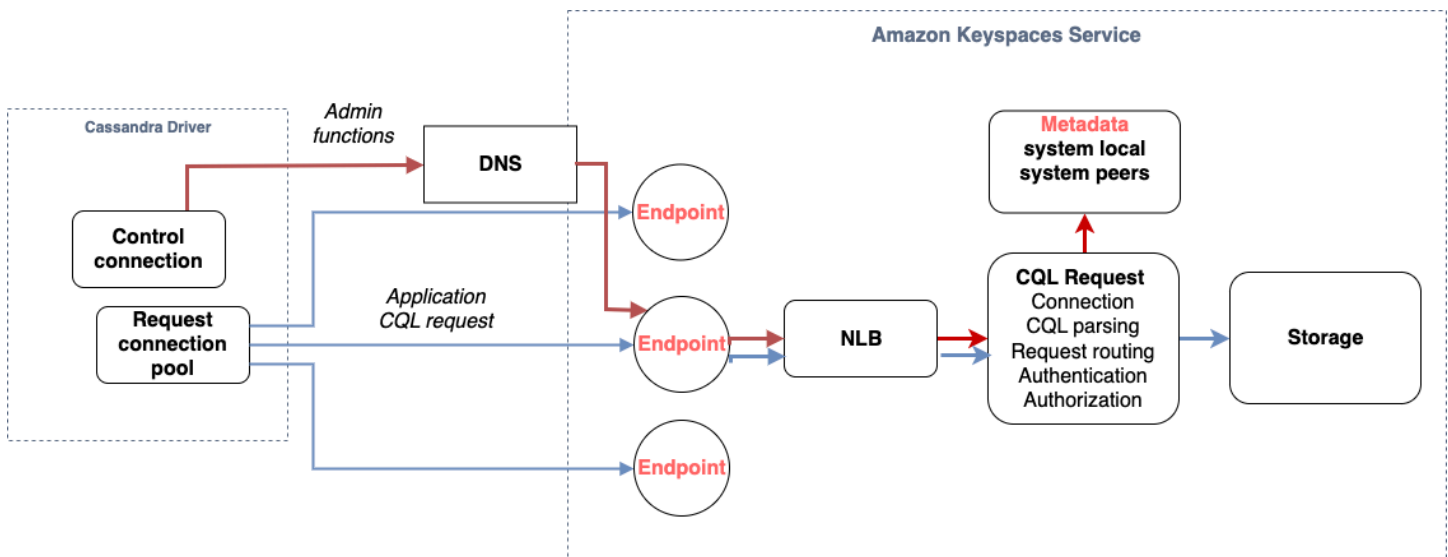
### Topik

- [Cara kerja koneksi di Amazon Keyspaces](#)
- [Cara mengonfigurasi koneksi di Amazon Keyspaces](#)
- [Cara mengonfigurasi kebijakan coba lagi untuk koneksi di Amazon Keyspaces](#)
- [Cara mengonfigurasi koneksi melalui titik akhir VPC di Amazon Keyspaces](#)
- [Cara memantau koneksi di Amazon Keyspaces](#)
- [Cara menangani kesalahan koneksi di Amazon Keyspaces](#)

## Cara kerja koneksi di Amazon Keyspaces

Bagian ini memberikan gambaran umum tentang cara kerja koneksi driver klien di Amazon Keyspaces. Karena kesalahan konfigurasi driver klien Cassandra dapat mengakibatkan peristiwa `PerConnectionRequestExceeded` di Amazon Keyspaces, mengonfigurasi jumlah koneksi yang tepat dalam konfigurasi driver klien diperlukan untuk menghindari kesalahan koneksi ini dan serupa.

Saat menyambungkan ke Amazon Keyspaces, driver memerlukan titik akhir seed untuk membuat koneksi awal. Amazon Keyspaces menggunakan DNS untuk merutekan koneksi awal ke salah satu dari banyak titik akhir yang tersedia. Titik akhir dilampirkan ke penyeimbang beban jaringan yang pada gilirannya membuat koneksi ke salah satu penanganan permintaan di armada. Setelah koneksi awal dibuat, driver klien mengumpulkan informasi tentang semua titik akhir yang tersedia dari tabel `system.peers`. Dengan informasi ini, driver klien dapat membuat koneksi tambahan ke titik akhir yang terdaftar. Jumlah koneksi yang dapat dibuat oleh driver klien dibatasi oleh jumlah koneksi lokal yang ditentukan dalam pengaturan driver klien. Secara default, sebagian besar driver klien membuat satu koneksi per titik akhir dan membuat kumpulan koneksi ke Cassandra dan memuat kueri keseimbangan melalui kumpulan koneksi tersebut. Meskipun beberapa koneksi dapat dibuat ke titik akhir yang sama, di belakang penyeimbang beban jaringan mereka mungkin terhubung ke banyak penanganan permintaan yang berbeda. Saat menghubungkan melalui titik akhir publik, membuat satu koneksi ke masing-masing dari sembilan titik akhir yang tercantum dalam tabel `system.peers` menghasilkan sembilan koneksi ke penanganan permintaan yang berbeda.



## Cara mengonfigurasi koneksi di Amazon Keyspaces

Amazon Keyspaces mendukung hingga 3.000 kueri CQL per koneksi TCP per detik. Karena tidak ada batasan jumlah koneksi yang dapat dibuat oleh pengemudi, kami sarankan untuk menargetkan hanya 500 permintaan CQL per detik per koneksi untuk memungkinkan overhead, ledakan lalu lintas, dan penyeimbangan beban yang lebih baik. Ikuti langkah-langkah ini untuk memastikan bahwa koneksi driver Anda dikonfigurasi dengan benar untuk kebutuhan aplikasi Anda.

Tingkatkan jumlah koneksi per alamat IP yang dipertahankan driver Anda di kolam koneksinya.

- Sebagian besar driver Cassandra membuat kumpulan koneksi ke Cassandra dan memuat kueri keseimbangan di atas kumpulan koneksi itu. Perilaku default sebagian besar driver adalah membuat satu koneksi ke setiap titik akhir. Amazon Keyspaces mengekspos sembilan alamat IP peer ke driver, jadi berdasarkan perilaku default sebagian besar driver, ini menghasilkan 9 koneksi. Amazon Keyspaces mendukung hingga 3.000 kueri CQL per koneksi TCP per detik, oleh karena itu, throughput kueri CQL maksimum driver yang menggunakan pengaturan default adalah 27.000 kueri CQL per detik. Jika Anda menggunakan pengaturan default driver, satu koneksi mungkin harus memproses lebih dari throughput kueri CQL maksimum 3.000 kueri CQL per detik. Ini bisa mengakibatkan `PerConnectionRequestExceeded` peristiwa.
- Untuk menghindari `PerConnectionRequestExceeded` kejadian, Anda harus mengonfigurasi driver untuk membuat koneksi tambahan per titik akhir untuk mendistribusikan throughput.
- Sebagai praktik terbaik di Amazon Keyspaces, asumsikan bahwa setiap koneksi dapat mendukung 500 kueri CQL per detik.
- Itu berarti bahwa untuk aplikasi produksi yang perlu mendukung sekitar 27.000 kueri CQL per detik yang didistribusikan di sembilan titik akhir yang tersedia, Anda harus mengonfigurasi enam koneksi per titik akhir. Ini memastikan bahwa setiap koneksi memproses tidak lebih dari 500 permintaan per detik.

Hitung jumlah koneksi per alamat IP yang perlu Anda konfigurasikan untuk driver Anda berdasarkan kebutuhan aplikasi Anda.

Untuk menentukan jumlah koneksi yang perlu Anda konfigurasikan per titik akhir untuk aplikasi Anda, pertimbangkan contoh berikut. Anda memiliki aplikasi yang perlu mendukung 20.000 kueri CQL per detik yang terdiri dari 10.000 `INSERT`, 5.000 `SELECT`, dan 5.000 operasi `DELETE`. Aplikasi Java berjalan pada tiga instance di Amazon Elastic Container Service (Amazon ECS) di mana setiap instance menetapkan satu sesi ke Amazon Keyspaces. Perhitungan yang dapat Anda gunakan untuk memperkirakan berapa banyak koneksi yang perlu Anda konfigurasi untuk driver Anda menggunakan input berikut.

1. Jumlah permintaan per detik aplikasi Anda perlu mendukung.
2. Jumlah instance yang tersedia dengan satu dikurangi untuk memperhitungkan pemeliharaan atau kegagalan.
3. Jumlah titik akhir yang tersedia. Jika Anda terhubung melalui titik akhir publik, Anda memiliki sembilan titik akhir yang tersedia. Jika Anda menggunakan titik akhir VPC, Anda memiliki antara dua dan lima titik akhir yang tersedia, tergantung pada Wilayah.
4. Gunakan 500 kueri CQL per detik per koneksi sebagai praktik terbaik untuk Amazon Keyspaces.

## 5. Bulatkan hasilnya.

Untuk contoh ini, rumusnya terlihat seperti ini.

```
20,000 CQL queries / (3 instances - 1 failure) / 9 public endpoints / 500 CQL queries
per second = ROUND(2.22) = 3
```

Berdasarkan perhitungan ini, Anda perlu menentukan tiga koneksi lokal per titik akhir dalam konfigurasi driver. Untuk koneksi jarak jauh, konfigurasikan hanya satu koneksi per titik akhir.

## Cara mengonfigurasi kebijakan coba lagi untuk koneksi di Amazon Keyspaces

Saat mengonfigurasi kebijakan coba lagi untuk koneksi ke Amazon Keyspaces, sebaiknya Anda menerapkan kebijakan coba ulang Amazon Keyspaces.

`AmazonKeyspacesExponentialRetryPolicy` Kebijakan coba lagi ini lebih cocok untuk mencoba lagi di berbagai koneksi ke Amazon Keyspaces daripada driver. `DefaultRetryPolicy`

Dengan `AmazonKeyspacesExponentialRetryPolicy`, Anda dapat mengonfigurasi jumlah upaya coba lagi untuk koneksi yang memenuhi kebutuhan Anda. Secara default, jumlah percobaan ulang untuk `AmazonKeyspacesExponentialRetryPolicy` diatur ke 3.

Keuntungan tambahan adalah kebijakan coba ulang Amazon Keyspaces meneruskan kembali pengecualian asli yang dikembalikan oleh layanan, yang menunjukkan mengapa upaya permintaan gagal. Kebijakan coba ulang default hanya menampilkan generik `NoHostAvailableException`, yang mungkin menyembunyikan wawasan tentang kegagalan permintaan.

Untuk mengonfigurasi kebijakan coba ulang permintaan menggunakan `AmazonKeyspacesExponentialRetryPolicy`, sebaiknya Anda mengonfigurasi sejumlah kecil percobaan ulang, dan menangani pengecualian yang dikembalikan dalam kode aplikasi Anda.

Untuk contoh kode yang menerapkan kebijakan coba lagi, lihat kebijakan coba ulang [Amazon Keyspaces di Github](#).

## Cara mengonfigurasi koneksi melalui titik akhir VPC di Amazon Keyspaces

Saat menghubungkan melalui titik akhir VPC pribadi, kemungkinan besar Anda memiliki 3 titik akhir yang tersedia. Jumlah titik akhir VPC dapat berbeda per Wilayah, berdasarkan jumlah Availability

Zone, dan jumlah subnet dalam VPC yang ditetapkan. Wilayah AS Timur (Virginia N.) memiliki lima Availability Zone dan Anda dapat memiliki hingga lima titik akhir Amazon Keyspaces. Wilayah AS Barat (California Utara) memiliki dua Availability Zone dan Anda dapat memiliki hingga dua titik akhir Amazon Keyspaces. Jumlah titik akhir tidak memengaruhi skala, tetapi meningkatkan jumlah koneksi yang perlu Anda buat dalam konfigurasi driver. Pertimbangkan contoh berikut. Aplikasi Anda perlu mendukung 20.000 kueri CQL dan berjalan pada tiga instans di Amazon ECS di mana setiap instans menetapkan satu sesi ke Amazon Keyspaces. Satu-satunya perbedaan adalah berapa banyak titik akhir yang tersedia di berbagai Wilayah AWS titik.

Koneksi yang diperlukan di Wilayah AS Timur (Virginia N.):

```
20,000 CQL queries / (3 instances - 1 failure) / 5 private VPC endpoints / 500 CQL queries per second = 4 local connections
```

Koneksi yang diperlukan di Wilayah AS Barat (California Utara):

```
20,000 CQL queries / (3 instances - 1 failure) / 2 private VPC endpoints / 500 CQL queries per second = 10 local connections
```

#### Important

Saat menggunakan titik akhir VPC pribadi, izin tambahan diperlukan untuk Amazon Keyspaces untuk menemukan titik akhir VPC yang tersedia secara dinamis dan mengisi tabel. `system.peers` Untuk informasi selengkapnya, lihat [the section called “Mengisi entri `system.peers` tabel dengan informasi titik akhir VPC antarmuka”](#).

Saat mengakses Amazon Keyspaces melalui titik akhir VPC pribadi menggunakan yang Akun AWS berbeda, kemungkinan Anda hanya melihat satu titik akhir Amazon Keyspaces. Sekali lagi ini tidak memengaruhi skala kemungkinan throughput ke Amazon Keyspaces, tetapi mungkin mengharuskan Anda untuk meningkatkan jumlah koneksi dalam konfigurasi driver Anda. Contoh ini menunjukkan perhitungan yang sama untuk satu titik akhir yang tersedia.

```
20,000 CQL queries / (3 instances - 1 failure) / 1 private VPC endpoints / 500 CQL queries per second = 20 local connections
```

Untuk mempelajari lebih lanjut tentang akses lintas akun ke Amazon Keyspaces menggunakan VPC bersama, lihat. [the section called “Konfigurasi akses lintas akun di VPC bersama”](#)

## Cara memantau koneksi di Amazon Keyspaces

Untuk membantu mengidentifikasi jumlah titik akhir yang terhubung dengan aplikasi Anda, Anda dapat mencatat jumlah rekan yang ditemukan dalam `system.peers` tabel. Contoh berikut adalah contoh kode Java yang mencetak jumlah peer setelah koneksi dibuat.

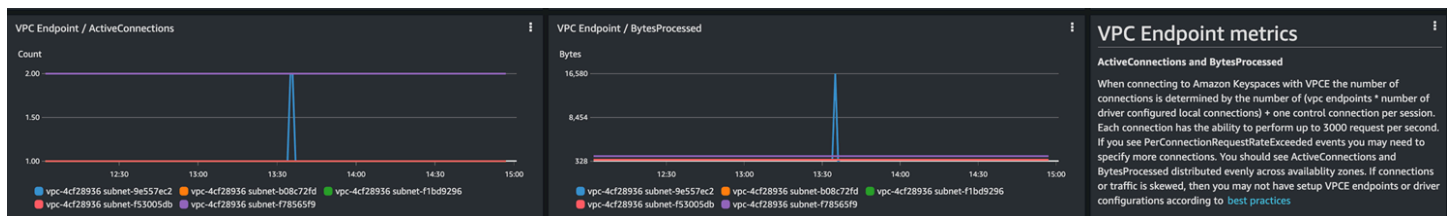
```
ResultSet result = session.execute(new SimpleStatement("SELECT * FROM system.peers"));

logger.info("number of Amazon Keyspaces endpoints:" + result.all().stream().count());
```

### Note

Konsol atau AWS konsol CQL tidak digunakan dalam VPC dan oleh karena itu menggunakan titik akhir publik. Akibatnya, menjalankan `system.peers` kueri dari aplikasi yang terletak di luar VPCE sering menghasilkan 9 rekan. Mungkin juga bermanfaat untuk mencetak alamat IP masing-masing rekan.

Anda juga dapat mengamati jumlah peer saat menggunakan titik akhir VPC dengan menyiapkan metrik Amazon VPCE. CloudWatch Di CloudWatch, Anda dapat melihat jumlah koneksi yang dibuat ke titik akhir VPC. Driver Cassandra membuat koneksi untuk setiap titik akhir untuk mengirim kueri CQL dan koneksi kontrol untuk mengumpulkan informasi tabel sistem. Gambar di bawah ini menunjukkan CloudWatch metrik titik akhir VPC setelah tersambung ke Amazon Keyspaces dengan 1 koneksi yang dikonfigurasi dalam pengaturan driver. Metrik menunjukkan enam koneksi aktif yang terdiri dari satu koneksi kontrol dan lima koneksi (1 per titik akhir di seluruh Availability Zones).



Untuk memulai memantau jumlah koneksi menggunakan CloudWatch grafik, Anda dapat menerapkan CloudFormation template ini yang tersedia GitHub di repositori template [Amazon Keyspaces](#).

## Cara menangani kesalahan koneksi di Amazon Keyspaces

Saat melebihi 3.000 permintaan per kuota koneksi, Amazon Keyspaces mengembalikan suatu `PerConnectionRequestExceeded` peristiwa dan driver Cassandra menerima atau pengecualian. `WriteTimeout` `ReadTimeout` Anda harus mencoba lagi pengecualian ini dengan dukungan eksponensial dalam kebijakan percobaan ulang Cassandra Anda atau dalam aplikasi Anda. Anda harus memberikan backoff eksponensial untuk menghindari pengiriman permintaan tambahan.

Kebijakan coba ulang default mencoba `try next host` dalam paket kueri. Karena Amazon Keyspaces mungkin memiliki satu hingga tiga titik akhir yang tersedia saat menghubungkan ke titik akhir VPC, Anda juga dapat melihat tambahan `WriteTimeout` dan `ReadTimeout` pengecualian `NoHostAvailableException` di log aplikasi Anda. Anda dapat menggunakan kebijakan coba ulang yang disediakan Amazon Keyspaces, yang mencoba lagi pada titik akhir yang sama tetapi di koneksi yang berbeda.

Anda dapat menemukan contoh kebijakan percobaan ulang eksponensial untuk Java di repositori contoh GitHub kode Java Amazon [Keyspaces](#). Anda dapat menemukan contoh bahasa tambahan di Github di repositori contoh [kode Amazon Keyspaces](#).

## Praktik terbaik pemodelan data: rekomendasi untuk merancang model data

Pemodelan data yang efektif sangat penting untuk mengoptimalkan kinerja dan meminimalkan biaya saat bekerja dengan Amazon Keyspaces (untuk Apache Cassandra). Topik ini mencakup pertimbangan dan rekomendasi utama untuk merancang model data yang sesuai dengan pola akses data aplikasi Anda.

- **Desain Kunci Partisi** — Kunci partisi memainkan peran penting dalam menentukan bagaimana data didistribusikan di seluruh partisi di Amazon Keyspaces. Memilih kunci partisi yang sesuai dapat secara signifikan memengaruhi kinerja kueri dan biaya throughput. Bagian ini membahas strategi untuk merancang kunci partisi yang mempromosikan distribusi aktivitas baca dan tulis yang merata di seluruh partisi.
- **Pertimbangan Utama:**
  - **Distribusi aktivitas seragam** — Bertujuan untuk aktivitas baca dan tulis yang seragam di semua partisi untuk meminimalkan biaya throughput dan memanfaatkan kapasitas burst secara efektif.
  - **Pola akses** - Sejajarkan desain kunci partisi Anda dengan pola akses data utama aplikasi Anda.



- Ukuran partisi — Hindari membuat partisi yang tumbuh terlalu besar, karena ini dapat memengaruhi kinerja dan meningkatkan biaya.

Untuk memvisualisasikan dan mendesain model data dengan lebih mudah, Anda dapat menggunakan [NoSQL Workbench](#).

## Topik

- [Cara menggunakan kunci partisi secara efektif di Amazon Keyspaces](#)

# Cara menggunakan kunci partisi secara efektif di Amazon Keyspaces

Kunci primer yang secara unik mengidentifikasi setiap baris dalam tabel Amazon Keyspaces dapat terdiri dari satu atau beberapa kolom kunci partisi, yang menentukan partisi mana data disimpan, dan satu atau lebih kolom pengelompokan opsional, yang menentukan bagaimana data dikelompokkan dan diurutkan dalam partisi.

Karena kunci partisi menetapkan jumlah partisi data Anda disimpan dan bagaimana data didistribusikan di seluruh partisi ini, bagaimana Anda memilih kunci partisi Anda dapat memiliki dampak yang signifikan pada kinerja kueri Anda. Secara umum, Anda harus mendesain aplikasi Anda untuk aktivitas seragam di semua partisi pada disk.

Mendistribusikan aktivitas baca dan tulis aplikasi Anda secara merata di semua partisi membantu meminimalkan biaya throughput dan ini berlaku untuk mode kapasitas sesuai permintaan serta yang disediakan. read/write Misalnya, jika Anda menggunakan mode kapasitas yang disediakan, Anda dapat menentukan pola akses yang dibutuhkan aplikasi Anda, dan memperkirakan total unit kapasitas baca (RCU) dan unit kapasitas tulis (WCU) yang dibutuhkan setiap tabel. Amazon Keyspaces mendukung pola akses Anda menggunakan throughput yang Anda berikan selama lalu lintas terhadap partisi tertentu tidak melebihi 3.000 RCU dan 1.000 WCU.

Ketika partisi mengalami throughput baca atau tulis yang tinggi, tergantung pada pola lalu lintas Amazon Keyspaces dapat secara otomatis membagi partisi menjadi dua partisi baru. Setiap partisi baru berisi subset dari baris partisi asli, mendistribusikan throughput secara merata di kedua partisi.

Amazon Keyspaces menawarkan fleksibilitas tambahan dalam penyediaan throughput per partisi Anda dengan menyediakan kapasitas burst, untuk informasi selengkapnya lihat. [the section called “Gunakan kapasitas burst”](#)

## Topik

- [Gunakan sharding tulis untuk mendistribusikan beban kerja secara merata di seluruh partisi](#)

## Gunakan sharding tulis untuk mendistribusikan beban kerja secara merata di seluruh partisi

Salah satu cara untuk mendistribusikan tulisan dengan lebih baik di seluruh partisi di Amazon Keyspaces adalah dengan memperluas ruang. Hal ini dapat dilakukan dengan berbagai cara. Anda dapat menambahkan kolom kunci partisi tambahan tempat Anda menulis angka acak untuk mendistribusikan baris di antara partisi. Atau, Anda dapat menggunakan angka yang dihitung berdasarkan sesuatu yang Anda kueri.

Sharding menggunakan kunci partisi majemuk dan nilai acak

Salah satu strategi untuk mendistribusikan beban secara lebih merata di seluruh partisi adalah dengan menambahkan kolom kunci partisi tambahan tempat Anda menulis angka acak. Kemudian Anda mengacak penulisan di ruang yang lebih besar.

Misalnya, perhatikan tabel berikut yang memiliki kunci partisi tunggal yang mewakili tanggal.

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
 publish_date date,
 title text,
 description int,
 PRIMARY KEY (publish_date));
```

Untuk lebih merata mendistribusikan tabel ini di seluruh partisi, Anda dapat menyertakan kolom kunci partisi tambahan shard yang menyimpan angka acak. Contoh:

```
CREATE TABLE IF NOT EXISTS tracker.blogs (
 publish_date date,
 shard int,
 title text,
 description int,
 PRIMARY KEY ((publish_date, shard)));
```

Saat memasukkan data, Anda mungkin memilih nomor acak antara 1 dan 200 untuk shard kolom. Ini menghasilkan nilai kunci partisi majemuk seperti (2020-07-09, 1)(2020-07-09, 2),, dan seterusnya, melalui (2020-07-09, 200). Karena Anda mengacak kunci partisi, penulisan ke tabel pada setiap hari tersebar merata di sejumlah partisi. Hal ini menghasilkan paralelisme yang lebih baik dan throughput keseluruhan yang lebih tinggi.

Namun, untuk membaca semua baris untuk hari tertentu, Anda harus menanyakan baris untuk semua pecahan dan kemudian menggabungkan hasilnya. Misalnya, pertama-tama Anda akan mengeluarkan `SELECT` pernyataan untuk nilai kunci partisi (`2020-07-09, 1`). Kemudian keluarkan `SELECT` pernyataan lain untuk (`2020-07-09, 2`), dan seterusnya, melalui (`2020-07-09, 200`). Akhirnya, aplikasi Anda harus menggabungkan hasil dari semua `SELECT` pernyataan tersebut.

Sharding menggunakan kunci partisi majemuk dan nilai yang dihitung

Strategi pengacakan dapat meningkatkan throughput tulis secara signifikan. Tetapi sulit untuk membaca baris tertentu karena Anda tidak tahu nilai mana yang ditulis ke `shard` kolom ketika baris itu ditulis. Untuk membuatnya lebih mudah untuk membaca baris individu, Anda dapat menggunakan strategi yang berbeda. Alih-alih menggunakan nomor acak untuk mendistribusikan baris di antara partisi, gunakan angka yang dapat Anda hitung berdasarkan sesuatu yang ingin Anda kueri.

Pertimbangkan contoh sebelumnya, yaitu tabel menggunakan tanggal hari ini dalam kunci partisi. Sekarang anggaplah setiap baris memiliki `title` kolom yang dapat diakses, dan Anda paling sering perlu menemukan baris berdasarkan judul selain tanggal. Sebelum aplikasi Anda menulis baris ke tabel, itu bisa menghitung nilai hash berdasarkan judul dan menggunakannya untuk mengisi kolom. `shard` Penghitungannya dapat menghasilkan angka antara 1 dan 200 yang terdistribusi cukup merata, mirip dengan yang dihasilkan strategi acak.

Perhitungan sederhana kemungkinan akan cukup, seperti produk dari nilai titik UTF-8 kode untuk karakter dalam judul, modulo 200, + 1. Nilai kunci partisi majemuk kemudian akan menjadi kombinasi tanggal dan hasil perhitungan.

Dengan strategi ini, penulisan tersebar merata di seluruh nilai kunci partisi, serta di partisi fisik. Anda dapat dengan mudah melakukan `SELECT` pernyataan untuk baris dan tanggal tertentu karena Anda dapat menghitung nilai kunci partisi untuk `title` nilai tertentu.

Untuk membaca semua baris untuk hari tertentu, Anda masih harus `SELECT` masing-masing (`2020-07-09, N`) kunci (di mana `N` 1-200), dan aplikasi Anda kemudian harus menggabungkan semua hasil. Manfaatnya adalah Anda menghindari satu nilai kunci partisi "panas" yang mengambil semua beban kerja.

## Mengoptimalkan biaya tabel Amazon Keyspaces

Bagian ini mencakup praktik terbaik tentang cara mengoptimalkan biaya untuk tabel Amazon Keyspaces yang ada. Anda harus melihat strategi berikut untuk mengetahui strategi pengoptimalan biaya yang paling sesuai dengan kebutuhan Anda dan mendekatinya secara berulang. Setiap strategi

memberikan gambaran umum tentang apa yang mungkin memengaruhi biaya Anda, cara mencari peluang untuk mengoptimalkan biaya, dan panduan preskriptif tentang cara menerapkan praktik terbaik ini untuk membantu Anda menghemat.

## Topik

- [Evaluasi biaya Anda di tingkat tabel](#)
- [Evaluasi mode kapasitas tabel Anda](#)
- [Evaluasi pengaturan Application Auto Scaling tabel Anda](#)
- [Identifikasi sumber daya yang tidak digunakan untuk mengoptimalkan biaya di Amazon Keyspaces](#)
- [Evaluasi pola penggunaan tabel Anda untuk mengoptimalkan kinerja dan biaya](#)
- [Evaluasi kapasitas yang disediakan untuk penyediaan ukuran yang tepat](#)

## Evaluasi biaya Anda di tingkat tabel

Alat Cost Explorer yang ditemukan di dalamnya Konsol Manajemen AWS memungkinkan Anda melihat biaya yang dipecah berdasarkan jenis, misalnya biaya baca, tulis, penyimpanan, dan cadangan. Anda juga dapat melihat biaya-biaya ini dirangkum berdasarkan periode seperti bulan atau hari.

Salah satu tantangan umum dengan Cost Explorer adalah Anda tidak dapat meninjau biaya hanya satu tabel tertentu dengan mudah, karena Cost Explorer tidak memungkinkan Anda memfilter atau mengelompokkan berdasarkan biaya tabel tertentu. Anda dapat melihat ukuran tabel Billable metrik (Byte) dari setiap tabel di konsol Amazon Keyspaces pada tab Monitor tabel. Jika Anda memerlukan lebih banyak informasi terkait biaya per tabel, bagian ini menunjukkan cara menggunakan [penandaan](#) untuk melakukan analisis biaya tabel individual di Cost Explorer.

## Topik

- [Cara melihat biaya satu tabel Amazon Keyspaces](#)
- [Tampilan default Cost Explorer](#)
- [Cara menggunakan dan menerapkan tanda tabel di Cost Explorer](#)

## Cara melihat biaya satu tabel Amazon Keyspaces

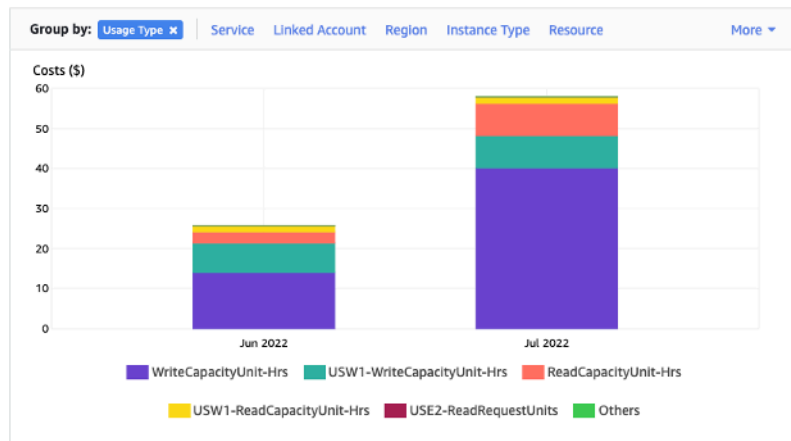
Anda dapat melihat informasi dasar tentang tabel Amazon Keyspaces di konsol, termasuk skema kunci utama, ukuran tabel yang dapat ditagih, dan metrik terkait kapasitas. Anda dapat menggunakan

ukuran tabel untuk menghitung biaya penyimpanan bulanan untuk tabel. Misalnya, \$0,25 per GB di AS Timur (Virginia N.). Wilayah AWS

Jika tabel menggunakan mode kapasitas yang disediakan, pengaturan unit kapasitas baca saat ini (RCU) dan unit kapasitas tulis (WCU) dikembalikan juga. Anda dapat menggunakan informasi ini untuk menghitung biaya baca dan tulis saat ini untuk tabel. Perhatikan bahwa biaya ini dapat berubah, terutama jika Anda telah mengonfigurasi tabel dengan penskalaan otomatis Amazon Keyspaces.

## Tampilan default Cost Explorer

Tampilan default di Cost Explorer menyediakan bagan yang menunjukkan biaya sumber daya yang dikonsumsi, misalnya throughput dan penyimpanan. Anda dapat memilih untuk mengelompokkan biaya ini berdasarkan periode, seperti total berdasarkan bulan atau hari. Biaya penyimpanan, membaca, menulis, dan kategori lainnya dapat dipecah dan dibandingkan juga.



## Cara menggunakan dan menerapkan tanda tabel di Cost Explorer

Secara default, Cost Explorer tidak memberikan ringkasan biaya untuk satu tabel tertentu, karena menggabungkan biaya beberapa tabel menjadi total. Namun, Anda dapat menggunakan [penandaan sumber daya AWS](#) untuk mengidentifikasi setiap tabel dengan tanda metadata. Tag adalah pasangan nilai kunci yang dapat Anda gunakan untuk berbagai tujuan, misalnya untuk mengidentifikasi semua sumber daya milik proyek atau departemen. Untuk informasi selengkapnya, lihat [the section called “Bekerja dengan tag”](#).

Untuk contoh ini, kami menggunakan tabel dengan nama MyTable.

1. Tetapkan tag dengan kunci `table_name` dan nilai `MyTable`

2. [Aktifkan tanda di dalam Cost Explorer](#) lalu filter pada nilai tanda untuk mendapatkan lebih banyak visibilitas ke setiap biaya tabel.

 Note

Tanda mungkin akan mulai muncul di Cost Explorer setelah satu atau dua hari

Anda dapat mengatur sendiri tag metadata di konsol, atau secara terprogram dengan CQL, the, atau SDK. AWS CLI AWS Pertimbangkan untuk mewajibkan tanda `table_name` ditetapkan sebagai bagian dari proses pembuatan tabel baru organisasi Anda. Untuk informasi selengkapnya, lihat [the section called “Buat laporan alokasi biaya”](#).

## Evaluasi mode kapasitas tabel Anda

Bagian ini memberikan gambaran umum tentang cara memilih mode kapasitas yang sesuai untuk tabel Amazon Keyspaces Anda. Setiap mode disesuaikan untuk memenuhi kebutuhan beban kerja yang berbeda dalam hal respons terhadap perubahan throughput, serta cara penagihan penggunaan tersebut. Anda harus menyeimbangkan faktor-faktor ini saat membuat keputusan.

### Topik

- [Mode kapasitas tabel yang tersedia](#)
- [Kapan harus memilih mode kapasitas sesuai permintaan](#)
- [Kapan harus mode kapasitas yang disediakan](#)
- [Faktor lain yang perlu dipertimbangkan saat memilih mode kapasitas tabel](#)

### Mode kapasitas tabel yang tersedia

Saat membuat tabel Amazon Keyspaces, Anda harus memilih mode kapasitas sesuai permintaan atau yang disediakan. Untuk informasi selengkapnya, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).

### On-demand modus kapasitas

Mode kapasitas sesuai permintaan dirancang untuk menghilangkan kebutuhan untuk merencanakan atau menyediakan kapasitas tabel Amazon Keyspaces Anda. Dalam mode ini, tabel Anda langsung

mengakomodasi permintaan tanpa perlu menskalakan sumber daya apa pun ke atas atau ke bawah (hingga dua kali throughput puncak tabel sebelumnya).

On-demand tabel ditagih dengan menghitung jumlah permintaan aktual terhadap tabel, jadi Anda hanya membayar untuk apa yang Anda gunakan daripada apa yang telah disediakan.

### Tabel kapasitas yang disediakan

Mode kapasitas yang disediakan adalah model yang lebih tradisional di mana Anda dapat menentukan berapa banyak kapasitas tabel yang tersedia untuk permintaan baik secara langsung atau dengan bantuan Application Auto Scaling. Karena kapasitas tertentu disediakan untuk tabel pada waktu tertentu, penagihan didasarkan pada kapasitas yang disediakan, bukan jumlah permintaan. Melebihi kapasitas yang dialokasikan juga dapat menyebabkan tabel menolak permintaan dan mengurangi pengalaman pengguna aplikasi Anda.

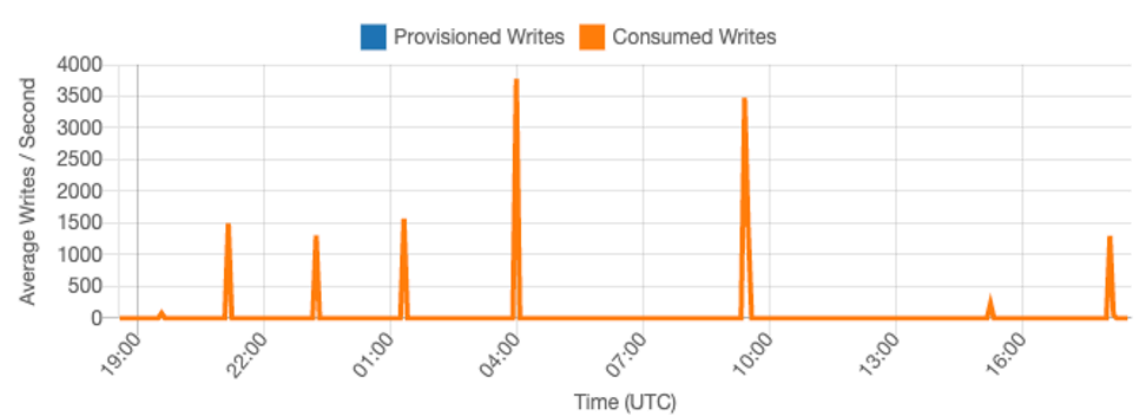
Mode kapasitas yang disediakan memerlukan keseimbangan antara tidak penyediaan berlebihan atau di bawah penyediaan tabel untuk mencapai keduanya, rendahnya terjadinya kesalahan kapasitas throughput yang tidak mencukupi, dan biaya yang dioptimalkan.

### Kapan harus memilih mode kapasitas sesuai permintaan

Saat mengoptimalkan biaya, mode sesuai permintaan adalah pilihan terbaik Anda ketika Anda memiliki beban kerja yang tidak terduga mirip dengan yang ditunjukkan pada grafik berikut.

Faktor-faktor ini berkontribusi pada jenis beban kerja ini:

- Waktu permintaan yang tidak dapat diprediksi (mengakibatkan lonjakan lalu lintas)
- Volume permintaan variabel (dihasilkan dari beban kerja batch)
- Turun ke nol atau di bawah 18% dari puncak selama satu jam tertentu (dihasilkan dari lingkungan pengembangan atau pengujian)



Untuk beban kerja dengan karakteristik di atas, menggunakan Application Auto Scaling untuk mempertahankan kapasitas yang cukup bagi tabel untuk merespons lonjakan lalu lintas dapat menyebabkan hasil yang tidak diinginkan. Entah tabel dapat disediakan secara berlebihan dan biaya lebih dari yang diperlukan, atau tabel dapat disediakan dan permintaan menyebabkan kesalahan throughput kapasitas rendah yang tidak perlu. Dalam kasus seperti ini, tabel berdasarkan permintaan adalah pilihan yang lebih baik.

Karena tabel berdasarkan permintaan ditagih berdasarkan permintaan, tidak ada lagi yang perlu Anda lakukan di tingkat tabel untuk mengoptimalkan biaya. Anda harus secara teratur mengevaluasi tabel sesuai permintaan Anda untuk memverifikasi beban kerja masih memiliki karakteristik di atas. Jika beban kerja telah stabil, pertimbangkan untuk mengubah ke mode yang disediakan untuk mempertahankan optimalisasi biaya.

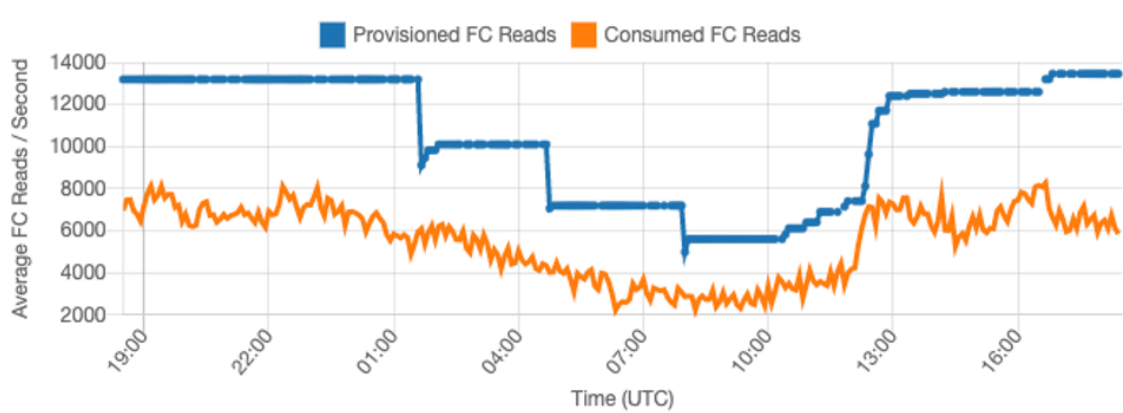
## Kapan harus mode kapasitas yang disediakan

Beban kerja yang ideal untuk mode kapasitas yang disediakan adalah beban kerja dengan pola penggunaan yang lebih dapat diprediksi seperti yang ditunjukkan pada grafik di bawah ini.

Faktor-faktor berikut berkontribusi pada beban kerja yang dapat diprediksi:

- Predictable/cyclical lalu lintas untuk jam atau hari tertentu
- Lonjakan lalu lintas jangka pendek terbatas





Karena volume lalu lintas dalam waktu atau hari tertentu lebih stabil, Anda dapat mengatur kapasitas yang disediakan relatif dekat dengan kapasitas konsumsi tabel yang sebenarnya. Pengoptimalan biaya tabel kapasitas yang disediakan pada akhirnya merupakan latihan untuk mendapatkan kapasitas yang disediakan (garis biru) sedekat mungkin dengan kapasitas yang dikonsumsi (garis oranye) tanpa meningkatkan `ThrottledRequests` peristiwa untuk tabel. Ruang antara kedua jalur adalah kapasitas yang terbuang serta asuransi terhadap pengalaman pengguna yang buruk karena kesalahan kapasitas throughput yang tidak mencukupi.

Amazon Keyspaces menyediakan Application Auto Scaling untuk tabel kapasitas yang disediakan, yang secara otomatis menyeimbangkannya atas nama Anda. Anda dapat melacak kapasitas yang dikonsumsi sepanjang hari dan mengonfigurasi kapasitas tabel yang disediakan berdasarkan beberapa variabel.

### Unit kapasitas minimum

Anda dapat mengatur kapasitas minimum tabel untuk membatasi terjadinya kesalahan kapasitas throughput yang tidak mencukupi, tetapi itu tidak mengurangi biaya tabel. Jika tabel Anda memiliki periode penggunaan rendah diikuti dengan ledakan penggunaan tinggi yang tiba-tiba, pengaturan minimum dapat mencegah Application Auto Scaling mengatur kapasitas tabel terlalu rendah.

### Unit kapasitas maksimum

Anda dapat mengatur kapasitas tabel maksimum untuk membatasi penskalaan tabel yang lebih tinggi dari yang dimaksudkan. Pertimbangkan untuk menerapkan maksimum untuk tabel pengembangan atau pengujian, di mana pengujian beban skala besar tidak diinginkan. Anda dapat mengatur maksimum untuk tabel apa pun, tetapi pastikan untuk mengevaluasi pengaturan ini secara teratur terhadap baseline tabel saat menggunakannya dalam produksi, untuk mencegah kesalahan kapasitas throughput yang tidak disengaja.

### Pemanfaatan target

Menetapkan pemanfaatan target pada tabel adalah cara utama pengoptimalan biaya untuk kapasitas tabel yang disediakan. Menetapkan nilai persen yang lebih rendah di sini meningkatkan berapa banyak tabel yang disediakan secara berlebihan, meningkatkan biaya, tetapi mengurangi risiko kesalahan kapasitas throughput yang tidak mencukupi. Menetapkan nilai persentase yang lebih tinggi berkurang dengan seberapa banyak tabel disediakan secara berlebihan, tetapi meningkatkan risiko kesalahan kapasitas throughput yang tidak mencukupi.

## Faktor lain yang perlu dipertimbangkan saat memilih mode kapasitas tabel

Saat memutuskan antara dua mode kapasitas, ada beberapa faktor tambahan yang perlu dipertimbangkan.

Saat memutuskan antara dua mode tabel, pertimbangkan seberapa besar diskon tambahan ini memengaruhi biaya tabel. Dalam banyak kasus, bahkan beban kerja yang relatif tidak dapat diprediksi dapat lebih hemat biaya untuk dijalankan pada tabel kapasitas yang disediakan secara berlebihan dengan kapasitas cadangan.

### Meningkatkan prediktabilitas beban kerja Anda

Dalam beberapa situasi, beban kerja tampaknya memiliki keduanya, pola yang dapat diprediksi dan tidak dapat diprediksi. Meskipun ini dapat dengan mudah didukung dengan tabel sesuai permintaan, biaya kemungkinan akan lebih rendah jika pola beban kerja yang tidak terduga dapat ditingkatkan.

Salah satu penyebab paling umum dari pola ini adalah impor batch. Jenis lalu lintas ini seringkali dapat melebihi kapasitas dasar tabel sedemikian rupa sehingga kesalahan kapasitas throughput yang tidak mencukupi akan terjadi jika dijalankan. Agar beban kerja seperti ini tetap berjalan pada kapasitas tabel yang disediakan, pertimbangkan opsi berikut:

- Jika batch terjadi pada waktu yang dijadwalkan, Anda dapat menjadwalkan peningkatan kapasitas penskalaan otomatis aplikasi Anda sebelum dijalankan.
- Jika batch terjadi secara acak, pertimbangkan untuk mencoba memperpanjang waktu yang diperlukan untuk menjalankan daripada mengeksekusi secepat mungkin.
- Tambahkan periode ramp up ke impor, di mana kecepatan impor mulai kecil tetapi perlahan-lahan meningkat selama beberapa menit sampai Application Auto Scaling memiliki kesempatan untuk mulai menyesuaikan kapasitas tabel.

## Evaluasi pengaturan Application Auto Scaling tabel Anda

Bagian ini memberikan gambaran umum tentang cara mengevaluasi pengaturan Application Auto Scaling pada tabel Amazon Keyspaces Anda. [Amazon Keyspaces Application Auto Scaling](#) adalah fitur yang mengelola throughput tabel berdasarkan lalu lintas aplikasi dan metrik pemanfaatan target Anda. Ini memastikan tabel Anda memiliki kapasitas yang diperlukan untuk pola aplikasi Anda.

Layanan Application Auto Scaling memantau pemanfaatan tabel Anda saat ini dan membandingkannya dengan nilai pemanfaatan target. `TargetValue` Ini memberi tahu Anda jika sudah waktunya untuk menambah atau mengurangi kapasitas yang dialokasikan.

### Topik

- [Memahami pengaturan Application Auto Scaling](#)
- [Cara mengidentifikasi tabel dengan pemanfaatan target rendah \( \$\leq 50\%\$ \)](#)
- [Cara mengatasi beban kerja dengan varian musiman](#)
- [Cara mengatasi lonjakan beban kerja dengan pola yang tidak diketahui](#)
- [Cara mengatasi beban kerja dengan aplikasi tertaut](#)

## Memahami pengaturan Application Auto Scaling

Mendefinisikan nilai yang benar untuk pemanfaatan target, langkah awal, dan nilai akhir adalah aktivitas yang memerlukan keterlibatan dari tim operasi Anda. Ini memungkinkan Anda untuk menentukan nilai dengan benar berdasarkan penggunaan aplikasi historis, yang digunakan untuk memicu kebijakan Application Auto Scaling. Target pemanfaatan adalah persentase dari total kapasitas Anda yang perlu dipenuhi selama periode waktu tertentu sebelum aturan Application Auto Scaling berlaku.

Ketika Anda menetapkan target pemanfaatan yang tinggi (target sekitar 90%) itu berarti lalu lintas Anda harus lebih tinggi dari 90% untuk jangka waktu tertentu sebelum Application Auto Scaling diaktifkan. Jangan menggunakan pemanfaatan target yang tinggi kecuali aplikasi Anda sangat konstan dan tidak menerima lonjakan lalu lintas.

Ketika Anda menetapkan pemanfaatan yang sangat rendah (target kurang dari 50%) itu berarti aplikasi Anda harus mencapai 50% dari kapasitas yang disediakan sebelum memicu kebijakan Application Auto Scaling. Kecuali lalu lintas aplikasi Anda tumbuh pada tingkat yang sangat agresif, ini biasanya diterjemahkan ke dalam kapasitas yang tidak terpakai dan sumber daya yang terbuang.

## Cara mengidentifikasi tabel dengan pemanfaatan target rendah ( $\leq 50\%$ )

Anda dapat menggunakan AWS CLI atau Konsol Manajemen AWS untuk memantau dan mengidentifikasi kebijakan Application Auto Scaling di resource Amazon Keyspaces Anda.

TargetValues

### Note

Saat Anda menggunakan tabel Multi-wilayah dalam mode kapasitas yang disediakan dengan penskalaan otomatis Amazon Keyspaces, pastikan untuk menggunakan operasi Amazon Keyspaces API untuk mengonfigurasi penskalaan otomatis. Operasi Application Auto Scaling API yang mendasari yang dipanggil Amazon Keyspaces atas nama Anda tidak memiliki kemampuan Multi-region. Untuk informasi selengkapnya, lihat [the section called “Melihat pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”](#).

## AWS CLI

1. Kembalikan seluruh daftar sumber daya dengan menjalankan perintah berikut:

```
aws application-autoscaling describe-scaling-policies --service-namespace
cassandra
```

Perintah ini akan mengembalikan seluruh daftar kebijakan Application Auto Scaling yang dikeluarkan untuk sumber daya Amazon Keyspaces apa pun. Jika hanya ingin mengambil sumber daya dari tabel tertentu, Anda dapat menambahkan `--resource-id` parameter. Contoh:

```
aws application-autoscaling describe-scaling-policies --service-namespace
cassandra --resource-id "keyspace/keyspace-name/table/table-name"
```

2. Kembalikan hanya kebijakan penskalaan otomatis untuk tabel tertentu dengan menjalankan perintah berikut

```
aws application-autoscaling describe-scaling-policies --service-namespace
cassandra --resource-id "keyspace/keyspace-name/table/table-name"
```

Nilai untuk kebijakan Application Auto Scaling disorot di bawah ini. Anda perlu memastikan bahwa nilai target lebih besar dari 50% untuk menghindari penyediaan berlebih. Anda akan mendapatkan hasil yang mirip dengan berikut ini:

```
{
 "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"
 }
]
}

```

## Konsol Manajemen AWS

1. Masuk ke Konsol Manajemen AWS dan navigasikan ke halaman CloudWatch layanan di [Memulai dengan Konsol Manajemen AWS](#). Pilih yang sesuai Wilayah AWS jika perlu.
2. Di bilah navigasi kiri, pilih Tabel. Di halaman Tabel, pilih Nama tabel.
3. Pada halaman Detail Tabel pada tab Kapasitas, tinjau pengaturan Application Auto Scaling tabel Anda.

Jika nilai pemanfaatan target Anda kurang dari atau sama dengan 50%, Anda harus mempelajari metrik pemanfaatan tabel Anda untuk mengetahui apakah nilai tersebut [kurang tersedia atau disediakan secara berlebihan](#).

## Cara mengatasi beban kerja dengan varian musiman

Pertimbangkan skenario berikut: aplikasi Anda sering kali beroperasi di bawah nilai rata-rata minimum, tetapi pemanfaatan targetnya rendah sehingga aplikasi Anda dapat bereaksi dengan cepat terhadap peristiwa yang terjadi pada jam-jam tertentu dalam sehari dan Anda memiliki kapasitas yang memadai serta menghindari throttling. Skenario ini umum terjadi ketika Anda memiliki aplikasi yang sangat sibuk selama jam kantor normal (9 pagi hingga 5 sore) tetapi kemudian berfungsi pada tingkat dasar setelah jam kerja. Karena beberapa pengguna mulai terhubung sebelum jam 9 pagi, aplikasi menggunakan ambang batas rendah ini untuk meningkatkan dengan cepat untuk mencapai kapasitas yang diperlukan selama jam sibuk.

Skenario ini akan seperti berikut:

- Antara jam 5 sore dan 9 pagi ConsumedWriteCapacityUnits unit tetap berada di antara 90 dan 100
- Pengguna mulai terhubung ke aplikasi sebelum jam 9 pagi dan unit kapasitas meningkat pesat (nilai maksimum yang Anda lihat adalah 1500 WCU)
- Rata-rata, penggunaan aplikasi Anda berkisar antara 800 hingga 1200 selama jam kerja

Jika skenario sebelumnya berlaku untuk aplikasi Anda, pertimbangkan untuk menggunakan [penskalaan otomatis aplikasi terjadwal](#), di mana tabel Anda masih dapat memiliki aturan Application Auto Scaling yang dikonfigurasi, tetapi dengan pemanfaatan target yang kurang agresif yang hanya menyediakan kapasitas ekstra pada interval tertentu yang Anda butuhkan.

Anda dapat menggunakan AWS CLI untuk menjalankan langkah-langkah berikut untuk membuat aturan penskalaan otomatis terjadwal yang dijalankan berdasarkan waktu hari dan hari dalam seminggu.

1. Daftarkan tabel Amazon Keyspaces Anda sebagai target yang dapat diskalakan dengan. Application Auto Scaling Target yang dapat diskalakan adalah sumber daya yang skalanya dapat diperkecil atau diperbesar oleh Application Auto Scaling .

```
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. Siapkan tindakan terjadwal sesuai dengan kebutuhan Anda.

Anda memerlukan dua aturan untuk menutupi skenario: satu untuk meningkatkan dan satu lagi untuk menurunkan skala. Aturan pertama untuk meningkatkan tindakan terjadwal ditampilkan dalam contoh berikut.

```
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"
```

Aturan kedua untuk mengurangi tindakan terjadwal ditunjukkan dalam contoh ini.

```
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
```

```
--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. Jalankan perintah berikut untuk memvalidasi bahwa kedua aturan telah diaktifkan:

```
aws application-autoscaling describe-scheduled-actions --service-namespace
cassandra
```

Anda akan mendapatkan hasil seperti ini:

```
{
 "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
 }
 }
]
}
```

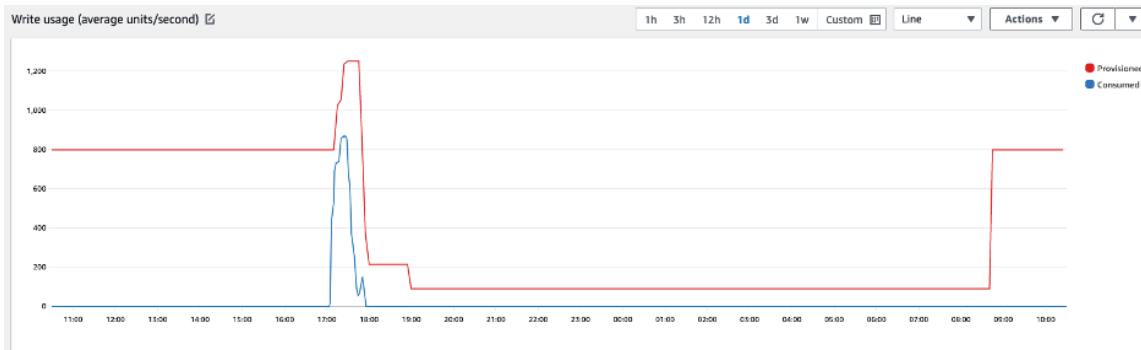


```

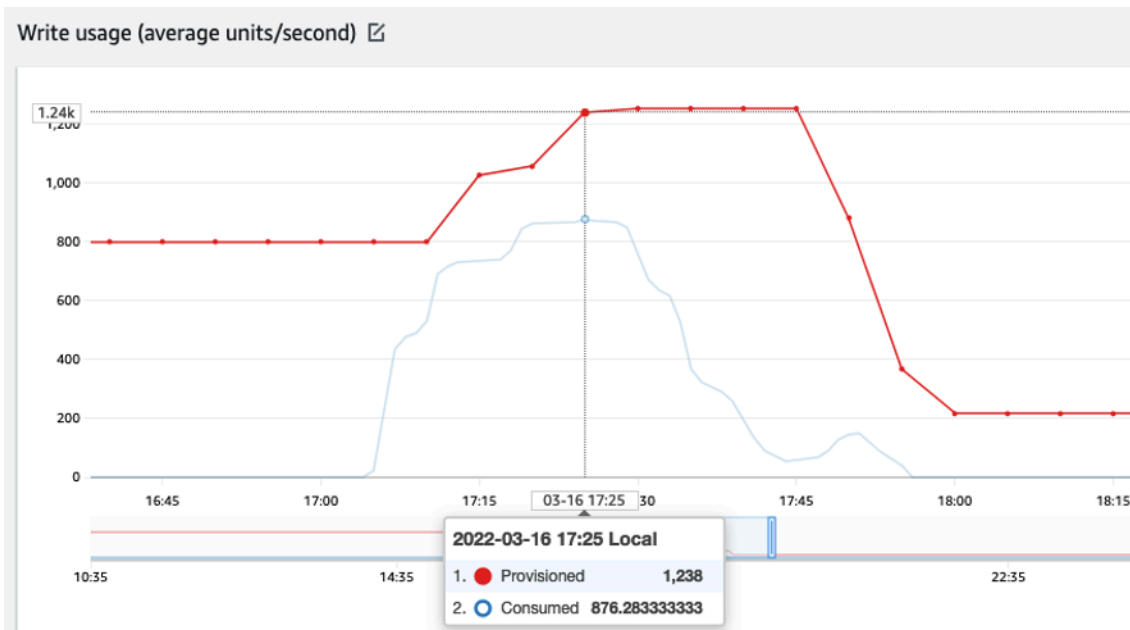
 },
 "CreationTime": "2022-03-15T17:28:57.816000+10:00"
 }
]
}

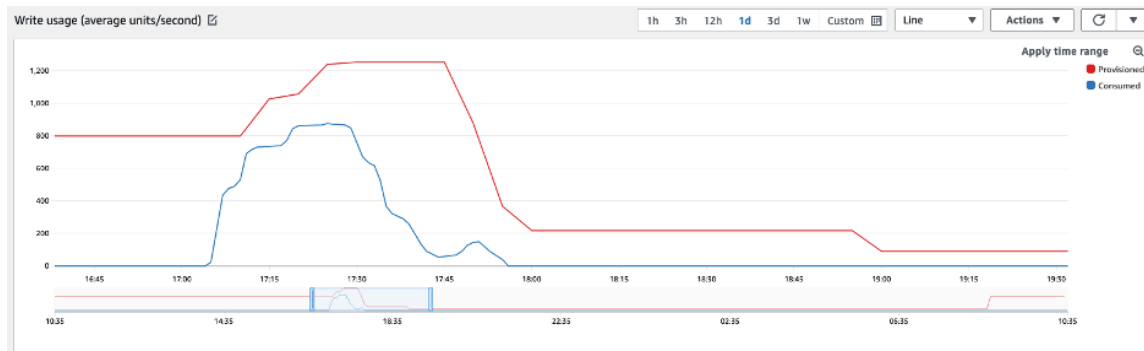
```

Gambar berikut menunjukkan beban kerja sampel yang selalu mempertahankan pemanfaatan target 70%. Perhatikan bagaimana aturan penskalaan otomatis masih berlaku dan throughputnya tidak berkurang.



Jika diperbesar, kita dapat melihat adanya lonjakan dalam aplikasi yang memicu ambang penskalaan otomatis sebesar 70%, sehingga memaksa penskalaan otomatis untuk memulai dan menyediakan kapasitas tambahan yang diperlukan untuk tabel. Tindakan penskalaan otomatis terjadwal akan memengaruhi nilai maksimum dan minimum, dan Anda bertanggung jawab untuk mengaturnya.





## Cara mengatasi lonjakan beban kerja dengan pola yang tidak diketahui

Dalam skenario ini, aplikasi menggunakan target pemanfaatan yang sangat rendah, karena Anda belum mengetahui pola aplikasi, dan Anda ingin memastikan beban kerja Anda tidak mengalami kesalahan throughput kapasitas rendah.

Pertimbangkan untuk menggunakan [mode kapasitas sesuai permintaan](#) sebagai gantinya. On-demand tabel sangat cocok untuk beban kerja runcing di mana Anda tidak tahu pola lalu lintas. Dengan mode kapasitas sesuai permintaan, Anda membayar per permintaan atas pembacaan dan penulisan data yang dilakukan aplikasi Anda pada tabel Anda. Anda tidak perlu menentukan berapa banyak throughput baca dan tulis yang Anda harapkan untuk dilakukan aplikasi Anda, karena Amazon Keyspaces langsung mengakomodasi beban kerja Anda saat naik atau turun.

## Cara mengatasi beban kerja dengan aplikasi tertaut

Dalam skenario ini, aplikasi bergantung pada sistem lain, seperti skenario pemrosesan batch yang dapat menghasilkan lonjakan besar dalam lalu lintas sesuai dengan peristiwa dalam logika aplikasi.

Pertimbangkan untuk mengembangkan logika auto-scaling aplikasi khusus yang bereaksi terhadap peristiwa di mana Anda dapat meningkatkan kapasitas tabel `TargetValues` dan tergantung pada kebutuhan spesifik Anda. Anda bisa mendapatkan keuntungan dari Amazon EventBridge dan menggunakan kombinasi AWS layanan seperti `Lambda` dan Step Functions untuk menanggapi kebutuhan aplikasi spesifik Anda.

## Identifikasi sumber daya yang tidak digunakan untuk mengoptimalkan biaya di Amazon Keyspaces

Bagian ini memberikan gambaran umum tentang cara mengevaluasi sumber daya yang Anda tidak terpakai secara berkala. Saat persyaratan aplikasi Anda berkembang, Anda harus memastikan tidak ada sumber daya yang tidak digunakan dan berkontribusi terhadap biaya Amazon Keyspaces yang

tidak perlu. Prosedur yang dijelaskan di bawah ini menggunakan CloudWatch metrik Amazon untuk mengidentifikasi sumber daya yang tidak digunakan dan mengambil tindakan untuk mengurangi biaya.

Anda dapat memantau Amazon Keyspaces menggunakan CloudWatch, yang mengumpulkan dan memproses data mentah dari Amazon Keyspaces menjadi metrik hampir real-time yang dapat dibaca. Statistik ini disimpan selama jangka waktu tertentu, sehingga Anda dapat mengakses informasi historis untuk lebih memahami pemanfaatan Anda. Secara default, data metrik Amazon Keyspaces dikirim secara otomatis. CloudWatch Untuk informasi selengkapnya, lihat [Apa itu Amazon CloudWatch?](#) dan [Retensi metrik](#) di Panduan CloudWatch Pengguna Amazon.

## Topik

- [Cara mengidentifikasi sumber daya yang tidak terpakai](#)
- [Mengidentifikasi sumber daya tabel yang tidak terpakai](#)
- [Membersihkan sumber daya tabel yang tidak terpakai](#)
- [Membersihkan cadangan point-in-time recovery \(PITR\) yang tidak terpakai](#)

## Cara mengidentifikasi sumber daya yang tidak terpakai

Untuk mengidentifikasi tabel yang tidak digunakan, Anda dapat melihat CloudWatch metrik berikut selama 30 hari untuk memahami apakah ada pembacaan atau penulisan aktif pada tabel tertentu:

### **ConsumedReadCapacityUnits**

Jumlah unit kapasitas baca yang terpakai selama jangka waktu tertentu, sehingga Anda dapat melacak jumlah kapasitas terpakai yang telah Anda gunakan. Anda dapat mengambil total kapasitas baca yang dikonsumsi untuk sebuah tabel.

### **ConsumedWriteCapacityUnits**

Jumlah unit kapasitas tulis yang terpakai selama jangka waktu tertentu, sehingga Anda dapat melacak jumlah kapasitas terpakai yang telah Anda gunakan. Anda dapat mengambil total kapasitas tulis yang dikonsumsi untuk sebuah tabel.

## Mengidentifikasi sumber daya tabel yang tidak terpakai

Amazon CloudWatch adalah layanan pemantauan dan observabilitas yang menyediakan metrik tabel Amazon Keyspaces yang dapat Anda gunakan untuk mengidentifikasi sumber daya yang tidak

digunakan. CloudWatch metrik dapat dilihat melalui Konsol Manajemen AWS maupun melalui AWS Command Line Interface

## AWS Command Line Interface

Untuk melihat metrik tabel Anda melalui AWS Command Line Interface, Anda dapat menggunakan perintah berikut.

1. Pertama, evaluasi pembacaan tabel Anda:



### Note

Jika nama tabel tidak unik dalam akun Anda, Anda juga harus menentukan nama ruang kunci.

```
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>
```

Untuk menghindari kesalahan dalam mengidentifikasi tabel sebagai tidak terpakai, evaluasi metrik dalam jangka waktu yang lebih lama. Pilih rentang waktu mulai dan akhir waktu yang sesuai, seperti 30 hari, dan periode yang sesuai, seperti 86400.

Dalam data yang dikembalikan, setiap Jumlah di atas 0 menunjukkan bahwa tabel yang Anda evaluasi menerima lalu lintas baca selama periode tersebut.

Hasil berikut menunjukkan tabel yang menerima lalu lintas baca pada periode yang dievaluasi:

```
{
 "Timestamp": "2022-08-25T19:40:00Z",
 "Sum": 36023355.0,
 "Unit": "Count"
},
{
 "Timestamp": "2022-08-12T19:40:00Z",
 "Sum": 38025777.5,
 "Unit": "Count"
}
```

```
},
```

Hasil berikut menunjukkan tabel yang tidak menerima lalu lintas baca pada periode yang dievaluasi:

```
{
 "Timestamp": "2022-08-01T19:50:00Z",
 "Sum": 0.0,
 "Unit": "Count"
},
{
 "Timestamp": "2022-08-20T19:50:00Z",
 "Sum": 0.0,
 "Unit": "Count"
},
```

## 2. Selanjutnya, evaluasi penulisan tabel Anda:

```
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>
```

Untuk menghindari kesalahan dalam mengidentifikasi tabel sebagai tidak terpakai, sebaiknya Anda mengevaluasi metrik dalam jangka waktu yang lebih lama. Pilih rentang waktu mulai dan waktu berakhir yang sesuai, seperti 30 hari, dan periode yang sesuai, seperti 86400.

Dalam data yang dikembalikan, setiap Jumlah di atas 0 menunjukkan bahwa tabel yang Anda evaluasi menerima lalu lintas baca selama periode tersebut.

Hasil berikut menunjukkan tabel yang menerima lalu lintas tulis pada periode yang dievaluasi:

```
{
 "Timestamp": "2022-08-19T20:15:00Z",
 "Sum": 41014457.0,
 "Unit": "Count"
},
{
 "Timestamp": "2022-08-18T20:15:00Z",
 "Sum": 40048531.0,
 "Unit": "Count"
}
```

```
},
```

Hasil berikut menunjukkan tabel yang tidak menerima lalu lintas tulis pada periode yang dievaluasi:

```
{
 "Timestamp": "2022-07-31T20:15:00Z",
 "Sum": 0.0,
 "Unit": "Count"
},
{
 "Timestamp": "2022-08-19T20:15:00Z",
 "Sum": 0.0,
 "Unit": "Count"
},
```

## Konsol Manajemen AWS

Langkah-langkah berikut memungkinkan Anda untuk mengevaluasi pemanfaatan sumber daya Anda melalui Konsol Manajemen AWS

1. Masuk ke Konsol Manajemen AWS dan navigasikan ke halaman CloudWatch layanan di <https://console.aws.amazon.com/cloudwatch/>. Pilih yang sesuai Wilayah AWS di kanan atas konsol, jika perlu.
2. Di bilah navigasi kiri, cari bagian Metrik dan pilih Semua metrik.
3. Tindakan di atas membuka dasbor dengan dua panel. Di panel atas, Anda dapat melihat metrik grafik saat ini. Di bagian bawah Anda dapat memilih metrik yang tersedia untuk grafik. Pilih Amazon Keyspaces di panel bawah.
4. Di panel pemilihan metrik Amazon Keyspaces, pilih kategori Metrik Tabel untuk menampilkan metrik tabel di wilayah saat ini.
5. Identifikasi nama tabel Anda dengan menggulir ke bawah menu, lalu pilih metrik ConsumedReadCapacityUnits dan ConsumedWriteCapacityUnits untuk tabel Anda.
6. Pilih Metrik grafik (2) tab dan sesuaikan kolom Statistik ke Jumlah.
7. Untuk menghindari kesalahan mengidentifikasi tabel sebagai tidak terpakai, evaluasi metrik tabel selama periode yang lebih lama. Di bagian atas panel grafik, pilih kerangka waktu yang sesuai, seperti 1 bulan, untuk mengevaluasi tabel Anda. Pilih Kustom, pilih 1 Bulan di menu tarik-turun, dan pilih Terapkan.

8. Evaluasi metrik bergrafik untuk tabel Anda guna menentukan apakah tabel sedang digunakan. Metrik di atas 0 menunjukkan bahwa tabel telah digunakan selama jangka waktu evaluasi. Grafik datar pada 0 untuk membaca dan menulis menunjukkan bahwa tabel tidak digunakan.

## Membersihkan sumber daya tabel yang tidak terpakai

Jika Anda telah mengidentifikasi sumber daya tabel yang tidak terpakai, Anda dapat mengurangi biaya berkelanjutannya dengan cara berikut.

### Note

Jika Anda telah mengidentifikasi tabel yang tidak terpakai tetapi masih ingin tetap tersedia jika tabel tersebut perlu diakses di masa mendatang, pertimbangkan untuk mengalihkannya ke mode sesuai permintaan. Jika tidak, Anda dapat mempertimbangkan untuk menghapus tabel.

## Mode kapasitas

Amazon Keyspaces mengenakan biaya untuk membaca, menulis, dan menyimpan data di tabel Amazon Keyspaces Anda.

Amazon Keyspaces memiliki [dua mode kapasitas](#), yang dilengkapi dengan opsi penagihan khusus untuk memproses pembacaan dan penulisan di tabel Anda: sesuai permintaan dan disediakan. Mode read/write kapasitas mengontrol bagaimana Anda dikenakan biaya untuk throughput baca dan tulis dan bagaimana Anda mengelola kapasitas.

Untuk tabel mode sesuai permintaan, Anda tidak perlu menentukan jumlah throughput baca dan tulis yang Anda harapkan untuk dijalankan oleh aplikasi Anda. Amazon Keyspaces menagih Anda untuk membaca dan menulis bahwa aplikasi Anda bekerja pada tabel Anda dalam hal unit permintaan baca dan unit permintaan tulis. Jika tidak ada aktivitas di meja Anda, Anda tidak membayar untuk throughput tetapi Anda masih dikenakan biaya penyimpanan.

## Menghapus tabel

Jika Anda telah menemukan tabel yang tidak digunakan dan ingin menghapusnya, pertimbangkan untuk membuat cadangan atau mengeksport data terlebih dahulu.

Pencadangan yang dilakukan AWS Backup dapat memanfaatkan tiering cold storage, yang selanjutnya mengurangi biaya. Lihat dokumentasi [Mengelola rencana cadangan](#) untuk informasi tentang cara menggunakan siklus hidup untuk memindahkan cadangan Anda ke penyimpanan dingin.

Setelah tabel Anda telah dicadangkan, Anda dapat menghapusnya melalui Konsol Manajemen AWS atau AWS Command Line Interface.

## Membersihkan cadangan point-in-time recovery (PITR) yang tidak terpakai

Amazon Keyspaces menawarkan Point-in-time pemulihan, yang menyediakan pencadangan berkelanjutan selama 35 hari untuk membantu Anda melindungi dari penulisan atau penghapusan yang tidak disengaja. Pencadangan PITR memiliki biaya yang terkait dengannya.

Lihat dokumentasi di [the section called “Backup dan restore dengan pemulihan point-in-time”](#) untuk menentukan apakah tabel Anda memiliki cadangan yang diaktifkan yang mungkin tidak lagi diperlukan.

## Evaluasi pola penggunaan tabel Anda untuk mengoptimalkan kinerja dan biaya

Bagian ini memberikan ikhtisar tentang cara mengevaluasi jika Anda menggunakan tabel Amazon Keyspaces secara efisien. Ada pola penggunaan tertentu yang tidak optimal untuk Amazon Keyspaces, dan mereka memungkinkan ruang untuk pengoptimalan baik dari perspektif kinerja maupun biaya.

### Topik

- [Lakukan lebih sedikit operasi bacaan sangat konsisten](#)
- [Aktifkan Waktu untuk Tayang \(TTL\)](#)

## Lakukan lebih sedikit operasi bacaan sangat konsisten

Amazon Keyspaces memungkinkan Anda mengonfigurasi [konsistensi baca](#) berdasarkan permintaan. Permintaan baca pada akhirnya konsisten secara default. Akhirnya pembacaan yang konsisten dibebankan pada 0,5 RCU hingga 4 KB data.

Sebagian besar beban kerja terdistribusi bersifat fleksibel dan dapat menoleransi konsistensi akhir. Namun, mungkin ada pola akses yang membutuhkan bacaan sangat konsisten. Pembacaan yang



sangat konsisten dibebankan pada 1 RCU hingga 4 KB data, yang pada dasarnya menggandakan biaya baca Anda. Amazon Keyspaces memberi Anda fleksibilitas untuk menggunakan kedua model konsistensi pada tabel yang sama.

Anda dapat mengevaluasi beban kerja dan kode aplikasi untuk mengonfirmasi apakah bacaan sangat konsisten hanya digunakan jika diperlukan.

## Aktifkan Waktu untuk Tayang (TTL)

[Time to Live \(TTL\)](#) membantu Anda menyederhanakan logika aplikasi Anda dan mengoptimalkan harga penyimpanan dengan kedaluwarsa data dari tabel secara otomatis. Data yang tidak lagi Anda perlukan akan dihapus secara otomatis dari tabel berdasarkan nilai Time to Live yang Anda tetapkan.

## Evaluasi kapasitas yang disediakan untuk penyediaan ukuran yang tepat

Bagian ini memberikan ikhtisar tentang cara mengevaluasi apakah Anda memiliki penyediaan ukuran yang tepat di tabel Amazon Keyspaces Anda. Seiring berkembangnya beban kerja, Anda harus mengubah prosedur operasional dengan tepat, terutama ketika tabel Amazon Keyspaces Anda dikonfigurasi dalam mode yang disediakan dan Anda berisiko untuk menyediakan tabel Anda secara berlebihan atau kurang menyediakan tabel Anda.

Prosedur yang dijelaskan di bagian ini memerlukan informasi statistik yang harus diambil dari tabel Amazon Keyspaces yang mendukung aplikasi produksi Anda. Untuk memahami perilaku aplikasi Anda, Anda harus menentukan periode waktu yang cukup signifikan untuk menangkap data musiman aplikasi Anda. Misalnya, jika aplikasi Anda menunjukkan pola mingguan, menggunakan periode tiga minggu akan memberi Anda cukup ruang untuk menganalisis kebutuhan throughput aplikasi.

Jika Anda tidak tahu harus mulai dari mana, gunakan penggunaan data setidaknya selama satu bulan untuk penghitungan di bawah ini.

Saat mengevaluasi kapasitas, untuk tabel Amazon Keyspaces Anda dapat mengonfigurasi Unit Kapasitas Baca (RCU) dan Unit Kapasitas Tulis (WCU) secara independen.

### Topik

- [Cara mengambil metrik konsumsi dari tabel Amazon Keyspaces](#)
- [Cara mengidentifikasi tabel Amazon Keyspaces yang kurang disediakan](#)
- [Cara mengidentifikasi tabel Amazon Keyspaces yang disediakan secara berlebihan](#)

## Cara mengambil metrik konsumsi dari tabel Amazon Keyspaces

Untuk mengevaluasi kapasitas tabel, pantau CloudWatch metrik berikut dan pilih dimensi yang sesuai untuk mengambil informasi tabel:

| Unit Kapasitas Baca          | Unit Kapasitas Tulis          |
|------------------------------|-------------------------------|
| ConsumedReadCapacityUnits    | ConsumedWriteCapacityUnits    |
| ProvisionedReadCapacityUnits | ProvisionedWriteCapacityUnits |
| ReadThrottleEvents           | WriteThrottleEvents           |

Anda dapat melakukan ini baik melalui AWS CLI atau Konsol Manajemen AWS.

### AWS CLI

Sebelum mengambil metrik konsumsi tabel, Anda harus memulai dengan menangkap beberapa titik data historis menggunakan API. CloudWatch

Mulailah dengan membuat dua file: `write-calc.json` dan `read-calc.json`. File-file ini mewakili perhitungan untuk tabel. Anda perlu memperbarui beberapa bidang, seperti yang ditunjukkan pada tabel di bawah ini, agar sesuai dengan lingkungan Anda.

#### Note

Jika nama tabel tidak unik dalam akun Anda, Anda juga harus menentukan nama ruang kunci.

| Nama Bidang                     | Definisi                                       | Contoh                                           |
|---------------------------------|------------------------------------------------|--------------------------------------------------|
| <code>&lt;table-name&gt;</code> | Nama tabel yang Anda analisis                  | SampleTable                                      |
| <code>&lt;period&gt;</code>     | Jangka waktu yang Anda gunakan untuk mengevalu | Untuk periode 1 jam, Anda harus menentukan: 3600 |

| Nama Bidang  | Definisi                                                      | Contoh               |
|--------------|---------------------------------------------------------------|----------------------|
|              | asi target pemanfaatan, berdasarkan detik                     |                      |
| <start-time> | Awal interval evaluasi Anda, ditentukan dalam format ISO8601  | 2022-02-21 T23:00:00 |
| <end-time>   | Akhir interval evaluasi Anda, ditentukan dalam format ISO8601 | 2022-02-22 T06:00:00 |

File perhitungan tulis mengambil jumlah WCU yang disediakan dan dikonsumsi dalam periode waktu untuk rentang tanggal yang ditentukan. Ini juga menghasilkan persentase pemanfaatan yang dapat digunakan untuk analisis. Isi lengkap `write-calc.json` file akan terlihat seperti pada contoh berikut.

```
{
 "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
}

```

File perhitungan baca menggunakan metrik serupa. File ini mengambil berapa banyak RCU yang disediakan dan dikonsumsi selama periode waktu untuk rentang tanggal yang ditentukan. Isi `read-calc.json` file akan terlihat seperti dalam contoh ini.

```

{
 "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
}

```

Setelah Anda membuat file, Anda dapat mulai mengambil data pemanfaatan.

1. Untuk mengambil data pemanfaatan tulis, keluarkan perintah berikut:

```
aws cloudwatch get-metric-data --cli-input-json file://write-calc.json
```

2. Untuk mengambil data pemanfaatan baca, keluarkan perintah berikut:

```
aws cloudwatch get-metric-data --cli-input-json file://read-calc.json
```

Hasil untuk kedua kueri adalah serangkaian titik data dalam format JSON yang dapat digunakan untuk analisis. Hasil Anda bergantung pada jumlah titik data yang Anda tentukan, periode, dan data beban kerja spesifik Anda sendiri. Itu bisa terlihat seperti pada contoh berikut.

```

{
 "MetricDataResults": [
 {
 "Id": "utilizationPercentage",
 "Label": "Utilization Percentage",
 "Timestamps": [
 "2022-02-22T05:00:00+00:00",
 "2022-02-22T04:00:00+00:00",
 "2022-02-22T03:00:00+00:00",
 "2022-02-22T02:00:00+00:00",
 "2022-02-22T01:00:00+00:00",
 "2022-02-22T00:00:00+00:00",
 "2022-02-21T23:00:00+00:00"
],
 "Values": [

```

```
 91.55364583333333,
 55.066631944444445,
 2.6114930555555556,
 24.9496875,
 40.94725694444445,
 25.61819444444444,
 0.0
],
 "StatusCode": "Complete"
 }
],
"Messages": []
}
```

#### Note

Jika Anda menentukan periode pendek dan rentang waktu yang lama, Anda mungkin perlu memodifikasi `MaxDatapoints` nilainya, yang secara default disetel ke 24 dalam skrip. Ini mewakili satu titik data per jam dan 24 per hari.

## Konsol Manajemen AWS

1. Masuk ke Konsol Manajemen AWS dan navigasikan ke halaman CloudWatch layanan di [Memulai dengan Konsol Manajemen AWS](#). Pilih yang sesuai Wilayah AWS jika perlu.
2. Temukan bagian Metrik di bilah navigasi kiri dan pilih Semua metrik.
3. Ini membuka dasbor dengan dua panel. Panel atas menunjukkan grafik, dan panel bawah memiliki metrik yang ingin Anda grafik. Pilih panel Amazon Keyspaces.
4. Pilih kategori Metrik Tabel dari sub panel. Ini menunjukkan kepada Anda tabel di saat ini Wilayah AWS.
5. Identifikasi nama tabel Anda dengan menggulir menu ke bawah, lalu memilih metrik operasi tulis: `ConsumedWriteCapacityUnits` dan `ProvisionedWriteCapacityUnits`

#### Note

Contoh ini membahas metrik operasi tulis, tetapi Anda juga dapat menggunakan langkah-langkah ini untuk membuat grafik metrik operasi baca.

6. Pilih tab Metrik bergrafik (2) untuk memodifikasi rumus. Secara default CloudWatch memilih fungsi statistik Rata-rata untuk grafik.
7. Saat memilih kedua metrik bergrafik (kotak centang di sebelah kiri) pilih menu Tambahkan perhitungan, diikuti oleh Umum, lalu pilih fungsi Persentase. Ulangi prosedur ini dua kali.

Pertama kali memilih fungsi Persentase.

Kedua kalinya memilih fungsi Persentase.

8. Pada titik ini, Anda memiliki empat metrik di menu bawah. Mari kita kerjakan penghitungan ConsumedWriteCapacityUnits. Agar konsisten, Anda harus mencocokkan nama dengan yang Anda gunakan di AWS CLI bagian ini. Klik m1 ID dan ubah nilai ini menjadi consumedWCU.
9. Ubah statistik dari Average menjadi Sum. Tindakan ini secara otomatis membuat metrik lain yang disebut ANOMALY\_DETECTION\_BAND. Untuk cakupan prosedur ini, Anda dapat mengabaikannya dengan menghapus kotak centang pada metrik ad1 yang baru dibuat.
10. Ulangi langkah 8 untuk mengganti nama m2 ID menjadi ProvisionedWCU. Biarkan statistik diatur ke Average.
11. Pilih label Expression1 dan perbarui nilainya ke m1 dan label ke Consumed WCU.

 Note

Pastikan Anda hanya memilih m1 (kotak centang di sebelah kiri) dan ProvisionedWCU untuk memvisualisasikan data dengan benar. Perbarui rumus dengan mengklik Detail dan mengubah rumus menjadi consumedWCU/PERIOD(ConsumedWCU). Langkah ini mungkin juga menghasilkan metrik ANOMALY\_DETECTION\_BAND lain, tetapi untuk cakupan prosedur ini Anda dapat mengabaikannya.

12. Anda sekarang harus memiliki dua grafik: satu yang menunjukkan WCU yang Anda sediakan di atas meja dan yang lain yang menunjukkan WCU yang dikonsumsi.
13. Perbarui rumus persentase dengan memilih grafik Expression2 (e2). Ganti nama label dan ID menjadi utilizationPercentage. Ganti nama rumus agar cocok dengan  $100 * () m1 / provisionedWCU$ .
14. Hapus kotak centang dari semua metrik kecuali UtilizationPercentage untuk memvisualisasikan pola pemanfaatan Anda. Interval default diatur ke 1 menit, tetapi jangan ragu untuk memodifikasinya sesuai kebutuhan.



Hasil yang Anda dapatkan tergantung pada data aktual dari beban kerja Anda. Interval dengan pemanfaatan lebih dari 100% rentan terhadap peristiwa kesalahan kapasitas throughput yang rendah. Amazon Keyspaces menawarkan [kapasitas burst](#), tetapi segera setelah kapasitas burst habis, apa pun di atas 100% mengalami peristiwa kesalahan kapasitas throughput yang rendah.

## Cara mengidentifikasi tabel Amazon Keyspaces yang kurang disediakan

Untuk sebagian besar beban kerja, tabel dianggap kurang disediakan ketika terus-menerus mengkonsumsi lebih dari 80% dari kapasitas yang disediakan.

[Kapasitas burst](#) adalah fitur Amazon Keyspaces yang memungkinkan pelanggan untuk sementara mengkonsumsi lebih RCUs/WCUs dari yang disediakan semula (lebih dari throughput yang disediakan per detik yang ditentukan untuk tabel). Kapasitas lonjakan diciptakan untuk menyerap peningkatan lalu lintas tiba-tiba karena peristiwa khusus atau lonjakan penggunaan. Kapasitas ledakan ini terbatas, untuk informasi lebih lanjut, lihat [the section called “Gunakan kapasitas burst”](#). Segera setelah RCU dan WCU yang tidak digunakan habis, Anda dapat mengalami peristiwa kesalahan throughput berkapasitas rendah jika Anda mencoba mengkonsumsi lebih banyak kapasitas daripada yang disediakan. Ketika lalu lintas aplikasi Anda mendekati tingkat pemanfaatan 80%, risiko Anda mengalami peristiwa kesalahan throughput kapasitas rendah secara signifikan lebih tinggi.

Aturan tingkat penggunaan 80% bervariasi berdasarkan musim data dan pertumbuhan lalu lintas Anda. Pertimbangkan skenario berikut:

- Jika lalu lintas Anda stabil pada tingkat penggunaan ~90% selama 12 bulan terakhir, tabel Anda memiliki kapasitas yang tepat
- Jika lalu lintas aplikasi Anda tumbuh sebesar 8% setiap bulan dalam waktu kurang dari 3 bulan, Anda akan mencapai 100%
- Jika lalu lintas aplikasi Anda tumbuh sebesar 5% dalam waktu lebih dari 4 bulan, Anda masih akan mencapai 100%

Hasil dari kueri di atas memberikan gambaran tingkat penggunaan Anda. Gunakan hasil tersebut sebagai panduan untuk mengevaluasi lebih lanjut metrik lain yang dapat membantu Anda meningkatkan kapasitas tabel sesuai kebutuhan (misalnya: tingkat pertumbuhan bulanan atau mingguan). Bekerjalah dengan tim operasi Anda untuk menentukan persentase yang baik untuk beban kerja dan tabel Anda.

Ada skenario khusus di mana data miring ketika Anda menganalisisnya setiap hari atau mingguan. Misalnya, dengan aplikasi musiman yang memiliki lonjakan penggunaan selama jam kerja (tetapi kemudian turun menjadi hampir nol di luar jam kerja), Anda bisa mendapatkan keuntungan dari [penjadwalan auto-scaling aplikasi](#), di mana Anda menentukan jam dalam sehari (dan hari dalam seminggu) untuk meningkatkan kapasitas yang disediakan, serta kapan harus mengurangnya. Alih-alih bertujuan untuk kapasitas yang lebih tinggi sehingga Anda dapat menutupi jam sibuk, Anda juga dapat memanfaatkan konfigurasi [auto-scaling tabel Amazon Keyspaces](#) jika musim Anda kurang terasa.

## Cara mengidentifikasi tabel Amazon Keyspaces yang disediakan secara berlebihan

Hasil kueri yang diperoleh dari skrip di atas memberikan titik data yang diperlukan untuk melakukan beberapa analisis awal. Jika set data Anda menyajikan nilai penggunaan yang lebih rendah dari 20% untuk beberapa interval, tabel Anda mungkin disediakan secara berlebihan. Untuk menentukan lebih lanjut apakah Anda perlu mengurangi jumlah WCU dan RCU, Anda harus meninjau kembali pembacaan lain dalam interval tersebut.

Jika tabel berisi beberapa interval penggunaan rendah, Anda bisa mendapatkan keuntungan dari menggunakan kebijakan Application Auto Scaling, baik dengan menjadwalkan Application Auto Scaling atau hanya dengan mengonfigurasi kebijakan Application Auto Scaling default untuk tabel yang didasarkan pada pemanfaatan.

Jika Anda memiliki beban kerja dengan pemanfaatan rendah terhadap rasio throttle tinggi (Max (ThrottleEvents) / Min () dalam intervalThrottleEvents), ini bisa terjadi ketika Anda memiliki beban kerja yang sangat runcing di mana lalu lintas meningkat secara signifikan pada hari-hari tertentu (atau waktu dalam sehari), tetapi sebaliknya secara konsisten rendah. Dalam skenario ini, mungkin bermanfaat untuk menggunakan [Application Auto Scaling terjadwal](#).

# Memecahkan Masalah Amazon Keyspaces (untuk Apache Cassandra)

Panduan ini mencakup langkah-langkah pemecahan masalah untuk berbagai skenario saat bekerja dengan Amazon Keyspaces (untuk Apache Cassandra). Ini mencakup informasi tentang penyelesaian kesalahan umum, masalah koneksi, masalah manajemen kapasitas, dan kesalahan Data Definition Language (DDL).

- Kesalahan umum
  - Memecahkan masalah pengecualian tingkat atas seperti `NoNodeAvailableException`, dan `NoHostAvailableException` `AllNodesFailedException`
  - Mengisolasi kesalahan mendasar dari pengecualian driver Java.
  - Menerapkan kebijakan coba lagi dan mengonfigurasi koneksi dengan benar.
- Masalah koneksi
  - Mengatasi kesalahan saat menghubungkan ke titik akhir Amazon Keyspaces `cqlsh` menggunakan atau driver klien Apache Cassandra.
  - Memecahkan masalah koneksi titik akhir VPC, Cassandra-stress koneksi, dan kesalahan konfigurasi IAM.
  - Menangani kerugian koneksi selama impor data.
- Kesalahan manajemen kapasitas
  - Mengenali dan menyelesaikan kesalahan kapasitas yang tidak memadai terkait dengan tabel, partisi, dan koneksi.
  - Memantau metrik Amazon Keyspaces yang relevan di Log Amazon. CloudWatch
  - Mengoptimalkan koneksi dan throughput untuk meningkatkan kinerja.
- Kesalahan Bahasa Definisi Data (DDL)
  - Memecahkan masalah kesalahan saat membuat, mengakses, atau memulihkan ruang kunci dan tabel.
  - Menangani kegagalan yang terkait dengan pengaturan Time to Live (TTL) kustom, batas kolom, dan penghapusan rentang.
  - Pertimbangan untuk beban kerja penghapusan berat.

Untuk panduan pemecahan masalah khusus untuk akses IAM, lihat [the section called “Pemecahan masalah”](#) Untuk informasi selengkapnya tentang praktik terbaik keamanan, lihat [the section called “Praktik terbaik keamanan”](#).

## Topik

- [Memecahkan masalah kesalahan umum di Amazon Keyspaces](#)
- [Memecahkan masalah kesalahan koneksi di Amazon Keyspaces](#)
- [Memecahkan masalah kesalahan manajemen kapasitas di Amazon Keyspaces](#)
- [Memecahkan masalah kesalahan bahasa definisi data di Amazon Keyspaces](#)

# Memecahkan masalah kesalahan umum di Amazon Keyspaces

Mendapatkan kesalahan umum? Berikut adalah beberapa masalah umum dan cara mengatasinya.

## Kesalahan umum

Anda mendapatkan salah satu pengecualian tingkat atas berikut yang dapat terjadi karena berbagai alasan.

- `NoNodeAvailableException`
- `NoHostAvailableException`
- `AllNodesFailedException`

Pengecualian ini dihasilkan oleh driver klien dan dapat terjadi baik ketika Anda membuat koneksi kontrol atau ketika Anda melakukan permintaan `read/write prepare/execute //batch`.

Ketika kesalahan terjadi saat Anda membuat koneksi kontrol, itu adalah tanda bahwa semua titik kontak yang ditentukan dalam aplikasi Anda tidak dapat dijangkau. Ketika kesalahan terjadi saat melakukan `prepare/execute` kueri `read/write/`, ini menunjukkan bahwa semua percobaan ulang untuk permintaan itu telah habis. Setiap percobaan ulang dicoba pada node yang berbeda saat Anda menggunakan kebijakan coba ulang default.

## Cara mengisolasi kesalahan mendasar dari pengecualian driver Java tingkat atas

Kesalahan umum ini dapat disebabkan oleh masalah koneksi atau saat melakukan `read/write/prepare/execute` operasi. Kegagalan sementara harus diharapkan dalam sistem terdistribusi, dan harus ditangani dengan mencoba kembali permintaan. Driver Java tidak secara otomatis mencoba

lagi ketika terjadi kesalahan koneksi, jadi disarankan untuk menerapkan kebijakan coba lagi saat membuat koneksi driver di aplikasi Anda. Untuk ikhtisar rinci tentang praktik terbaik koneksi, lihat [the section called “Koneksi”](#).

Secara default, driver Java disetel idempotence ke false untuk semua permintaan, yang berarti driver Java tidak secara otomatis mencoba kembali permintaan read/write /prepare yang gagal. Untuk mengatur true dan memberi tahu pengemudi untuk mencoba kembali permintaan yang gagal, Anda dapat melakukannya dengan beberapa cara berbeda. Berikut adalah salah satu contoh bagaimana Anda dapat mengatur idempotence secara terprogram untuk satu permintaan dalam aplikasi Java Anda.

```
Statement s = new SimpleStatement("SELECT * FROM my_table WHERE id = 1");
s.setIdempotent(true);
```

Atau Anda dapat mengatur idempotence default untuk seluruh aplikasi Java Anda secara terprogram seperti yang ditunjukkan pada contoh berikut.

```
// 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
```

Rekomendasi lain adalah membuat kebijakan coba lagi di tingkat aplikasi. Dalam hal ini, aplikasi perlu menangkap `NoNodeAvailableException` dan mencoba kembali permintaan. Kami merekomendasikan 10 percobaan ulang dengan backoff eksponensial mulai dari 10 ms dan bekerja hingga 100 ms dengan total waktu 1 detik untuk semua percobaan ulang.

[Pilihan lainnya adalah menerapkan kebijakan coba lagi eksponensial Amazon Keyspaces saat membuat koneksi driver Java yang tersedia di Github.](#)

Konfirmasikan bahwa Anda telah membuat koneksi ke lebih dari satu node saat menggunakan kebijakan coba ulang default. Anda dapat melakukannya menggunakan kueri berikut di Amazon Keyspaces.

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

Jika respons untuk kueri ini kosong, ini menunjukkan bahwa Anda bekerja dengan satu node untuk Amazon Keyspaces. Jika Anda menggunakan kebijakan coba ulang default, tidak akan ada percobaan ulang karena percobaan ulang default selalu terjadi pada node yang berbeda. Untuk

mempelajari lebih lanjut tentang membuat koneksi melalui titik akhir VPC, lihat. [the section called “Koneksi titik akhir VPC”](#)

Untuk tutorial langkah demi langkah yang menunjukkan cara membuat koneksi ke Amazon Keyspaces menggunakan driver Datastax 4.x Cassandra, lihat. [the section called “Plugin otentikasi untuk Java 4.x”](#)

## Memecahkan masalah kesalahan koneksi di Amazon Keyspaces

Mengalami masalah saat menghubungkan? Berikut adalah beberapa masalah umum dan cara mengatasinya.

### Kesalahan saat menghubungkan ke titik akhir Amazon Keyspaces

Kesalahan koneksi dan koneksi yang gagal dapat mengakibatkan pesan kesalahan yang berbeda. Bagian berikut mencakup skenario yang paling umum.

Topik

- [Saya tidak dapat terhubung ke Amazon Keyspaces dengan cqlsh](#)
- [Saya tidak dapat terhubung ke Amazon Keyspaces menggunakan driver klien Cassandra](#)

### Saya tidak dapat terhubung ke Amazon Keyspaces dengan cqlsh

Anda mencoba terhubung ke titik akhir Amazon Keyspaces menggunakan cqlsh dan koneksi gagal dengan file. **Connection error**

Jika Anda mencoba menyambung ke tabel Amazon Keyspaces dan cqlsh belum dikonfigurasi dengan benar, koneksi gagal. Bagian berikut memberikan contoh masalah konfigurasi paling umum yang mengakibatkan kesalahan koneksi saat Anda mencoba membuat koneksi menggunakan cqlsh.

#### Note

Jika Anda mencoba terhubung ke Amazon Keyspaces dari VPC, izin tambahan diperlukan. Agar berhasil mengonfigurasi koneksi menggunakan titik akhir VPC, ikuti langkah-langkah di file. [the section called “Menghubungkan dengan titik akhir VPC”](#)

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda mendapatkan kesalahan koneksi. **timed out**

Ini mungkin terjadi jika Anda tidak menyediakan port yang benar, yang menghasilkan kesalahan berikut.

```
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')})
```

Untuk mengatasi masalah ini, verifikasi bahwa Anda menggunakan port 9142 untuk koneksi.

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda mendapatkan kesalahan. **Name or service not known**

Ini mungkin terjadi jika Anda menggunakan titik akhir yang salah saja atau tidak ada. Dalam contoh berikut, nama titik akhir salah saja.

```
cqlsh cassandra.us-east-1.amazon.com 9142 -u "USERNAME" -p "PASSWORD" --ssl
Traceback (most recent call last):
 File "/usr/bin/cqlsh.py", line 2458, in <module>
 main(*read_options(sys.argv[1:], os.environ))
 File "/usr/bin/cqlsh.py", line 2436, in main
 encoding=options.encoding)
 File "/usr/bin/cqlsh.py", line 484, in __init__
 load_balancing_policy=WhiteListRoundRobinPolicy([self.hostname]),
 File "/usr/share/cassandra/lib/cassandra-driver-internal-only-3.11.0-bb96859b.zip/
cassandra-driver-3.11.0-bb96859b/cassandra/policies.py", line 417, in __init__
socket.gaierror: [Errno -2] Name or service not known
```

Untuk mengatasi masalah ini saat Anda menggunakan titik akhir publik untuk terhubung, pilih titik akhir yang tersedia [the section called “Titik akhir layanan”](#), dan verifikasi bahwa nama titik akhir tidak memiliki kesalahan. Jika Anda menggunakan titik akhir VPC untuk terhubung, verifikasi bahwa informasi titik akhir VPC sudah benar dalam konfigurasi cqlsh Anda.

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **OperationTimedOut**

Amazon Keyspaces mengharuskan SSL diaktifkan untuk koneksi guna memastikan keamanan yang kuat. Parameter SSL mungkin hilang jika Anda menerima kesalahan berikut.

```
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',)})
```

#

Untuk mengatasi masalah ini, tambahkan tanda berikut ke perintah koneksi cqlsh.

`--ssl`

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, dan Anda menerima kesalahan. **SSL transport factory requires a valid certfile to be specified**

Dalam hal ini, jalur ke SSL/TLS sertifikat tidak ada, yang menghasilkan kesalahan berikut.

```
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.
#
```

Untuk mengatasi masalah ini, tambahkan path ke certfile di komputer Anda. Untuk informasi selengkapnya, lihat [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#).

```
certfile = path_to_file/keyspaces-bundle.pem
```

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **No such file or directory**

Ini mungkin terjadi jika jalur ke file sertifikat di komputer Anda salah, yang menghasilkan kesalahan berikut.

```
cat .cassandra/cqlshrc
[connection]
port = 9142
factory = cqlshlib.ssl.ssl_transport_factory
```



```
[ssl]
validate = true
certfile = /root/wrong_path/keysaces-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')})
#
```

Untuk mengatasi masalah ini, verifikasi bahwa jalur ke certfile di komputer Anda sudah benar. Untuk informasi selengkapnya, lihat [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#).

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **[X509] PEM lib**

Ini mungkin terjadi jika pem file SSL/TLS sertifikat tidak valid, yang menghasilkan kesalahan berikut.

```
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)"))
#
```

Untuk mengatasi masalah ini, pastikan Anda mengunduh sertifikat digital yang diperlukan. Untuk informasi selengkapnya, lihat [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#).

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan SSL. **unknown**

Ini mungkin terjadi jika pem file SSL/TLS sertifikat kosong, yang menghasilkan kesalahan berikut.

```
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)"))
#
```

Untuk mengatasi masalah ini, pastikan Anda mengunduh sertifikat digital yang diperlukan. Anda dapat mengonfirmasi ini menggunakan langkah-langkah dalam topik berikut [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#).

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **SSL: CERTIFICATE\_VERIFY\_FAILED**

Ini mungkin terjadi jika file SSL/TLS sertifikat tidak dapat diverifikasi, yang menghasilkan kesalahan berikut.

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

Untuk mengatasi masalah ini, pastikan Anda mengunduh sertifikat digital yang diperlukan. Anda dapat mengonfirmasi ini menggunakan langkah-langkah dalam topik berikut [the section called “Cara mengkonfigurasi cqlsh koneksi secara manual untuk TLS”](#).

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **Last error: timed out**

Ini mungkin terjadi jika Anda tidak mengonfigurasi aturan keluar untuk Amazon Keyspaces di grup keamanan Amazon EC2 Anda, yang menghasilkan kesalahan berikut.

```
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")})
#
```

Untuk mengonfirmasi bahwa masalah ini disebabkan oleh konfigurasi instans Amazon EC2 dan tidak cqlsh, Anda dapat mencoba menghubungkan ke ruang kunci Anda menggunakan AWS CLI, misalnya dengan perintah berikut.

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

Jika perintah ini juga habis waktu, instans Amazon EC2 tidak dikonfigurasi dengan benar.

Untuk mengonfirmasi bahwa Anda memiliki izin yang cukup untuk mengakses Amazon Keyspaces, Anda dapat menggunakan file untuk AWS CloudShell terhubung. cqlsh Jika koneksi tersebut dibuat, Anda perlu mengonfigurasi instans Amazon EC2.

Untuk mengatasi masalah ini, konfirmasikan bahwa instans Amazon EC2 Anda memiliki aturan keluar yang memungkinkan lalu lintas ke Amazon Keyspaces. Jika bukan itu masalahnya, Anda perlu membuat grup keamanan baru untuk instans EC2, dan menambahkan aturan yang memungkinkan lalu lintas keluar ke sumber daya Amazon Keyspaces. Untuk memperbarui aturan keluar untuk mengizinkan lalu lintas ke Amazon Keyspaces, CQLSH/CASSANDRA pilih dari menu tarik-turun Ketik.

Setelah membuat grup keamanan baru dengan aturan lalu lintas keluar, Anda perlu menambahkannya ke instance. Pilih instance dan kemudian pilih Actions, lalu Security, dan kemudian Change security groups. Tambahkan grup keamanan baru dengan aturan keluar, tetapi pastikan bahwa grup default juga tetap tersedia.

Untuk informasi selengkapnya tentang cara melihat dan mengedit aturan keluar EC2, lihat [Menambahkan aturan ke grup keamanan di Panduan Pengguna Amazon EC2](#).

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **Unauthorized**

Ini mungkin terjadi jika Anda kehilangan izin Amazon Keyspaces dalam kebijakan pengguna IAM, yang mengakibatkan kesalahan berikut.

```
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."',))
#
```

Untuk mengatasi masalah ini, pastikan pengguna IAM testuser-at-12345678910 memiliki izin untuk mengakses Amazon Keyspaces. Untuk contoh kebijakan IAM yang memberikan akses ke Amazon Keyspaces, lihat [the section called “Identity-based contoh kebijakan”](#)

Untuk panduan pemecahan masalah yang khusus untuk akses IAM, lihat [the section called “Pemecahan masalah”](#)

Anda mencoba terhubung ke Amazon Keyspaces menggunakan cqlsh, tetapi Anda menerima kesalahan. **Bad credentials**

Ini mungkin terjadi jika nama pengguna atau kata sandi salah, yang mengakibatkan kesalahan berikut.

```
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"',))
#
```

Untuk mengatasi masalah ini, verifikasi bahwa **USERNAME** dan **PASSWORD** dalam kode Anda cocok dengan nama pengguna dan kata sandi yang Anda peroleh saat Anda membuat kredensial [khusus layanan](#).

#### Important

Jika Anda terus melihat kesalahan saat mencoba terhubung dengan cqlsh, jalankan kembali perintah dengan `--debug` opsi dan sertakan output terperinci saat menghubungi. Dukungan

## Saya tidak dapat terhubung ke Amazon Keyspaces menggunakan driver klien Cassandra

Bagian berikut menunjukkan kesalahan paling umum saat menghubungkan dengan driver klien Cassandra.

Anda mencoba untuk terhubung ke tabel Amazon Keyspaces menggunakan driver DataStax Java, tetapi Anda menerima kesalahan **NodeUnavailableException**.

Jika koneksi di mana permintaan dicoba rusak, itu menghasilkan kesalahan berikut.

```
[com.datastax.oss.driver.api.core.NodeUnavailableException: No connection
was available to Node(endPoint=vpce-22ff22f2f2222fff-aa1bb234.cassandra.us-
west-2.vpce.amazonaws.com/11.1.1111.222:9142, hostId=1a23456b-
c77d-8888-9d99-146cb22d6ef6, hashCode=123ca4567)]
```

Untuk mengatasi masalah ini, temukan nilai detak jantung dan turunkan menjadi 30 detik jika lebih tinggi.

```
advanced.heartbeat.interval = 30 seconds
```

Kemudian cari waktu habis yang terkait dan pastikan nilainya disetel ke setidaknya 5 detik.

```
advanced.connection.init-query-timeout = 5 seconds
```

Anda mencoba terhubung ke tabel Amazon Keyspaces menggunakan driver dan plugin SigV4, tetapi Anda menerima kesalahan. **AttributeError**

Jika kredensial tidak dikonfigurasi dengan benar, itu menghasilkan kesalahan berikut.

```
cassandra.cluster.NoHostAvailable: ('Unable to connect to any servers',
{ '44.234.22.154:9142': AttributeError("'NoneType' object has no attribute
'access_key'") })
```

Untuk mengatasi masalah ini, verifikasi bahwa Anda meneruskan kredensial yang terkait dengan pengguna atau peran IAM Anda saat menggunakan plugin SiGv4. Plugin SiGv4 membutuhkan kredensial berikut.

- **AWS\_ACCESS\_KEY\_ID**— Menentukan kunci AWS akses yang terkait dengan pengguna IAM atau peran.
- **AWS\_SECRET\_ACCESS\_KEY**— Menentukan kunci rahasia yang terkait dengan kunci akses. Ini pada dasarnya adalah “kata sandi” untuk kunci akses.

Untuk mempelajari selengkapnya tentang kunci akses dan plugin SiGv4, lihat. [the section called “Buat kredensi IAM untuk otentikasi AWS”](#)

Anda mencoba menyambung ke tabel Amazon Keyspaces menggunakan driver, tetapi Anda menerima kesalahan. **PartialCredentialsError**

Jika **AWS\_SECRET\_ACCESS\_KEY** hilang, itu dapat mengakibatkan kesalahan berikut.

```
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') })
```

Untuk mengatasi masalah ini, verifikasi bahwa Anda meneruskan plugin SiGv4 **AWS\_ACCESS\_KEY\_ID** dan **AWS\_SECRET\_ACCESS\_KEY** saat menggunakan plugin SiGv4. Untuk mempelajari selengkapnya tentang kunci akses dan plugin SiGv4, lihat. [the section called “Buat kredensi IAM untuk otentikasi AWS”](#)

Anda mencoba menyambung ke tabel Amazon Keyspaces menggunakan driver, tetapi Anda menerima kesalahan **Invalid signature**.

Ini mungkin terjadi jika salah satu komponen yang diperlukan untuk tanda tangan salah atau tidak didefinisikan dengan benar untuk sesi tersebut.

- AWS\_ACCESS\_KEY\_ID
- AWS\_SECRET\_ACCESS\_KEY
- AWS\_DEFAULT\_REGION

Kesalahan berikut adalah contoh kunci akses yang tidak valid.

```
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"'}})
```

Untuk mengatasi masalah ini, verifikasi bahwa kunci akses dan Wilayah AWS telah dikonfigurasi dengan benar untuk plugin SiGv4 untuk mengakses Amazon Keyspaces. Untuk mempelajari selengkapnya tentang kunci akses dan plugin SiGv4, lihat. [the section called “Buat kredensi IAM untuk otentikasi AWS”](#)

Koneksi titik akhir VPC saya tidak berfungsi dengan baik

Anda mencoba terhubung ke Amazon Keyspaces menggunakan titik akhir VPC, tetapi Anda menerima kesalahan peta token atau Anda mengalami throughput rendah.

Ini mungkin terjadi jika koneksi titik akhir VPC tidak dikonfigurasi dengan benar.

Untuk mengatasi masalah ini, verifikasi detail konfigurasi berikut. Untuk mengikuti tutorial langkah demi langkah untuk mempelajari cara mengonfigurasi koneksi melalui antarmuka VPC endpoint untuk Amazon Keyspaces lihat. [the section called “Menghubungkan dengan titik akhir VPC”](#)

1. Konfirmasikan bahwa entitas IAM yang digunakan untuk menyambung ke Amazon Keyspaces read/write memiliki akses ke tabel pengguna dan membaca akses ke tabel sistem seperti yang ditunjukkan pada contoh berikut.

```
{
 "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. Konfirmasikan bahwa entitas IAM yang digunakan untuk menyambung ke Amazon Keyspaces memiliki izin baca yang diperlukan untuk mengakses informasi titik akhir VPC di instans Amazon EC2 Anda seperti yang ditunjukkan pada contoh berikut.

```

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

```

#### Note

Kebijakan terkelola `AmazonKeyspacesReadOnlyAccess_v2` dan `AmazonKeyspacesFullAccess` menyertakan izin yang diperlukan agar Amazon Keyspaces mengakses instans Amazon EC2 untuk membaca informasi tentang titik akhir VPC antarmuka yang tersedia.

Untuk informasi selengkapnya tentang titik akhir VPC, lihat [the section called “Menggunakan titik akhir VPC antarmuka untuk Amazon Keyspaces”](#)

3. Konfirmasikan bahwa konfigurasi SSL driver Java menetapkan validasi nama host ke false seperti yang ditunjukkan dalam contoh ini.

```
hostname-validation = false
```

Untuk informasi selengkapnya tentang konfigurasi driver, lihat [the section called “Langkah 2: Konfigurasi driver”](#).

4. Untuk mengonfirmasi bahwa titik akhir VPC telah dikonfigurasi dengan benar, Anda dapat menjalankan pernyataan berikut dari dalam VPC Anda.

 Note

Anda tidak dapat menggunakan lingkungan pengembang lokal atau editor CQL Amazon Keyspaces untuk mengonfirmasi konfigurasi ini, karena mereka menggunakan titik akhir publik.

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

Outputnya akan terlihat mirip dengan contoh ini dan kembali antara 2 hingga 6 node dengan alamat IPv4 pribadi saat menghubungkan dari jaringan IPv4, tergantung pada pengaturan dan Wilayah VPC Anda. 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)
```



## Saya tidak dapat terhubung menggunakan **cassandra-stress**

Anda mencoba terhubung ke Amazon Keyspaces menggunakan **cassandra-stress** perintah, tetapi Anda menerima kesalahan **SSL context**.

Ini terjadi jika Anda mencoba terhubung ke Amazon Keyspaces, tetapi Anda tidak memiliki pengaturan TrustStore dengan benar. Amazon Keyspaces memerlukan penggunaan Transport Layer Security (TLS) untuk membantu mengamankan koneksi dengan klien.

Dalam hal ini, Anda melihat kesalahan berikut.

```
Error creating the initializing the SSL Context
```

Untuk mengatasi masalah ini, ikuti petunjuk untuk menyiapkan TrustStore seperti yang ditunjukkan dalam topik ini. [the section called “Sebelum Anda mulai”](#)

Setelah TrustStore diatur, Anda harus dapat terhubung dengan perintah berikut.

```
./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"
```

## Saya tidak dapat terhubung menggunakan identitas IAM

Anda mencoba menyambung ke tabel Amazon Keyspaces menggunakan identitas IAM, tetapi Anda menerima kesalahan. **Unauthorized**

Ini terjadi jika Anda mencoba menyambung ke tabel Amazon Keyspaces menggunakan identitas IAM (misalnya, pengguna IAM) tanpa menerapkan kebijakan dan memberi pengguna izin yang diperlukan terlebih dahulu.

Dalam hal ini, Anda melihat kesalahan berikut.

```
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."',))
```

Untuk mengatasi masalah ini, verifikasi izin pengguna IAM. Untuk terhubung dengan driver standar, pengguna harus memiliki setidaknya SELECT akses ke tabel sistem, karena sebagian besar driver membaca sistem keyspaces/tables ketika mereka membuat koneksi.

Misalnya kebijakan IAM yang memberikan akses ke sistem Amazon Keyspaces dan tabel pengguna, lihat. [the section called “Mengakses tabel Amazon Keyspaces”](#)

Untuk meninjau bagian pemecahan masalah khusus untuk IAM, lihat. [the section called “Pemecahan masalah”](#)

Saya mencoba mengimpor data dengan cqlsh dan koneksi ke tabel Amazon Keyspaces saya terputus

Anda mencoba mengunggah data ke Amazon Keyspaces dengan cqlsh, tetapi Anda menerima kesalahan koneksi.

Koneksi ke Amazon Keyspaces gagal setelah klien cqlsh menerima tiga kesalahan berturut-turut dari jenis apa pun dari server. Klien cqlsh gagal dengan pesan berikut.

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

Untuk mengatasi kesalahan ini, Anda perlu memastikan bahwa data yang akan diimpor cocok dengan skema tabel di Amazon Keyspaces. Tinjau file impor untuk kesalahan penguraian. Anda dapat mencoba menggunakan satu baris data dengan menggunakan pernyataan INSERT untuk mengisolasi kesalahan.

Klien secara otomatis mencoba membangun kembali koneksi.

## Memecahkan masalah kesalahan manajemen kapasitas di Amazon Keyspaces

Mengalami masalah dengan kapasitas tanpa server? Berikut adalah beberapa masalah umum dan cara mengatasinya.

### Kesalahan kapasitas tanpa server

Bagian ini menguraikan cara mengenali kesalahan yang terkait dengan manajemen kapasitas tanpa server dan cara mengatasinya. Misalnya, Anda mungkin mengamati peristiwa kapasitas yang tidak mencukupi ketika aplikasi Anda melebihi kapasitas throughput yang disediakan.

Karena Apache Cassandra adalah perangkat lunak berbasis cluster yang dirancang untuk berjalan pada armada node, ia tidak memiliki pesan pengecualian yang terkait dengan fitur tanpa server

seperti kapasitas throughput. Sebagian besar driver hanya memahami kode kesalahan yang tersedia di Apache Cassandra, jadi Amazon Keyspaces menggunakan kumpulan kode kesalahan yang sama untuk menjaga kompatibilitas.

Untuk memetakan kesalahan Cassandra ke peristiwa kapasitas yang mendasarinya, Anda dapat menggunakan Amazon CloudWatch untuk memantau metrik Amazon Keyspaces yang relevan. Insufficient-capacity peristiwa yang mengakibatkan kesalahan sisi klien dapat dikategorikan ke dalam tiga kelompok ini berdasarkan sumber daya yang menyebabkan peristiwa:

- **Tabel** — Jika Anda memilih mode kapasitas yang disediakan untuk tabel, dan aplikasi Anda melebihi throughput yang disediakan, Anda mungkin akan melihat kesalahan kapasitas yang tidak memadai. Untuk informasi selengkapnya, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).
- **Partisi** — Anda mungkin mengalami peristiwa kapasitas yang tidak memadai jika lalu lintas terhadap partisi tertentu melebihi 3.000 RCU atau 1.000 WCU. Kami merekomendasikan untuk mendistribusikan lalu lintas secara seragam di seluruh partisi sebagai praktik terbaik. Untuk informasi selengkapnya, lihat [the section called “Pemodelan data”](#).
- **Koneksi** — Anda mungkin mengalami throughput yang tidak mencukupi jika melebihi kuota untuk jumlah maksimum operasi per detik, per koneksi. Untuk meningkatkan throughput, Anda dapat meningkatkan jumlah koneksi default saat mengonfigurasi koneksi dengan driver.

Untuk mempelajari cara mengonfigurasi koneksi untuk Amazon Keyspaces, lihat [the section called “Cara mengonfigurasi koneksi”](#). Untuk informasi selengkapnya tentang mengoptimalkan koneksi melalui titik akhir VPC, lihat [the section called “Koneksi titik akhir VPC”](#).

Untuk menentukan sumber daya mana yang menyebabkan peristiwa kekurangan kapasitas yang mengembalikan kesalahan sisi klien, Anda dapat memeriksa dasbor di konsol Amazon Keyspaces. Secara default, konsol menyediakan tampilan agregat dari CloudWatch metrik kapasitas dan lalu lintas yang paling umum di bagian Kapasitas dan metrik terkait pada tab Kapasitas untuk tabel.

Untuk membuat dasbor Anda sendiri menggunakan Amazon CloudWatch, periksa metrik Amazon Keyspaces berikut.

- **PerConnectionRequestRateExceeded**— Permintaan ke Amazon Keyspaces yang melebihi kuota untuk tingkat permintaan per koneksi. Setiap koneksi klien ke Amazon Keyspaces dapat mendukung hingga 3000 permintaan CQL per detik. Anda dapat melakukan lebih dari 3000 permintaan per detik dengan membuat beberapa koneksi.

- **ReadThrottleEvents**— Permintaan ke Amazon Keyspaces yang melebihi kapasitas baca untuk tabel.
- **StoragePartitionThroughputCapacityExceeded**— Permintaan ke partisi penyimpanan Amazon Keyspaces yang melebihi kapasitas throughput partisi. Partisi penyimpanan Amazon Keyspaces dapat mendukung hingga 1000 WCU/WRU per detik dan 3000 RCU/RRU per detik. Untuk mengurangi pengecualian ini, kami sarankan Anda meninjau model data Anda untuk mendistribusikan read/write lalu lintas di lebih banyak partisi.
- **WriteThrottleEvents**— Permintaan ke Amazon Keyspaces yang melebihi kapasitas tulis untuk tabel.

Untuk mempelajari lebih lanjut tentang CloudWatch, lihat [the section called “Pemantauan CloudWatch dengan”](#). Untuk daftar semua CloudWatch metrik yang tersedia untuk Amazon Keyspaces, lihat [the section called “Metrik dan dimensi”](#)

 Note

[Untuk memulai dasbor khusus yang menampilkan semua metrik yang umum diamati untuk Amazon Keyspaces, Anda dapat menggunakan templat CloudWatch bawaan yang tersedia GitHub di AWS repositori sampel.](#)

## Topik

- [Saya menerima kesalahan kapasitas NoHostAvailable yang tidak mencukupi dari driver klien saya](#)
- [Saya menerima kesalahan batas waktu tulis selama impor data](#)
- [Saya tidak dapat melihat ukuran penyimpanan sebenarnya dari ruang kunci atau tabel](#)

Saya menerima kesalahan kapasitas **NoHostAvailable** yang tidak mencukupi dari driver klien saya

Anda melihat **Read\_Timeout** atau **Write\_Timeout** pengecualian untuk tabel.

Berulang kali mencoba menulis atau membaca dari tabel Amazon Keyspaces dengan kapasitas yang tidak mencukupi dapat mengakibatkan kesalahan sisi klien yang khusus untuk driver.

Gunakan CloudWatch untuk memantau metrik throughput yang disediakan dan aktual, serta peristiwa kapasitas yang tidak mencukupi untuk tabel. Misalnya, permintaan baca yang tidak memiliki

kapasitas throughput yang cukup gagal dengan `Read_Timeout` pengecualian dan diposting ke `ReadThrottleEvents` metrik. Permintaan tulis yang tidak memiliki kapasitas throughput yang cukup gagal dengan `Write_Timeout` pengecualian dan diposting ke `WriteThrottleEvents` metrik. Untuk informasi selengkapnya tentang metrik ini, lihat [the section called “Metrik dan dimensi”](#).

Untuk mengatasi masalah ini, pertimbangkan salah satu opsi berikut.

- Tingkatkan throughput yang disediakan untuk tabel, yang merupakan jumlah maksimum kapasitas throughput yang dapat dikonsumsi aplikasi. Untuk informasi selengkapnya, lihat [the section called “Unit kapasitas baca dan unit kapasitas tulis”](#).
- Biarkan layanan mengelola kapasitas throughput atas nama Anda dengan penskalaan otomatis. Untuk informasi selengkapnya, lihat [the section called “Kelola kapasitas throughput dengan penskalaan otomatis”](#).
- Pilih mode On-demand kapasitas untuk tabel. Untuk informasi selengkapnya, lihat [the section called “Konfigurasi mode kapasitas sesuai permintaan”](#).

Jika Anda perlu meningkatkan kuota kapasitas default untuk akun Anda, lihat [Kuota](#).

Anda melihat kesalahan yang terkait dengan kapasitas partisi yang terlampaui.

Saat Anda melihat kesalahan, kapasitas `StoragePartitionThroughputCapacityExceeded` partisi untuk sementara terlampaui. Ini mungkin secara otomatis ditangani oleh kapasitas adaptif atau kapasitas sesuai permintaan. Sebaiknya tinjau model data Anda untuk mendistribusikan read/write lalu lintas di lebih banyak partisi untuk mengurangi kesalahan ini. Partisi penyimpanan Amazon Keyspaces dapat mendukung hingga 1000 WCU/WRU per detik dan 3000 RCU/RRU per detik. Untuk mempelajari lebih lanjut tentang cara meningkatkan model data Anda untuk mendistribusikan read/write lalu lintas di lebih banyak partisi, lihat [the section called “Pemodelan data”](#).

`Write_Timeout` pengecualian juga dapat disebabkan oleh peningkatan tingkat operasi penulisan bersamaan yang mencakup data statis dan nonstatis dalam partisi logis yang sama. Jika lalu lintas diharapkan menjalankan beberapa operasi penulisan bersamaan yang menyertakan data statis dan nonstatis dalam partisi logis yang sama, sebaiknya tulis data statis dan nonstatis secara terpisah. Menulis data secara terpisah juga membantu mengoptimalkan biaya throughput.

Anda melihat kesalahan terkait dengan tingkat permintaan koneksi yang terlampaui.

Anda melihat `PerConnectionRequestRateExceeded` karena salah satu penyebab berikut.

- Anda mungkin tidak memiliki cukup koneksi yang dikonfigurasi per sesi.

- Anda mungkin mendapatkan lebih sedikit koneksi daripada rekan yang tersedia, karena Anda tidak memiliki izin titik akhir VPC yang dikonfigurasi dengan benar. Untuk informasi selengkapnya tentang kebijakan titik akhir VPC, lihat [the section called “Menggunakan titik akhir VPC antarmuka untuk Amazon Keyspaces”](#)
- Jika Anda menggunakan driver 4.x, periksa untuk melihat apakah Anda mengaktifkan validasi nama host. Driver mengaktifkan verifikasi nama host TLS secara default. Konfigurasi ini mengarah ke Amazon Keyspaces muncul sebagai cluster simpul tunggal ke driver. Kami menyarankan Anda menonaktifkan verifikasi nama host.

Kami menyarankan Anda mengikuti praktik terbaik ini untuk memastikan bahwa koneksi dan throughput Anda dioptimalkan:

- Konfigurasi penyetelan throughput kueri CQL.

Amazon Keyspaces mendukung hingga 3.000 kueri CQL per koneksi TCP per detik, tetapi tidak ada batasan jumlah koneksi yang dapat dibuat oleh driver.

Sebagian besar driver Cassandra open-source membuat kumpulan koneksi ke Cassandra dan memuat kueri keseimbangan melalui kumpulan koneksi tersebut. Amazon Keyspaces mengekspos 9 alamat IP peer ke driver. Perilaku default sebagian besar driver adalah membuat koneksi tunggal ke setiap alamat IP peer. Oleh karena itu, throughput kueri CQL maksimum dari driver yang menggunakan pengaturan default adalah 27.000 kueri CQL per detik.

Untuk meningkatkan jumlah ini, kami sarankan Anda meningkatkan jumlah koneksi per alamat IP yang dipertahankan driver Anda di kumpulan koneksinya. Misalnya, mengatur koneksi maksimum per alamat IP ke 2 akan menggandakan throughput maksimum driver Anda menjadi 54.000 kueri CQL per detik.

- Optimalkan koneksi simpul tunggal Anda.

Secara default, sebagian besar driver Cassandra open-source membuat satu atau lebih koneksi ke setiap alamat IP yang diiklankan dalam `system.peers` tabel saat membuat sesi. Namun, konfigurasi tertentu dapat menyebabkan driver terhubung ke satu alamat IP Amazon Keyspaces. Ini dapat terjadi jika driver mencoba validasi nama host SSL dari node rekan (misalnya, driver DataStax Java), atau saat terhubung melalui titik akhir VPC.

Untuk mendapatkan ketersediaan dan kinerja yang sama dengan driver dengan koneksi ke beberapa alamat IP, kami sarankan Anda melakukan hal berikut:

- Tingkatkan jumlah koneksi per IP menjadi 9 atau lebih tinggi tergantung pada throughput klien yang diinginkan.
- Buat kebijakan coba ulang khusus yang memastikan bahwa percobaan ulang dijalankan terhadap node yang sama. Untuk informasi selengkapnya, silakan lihat

[the section called “Cara mengonfigurasi kebijakan coba lagi”](#).

- Jika Anda menggunakan titik akhir VPC, berikan entitas IAM yang digunakan untuk menyambung ke izin akses Amazon Keyspaces untuk menanyakan VPC Anda untuk informasi titik akhir dan antarmuka jaringan. Hal ini meningkatkan load balancing dan meningkatkan read/write throughput. Untuk informasi selengkapnya, lihat [???](#).

Saya menerima kesalahan batas waktu tulis selama impor data

Anda menerima kesalahan batas waktu saat mengunggah data menggunakan perintah. **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 menggunakan WriteTimeout pengecualian ReadTimeout dan untuk menunjukkan kapan permintaan tulis gagal karena kapasitas throughput yang tidak mencukupi. Untuk membantu mendiagnosis pengecualian kapasitas yang tidak mencukupi, Amazon Keyspaces menerbitkan metrik berikut di Amazon. CloudWatch

- WriteThrottleEvents
- ReadThrottledEvents
- StoragePartitionThroughputCapacityExceeded

Untuk mengatasi kesalahan kapasitas yang tidak memadai selama pemuatan data, turunkan tingkat penulisan per pekerja atau tingkat konsumsi total, lalu coba lagi untuk mengunggah baris. Untuk informasi selengkapnya, lihat [the section called “Langkah 4: Konfigurasi cqlsh COPY FROM pengaturan”](#). Untuk opsi unggah data yang lebih kuat, pertimbangkan untuk menggunakan DSBulk, yang tersedia dari [GitHub repositori](#). Untuk instruksi langkah demi langkah, lihat [the section called “Memuat data menggunakan DSBulk”](#).

Saya tidak dapat melihat ukuran penyimpanan sebenarnya dari ruang kunci atau tabel

Anda tidak dapat melihat ukuran penyimpanan sebenarnya dari keyspace atau tabel.

Untuk mempelajari lebih lanjut tentang ukuran penyimpanan tabel Anda, lihat [the section called “Mengevaluasi biaya di tingkat tabel”](#). Anda juga dapat memperkirakan ukuran penyimpanan dengan mulai menghitung ukuran baris dalam tabel. Instruksi terperinci untuk menghitung ukuran baris tersedia di [the section called “Perkiraan ukuran baris”](#).

## Memecahkan masalah kesalahan bahasa definisi data di Amazon Keyspaces

Kesulitan menciptakan sumber daya? Berikut adalah beberapa masalah umum dan cara mengatasinya.

### Kesalahan bahasa definisi data

Amazon Keyspaces melakukan operasi bahasa definisi data (DDL) secara asinkron—misalnya, membuat dan menghapus ruang kunci dan tabel. Jika aplikasi mencoba menggunakan sumber daya sebelum siap, operasi gagal.

Anda dapat memantau status pembuatan ruang kunci dan tabel baru di Konsol Manajemen AWS, yang menunjukkan kapan ruang kunci atau tabel tertunda atau aktif. Anda juga dapat memantau status pembuatan keyspace atau tabel baru secara terprogram dengan menanyakan tabel skema sistem. Sebuah keyspace atau tabel menjadi terlihat dalam skema sistem ketika siap untuk digunakan.

#### Note

Untuk mengoptimalkan pembuatan ruang kunci menggunakan CloudFormation, Anda dapat menggunakan utilitas ini untuk mengonversi skrip CQL menjadi templat. CloudFormation Alat ini tersedia dari [GitHub repositori](#).

### Topik

- [Saya membuat ruang kunci baru, tetapi saya tidak dapat melihat atau mengaksesnya](#)
- [Saya membuat tabel baru, tetapi saya tidak dapat melihat atau mengaksesnya](#)



- [Saya mencoba memulihkan tabel menggunakan Amazon Keyspaces point-in-time recovery \(PITR\), tetapi pemulihan gagal](#)
- [Saya mencoba menggunakan INSERT/UPDATE untuk mengedit pengaturan Time to Live \(TTL\) kustom, tetapi operasi gagal](#)
- [Saya mencoba mengunggah data ke tabel Amazon Keyspaces saya dan saya mendapatkan kesalahan tentang melebihi jumlah kolom](#)
- [Saya mencoba menghapus data di tabel Amazon Keyspaces saya dan penghapusan gagal untuk rentang tersebut](#)

Saya membuat ruang kunci baru, tetapi saya tidak dapat melihat atau mengaksesnya

Anda menerima kesalahan dari aplikasi Anda yang mencoba mengakses keyspace baru.

Jika Anda mencoba mengakses ruang kunci Amazon Keyspaces yang baru dibuat yang masih dibuat secara asinkron, Anda akan mendapatkan kesalahan. Kesalahan berikut adalah contohnya.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured keyspace mykeyspace"
```

Pola desain yang disarankan untuk memeriksa kapan ruang kunci baru siap digunakan adalah dengan melakukan polling tabel skema sistem Amazon Keyspaces (system\_schema\_mcs.\*).

Untuk informasi selengkapnya, lihat [the section called “Periksa status pembuatan keyspace”](#).

Saya membuat tabel baru, tetapi saya tidak dapat melihat atau mengaksesnya

Anda menerima kesalahan dari aplikasi Anda yang mencoba mengakses tabel baru.

Jika Anda mencoba mengakses tabel Amazon Keyspaces yang baru dibuat yang masih dibuat secara asinkron, Anda akan mendapatkan kesalahan. Misalnya, mencoba menanyakan tabel yang tidak tersedia namun gagal dengan `unconfigured table` kesalahan.

```
InvalidRequest: Error from server: code=2200 [Invalid query] message="unconfigured table mykeyspace.mytable"
```

Mencoba melihat tabel dengan `sync_table()` gagal dengan `aKeyError`.

```
KeyError: 'mytable'
```

Pola desain yang disarankan untuk memeriksa kapan tabel baru siap digunakan adalah dengan melakukan polling tabel skema sistem Amazon Keyspaces (`system_schema_mcs.*`).

Ini adalah contoh output untuk tabel yang sedang dibuat.

```
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)
```

Ini adalah contoh output untuk tabel yang aktif.

```
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)
```

Untuk informasi selengkapnya, lihat [the section called “Periksa status pembuatan tabel”](#).

Saya mencoba memulihkan tabel menggunakan Amazon Keyspaces point-in-time recovery (PITR), tetapi pemulihan gagal

Jika Anda mencoba memulihkan tabel Amazon Keyspaces dengan point-in-time recovery (PITR), dan Anda melihat proses pemulihan dimulai tetapi tidak berhasil diselesaikan, Anda mungkin belum mengonfigurasi semua izin yang diperlukan yang diperlukan oleh proses pemulihan untuk tabel khusus ini.

Selain izin pengguna, Amazon Keyspaces mungkin memerlukan izin untuk melakukan tindakan selama proses pemulihan atas nama kepala sekolah Anda. Ini adalah kasus jika tabel dienkripsi dengan kunci yang dikelola pelanggan, atau jika Anda menggunakan kebijakan IAM yang membatasi lalu lintas masuk.

Misalnya, jika Anda menggunakan kunci kondisi dalam kebijakan IAM untuk membatasi lalu lintas sumber ke titik akhir atau rentang IP tertentu, operasi pemulihan gagal. Untuk mengizinkan Amazon Keyspaces menjalankan operasi pemulihan tabel atas nama kepala sekolah, Anda harus menambahkan kunci kondisi `aws:ViaAWSService` global dalam kebijakan IAM.

Untuk informasi selengkapnya tentang izin untuk memulihkan tabel, lihat [the section called “Konfigurasi izin IAM untuk memulihkan”](#).

Saya mencoba menggunakan INSERT/UPDATE untuk mengedit pengaturan Time to Live (TTL) kustom, tetapi operasi gagal

Jika Anda mencoba menyisipkan atau memperbarui nilai TTL kustom, operasi mungkin gagal dengan kesalahan berikut.

```
TTL is not yet supported.
```

Untuk menentukan nilai TTL kustom untuk baris atau kolom dengan menggunakan INSERT atau UPDATE operasi, Anda harus terlebih dahulu mengaktifkan TTL untuk tabel. Anda dapat mengaktifkan TTL untuk tabel menggunakan properti `tTL` kustom.

Untuk informasi selengkapnya tentang mengaktifkan pengaturan TTL kustom untuk tabel, lihat [the section called “Perbarui tabel kustom TTL”](#)

Saya mencoba mengunggah data ke tabel Amazon Keyspaces saya dan saya mendapatkan kesalahan tentang melebihi jumlah kolom

Anda mengunggah data dan telah melampaui jumlah kolom yang dapat diperbarui secara bersamaan.

Kesalahan ini terjadi ketika skema tabel Anda melebihi ukuran maksimum 350 KB. Untuk informasi selengkapnya, lihat [Kuota](#).

Saya mencoba menghapus data di tabel Amazon Keyspaces saya dan penghapusan gagal untuk rentang tersebut

Anda mencoba menghapus data dengan kunci partisi dan menerima kesalahan penghapusan rentang.

Kesalahan ini terjadi ketika Anda mencoba menghapus lebih dari 1.000 baris dalam satu operasi penghapusan.

```
Range delete requests are limited by the amount of items that can be deleted in a single range.
```

Untuk informasi selengkapnya, lihat [the section called “Hapus rentang”](#).

Untuk menghapus lebih dari 1.000 baris dalam satu partisi, pertimbangkan opsi berikut.

- Hapus dengan partisi - Jika mayoritas partisi berada di bawah 1.000 baris, Anda dapat mencoba menghapus data dengan partisi. Jika partisi berisi lebih dari 1.000 baris, cobalah untuk menghapus dengan kolom pengelompokan sebagai gantinya.
- Hapus dengan pengelompokan kolom - Jika model Anda berisi beberapa kolom pengelompokan, Anda dapat menggunakan hierarki kolom untuk menghapus beberapa baris. Kolom pengelompokan adalah struktur bersarang, dan Anda dapat menghapus banyak baris dengan beroperasi pada kolom tingkat atas.
- Hapus berdasarkan baris individual - Anda dapat mengulangi baris dan menghapus setiap baris dengan kunci utama lengkapnya (kolom partisi dan kolom pengelompokan).
- Sebagai praktik terbaik, pertimbangkan untuk membagi baris Anda di seluruh partisi — Di Amazon Keyspaces, kami sarankan Anda mendistribusikan throughput Anda di seluruh partisi tabel. Ini mendistribusikan data dan akses secara merata di seluruh sumber daya fisik, yang memberikan throughput terbaik. Untuk informasi selengkapnya, lihat [the section called “Pemodelan data”](#).

Pertimbangkan juga rekomendasi berikut saat Anda berencana menghapus operasi untuk beban kerja yang berat.

- Dengan Amazon Keyspaces, partisi dapat berisi jumlah baris yang hampir tidak terbatas. Ini memungkinkan Anda untuk menskalakan partisi “lebih luas” daripada panduan Cassandra tradisional 100 MB. Tidak jarang deret waktu atau buku besar tumbuh lebih dari satu gigabyte data dari waktu ke waktu.

- Dengan Amazon Keyspaces, tidak ada strategi pemadatan atau batu nisan yang perlu dipertimbangkan ketika Anda harus melakukan operasi penghapusan untuk beban kerja yang berat. Anda dapat menghapus data sebanyak yang Anda inginkan tanpa memengaruhi kinerja baca.

# Memantau Amazon Keyspaces (untuk Apache Cassandra)

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon Keyspaces dan solusi Anda yang lain AWS . AWS menyediakan alat pemantauan berikut untuk menonton Amazon Keyspaces, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu:

- Amazon Keyspaces menawarkan dasbor yang telah dikonfigurasi sebelumnya dalam Konsol Manajemen AWS menampilkan latensi dan kesalahan yang dikumpulkan di semua tabel di akun.
- Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat mengumpulkan dan melacak metrik dengan dasbor khusus. Misalnya, Anda dapat membuat garis dasar untuk kinerja Amazon Keyspaces normal di lingkungan Anda dengan mengukur kinerja pada berbagai waktu dan dalam kondisi beban yang berbeda. Saat Anda memantau Amazon Keyspaces, simpan data pemantauan historis sehingga Anda dapat membandingkannya dengan data kinerja saat ini, mengidentifikasi pola kinerja normal dan anomali kinerja, dan merancang metode untuk mengatasi masalah. Untuk menetapkan garis dasar, Anda harus, setidaknya, memantau kesalahan sistem. Untuk informasi selengkapnya, lihat [Panduan CloudWatch Pengguna Amazon](#).
- CloudWatch Alarm Amazon memantau satu metrik selama periode waktu yang Anda tentukan, dan melakukan satu atau beberapa tindakan berdasarkan nilai metrik relatif terhadap ambang batas tertentu selama beberapa periode waktu. Misalnya, jika Anda menggunakan Amazon Keyspaces dalam mode yang disediakan dengan penskalaan otomatis aplikasi, tindakannya adalah notifikasi yang dikirim oleh Amazon Simple Notification Service (Amazon SNS) untuk mengevaluasi kebijakan Application Auto Scaling.

CloudWatch alarm tidak memanggil tindakan hanya karena mereka berada dalam keadaan tertentu. Status harus diubah dan dipelihara selama jangka waktu tertentu. Untuk informasi selengkapnya, lihat [Memantau Amazon Keyspaces dengan Amazon CloudWatch](#).

- Amazon CloudWatch Logs memungkinkan Anda memantau, menyimpan, dan mengakses file log Anda dari tabel Amazon Keyspaces CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log dalam penyimpanan yang sangat durabel. Untuk informasi selengkapnya, lihat [Panduan Pengguna Amazon CloudWatch Logs](#).
- AWS CloudTrail merekam panggilan API dan kejadian terkait yang dilakukan oleh atau atas Akun AWS Anda dan mengirimkan berkas log ke bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun yang memanggil AWS, alamat IP asal panggilan

dilakukan, dan waktu panggilan terjadi. Untuk informasi selengkapnya, lihat [Panduan Pengguna AWS CloudTrail](#).

Amazon EventBridge adalah layanan bus acara tanpa server yang memudahkan untuk menghubungkan aplikasi Anda dengan data dari berbagai sumber. EventBridge mengirimkan aliran data real-time dari aplikasi Anda sendiri, aplikasi Software-as-a-Service (SaaS), AWS dan layanan dan rute data tersebut ke target seperti Lambda. Hal ini memungkinkan Anda memantau kejadian yang terjadi dalam layanan, dan membangun arsitektur yang didorong kejadian. Untuk informasi selengkapnya, lihat [Panduan EventBridge Pengguna Amazon](#).

#### Topik

- [Memantau Amazon Keyspaces dengan Amazon CloudWatch](#)
- [Mencatat panggilan API Amazon Keyspaces dengan AWS CloudTrail](#)

## Memantau Amazon Keyspaces dengan Amazon CloudWatch

Anda dapat memantau Amazon Keyspaces menggunakan Amazon CloudWatch, yang mengumpulkan data mentah dan memprosesnya menjadi metrik yang dapat dibaca, mendekati real-time. Statistik ini disimpan untuk jangka waktu 15 bulan, sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang performa aplikasi atau layanan web Anda.

Anda juga dapat mengatur alarm yang memperhatikan ambang batas tertentu dan mengirim notifikasi atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi selengkapnya, lihat [Panduan CloudWatch Pengguna Amazon](#).

#### Note

Untuk memulai dengan cepat dengan CloudWatch dasbor yang telah dikonfigurasi sebelumnya yang menampilkan metrik umum untuk Amazon Keyspaces, Anda dapat menggunakan templat yang CloudFormation tersedia. <https://github.com/aws-samples/amazon-keyspaces-cloudwatch-cloudformation-templates>

#### Topik

- [Bagaimana cara menggunakan metrik Amazon Keyspaces?](#)

- [Metrik dan dimensi Amazon Keyspaces](#)
- [Melihat metrik Amazon Keyspaces di CloudWatch](#)
- [Membuat CloudWatch alarm untuk memantau Amazon Keyspaces](#)

## Bagaimana cara menggunakan metrik Amazon Keyspaces?

Metrik yang dilaporkan oleh Amazon Keyspaces memberikan informasi yang dapat Anda analisis dengan berbagai cara. Daftar berikut menunjukkan beberapa penggunaan umum untuk metrik. Daftar ini berisi saran agar Anda dapat memulai, bukan daftar komprehensif. Untuk informasi selengkapnya tentang metrik dan retensi, lihat [Metrik](#).

| Bagaimana saya dapat melakukannya?                                                                           | Metrik terkait                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Bagaimana saya bisa menentukan apakah ada kesalahan sistem yang terjadi?                                     | Anda dapat memantau <code>SystemErrors</code> untuk menentukan apakah ada permintaan yang menghasilkan kode kesalahan server. Biasanya, metrik ini sama dengan nol. Jika tidak, Anda mungkin ingin menyelidikinya.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Bagaimana saya bisa membandingkan rata-rata pembacaan yang disediakan dengan kapasitas baca yang dikonsumsi? | <p>Untuk memantau rata-rata kapasitas baca yang disediakan dan kapasitas baca yang dikonsumsi</p> <ol style="list-style-type: none"> <li>1. Atur Periode untuk <code>ConsumedReadCapacityUnits</code> dan <code>ProvisionedReadCapacityUnits</code> ke interval yang ingin Anda pantau.</li> <li>2. Ubah Statistik untuk <code>ConsumedReadCapacityUnits</code> dari <code>Average</code> ke <code>Sum</code>.</li> <li>3. Buat ekspresi Matematika kosong baru.</li> <li>4. Di bagian Detail dari ekspresi matematika baru, masukkan Id dari <code>ConsumedReadCapacityUnits</code> dan bagi metrik dengan fungsi CloudWatch <code>PERIOD</code> dari metrik (<code>metric_id/PERIOD (metric_id)</code>).</li> <li>5. Batalkan pilihan <code>ConsumedReadCapacityUnits</code>.</li> </ol> |



| Bagaimana saya dapat melakukannya?                                                                            | Metrik terkait                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                               | Anda sekarang dapat membandingkan kapasitas baca rata-rata yang dikonsumsi dengan kapasitas yang disediakan. Untuk informasi lebih lanjut tentang fungsi aritmatika dasar dan cara membuat deret waktu lihat <a href="#">Menggunakan</a> matematika metrik.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Bagaimana saya bisa membandingkan rata-rata penulisan yang disediakan dengan kapasitas tulis yang dikonsumsi? | <p>Untuk memantau rata-rata kapasitas tulis yang disediakan dan kapasitas tulis yang dikonsumsi</p> <ol style="list-style-type: none"> <li>1. Atur Periode untuk ConsumedWriteCapacityUnits dan ProvisionedWriteCapacityUnits ke interval yang ingin Anda pantau.</li> <li>2. Ubah Statistik untuk ConsumedWriteCapacityUnits dari Average ke Sum.</li> <li>3. Buat ekspresi Matematika kosong baru.</li> <li>4. Di bagian Detail dari ekspresi matematika baru, masukkan Id dari ConsumedWriteCapacityUnits dan bagi metrik dengan fungsi CloudWatch PERIOD dari metrik (metric_id/ (PERIOD (metric_id))).</li> <li>5. Batalkan pilihan ConsumedWriteCapacityUnits .</li> </ol> <p>Anda sekarang dapat membandingkan kapasitas tulis rata-rata yang Anda konsumsi dengan kapasitas yang disediakan. Untuk informasi lebih lanjut tentang fungsi aritmatika dasar dan cara membuat deret waktu lihat <a href="#">Menggunakan</a> matematika metrik.</p> |

## Metrik dan dimensi Amazon Keyspaces

Saat Anda berinteraksi dengan Amazon Keyspaces, Google akan mengirimkan metrik dan dimensi ke Amazon. CloudWatch Semua metrik dikumpulkan dan dilaporkan setiap menit. Bagian berikut adalah ikhtisar dari semua metrik yang tersedia untuk Amazon Keyspaces.

### Topik

- [Metrik Amazon Keyspaces](#)

- [Dimensi untuk metrik Amazon Keyspaces](#)
- [Metrik untuk Amazon Keyspaces mengubah pengambilan data \(CDC\)](#)
- [Dimensi untuk Amazon Keyspaces mengubah metrik pengambilan data \(CDC\)](#)

## Metrik Amazon Keyspaces

Amazon CloudWatch mengumpulkan metrik Amazon Keyspaces dengan interval satu menit.

Tidak semua statistik, seperti Average atau Sum, berlaku untuk setiap metrik. Namun, semua nilai ini tersedia melalui konsol Amazon Keyspaces, atau dengan menggunakan konsol, AWS CLI, atau AWS SDK untuk semua metrik. CloudWatch Dalam tabel berikut, setiap metrik memiliki daftar statistik valid yang berlaku untuk metrik tersebut.

| Metrik                     | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AccountMaxTableLevelReads  | <p>Jumlah maksimum unit kapasitas baca yang dapat digunakan oleh tabel akun. Untuk tabel sesuai permintaan, batas ini membatasi unit permintaan baca maksimum yang dapat digunakan tabel.</p> <p>Unit: Count</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Maximum</b>— Jumlah maksimum unit kapasitas baca yang dapat digunakan oleh tabel akun.</li> </ul>    |
| AccountMaxTableLevelWrites | <p>Jumlah maksimum unit kapasitas tulis yang dapat digunakan oleh tabel akun. Untuk tabel sesuai permintaan, batas ini membatasi unit permintaan tulis maksimum yang dapat digunakan tabel.</p> <p>Unit: Count</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Maximum</b>— Jumlah maksimum unit kapasitas tulis yang dapat digunakan oleh tabel akun.</li> </ul> |

| Metrik                                         | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AccountProvisioned<br>ReadCapacityUtilization  | <p>Persentase unit kapasitas baca yang disediakan dan digunakan oleh suatu akun.</p> <p>Unit: Percent</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Maximum</b> – Persentase maksimum unit kapasitas baca yang disediakan dan digunakan oleh suatu akun.</li> <li>• <b>Minimum</b> – Persentase minimum unit kapasitas baca yang disediakan dan digunakan oleh suatu akun.</li> <li>• <b>Average</b> – Persentase rata-rata unit kapasitas baca yang disediakan dan digunakan oleh suatu akun.</li> </ul> <p>Metrik diterbitkan selama interval lima menit. Oleh karena itu, jika Anda dengan cepat menyesuaikan unit kapasitas baca yang disediakan, statistik ini mungkin tidak mencerminkan rata-rata sebenarnya.</p>      |
| AccountProvisioned<br>WriteCapacityUtilization | <p>Persentase unit kapasitas tulis yang disediakan dan digunakan oleh suatu akun.</p> <p>Unit: Percent</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Maximum</b> – Persentase maksimum unit kapasitas tulis yang disediakan dan digunakan oleh suatu akun.</li> <li>• <b>Minimum</b> – Persentase minimum unit kapasitas tulis yang disediakan dan digunakan oleh suatu akun.</li> <li>• <b>Average</b> – Persentase rata-rata unit kapasitas tulis yang disediakan dan digunakan oleh suatu akun.</li> </ul> <p>Metrik diterbitkan selama interval lima menit. Oleh karena itu, jika Anda dengan cepat menyesuaikan unit kapasitas tulis yang disediakan, statistik ini mungkin tidak mencerminkan rata-rata sebenarnya.</p> |

| Metrik                                | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>BillableTableSizeInBytes</code> | <p>Ukuran tabel yang dapat ditagih dalam byte. Ini adalah jumlah ukuran yang dikodekan dari semua baris dalam tabel. Metrik ini membantu Anda melacak biaya penyimpanan tabel Anda dari waktu ke waktu.</p> <p>Unit: Bytes</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• <code>Maximum</code>— Ukuran penyimpanan maksimum meja.</li><li>• <code>Minimum</code>— Ukuran penyimpanan minimum meja.</li><li>• <code>Average</code>— Ukuran penyimpanan rata-rata meja.</li></ul> <p>Metrik ini dihitung lebih dari interval 4 - 6 jam.</p> |

| Metrik                         | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ConditionalCheckFailedRequests | <p>Jumlah permintaan tulis transaksi ringan yang gagal (LWT). Operasi INSERT, UPDATE, dan DELETE memungkinkan Anda memberikan syarat logis yang harus bernilai true sebelum operasi dapat dilanjutkan. Jika kondisi ini dievaluasi menjadi false, bertambah ConditionalCheckFailedRequests satu. Pemeriksaan kondisi yang mengevaluasi unit kapasitas penulisan konsumsi palsu berdasarkan ukuran baris. Untuk informasi selengkapnya, lihat <a href="#">the section called “Perkirakan konsumsi kapasitas LWT”</a>.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• Minimum</li><li>• Maximum</li><li>• Average</li><li>• SampleCount</li><li>• Sum</li></ul> |

| Metrik                    | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ConsumedReadCapacityUnits | <p>Jumlah unit kapasitas baca yang dikonsumsi selama periode waktu yang ditentukan. Untuk informasi selengkapnya, lihat <a href="#">mode Read/Write kapasitas</a>.</p> <div data-bbox="688 401 1510 1192"> <p> <b>Note</b></p> <p>Untuk memahami pemanfaatan throughput rata-rata per detik, gunakan Sum statistik untuk menghitung throughput yang dikonsumsi untuk periode satu menit. Kemudian bagi jumlah dengan jumlah detik dalam satu menit (60) untuk menghitung rata-rata ConsumedReadCapacityUnits per detik (mengakui bahwa rata-rata ini tidak menyoroti lonjakan besar tetapi singkat dalam aktivitas membaca yang terjadi selama menit itu). Untuk informasi lebih lanjut tentang membandingkan rata-rata kapasitas baca yang dikonsumsi dengan kapasitas baca yang disediakan, lihat <a href="#">the section called “Menggunakan metrik”</a></p> </div> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Minimum</b>— Jumlah minimum unit kapasitas baca yang dikonsumsi oleh setiap permintaan individu ke meja.</li> <li>• <b>Maximum</b>— Jumlah maksimum unit kapasitas baca yang dikonsumsi oleh setiap permintaan individu ke meja.</li> <li>• <b>Average</b> – Rata-rata kapasitas baca per permintaan yang digunakan.</li> </ul> |

| Metrik | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|        | <div data-bbox="716 212 1507 428"> <b>Note</b><br/>Nilai Average dipengaruhi oleh periode tidak aktif dengan nilai sampel akan menjadi nol.</div> <ul style="list-style-type: none"><li>• <b>Sum</b> – Total unit kapasitas baca yang digunakan. Ini adalah statistik yang paling berguna untuk metrik <code>ConsumedReadCapacityUnits</code> .</li><li>• <b>SampleCount</b> — Jumlah permintaan ke Amazon Keyspaces, bahkan jika tidak ada kapasitas baca yang dikonsumsi.</li></ul> <div data-bbox="716 768 1507 1035"> <b>Note</b><br/>Nilai <code>SampleCount</code> dipengaruhi oleh periode tidak aktif dengan nilai sampel akan menjadi nol.</div> |

| Metrik                     | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ConsumedWriteCapacityUnits | <p>Jumlah unit kapasitas tulis yang dikonsumsi selama periode waktu yang ditentukan. Anda dapat mengambil total kapasitas tulis yang dikonsumsi untuk sebuah tabel. Untuk informasi selengkapnya, lihat <a href="#">mode Read/Write kapasitas</a>.</p> <div data-bbox="690 493 1510 1239"> <p> <b>Note</b></p> <p>Untuk memahami pemanfaatan throughput rata-rata per detik, gunakan Sum statistik untuk menghitung throughput yang dikonsumsi untuk periode satu menit. Kemudian bagi jumlah dengan jumlah detik dalam satu menit (60) untuk menghitung rata-rata ConsumedWriteCapacityUnits per detik (mengakui bahwa rata-rata ini tidak menyoroti lonjakan besar tetapi singkat dalam aktivitas menulis yang terjadi selama menit itu). Untuk informasi lebih lanjut tentang membandingkan rata-rata kapasitas tulis yang dikonsumsi dengan kapasitas tulis yang disediakan, lihat <a href="#">the section called “Menggunakan metrik”</a></p> </div> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Minimum</b>— Jumlah minimum unit kapasitas tulis yang dikonsumsi oleh setiap permintaan individu ke meja.</li> <li>• <b>Maximum</b>— Jumlah maksimum unit kapasitas tulis yang dikonsumsi oleh setiap permintaan individu ke meja.</li> <li>• <b>Average</b> – Rata-rata kapasitas tulis per permintaan yang digunakan.</li> </ul> |



| Metrik | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|        | <div data-bbox="719 210 1507 426">  <b>Note</b><br/>           Nilai Average dipengaruhi oleh periode tidak aktif dengan nilai sampel akan menjadi nol.         </div> <ul style="list-style-type: none"> <li>• <b>Sum</b> – Total unit kapasitas tulis yang digunakan. Ini adalah statistik yang paling berguna untuk metrik <code>ConsumedWriteCapacityUnits</code> .</li> <li>• <b>SampleCount</b> — Jumlah permintaan ke Amazon Keyspaces, bahkan jika tidak ada kapasitas tulis yang dikonsumsi.</li> </ul> <div data-bbox="719 768 1507 1035">  <b>Note</b><br/>           Nilai <code>SampleCount</code> dipengaruhi oleh periode tidak aktif dengan nilai sampel akan menjadi nol.         </div> |

| Metrik                                     | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MaxProvisionedTableReadCapacityUtilization | <p>Persentase unit kapasitas baca yang disediakan yang digunakan oleh tabel baca akun tertinggi yang disediakan.</p> <p>Unit: Percent</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• <b>Maximum</b> – Persentase maksimum unit kapasitas baca yang disediakan dan digunakan oleh tabel baca tertinggi yang disediakan pada akun.</li><li>• <b>Minimum</b> – Persentase minimum unit kapasitas baca yang disediakan dan digunakan oleh tabel baca tertinggi yang disediakan pada akun.</li><li>• <b>Average</b> – Persentase rata-rata unit kapasitas baca yang disediakan dan digunakan oleh tabel baca tertinggi yang disediakan pada akun. Metrik diterbitkan selama interval lima menit. Oleh karena itu, jika Anda dengan cepat menyesuaikan unit kapasitas baca yang disediakan, statistik ini mungkin tidak mencerminkan rata-rata sebenarnya.</li></ul> |

| Metrik                                      | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MaxProvisionedTableWriteCapacityUtilization | <p>Persentase kapasitas tulis yang disediakan yang digunakan oleh tabel tulis akun tertinggi yang disediakan.</p> <p>Unit: Percent</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• <b>Maximum</b>— Persentase maksimum unit kapasitas tulis yang disediakan yang digunakan oleh tabel tulis akun tertinggi yang disediakan.</li><li>• <b>Minimum</b>— Persentase minimum unit kapasitas tulis yang disediakan yang digunakan oleh tabel tulis akun tertinggi yang disediakan.</li><li>• <b>Average</b>— Persentase rata-rata unit kapasitas tulis yang disediakan yang digunakan oleh tabel tulis akun tertinggi yang disediakan. Metrik diterbitkan selama interval lima menit. Oleh karena itu, jika Anda dengan cepat menyesuaikan unit kapasitas tulis yang disediakan, statistik ini mungkin tidak mencerminkan rata-rata sebenarnya.</li></ul> |

| Metrik                                        | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>PerConnectionRequestRateExceeded</code> | <p>Permintaan ke Amazon Keyspaces yang melebihi kuota tingkat permintaan per koneksi. Setiap koneksi klien ke Amazon Keyspaces dapat mendukung hingga 3000 permintaan CQL per detik. Klien dapat membuat beberapa koneksi untuk meningkatkan throughput.</p> <p>Saat Anda menggunakan replikasi Multi-wilayah, setiap penulisan yang direplikasi juga berkontribusi pada kuota ini. Sebagai praktik terbaik, kami sarankan untuk meningkatkan jumlah koneksi ke tabel Anda untuk menghindari <code>PerConnectionRequestRateExceeded</code> kesalahan. Tidak ada batasan jumlah koneksi yang dapat Anda miliki di Amazon Keyspaces.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• <code>SampleCount</code></li><li>• <code>Sum</code></li></ul> |

| Metrik                       | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ProvisionedReadCapacityUnits | <p>Jumlah unit kapasitas baca yang disediakan untuk sebuah tabel.</p> <p>TableName Dimensi mengembalikan ProvisionedReadCapacityUnits untuk tabel.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• <b>Minimum</b> – Pengaturan terendah untuk kapasitas baca yang disediakan. Jika Anda menggunakan ALTER TABLE untuk menambah kapasitas baca, metrik ini menunjukkan nilai terendah ReadCapacityUnits yang disediakan selama periode waktu ini.</li> <li>• <b>Maximum</b> – Pengaturan tertinggi untuk kapasitas baca yang disediakan. Jika Anda menggunakan ALTER TABLE untuk mengurangi kapasitas baca, metrik ini menunjukkan nilai tertinggi ReadCapacityUnits yang disediakan selama periode waktu ini.</li> <li>• <b>Average</b> – Rata-rata kapasitas baca yang disediakan. Metrik ProvisionedReadCapacityUnits diterbitkan pada interval lima menit. Oleh karena itu, jika Anda dengan cepat menyesuaikan unit kapasitas baca yang disediakan, statistik ini mungkin tidak mencerminkan rata-rata sebenarnya.</li> </ul> |

| Metrik                        | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ProvisionedWriteCapacityUnits | <p>Jumlah unit kapasitas tulis yang disediakan untuk sebuah tabel.</p> <p>TableName Dimensi mengembalikan ProvisionedWriteCapacityUnits untuk tabel.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• Minimum – Pengaturan terendah untuk kapasitas tulis yang disediakan. Jika Anda menggunakan ALTER TABLE untuk menambah kapasitas tulis, metrik ini menunjukkan nilai terendah WriteCapacityUnits yang disediakan selama periode waktu ini.</li><li>• Maximum – Pengaturan tertinggi untuk kapasitas tulis yang disediakan. Jika Anda menggunakan ALTER TABLE untuk mengurangi kapasitas tulis, metrik ini menunjukkan nilai tertinggi WriteCapacityUnits yang disediakan selama periode waktu ini.</li><li>• Average – Rata-rata kapasitas tulis yang disediakan. Metrik ProvisionedWriteCapacityUnits diterbitkan pada interval lima menit. Oleh karena itu, jika Anda dengan cepat menyesuaikan unit kapasitas tulis yang disediakan, statistik ini mungkin tidak mencerminkan rata-rata sebenarnya.</li></ul> |

| Metrik             | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ReadThrottleEvents | <p>Permintaan ke Amazon Keyspaces yang melebihi kapasitas baca yang disediakan untuk tabel, atau kuota tingkat akun, permintaan per kuota koneksi, atau kuota tingkat partisi.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• SampleCount</li><li>• Sum</li></ul>                                                                    |
| ReplicationLatency | <p>Metrik ini hanya berlaku untuk ruang kunci Multi-wilayah dan mengukur waktu yang diperlukan untuk mereplika siupdates,inserts, atau deletes dari satu tabel replika ke tabel replika lainnya di ruang kunci Multi-wilayah.</p> <p>Unit: Millisecond</p> <p>Dimensi: TableName, ReceivingRegion</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• Average</li><li>• Maximum</li><li>• Minimum</li></ul> |

| Metrik                    | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ReturnedItemCountBySelect | <p>Jumlah baris yang dikembalikan oleh SELECT kueri multi-baris selama periode waktu yang ditentukan. Multi-row SELECT kueri adalah kueri yang tidak berisi kunci utama yang memenuhi syarat, seperti pemindaian tabel lengkap dan kueri rentang.</p> <p>Jumlah baris yang dikembalikan belum tentu sama dengan jumlah baris yang dievaluasi. Misalnya, Anda meminta <code>SELECT *</code> dengan <code>ALLOW FILTERING</code> pada tabel yang memiliki 100 baris, tetapi menentukan <code>WHERE</code> klausa yang mempersempit hasil sehingga hanya 15 baris yang dikembalikan. Dalam hal ini, respons dari SELECT akan berisi <code>ScanCount</code> 100 dan 15 baris <code>Count</code> yang dikembalikan.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• Minimum</li><li>• Maximum</li><li>• Average</li><li>• SampleCount</li><li>• Sum</li></ul> |



| Metrik                                     | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| StoragePartitionThroughputCapacityExceeded | <p>Permintaan ke partisi penyimpanan Amazon Keyspaces yang melebihi kapasitas throughput partisi. Partisi penyimpanan Amazon Keyspaces dapat mendukung hingga 1000 WCU/WRU per detik dan 3000 RCU/RRU per detik. Sebaiknya tinjau model data Anda untuk mendistribusikan read/write lalu lintas di lebih banyak partisi untuk mengurangi pengecualian ini.</p> <div>  <b>Note</b><br/> Partisi Amazon Keyspaces yang logis dapat menjangkau beberapa partisi penyimpanan dan ukurannya hampir tidak terbatas. </div> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• SampleCount</li> <li>• Sum</li> </ul> |
| SuccessfulRequestCount                     | <p>Jumlah permintaan yang berhasil diproses selama periode waktu yang ditentukan.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• SampleCount</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

| Metrik                   | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SuccessfulRequestLatency | <p>Permintaan yang berhasil ke Amazon Keyspaces selama periode waktu yang ditentukan. SuccessfulRequestLatency dapat memberikan dua jenis informasi yang berbeda:</p> <ul style="list-style-type: none"><li>• Waktu berlalu untuk permintaan yang berhasil (Minimum, Maximum, Sum, atau Average).</li><li>• Jumlah permintaan yang berhasil (SampleCount ).</li></ul> <p>SuccessfulRequestLatency mencerminkan aktivitas hanya dalam Amazon Keyspaces dan tidak memperhitungkan latensi jaringan atau aktivitas sisi klien.</p> <p>Unit: Milliseconds</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• Minimum</li><li>• Maximum</li><li>• Average</li><li>• SampleCount</li></ul> |

| Metrik                      | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SystemErrors                | <p>Permintaan ke Amazon Keyspaces yang menghasilkan a <code>ServerError</code> selama periode waktu yang ditentukan. A <code>ServerError</code> biasanya menunjukkan kesalahan layanan internal.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• Sum</li> <li>• SampleCount</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| SystemReconciliationDeletes | <p>Unit yang digunakan untuk menghapus data batuan saat stempel waktu sisi klien diaktifkan. Masing-masing <code>SystemReconciliationDelete</code> menyediakan kapasitas yang cukup untuk menghapus atau memperbaiki hingga 1KB data per baris. Misalnya, untuk memperbaiki baris yang menyimpan 2,5 KB data dan untuk menghapus satu atau lebih kolom dalam baris pada saat yang sama membutuhkan 3 <code>SystemReconciliationDeletes</code>. Atau, untuk menghapus seluruh baris yang berisi 3,5 KB data membutuhkan 4 <code>SystemReconciliationDeletes</code>.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• Sum— Jumlah total yang <code>SystemReconciliationDeletes</code> dikonsumsi dalam jangka waktu tertentu.</li> </ul> |

| Metrik     | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TTLDeletes | <p>Unit yang digunakan untuk menghapus atau memperbarui data berturut-turut dengan menggunakan Time to Live (TTL). Masing-masing TTLDelete menyediakan kapasitas yang cukup untuk menghapus atau memperbarui hingga 1KB data per baris. Misalnya, untuk memperbarui baris yang menyimpan 2,5 KB data dan untuk menghapus satu atau lebih kolom dalam baris pada saat yang sama membutuhkan 3 penghapusan TTL. Atau, untuk menghapus seluruh baris yang berisi 3,5 KB data membutuhkan 4 penghapusan TTL.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"><li>• Sum— Jumlah total yang TTLDeletes dikonsumsi dalam jangka waktu tertentu.</li></ul> |

| Metrik              | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UserErrors          | <p>Permintaan ke Amazon Keyspaces yang menghasilkan <code>InvalidRequest</code> kesalahan selama periode waktu yang ditentukan.</p> <p><code>InvalidRequest</code> kesalahan biasanya mewakili kesalahan sisi klien, misalnya kombinasi parameter yang tidak valid, upaya untuk memperbarui tabel yang tidak ada, atau tanda tangan permintaan yang salah.</p> <p><code>UserErrors</code> mewakili agregat permintaan yang tidak valid untuk saat ini Wilayah AWS dan saat ini. Akun AWS</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• Sum</li> <li>• SampleCount</li> </ul> |
| WriteThrottleEvents | <p>Permintaan ke Amazon Keyspaces yang melebihi kapasitas penulisan yang disediakan untuk tabel, atau kuota tingkat akun, permintaan per kuota koneksi, atau kuota tingkat partisi.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• SampleCount</li> <li>• Sum</li> </ul>                                                                                                                                                                                                                                                                                                      |

## Dimensi untuk metrik Amazon Keyspaces

Metrik untuk Amazon Keyspaces dikualifikasikan berdasarkan nilai untuk akun, nama tabel, atau operasi. Anda dapat menggunakan CloudWatch konsol untuk mengambil data Amazon Keyspaces sepanjang salah satu dimensi dalam tabel berikut.

| Dimensi   | Deskripsi                                                                                                                                                                                                        |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Keyspace  | Dimensi ini membatasi data ke ruang kunci tertentu. Nilai ini dapat berupa ruang kunci apa pun di Wilayah saat ini dan saat ini Akun AWS.                                                                        |
| Operation | Dimensi ini membatasi data ke salah satu operasi CQL Amazon Keyspaces, seperti atau operasi. INSERT SELECT                                                                                                       |
| TableName | Dimensi ini membatasi data ke tabel tertentu. Nilai ini dapat berupa nama tabel apa pun di Wilayah saat ini dan saat ini Akun AWS. Jika nama tabel tidak unik di dalam akun, Anda juga harus menentukanKeyspace. |

## Metrik untuk Amazon Keyspaces mengubah pengambilan data (CDC)

Metrik berikut tersedia di CloudWatch akun Anda yang memantau panggilan API ke titik akhir perubahan pengambilan data (CDC) Amazon Keyspaces.

| Metrik               | Deskripsi                                                                                                                                                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ReturnedRecordsCount | <p>Jumlah catatan aliran yang dikembalikan oleh GetRecords operasi selama periode waktu yang ditentukan.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>Minimum</li> </ul> |

| Metrik                   | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                          | <ul style="list-style-type: none"> <li>• Maximum</li> <li>• Average</li> <li>• SampleCount</li> <li>• Sum</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| SuccessfulRequestLatency | <p>Permintaan yang berhasil ke Amazon Keyspaces selama periode waktu yang ditentukan. SuccessfulRequestLatency dapat memberikan dua jenis informasi yang berbeda:</p> <ul style="list-style-type: none"> <li>• Waktu berlalu untuk permintaan yang berhasil (Minimum, Maximum, Sum, atau Average).</li> <li>• Jumlah permintaan yang berhasil (SampleCount ).</li> </ul> <p>SuccessfulRequestLatency mencerminkan aktivitas hanya dalam Amazon Keyspaces CDC API dan tidak memperhitungkan latensi jaringan atau aktivitas sisi klien.</p> <p>Unit: Milliseconds</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• Minimum</li> <li>• Maximum</li> <li>• Average</li> <li>• SampleCount</li> </ul> |

| Metrik       | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SystemErrors | <p>Permintaan ke Amazon Keyspaces CDC yang menghasilkan kode HTTP 500 status selama periode waktu yang ditentukan. Kode HTTP 500 status biasanya menunjukkan kesalahan layanan internal.</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• Sum</li> <li>• SampleCount</li> </ul>                                                        |
| UserErrors   | <p>Permintaan ke Amazon Keyspaces CDC yang menghasilkan kode HTTP 400 status selama periode waktu yang ditentukan.</p> <p>UserErrors mewakili agregat HTTP 400 kesalahan untuk Amazon Keyspaces CDC saat ini dan Wilayah AWS saat ini. Akun AWS</p> <p>Unit: Count</p> <p>Dimensi: Keyspace, TableName, Operation</p> <p>Statistik Valid:</p> <ul style="list-style-type: none"> <li>• Sum</li> <li>• SampleCount</li> </ul> |

## Dimensi untuk Amazon Keyspaces mengubah metrik pengambilan data (CDC)

Metrik untuk Amazon Keyspaces dikualifikasikan berdasarkan nilai untuk akun, nama tabel, atau operasi. Anda dapat menggunakan CloudWatch konsol untuk mengambil data Amazon Keyspaces sepanjang salah satu dimensi dalam tabel berikut.



| Dimensi                  | Deskripsi                                                                                                                                                                                                                       |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>StreamLabel</code> | Dimensi ini membatasi data ke aliran pengambilan data perubahan tertentu (CDC) kapan <code>KeyspaceName</code> dan <code>TableName</code> ditentukan.                                                                           |
| <code>Keyspace</code>    | Dimensi ini membatasi data ke ruang kunci tertentu. Nilai ini dapat berupa ruang kunci apa pun di Wilayah saat ini dan saat ini Akun AWS.                                                                                       |
| <code>TableName</code>   | Dimensi ini membatasi data ke tabel tertentu. Nilai ini dapat berupa nama tabel apa pun di Wilayah saat ini dan saat ini Akun AWS. Jika nama tabel tidak unik di dalam akun, Anda juga harus menentukan <code>Keyspace</code> . |
| <code>Operation</code>   | Dimensi ini membatasi data ke salah satu operasi API Amazon Keyspaces Streams berikut: <code>GetRecords</code> , <code>GetStream</code> <code>GetShardIterator</code> dan <code>ListStreams</code>                              |

## Melihat metrik Amazon Keyspaces di CloudWatch

Saat Anda berinteraksi dengan Amazon Keyspaces, Amazon mengirimkan metrik dan dimensi berikut ke Amazon. CloudWatch Semua metrik dikumpulkan dan dilaporkan setiap menit. Anda dapat menggunakan prosedur berikut untuk melihat metrik untuk Amazon Keyspaces.

Untuk melihat metrik menggunakan konsol CloudWatch

Metrik dikelompokkan terlebih dahulu berdasarkan namespace layanan, lalu berdasarkan berbagai kombinasi dimensi dalam setiap namespace.

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Jika perlu, ubah Wilayah AWS. Pada bilah navigasi, pilih Wilayah tempat AWS sumber daya Anda berada. Untuk informasi selengkapnya, lihat [AWS titik akhir layanan](#).
3. Pada panel navigasi, silakan pilih Metrik.
4. Di bawah tab Semua metrik, pilih AWS/Cassandra.

## Untuk melihat metrik menggunakan CLI AWS

- Pada prompt perintah, gunakan perintah berikut.

```
aws cloudwatch list-metrics --namespace "AWS/Cassandra"
```

## Membuat CloudWatch alarm untuk memantau Amazon Keyspaces

Anda dapat membuat CloudWatch alarm Amazon untuk Amazon Keyspaces yang mengirimkan pesan Amazon Simple Notification Service (Amazon SNS) saat alarm berubah status. Alarm mengawasi metrik tunggal selama periode waktu yang Anda tentukan. Alarm tersebut melakukan satu atau beberapa tindakan berdasarkan nilai metrik yang relatif terhadap ambang batas tertentu selama beberapa periode waktu. Tindakan ini adalah pemberitahuan yang dikirim ke topik Amazon SNS atau kebijakan Application Auto Scaling.

Saat Anda menggunakan Amazon Keyspaces dalam mode yang disediakan dengan Application Auto Scaling, layanan akan membuat dua pasang alarm atas nama Anda. CloudWatch Setiap pasangan mewakili batas atas dan bawah Anda untuk pengaturan throughput yang disediakan dan dikonsumsi. CloudWatch Alarm ini dipicu ketika penggunaan tabel yang sebenarnya menyimpang dari penggunaan target Anda untuk jangka waktu yang berkelanjutan. Untuk mempelajari lebih lanjut tentang CloudWatch alarm yang dibuat oleh Application Auto Scaling, lihat [the section called “Cara kerja penskalaan otomatis Amazon Keyspaces”](#)

Alarm memanggil tindakan untuk perubahan status berkelanjutan saja. CloudWatch alarm tidak memanggil tindakan hanya karena mereka berada dalam keadaan tertentu. Status harus diubah dan dipertahankan selama jangka waktu tertentu.

Untuk informasi selengkapnya tentang membuat CloudWatch alarm, lihat [Menggunakan CloudWatch alarm Amazon](#) di CloudWatch Panduan Pengguna Amazon.

## Mencatat panggilan API Amazon Keyspaces dengan AWS CloudTrail

Amazon Keyspaces terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon Keyspaces. CloudTrail menangkap panggilan API Data Definition Language (DDL) dan panggilan API Data Manipulation Language (DHTML) untuk Amazon Keyspaces sebagai peristiwa. Panggilan yang diambil termasuk

panggilan dari konsol Amazon Keyspaces dan panggilan terprogram ke operasi Amazon Keyspaces API.

Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon Simple Storage Service (Amazon S3), termasuk peristiwa untuk Amazon Keyspaces.

Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat acara terbaru yang didukung di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke Amazon Keyspaces, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat [Panduan AWS CloudTrail Pengguna](#).

## Topik

- [Mengonfigurasi entri file log Amazon Keyspaces di CloudTrail](#)
- [Informasi Amazon Keyspaces Data Definition Language \(DDL\) di CloudTrail](#)
- [Informasi Amazon Keyspaces Data Manipulation Language \(DHTML\) di CloudTrail](#)
- [Memahami entri file log Amazon Keyspaces](#)

## Mengonfigurasi entri file log Amazon Keyspaces di CloudTrail

Setiap tindakan Amazon Keyspaces API yang masuk CloudTrail menyertakan parameter permintaan yang dinyatakan dalam bahasa kueri CQL. Untuk informasi selengkapnya, lihat [Referensi bahasa CQL](#).

Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di Akun AWS Anda. Untuk informasi selengkapnya, lihat [Melihat peristiwa dengan riwayat CloudTrail acara](#).

Untuk catatan peristiwa yang sedang berlangsung di Anda Akun AWS, termasuk acara untuk Amazon Keyspaces, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua AWS Wilayah. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log.

Untuk informasi selengkapnya, lihat topik berikut di Panduan Pengguna AWS CloudTrail :

- [Ikhtisar untuk membuat jejak](#)
- [CloudTrail layanan dan integrasi yang didukung](#)
- [Mengonfigurasi notifikasi Amazon SNS untuk CloudTrail](#)
- [Menerima file CloudTrail log dari beberapa Wilayah](#)
- [Menerima file CloudTrail log dari beberapa akun](#)

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan itu dibuat dengan kredensial pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat [Elemen userIdentity CloudTrail](#).

## Informasi Amazon Keyspaces Data Definition Language (DDL) di CloudTrail

CloudTrail diaktifkan pada Akun AWS saat Anda membuat akun. Ketika aktivitas DDL terjadi di Amazon Keyspaces, aktivitas tersebut secara otomatis direkam CloudTrail sebagai peristiwa bersama dengan peristiwa layanan AWS lainnya dalam riwayat Acara. Tabel berikut menunjukkan pernyataan DDL yang dicatat untuk Amazon Keyspaces.

| CloudTrail<br><b>eventName</b> | Pernyataan | Tindakan CQL    | AWS Tindakan SDK |
|--------------------------------|------------|-----------------|------------------|
| CreateKeyspace                 | DDL        | CREATE KEYSPACE | CreateKeyspace   |
| AlterKeyspace                  | DDL        | ALTER KEYSPACE  | UpdateKeyspace   |
| DropKeyspace                   | DDL        | DROP KEYSPACE   | DeleteKeyspace   |
| CreateTable                    | DDL        | CREATE TABLE    | CreateTable      |
| DropTable                      | DDL        | DROP TABLE      | DeleteTable      |

| CloudTrail eventName | Pernyataan | Tindakan CQL                         | AWS Tindakan SDK                                |
|----------------------|------------|--------------------------------------|-------------------------------------------------|
| AlterTable           | DDL        | ALTER TABLE                          | UpdateTable ,<br>TagResource ,<br>UntagResource |
| CreateUdt            | DDL        | CREATE TYPE                          | CreateType                                      |
| DropUdt              | DDL        | DROP TYPE                            | DeleteType                                      |
| GetStream            | DDL        | tidak ada tindakan CQL yang tersedia | GetStream                                       |
| ListStreams          | DDL        | tidak ada tindakan CQL yang tersedia | ListStreams                                     |

## Informasi Amazon Keyspaces Data Manipulation Language (DHTML) di CloudTrail

Untuk mengaktifkan pencatatan pernyataan DHTML Amazon Keyspaces dengan CloudTrail, Anda harus terlebih dahulu mengaktifkan pencatatan aktivitas API bidang data di CloudTrail. Anda dapat mulai mencatat peristiwa DHTML Amazon Keyspaces di jalur baru atau yang sudah ada dengan memilih untuk mencatat aktivitas untuk tabel Cassandra tipe peristiwa data menggunakan CloudTrail konsol, atau dengan menyetel nilai `resources.type` menggunakan `AWS::Cassandra::Table` CLI AWS, atau operasi API. CloudTrail. Untuk informasi selengkapnya, lihat [Mencatat peristiwa data](#).

Untuk informasi selengkapnya dan contoh yang menunjukkan cara membuat alarm untuk peristiwa data, lihat posting berikut di blog AWS Database [Menggunakan audit DHTML untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#).

Tabel berikut menunjukkan peristiwa data yang dicatat oleh CloudTrail untuk `Cassandra` table.

| CloudTrail eventName | Pernyataan | Tindakan CQL | AWS Tindakan SDK                    |
|----------------------|------------|--------------|-------------------------------------|
| Select               | DML        | SELECT       | GetKeyspace ,<br>GetTable, GetType, |

| CloudTrail eventName | Pernyataan | Tindakan CQL                         | AWS Tindakan SDK                                          |
|----------------------|------------|--------------------------------------|-----------------------------------------------------------|
|                      |            |                                      | ListKeyspaces, ListTables, ListTypes, ListTagsForResource |
| Sisipkan             | DML        | INSERT                               | tidak ada tindakan AWS SDK yang tersedia                  |
| Perbarui             | DML        | UPDATE                               | tidak ada tindakan AWS SDK yang tersedia                  |
| Delete               | DML        | DELETE                               | tidak ada tindakan AWS SDK yang tersedia                  |
| GetShardIterator     | DML        | tidak ada tindakan CQL yang tersedia | GetShardIterator                                          |
| GetRecords           | DML        | tidak ada tindakan CQL yang tersedia | GetRecords                                                |

## Memahami entri file log Amazon Keyspaces

CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateKeyspace`, `DropKeyspace`, `CreateTable`, dan `DropTable` tindakan:

```
{
 "Records": [
```

```

{
 "eventVersion": "1.05",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
 "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
 "accountId": "111122223333",
 "sessionContext": {
 "sessionIssuer": {
 "type": "Role",
 "principalId": "AKIAIOSFODNN7EXAMPLE",
 "arn": "arn:aws:iam::111122223333:role/Admin",
 "accountId": "111122223333",
 "userName": "Admin"
 },
 "webIdFederationData": {},
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2020-01-15T18:47:56Z"
 }
 }
 },
 "eventTime": "2020-01-15T18:53:04Z",
 "eventSource": "cassandra.amazonaws.com",
 "eventName": "CreateKeyspace",
 "awsRegion": "us-east-1",
 "sourceIPAddress": "10.24.34.01",
 "userAgent": "Cassandra Client/ProtocolV4",
 "requestParameters": {
 "rawQuery": "\n\tCREATE KEYSPACE \"mykeyspace\"\n\tWITH\n\t\tREPLICATION =
{'class': 'SingleRegionStrategy'}\n\t\t",
 "keyspaceName": "mykeyspace"
 },
 "responseElements": null,
 "requestID": "bfa3e75d-bf4d-4fc0-be5e-89d15850eb41",
 "eventID": "d25beae8-f611-4229-877a-921557a07bb9",
 "readOnly": false,
 "resources": [
 {
 "accountId": "111122223333",
 "type": "AWS::Cassandra::Keyspace",
 "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
 }
],

```

```

 "eventType": "AwsApiCall",
 "apiVersion": "3.4.4",
 "recipientAccountId": "111122223333",
 "managementEvent": true,
 "eventCategory": "Management",
 "tlsDetails": {
 "tlsVersion": "TLSv1.2",
 "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
 "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
 },
 },
 {
 "eventVersion": "1.05",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
 "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
 "accountId": "111122223333",
 "sessionContext": {
 "sessionIssuer": {
 "type": "Role",
 "principalId": "AKIAIOSFODNN7EXAMPLE",
 "arn": "arn:aws:iam::111122223333:role/Admin",
 "accountId": "111122223333",
 "userName": "Admin"
 },
 "webIdFederationData": {},
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2020-01-15T18:47:56Z"
 }
 }
 },
 },
 "eventTime": "2020-01-15T19:28:39Z",
 "eventSource": "cassandra.amazonaws.com",
 "eventName": "DropKeyspace",
 "awsRegion": "us-east-1",
 "sourceIPAddress": "10.24.34.01",
 "userAgent": "Cassandra Client/ProtocolV4",
 "requestParameters": {
 "rawQuery": "DROP KEYSPACE \"mykeyspace\"",
 "keyspaceName": "mykeyspace"
 },
 "responseElements": null,
 "requestID": "66f3d86a-56ae-4c29-b46f-abcd489ed86b",

```



```

 "eventID": "e5aebec-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\tID\` int,
\n\t\t\tusername\` text,\n\t\t\temail\` text,\n\t\t\tpost_type\` text,\n\t\t\tPRIMARY
KEY((\`ID\`, \`username\`, \`email\`)))",
 "keyspaceName": "mykeyspace",
 "tableName": "mytable"
},
"responseElements": null,
"requestID": "5f845963-70ea-4988-8a7a-2e66d061aacb",
"eventID": "fe0dbd2b-7b34-4675-a30c-740f9d8d73f9",
"readOnly": false,
"resources": [
 {
 "accountId": "111122223333",
 "type": "AWS::Cassandra::Table",
 "ARN": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable"
 }
],
"eventType": "AwsApiCall",
"apiVersion": "3.4.4",
"recipientAccountId": "111122223333",
"managementEvent": true,
"eventCategory": "Management",
"tlsDetails": {
 "tlsVersion": "TLSv1.2",
 "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
 "clientProvidedHostHeader": "cassandra.us-east-1.amazonaws.com"
},
{
 "eventVersion": "1.05",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "AKIAIOSFODNN7EXAMPLE:alice",
 "arn": "arn:aws:sts::111122223333:assumed-role/users/alice",
 "accountId": "111122223333",
 "sessionContext": {
 "sessionIssuer": {
 "type": "Role",
 "principalId": "AKIAIOSFODNN7EXAMPLE",
 "arn": "arn:aws:iam::111122223333:role/Admin",
```

```

 "accountId": "111122223333",
 "userName": "Admin"
 },
 "webIdFederationData": {},
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2020-01-15T18:47:56Z"
 }
},
"eventTime": "2020-01-15T19:27:59Z",
"eventSource": "cassandra.amazonaws.com",
"eventName": "DropTable",
"awsRegion": "us-east-1",
"sourceIPAddress": "10.24.34.01",
"userAgent": "Cassandra Client/ProtocolV4",
"requestParameters": {
 "rawQuery": "DROP TABLE \"mykeyspace\".\"mytable\"",
 "keyspaceName": "mykeyspace",
 "tableName": "mytable"
},
"responseElements": null,
"requestID": "025501b0-3582-437e-9d18-8939e9ef262f",
"eventID": "1a5cbcdc-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"
}
]

```

```
}
```

File log berikut menunjukkan contoh SELECT pernyataan.

```
{
 "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"
 }
}

```

File log berikut menunjukkan contoh INSERT pernyataan.

```

{
 "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"
 }
}

```

File log berikut menunjukkan contoh UPDATE pernyataan.

```

{
 "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"
 }
 }
}

```

File log berikut menunjukkan contoh DELETE pernyataan.

```
{
 "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"
}
}
```

# Keamanan di Amazon Keyspaces (untuk Apache Cassandra)

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model tanggung jawab bersama](#) menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Efektivitas keamanan kami diuji dan diverifikasi secara rutin oleh auditor pihak ketiga sebagai bagian dari [program kepatuhan AWS](#). Untuk mempelajari tentang program kepatuhan yang berlaku untuk Amazon Keyspaces, lihat [AWS Layanan dalam cakupan berdasarkan program kepatuhan](#).
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor-faktor lain termasuk sensitivitas data, kebutuhan organisasi, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon Keyspaces. Topik berikut menunjukkan cara mengonfigurasi Amazon Keyspaces untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga akan mempelajari cara menggunakan AWS layanan lain yang dapat membantu Anda memantau dan mengamankan sumber daya Amazon Keyspaces Anda.

## Topik

- [Perlindungan data di Amazon Keyspaces](#)
- [AWS Identity and Access Management untuk Amazon Keyspaces](#)
- [Validasi kepatuhan untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#)
- [Ketahanan dan pemulihan bencana di Amazon Keyspaces](#)
- [Keamanan infrastruktur di Amazon Keyspaces](#)
- [Analisis konfigurasi dan kerentanan untuk Amazon Keyspaces](#)
- [Praktik terbaik keamanan untuk Amazon Keyspaces](#)

# Perlindungan data di Amazon Keyspaces

[Model tanggung jawab AWS bersama model](#) berlaku untuk perlindungan data di Amazon Keyspaces (untuk Apache Cassandra). Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, lihat [FAQ Privasi Data AWS](#). Untuk informasi tentang perlindungan data di Eropa, lihat [Pusat Peraturan Perlindungan Data Umum \(GDPR\)](#).

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensi dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat [Bekerja dengan CloudTrail jejak](#) di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di [Standar Pemrosesan Informasi Federal \(FIPS\) 140-3](#).

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan Amazon Keyspaces atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log

penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

## Topik

- [Enkripsi saat istirahat di Amazon Keyspaces](#)
- [Enkripsi dalam perjalanan di Amazon Keyspaces](#)
- [Privasi lalu lintas internetwork di Amazon Keyspaces](#)

## Enkripsi saat istirahat di Amazon Keyspaces

[Enkripsi Amazon Keyspaces \(untuk Apache Cassandra\) saat istirahat memberikan keamanan yang ditingkatkan dengan mengenkripsi semua data Anda saat istirahat menggunakan kunci enkripsi yang disimpan di \(.AWS Key Management Service\) AWS KMS](#) Fungsi ini membantu mengurangi beban operasional dan kompleksitas yang terlibat dalam melindungi data sensitif. Dengan enkripsi saat istirahat, Anda dapat membangun aplikasi yang sensitif terhadap keamanan yang memenuhi kepatuhan ketat dan persyaratan peraturan untuk perlindungan data.

Enkripsi Amazon Keyspaces saat istirahat mengenkripsi data Anda menggunakan Standar Enkripsi Lanjutan ( ) 256-bit. AES-256 Ini membantu mengamankan data Anda dari akses tidak sah ke penyimpanan yang mendasarinya.

Amazon Keyspaces mengenkripsi dan mendekripsi data dalam tabel dan streaming secara transparan. Amazon Keyspaces menggunakan enkripsi amplop dan hierarki kunci untuk melindungi kunci enkripsi data. Ini terintegrasi dengan AWS KMS untuk menyimpan dan mengelola kunci enkripsi root. Untuk informasi selengkapnya tentang hierarki kunci enkripsi, lihat [the section called “Cara kerjanya”](#). Untuk informasi selengkapnya tentang AWS KMS konsep seperti enkripsi amplop, lihat [konsep layanan AWS KMS manajemen](#) di Panduan AWS Key Management Service Pengembang.

Saat membuat tabel baru, Anda dapat memilih salah satu tombol berikut (AWS KMS tombol KMS):

- Kunci milik AWS — Ini adalah jenis enkripsi default. Kuncinya dimiliki oleh Amazon Keyspaces (tanpa biaya tambahan).
- Kunci yang dikelola pelanggan — Kunci ini disimpan di akun Anda dan dibuat, dimiliki, dan dikelola oleh Anda. Anda memiliki kendali penuh atas kunci yang dikelola pelanggan (AWS KMS dikenakan biaya).

Amazon Keyspaces secara otomatis mengenkripsi aliran change data capture (CDC) dengan kunci yang sama dengan tabel yang mendasarinya. Untuk informasi lebih lanjut tentang CDC, lihat [the section called “Bekerja dengan aliran CDC”](#).

Anda dapat beralih antara kunci yang dikelola pelanggan Kunci milik AWS dan pada waktu tertentu. Anda dapat menentukan kunci terkelola pelanggan saat membuat tabel baru atau mengubah kunci KMS dari tabel yang ada dengan menggunakan konsol atau secara terprogram menggunakan pernyataan CQL. Untuk mempelajari caranya, lihat [Enkripsi saat diam: Cara menggunakan kunci yang dikelola pelanggan untuk mengenkripsi tabel di Amazon Keyspaces](#).

Enkripsi saat istirahat menggunakan opsi default Kunci milik AWS ditawarkan tanpa biaya tambahan. Namun, AWS KMS biaya berlaku untuk kunci yang dikelola pelanggan. Untuk informasi selengkapnya tentang harga, silakan lihat [harga AWS KMS](#).

Enkripsi Amazon Keyspaces saat istirahat tersedia di semua Wilayah AWS, termasuk Wilayah AWS Tiongkok (Beijing) dan China (AWS Ningxia). Lihat informasi yang lebih lengkap di [Enkripsi saat istirahat: Cara kerjanya di Amazon Keyspaces](#).

## Topik

- [Enkripsi saat istirahat: Cara kerjanya di Amazon Keyspaces](#)
- [Enkripsi saat diam: Cara menggunakan kunci yang dikelola pelanggan untuk mengenkripsi tabel di Amazon Keyspaces](#)

## Enkripsi saat istirahat: Cara kerjanya di Amazon Keyspaces

Enkripsi Amazon Keyspaces (untuk Apache Cassandra) saat istirahat mengenkripsi data Anda menggunakan Standar Enkripsi Lanjutan () 256-bit. AES-256 Ini membantu mengamankan data Anda dari akses tidak sah ke penyimpanan yang mendasarinya. Semua data pelanggan di tabel Amazon Keyspaces dienkripsi saat istirahat secara default, dan enkripsi sisi server transparan, yang berarti bahwa perubahan pada aplikasi tidak diperlukan.

Enkripsi saat istirahat terintegrasi dengan AWS Key Management Service (AWS KMS) untuk mengelola kunci enkripsi yang digunakan untuk mengenkripsi tabel Anda. Saat membuat tabel baru atau memperbarui tabel yang ada, Anda dapat memilih salah satu opsi AWS KMS utama berikut:

- Kunci milik AWS — Ini adalah jenis enkripsi default. Kuncinya dimiliki oleh Amazon Keyspaces (tanpa biaya tambahan).

- Kunci yang dikelola pelanggan — Kunci ini disimpan di akun Anda dan dibuat, dimiliki, dan dikelola oleh Anda. Anda memiliki kendali penuh atas kunci yang dikelola pelanggan (AWS KMS dikenakan biaya).

## AWS KMS kunci (kunci KMS)

Enkripsi saat istirahat melindungi semua data Amazon Keyspaces Anda dengan AWS KMS kunci. Secara default, Amazon Keyspaces menggunakan [Kunci milik AWS](#), kunci enkripsi multi-penyewa yang dibuat dan dikelola di akun layanan Amazon Keyspaces.

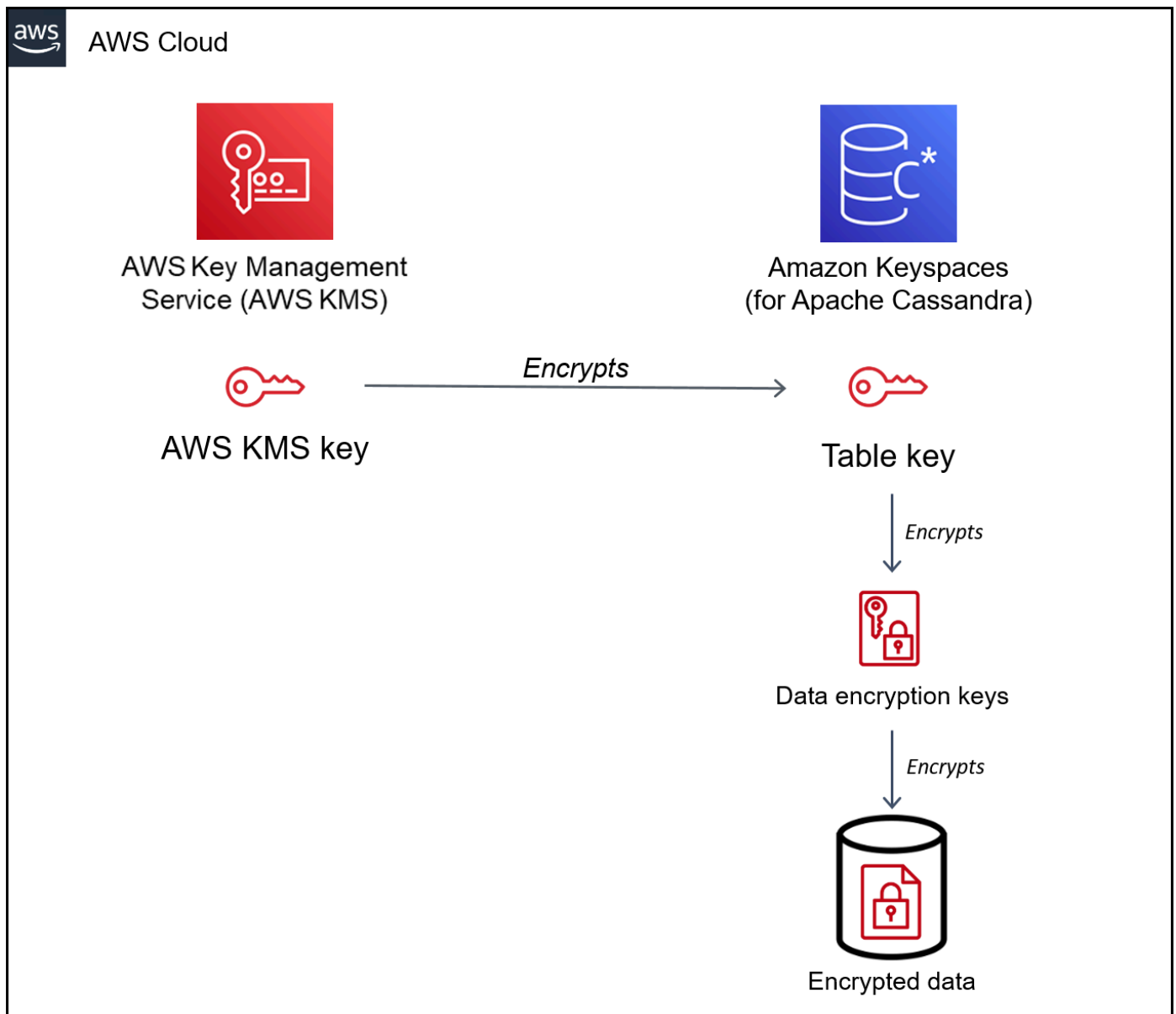
Namun, Anda dapat mengenkripsi tabel Amazon Keyspaces menggunakan kunci [yang dikelola pelanggan di situs](#) Anda. Akun AWS Anda dapat memilih tombol KMS yang berbeda untuk setiap tabel di ruang kunci. Kunci KMS yang Anda pilih untuk tabel juga digunakan untuk mengenkripsi semua metadata dan cadangan yang dapat dipulihkan.

Anda memilih kunci KMS untuk tabel saat membuat atau memperbarui tabel. Anda dapat mengubah kunci KMS untuk tabel kapan saja, baik di konsol Amazon Keyspaces atau dengan menggunakan pernyataan [ALTER](#) TABLE. Proses peralihan kunci KMS mulus, dan tidak memerlukan waktu henti atau menyebabkan degradasi layanan.

## Hirarki kunci

Amazon Keyspaces menggunakan hierarki kunci untuk mengenkripsi data. Dalam hierarki kunci ini, kunci KMS adalah kunci root. Ini digunakan untuk mengenkripsi dan mendekripsi kunci enkripsi tabel Amazon Keyspaces. Kunci enkripsi tabel digunakan untuk mengenkripsi kunci enkripsi yang digunakan secara internal oleh Amazon Keyspaces untuk mengenkripsi dan mendekripsi data saat melakukan operasi baca dan tulis.

Dengan hierarki kunci enkripsi, Anda dapat membuat perubahan pada kunci KMS tanpa harus mengenkripsi ulang data atau memengaruhi aplikasi dan operasi data yang sedang berlangsung.



## Tombol tabel

Tombol tabel Amazon Keyspaces digunakan sebagai kunci enkripsi kunci. Amazon Keyspaces menggunakan tombol tabel untuk melindungi kunci enkripsi data internal yang digunakan untuk mengenkripsi data yang disimpan dalam tabel, file log, dan cadangan yang dapat dipulihkan. Amazon Keyspaces menghasilkan kunci enkripsi data unik untuk setiap struktur yang mendasarinya dalam tabel. Namun, beberapa baris tabel mungkin dilindungi oleh kunci enkripsi data yang sama.

Saat pertama kali mengatur kunci KMS ke kunci yang dikelola pelanggan, AWS KMS buat kunci data. Kunci AWS KMS data mengacu pada kunci tabel di Amazon Keyspaces.

Saat Anda mengakses tabel terenkripsi, Amazon Keyspaces mengirimkan permintaan untuk menggunakan kunci KMS AWS KMS untuk mendekripsi kunci tabel. Kemudian, ia menggunakan kunci tabel plaintext untuk mendekripsi kunci enkripsi data Amazon Keyspaces, dan menggunakan kunci enkripsi data plaintext untuk mendekripsi data tabel.

Amazon Keyspaces menggunakan dan menyimpan kunci tabel dan kunci enkripsi data di luar. AWS KMS Layanan ini melindungi semua kunci dengan enkripsi [Advanced Encryption Standard](#) (AES) dan kunci enkripsi 256-bit. Kemudian, ia menyimpan kunci terenkripsi dengan data terenkripsi sehingga tersedia untuk mendekripsi data tabel sesuai permintaan.

### Melakukan cache pada kunci tabel

Untuk menghindari panggilan AWS KMS untuk setiap operasi Amazon Keyspaces, Amazon Keyspaces menyimpan tombol tabel plaintext untuk setiap koneksi dalam memori. Jika Amazon Keyspaces mendapatkan permintaan untuk kunci tabel cache setelah lima menit tidak aktif, ia akan mengirimkan permintaan baru AWS KMS untuk mendekripsi kunci tabel. Panggilan ini menangkap setiap perubahan yang dibuat pada kebijakan akses kunci KMS di AWS KMS atau AWS Identity and Access Management (IAM) sejak permintaan terakhir untuk mendekripsi kunci tabel.

### Enkripsi amplop

Jika Anda mengubah kunci terkelola pelanggan untuk tabel Anda, Amazon Keyspaces akan menghasilkan kunci tabel baru. Kemudian, ia menggunakan kunci tabel baru untuk mengenkripsi ulang kunci enkripsi data. Ini juga menggunakan kunci tabel baru untuk mengenkripsi kunci tabel sebelumnya yang digunakan untuk melindungi cadangan yang dapat dipulihkan. Proses ini disebut enkripsi amplop. Ini memastikan bahwa Anda dapat mengakses cadangan yang dapat dipulihkan bahkan jika Anda memutar kunci yang dikelola pelanggan. Untuk informasi selengkapnya tentang enkripsi amplop, lihat [Enkripsi Amplop](#) di Panduan AWS Key Management Service Pengembang.

### Topik

- [AWS kunci yang dimiliki](#)
- [Kunci yang dikelola pelanggan](#)
- [Enkripsi saat istirahat catatan penggunaan](#)



## AWS kunci yang dimiliki

Kunci milik AWS tidak disimpan di Akun AWS Anda. Mereka adalah bagian dari kumpulan kunci KMS yang AWS memiliki dan mengelola untuk digunakan dalam beberapa Akun AWS. Layanan AWS dapat menggunakan Kunci milik AWS untuk melindungi data Anda.

Anda tidak dapat melihat, mengelola, atau menggunakan Kunci milik AWS, atau mengaudit penggunaannya. Namun, Anda tidak perlu melakukan pekerjaan apa pun atau mengubah program apa pun untuk melindungi kunci yang mengenkripsi data Anda.

Anda tidak dikenakan biaya bulanan atau biaya penggunaan untuk penggunaan Kunci milik AWS, dan mereka tidak dihitung terhadap AWS KMS kuota untuk akun Anda.

## Kunci yang dikelola pelanggan

Kunci yang dikelola pelanggan adalah kunci Akun AWS yang Anda buat, miliki, dan kelola. Anda memiliki kontrol penuh atas kunci KMS ini.

Gunakan kunci yang dikelola pelanggan untuk mendapatkan fitur berikut:

- Anda membuat dan mengelola kunci yang dikelola pelanggan, termasuk menetapkan dan memelihara [kebijakan utama](#), [kebijakan IAM](#), dan [hibah](#) untuk mengontrol akses ke kunci yang dikelola pelanggan. Anda dapat [mengaktifkan dan menonaktifkan](#) kunci yang dikelola pelanggan, mengaktifkan dan menonaktifkan [rotasi kunci otomatis](#), dan [menjadwalkan kunci terkelola pelanggan](#) untuk dihapus saat tidak lagi digunakan. Anda dapat membuat tag dan alias untuk kunci terkelola pelanggan yang Anda kelola.
- Anda dapat menggunakan kunci yang dikelola pelanggan dengan [material kunci yang diimpor](#) atau kunci yang dikelola pelanggan di [penyimpanan kunci kustom](#) yang Anda miliki dan kelola.
- Anda dapat menggunakan AWS CloudTrail dan Amazon CloudWatch Logs untuk melacak permintaan yang dikirimkan Amazon Keyspaces AWS KMS atas nama Anda. Untuk informasi selengkapnya, lihat [the section called “Langkah 6: Konfigurasi pemantauan dengan AWS CloudTrail”](#).

Kunci yang dikelola pelanggan [dikenakan biaya](#) untuk setiap panggilan API, dan AWS KMS kuota berlaku untuk kunci KMS ini. Untuk informasi selengkapnya, lihat [AWS KMS sumber daya atau permintaan kuota](#).

Saat Anda menentukan kunci yang dikelola pelanggan sebagai kunci enkripsi root untuk tabel, cadangan yang dapat dipulihkan dienkripsi dengan kunci enkripsi yang sama yang ditentukan untuk

tabel pada saat cadangan dibuat. Jika kunci KMS untuk tabel diputar, pembungkus kunci memastikan bahwa kunci KMS terbaru memiliki akses ke semua cadangan yang dapat dipulihkan.

Amazon Keyspaces harus memiliki akses ke kunci yang dikelola pelanggan Anda untuk memberi Anda akses ke data tabel Anda. Jika status kunci enkripsi disetel ke dinonaktifkan atau dijadwalkan untuk dihapus, Amazon Keyspaces tidak dapat mengenkripsi atau mendekripsi data. Akibatnya, Anda tidak dapat melakukan operasi baca dan tulis di atas meja. Segera setelah layanan mendeteksi bahwa kunci enkripsi Anda tidak dapat diakses, Amazon Keyspaces mengirimkan pemberitahuan email untuk mengingatkan Anda.

Anda harus memulihkan akses ke kunci enkripsi Anda dalam waktu tujuh hari atau Amazon Keyspaces menghapus tabel Anda secara otomatis. Sebagai tindakan pencegahan, Amazon Keyspaces membuat cadangan data tabel yang dapat dipulihkan sebelum menghapus tabel. Amazon Keyspaces mempertahankan cadangan yang dapat dipulihkan selama 35 hari. Setelah 35 hari, Anda tidak dapat lagi memulihkan data tabel Anda. Anda tidak ditagih untuk cadangan yang dapat dipulihkan, tetapi biaya [pemulihan](#) standar berlaku.

Anda dapat menggunakan cadangan yang dapat dipulihkan ini untuk memulihkan data Anda ke tabel baru. Untuk memulai pemulihan, kunci terkelola pelanggan terakhir yang digunakan untuk tabel harus diaktifkan, dan Amazon Keyspaces harus memiliki akses ke sana.

#### Note

Saat Anda membuat tabel yang dienkripsi menggunakan kunci terkelola pelanggan yang tidak dapat diakses atau dijadwalkan untuk dihapus sebelum proses pembuatan selesai, terjadi kesalahan. Operasi buat tabel gagal, dan Anda akan dikirim pemberitahuan email.

## Enkripsi saat istirahat catatan penggunaan

Pertimbangkan hal berikut saat Anda menggunakan enkripsi saat istirahat di Amazon Keyspaces.

- Server-side enkripsi saat istirahat diaktifkan di semua tabel Amazon Keyspaces dan tidak dapat dinonaktifkan. Seluruh tabel dienkripsi saat istirahat, Anda tidak dapat memilih kolom atau baris tertentu untuk enkripsi.
- Secara default, Amazon Keyspaces menggunakan kunci default layanan tunggal (Kunci milik AWS) untuk mengenkripsi semua tabel Anda. Jika kunci ini tidak ada, itu dibuat untuk Anda. Kunci default layanan tidak dapat dinonaktifkan.

- Enkripsi saat istirahat hanya mengenkripsi data saat statis (saat istirahat) pada media penyimpanan persisten. Jika keamanan data menjadi perhatian data dalam perjalanan atau data yang digunakan, Anda harus mengambil langkah-langkah tambahan:
  - Data dalam perjalanan: Semua data Anda di Amazon Keyspaces dienkripsi saat transit. Secara default, komunikasi ke dan dari Amazon Keyspaces dilindungi dengan menggunakan enkripsi Secure Sockets Layer (SSL) /Transport Layer Security (TLS).
  - Data yang digunakan: Lindungi data Anda sebelum mengirimnya ke Amazon Keyspaces dengan menggunakan enkripsi sisi klien.
  - Kunci terkelola pelanggan: Data saat istirahat di tabel Anda selalu dienkripsi menggunakan kunci yang dikelola pelanggan Anda. Namun operasi yang melakukan pembaruan atom dari beberapa baris mengenkripsi data sementara digunakan Kunci milik AWS selama pemrosesan. Ini termasuk operasi penghapusan jangkauan dan operasi yang secara bersamaan mengakses data statis dan non-statis.
- Satu kunci yang dikelola pelanggan dapat memiliki hingga 50.000 [hibah](#). Setiap tabel Amazon Keyspaces yang terkait dengan kunci terkelola pelanggan menggunakan 2 hibah. Satu hibah dirilis saat tabel dihapus. Hibah kedua digunakan untuk membuat snapshot otomatis tabel untuk melindungi dari kehilangan data jika Amazon Keyspaces kehilangan akses ke kunci yang dikelola pelanggan secara tidak sengaja. Hibah ini dirilis 42 hari setelah penghapusan tabel.

## Enkripsi saat diam: Cara menggunakan kunci yang dikelola pelanggan untuk mengenkripsi tabel di Amazon Keyspaces

Anda dapat menggunakan pernyataan konsol atau CQL untuk menentukan tabel baru dan memperbarui kunci enkripsi tabel yang ada di Amazon Keyspaces. AWS KMS key Topik berikut menguraikan cara menerapkan kunci terkelola pelanggan untuk tabel baru dan yang sudah ada.

### Topik

- [Prasyarat: Buat kunci terkelola pelanggan menggunakan AWS KMS dan berikan izin ke Amazon Keyspaces](#)
- [Langkah 3: Tentukan kunci yang dikelola pelanggan untuk tabel baru](#)
- [Langkah 4: Perbarui kunci enkripsi tabel yang ada](#)
- [Langkah 5: Gunakan konteks enkripsi Amazon Keyspaces di log](#)
- [Langkah 6: Konfigurasi pemantauan dengan AWS CloudTrail](#)

Prasyarat: Buat kunci terkelola pelanggan menggunakan AWS KMS dan berikan izin ke Amazon Keyspaces

Sebelum Anda dapat melindungi tabel Amazon Keyspaces dengan kunci yang [dikelola pelanggan](#), [Anda harus terlebih dahulu membuat kunci](#) di AWS Key Management Service (AWS KMS) dan kemudian mengotorisasi Amazon Keyspaces untuk menggunakan kunci tersebut.

Langkah 1: Buat kunci yang dikelola pelanggan menggunakan AWS KMS

Untuk membuat kunci terkelola pelanggan yang akan digunakan untuk melindungi tabel Amazon Keyspaces, Anda dapat mengikuti langkah-langkah dalam [Membuat kunci KMS enkripsi simetris](#) menggunakan konsol atau API. AWS

Langkah 2: Otorisasi penggunaan kunci yang dikelola pelanggan Anda

Sebelum Anda dapat memilih [kunci yang dikelola pelanggan](#) untuk melindungi tabel Amazon Keyspaces, kebijakan pada kunci yang dikelola pelanggan tersebut harus memberikan izin kepada Amazon Keyspaces untuk menggunakannya atas nama Anda. Anda memiliki kendali penuh atas kebijakan dan hibah pada kunci yang dikelola pelanggan. Anda dapat memberikan izin ini dalam [kebijakan kunci](#), [kebijakan IAM](#), atau [pemberian izin](#).

Amazon Keyspaces tidak memerlukan otorisasi tambahan untuk menggunakan default [Kunci milik AWS](#) untuk melindungi tabel Amazon Keyspaces di akun Anda. AWS

Topik berikut menunjukkan cara mengonfigurasi izin yang diperlukan menggunakan kebijakan dan hibah IAM yang memungkinkan tabel Amazon Keyspaces menggunakan kunci yang dikelola pelanggan.

Topik

- [Kebijakan utama untuk kunci yang dikelola pelanggan](#)
- [Contoh kebijakan kunci](#)
- [Menggunakan hibah untuk mengotorisasi Amazon Keyspaces](#)

Kebijakan utama untuk kunci yang dikelola pelanggan

Saat Anda memilih [kunci yang dikelola pelanggan](#) untuk melindungi tabel Amazon Keyspaces, Amazon Keyspaces mendapatkan izin untuk menggunakan kunci yang dikelola pelanggan atas nama prinsipal yang membuat pilihan. Prinsipal tersebut, pengguna atau peran, harus memiliki izin pada kunci terkelola pelanggan yang diperlukan Amazon Keyspaces.

Minimal, Amazon Keyspaces memerlukan izin berikut pada kunci yang dikelola pelanggan:

- [kms:Encrypt](#)
- [kms:Decrypt](#)
- [kms: ReEncrypt](#) \* (untuk kms: ReEncryptFrom dan kms:) ReEncryptTo
- kms: GenerateDataKey \* (untuk [kms: GenerateDataKey](#) dan kms:) GenerateDataKeyWithoutPlaintext
- [km: DescribeKey](#)
- [km: CreateGrant](#)

### Contoh kebijakan kunci

Sebagai contoh, kebijakan kunci berikut hanya menyediakan izin yang diperlukan. Kebijakan ini memiliki efek sebagai berikut:

- Memungkinkan Amazon Keyspaces menggunakan kunci terkelola pelanggan dalam operasi kriptografi dan membuat hibah—tetapi hanya jika itu bertindak atas nama kepala sekolah di akun yang memiliki izin untuk menggunakan Amazon Keyspaces. Jika prinsipal yang ditentukan dalam pernyataan kebijakan tidak memiliki izin untuk menggunakan Amazon Keyspaces, panggilan gagal, meskipun berasal dari layanan Amazon Keyspaces.
- Kunci ViaService kondisi [kms:](#) mengizinkan izin hanya jika permintaan berasal dari Amazon Keyspaces atas nama prinsipal yang tercantum dalam pernyataan kebijakan. Pengguna utama ini tidak dapat memanggil operasi ini secara langsung. Perhatikan bahwa nilai kms:ViaService, `cassandra.*.amazonaws.com`, memiliki tanda bintang (\*) di posisi Wilayah. Amazon Keyspaces memerlukan izin untuk independen dari yang tertentu. Wilayah AWS
- Memberikan administrator kunci terkelola pelanggan (pengguna yang dapat mengambil db-team peran) akses hanya-baca ke kunci terkelola pelanggan dan izin untuk mencabut hibah, termasuk hibah yang diperlukan [Amazon Keyspaces](#) untuk melindungi tabel.
- Memberikan akses hanya-baca Amazon Keyspaces ke kunci yang dikelola pelanggan. Dalam hal ini, Amazon Keyspaces dapat memanggil operasi ini secara langsung. Itu tidak harus bertindak atas nama prinsipal akun.

Sebelum menggunakan kebijakan kunci contoh, ganti prinsip contoh dengan prinsip aktual dari Anda. Akun 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": "*"
 }
]
 }

```

## Menggunakan hibah untuk mengotorisasi Amazon Keyspaces

Selain kebijakan utama, Amazon Keyspaces menggunakan hibah untuk menetapkan izin pada kunci yang dikelola pelanggan. Untuk melihat hibah pada kunci yang dikelola pelanggan di akun Anda, gunakan [ListGrants](#) operasi. Amazon Keyspaces tidak memerlukan hibah, atau izin tambahan apa pun, untuk menggunakan file untuk melindungi tabel [Kunci milik AWS](#) Anda.

Amazon Keyspaces menggunakan izin hibah saat melakukan pemeliharaan sistem latar belakang dan tugas perlindungan data berkelanjutan. Layanan ini juga menggunakan pemberian izin untuk menghasilkan kunci tabel.

Setiap pemberian izin berlaku spesifik pada sebuah tabel. Jika akun menyertakan beberapa tabel yang dienkripsi di bawah kunci terkelola pelanggan yang sama, ada hibah untuk setiap jenis untuk setiap tabel. Hibah dibatasi oleh [konteks enkripsi Amazon Keyspaces](#), yang mencakup nama tabel dan ID. Akun AWS Hibah termasuk izin untuk [pensium hibah](#) jika tidak lagi diperlukan.

Untuk membuat hibah, Amazon Keyspaces harus memiliki izin untuk `CreateGrant` memanggil atas nama pengguna yang membuat tabel terenkripsi.

Kebijakan utama juga dapat memungkinkan akun untuk [mencabut hibah pada](#) kunci yang dikelola pelanggan. Namun, jika Anda mencabut hibah pada tabel terenkripsi aktif, Amazon Keyspaces tidak akan dapat melindungi dan memelihara tabel.

### Langkah 3: Tentukan kunci yang dikelola pelanggan untuk tabel baru

Ikuti langkah-langkah berikut untuk menentukan kunci yang dikelola pelanggan pada tabel baru menggunakan konsol Amazon Keyspaces atau CQL.

Membuat tabel terenkripsi menggunakan kunci yang dikelola pelanggan (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel, lalu pilih Buat tabel.
3. Pada halaman Buat tabel di bagian Rincian tabel, pilih ruang kunci dan berikan nama untuk tabel baru.
4. Di bagian Skema, buat skema untuk tabel Anda.
5. Di bagian Pengaturan tabel, pilih Sesuaikan pengaturan.
6. Lanjutkan ke Pengaturan enkripsi.

Pada langkah ini, Anda memilih pengaturan enkripsi untuk tabel.

Di bagian Enkripsi saat istirahat di bawah Pilih AWS KMS key, pilih opsi Pilih kunci KMS yang berbeda (lanjutan), dan di bidang pencarian, pilih AWS KMS key atau masukkan Nama Sumber Daya Amazon (ARN).

#### Note

Jika kunci yang Anda pilih tidak dapat diakses atau tidak memiliki izin yang diperlukan, lihat [Memecahkan masalah akses kunci](#) di Panduan Pengembang AWS Key Management Service .

### 7. Pilih Buat untuk membuat tabel yang dienkripsi.

Buat tabel baru menggunakan kunci terkelola pelanggan untuk enkripsi saat istirahat (CQL)

Untuk membuat tabel baru yang menggunakan kunci terkelola pelanggan untuk enkripsi saat istirahat, Anda dapat menggunakan CREATE TABLE pernyataan seperti yang ditunjukkan pada contoh berikut. Pastikan untuk mengganti kunci ARN dengan ARN untuk kunci yang valid dengan izin yang diberikan ke 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'
 }
};
```

Jika Anda menerima `Invalid Request Exception`, Anda perlu mengonfirmasi bahwa kunci yang dikelola pelanggan valid dan Amazon Keyspaces memiliki izin yang diperlukan. Untuk mengonfirmasi bahwa kunci telah dikonfigurasi dengan benar, lihat [Akses kunci pemecahan masalah di Panduan AWS Key Management Service Pengembang](#).

Langkah 4: Perbarui kunci enkripsi tabel yang ada

Anda juga dapat menggunakan konsol Amazon Keyspaces atau CQL untuk mengubah kunci enkripsi tabel yang ada antara kunci KMS yang dikelola pelanggan Kunci milik AWS dan kapan saja.



Perbarui tabel yang ada dengan kunci terkelola pelanggan baru (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Di panel navigasi, pilih Tabel.
3. Pilih tabel yang ingin Anda perbarui, lalu pilih tab Pengaturan tambahan.
4. Di bagian Enkripsi saat istirahat, pilih Kelola Enkripsi untuk mengedit pengaturan enkripsi untuk tabel.

Di bawah Pilih AWS KMS key, pilih opsi Pilih tombol KMS yang berbeda (lanjutan), dan di bidang pencarian, pilih AWS KMS key atau masukkan Nama Sumber Daya Amazon (ARN).

 Note

Jika kunci yang Anda pilih tidak valid, lihat [Akses kunci pemecahan masalah di Panduan AWS Key Management Service Pengembang](#).

Atau, Anda dapat memilih Kunci milik AWS untuk tabel yang dienkripsi dengan kunci yang dikelola pelanggan.

5. Pilih Simpan perubahan untuk menyimpan perubahan Anda ke tabel.

Memperbarui kunci enkripsi yang digunakan untuk tabel yang ada

Untuk mengubah kunci enkripsi tabel yang ada, Anda menggunakan ALTER TABLE pernyataan untuk menentukan kunci terkelola pelanggan untuk enkripsi saat istirahat. Pastikan untuk mengganti kunci ARN dengan ARN untuk kunci yang valid dengan izin yang diberikan ke 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'
 }
};
```

Jika Anda menerima `Invalid Request Exception`, Anda perlu mengonfirmasi bahwa kunci yang dikelola pelanggan valid dan Amazon Keyspaces memiliki izin yang diperlukan. Untuk mengonfirmasi

bahwa kunci telah dikonfigurasi dengan benar, lihat [Akses kunci pemecahan masalah di Panduan AWS Key Management Service Pengembang](#).

Untuk mengubah kunci enkripsi kembali ke opsi enkripsi default saat istirahat dengan Kunci milik AWS, Anda dapat menggunakan ALTER TABLE pernyataan seperti yang ditunjukkan pada contoh berikut.

```
ALTER TABLE my_keyspace.my_table WITH CUSTOM_PROPERTIES = {
 'encryption_specification': {
 'encryption_type' : 'AWS_OWNED_KMS_KEY'
 }
};
```

Langkah 5: Gunakan konteks enkripsi Amazon Keyspaces di log

[Konteks enkripsi](#) adalah seperangkat pasangan kunci-nilai yang berisi data non-rahasia yang arbitrer. Ketika Anda menyertakan konteks enkripsi dalam permintaan untuk mengenkripsi data, secara AWS KMS kriptografis mengikat konteks enkripsi ke data terenkripsi. Untuk mendekripsi data, Anda harus meneruskan konteks enkripsi yang sama.

Amazon Keyspaces menggunakan konteks enkripsi yang sama di semua operasi AWS KMS kriptografi. Jika Anda menggunakan [kunci terkelola pelanggan](#) untuk melindungi tabel Amazon Keyspaces, Anda dapat menggunakan konteks enkripsi untuk mengidentifikasi penggunaan kunci terkelola pelanggan dalam catatan audit dan log. Itu juga muncul dalam teks biasa di log, seperti di log untuk dan [AWS CloudTrail](#)[Amazon CloudWatch](#) Logs.

Dalam permintaannya AWS KMS, Amazon Keyspaces menggunakan konteks enkripsi dengan tiga pasangan kunci-nilai.

```
"encryptionContextSubset": {
 "aws:cassandra:keyspaceName": "my_keyspace",
 "aws:cassandra:tableName": "mytable"
 "aws:cassandra:subscriberId": "111122223333"
}
```

- Keyspace - Pasangan kunci-nilai pertama mengidentifikasi ruang kunci yang menyertakan tabel yang dienkripsi Amazon Keyspaces. Kuncinya adalah `aws:cassandra:keyspaceName`. Nilai adalah nama keyspace.

```
"aws:cassandra:keyspaceName": "<keyspace-name>"
```

Contoh:

```
"aws:cassandra:keyspaceName": "my_keyspace"
```

- Tabel — Pasangan kunci-nilai kedua mengidentifikasi tabel yang dienkripsi Amazon Keyspaces. Kuncinya adalah `aws:cassandra:tableName`. Nilainya adalah nama tabel.

```
"aws:cassandra:tableName": "<table-name>"
```

Contoh:

```
"aws:cassandra:tableName": "my_table"
```

- Akun — Pasangan kunci-nilai ketiga mengidentifikasi pasangan. Akun AWS Kuncinya adalah `aws:cassandra:subscriberId`. Nilainya adalah ID akun.

```
"aws:cassandra:subscriberId": "<account-id>"
```

Contoh:

```
"aws:cassandra:subscriberId": "111122223333"
```

## Langkah 6: Konfigurasi pemantauan dengan AWS CloudTrail

Jika Anda menggunakan [kunci yang dikelola pelanggan](#) untuk melindungi tabel Amazon Keyspaces, Anda dapat menggunakan AWS CloudTrail log untuk melacak permintaan yang dikirimkan Amazon Keyspaces atas nama Anda. AWS KMS

Permintaan `GenerateDataKey` `DescribeKeyDecrypt`, dan `CreateGrant` permintaan dibahas di bagian ini. Selain itu, Amazon Keyspaces menggunakan [RetireGrant](#) operasi untuk menghapus hibah saat Anda menghapus tabel.

### Note

Saat Anda bekerja dengan Amazon Keyspaces, beberapa operasi dapat menghasilkan CloudTrail peristiwa dengan `invokedBy` bidang `dynamodb.amazonaws.com` Ini

diharapkan dan terjadi karena Amazon Keyspaces terintegrasi dengan Amazon DynamoDB untuk menyediakan layanannya.

## GenerateDataKey

Amazon Keyspaces membuat kunci tabel unik untuk mengenkripsi data saat istirahat. Ini mengirimkan [GenerateDataKey](#) permintaan untuk AWS KMS yang menentukan kunci KMS untuk tabel.

Peristiwa yang mencatat operasi GenerateDataKey serupa dengan peristiwa contoh berikut. Pengguna adalah akun layanan Amazon Keyspaces. Parameter tersebut mencakup Nama Sumber Daya Amazon (ARN) dari kunci yang dikelola pelanggan, penentu kunci yang memerlukan kunci 256-bit, dan [konteks enkripsi](#) yang mengidentifikasi ruang kunci, tabel, dan file. Akun 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 menggunakan [DescribeKey](#) operasi untuk menentukan apakah kunci KMS yang Anda pilih ada di akun dan Wilayah.

Peristiwa yang mencatat operasi `DescribeKey` serupa dengan peristiwa contoh berikut. Pengguna adalah akun layanan Amazon Keyspaces. Parameter termasuk ARN dari kunci yang dikelola pelanggan dan penentu kunci yang memerlukan kunci 256-bit.

```

{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "IAMUser",
 "principalId": "AIDAZ3FNIIIVIZZ6H7CFQG",
 "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"
 }
}

```

## Dekripsi

Saat Anda mengakses tabel Amazon Keyspaces, Amazon Keyspaces perlu mendekripsi kunci tabel sehingga dapat mendekripsi kunci di bawahnya dalam hierarki. Layanan ini kemudian mendekripsi data dalam tabel. Untuk mendekripsi kunci tabel, Amazon Keyspaces mengirimkan permintaan [Dekripsi](#) ke AWS KMS yang menentukan kunci KMS untuk tabel.

Peristiwa yang mencatat operasi Decrypt serupa dengan peristiwa contoh berikut. Pengguna adalah kepala sekolah Anda Akun AWS yang mengakses tabel. Parameter termasuk kunci tabel terenkripsi (sebagai gumpalan ciphertext) dan [konteks enkripsi](#) yang mengidentifikasi tabel dan file. Akun AWS AWS KMS memperoleh ID kunci yang dikelola pelanggan dari ciphertext.

```

{
 "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

Saat Anda menggunakan [kunci yang dikelola pelanggan](#) untuk melindungi tabel Amazon Keyspaces Anda, Amazon Keyspaces menggunakan [hibah](#) untuk memungkinkan layanan melakukan tugas perlindungan dan pemeliharaan serta daya tahan data secara berkelanjutan. Hibah ini tidak diperlukan. [Kunci milik AWS](#)

Hibah yang dibuat Amazon Keyspaces khusus untuk tabel. Prinsipal dalam [CreateGrant](#) permintaan adalah pengguna yang membuat tabel.

Peristiwa yang mencatat operasi CreateGrant serupa dengan peristiwa contoh berikut. Parameter termasuk ARN kunci yang dikelola pelanggan untuk tabel, pokok penerima hibah dan kepala pensiun (layanan Amazon Keyspaces), dan operasi yang dicakup oleh hibah. [Ini juga mencakup kendala yang mengharuskan semua operasi enkripsi menggunakan konteks enkripsi yang ditentukan.](#)

```
{
 "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"
}

```

## Enkripsi dalam perjalanan di Amazon Keyspaces

Amazon Keyspaces hanya menerima koneksi aman menggunakan Transport Layer Security (TLS). Enkripsi dalam perjalanan menyediakan lapisan perlindungan data tambahan dengan mengenkripsi data Anda saat melakukan perjalanan ke dan dari Amazon Keyspaces. Kebijakan organisasi, peraturan industri atau pemerintah, dan persyaratan kepatuhan sering kali memerlukan

penggunaan enkripsi dalam perjalanan untuk meningkatkan keamanan data aplikasi Anda ketika mereka mengirimkan data melalui jaringan.

Untuk mempelajari cara mengenkripsi `cqlsh` koneksi ke Amazon Keyspaces menggunakan TLS, lihat [the section called “Cara mengkonfigurasi `cqlsh` koneksi secara manual untuk TLS”](#). Untuk mempelajari cara menggunakan enkripsi TLS dengan driver klien, lihat [the section called “Menggunakan driver klien Cassandra”](#).

## Privasi lalu lintas internetwork di Amazon Keyspaces

Topik ini menjelaskan cara Amazon Keyspaces (untuk Apache Cassandra) mengamankan koneksi dari aplikasi lokal ke Amazon Keyspaces dan antara Amazon Keyspaces dan sumber daya lain dalam hal yang sama. AWS Wilayah AWS

### Lalu lintas antara layanan dan aplikasi serta klien on-premise

Anda memiliki dua opsi konektivitas antara jaringan pribadi Anda dan AWS:

- AWS Site-to-Site VPN Koneksi. Untuk informasi selengkapnya, lihat [Apa itu AWS Site-to-Site VPN?](#) dalam Panduan Pengguna AWS Site-to-Site VPN .
- Direct Connect Koneksi. Untuk informasi selengkapnya, lihat [Apa itu Direct Connect?](#) dalam Panduan Pengguna Direct Connect .

Sebagai layanan terkelola, Amazon Keyspaces (untuk Apache Cassandra) dilindungi oleh keamanan jaringan global. AWS Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat [Keamanan AWS Cloud](#). Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur dalam Kerangka Kerja](#) yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon Keyspaces melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Cipher suite dengan perfect forward secrecy (PFS) seperti DHE (Ephemeral) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Amazon Keyspaces mendukung dua metode otentikasi permintaan klien. Metode pertama menggunakan kredensi khusus layanan, yang merupakan kredensial berbasis kata sandi yang dihasilkan untuk pengguna IAM tertentu. Anda dapat membuat dan mengelola kata sandi menggunakan konsol IAM, the AWS CLI, atau AWS API. Untuk informasi selengkapnya, lihat [Menggunakan IAM dengan Amazon Keyspaces](#).

Metode kedua menggunakan plugin otentikasi untuk Driver DataStax Java open-source untuk Cassandra. [Plugin ini memungkinkan pengguna IAM, peran, dan identitas federasi untuk menambahkan informasi otentikasi ke permintaan API Amazon Keyspaces \(untuk Apache Cassandra\) menggunakan proses Signature Version 4 \(SigV4\).AWS](#) Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Lalu lintas antar AWS sumber daya di Wilayah yang sama

Endpoint VPC antarmuka memungkinkan komunikasi pribadi antara virtual private cloud (VPC) Anda yang berjalan di Amazon VPC dan Amazon Keyspaces. Endpoint VPC antarmuka didukung oleh AWS PrivateLink, yang merupakan AWS layanan yang memungkinkan komunikasi pribadi antara VPC dan layanan. AWS PrivateLink memungkinkan ini dengan menggunakan elastic network interface dengan IP pribadi di VPC Anda sehingga lalu lintas jaringan tidak meninggalkan jaringan Amazon. Endpoint VPC antarmuka tidak memerlukan gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct Connect. Untuk informasi selengkapnya, lihat [titik akhir Amazon Virtual Private Cloud dan Interface VPC](#) ().AWS PrivateLink Untuk kebijakan-kebijakan contoh, lihat [the section called “Menggunakan titik akhir VPC antarmuka untuk Amazon Keyspaces”](#).

## AWS Identity and Access Management untuk Amazon Keyspaces

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya Amazon Keyspaces. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Cara Amazon Keyspaces bekerja dengan IAM](#)
- [Contoh kebijakan berbasis identitas Amazon Keyspaces](#)

- [AWS kebijakan terkelola untuk Amazon Keyspaces](#)
- [Memecahkan masalah identitas dan akses Amazon Keyspaces](#)
- [Menggunakan peran terkait layanan untuk Amazon Keyspaces](#)

## Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda berdasarkan peran Anda:

- Pengguna layanan - minta izin dari administrator Anda jika Anda tidak dapat mengakses fitur (lihat [Memecahkan masalah identitas dan akses Amazon Keyspaces](#))
- Administrator layanan - tentukan akses pengguna dan mengirimkan permintaan izin (lihat [Cara Amazon Keyspaces bekerja dengan IAM](#))
- Administrator IAM - tulis kebijakan untuk mengelola akses (lihat [Contoh kebijakan berbasis identitas Amazon Keyspaces](#))

## Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensial identitas Anda. Anda harus diautentikasi sebagai Pengguna root akun AWS, pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk sebagai identitas federasi menggunakan kredensial dari sumber identitas seperti AWS IAM Identity Center (Pusat Identitas IAM), autentikasi masuk tunggal, atau kredensial. Google/Facebook Untuk informasi selengkapnya tentang cara masuk, lihat [Cara masuk ke Akun AWS Anda](#) dalam Panduan Pengguna AWS Sign-In .

Untuk akses terprogram, AWS sediakan SDK dan CLI untuk menandatangani permintaan secara kriptografis. Untuk informasi selengkapnya, lihat [AWS Signature Version 4 untuk permintaan API](#) dalam Panduan Pengguna IAM.

## Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang disebut pengguna Akun AWS root yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Untuk

tugas yang memerlukan kredensial pengguna root, lihat [Tugas yang memerlukan kredensial pengguna root](#) dalam Panduan Pengguna IAM.

## Pengguna dan grup IAM

[Pengguna IAM](#) adalah identitas dengan izin khusus untuk satu orang atau aplikasi. Sebaiknya gunakan kredensial sementara alih-alih pengguna IAM dengan kredensial jangka panjang. Untuk informasi selengkapnya, lihat [Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara](#) di Panduan Pengguna IAM.

[Grup IAM](#) menentukan kumpulan pengguna IAM dan mempermudah pengelolaan izin untuk pengguna dalam jumlah besar. Untuk mempelajari selengkapnya, lihat [Kasus penggunaan untuk pengguna IAM](#) dalam Panduan Pengguna IAM.

## Peran IAM

[Peran IAM](#) adalah identitas dengan izin khusus yang menyediakan kredensial sementara. Anda dapat mengambil peran dengan [beralih dari pengguna ke peran IAM \(konsol\)](#) atau dengan memanggil operasi AWS CLI atau AWS API. Untuk informasi selengkapnya, lihat [Metode untuk mengambil peran](#) dalam Panduan Pengguna IAM.

Peran IAM berguna untuk akses pengguna terfederasi, izin pengguna IAM sementara, akses lintas akun, akses lintas layanan, dan aplikasi yang berjalan di Amazon EC2. Untuk informasi selengkapnya, lihat [Akses sumber daya lintas akun di IAM](#) dalam Panduan Pengguna IAM.

## Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan menentukan izin saat dikaitkan dengan identitas atau sumber daya. AWS mengevaluasi kebijakan ini ketika kepala sekolah membuat permintaan. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang dokumen kebijakan JSON, lihat [Gambaran umum kebijakan JSON](#) dalam Panduan Pengguna IAM.

Menggunakan kebijakan, administrator menentukan siapa yang memiliki akses ke apa dengan mendefinisikan principal mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Administrator IAM membuat kebijakan IAM dan menambahkannya ke peran, yang kemudian dapat diambil oleh pengguna. Kebijakan IAM mendefinisikan izin terlepas dari metode yang Anda gunakan untuk melakukannya.

## Identity-based kebijakan

Identity-based kebijakan adalah dokumen kebijakan izin JSON yang Anda lampirkan ke identitas (pengguna, grup, atau peran). Kebijakan ini mengontrol tindakan apa yang bisa dilakukan oleh identitas tersebut, terhadap sumber daya yang mana, dan dalam kondisi apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Tentukan izin IAM kustom dengan kebijakan yang dikelola pelanggan](#) dalam Panduan Pengguna IAM.

Identity-based kebijakan dapat berupa kebijakan inline (disematkan langsung ke dalam satu identitas) atau kebijakan terkelola (kebijakan mandiri yang dilampirkan pada beberapa identitas). Untuk mempelajari cara memilih antara kebijakan terkelola dan kebijakan inline, lihat [Pilih antara kebijakan terkelola dan kebijakan inline](#) dalam Panduan Pengguna IAM.

## Resource-based kebijakan

Resource-based kebijakan adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contohnya termasuk kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Anda harus [menentukan principal](#) dalam kebijakan berbasis sumber daya.

Resource-based kebijakan adalah kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

## Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan principal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat [Gambaran umum daftar kontrol akses \(ACL\)](#) dalam Panduan Developer Amazon Simple Storage Service.

## Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang dapat menetapkan izin maksimum yang diberikan oleh jenis kebijakan yang lebih umum:

- Batasan izin – Menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM. Untuk informasi selengkapnya, lihat [Batasan izin untuk entitas IAM](#) dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) – Menentukan izin maksimum untuk organisasi atau unit organisasi di AWS Organizations. Untuk informasi selengkapnya, lihat [Kebijakan kontrol layanan](#) dalam Panduan Pengguna AWS Organizations .
- Kebijakan kontrol sumber daya (RCP) – Menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda. Untuk informasi selengkapnya, lihat [Kebijakan kontrol sumber daya \(RCP\)](#) dalam Panduan Pengguna AWS Organizations .
- Kebijakan sesi – Kebijakan lanjutan yang diteruskan sebagai parameter saat membuat sesi sementara untuk peran atau pengguna terfederasi. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dalam Panduan Pengguna IAM.

## Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat [Logika evaluasi kebijakan](#) di Panduan Pengguna IAM.

## Cara Amazon Keyspaces bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon Keyspaces, Anda harus memahami fitur IAM apa yang tersedia untuk digunakan dengan Amazon Keyspaces. Untuk mendapatkan tampilan tingkat tinggi tentang cara Amazon Keyspaces dan layanan AWS lainnya bekerja dengan IAM, [AWS lihat layanan yang bekerja dengan IAM di Panduan Pengguna](#) IAM.

### Topik

- [Kebijakan berbasis identitas Amazon Keyspaces](#)
- [Kebijakan berbasis sumber daya Amazon Keyspaces](#)
- [Otorisasi berdasarkan tag Amazon Keyspaces](#)
- [Peran IAM Amazon Keyspaces](#)

## Kebijakan berbasis identitas Amazon Keyspaces

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Amazon Keyspaces mendukung tindakan dan sumber daya tertentu, serta kunci kondisi. Untuk mempelajari semua elemen yang Anda gunakan dalam kebijakan JSON, lihat [Referensi elemen kebijakan IAM JSON](#) dalam Panduan Pengguna IAM.

Untuk melihat sumber daya dan tindakan khusus layanan Amazon Keyspaces, serta kunci konteks kondisi yang dapat digunakan untuk kebijakan izin IAM, lihat kunci [Tindakan, sumber daya, dan kondisi untuk Amazon Keyspaces \(untuk Apache Cassandra\)](#) di Referensi Otorisasi Layanan.

### Tindakan

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Tindakan kebijakan di Amazon Keyspaces menggunakan awalan berikut sebelum tindakan: `cassandra`: Misalnya, untuk memberikan izin kepada seseorang untuk membuat ruang kunci Amazon Keyspaces dengan pernyataan `CREATE CQL` Amazon Keyspaces, Anda menyertakan tindakan tersebut dalam kebijakan mereka. `cassandra:Create` Pernyataan kebijakan harus memuat elemen `Action` atau `NotAction`. Amazon Keyspaces mendefinisikan serangkaian tindakannya sendiri yang menjelaskan tugas yang dapat Anda lakukan dengan layanan ini.

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan tindakan-tindakan tersebut menggunakan koma seperti berikut:

```
"Action": [
 "cassandra:CREATE",
 "cassandra:MODIFY"
]
```

Untuk melihat daftar tindakan Amazon Keyspaces, lihat [Tindakan yang Ditentukan oleh Amazon Keyspaces \(untuk Apache Cassandra\)](#) di Referensi Otorisasi Layanan.



## Sumber daya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Praktik terbaiknya, tentukan sumber daya menggunakan [Amazon Resource Name \(ARN\)](#). Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, gunakan wildcard (\*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Di Amazon Keyspaces, keyspaces, tabel, dan stream dapat digunakan dalam `Resource` elemen izin IAM.

### Note

Untuk mengakses ruang kunci dan tabel pengguna di Amazon Keyspaces, kebijakan IAM Anda harus `cassandra:Select` menyertakan izin pada tabel sistem:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/system*
```

Ini berlaku untuk skenario berikut:

- AWS Akses Konsol Manajemen
- Operasi sumber daya SDK, misalnya `GetKeyspace`, `GetTableListKeyspaces`, dan `ListTables`
- Koneksi driver klien Apache Cassandra standar, karena driver secara otomatis membaca tabel sistem selama inisialisasi koneksi

Tabel sistem hanya-baca dan tidak dapat dimodifikasi.

Sumber daya keyspace Amazon Keyspaces memiliki ARN berikut:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/
```

Sumber daya tabel Amazon Keyspaces memiliki ARN berikut:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}
```

Sumber daya aliran Amazon Keyspaces memiliki ARN berikut:

```
arn:${Partition}:cassandra:${Region}:${Account}:/keyspace/${keyspaceName}/table/
${tableName}/stream/${streamLabel}
```

Untuk informasi selengkapnya tentang format ARN, lihat [Amazon Resource Names \(ARN\) dan ruang nama AWS layanan](#).

Misalnya, untuk menentukan mykeyspace ruang kunci dalam pernyataan Anda, gunakan ARN berikut:

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/"
```

Untuk menentukan semua ruang kunci milik akun tertentu, gunakan wildcard (\*):

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/*"
```

Beberapa tindakan Amazon Keyspaces, seperti untuk membuat sumber daya, tidak dapat dilakukan pada sumber daya tertentu. Dalam kasus tersebut, Anda harus menggunakan wildcard (\*).

```
"Resource": ""
```

Misalnya, untuk memberikan SELECT izin kepada prinsipal IAM untuk mytable inmykeyspace, kepala sekolah harus memiliki izin untuk membaca keduanya, dan. mytable keyspace/system\* Untuk menentukan beberapa sumber daya dalam satu pernyataan, pisahkan ARN dengan koma.

```
"Resource": "arn:aws:cassandra:us-east-1:111122223333:/keyspace/mykeyspace/table/
mytable",
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system"
```

Untuk melihat daftar jenis sumber daya Amazon Keyspaces dan ARNnya, lihat Sumber Daya yang Ditentukan [oleh Amazon Keyspaces \(untuk Apache Cassandra\)](#) di Referensi Otorisasi Layanan.

Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat [Tindakan yang Ditentukan oleh Amazon Keyspaces \(untuk Apache Cassandra\)](#).

## Kunci syarat

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

Elemen `Condition` menentukan ketika pernyataan dieksekusi berdasarkan kriteria yang ditetapkan. Anda dapat membuat ekspresi bersyarat yang menggunakan [operator kondisi](#), misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta. Untuk melihat semua kunci kondisi AWS global, lihat [kunci konteks kondisi AWS global](#) di Panduan Pengguna IAM.

Amazon Keyspaces mendefinisikan kumpulan kunci kondisinya sendiri dan juga mendukung penggunaan beberapa kunci kondisi global. Untuk melihat semua kunci kondisi AWS global, lihat [kunci konteks kondisi AWS global](#) di Panduan Pengguna IAM.

Semua tindakan Amazon Keyspaces mendukung `aws:RequestTag/${TagKey}`, `tombolaws:ResourceTag/${TagKey}`, dan `aws:TagKeys` kondisi. Untuk informasi selengkapnya, lihat [the section called “Akses sumber daya Amazon Keyspaces berdasarkan tag”](#).

Untuk melihat daftar kunci kondisi Amazon Keyspaces, lihat Kunci Kondisi untuk [Amazon Keyspaces \(untuk Apache Cassandra\)](#) di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat [Tindakan yang Ditentukan oleh Amazon Keyspaces \(untuk Apache Cassandra\)](#).

## Contoh

Untuk melihat contoh kebijakan berbasis identitas Amazon Keyspaces, lihat. [Contoh kebijakan berbasis identitas Amazon Keyspaces](#)

## Kebijakan berbasis sumber daya Amazon Keyspaces

Amazon Keyspaces tidak mendukung kebijakan berbasis sumber daya. Untuk melihat contoh halaman detail kebijakan berbasis sumber daya, lihat <https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html>.

## Otorisasi berdasarkan tag Amazon Keyspaces

Anda dapat mengelola akses ke sumber daya Amazon Keyspaces dengan menggunakan tag. Untuk mengelola akses sumber daya berdasarkan tag, Anda memberikan informasi tag dalam [elemen kondisi](#) kebijakan menggunakan `kuncicassandra:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys` kondisi. Untuk informasi selengkapnya tentang menandai resource Amazon Keyspaces, lihat [the section called “Bekerja dengan tag”](#)

Untuk melihat contoh kebijakan-kebijakan berbasis identitas untuk membatasi akses ke sumber daya berdasarkan tanda pada sumber daya tersebut, lihat [Akses sumber daya Amazon Keyspaces berdasarkan tag](#).

## Peran IAM Amazon Keyspaces

[Peran IAM](#) adalah entitas di dalam Anda Akun AWS yang memiliki izin khusus.

### Menggunakan kredensyal sementara dengan Amazon Keyspaces

Anda dapat menggunakan kredensial sementara untuk masuk dengan federasi, untuk memainkan peran IAM, atau untuk mengambil peran lintas akun. Anda memperoleh kredensial keamanan sementara dengan memanggil operasi AWS STS API seperti [AssumeRole](#) atau [GetFederationToken](#)

Amazon Keyspaces mendukung penggunaan kredensial sementara dengan plugin otentikasi AWS Signature Version 4 (SigV4) yang tersedia dari repo Github untuk bahasa berikut:

- Jawa: <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>.
- Pergi: <https://github.com/aws/aws-sigv4-auth-cassandra-gocql-driver-plugin>.

Untuk contoh dan tutorial yang menerapkan plugin otentikasi untuk mengakses Amazon Keyspaces secara terprogram, lihat [the section called “Menggunakan driver klien Cassandra”](#)

### Service-linked peran

[Service-linked peran](#) memungkinkan AWS layanan mengakses sumber daya di layanan lain untuk menyelesaikan tindakan atas nama Anda. Service-linked peran muncul di akun IAM Anda dan dimiliki oleh layanan. Administrator IAM dapat melihat tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang membuat atau mengelola peran terkait layanan Amazon Keyspaces, lihat [the section called “Menggunakan Peran Terkait Layanan”](#)

## Peran layanan

Amazon Keyspaces tidak mendukung peran layanan.

## Contoh kebijakan berbasis identitas Amazon Keyspaces

Secara default, pengguna dan peran IAM tidak memiliki izin untuk membuat atau memodifikasi sumber daya Amazon Keyspaces. Mereka juga tidak dapat melakukan tugas menggunakan konsol, CQLSH AWS CLI, atau API. AWS Administrator IAM harus membuat kebijakan IAM yang memberikan izin kepada pengguna dan peran untuk melakukan operasi API tertentu pada sumber daya yang diperlukan. Administrator kemudian harus melampirkan kebijakan tersebut ke pengguna IAM atau grup yang memerlukan izin tersebut.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat [Membuat kebijakan di tab JSON](#) dalam Panduan Pengguna IAM.

### Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan konsol Amazon Keyspaces](#)
- [Mengizinkan pengguna melihat izin mereka sendiri](#)
- [Mengakses tabel Amazon Keyspaces](#)
- [Akses sumber daya Amazon Keyspaces berdasarkan tag](#)

## Praktik terbaik kebijakan

Identity-based kebijakan menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya Amazon Keyspaces di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi

selengkapnya, lihat [Kebijakan yang dikelola AWS](#) atau [Kebijakan yang dikelola AWS untuk fungsi tugas](#) dalam Panduan Pengguna IAM.

- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat [Kebijakan dan izin dalam IAM](#) dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Kondisi](#) dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat [Validasi kebijakan dengan IAM Access Analyzer](#) dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat [Amankan akses API dengan MFA](#) dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat [Praktik terbaik keamanan di IAM](#) dalam Panduan Pengguna IAM.

## Menggunakan konsol Amazon Keyspaces

Amazon Keyspaces tidak memerlukan izin khusus untuk mengakses konsol Amazon Keyspaces. Anda memerlukan setidaknya izin hanya-baca untuk membuat daftar dan melihat detail tentang sumber daya Amazon Keyspaces di Anda. Akun AWS Jika Anda membuat kebijakan berbasis

identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tersebut tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna IAM atau peran) dengan kebijakan tersebut.

Dua kebijakan AWS terkelola tersedia untuk entitas untuk akses konsol Amazon Keyspaces.

- [AmazonKeyspacesReadOnlyAccess\\_v2](#) — Kebijakan ini memberikan akses hanya-baca ke Amazon Keyspaces.
- [AmazonKeyspacesFullAccess](#) — Kebijakan ini memberikan izin untuk menggunakan Amazon Keyspaces dengan akses penuh ke semua fitur.

Untuk informasi selengkapnya tentang kebijakan terkelola Amazon Keyspaces, lihat. [the section called “AWS kebijakan terkelola”](#)

## Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI 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": "*"
}
]
}

```

## Mengakses tabel Amazon Keyspaces

### Note

Untuk mengakses ruang kunci dan tabel pengguna di Amazon Keyspaces, kebijakan IAM Anda harus `cassandra:Select` menyertakan izin pada tabel sistem:

```
arn:${Partition}:cassandra:${Region}:${Account}:./keyspace/system*
```

Ini berlaku untuk skenario berikut:

- AWS Akses Konsol Manajemen
- Operasi sumber daya SDK, misalnya `GetKeyspace`, `GetTableListKeyspaces`, dan `ListTables`
- Koneksi driver klien Apache Cassandra standar, karena driver secara otomatis membaca tabel sistem selama inisialisasi koneksi

Tabel sistem hanya-baca dan tidak dapat dimodifikasi.

Berikut ini adalah contoh kebijakan yang memberikan akses read-only (SELECT) ke tabel sistem Amazon Keyspaces. Untuk semua sampel, ganti Region dan ID akun di Amazon Resource Name (ARN) dengan milik Anda.

```

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

```



```

"Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "cassandra:Select"
],
 "Resource": [
 "arn:aws:cassandra:us-east-1:111122223333:/keyspace/system*"
]
 }
]
}

```

Kebijakan contoh berikut menambahkan akses hanya-baca ke tabel pengguna mytable di ruang kunci. 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*"
]
 }
]
}

```

Kebijakan contoh berikut menetapkan read/write akses ke tabel pengguna dan akses baca ke tabel sistem.

```

{
 "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*"
]
}
]
}

```

Kebijakan contoh berikut memungkinkan pengguna untuk membuat tabel di keyspace 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*"
]
 }
]
}

```

Kebijakan contoh berikut menetapkan akses baca ke tabel sistem, tetapi membatasi akses SELECT (baca) dan MODIFY (tuliskan) ke tabel pengguna. 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"
]
}
]
}

```

## Akses sumber daya Amazon Keyspaces berdasarkan tag

Anda dapat menggunakan kondisi dalam kebijakan berbasis identitas untuk mengontrol akses ke sumber daya Amazon Keyspaces berdasarkan tag. Kebijakan ini mengontrol visibilitas ruang kunci dan tabel di akun. Perhatikan bahwa izin berbasis tag untuk tabel sistem berperilaku berbeda saat permintaan dibuat menggunakan AWS SDK dibandingkan dengan panggilan API Cassandra Query Language (CQL) melalui driver Cassandra dan alat pengembang.

- Untuk membuat List dan meminta Get sumber daya dengan AWS SDK saat menggunakan akses berbasis tag, pemanggil harus memiliki akses baca ke tabel sistem. Misalnya, izin Select tindakan diperlukan untuk membaca data dari tabel sistem melalui GetTable operasi. Jika penelepon hanya memiliki akses berbasis tag ke tabel tertentu, operasi yang memerlukan akses tambahan ke tabel sistem akan gagal.
- Untuk kompatibilitas dengan perilaku driver Cassandra yang mapan, kebijakan otorisasi berbasis tag tidak diberlakukan saat melakukan operasi pada tabel sistem menggunakan panggilan API Cassandra Query Language (CQL) melalui driver Cassandra dan alat pengembang.

Contoh berikut menunjukkan cara membuat kebijakan yang memberikan izin kepada pengguna untuk melihat tabel jika tabel Owner berisi nilai nama pengguna tersebut. Dalam contoh ini Anda juga memberikan akses baca ke tabel sistem.

```

{
 "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}"
 }
 }
}
```

Anda dapat melampirkan kebijakan ini ke pengguna IAM di akun Anda. Jika pengguna bernama `richard-roe` mencoba melihat tabel Amazon Keyspaces, tabel harus diberi tag `Owner=richard-roe` atau `owner=richard-roe`. Jika tidak, aksesnya akan ditolak. Kunci tanda syarat `Owner` sama dengan kedua `Owner` dan `owner` karena nama kunci syarat tidak terpengaruh huruf besar/kecil. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM JSON: Syarat](#) dalam Panduan Pengguna IAM.

Kebijakan berikut memberikan izin kepada pengguna untuk membuat tabel dengan tag jika tabel `Owner` berisi nilai nama pengguna tersebut.

```
{
 "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 kebijakan terkelola untuk Amazon Keyspaces

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan [kebijakan yang dikelola pelanggan](#) yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pemutakhiran akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat [Kebijakan terkelola AWS](#) dalam Panduan Pengguna IAM.

### AWS kebijakan terkelola: AmazonKeyspacesReadOnlyAccess \_v2

Anda dapat melampirkan kebijakan AmazonKeyspacesReadOnlyAccess\_v2 ke identitas IAM Anda.

Kebijakan ini memberikan akses hanya-baca ke Amazon Keyspaces dan menyertakan izin yang diperlukan saat menghubungkan melalui titik akhir VPC pribadi.

#### Detail izin

Kebijakan ini mencakup izin berikut.

- **Amazon Keyspaces**— Menyediakan akses hanya-baca ke Amazon Keyspaces.
- **Amazon Keyspaces CDC streams**— Memungkinkan kepala sekolah untuk melihat aliran CDC Amazon Keyspaces.
- **Application Auto Scaling**— Memungkinkan prinsipal untuk melihat konfigurasi dari Application Auto Scaling. Ini diperlukan agar pengguna dapat melihat kebijakan penskalaan otomatis yang dilampirkan ke tabel.
- **CloudWatch**— Memungkinkan kepala sekolah untuk melihat data metrik dan alarm yang dikonfigurasi. CloudWatch Ini diperlukan agar pengguna dapat melihat ukuran tabel yang dapat ditagih dan CloudWatch alarm yang telah dikonfigurasi untuk tabel.
- **AWS KMS**— Memungkinkan kepala sekolah untuk melihat kunci yang dikonfigurasi di AWS KMS. Ini diperlukan agar pengguna dapat melihat AWS KMS kunci yang mereka buat dan kelola di akun mereka untuk mengonfirmasi bahwa kunci yang ditetapkan ke Amazon Keyspaces adalah kunci enkripsi simetris yang diaktifkan.
- **Amazon EC2**— Memungkinkan prinsipal yang terhubung ke Amazon Keyspaces melalui titik akhir VPC untuk menanyakan VPC di instans Amazon EC2 Anda untuk informasi titik akhir dan antarmuka jaringan. Akses hanya-baca ke instans Amazon EC2 ini diperlukan agar Amazon Keyspaces dapat mencari dan menyimpan titik akhir VPC antarmuka yang tersedia dalam tabel yang digunakan untuk penyeimbangan beban koneksi. `system.peers`

Untuk meninjau kebijakan dalam JSON format, lihat [AmazonKeyspacesReadOnlyAccess\\_v2](#).

## AWS kebijakan terkelola: AmazonKeyspacesReadOnlyAccess

Anda dapat melampirkan kebijakan AmazonKeyspacesReadOnlyAccess ke identitas IAM Anda.

Kebijakan ini memberikan akses hanya-baca ke Amazon Keyspaces.

### Detail izin

Kebijakan ini mencakup izin berikut.

- **Amazon Keyspaces**— Menyediakan akses hanya-baca ke Amazon Keyspaces.
- **Amazon Keyspaces CDC streams**— Memungkinkan kepala sekolah untuk melihat aliran CDC Amazon Keyspaces.
- **Application Auto Scaling**— Memungkinkan prinsipal untuk melihat konfigurasi dari Application Auto Scaling. Ini diperlukan agar pengguna dapat melihat kebijakan penskalaan otomatis yang dilampirkan ke tabel.
- **CloudWatch**— Memungkinkan kepala sekolah untuk melihat data metrik dan alarm yang dikonfigurasi. CloudWatch Ini diperlukan agar pengguna dapat melihat ukuran tabel yang dapat ditagih dan CloudWatch alarm yang telah dikonfigurasi untuk tabel.
- **AWS KMS**— Memungkinkan kepala sekolah untuk melihat kunci yang dikonfigurasi di AWS KMS. Ini diperlukan agar pengguna dapat melihat AWS KMS kunci yang mereka buat dan kelola di akun mereka untuk mengonfirmasi bahwa kunci yang ditetapkan ke Amazon Keyspaces adalah kunci enkripsi simetris yang diaktifkan.

Untuk meninjau kebijakan dalam JSON format, lihat [AmazonKeyspacesReadOnlyAccess](#).

## AWS kebijakan terkelola: AmazonKeyspacesFullAccess

Anda dapat melampirkan kebijakan AmazonKeyspacesFullAccess ke identitas IAM Anda.

Kebijakan ini memberikan izin administratif yang memungkinkan administrator Anda mengakses tanpa batas ke Amazon Keyspaces.

### Detail izin

Kebijakan ini mencakup izin berikut.

- **Amazon Keyspaces**— Memungkinkan prinsipal mengakses sumber daya Amazon Keyspaces apa pun dan melakukan semua tindakan.
- **Application Auto Scaling**— Memungkinkan prinsipal untuk membuat, melihat, dan menghapus kebijakan penskalaan otomatis untuk tabel Amazon Keyspaces. Ini diperlukan agar administrator dapat mengelola kebijakan penskalaan otomatis untuk tabel Amazon Keyspaces.
- **CloudWatch**— Memungkinkan kepala sekolah melihat ukuran tabel yang dapat ditagih serta membuat, melihat, dan menghapus alarm CloudWatch untuk kebijakan penskalaan otomatis

Amazon Keyspaces. Ini diperlukan agar administrator dapat melihat ukuran tabel yang dapat ditagih dan membuat dasbor. CloudWatch

- IAM— Memungkinkan Amazon Keyspaces untuk membuat peran terkait layanan dengan IAM secara otomatis ketika fitur berikut diaktifkan:
  - Amazon Keyspaces CDC streams— Saat administrator mengaktifkan aliran untuk tabel, Amazon Keyspaces membuat peran terkait layanan [AWSServiceRoleForAmazonKeyspacesCDC](#) untuk mempublikasikan CloudWatch metrik ke akun Anda atas nama Anda.
  - Application Auto Scaling— Saat administrator mengaktifkan Application Auto Scaling untuk sebuah tabel, Amazon Keyspaces membuat peran terkait layanan [AWSServiceRoleForApplicationAutoScaling\\_CassandraTable](#) untuk melakukan tindakan penskalaan otomatis atas nama Anda.
  - Amazon Keyspaces multi-Region replication— Saat administrator membuat ruang kunci Multi-wilayah baru, atau menambahkan yang baru Wilayah AWS ke ruang kunci Wilayah Tunggal yang ada, Amazon Keyspaces membuat [AWSServiceRoleForAmazonKeyspacesReplication](#) peran terkait layanan untuk melakukan replikasi tabel, data, dan metadata ke Wilayah yang dipilih atas nama Anda.
- AWS KMS— Memungkinkan kepala sekolah untuk melihat kunci yang dikonfigurasi di AWS KMS. Ini diperlukan agar pengguna dapat melihat AWS KMS kunci yang mereka buat dan kelola di akun mereka untuk mengonfirmasi bahwa kunci yang ditetapkan ke Amazon Keyspaces adalah kunci enkripsi simetris yang diaktifkan.
- Amazon EC2— Memungkinkan prinsipal yang terhubung ke Amazon Keyspaces melalui titik akhir VPC untuk menanyakan VPC di instans Amazon EC2 Anda untuk informasi titik akhir dan antarmuka jaringan. Akses hanya-baca ke instans Amazon EC2 ini diperlukan agar Amazon Keyspaces dapat mencari dan menyimpan titik akhir VPC antarmuka yang tersedia dalam tabel yang digunakan untuk penyeimbangan beban koneksi. `system.peers`

Untuk meninjau kebijakan dalam JSON format, lihat [AmazonKeyspacesFullAccess](#).

## AWS kebijakan terkelola: KeyspacesCDCServiceRolePolicy

Anda tidak dapat melampirkan KeyspacesCDCServiceRolePolicy ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran terkait layanan yang memungkinkan Amazon Keyspaces



melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat [the section called “Aliran CDC”](#).

Kebijakan ini memberikan izin yang diperlukan ke peran terkait layanan untuk memublikasikan data metrik aliran CDC `AWSServiceRoleForAmazonKeyspacesCDC` Amazon Keyspaces atas nama Anda. CloudWatch

Detail izin

Kebijakan ini mencakup izin berikut.

- CloudWatch— Memungkinkan peran yang ditautkan layanan [AWSServiceRoleForAmazonKeyspacesCDC](#) untuk memublikasikan data metrik dari aliran CDC Amazon Keyspaces ke dalam akun Anda atas nama Anda. `"cloudwatch:namespace": "AWS/Cassandra"` CloudWatch

Untuk meninjau kebijakan dalam JSON format, lihat [KeyspacesCDCServiceRolePolicy](#).

## Amazon Keyspaces memperbarui kebijakan terkelola AWS

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Amazon Keyspaces sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan ke umpan RSS pada halaman [Riwayat dokumen](#).

| Perubahan                                                      | Deskripsi                                                                                                                                                             | Date         |
|----------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| <a href="#">KeyspacesCDCServiceRolePolicy</a> – Kebijakan baru | Amazon Keyspaces menambahkan kebijakan terkelola baru yang memberikan izin Keyspaces <code>CDCServiceRolePolicy</code> yang diperlukan ke peran terkait layanan untuk | 02 Juli 2025 |

| Perubahan                                                                         | Deskripsi                                                                                                                                                                                                                       | Date         |
|-----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
|                                                                                   | <p><code>AWSServiceRoleForAmazonKeyspacesCDC</code> mempublikasikan data metrik aliran CDC Amazon Keyspaces atas nama Anda. CloudWatch Untuk informasi selengkapnya, lihat <a href="#">the section called “Aliran CDC”</a>.</p> |              |
| <a href="#">AmazonKeyspacesReadOnlyAccess_v2</a> - Perbarui ke kebijakan yang ada | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan prinsipal IAM melihat aliran CDC Amazon Keyspaces. Untuk informasi selengkapnya, lihat <a href="#">the section called “Lihat aliran CDC”</a>.</p>                  | 02 Juli 2025 |
| <a href="#">AmazonKeyspacesReadOnlyAccess</a> – Pembaruan ke kebijakan yang ada   | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan prinsipal IAM melihat aliran CDC Amazon Keyspaces. Untuk informasi selengkapnya, lihat <a href="#">the section called “Lihat aliran CDC”</a>.</p>                  | 02 Juli 2025 |

| Perubahan                                                                      | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Date         |
|--------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| <a href="#">AmazonKeyspacesFullAccess</a><br>– Pembaruan ke kebijakan yang ada | <p>Amazon Keyspaces membuat kebijakan Keyspaces <code>CDCServiceRolePolicy</code> terkelola <a href="#">AWSServiceRoleForAmazonKeyspacesCDC</a> untuk peran terkait layanan guna menambahkan izin yang diperlukan saat administrator mengaktifkan aliran untuk tabel.</p> <p>Amazon Keyspaces menggunakan peran terkait layanan <code>AWSServiceRoleForAmazonKeyspacesCDC</code> untuk mempublikasikan CloudWatch metrik ke akun Anda atas nama Anda. Untuk informasi selengkapnya, lihat <a href="#">the section called “Aliran CDC”</a>.</p> | 02 Juli 2025 |

| Perubahan                                                                      | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Date              |
|--------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <a href="#">AmazonKeyspacesFullAccess</a><br>– Pembaruan ke kebijakan yang ada | <p>Amazon Keyspaces memperbarui peran yang KeyspacesReplicationServiceRolePolicy ditautkan layanan <a href="#">AWSServiceRoleForAmazonKeyspacesReplication</a> untuk menambahkan izin yang diperlukan saat administrator menambahkan yang baru Wilayah AWS ke ruang kunci tunggal atau Multi-wilayah.</p> <p>Amazon Keyspaces menggunakan peran terkait layanan AWSServiceRoleForAmazonKeyspacesReplication untuk mereplikasi tabel, pengaturannya, dan data atas nama Anda. Untuk informasi selengkapnya, lihat <a href="#">the section called “Multi-Region Replikasi”</a>.</p> | November 19, 2024 |

| Perubahan                                                                      | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                              | Date           |
|--------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|
| <a href="#">AmazonKeyspacesFullAccess</a><br>– Pembaruan ke kebijakan yang ada | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan Amazon Keyspaces membuat peran terkait layanan saat administrator menambahkan Wilayah baru ke ruang kunci tunggal atau Multi-wilayah.</p> <p>Amazon Keyspaces menggunakan peran terkait layanan untuk melakukan tugas replikasi data atas nama Anda. Untuk informasi selengkapnya, lihat <a href="#">the section called “Multi-Region Replikasi”</a>.</p> | 3 Oktober 2023 |

| Perubahan                                                         | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Date              |
|-------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <a href="#">AmazonKeyspacesReadOnlyAccess_v2</a> - Kebijakan baru | <p>Amazon Keyspaces membuat kebijakan baru untuk menambahkan izin hanya-baca bagi klien yang terhubung ke Amazon Keyspaces melalui titik akhir VPC antarmuka untuk mengakses instans Amazon EC2 guna mencari informasi jaringan.</p> <p>Amazon Keyspaces menyimpan titik akhir VPC antarmuka yang tersedia dalam <code>system.peers</code> tabel untuk penyeimbangan beban koneksi. Untuk informasi selengkapnya, lihat <a href="#">the section called “Menggunakan VPC endpoint antarmuka”</a>.</p> | 12 September 2023 |

| Perubahan                                                                          | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                      | Date         |
|------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| <a href="#">AmazonKeyspacesFullAccess</a><br>– Pembaruan ke kebijakan yang ada     | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan Amazon Keyspaces membuat peran terkait layanan saat administrator membuat ruang kunci Multi-wilayah.</p> <p>Amazon Keyspaces menggunakan peran terkait layanan <code>AWSServiceRoleForAmazonKeyspacesReplication</code> untuk melakukan tugas replikasi data atas nama Anda. Untuk informasi selengkapnya, lihat <a href="#">the section called “Multi-Region Replikasi”</a>.</p> | Juni 5, 2023 |
| <a href="#">AmazonKeyspacesReadOnlyAccess</a><br>– Pembaruan ke kebijakan yang ada | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan pengguna melihat ukuran tabel yang dapat ditagih menggunakan CloudWatch</p> <p>Amazon Keyspaces terintegrasi dengan Amazon CloudWatch untuk memungkinkan Anda memantau ukuran tabel yang dapat ditagih. Untuk informasi selengkapnya, lihat <a href="#">the section called “Metrik Amazon Keyspaces”</a>.</p>                                                                     | Juli 7, 2022 |

| Perubahan                                                                          | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Date         |
|------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| <a href="#">AmazonKeyspacesFullAccess</a><br>– Pembaruan ke kebijakan yang ada     | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan pengguna melihat ukuran tabel yang dapat ditagih menggunakan CloudWatch</p> <p>Amazon Keyspaces terintegrasi dengan Amazon CloudWatch untuk memungkinkan Anda memantau ukuran tabel yang dapat ditagih. Untuk informasi selengkapnya, lihat <a href="#">the section called “Metrik Amazon Keyspaces”</a>.</p>                                                                                          | Juli 7, 2022 |
| <a href="#">AmazonKeyspacesReadOnlyAccess</a><br>– Pembaruan ke kebijakan yang ada | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan pengguna melihat AWS KMS kunci yang telah dikonfigurasi untuk enkripsi Amazon Keyspaces saat istirahat.</p> <p>Enkripsi Amazon Keyspaces saat istirahat terintegrasi dengan AWS KMS untuk melindungi dan mengelola kunci enkripsi yang digunakan untuk mengenkripsi data saat istirahat. Untuk melihat AWS KMS kunci yang dikonfigurasi untuk Amazon Keyspaces, izin hanya-baca telah ditambahkan.</p> | 1 Juni 2021  |



| Perubahan                                                                      | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Date        |
|--------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| <a href="#">AmazonKeyspacesFullAccess</a><br>– Pembaruan ke kebijakan yang ada | <p>Amazon Keyspaces menambahkan izin baru untuk memungkinkan pengguna melihat AWS KMS kunci yang telah dikonfigurasi untuk enkripsi Amazon Keyspaces saat istirahat.</p> <p>Enkripsi Amazon Keyspaces saat istirahat terintegrasi dengan AWS KMS untuk melindungi dan mengelola kunci enkripsi yang digunakan untuk mengenkripsi data saat istirahat. Untuk melihat AWS KMS kunci yang dikonfigurasi untuk Amazon Keyspaces, izin hanya-baca telah ditambahkan.</p> | 1 Juni 2021 |
| Amazon Keyspaces mulai melacak perubahan                                       | Amazon Keyspaces mulai melacak perubahan untuk kebijakan yang AWS dikelola.                                                                                                                                                                                                                                                                                                                                                                                         | 1 Juni 2021 |

## Memecahkan masalah identitas dan akses Amazon Keyspaces

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon Keyspaces dan IAM.

### Topik

- [Saya tidak berwenang untuk melakukan tindakan di Amazon Keyspaces](#)
- [Saya memodifikasi pengguna atau peran IAM dan perubahan tidak segera berlaku](#)
- [Saya tidak dapat memulihkan tabel menggunakan pemulihan point-in-time Amazon Keyspaces \(PITR\)](#)

- [Saya tidak berwenang untuk melakukan iam: PassRole](#)
- [Saya seorang administrator dan ingin mengizinkan orang lain mengakses Amazon Keyspaces](#)
- [Saya ingin mengizinkan orang di luar saya Akun AWS mengakses sumber daya Amazon Keyspaces saya](#)

## Saya tidak berwenang untuk melakukan tindakan di Amazon Keyspaces

Jika Konsol Manajemen AWS memberitahu Anda bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk bantuan. Administrator Anda adalah orang yang memberikan nama pengguna dan kata sandi Anda.

Contoh kesalahan berikut terjadi ketika pengguna mateojackson IAM mencoba menggunakan konsol untuk melihat detail tentang *table* tetapi tidak memiliki `cassandra:Select` izin untuk tabel.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
cassandra:Select on resource: mytable
```

Dalam hal ini, Mateo meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan dia mengakses sumber daya *mytable* menggunakan tindakan `cassandra:Select`.

## Saya memodifikasi pengguna atau peran IAM dan perubahan tidak segera berlaku

Perubahan kebijakan IAM dapat memakan waktu hingga 10 menit untuk diterapkan pada aplikasi dengan koneksi yang sudah ada dan telah ditetapkan ke Amazon Keyspaces. Perubahan kebijakan IAM segera berlaku ketika aplikasi membuat koneksi baru. Jika Anda telah membuat modifikasi pada pengguna atau peran IAM yang ada, dan itu belum segera berlaku, tunggu selama 10 menit atau putuskan sambungan dan sambungkan kembali ke Amazon Keyspaces.

## Saya tidak dapat memulihkan tabel menggunakan pemulihan point-in-time Amazon Keyspaces (PITR)

Jika Anda mencoba memulihkan tabel Amazon Keyspaces dengan point-in-time recovery (PITR), dan Anda melihat proses pemulihan dimulai, tetapi tidak berhasil diselesaikan, Anda mungkin belum mengonfigurasi semua izin yang diperlukan yang diperlukan oleh proses pemulihan. Anda harus menghubungi administrator untuk mendapatkan bantuan dan meminta orang tersebut memperbarui kebijakan Anda agar Anda dapat memulihkan tabel di Amazon Keyspaces.

Selain izin pengguna, Amazon Keyspaces mungkin memerlukan izin untuk melakukan tindakan selama proses pemulihan atas nama kepala sekolah Anda. Ini adalah kasus jika tabel dienkripsi

dengan kunci yang dikelola pelanggan, atau jika Anda menggunakan kebijakan IAM yang membatasi lalu lintas masuk. Misalnya, jika Anda menggunakan kunci kondisi dalam kebijakan IAM Anda untuk membatasi lalu lintas sumber ke titik akhir atau rentang IP tertentu, operasi pemulihan gagal. Untuk mengizinkan Amazon Keyspaces menjalankan operasi pemulihan tabel atas nama kepala sekolah, Anda harus menambahkan kunci kondisi `aws:ViaAWSService` global dalam kebijakan IAM.

Untuk informasi selengkapnya tentang izin untuk memulihkan tabel, lihat [the section called “Konfigurasi izin IAM untuk memulihkan”](#).

## Saya tidak berwenang untuk melakukan `iam:PassRole`

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke Amazon Keyspaces.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di Amazon Keyspaces. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

## Saya seorang administrator dan ingin mengizinkan orang lain mengakses Amazon Keyspaces

Untuk mengizinkan orang lain mengakses Amazon Keyspaces, Anda harus memberikan izin kepada orang atau aplikasi yang memerlukan akses. Jika Anda menggunakan AWS IAM Identity Center untuk mengelola orang dan aplikasi, Anda menetapkan set izin kepada pengguna atau grup untuk menentukan tingkat akses mereka. Set izin secara otomatis membuat dan menetapkan kebijakan

IAM ke peran IAM yang terkait dengan orang atau aplikasi. Untuk informasi selengkapnya, lihat [Set izin](#) di Panduan AWS IAM Identity Center Pengguna.

Jika Anda tidak menggunakan IAM Identity Center, Anda harus membuat entitas IAM (pengguna atau peran) untuk orang atau aplikasi yang membutuhkan akses. Anda kemudian harus melampirkan kebijakan ke entitas yang memberi mereka izin yang benar di Amazon Keyspaces. Setelah izin diberikan, berikan kredensialnya kepada pengguna atau pengembang aplikasi. Mereka akan menggunakan kredensial tersebut untuk mengakses AWS. Untuk mempelajari selengkapnya tentang membuat pengguna, grup, kebijakan, dan izin IAM, lihat [Identitas dan Kebijakan IAM dan izin di IAM di Panduan Pengguna IAM](#).

Saya ingin mengizinkan orang di luar saya Akun AWS mengakses sumber daya Amazon Keyspaces saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mengetahui apakah Amazon Keyspaces mendukung fitur ini, lihat [Cara Amazon Keyspaces bekerja dengan IAM](#)
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat [Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna terautentikasi eksternal \(federasi identitas\)](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM](#).

## Menggunakan peran terkait layanan untuk Amazon Keyspaces

[Amazon Keyspaces \(untuk Apache Cassandra\) menggunakan peran terkait layanan AWS Identity and Access Management \(IAM\)](#). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke Amazon Keyspaces. Service-linked peran telah ditentukan sebelumnya oleh Amazon Keyspaces dan menyertakan semua izin yang diperlukan layanan untuk memanggil layanan AWS lain atas nama Anda.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat [AWS layanan yang bekerja dengan IAM](#) dan cari layanan yang memiliki Ya di kolom peran. Service-linked Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

### Topik

- [Menggunakan peran untuk penskalaan otomatis aplikasi Amazon Keyspaces](#)
- [Menggunakan peran untuk Replikasi Amazon Keyspaces Multi-Region](#)
- [Menggunakan peran untuk aliran CDC Amazon Keyspaces](#)

## Menggunakan peran untuk penskalaan otomatis aplikasi Amazon Keyspaces

[Amazon Keyspaces \(untuk Apache Cassandra\) menggunakan peran terkait layanan AWS Identity and Access Management \(IAM\)](#). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke Amazon Keyspaces. Service-linked peran telah ditentukan sebelumnya oleh Amazon Keyspaces dan menyertakan semua izin yang diperlukan layanan untuk memanggil layanan AWS lain atas nama Anda.

Peran terkait layanan membuat pengaturan Amazon Keyspaces lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon Keyspaces mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya Amazon Keyspaces yang dapat mengambil perannya. Izin-izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan hanya setelah terlebih dahulu menghapus sumber dayanya yang terkait. Ini melindungi sumber daya Amazon Keyspaces karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

## Service-linked izin peran untuk Amazon Keyspaces

Amazon Keyspaces menggunakan peran terkait layanan bernama `AWSServiceRoleForApplicationAutoScaling_CassandraTable` untuk memungkinkan Application Auto Scaling memanggil Amazon Keyspaces dan Amazon atas nama Anda. CloudWatch

Peran `CassandraTable` terkait layanan `AWSServiceRoleForApplicationAutoScaling` \_ mempercayai layanan berikut untuk mengambil peran:

- `cassandra.application-autoscaling.amazonaws.com`

Kebijakan izin peran memungkinkan Application Auto Scaling untuk menyelesaikan tindakan berikut pada resource Amazon Keyspaces yang ditentukan:

- Tindakan: `cassandra:Select` pada `arn:*:cassandra:*:*:/keyspace/system/table/*`
- Tindakan: `cassandra:Select` pada sumber daya `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
- Tindakan: `cassandra:Select` pada sumber daya `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Tindakan: `cassandra:Alter` pada sumber daya `arn:*:cassandra:*:*:""`

## Membuat peran terkait layanan untuk Amazon Keyspaces

Anda tidak perlu membuat peran terkait layanan secara manual untuk penskalaan otomatis Amazon Keyspaces. Saat Anda mengaktifkan penskalaan otomatis Amazon Keyspaces pada tabel dengan Konsol Manajemen AWS, CQL, atau API, Application AWS Auto Scaling akan membuat peran terkait layanan untuk Anda. AWS CLI

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda mengaktifkan penskalaan otomatis Amazon Keyspaces untuk sebuah tabel, Application Auto Scaling akan membuat peran terkait layanan untuk Anda lagi.

 Important

Peran tertaut layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang disupport oleh peran ini. Untuk mempelajari lebih lanjut, lihat [Peran baru muncul di saya Akun AWS](#).

## Mengedit peran terkait layanan untuk Amazon Keyspaces

Amazon Keyspaces tidak memungkinkan Anda mengedit peran terkait CassandraTable layanan AWSServiceRoleForApplicationAutoScaling \_ . Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat [Mengedit peran terkait layanan](#) dalam Panduan Pengguna IAM.

## Menghapus peran terkait layanan untuk Amazon Keyspaces

Jika Anda tidak lagi memerlukan penggunaan fitur atau layanan yang memerlukan peran terkait layanan, kami menyarankan Anda untuk menghapus peran tersebut. Dengan begitu Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus terlebih dahulu menonaktifkan penskalaan otomatis pada semua tabel di akun Wilayah AWS sebelum Anda dapat menghapus peran terkait layanan secara manual. Untuk menonaktifkan penskalaan otomatis pada tabel Amazon Keyspaces, lihat. [the section called “Matikan penskalaan otomatis Amazon Keyspaces untuk sebuah tabel”](#)

 Note

Jika penskalaan otomatis Amazon Keyspaces menggunakan peran saat Anda mencoba memodifikasi sumber daya, maka deregistrasi mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

## Untuk menghapus peran tertaut layanan secara manual menggunakan IAM

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran AWSServiceRoleForApplicationAutoScaling \_ CassandraTable service-linked. Untuk informasi selengkapnya, lihat [Menghapus Service-Linked Peran](#) di Panduan Pengguna IAM.

 Note

Untuk menghapus peran terkait layanan yang digunakan oleh penskalaan otomatis Amazon Keyspaces, Anda harus terlebih dahulu menonaktifkan penskalaan otomatis pada semua tabel di akun.

## Wilayah yang Didukung untuk peran terkait layanan Amazon Keyspaces

Amazon Keyspaces mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat [Titik akhir layanan untuk Amazon Keyspaces](#).

## Menggunakan peran untuk Replikasi Amazon Keyspaces Multi-Region

[Amazon Keyspaces \(untuk Apache Cassandra\) menggunakan peran terkait layanan AWS Identity and Access Management \(IAM\)](#). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke Amazon Keyspaces. Service-linked peran telah ditentukan sebelumnya oleh Amazon Keyspaces dan menyertakan semua izin yang diperlukan layanan untuk memanggil layanan AWS lain atas nama Anda.

Peran terkait layanan membuat pengaturan Amazon Keyspaces lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon Keyspaces mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya Amazon Keyspaces yang dapat mengambil perannya. Izin-izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan hanya setelah terlebih dahulu menghapus sumber dayanya yang terkait. Ini melindungi sumber daya Amazon Keyspaces karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

### Service-linked izin peran untuk Amazon Keyspaces

Amazon Keyspaces menggunakan peran terkait layanan yang diberi nama untuk memungkinkan `AWSServiceRoleForAmazonKeyspacesReplication` Amazon Keyspaces menambahkan baru Wilayah AWS ke ruang kunci atas nama Anda, dan mereplikasi tabel serta semua data serta pengaturannya ke Wilayah baru. Peran ini juga memungkinkan Amazon Keyspaces untuk mereplikasi penulisan ke tabel di semua Wilayah atas nama Anda.

Peran `AWSServiceRoleForAmazonKeyspacesReplication` terkait layanan mempercayai layanan berikut untuk mengambil peran:



- `replication.cassandra.amazonaws.com`

Kebijakan izin peran bernama `KeyspacesReplicationServiceRolePolicy` memungkinkan Amazon Keyspaces untuk menyelesaikan tindakan berikut:

- Tindakan: `cassandra:Select`
- Tindakan: `cassandra:SelectMultiRegionResource`
- Tindakan: `cassandra:Modify`
- Tindakan: `cassandra:ModifyMultiRegionResource`
- Tindakan: `cassandra:AlterMultiRegionResource`
- Tindakan: `application-autoscaling:RegisterScalableTarget` — Amazon Keyspaces menggunakan izin penskalaan otomatis aplikasi saat Anda menambahkan replika ke satu tabel Wilayah dalam mode yang disediakan dengan penskalaan otomatis diaktifkan.
- Tindakan: `application-autoscaling:DeregisterScalableTarget`
- Tindakan: `application-autoscaling:DescribeScalableTargets`
- Tindakan: `application-autoscaling:PutScalingPolicy`
- Tindakan: `application-autoscaling:DescribeScalingPolicies`
- Tindakan: `cassandra:Alter`
- Tindakan: `cloudwatch:DeleteAlarms`
- Tindakan: `cloudwatch:DescribeAlarms`
- Tindakan: `cloudwatch:PutMetricAlarm`

Meskipun peran terkait layanan Amazon Keyspaces

`AWSServiceRoleForAmazonKeyspacesReplication` memberikan izin: “Tindakan:” untuk Nama Sumber Daya Amazon (ARN) yang ditentukan “arn: \*” dalam kebijakan, Amazon Keyspaces menyediakan ARN akun Anda.

Izin untuk membuat peran terkait layanan disertakan dalam

`AWSServiceRoleForAmazonKeyspacesReplication` kebijakan terkelola.

`AmazonKeyspacesFullAccess` Untuk informasi selengkapnya, lihat [the section called “AmazonKeyspacesFullAccess”](#).

Anda harus mengonfigurasi izin agar pengguna, grup, atau peran Anda membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat [izin Service-linked peran](#) di Panduan Pengguna IAM.

### Membuat peran terkait layanan untuk Amazon Keyspaces

Anda tidak dapat membuat peran terkait layanan secara manual. Saat Anda membuat ruang kunci Multi-wilayah di, API Konsol Manajemen AWS, atau AWS API AWS CLI, Amazon Keyspaces akan membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda membuat ruang kunci Multi-wilayah, Amazon Keyspaces akan membuat peran terkait layanan untuk Anda lagi.

### Mengedit peran terkait layanan untuk Amazon Keyspaces

Amazon Keyspaces tidak memungkinkan Anda mengedit peran terkait `AWSServiceRoleForAmazonKeyspacesReplication` layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat [Mengedit peran terkait layanan](#) dalam Panduan Pengguna IAM.

### Menghapus peran terkait layanan untuk Amazon Keyspaces

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, sebaiknya hapus peran tersebut. Dengan begitu, Anda tidak perlu lagi memantau atau memelihara entitas yang tidak digunakan. Namun, Anda harus terlebih dahulu menghapus semua ruang kunci Multi-wilayah di seluruh akun Wilayah AWS sebelum Anda dapat menghapus peran terkait layanan secara manual.

### Membersihkan peran terkait layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran terkait layanan, Anda harus terlebih dahulu menghapus ruang kunci Multi-wilayah dan tabel yang digunakan oleh peran tersebut.

#### Note

Jika layanan Amazon Keyspaces menggunakan peran saat Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus sumber daya Amazon Keyspaces yang digunakan oleh `AWSServiceRoleForAmazonKeyspacesReplication` (konsol)

1. Masuk ke Konsol Manajemen AWS, dan buka konsol Amazon Keyspaces di. <https://console.aws.amazon.com/keyspaces/home>
2. Pilih Keyspaces dari panel sisi kiri.
3. Pilih semua ruang kunci Multi-wilayah dari daftar.
4. Pilih Hapus konfirmasi penghapusan dan pilih Hapus ruang kunci.

Anda juga dapat menghapus ruang kunci Multi-region secara terprogram menggunakan salah satu metode berikut.

- Pernyataan Cassandra Query Language (CQL). [???](#)
- Operasi [delete-keyspace](#) dari CLI. AWS
- [DeleteKeyspace](#) Pengoperasian Amazon Keyspaces API.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran `AWSServiceRoleForAmazonKeyspacesReplication` terkait layanan. Untuk informasi selengkapnya, lihat [Menghapus peran terkait layanan](#) dalam Panduan Pengguna IAM.

Wilayah yang Didukung untuk peran terkait layanan Amazon Keyspaces

Amazon Keyspaces tidak mendukung penggunaan peran terkait layanan di setiap Wilayah tempat layanan tersedia. Anda dapat menggunakan `AWSServiceRoleForAmazonKeyspacesReplication` peran di Wilayah berikut.

| Nama wilayah                | Identitas wilayah | Support di Amazon Keyspaces |
|-----------------------------|-------------------|-----------------------------|
| US East (Northern Virginia) | us-east-1         | Ya                          |
| US East (Ohio)              | us-east-2         | Ya                          |
| US West (N. California)     | us-west-1         | Ya                          |
| US West (Oregon)            | us-west-2         | Ya                          |

| Nama wilayah              | Identitas wilayah | Support di Amazon Keyspaces |
|---------------------------|-------------------|-----------------------------|
| Asia Pacific (Mumbai)     | ap-south-1        | Ya                          |
| Asia Pacific (Osaka)      | ap-northeast-3    | Ya                          |
| Asia Pacific (Seoul)      | ap-northeast-2    | Ya                          |
| Asia Pacific (Singapore)  | ap-southeast-1    | Ya                          |
| Asia Pacific (Sydney)     | ap-southeast-2    | Ya                          |
| Asia Pacific (Tokyo)      | ap-northeast-1    | Ya                          |
| Canada (Central)          | ca-central-1      | Ya                          |
| Eropa (Frankfurt)         | eu-central-1      | Ya                          |
| Eropa (Irlandia)          | eu-west-1         | Ya                          |
| Eropa (London)            | eu-west-2         | Ya                          |
| Europe (Paris)            | eu-west-3         | Ya                          |
| Africa (Cape Town)        | af-south-1        | Ya                          |
| South America (São Paulo) | sa-east-1         | Ya                          |
| AWS GovCloud (US-East)    | us-gov-east-1     | Tidak                       |
| AWS GovCloud (US-West)    | us-gov-west-1     | Tidak                       |

## Menggunakan peran untuk aliran CDC Amazon Keyspaces

[Amazon Keyspaces \(untuk Apache Cassandra\) menggunakan peran terkait layanan AWS Identity and Access Management \(IAM\)](#). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke Amazon Keyspaces. Service-linked peran telah ditentukan sebelumnya oleh Amazon Keyspaces dan menyertakan semua izin yang diperlukan layanan untuk memanggil layanan AWS lain atas nama Anda.

Peran terkait layanan membuat pengaturan Amazon Keyspaces lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon Keyspaces mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya Amazon Keyspaces yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda tidak dapat menghapus peran terkait layanan.

Service-linked izin peran untuk Amazon Keyspaces

Amazon Keyspaces menggunakan peran terkait layanan bernama untuk memungkinkan aliran CDC `AWSServiceRoleForAmazonKeyspacesCDC`. Amazon Keyspaces mempublikasikan CloudWatch metrik ke akun Anda atas nama Anda.

Peran `AWSServiceRoleForAmazonKeyspacesCDC` terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `cassandra-streams.amazonaws.com`

Kebijakan izin peran bernama [KeyspacesCDCServiceRolePolicy](#) memungkinkan Amazon Keyspaces menyelesaikan tindakan berikut pada resource di CloudWatch namespace: `AWS/Cassandra`

- Tindakan: `cloudwatch:PutMetricData` pada \*

`AWSServiceRoleForAmazonKeyspacesCDC` Menyediakan izin: Tindakan: `cloudwatch:PutMetricData` pada semua sumber daya yang cocok dengan kondisi berikut:  
"`cloudwatch:namespace`": "`AWS/Cassandra`"

Untuk informasi lebih lanjut tentang `KeyspacesCDCServiceRolePolicy`, lihat [the section called "KeyspacesCDCServiceRolePolicy"](#).

Untuk mengaktifkan aliran CDC untuk tabel, yang secara otomatis membuat peran terkait layanan `AWSServiceRoleForAmazonKeyspacesCDC`, prinsipal IAM memerlukan izin berikut.

```
{
 "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"
 }
}
```

Izin untuk membuat peran terkait layanan disertakan dalam `AWSServiceRoleForAmazonKeyspacesCDC` kebijakan terkelola. `AmazonKeyspacesFullAccess`. Untuk informasi selengkapnya, lihat [the section called “AmazonKeyspacesFullAccess”](#).

### Membuat peran terkait layanan untuk Amazon Keyspaces

Anda tidak perlu membuat peran terkait layanan secara manual untuk aliran CDC Amazon Keyspaces. Saat Anda mengaktifkan aliran CDC Amazon Keyspaces pada tabel dengan Konsol Manajemen AWS, CQL, API, AWS CLI atau API, AWS Amazon Keyspaces akan membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda mengaktifkan aliran CDC Amazon Keyspaces untuk tabel, Amazon Keyspaces akan membuat peran terkait layanan untuk Anda lagi.

### Mengedit peran terkait layanan untuk Amazon Keyspaces

Amazon Keyspaces tidak memungkinkan Anda mengedit peran terkait `AWSServiceRoleForAmazonKeyspacesCDC` layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat [Mengedit peran terkait layanan](#) dalam Panduan Pengguna IAM.

### Wilayah yang Didukung untuk peran terkait layanan Amazon Keyspaces

Amazon Keyspaces mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat [AWS Wilayah dan titik akhir](#).

## Validasi kepatuhan untuk Amazon Keyspaces (untuk Apache Cassandra)

Third-party auditor menilai keamanan dan kepatuhan Amazon Keyspaces (untuk Apache Cassandra) sebagai bagian dari beberapa program kepatuhan. AWS Ini termasuk:

- ISO/IEC 27001:2013, 27017:2015, 27018:2019, dan 9001:2015. ISO/IEC Untuk informasi selengkapnya, lihat [sertifikasi dan layanan AWS ISO dan CSA STAR](#).
- Kontrol Sistem dan Organisasi (System and Organization Controls/SOC)
- Industri Kartu Pembayaran (Payment Card Industry/PCI)
- Program Manajemen Risiko dan Otorisasi Federal (FedRAMP) Tinggi
- Undang-Undang Akuntabilitas dan Portabilitas Asuransi Kesehatan (HIPAA)

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat [Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS](#) dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program AWS Kepatuhan Program AWS](#) .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifact](#) .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. Untuk informasi selengkapnya tentang tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS, lihat [Dokumentasi AWS Keamanan](#).

## Ketahanan dan pemulihan bencana di Amazon Keyspaces

Infrastruktur AWS global dibangun di sekitar Wilayah AWS dan Availability Zones. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang dan mengoperasikan aplikasi dan basis data yang secara otomatis melakukan failover di antara Zona Ketersediaan tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur biasa yang terdiri dari satu atau beberapa pusat data.

Amazon Keyspaces mereplikasi data secara otomatis tiga kali di beberapa AWS Availability Zone dalam hal yang sama Wilayah AWS untuk daya tahan dan ketersediaan tinggi.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat [infrastruktur AWS global](#).

Selain infrastruktur AWS global, Amazon Keyspaces menawarkan beberapa fitur untuk membantu mendukung ketahanan data dan kebutuhan pencadangan Anda.

## Replikasi multi-wilayah

Amazon Keyspaces menyediakan replikasi Multi-wilayah jika Anda perlu mereplikasi data atau aplikasi Anda pada jarak geografis yang lebih jauh. Anda dapat mereplikasi tabel Amazon Keyspaces Anda di Wilayah AWS berbagai pilihan Anda. Untuk informasi selengkapnya, lihat [the section called “Multi-Region replikasi”](#).

## Point-in-time pemulihan (PITR)

PITR membantu melindungi tabel Amazon Keyspaces Anda dari operasi penulisan atau penghapusan yang tidak disengaja dengan menyediakan pencadangan data tabel Anda secara terus menerus. Untuk informasi selengkapnya, lihat [Point-in-time pemulihan untuk Amazon Keyspaces](#).

## Keamanan infrastruktur di Amazon Keyspaces

Sebagai layanan terkelola, Amazon Keyspaces (untuk Apache Cassandra) dilindungi oleh keamanan jaringan global. AWS Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat [Keamanan AWS Cloud](#). Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan](#).

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon Keyspaces melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Cipher suite dengan perfect forward secrecy (PFS) seperti DHE (Ephemeral) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Amazon Keyspaces mendukung dua metode otentikasi permintaan klien. Metode pertama menggunakan kredensi khusus layanan, yang merupakan kredensial berbasis kata sandi yang dihasilkan untuk pengguna IAM tertentu. Anda dapat membuat dan mengelola kata sandi menggunakan konsol IAM, the AWS CLI, atau AWS API. Untuk informasi selengkapnya, lihat [Menggunakan IAM dengan Amazon Keyspaces](#).

Metode kedua menggunakan plugin otentikasi untuk Driver DataStax Java open-source untuk Cassandra. [Plugin ini memungkinkan pengguna IAM, peran, dan identitas federasi untuk](#)



[menambahkan informasi otentikasi ke permintaan API Amazon Keyspaces \(untuk Apache Cassandra\) menggunakan proses Signature Version 4 \(SigV4\).AWS](#) Untuk informasi selengkapnya, lihat [the section called “Buat kredensi IAM untuk otentikasi AWS”](#).

Anda dapat memanggil operasi API ini dari lokasi jaringan mana pun, tetapi Amazon Keyspaces mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan kebijakan Amazon Keyspaces untuk mengontrol akses dari titik akhir Amazon Virtual Private Cloud (Amazon VPC) tertentu atau VPC tertentu. Secara efektif, ini mengisolasi akses jaringan ke sumber daya Amazon Keyspaces tertentu hanya dari VPC tertentu dalam jaringan. AWS

Anda dapat menggunakan titik akhir VPC antarmuka untuk menjaga lalu lintas antara Amazon VPC dan Amazon Keyspaces agar tidak meninggalkan jaringan Amazon. Endpoint VPC antarmuka didukung oleh AWS PrivateLink, sebuah AWS teknologi yang memungkinkan komunikasi pribadi antar AWS layanan menggunakan antarmuka network elastis dengan IP pribadi di Amazon VPC Anda. Lihat informasi yang lebih lengkap di [the section called “Menggunakan VPC endpoint antarmuka”](#).

## Menggunakan Amazon Keyspaces dengan titik akhir VPC antarmuka

Endpoint VPC antarmuka memungkinkan komunikasi pribadi antara virtual private cloud (VPC) Anda yang berjalan di Amazon VPC dan Amazon Keyspaces. Endpoint VPC antarmuka didukung oleh AWS PrivateLink, yang merupakan AWS layanan yang memungkinkan komunikasi pribadi antara VPC dan layanan. AWS

AWS PrivateLink memungkinkan ini dengan menggunakan elastic network interface dengan alamat IP pribadi di VPC Anda sehingga lalu lintas jaringan tidak meninggalkan jaringan Amazon. VPC endpoint antarmuka tidak memerlukan gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct Connect . Untuk informasi selengkapnya, lihat [titik akhir Amazon Virtual Private Cloud dan Interface VPC](#) ().AWS PrivateLink

### Topik

- [Menggunakan titik akhir VPC antarmuka untuk Amazon Keyspaces](#)
- [Mengisi entri system.peers tabel dengan informasi titik akhir VPC antarmuka](#)
- [Mengontrol akses ke titik akhir VPC antarmuka untuk Amazon Keyspaces](#)
- [Ketersediaan](#)
- [Kebijakan titik akhir VPC dan pemulihan point-in-time \(PITR\) Amazon Keyspaces](#)
- [Kesalahan dan peringatan umum](#)

## Menggunakan titik akhir VPC antarmuka untuk Amazon Keyspaces

Anda dapat membuat titik akhir VPC antarmuka sehingga lalu lintas antara Amazon Keyspaces dan sumber daya Amazon VPC Anda mulai mengalir melalui titik akhir VPC antarmuka. Untuk memulai, ikuti langkah-langkah untuk [membuat titik akhir antarmuka](#). Selanjutnya, edit grup keamanan yang terkait dengan titik akhir yang Anda buat pada langkah sebelumnya, dan konfigurasi aturan masuk untuk port 9142. Untuk informasi selengkapnya, lihat [Menambahkan, menghapus, dan memperbarui aturan](#).

Untuk tutorial langkah demi langkah untuk mengonfigurasi koneksi ke Amazon Keyspaces melalui titik akhir VPC, lihat [the section called “Menghubungkan dengan titik akhir VPC”](#) Untuk mempelajari cara mengonfigurasi akses lintas akun untuk sumber daya Amazon Keyspaces yang terpisah dari aplikasi yang Akun AWS berbeda di VPC, lihat [the section called “Konfigurasi akses lintas akun”](#)

### Mengisi entri **system.peers** tabel dengan informasi titik akhir VPC antarmuka

Driver Apache Cassandra menggunakan `system.peers` tabel untuk meminta informasi node tentang cluster. Driver Cassandra menggunakan informasi node untuk memuat koneksi keseimbangan dan mencoba lagi operasi. Amazon Keyspaces mengisi sembilan entri dalam `system.peers` tabel secara otomatis untuk klien yang terhubung melalui titik akhir publik.

Untuk menyediakan klien yang terhubung melalui titik akhir VPC antarmuka dengan fungsionalitas serupa, Amazon Keyspaces mengisi `system.peers` tabel di akun Anda dengan entri untuk setiap Availability Zone tempat titik akhir VPC tersedia. Untuk mencari dan menyimpan titik akhir VPC antarmuka yang tersedia dalam tabel `system.peers`, Amazon Keyspaces mengharuskan Anda memberikan entitas IAM yang digunakan untuk menyambung ke izin akses Amazon Keyspaces untuk menanyakan VPC Anda untuk informasi titik akhir dan antarmuka jaringan.

#### Important

Mengisi `system.peers` tabel dengan titik akhir VPC antarmuka yang tersedia meningkatkan penyeimbangan beban dan meningkatkan throughput. read/write Disarankan untuk semua klien yang mengakses Amazon Keyspaces menggunakan titik akhir VPC antarmuka dan diperlukan untuk Apache Spark.

Untuk memberikan entitas IAM yang digunakan untuk menyambung ke izin Amazon Keyspaces untuk mencari informasi titik akhir VPC antarmuka yang diperlukan, Anda dapat memperbarui peran IAM

atau kebijakan pengguna yang ada, atau membuat kebijakan IAM baru seperti yang ditunjukkan pada contoh berikut.

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

#### Note

Kebijakan terkelola `AmazonKeyspacesReadOnlyAccess_v2` dan `AmazonKeyspacesFullAccess` menyertakan izin yang diperlukan agar Amazon Keyspaces mengakses instans Amazon EC2 untuk membaca informasi tentang titik akhir VPC antarmuka yang tersedia.

Untuk mengonfirmasi bahwa kebijakan telah disiapkan dengan benar, kueri `system.peers` tabel untuk melihat informasi jaringan. Jika `system.peers` tabel kosong, ini dapat menunjukkan bahwa kebijakan belum berhasil dikonfigurasi atau bahwa Anda telah melampaui kuota tingkat permintaan untuk `DescribeNetworkInterfaces` dan tindakan `DescribeVPCEndpoints` API. `DescribeVPCEndpoint` termasuk dalam `Describe*` kategori dan dianggap sebagai tindakan non-mutasi. `DescribeNetworkInterfaces` jatuh ke dalam subset tindakan non-mutasi tanpa filter dan tanpa paginasi, dan kuota yang berbeda berlaku. Untuk informasi selengkapnya, lihat [Meminta ukuran bucket token dan tarif isi ulang di Referensi](#) API Amazon EC2.

Jika Anda melihat tabel kosong, coba lagi beberapa menit kemudian untuk mengesampingkan masalah kuota tingkat permintaan. Untuk memverifikasi bahwa Anda telah mengonfigurasi titik akhir VPC dengan benar, lihat [the section called “Kesalahan koneksi titik akhir VPC”](#) Jika kueri Anda menampilkan hasil dari tabel, kebijakan Anda telah dikonfigurasi dengan benar.

## Mengontrol akses ke titik akhir VPC antarmuka untuk Amazon Keyspaces

Dengan kebijakan titik akhir VPC, Anda dapat mengontrol akses ke sumber daya dengan dua cara:

- Kebijakan IAM — Anda dapat mengontrol permintaan, pengguna, atau grup yang diizinkan mengakses Amazon Keyspaces melalui titik akhir VPC tertentu. Anda dapat melakukannya dengan menggunakan [kunci kondisi](#) dalam kebijakan yang dilampirkan ke pengguna, grup, atau peran IAM.
- Kebijakan VPC — Anda dapat mengontrol titik akhir VPC mana yang memiliki akses ke sumber daya Amazon Keyspaces Anda dengan melampirkan kebijakan padanya. Untuk membatasi akses ke ruang kunci atau tabel tertentu agar hanya mengizinkan lalu lintas masuk melalui titik akhir VPC tertentu, edit kebijakan IAM yang ada yang membatasi akses sumber daya dan menambahkan titik akhir VPC tersebut.

Berikut ini adalah contoh kebijakan titik akhir untuk mengakses sumber daya Amazon Keyspaces.

- Contoh kebijakan IAM: Batasi semua akses ke tabel Amazon Keyspaces tertentu kecuali lalu lintas berasal dari titik akhir VPC yang ditentukan — Kebijakan sampel ini dapat dilampirkan ke pengguna, peran, atau grup IAM. Ini membatasi akses ke tabel Amazon Keyspaces tertentu kecuali lalu lintas masuk berasal dari titik akhir VPC tertentu.

```
{
 "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**

Untuk membatasi akses ke tabel tertentu, Anda juga harus menyertakan akses ke tabel sistem. Tabel sistem hanya-baca.

- Contoh kebijakan VPC: Read-only akses — Kebijakan sampel ini dapat dilampirkan ke titik akhir VPC. (Untuk informasi selengkapnya, lihat [Mengontrol akses ke sumber daya Amazon VPC](#)). Ini membatasi tindakan untuk akses hanya-baca ke sumber daya Amazon Keyspaces melalui titik akhir VPC yang dilampirkan.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ReadOnly",
 "Principal": "*",
 "Action": [
 "cassandra:Select"
],
 "Effect": "Allow",
 "Resource": "*"
 }
]
}
```

- Contoh kebijakan VPC: Batasi akses ke tabel Amazon Keyspaces tertentu — Kebijakan contoh ini dapat dilampirkan ke titik akhir VPC. Ini membatasi akses ke tabel tertentu melalui titik akhir VPC yang dilampirkan.

```
{
 "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

Untuk membatasi akses ke tabel tertentu, Anda juga harus menyertakan akses ke tabel sistem. Tabel sistem hanya-baca.

## Ketersediaan

Amazon Keyspaces mendukung penggunaan titik akhir VPC antarmuka di semua Wilayah AWS tempat layanan tersedia. Untuk informasi selengkapnya, lihat [???](#).

## Kebijakan titik akhir VPC dan pemulihan point-in-time (PITR) Amazon Keyspaces

Jika Anda menggunakan kebijakan IAM dengan [kunci kondisi](#) untuk membatasi lalu lintas masuk, operasi pemulihan tabel mungkin gagal. Misalnya, jika Anda membatasi lalu lintas sumber ke titik akhir VPC tertentu `aws:SourceVpce` menggunakan tombol kondisi, operasi pemulihan tabel gagal. Untuk mengizinkan Amazon Keyspaces melakukan operasi pemulihan atas nama kepala sekolah, Anda harus menambahkan kunci `aws:ViaAWSService` kondisi ke kebijakan IAM Anda. Kunci `aws:ViaAWSService` kondisi memungkinkan akses ketika AWS layanan apa pun membuat permintaan menggunakan kredensi kepala sekolah. Untuk informasi selengkapnya, lihat [elemen kebijakan IAM JSON: Kunci kondisi](#) di Panduan Pengguna IAM. Kebijakan berikut adalah contoh dari ini.

```

{
 "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"
 }
 }
 }
]
}

```

## Kesalahan dan peringatan umum

Jika Anda menggunakan Amazon Virtual Private Cloud dan Anda terhubung ke Amazon Keyspaces, Anda mungkin melihat peringatan berikut.

```

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.

```

Peringatan ini terjadi karena `system.peers` tabel berisi entri untuk semua titik akhir VPC Amazon yang memiliki izin untuk dilihat oleh Amazon Keyspaces, termasuk titik akhir Amazon VPC yang Anda sambungkan. Anda dapat dengan aman mengabaikan peringatan ini.

Untuk kesalahan lainnya, lihat [the section called “Kesalahan koneksi titik akhir VPC”](#).

## Menggunakan Amazon Keyspaces CDC stream dengan antarmuka VPC endpoint

Endpoint VPC antarmuka memungkinkan komunikasi pribadi antara virtual private cloud (VPC) Anda yang berjalan di Amazon VPC dan Amazon Keyspaces. Endpoint VPC antarmuka didukung oleh AWS PrivateLink, yang merupakan AWS layanan yang memungkinkan komunikasi pribadi antara VPC dan layanan. AWS

AWS PrivateLink memungkinkan ini dengan menggunakan elastic network interface dengan alamat IP pribadi di VPC Anda sehingga lalu lintas jaringan tidak meninggalkan jaringan Amazon. VPC endpoint antarmuka tidak memerlukan gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct Connect . Untuk informasi selengkapnya, lihat [titik akhir Amazon Virtual Private Cloud dan Interface VPC](#) ().AWS PrivateLink

### Topik

- [Menggunakan titik akhir VPC antarmuka untuk aliran CDC Amazon Keyspaces](#)
- [Amazon Keyspaces CDC mengalirkan antarmuka titik akhir VPC](#)
- [Buat Amazon Keyspaces CDC stream antarmuka titik akhir VPC](#)
- [Memperbarui titik akhir VPC antarmuka streaming Amazon Keyspaces CDC](#)
- [Daftar aliran menggunakan antarmuka aliran Amazon Keyspaces CDC titik akhir VPC](#)
- [Membuat kebijakan untuk titik akhir VPC antarmuka streaming Amazon Keyspaces CDC](#)

## Menggunakan titik akhir VPC antarmuka untuk aliran CDC Amazon Keyspaces

Anda dapat menggunakan titik akhir VPC antarmuka sehingga lalu lintas antara aliran CDC Amazon Keyspaces dan sumber daya Amazon VPC Anda mulai mengalir melalui titik akhir VPC antarmuka. Anda dapat menggunakan kebijakan titik akhir VPC untuk membatasi akses ke aliran CDC Anda.

Untuk informasi selengkapnya tentang aliran CDC Amazon Keyspaces, lihat. [the section called “Bekerja dengan aliran CDC”](#)

## Amazon Keyspaces CDC mengalirkan antarmuka titik akhir VPC

Saat Anda membuat titik akhir antarmuka, Amazon Keyspaces CDC stream menghasilkan dua jenis nama DNS khusus titik akhir untuk aliran: Regional dan Zonal.



## Regional

Nama DNS Regional mencakup informasi berikut:

- ID titik akhir VPC Amazon yang unik
- pengidentifikasi layanan
- yang Wilayah AWS
- `vpce.amazonaws.com`akhiran

Untuk titik akhir VPC Amazon dengan ID `vpce-1a2b3c4d`, nama DNS yang dihasilkan mungkin terlihat mirip dengan contoh berikut: `vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com`

## Zona

Nama DNS Zonal mencakup [Availability Zone](#) selain informasi dalam nama DNS Regional. Nama DNS yang dihasilkan untuk titik akhir VPC Amazon dengan `vpce-1a2b3c4d` ID akan terlihat seperti pada contoh berikut, perhatikan bahwa sekarang menyertakan Availability Wilayah AWS Zone: `vpce-1a2b3c4d-5e6f-us-east-1a.cassandra-streams.us-east-1.vpce.amazonaws.com`

Anda dapat menggunakan opsi ini jika arsitektur Anda mengisolasi Availability Zones. Contoh, Anda bisa menggunakannya untuk kontainer kesalahan atau untuk mengurangi biaya transfer data Regional.

### Note

Untuk mencapai keandalan yang optimal, sebaiknya gunakan layanan Anda di minimal tiga Availability Zone.

## Buat Amazon Keyspaces CDC stream antarmuka titik akhir VPC

Anda dapat menggunakan AWS CLI atau AWS SDK untuk mengakses operasi Amazon Keyspaces CDC Streams API melalui titik akhir antarmuka Amazon Keyspaces CDC Streams. Untuk daftar lengkap semua operasi API yang tersedia, lihat Referensi API [Amazon Keyspaces Streams](#).

Untuk informasi selengkapnya tentang cara membuat titik akhir VPC, lihat [membuat titik akhir antarmuka di Panduan Pengguna](#) Amazon VPC.

Untuk membuat titik akhir VPC, Anda dapat menggunakan sintaks dalam contoh berikut.

```
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
```

## Memperbarui titik akhir VPC antarmuka streaming Amazon Keyspaces CDC

Untuk memperbarui titik akhir VPC, Anda dapat menggunakan sintaks dalam contoh berikut.

```
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
```

## Daftar aliran menggunakan antarmuka aliran Amazon Keyspaces CDC titik akhir VPC

Untuk membuat daftar aliran yang menggunakan titik akhir VPC, Anda dapat menggunakan sintaks dalam contoh berikut. Pastikan untuk mengganti Region dan nama DNS dari ID endpoint VPC dengan informasi Anda sendiri.

```
aws keyspacesstreams \
 --endpoint https://vpce-1a2b3c4d-5e6f.cassandra-streams.us-east-1.vpce.amazonaws.com \
 --region us-east-1 \
 list-streams
```

## Membuat kebijakan untuk titik akhir VPC antarmuka streaming Amazon Keyspaces CDC

Anda dapat melampirkan kebijakan titik akhir ke titik akhir VPC Amazon yang mengontrol akses ke aliran CDC Amazon Keyspaces. Kebijakan titik akhir menentukan informasi berikut:

- Prinsip AWS Identity and Access Management (IAM) yang dapat melakukan tindakan
- Tindakan-tindakan yang dapat dilakukan

- Sumber daya yang padanya tindakan dapat dilakukan

Untuk membatasi akses ke aliran CDC Amazon Keyspaces tertentu agar hanya mengizinkan AWS layanan tertentu di akses VPC Amazon Anda, Anda dapat menggunakan contoh berikut.

Kebijakan aliran berikut memberikan akses ke setiap prinsipal IAM untuk tindakan `cassandra:GetStream` dan `cassandra:GetRecords` untuk aliran tertentu yang 2025-02-20T11:22:33.444 dilampirkan ke sumber daya `/keyspace/mykeyspace/table/mytable/` milik akun. 123456788901 Untuk menggunakan kebijakan endpoint ini, pastikan untuk mengganti Region, ID akun, dan resource dengan label stream.

```
{
 "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 tidak mendukung titik akhir Gateway untuk aliran CDC.

## Analisis konfigurasi dan kerentanan untuk Amazon Keyspaces

AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patch database, konfigurasi firewall, dan pemulihan bencana. Prosedur ini telah ditinjau dan disertifikasi oleh pihak ketiga yang sesuai. Untuk detail selengkapnya, lihat sumber daya berikut:

- [Model tanggung jawab bersama Model](#)

- [Amazon Web Services: Gambaran umum proses keamanan](#)(laporan resmi)

## Praktik terbaik keamanan untuk Amazon Keyspaces

Amazon Keyspaces (untuk Apache Cassandra) menyediakan sejumlah fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau tidak memadai untuk lingkungan Anda, perlakukan itu sebagai pertimbangan yang bermanfaat, bukan sebagai resep.

### Topik

- [Praktik terbaik keamanan preventif untuk Amazon Keyspaces](#)
- [Praktik terbaik keamanan Detektif untuk Amazon Keyspaces](#)

## Praktik terbaik keamanan preventif untuk Amazon Keyspaces

Praktik terbaik keamanan berikut dianggap preventif karena dapat membantu Anda mengantisipasi dan mencegah insiden keamanan di Amazon Keyspaces.

### Gunakan enkripsi saat istirahat

[Amazon Keyspaces mengenkripsi semua data pengguna yang disimpan dalam tabel dengan menggunakan kunci enkripsi yang disimpan di \(.AWS Key Management ServiceAWS KMS](#) Hal ini memberi lapisan perlindungan data tambahan dengan mengamankan data Anda dari akses yang tidak sah ke penyimpanan dasar.

Secara default, Amazon Keyspaces menggunakan file Kunci milik AWS untuk mengenkripsi semua tabel Anda. Jika kunci ini tidak ada, itu dibuat untuk Anda. Kunci default layanan tidak dapat dinonaktifkan.

Atau, Anda dapat menggunakan [kunci yang dikelola pelanggan](#) untuk enkripsi saat istirahat. Untuk informasi selengkapnya, lihat [Enkripsi Amazon Keyspaces saat Istirahat](#).

### Menggunakan peran IAM untuk mengautentikasi akses ke Amazon Keyspaces

Agar pengguna, aplikasi, dan AWS layanan lain dapat mengakses Amazon Keyspaces, mereka harus menyertakan AWS kredensi yang valid dalam permintaan API mereka. AWS Anda tidak boleh menyimpan AWS kredensial secara langsung di aplikasi atau instans EC2. Ini adalah kredensial jangka panjang yang tidak dirotasi secara otomatis, dan karenanya dapat menimbulkan

dampak bisnis yang signifikan jika dibobol. Peran IAM memungkinkan Anda memperoleh kunci akses sementara yang dapat digunakan untuk mengakses layanan dan sumber daya AWS .

Untuk informasi lebih lanjut, lihat [Peran IAM](#).

### Menggunakan kebijakan IAM untuk otorisasi dasar Amazon Keyspaces

Saat memberikan izin, Anda memutuskan siapa yang mendapatkannya, API Amazon Keyspaces mana yang mereka dapatkan izin, dan tindakan spesifik yang ingin Anda izinkan pada sumber daya tersebut. Menerapkan hak istimewa paling sedikit adalah kunci dalam mengurangi risiko keamanan dan dampak yang dapat dihasilkan dari kesalahan atau niat jahat.

Lampirkan kebijakan izin ke identitas IAM (yaitu, pengguna, grup, dan peran) dan dengan demikian memberikan izin untuk melakukan operasi di sumber daya Amazon Keyspaces.

Anda dapat melakukan hal ini dengan cara berikut:

- [AWS kebijakan terkelola \(standar\)](#)
- [Kebijakan yang dikelola pelanggan](#)

### Menggunakan ketentuan kebijakan IAM untuk kontrol akses ketat

Bila Anda memberikan izin di Amazon Keyspaces, Anda dapat menentukan kondisi yang menentukan bagaimana kebijakan izin diterapkan. Menerapkan hak istimewa paling sedikit adalah kunci dalam mengurangi risiko keamanan dan dampak yang dapat dihasilkan dari kesalahan atau niat jahat.

Anda dapat menetapkan syarat saat memberikan izin menggunakan kebijakan IAM. Misalnya, Anda dapat melakukan hal berikut:

- Berikan izin untuk memungkinkan pengguna akses hanya-baca ke ruang kunci atau tabel tertentu.
- Berikan izin untuk mengizinkan pengguna menulis akses ke tabel tertentu, berdasarkan identitas pengguna tersebut.

Untuk informasi selengkapnya, lihat [Contoh Identity-Based Kebijakan](#).

### Pertimbangkan enkripsi di sisi klien

Jika Anda menyimpan data sensitif atau rahasia di Amazon Keyspaces, Anda mungkin ingin mengenkripsi data tersebut sedekat mungkin dengan asalnya sehingga data Anda terlindungi sepanjang siklus hidupnya. Mengenkripsi data bergerak dan data diam Anda yang sensitif membantu memastikan bahwa data plaintext Anda tidak tersedia untuk pihak ketiga mana pun.

## Praktik terbaik keamanan Detektif untuk Amazon Keyspaces

Praktik terbaik keamanan berikut dianggap detektif karena dapat membantu Anda mendeteksi potensi kelemahan dan insiden keamanan.

Gunakan AWS CloudTrail untuk memantau AWS Key Management Service (AWS KMS) penggunaan AWS KMS kunci

Jika Anda menggunakan [AWS KMS kunci terkelola pelanggan](#) untuk enkripsi saat istirahat, penggunaan kunci ini masuk AWS CloudTrail. CloudTrail memberikan visibilitas ke aktivitas pengguna dengan merekam tindakan yang diambil pada akun Anda. CloudTrail mencatat informasi penting tentang setiap tindakan, termasuk siapa yang membuat permintaan, layanan yang digunakan, tindakan yang dilakukan, parameter untuk tindakan, dan elemen respons yang dikembalikan oleh AWS layanan. Informasi ini membantu Anda melacak perubahan yang dibuat pada AWS sumber daya Anda dan memecahkan masalah operasional. CloudTrail membuatnya lebih mudah untuk memastikan kepatuhan terhadap kebijakan internal dan standar peraturan.

Anda dapat menggunakan CloudTrail untuk mengaudit penggunaan kunci. CloudTrail membuat file log yang berisi riwayat panggilan AWS API dan peristiwa terkait untuk akun Anda. File log ini mencakup semua permintaan AWS KMS API yang dibuat menggunakan konsol, AWS SDK, dan alat baris perintah, selain yang dibuat melalui AWS layanan terintegrasi. Anda dapat menggunakan file log ini untuk mendapatkan informasi tentang kapan AWS KMS kunci digunakan, operasi yang diminta, identitas pemohon, alamat IP tempat permintaan itu berasal, dan sebagainya. Untuk informasi selengkapnya, lihat [Pencatatan Panggilan API AWS Key Management Service dengan AWS CloudTrail](#) dan [Panduan Pengguna AWS CloudTrail](#).

Gunakan CloudTrail untuk memantau operasi bahasa definisi data (DDL) Amazon Keyspaces

CloudTrail memberikan visibilitas ke aktivitas pengguna dengan merekam tindakan yang diambil pada akun Anda. CloudTrail mencatat informasi penting tentang setiap tindakan, termasuk siapa yang membuat permintaan, layanan yang digunakan, tindakan yang dilakukan, parameter untuk tindakan, dan elemen respons yang dikembalikan oleh AWS layanan. Informasi ini membantu Anda melacak perubahan yang dilakukan pada AWS sumber daya Anda dan untuk memecahkan masalah operasional. CloudTrail membuatnya lebih mudah untuk memastikan kepatuhan terhadap kebijakan internal dan standar peraturan.

Semua [operasi Amazon Keyspaces DDL](#) masuk secara otomatis. CloudTrail Operasi DDL memungkinkan Anda membuat dan mengelola ruang kunci dan tabel Amazon Keyspaces.

Saat aktivitas terjadi di Amazon Keyspaces, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat acara. Untuk informasi selengkapnya, lihat [Mencatat operasi Amazon Keyspaces dengan menggunakan](#). AWS CloudTrail Anda dapat melihat, mencari, dan mengunduh acara terbaru di situs Anda Akun AWS. Untuk informasi selengkapnya, lihat [Melihat CloudTrail peristiwa dengan riwayat peristiwa](#) di Panduan AWS CloudTrail Pengguna.

[Untuk catatan peristiwa yang sedang berlangsung di Anda Akun AWS, termasuk acara untuk Amazon Keyspaces, buat jejak.](#) Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon Simple Storage Service (Amazon S3). Secara default, saat Anda membuat jejak di konsol, jejak berlaku untuk semua Wilayah AWS. Jejak mencatat peristiwa dari semua Wilayah di partisi AWS dan mengirimkan file log ke bucket S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log.

Tandai sumber daya Amazon Keyspaces Anda untuk identifikasi dan otomatisasi

Anda dapat menetapkan metadata ke AWS sumber daya Anda dalam bentuk tag. Setiap tag adalah label sederhana yang terdiri dari kunci yang ditentukan pelanggan dan nilai opsional yang dapat membuatnya lebih mudah untuk mengelola, mencari, dan memfilter sumber daya.

Penandaan memungkinkan implementasi kontrol berkelompok. Meskipun tidak ada jenis tanda yang melekat, tanda memungkinkan Anda untuk mengelompokkan sumber daya berdasarkan tujuan, pemilik, lingkungan, atau kriteria lainnya. Berikut ini beberapa contohnya:

- Akses - Digunakan untuk mengontrol akses ke sumber daya Amazon Keyspaces berdasarkan tag. Untuk informasi selengkapnya, lihat [the section called “Otorisasi berdasarkan tag Amazon Keyspaces”](#).
- Keamanan — Digunakan untuk menentukan persyaratan seperti pengaturan perlindungan data.
- Kerahasiaan — Pengidentifikasi untuk tingkat kerahasiaan data tertentu yang didukung sumber daya.
- Lingkungan – Digunakan untuk membedakan antara pengembangan, pengujian, dan infrastruktur produksi.

Untuk informasi selengkapnya, lihat [strategi AWS penandaan](#) dan [Menambahkan tag dan label ke sumber daya](#).

# Referensi bahasa CQL untuk Amazon Keyspaces (untuk Apache Cassandra)

Setelah Anda terhubung ke endpoint Amazon Keyspaces, Anda menggunakan Cassandra Query Language (CQL) untuk bekerja dengan database Anda. CQL mirip dalam banyak hal dengan Structured Query Language (SQL).

- **Elemen CQL** - Bagian ini mencakup elemen dasar CQL yang didukung di Amazon Keyspaces, termasuk pengidentifikasi, konstanta, istilah, dan tipe data. Ini menjelaskan konsep seperti tipe string, tipe numerik, tipe koleksi, dan banyak lagi.
- **Data Definition Language (DDL)** — Pernyataan DDL digunakan untuk mengelola struktur data seperti keyspaces dan tabel di Amazon Keyspaces. Bagian ini mencakup pernyataan untuk membuat, mengubah, dan menjatuhkan ruang kunci dan tabel, serta memulihkan tabel dari cadangan point-in-time.
- **Data Manipulation Language (DML)** — Pernyataan DML digunakan untuk mengelola data dalam tabel. Bagian ini mencakup pernyataan untuk memilih, menyisipkan, memperbarui, dan menghapus data. Ini juga menjelaskan kemampuan kueri tingkat lanjut seperti menggunakan IN operator, memesan hasil, dan pagination.
- **Built-in fungsi** - Amazon Keyspaces mendukung berbagai fungsi skalar bawaan yang dapat Anda gunakan dalam pernyataan CQL. Bagian ini memberikan gambaran umum tentang fungsi-fungsi ini, termasuk contoh penggunaannya.

Sepanjang topik ini, Anda akan menemukan sintaks terperinci, contoh, dan praktik terbaik untuk menggunakan CQL secara efektif di Amazon Keyspaces.

## Topik

- [Elemen Cassandra Query Language \(CQL\) di Amazon Keyspaces](#)
- [Pernyataan DDL \(bahasa definisi data\) di Amazon Keyspaces](#)
- [Pernyataan DHTML \(bahasa manipulasi data\) di Amazon Keyspaces](#)
- [Built-in fungsi di Amazon Keyspaces](#)



# Elemen Cassandra Query Language (CQL) di Amazon Keyspaces

Pelajari tentang elemen Cassandra Query Language (CQL) yang didukung oleh Amazon Keyspaces, termasuk pengidentifikasi, konstanta, istilah, dan tipe data.

Topik

- [Pengidentifikasi](#)
- [Konstanta](#)
- [Istilah](#)
- [Jenis Data](#)
- [Pengkodean JSON dari tipe data Amazon Keyspaces](#)

## Pengidentifikasi

Pengidentifikasi (atau nama) digunakan untuk mengidentifikasi tabel, kolom, dan objek lainnya. Pengenal dapat dikutip atau tidak dikutip. Berikut ini berlaku.

```
identifier ::= unquoted_identifier | quoted_identifier
unquoted_identifier ::= re('[a-zA-Z][a-zA-Z0-9_]')
quoted_identifier ::= ''' (any character where " can appear if doubled)+ '''
```

## Konstanta

Konstanta berikut didefinisikan.

```
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+
```

## Istilah

Sebuah istilah menunjukkan jenis nilai yang didukung. Ketentuan didefinisikan oleh yang berikut ini.

```

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

```

## Jenis Data

Amazon Keyspaces mendukung tipe data berikut:

### Jenis string

| Jenis data           | Deskripsi                                                                             |
|----------------------|---------------------------------------------------------------------------------------|
| <code>ascii</code>   | Merupakan string karakter ASCII.                                                      |
| <code>text</code>    | Merupakan string yang UTF-8 dikodekan.                                                |
| <code>varchar</code> | Merupakan string yang UTF-8 dikodekan (varchar adalah alias untuk). <code>text</code> |

### Jenis numerik

| Jenis data           | Deskripsi                                                                                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <code>bigint</code>  | Merupakan panjang bertanda 64-bit.                                                                                                         |
| <code>counter</code> | Merupakan penghitung bilangan bulat bertanda 64-bit. Untuk informasi selengkapnya, lihat <a href="#">the section called “Penghitung”</a> . |
| <code>decimal</code> | Merupakan desimal presisi variabel.                                                                                                        |
| <code>double</code>  | Merupakan titik mengambang IEEE 754 64-bit.                                                                                                |
| <code>float</code>   | Merupakan titik mengambang IEEE 754 32-bit.                                                                                                |
| <code>int</code>     | Merupakan int bertanda 32-bit.                                                                                                             |

| Jenis data          | Deskripsi                                             |
|---------------------|-------------------------------------------------------|
| <code>varint</code> | Merupakan nilai integer dalam rentang $\pm 10^{38}$ . |

## Penghitung

`counter` Kolom berisi integer bertanda 64-bit. Nilai penghitung bertambah atau dikurangi menggunakan [the section called “UPDATE”](#) pernyataan, dan tidak dapat diatur secara langsung. Ini membuat `counter` kolom berguna untuk melacak jumlah. Misalnya, Anda dapat menggunakan penghitung untuk melacak jumlah entri dalam file log atau berapa kali posting telah dilihat di jejaring sosial. Pembatasan berikut berlaku untuk `counter` kolom:

- Kolom tipe `counter` tidak dapat menjadi bagian `primary key` dari tabel.
- Dalam tabel yang berisi satu atau beberapa kolom tipe `counter`, semua kolom dalam tabel itu harus bertipe `counter`.

Dalam kasus di mana pembaruan penghitung gagal (misalnya, karena batas waktu atau kehilangan koneksi dengan Amazon Keyspaces), klien tidak tahu apakah nilai penghitung telah diperbarui. Jika pembaruan dicoba lagi, pembaruan ke nilai penghitung mungkin diterapkan untuk kedua kalinya.

## Jenis gumpalan

| Jenis data        | Deskripsi                |
|-------------------|--------------------------|
| <code>blob</code> | Merupakan byte arbitrer. |

## Jenis Boolean

| Jenis data           | Deskripsi                                            |
|----------------------|------------------------------------------------------|
| <code>boolean</code> | Mewakili <code>true</code> atau <code>false</code> . |

## Time-related jenis

| Jenis data | Deskripsi                                                                                                                |
|------------|--------------------------------------------------------------------------------------------------------------------------|
| date       | String dalam format<yyyy>-<mm>-<dd> .                                                                                    |
| timestamp  | Integer bertanda 64-bit yang mewakili tanggal dan waktu sejak epoch (1 Januari 1970 pukul 00:00:00 GMT) dalam milidetik. |
| timeuuid   | Merupakan <a href="#">versi 1 UUID</a> .                                                                                 |

## Jenis koleksi

| Jenis data | Deskripsi                                                                   |
|------------|-----------------------------------------------------------------------------|
| list       | Merupakan kumpulan elemen literal yang diurutkan.                           |
| map        | Merupakan kumpulan pasangan kunci-nilai yang tidak berurutan.               |
| set        | Merupakan koleksi yang tidak berurutan dari satu atau lebih elemen literal. |

Anda mendeklarasikan kolom koleksi dengan menggunakan tipe koleksi diikuti oleh tipe data lain (misalnya, TEXT atau INT) dalam tanda kurung miring. Anda dapat membuat kolom dengan SET dari TEXT, atau Anda dapat membuat pasangan MAP dari TEXT dan INT kunci-nilai, seperti yang ditunjukkan pada contoh berikut.

```
SET <TEXT>
MAP <TEXT, INT>
```

Koleksi non-beku memungkinkan Anda untuk membuat pembaruan untuk setiap elemen koleksi individu. Client-side stempel waktu dan pengaturan Time to Live (TTL) disimpan untuk masing-masing elemen.

Saat Anda menggunakan FROZEN kata kunci pada jenis koleksi, nilai koleksi diserialisasikan menjadi satu nilai yang tidak dapat diubah, dan Amazon Keyspaces memperlakukannya seperti a. BLOB Ini adalah koleksi beku. UPDATE Pernyataan INSERT atau menimpa seluruh koleksi beku. Anda tidak dapat membuat pembaruan untuk elemen individual di dalam koleksi beku.

Client-side stempel waktu dan pengaturan Time to Live (TTL) berlaku untuk seluruh koleksi beku, bukan untuk elemen individual. Frozen kolom koleksi dapat menjadi bagian PRIMARY KEY dari tabel.

Anda dapat membuat sarang koleksi beku. Misalnya, Anda dapat menentukan MAP dalam a SET jika menggunakan FROZEN kata kunci, seperti yang ditunjukkan pada contoh berikut. MAP

```
SET <FROZEN> <MAP <TEXT, INT>>>
```

Amazon Keyspaces mendukung penyarangan hingga 8 tingkat koleksi beku secara default. Untuk informasi selengkapnya, lihat [the section called “Kuota layanan Amazon Keyspaces”](#). Untuk informasi lebih lanjut tentang perbedaan fungsional dengan Apache Cassandra, lihat. [the section called “FROZEN koleksi”](#) Untuk informasi selengkapnya tentang sintaks CQL, lihat dan. [the section called “CREATE TABLE”](#) [the section called “ALTER TABLE”](#)

## Tipe Tuple

Tipe tuple data mewakili sekelompok elemen literal yang dibatasi. Anda dapat menggunakan tupel sebagai alternatif untuk a user defined type. Anda tidak perlu menggunakan FROZEN kata kunci untuk tupel. Ini karena Tuple selalu dibekukan dan Anda tidak dapat memperbarui elemen satu per satu.

## Tipe lainnya

| Jenis data | Deskripsi                                                         |
|------------|-------------------------------------------------------------------|
| inet       | String yang mewakili alamat IP, baik dalam format IPv4 atau IPv6. |

## Statis

Dalam tabel Amazon Keyspaces dengan kolom pengelompokan, Anda dapat menggunakan STATIC kata kunci untuk membuat kolom statis jenis apa pun.

Pernyataan berikut adalah contoh dari ini.

```
my_column INT STATIC
```

Untuk informasi selengkapnya tentang bekerja dengan kolom statis, lihat [the section called “Perkirakan konsumsi kapasitas kolom statis”](#).

## User-defined jenis (UDT)

Amazon Keyspaces mendukung tipe yang ditentukan pengguna (UDT). Anda dapat menggunakan tipe data Amazon Keyspaces yang valid untuk membuat UDT, termasuk koleksi dan UDT lain yang ada. Anda membuat UDT di ruang kunci dan dapat menggunakannya untuk menentukan kolom dalam tabel apa pun di ruang kunci.

Untuk informasi selengkapnya tentang sintaks CQL, lihat [the section called “Type”](#). Untuk informasi selengkapnya tentang bekerja dengan UDT, lihat [the section called “User-defined jenis \(UDT\)”](#).

Untuk meninjau berapa banyak UDT yang didukung per ruang kunci, tingkat penyarangan yang didukung, dan nilai dan kuota default lainnya yang terkait dengan UDT, lihat [the section called “Kuota dan nilai default untuk tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces”](#)

## Pengkodean JSON dari tipe data Amazon Keyspaces

Amazon Keyspaces menawarkan pemetaan tipe data JSON yang sama dengan Apache Cassandra. Tabel berikut menjelaskan tipe data yang diterima Amazon Keyspaces dalam `INSERT JSON` pernyataan dan tipe data yang digunakan Amazon Keyspaces saat mengembalikan data dengan pernyataan `SELECT JSON`

Untuk tipe data bidang tunggal seperti `float`, dan `int` `UUID`, Anda juga dapat menyisipkan data sebagai `string` file. Untuk tipe dan koleksi data gabungan, seperti, dan `tuple` `map`, Anda juga dapat menyisipkan data sebagai JSON atau sebagai JSON `string` encode.

| Jenis data JSON    | Tipe data diterima dalam <b>INSERT JSON</b> pernyataan | Tipe data dikembalikan dalam <b>SELECT JSON</b> pernyataan | Catatan                                             |
|--------------------|--------------------------------------------------------|------------------------------------------------------------|-----------------------------------------------------|
| <code>ascii</code> | <code>string</code>                                    | <code>string</code>                                        | Menggunakan pelarian <code>\u</code> karakter JSON. |

| Jenis data JSON      | Tipe data diterima dalam <b>INSERT JSON</b> pernyataan | Tipe data dikembalikan dalam <b>SELECT JSON</b> pernyataan | Catatan                                                                                     |
|----------------------|--------------------------------------------------------|------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| <code>bigint</code>  | <code>integer, string</code>                           | <code>integer</code>                                       | String harus berupa integer 64-bit yang valid.                                              |
| <code>blob</code>    | <code>string</code>                                    | <code>string</code>                                        | String harus dimulai dengan <code>0x</code> diikuti oleh jumlah digit hex genap.            |
| <code>boolean</code> | <code>boolean, string</code>                           | <code>boolean</code>                                       | String harus salah satu <code>true</code> atau <code>false</code> .                         |
| <code>date</code>    | <code>string</code>                                    | <code>string</code>                                        | Tanggal dalam format <code>YYYY-MM-DD</code> , zona waktu UTC.                              |
| <code>decimal</code> | <code>integer, float, string</code>                    | <code>float</code>                                         | Dapat melebihi presisi IEEE-754 floating point 32-bit atau 64-bit dalam decoder sisi klien. |
| <code>double</code>  | <code>integer, float, string</code>                    | <code>float</code>                                         | String harus berupa integer atau float yang valid.                                          |
| <code>float</code>   | <code>integer, float, string</code>                    | <code>float</code>                                         | String harus berupa integer atau float yang valid.                                          |
| <code>inet</code>    | <code>string</code>                                    | <code>string</code>                                        | Alamat IPv4 atau IPv6.                                                                      |

| Jenis data JSON       | Tipe data diterima dalam <b>INSERT JSON</b> pernyataan | Tipe data dikembalikan dalam <b>SELECT JSON</b> pernyataan | Catatan                                               |
|-----------------------|--------------------------------------------------------|------------------------------------------------------------|-------------------------------------------------------|
| <code>int</code>      | <code>integer, string</code>                           | <code>integer</code>                                       | String harus berupa bilangan bulat 32-bit yang valid. |
| <code>list</code>     | <code>list, string</code>                              | <code>list</code>                                          | Menggunakan representasi daftar JSON asli.            |
| <code>map</code>      | <code>map, string</code>                               | <code>map</code>                                           | Menggunakan representasi peta JSON asli.              |
| <code>smallint</code> | <code>integer, string</code>                           | <code>integer</code>                                       | String harus berupa bilangan bulat 16-bit yang valid. |
| <code>set</code>      | <code>list, string</code>                              | <code>list</code>                                          | Menggunakan representasi daftar JSON asli.            |
| <code>text</code>     | <code>string</code>                                    | <code>string</code>                                        | Menggunakan pelarian \u karakter JSON.                |
| <code>time</code>     | <code>string</code>                                    | <code>string</code>                                        | Waktu dalam formatHH-MM-SS[.fffffffffff] .            |



| Jenis data JSON | Tipe data diterima dalam <b>INSERT JSON</b> pernyataan | Tipe data dikembalikan dalam <b>SELECT JSON</b> pernyataan | Catatan                                                                                                                                                                  |
|-----------------|--------------------------------------------------------|------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| timestamp       | integer, string                                        | string                                                     | Sebuah stempel waktu. Konstanta string memungkinkan Anda menyimpan stempel waktu sebagai tanggal. Perangkat tanggal dengan format YYYY-MM-DD HH:MM:SS. SSS dikembalikan. |
| timeuuid        | string                                                 | string                                                     | Tipe 1 UUID. Lihat <a href="#">constants</a> untuk format UUID.                                                                                                          |
| tinyint         | integer, string                                        | integer                                                    | String harus berupa bilangan bulat 8-bit yang valid.                                                                                                                     |
| tuple           | list, string                                           | list                                                       | Menggunakan representasi daftar JSON asli.                                                                                                                               |
| UDT             | map, string                                            | map                                                        | Menggunakan representasi peta JSON asli dengan nama bidang sebagai kunci.                                                                                                |
| uuid            | string                                                 | string                                                     | Lihat <a href="#">constants</a> untuk format UUID.                                                                                                                       |

| Jenis data JSON      | Tipe data diterima dalam <b>INSERT JSON</b> pernyataan | Tipe data dikembalikan dalam <b>SELECT JSON</b> pernyataan | Catatan                                                                                   |
|----------------------|--------------------------------------------------------|------------------------------------------------------------|-------------------------------------------------------------------------------------------|
| <code>varchar</code> | <code>string</code>                                    | <code>string</code>                                        | Menggunakan pelarian \u karakter JSON.                                                    |
| <code>varint</code>  | <code>integer, string</code>                           | <code>integer</code>                                       | Panjang variabel; mungkin meluap bilangan bulat 32-bit atau 64-bit di dekoder sisi klien. |

## Pernyataan DDL (bahasa definisi data) di Amazon Keyspaces

Bahasa definisi data (DDL) adalah kumpulan pernyataan Cassandra Query Language (CQL) yang Anda gunakan untuk mengelola struktur data di Amazon Keyspaces (untuk Apache Cassandra), seperti ruang kunci dan tabel. Anda menggunakan DDL untuk membuat struktur data ini, memodifikasinya setelah dibuat, dan menghapusnya saat tidak lagi digunakan. Amazon Keyspaces melakukan operasi DDL secara asinkron. Untuk informasi selengkapnya tentang cara mengonfirmasi bahwa operasi asinkron telah selesai, lihat. [the section called “Pembuatan asinkron dan penghapusan ruang kunci dan tabel”](#)

Pernyataan DDL berikut didukung:

- [BUAT KEYSPACE](#)
- [UBAH KEYSPACE](#)
- [JATUHKAN KEYSPACE](#)
- [GUNAKAN](#)
- [BUAT TABEL](#)
- [UBAH TABEL](#)
- [KEMBALIKAN TABEL](#)
- [MEJA DROP](#)
- [BUAT JENIS](#)

- [TIPE DROP](#)

## Topik

- [Keyspaces](#)
- [Tabel](#)
- [User-defined jenis \(UDT\)](#)

## Keyspaces

Sebuah keyspace mengelompokkan tabel terkait yang relevan untuk satu atau beberapa aplikasi. Dalam hal sistem manajemen basis data relasional (RDBMS), ruang kunci kira-kira mirip dengan database, ruang tabel, atau konstruksi serupa.

 **Note**

Di Apache Cassandra, keyspace menentukan bagaimana data direplikasi di antara beberapa node penyimpanan. Namun, Amazon Keyspaces adalah layanan yang dikelola sepenuhnya: Detail lapisan penyimpanannya dikelola atas nama Anda. Untuk alasan ini, ruang kunci di Amazon Keyspaces hanya konstruksi logis, dan tidak terkait dengan penyimpanan fisik yang mendasarinya.

Untuk informasi tentang batas kuota dan batasan untuk ruang kunci Amazon Keyspaces, lihat. [Kuota](#)

## Pernyataan untuk ruang kunci

- [BUAT KEYSPACE](#)
- [MENGUBAH RUANG KUNCI](#)
- [JATUHKAN RUANG KUNCI](#)
- [USE](#)

## BUAT KEYSPACE

Gunakan `CREATE KEYSPACE` pernyataan untuk membuat keyspace baru.

## Sintaksis

```
create_keyspace_statement ::=
 CREATE KEYSpace [IF NOT EXISTS] keyspace_name
 WITH options
```

Di mana:

- *keyspace\_name* adalah nama keyspace yang akan dibuat.
- pilihan adalah satu atau lebih dari berikut ini:
  - REPLICATION— Peta yang menunjukkan strategi replikasi untuk keyspace:
    - SingleRegionStrategy— Untuk ruang kunci wilayah tunggal. (Diperlukan)
    - NetworkTopologyStrategy— Tentukan setidaknya dua Wilayah AWS. Faktor replikasi untuk setiap Wilayah adalah tiga. (Opsional)
  - DURABLE\_WRITES— Menulis ke Amazon Keyspaces selalu tahan lama, jadi opsi ini tidak diperlukan. Namun, jika ditentukan, nilainya harus `true`.
  - TAGS— Daftar tag pasangan kunci-nilai yang akan dilampirkan ke sumber daya saat Anda membuatnya. (Opsional)

Contoh

Buat keyspace sebagai berikut.

```
CREATE KEYSpace my_keyspace
 WITH REPLICATION = {'class': 'SingleRegionStrategy'} and TAGS ={'key1':'val1',
 'key2':'val2'} ;
```

Untuk membuat ruang kunci Multi-wilayah, tentukan `NetworkTopologyStrategy` dan sertakan setidaknya dua. Wilayah AWS Faktor replikasi untuk setiap Wilayah adalah tiga.

```
CREATE KEYSpace my_keyspace
 WITH REPLICATION = {'class': 'NetworkTopologyStrategy', 'us-east-1': '3', 'ap-
 southeast-1': '3', 'eu-west-1': '3'};
```

## MENGUBAH RUANG KUNCI

Anda dapat menggunakan `ALTER KEYSpace WITH` pernyataan untuk opsi berikut

- REPLICATION— Gunakan opsi ini untuk menambahkan Wilayah AWS replika baru ke ruang kunci. Anda dapat menambahkan Wilayah baru ke wilayah Tunggal atau ke ruang kunci Multi-wilayah.

- TAGS— Gunakan opsi ini untuk menambah atau menghapus tag dari ruang kunci.

## Sintaksis

```
alter_keyspace_statement ::=
 ALTER KEYSPACE keyspace_name
 WITH options
```

Di mana:

- *keyspace\_name* adalah nama keyspace yang akan diubah.
- pilihan adalah salah satu dari berikut:
  - ADD | DROP TAGS— Daftar tag pasangan kunci-nilai yang akan ditambahkan atau dihapus dari ruang kunci.
  - REPLICATION— Peta yang menunjukkan strategi replikasi untuk keyspace;
    - *class*— NetworkTopologyStrategy mendefinisikan keyspace sebagai ruang kunci Multi-wilayah.
    - *region*— Tentukan satu tambahan Wilayah AWS untuk keyspace ini. Faktor replikasi untuk setiap Wilayah adalah tiga.
    - CLIENT\_SIDE\_TIMESTAMPS— Defaultnya adalah DISABLED. Anda hanya dapat mengubah status menjadi ENABLED.

## Contoh

Mengubah keyspace seperti yang ditunjukkan pada contoh berikut untuk menambahkan tag.

```
ALTER KEYSPACE my_keyspace ADD TAGS {'key1':'val1', 'key2':'val2'};
```

Untuk menambahkan Region ketiga ke ruang kunci Multi-region, Anda dapat menggunakan pernyataan berikut.

```
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'};
```

## JATUHKAN RUANG KUNCI

Gunakan `DROP KEYSPACE` pernyataan untuk menghapus keyspace—termasuk semua isinya, seperti tabel.

### Sintaksis

```
drop_keyspace_statement ::=
 DROP KEYSPACE [IF EXISTS] keyspace_name
```

Di mana:

- `keyspace_name` adalah nama keyspace yang akan dijatuhkan.

### Contoh

```
DROP KEYSPACE my_keyspace;
```

## USE

Gunakan `USE` pernyataan untuk menentukan keyspace saat ini. Ini memungkinkan Anda untuk merujuk ke objek yang terikat ke ruang kunci tertentu, misalnya tabel dan jenis, tanpa menggunakan nama yang sepenuhnya memenuhi syarat yang menyertakan awalan keyspace.

### Sintaksis

```
use_statement ::=
 USE keyspace_name
```

Di mana:

- `keyspace_name` adalah nama keyspace yang akan digunakan.

### Contoh

```
USE my_keyspace;
```

## Tabel

Tabel adalah struktur data utama di Amazon Keyspaces. Data dalam tabel disusun menjadi baris dan kolom. Subset dari kolom tersebut digunakan untuk menentukan partisi (dan akhirnya penempatan data) melalui spesifikasi kunci partisi.

Kumpulan kolom lain dapat didefinisikan ke dalam kolom pengelompokan, yang berarti bahwa mereka dapat berpartisipasi sebagai predikat dalam eksekusi kueri.

Secara default, tabel baru dibuat dengan kapasitas throughput sesuai permintaan. Anda dapat mengubah mode kapasitas untuk tabel baru dan yang sudah ada. Untuk informasi selengkapnya tentang mode throughput read/write kapasitas, lihat [the section called “Konfigurasi mode read/write kapasitas”](#).

Untuk tabel dalam mode yang disediakan, Anda dapat mengonfigurasi opsional.

**AUTOSCALING\_SETTINGS** Untuk informasi selengkapnya tentang penskalaan otomatis Amazon Keyspaces dan opsi yang tersedia, lihat. [the section called “Konfigurasi penskalaan otomatis pada tabel yang ada”](#)

Untuk informasi tentang batas kuota dan batasan untuk tabel Amazon Keyspaces, lihat. [Kuota](#)

Pernyataan untuk tabel

- [CREATE TABLE](#)
- [ALTER TABLE](#)
- [MENGEMBALIKAN TABEL](#)
- [MEJA DROP](#)

## CREATE TABLE

Gunakan CREATE TABLE pernyataan untuk membuat tabel baru.

Sintaksis

```
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))*

```

Di mana:

- *table\_name* adalah nama tabel yang akan dibuat. Nama yang sepenuhnya memenuhi syarat termasuk awalan keyspace. Atau, Anda dapat mengatur keyspace saat ini dengan pernyataan `USE keyspace`.
- *column\_definition* terdiri dari yang berikut:
  - *column\_name*— Nama kolom.
  - *cql\_type*— Tipe data Amazon Keyspaces (lihat [Jenis Data](#)).
  - *FROZEN*— Menetapkan kolom ini yang ditentukan pengguna atau tipe `collection` (misalnya, `LISTSET`, atau `MAP`) sebagai beku. Koleksi beku diserialisasikan menjadi satu nilai yang tidak dapat diubah dan diperlakukan seperti a. `BLOB` Untuk informasi selengkapnya, lihat [the section called “Jenis koleksi”](#).
  - *STATIC*— Menunjuk kolom ini sebagai statis. Kolom statis menyimpan nilai yang dibagikan oleh semua baris di partisi yang sama.
  - *PRIMARY KEY*— Menetapkan kolom ini sebagai kunci utama tabel.
- *primary\_key* terdiri dari yang berikut:
  - *partition\_key*
  - *clustering\_columns*



- *partition\_key*:
  - Kunci partisi dapat berupa kolom tunggal, atau dapat berupa nilai majemuk yang terdiri dari dua atau lebih kolom. Bagian kunci partisi dari kunci utama diperlukan dan menentukan bagaimana Amazon Keyspaces menyimpan data Anda.
- *clustering\_columns*:
  - Bagian kolom pengelompokan opsional dari kunci utama Anda menentukan bagaimana data dikelompokkan dan diurutkan dalam setiap partisi.
- *table\_options* terdiri dari yang berikut:
  - *CLUSTERING ORDER BY*— *ORDER CLUSTERING* default pada tabel terdiri dari kunci pengelompokan Anda dalam arah pengurutan ASC (naik). Tentukan untuk mengganti perilaku pengurutan default.
  - *cdc*— Boolean yang menentukan apakah Amazon Keyspaces membuat aliran change data capture (CDC) untuk tabel. Nilai default-nya *false*. Untuk menentukan *view type* saat mengaktifkan aliran, atur *cdc\_specification* dengan *CUSTOM\_PROPERTIES*.
  - *CUSTOM\_PROPERTIES*— Peta pengaturan yang khusus untuk Amazon Keyspaces.
    - *capacity\_mode*: Menentukan modus kapasitas read/write throughput untuk tabel. Opsi nya adalah *throughput\_mode:PAY\_PER\_REQUEST* dan *throughput\_mode:PROVISIONED*. Mode kapasitas yang disediakan membutuhkan *read\_capacity\_units* dan *write\_capacity\_units* sebagai input. Nilai default-nya *throughput\_mode:PAY\_PER\_REQUEST*.
    - *cdc\_specification*: Menentukan *view\_type* aliran CDC. Opsinya adalah:
      - *NEW\_AND\_OLD\_IMAGES*— kedua versi baris, sebelum dan sesudah perubahan. Ini adalah opsi default.
      - *NEW\_IMAGE*— versi baris setelah perubahan.
      - *OLD\_IMAGE*— versi baris sebelum perubahan.
      - *KEYS\_ONLY*— kunci partisi dan pengelompokan baris yang diubah.

Untuk informasi selengkapnya tentang aliran CDC, lihat [the section called “Bekerja dengan aliran CDC”](#). Untuk contoh kode, lihat [the section called “Aktifkan aliran CDC untuk tabel baru”](#).

- *client\_side\_timestamps*: Menentukan apakah stempel waktu sisi klien diaktifkan atau dinonaktifkan untuk tabel. Opsi nya adalah `{ 'status': 'enabled' }` dan `{ 'status': 'disabled' }`. Jika tidak ditentukan, defaultnya adalah *status:disabled*. Setelah stempel waktu sisi klien diaktifkan untuk tabel, pengaturan ini tidak dapat dinonaktifkan.

- `encryption_specification`: Menentukan opsi enkripsi untuk enkripsi saat istirahat. Jika tidak ditentukan, defaultnya adalah `encryption_type:AWS_OWNED_KMS_KEY`. Opsi enkripsi kunci yang dikelola pelanggan memerlukan AWS KMS kunci dalam format Amazon Resource Name (ARN) sebagai input: `kms_key_identifier:ARN.kms_key_identifier:ARN`
- `point_in_time_recovery`: Menentukan apakah pemulihan point-in-time diaktifkan atau dinonaktifkan untuk tabel. Opsi nya adalah `status:enabled` dan `status:disabled`. Jika tidak ditentukan, defaultnya adalah `status:disabled`.
- `replica_updates`: Menentukan pengaturan tabel Multi-region yang khusus untuk. Wilayah AWS Untuk tabel Multi-region, Anda dapat mengonfigurasi kapasitas baca tabel secara berbeda per Wilayah AWS tabel. Anda dapat melakukan ini dengan mengonfigurasi parameter berikut. Untuk informasi selengkapnya dan contoh tambahan, lihat [the section called “Buat tabel Multi-wilayah dalam mode yang disediakan”](#).
  - `region`— Wilayah AWS Replika tabel dengan pengaturan berikut:
    - `read_capacity_units`
- TTL: Mengaktifkan Pengaturan kustom Time to Live untuk tabel. Untuk mengaktifkan, gunakan `status:enabled`. Nilai default-nya `status:disabled`. Setelah TTL diaktifkan, Anda tidak dapat menonaktifkannya untuk tabel.
- `AUTOSCALING_SETTING` termasuk pengaturan opsional berikut untuk tabel dalam mode yang disediakan. Untuk informasi selengkapnya dan contoh tambahan, lihat [the section called “Buat tabel baru dengan penskalaan otomatis”](#).
  - `provisioned_write_capacity_autoscaling_update`:
    - `autoscaling_disabled`— Untuk mengaktifkan penskalaan otomatis untuk kapasitas tulis, atur nilainya ke `false`. Nilai default-nya `true`. (Opsional)
    - `minimum_units`— Tingkat minimum throughput tulis yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimal per detik untuk akun Anda (40.000 secara default).
    - `maximum_units`— Tingkat maksimum throughput tulis yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimal per detik untuk akun Anda (40.000 secara default).
    - `scaling_policy`— Amazon Keyspaces mendukung kebijakan pelacakan target. Target penskalaan otomatis adalah kapasitas tulis tabel yang disediakan.
      - `target_tracking_scaling_policy_configuration`— Untuk menentukan kebijakan pelacakan target, Anda harus menentukan nilai target. Untuk informasi

selengkapnya tentang pelacakan target dan periode cooldown, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).

- `target_value`— Tingkat pemanfaatan target tabel. Penskalaan otomatis Amazon Keyspaces memastikan bahwa rasio kapasitas yang dikonsumsi terhadap kapasitas yang disediakan tetap pada atau mendekati nilai ini. Anda mendefinisikan `target_value` sebagai persentase. Ganda antara 20 dan 90. (Diperlukan)
- `scale_in_cooldown`— Periode cooldown dalam hitungan detik antara aktivitas penskalaan yang memungkinkan tabel stabil sebelum skala aktivitas lain dimulai. Jika tidak ada nilai yang diberikan, defaultnya adalah 0. (Opsional)
- `scale_out_cooldown`— Periode cooldown dalam hitungan detik antara aktivitas penskalaan yang memungkinkan tabel stabil sebelum aktivitas skala lain dimulai. Jika tidak ada nilai yang diberikan, defaultnya adalah 0. (Opsional)
- `disable_scale_in`: A boolean yang menentukan apakah `scale-in` dinonaktifkan atau diaktifkan untuk tabel. Parameter ini dinonaktifkan secara default. Untuk menghidupkan `scale-in`, atur boolean nilainya ke `FALSE`. Ini berarti bahwa kapasitas secara otomatis diperkecil untuk tabel atas nama Anda. (Opsional)
- `provisioned_read_capacity_autoscaling_update`:
  - `autoscaling_disabled`— Untuk mengaktifkan penskalaan otomatis untuk kapasitas baca, atur nilainya ke `false`. Nilai default-nya `true`. (Opsional)
  - `minimum_units`— Tingkat throughput minimum yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimal per detik untuk akun Anda (40.000 secara default).
  - `maximum_units`— Tingkat throughput maksimum yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimal per detik untuk akun Anda (40.000 secara default).
  - `scaling_policy`— Amazon Keyspaces mendukung kebijakan pelacakan target. Target penskalaan otomatis adalah kapasitas baca tabel yang disediakan.
  - `target_tracking_scaling_policy_configuration`— Untuk menentukan kebijakan pelacakan target, Anda harus menentukan nilai target. Untuk informasi selengkapnya tentang pelacakan target dan periode cooldown, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).
  - `target_value`— Tingkat pemanfaatan target tabel. Penskalaan otomatis Amazon Keyspaces memastikan bahwa rasio kapasitas yang dikonsumsi terhadap

kapasitas yang disediakan tetap pada atau mendekati nilai ini. Anda mendefinisikan `target_value` sebagai persentase. Ganda antara 20 dan 90. (Diperlukan)

- `scale_in_cooldown`— Periode cooldown dalam hitungan detik antara aktivitas penskalaan yang memungkinkan tabel stabil sebelum skala aktivitas lain dimulai. Jika tidak ada nilai yang diberikan, defaultnya adalah 0. (Opsional)
- `scale_out_cooldown`— Periode cooldown dalam hitungan detik antara aktivitas penskalaan yang memungkinkan tabel stabil sebelum aktivitas skala lain dimulai. Jika tidak ada nilai yang diberikan, defaultnya adalah 0. (Opsional)
- `disable_scale_in`: A boolean yang menentukan apakah `scale-in` dinonaktifkan atau diaktifkan untuk tabel. Parameter ini dinonaktifkan secara default. Untuk menghidupkan `scale-in`, atur boolean nilainya ke `FALSE`. Ini berarti bahwa kapasitas secara otomatis diperkecil untuk tabel atas nama Anda. (Opsional)
- `replica_updates`: Menentukan pengaturan penskalaan otomatis Wilayah AWS tertentu dari tabel Multi-wilayah. Untuk tabel Multi-region, Anda dapat mengonfigurasi kapasitas baca tabel secara berbeda per Wilayah AWS tabel. Anda dapat melakukan ini dengan mengonfigurasi parameter berikut. Untuk informasi selengkapnya dan contoh tambahan, lihat [the section called “Memperbarui pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”](#).
- `region`— Wilayah AWS Replika tabel dengan pengaturan berikut:
  - `provisioned_read_capacity_autoscaling_update`
  - `autoscaling_disabled`— Untuk mengaktifkan penskalaan otomatis untuk kapasitas baca tabel, atur nilainya ke `false`. Nilai default-nya `true`. (Opsional)

 Note

Penskalaan otomatis untuk tabel Multi-wilayah harus diaktifkan atau dinonaktifkan untuk semua replika tabel.

- `minimum_units`— Tingkat minimum throughput baca yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimal per detik untuk akun Anda (40.000 secara default).
- `maximum_units`— Tingkat maksimum throughput baca yang harus selalu siap didukung oleh tabel. Nilai harus antara 1 dan kuota throughput maksimal per detik untuk akun Anda (40.000 secara default).

- `scaling_policy`— Amazon Keyspaces mendukung kebijakan pelacakan target. Target penskalaan otomatis adalah kapasitas baca tabel yang disediakan.
- `target_tracking_scaling_policy_configuration`— Untuk menentukan kebijakan pelacakan target, Anda harus menentukan nilai target. Untuk informasi selengkapnya tentang pelacakan target dan periode cooldown, lihat [Kebijakan Penskalaan Pelacakan Target di Panduan Pengguna Application Auto Scaling](#).
- `target_value`— Tingkat pemanfaatan target tabel. Penskalaan otomatis Amazon Keyspaces memastikan bahwa rasio kapasitas baca yang dikonsumsi terhadap kapasitas baca yang disediakan tetap pada atau mendekati nilai ini. Anda mendefinisikan `target_value` sebagai persentase. Ganda antara 20 dan 90. (Diperlukan)
- `scale_in_cooldown`— Periode cooldown dalam hitungan detik antara aktivitas penskalaan yang memungkinkan tabel stabil sebelum skala aktivitas lain dimulai. Jika tidak ada nilai yang diberikan, defaultnya adalah 0. (Opsional)
- `scale_out_cooldown`— Periode cooldown dalam hitungan detik antara aktivitas penskalaan yang memungkinkan tabel stabil sebelum aktivitas skala lain dimulai. Jika tidak ada nilai yang diberikan, defaultnya adalah 0. (Opsional)
- `disable_scale_in`: A boolean yang menentukan apakah `scale-in` dinonaktifkan atau diaktifkan untuk tabel. Parameter ini dinonaktifkan secara default. Untuk menghidupkan `scale-in`, atur boolean nilainya ke `FALSE`. Ini berarti bahwa kapasitas baca secara otomatis diperkecil untuk tabel atas nama Anda. (Opsional)
- `default_time_to_live`— Pengaturan Waktu untuk Hidup default dalam hitungan detik untuk tabel.
- `TAGS`— Daftar tag pasangan kunci-nilai yang akan dilampirkan ke sumber daya saat dibuat.
- `clustering_order` terdiri dari yang berikut:
  - `column_name`— Nama kolom.
  - `ASC / DESC`— Mengatur pengubah urutan ascendant (ASC) atau descendant (DESC). Jika tidak ditentukan, urutan defaultnya adalah ASC.

## Contoh

```
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;

```

Dalam tabel yang menggunakan kolom pengelompokan, kolom non-clustering dapat dideklarasikan sebagai statis dalam definisi tabel. Untuk informasi selengkapnya tentang kolom statis, lihat [the section called “Perkiraan konsumsi kapasitas kolom statis”](#).

## Contoh

```

CREATE TABLE my_keyspace.my_table (
 id int,
 name text,
 region text,
 division text,
 project text STATIC,
 PRIMARY KEY (id,division));

```

Anda dapat membuat tabel dengan kolom yang menggunakan tipe yang ditentukan pengguna (UDT). Pernyataan pertama dalam contoh menciptakan tipe, pernyataan kedua membuat tabel dengan kolom yang menggunakan tipe.

### Contoh

```
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

Gunakan ALTER TABLE pernyataan untuk menambahkan kolom baru, menambahkan tag, atau mengubah properti kustom tabel.

### Sintaksis

```
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
```

Di mana:

- *table\_name* adalah nama tabel yang akan diubah.
- *column\_definition* adalah nama kolom dan tipe data yang akan ditambahkan.
- *column\_definition\_list* adalah daftar kolom yang dipisahkan koma yang ditempatkan di dalam tanda kurung.
- *table\_option* terdiri dari yang berikut:
  - *AUTOSCALING\_SETTING* termasuk pengaturan penskalaan otomatis opsional untuk tabel yang disediakan. Untuk sintaks dan deskripsi rinci, lihat [the section called “CREATE TABLE”](#) Sebagai contoh, lihat [the section called “Konfigurasi penskalaan otomatis pada tabel yang ada”](#).
  - *cdc*— Boolean yang menentukan apakah Amazon Keyspaces membuat aliran change data capture (CDC) untuk tabel. Nilai default-nya false. Untuk menentukan view type saat mengaktifkan aliran, atur *cdc\_specification* dengan CUSTOM\_PROPERTIES.

- **CUSTOM\_PROPERTIES**— Peta pengaturan khusus untuk Amazon Keyspaces.
  - **capacity\_mode**: Menentukan modus kapasitas read/write throughput untuk tabel. Opsinya adalah `throughput_mode:PAY_PER_REQUEST` dan `throughput_mode:PROVISIONED`. Mode kapasitas yang disediakan membutuhkan `read_capacity_units` dan `write_capacity_units` sebagai input. Nilai default-nya `throughput_mode:PAY_PER_REQUEST`.
  - **cdc\_specification**: Menentukan `view_type` aliran CDC. Opsinya adalah:
    - **NEW\_AND\_OLD\_IMAGES**— kedua versi baris, sebelum dan sesudah perubahan. Ini adalah opsi default.
    - **NEW\_IMAGE**— versi baris setelah perubahan.
    - **OLD\_IMAGE**— versi baris sebelum perubahan.
    - **KEYS\_ONLY**— kunci partisi dan pengelompokan baris yang diubah.

Untuk informasi selengkapnya tentang aliran CDC, lihat [the section called “Bekerja dengan aliran CDC”](#). Untuk contoh kode, lihat [the section called “Aktifkan aliran CDC untuk tabel yang ada”](#).

- **client\_side\_timestamps**: Menentukan apakah stempel waktu sisi klien diaktifkan atau dinonaktifkan untuk tabel. Opsinya adalah `{'status': 'enabled'}` dan `{'status': 'disabled'}`. Jika tidak ditentukan, defaultnya adalah `status:disabled`. Setelah stempel waktu sisi klien diaktifkan untuk tabel, pengaturan ini tidak dapat dinonaktifkan.
- **encryption\_specification**: Menentukan opsi enkripsi untuk enkripsi saat istirahat. Opsinya adalah `encryption_type:AWS_OWNED_KMS_KEY` dan `encryption_type:CUSTOMER_MANAGED_KMS_KEY`. Opsi enkripsi kunci yang dikelola pelanggan memerlukan AWS KMS kunci dalam format Amazon Resource Name (ARN) sebagai input: `kms_key_identifier:ARN`
- **point\_in\_time\_recovery**: Menentukan apakah pemulihan point-in-time diaktifkan atau dinonaktifkan untuk tabel. Opsinya adalah `status:enabled` dan `status:disabled`. Nilai default-nya `status:disabled`.
- **replica\_updates**: Menentukan pengaturan Wilayah AWS spesifik dari tabel Multi-region. Untuk tabel Multi-region, Anda dapat mengonfigurasi kapasitas baca tabel secara berbeda per Wilayah AWS tabel. Anda dapat melakukan ini dengan mengonfigurasi parameter berikut. Untuk informasi selengkapnya dan contoh tambahan, lihat [the section called “Memperbarui pengaturan kapasitas dan penskalaan otomatis yang disediakan untuk tabel Multi-wilayah”](#).
  - **region**— Wilayah AWS Replika tabel dengan pengaturan berikut:



- `read_capacity_units`
- `t1`: Mengaktifkan Pengaturan kustom Time to Live untuk tabel. Untuk mengaktifkan, gunakan `status:enabled`. Nilai default-nya `status:disabled`. Setelah `t1` diaktifkan, Anda tidak dapat menonaktifkannya untuk tabel.
- `default_time_to_live`: Pengaturan Waktu ke Langsung default dalam hitungan detik untuk tabel.
- `TAGS` adalah daftar tag pasangan kunci-nilai yang akan dilampirkan ke sumber daya.

#### Note

Dengan `ALTER TABLE`, Anda hanya dapat mengubah satu properti kustom. Anda tidak dapat menggabungkan lebih dari satu perintah `ALTER TABLE` dalam pernyataan yang sama.

## Contoh

Pernyataan berikut menunjukkan cara menambahkan kolom ke tabel yang ada.

```
ALTER TABLE mykeyspace.mytable ADD (ID int);
```

Pernyataan ini menunjukkan cara menambahkan dua kolom koleksi ke tabel yang ada:

- Kolom koleksi beku `col_frozen_list` yang berisi koleksi beku bersarang
- Kolom koleksi non-beku `col_map` yang berisi koleksi beku bersarang

```
ALTER TABLE my_Table ADD(col_frozen_list FROZEN<LIST<FROZEN<SET<TEXT>>>>, col_map
MAP<INT, FROZEN<SET<INT>>>>);
```

Contoh berikut menunjukkan cara menambahkan kolom yang menggunakan tipe yang ditentukan pengguna (UDT) ke tabel.

```
ALTER TABLE my_keyspace.my_table ADD (my_column, my_udt);
```

Untuk mengubah mode kapasitas tabel dan menentukan unit kapasitas baca dan tulis, Anda dapat menggunakan pernyataan berikut.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'capacity_mode':
{'throughput_mode': 'PROVISIONED', 'read_capacity_units': 10, 'write_capacity_units':
20}};
```

Pernyataan berikut menentukan kunci KMS yang dikelola pelanggan untuk tabel.

```
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'
 }
};
```

Untuk mengaktifkan pemulihan point-in-time untuk tabel, Anda dapat menggunakan pernyataan berikut.

```
ALTER TABLE mykeyspace.mytable WITH CUSTOM_PROPERTIES={'point_in_time_recovery':
{'status': 'enabled'}};
```

Untuk menetapkan nilai Time to Live default dalam hitungan detik untuk tabel, Anda dapat menggunakan pernyataan berikut.

```
ALTER TABLE my_table WITH default_time_to_live = 2592000;
```

Pernyataan ini memungkinkan pengaturan Waktu ke Langsung kustom untuk tabel.

```
ALTER TABLE mytable WITH CUSTOM_PROPERTIES={'ttl':{'status': 'enabled'}};
```

## MENGEMBALIKAN TABEL

Gunakan `RESTORE TABLE` pernyataan untuk mengembalikan tabel ke titik waktu. Pernyataan ini membutuhkan pemulihan point-in-time untuk diaktifkan di atas meja. Untuk informasi selengkapnya, lihat [the section called “Backup dan restore dengan pemulihan point-in-time”](#).

### Sintaksis

```
restore_table_statement ::=
```

```
RESTORE TABLE restored_table_name FROM TABLE source_table_name
[WITH table_options [, ...]];
```

Di mana:

- *restored\_table\_name* adalah nama tabel yang dipulihkan.
- *source\_table\_name* adalah nama tabel sumber.
- *table\_option* terdiri dari yang berikut:
  - *restore\_timestamp* adalah waktu titik pemulihan dalam format ISO 8601. Jika tidak ditentukan, stempel waktu saat ini digunakan.
  - *CUSTOM\_PROPERTIES*— Peta pengaturan khusus untuk Amazon Keyspaces.
    - *capacity\_mode*: Menentukan modus kapasitas read/write throughput untuk tabel. Opsi nya adalah *throughput\_mode:PAY\_PER\_REQUEST* dan *throughput\_mode:PROVISIONED*. Mode kapasitas yang disediakan membutuhkan *read\_capacity\_units* dan *write\_capacity\_units* sebagai input. Defaultnya adalah pengaturan saat ini dari tabel sumber.
    - *encryption\_specification*: Menentukan opsi enkripsi untuk enkripsi saat istirahat. Opsi nya adalah *encryption\_type:AWS\_OWNED\_KMS\_KEY* dan *encryption\_type:CUSTOMER\_MANAGED\_KMS\_KEY*. Opsi enkripsi kunci yang dikelola pelanggan memerlukan AWS KMS kunci dalam format Amazon Resource Name (ARN) sebagai input: *kms\_key\_identifier:ARN* Untuk memulihkan tabel yang dienkripsi dengan kunci terkelola pelanggan ke tabel yang dienkripsi dengan, Kunci milik AWS Amazon Keyspaces memerlukan akses ke kunci tabel sumber. AWS KMS
    - *point\_in\_time\_recovery*: Menentukan apakah pemulihan point-in-time diaktifkan atau dinonaktifkan untuk tabel. Opsi nya adalah *status:enabled* dan *status:disabled*. Tidak seperti ketika Anda membuat tabel baru, status default untuk tabel dipulihkan adalah *status:enabled* karena pengaturan diwarisi dari tabel sumber. Untuk menonaktifkan Pitr untuk tabel yang dipulihkan, Anda harus mengatur *status:disabled* secara eksplisit.
    - *replica\_updates*: Menentukan pengaturan Wilayah AWS spesifik dari tabel Multi-region. Untuk tabel Multi-region, Anda dapat mengonfigurasi kapasitas baca tabel secara berbeda per Wilayah AWS tabel. Anda dapat melakukan ini dengan mengonfigurasi parameter berikut.
      - *region*— Wilayah AWS Replika tabel dengan pengaturan berikut:
        - *read\_capacity\_units*
    - *AUTOSCALING\_SETTING* termasuk pengaturan penskalaan otomatis opsional untuk tabel yang disediakan. Untuk sintaks dan deskripsi terperinci, lihat [the section called “CREATE TABLE”](#)

- **TAGS** adalah daftar tag pasangan kunci-nilai yang akan dilampirkan ke sumber daya.

#### Note

Tabel yang dihapus hanya dapat dikembalikan ke waktu penghapusan.

## Contoh

```
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'};
```

## MEJA DROP

Gunakan **DROP TABLE** pernyataan untuk menghapus tabel dari keyspace.

### Sintaksis

```
drop_table_statement ::=
 DROP TABLE [IF EXISTS] table_name
```

Di mana:

- **IF EXISTS** **DROP TABLE** mencegah kegagalan jika tabel tidak ada. (Opsional)
- *table\_name* adalah nama tabel yang akan dijatuhkan.

## Contoh

```
DROP TABLE my_keyspace.my_table;
```

## User-defined jenis (UDT)

UDT — Pengelompokan bidang dan tipe data yang dapat Anda gunakan untuk menentukan satu kolom di Amazon Keyspaces. Tipe data yang valid untuk UDT adalah semua tipe data Cassandra

yang didukung, termasuk koleksi dan UDT lain yang telah Anda buat di ruang kunci yang sama. Untuk informasi selengkapnya tentang tipe data Cassandra yang didukung, lihat. [the section called “Dukungan tipe data Cassandra”](#)

```
user_defined_type ::= udt_name
udt_name ::= [keyspace_name '.'] identifier
```

Pernyataan untuk tipe

- [BUAT TIPE](#)
- [TIPE DROP](#)

## BUAT TIPE

Gunakan `CREATE TYPE` pernyataan untuk membuat tipe baru.

Sintaksis

```
create_type_statement ::= CREATE TYPE [IF NOT EXISTS] udt_name
 '(' field_definition (',' field_definition) * ')'
 field_definition ::= identifier cql_type
```

Di mana:

- `IF NOT EXISTS` mencegah kegagalan jika tipe sudah ada. (Opsional)
- *udt\_name* adalah nama UDT yang sepenuhnya memenuhi syarat dalam format tipe, misalnya. `my_keyspace.my_type` Jika Anda mendefinisikan keyspace saat ini dengan `USE` pernyataan, Anda tidak perlu menentukan nama keyspace.
- *field\_definition* terdiri dari nama dan tipe.

Tabel berikut menunjukkan contoh nama UDT yang diizinkan. Kolom pertama menunjukkan cara memasukkan nama saat Anda membuat jenis, kolom kedua menunjukkan bagaimana Amazon Keyspaces memformat nama secara internal. Amazon Keyspaces mengharapkan nama yang diformat untuk operasi seperti. `GetType`

| Nama yang dimasukkan      | Nama yang diformat     | Catatan                                                                                                                                |
|---------------------------|------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| MY_UDT                    | my_udt                 | Tanpa tanda kutip ganda, Amazon Keyspaces mengonversi semua karakter huruf besar menjadi huruf kecil.                                  |
| "MY_UDT"                  | MY_UDT                 | Dengan tanda kutip ganda, Amazon Keyspaces menghormati karakter huruf besar, dan menghapus tanda kutip ganda dari nama yang diformat.  |
| "1234"                    | 1234                   | Dengan tanda kutip ganda, nama dapat dimulai dengan angka, dan Amazon Keyspaces menghapus tanda kutip ganda dari nama yang diformat.   |
| "Special_Ch@r@cter s<>!!" | Special_Ch@r@cters<>!! | Dengan tanda kutip ganda, nama dapat berisi karakter khusus, dan Amazon Keyspaces menghapus tanda kutip ganda dari nama yang diformat. |
| "nested""""quote s"       | nested""""quotes       | Amazon Keyspaces menghapus tanda kutip ganda luar dan tanda kutip ganda escape dari nama yang diformat.                                |

## Contoh

```
CREATE TYPE my_keyspace.phone (
 country_code int,
 number text
);
```

Anda dapat membuat sarang UDT jika UDT bersarang dibekukan. Untuk informasi selengkapnya tentang nilai default dan kuota untuk tipe, lihat [the section called “Kuota UDT Amazon Keyspaces dan nilai default”](#).

```
CREATE TYPE my_keyspace.user (
 first_name text,
 last_name text,
 phones FROZEN<phone>
);
```

Untuk contoh kode lainnya yang menunjukkan cara membuat UDT, lihat [the section called “User-defined jenis \(UDT\)”](#).

## TIPE DROP

Gunakan `DROP TYPE` pernyataan untuk menghapus UDT. Anda hanya dapat menghapus tipe yang tidak digunakan oleh tipe atau tabel lain.

### Sintaksis

```
drop_type_statement ::= DROP TYPE [IF EXISTS] udt_name
```

Di mana:

- `IF EXISTS` `DROP TYPE` mencegah kegagalan jika tipenya tidak ada. (Opsional)
- *udt\_name* adalah nama UDT yang sepenuhnya memenuhi syarat dalam format tipe, misalnya, `my_keyspace.my_type`. Jika Anda mendefinisikan keyspace saat ini dengan `USE` pernyataan, Anda tidak perlu menentukan nama keyspace.

### Contoh

```
DROP TYPE udt_name;
```

## Pernyataan DHTML (bahasa manipulasi data) di Amazon Keyspaces

Bahasa manipulasi data (DHTML) adalah kumpulan pernyataan Cassandra Query Language (CQL) yang Anda gunakan untuk mengelola data di Amazon Keyspaces (untuk Apache Cassandra) tabel. Anda menggunakan pernyataan DML untuk menambah, mengubah, atau menghapus data dalam sebuah tabel.

Anda juga menggunakan pernyataan DML untuk query data dalam tabel. (Perhatikan bahwa CQL tidak mendukung gabungan atau subkueri.)

### Topik

- [SELECT](#)
- [INSERT](#)

- [UPDATE](#)
- [DELETE](#)

## SELECT

Gunakan pernyataan SELECT untuk menanyakan data.

### Sintaksis

```

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])*
```

### Contoh

```

SELECT name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;

SELECT JSON name, id, manager_id FROM "myGSGKeyspace".employees_tbl ;
```

Untuk tabel yang memetakan tipe JSON-encoded data ke tipe data Amazon Keyspaces, lihat. [the section called “Pengkodean JSON dari tipe data Amazon Keyspaces”](#)

### Menggunakan kata **IN** kunci

INKata kunci menentukan kesetaraan untuk satu atau lebih nilai. Ini dapat diterapkan ke kunci partisi dan kolom pengelompokan. Hasil dikembalikan dalam urutan kunci disajikan dalam SELECT pernyataan.



## Contoh

```
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;
```

Untuk informasi selengkapnya tentang IN kata kunci dan cara Amazon Keyspaces memproses pernyataan, lihat [the section called “Gunakan IN SELECT”](#)

## Hasil pemesanan

ORDER BY klausa menentukan urutan dari hasil yang dikembalikan. Dibutuhkan sebagai argumen daftar nama kolom bersama dengan urutan pengurutan untuk setiap kolom. Anda hanya dapat menentukan kolom pengelompokan dalam klausa pengurutan. Non-clustering kolom tidak diperbolehkan. Opsi urutan sortir adalah ASC untuk urutan naik dan DESC untuk urutan menurun. Jika urutan pengurutan dihilangkan, urutan default kolom pengelompokan digunakan. Untuk kemungkinan urutan pesanan, lihat [the section called “Hasil pesanan”](#).

## Contoh

```
SELECT name, id, division, manager_id FROM "myGSGKeyspace".employees_tbl WHERE id =
'012-34-5678' ORDER BY division;
```

Saat menggunakan ORDER BY dengan IN kata kunci, hasil diurutkan dalam halaman. Pemesanan ulang penuh dengan pagination dinonaktifkan tidak didukung.

## TOKEN

Anda dapat menerapkan TOKEN fungsi ke PARTITION KEY kolom SELECT dan WHERE klausa. Dengan TOKEN fungsi tersebut, Amazon Keyspaces mengembalikan baris berdasarkan nilai token yang dipetakan PARTITION\_KEY daripada nilai. PARTITION KEY

TOKEN hubungan tidak didukung dengan IN kata kunci.

## Contoh

```
SELECT TOKEN(id) from my_table;

SELECT TOKEN(id) from my_table WHERE TOKEN(id) > 100 and TOKEN(id) < 10000;
```

## Fungsi TTL

Anda dapat menggunakan TTL fungsi dengan SELECT pernyataan untuk mengambil waktu kedaluwarsa dalam detik yang disimpan untuk kolom. Jika tidak ada TTL nilai yang ditetapkan, fungsi kembalinya 11.

### Contoh

```
SELECT TTL(my_column) from my_table;
```

TTL Fungsi ini tidak dapat digunakan pada kolom multi-sel seperti koleksi.

## WRITETIME fungsi

Anda dapat menggunakan WRITETIME fungsi dengan SELECT pernyataan untuk mengambil stempel waktu yang disimpan sebagai metadata untuk nilai kolom hanya jika tabel menggunakan stempel waktu sisi klien. Untuk informasi selengkapnya, lihat [the section called “Client-side stempel waktu”](#).

```
SELECT WRITETIME(my_column) from my_table;
```

WRITETIME Fungsi ini tidak dapat digunakan pada kolom multi-sel seperti koleksi.

### Note

Untuk kompatibilitas dengan perilaku driver Cassandra yang telah ditetapkan, kebijakan otorisasi berbasis tag tidak diberlakukan saat Anda melakukan operasi pada tabel sistem dengan menggunakan panggilan API Cassandra Query Language (CQL) melalui driver Cassandra dan alat pengembang. Untuk informasi selengkapnya, lihat [the section called “Akses sumber daya Amazon Keyspaces berdasarkan tag”](#).

## INSERT

Gunakan INSERT pernyataan untuk menambahkan baris ke tabel.

### Sintaksis

```
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)* ')'
```

## Contoh

```
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') ;
```

## Perbarui parameter

INSERT mendukung nilai-nilai berikut sebagai `update_parameter`:

- **TTL**— Nilai waktu dalam hitungan detik. Nilai maksimum yang dapat dikonfigurasi adalah 630.720.000 detik, yang setara dengan 20 tahun.
- **TIMESTAMP**— `bigint` Nilai yang mewakili jumlah mikrodetik sejak waktu dasar standar yang dikenal sebagai epoch: 1 Januari 1970 pukul 00:00:00 GMT. Stempel waktu di Amazon Keyspaces harus berada di antara kisaran 2 hari di masa lalu dan 5 menit di masa depan.

## Contoh

```
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;
```

## Dukungan JSON

Untuk tabel yang memetakan tipe JSON-encoded data ke tipe data Amazon Keyspaces, lihat [the section called “Pengkodean JSON dari tipe data Amazon Keyspaces”](#)

Anda dapat menggunakan JSON kata kunci untuk menyisipkan peta JSON -encoded sebagai satu baris. Untuk kolom yang ada dalam tabel tetapi dihilangkan dalam pernyataan sisipan JSON, gunakan `DEFAULT UNSET` untuk mempertahankan nilai yang ada. Gunakan `DEFAULT NULL` untuk menulis nilai NULL ke setiap baris kolom yang dihilangkan dan menimpa nilai yang ada (biaya penulisan standar berlaku). `DEFAULT NULL` adalah opsi default.

## Contoh

```
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"}';
```

Jika data JSON berisi kunci duplikat, Amazon Keyspaces menyimpan nilai terakhir untuk kunci (mirip dengan Apache Cassandra). Dalam contoh berikut, di mana kunci duplikat `id`, nilainya `234-56-7890` digunakan.

## Contoh

```
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

Gunakan UPDATE pernyataan untuk memodifikasi baris dalam tabel.

### Sintaksis

```
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

```

## Contoh

```

UPDATE "myGSGKeyspace".employees_tbl SET pay_scale = 5 WHERE id = '567-89-0123' AND
division = 'Marketing' ;

```

Untuk menambah counter, gunakan sintaks berikut. Untuk informasi selengkapnya, lihat [the section called “Penghitung”](#).

```

UPDATE ActiveUsers SET counter = counter + 1 WHERE user = A70FE1C0-5408-4AE3-
BE34-8733E5K09F14 AND action = 'click';

```

## Perbarui parameter

UPDATE mendukung nilai-nilai berikut sebagai `update_parameter`:

- **TTL**— Nilai waktu dalam hitungan detik. Nilai maksimum yang dapat dikonfigurasi adalah 630.720.000 detik, yang setara dengan 20 tahun.
- **TIMESTAMP**— `bigint` Nilai yang mewakili jumlah mikrodetik sejak waktu dasar standar yang dikenal sebagai epoch: 1 Januari 1970 pukul 00:00:00 GMT. Stempel waktu di Amazon Keyspaces harus berada di antara kisaran 2 hari di masa lalu dan 5 menit di masa depan.

## Contoh

```

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

Gunakan DELETE pernyataan untuk menghapus baris dari tabel.

## Sintaksis

```

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

```

Di mana:

- *table\_name* adalah tabel yang berisi baris yang ingin Anda hapus.

Contoh

```

DELETE manager_id FROM "myGSGKeyspace".employees_tbl WHERE id='789-01-2345' AND
division='Executive' ;

```

DELETE mendukung nilai berikut sebagai *update\_parameter*:

- **TIMESTAMP**— *bigint* Nilai yang mewakili jumlah mikrodetik sejak waktu dasar standar yang dikenal sebagai epoch: 1 Januari 1970 pukul 00:00:00 GMT.

## Built-in fungsi di Amazon Keyspaces

Amazon Keyspaces (untuk Apache Cassandra) mendukung berbagai fungsi bawaan yang dapat Anda gunakan dalam pernyataan Cassandra Query Language (CQL).

Topik

- [Fungsi skalar](#)

## Fungsi skalar

Fungsi skalar melakukan perhitungan pada nilai tunggal dan mengembalikan hasilnya sebagai nilai tunggal. Amazon Keyspaces mendukung fungsi skalar berikut.

| Fungsi                        | Deskripsi                                                                                         |
|-------------------------------|---------------------------------------------------------------------------------------------------|
| <code>blobAsType</code>       | Mengembalikan nilai dari tipe data yang ditentukan.                                               |
| <code>cast</code>             | Mengkonversi satu tipe data asli ke tipe data asli lainnya.                                       |
| <code>currentDate</code>      | Mengembalikan arus date/time sebagai tanggal.                                                     |
| <code>currentTime</code>      | Mengembalikan arus date/time sebagai waktu.                                                       |
| <code>currentTimestamp</code> | Mengembalikan arus date/time sebagai stempel waktu.                                               |
| <code>currentTimeUUID</code>  | Mengembalikan arus date/time sebagai <code>timeuuid</code> .                                      |
| <code>fromJson</code>         | Mengkonversi string JSON ke tipe data kolom yang dipilih.                                         |
| <code>maxTimeuuid</code>      | Mengembalikan kemungkinan terbesar <code>timeuuid</code> untuk stempel waktu atau string tanggal. |
| <code>minTimeuuid</code>      | Mengembalikan sekecil mungkin <code>timeuuid</code> untuk stempel waktu atau string tanggal.      |
| <code>now</code>              | Mengembalikan unik baru <code>timeuuid</code> .                                                   |
| <code>toDate</code>           | Mengonversi salah satu <code>timeuuid</code> atau stempel waktu ke jenis tanggal.                 |
| <code>toJson</code>           | Mengembalikan nilai kolom kolom yang dipilih dalam format JSON.                                   |
| <code>token</code>            | Mengembalikan nilai hash dari kunci partisi.                                                      |

| Fungsi                       | Deskripsi                                                                                                                                    |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <code>toTimestamp</code>     | Mengonversi tanggal <code>timeuuid</code> atau tanggal menjadi stempel waktu.                                                                |
| <code>TTL</code>             | Mengembalikan waktu kedaluwarsa dalam hitungan detik untuk kolom.                                                                            |
| <code>typeAsBlob</code>      | Mengkonversi tipe data yang ditentukan menjadi <code>blob</code> .                                                                           |
| <code>toUnixTimestamp</code> | Mengkonversi salah satu <code>timeuuid</code> atau stempel waktu menjadi <code>bigint</code> .                                               |
| <code>uuid</code>            | Mengembalikan versi acak 4 UUID.                                                                                                             |
| <code>writetime</code>       | Mengembalikan stempel waktu dari nilai kolom yang ditentukan.                                                                                |
| <code>dateOf</code>          | (Usang) Mengekstrak stempel waktu dari <code>atimeuuid</code> , dan mengembalikan nilai sebagai tanggal.                                     |
| <code>unixTimestampOf</code> | (Deprecated) Mengekstrak stempel waktu dari <code>atimeuuid</code> , dan mengembalikan nilainya sebagai stempel waktu integer 64-bit mentah. |



# Kuota untuk Amazon Keyspaces (untuk Apache Cassandra)

Bagian ini menjelaskan kuota saat ini dan nilai default untuk Amazon Keyspaces (untuk Apache Cassandra).

## Topik

- [Kuota layanan Amazon Keyspaces](#)
- [Meningkatkan atau mengurangi throughput \(untuk tabel yang disediakan\)](#)
- [Enkripsi Amazon Keyspaces saat istirahat](#)
- [Kuota dan nilai default untuk tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces](#)

## Kuota layanan Amazon Keyspaces

Tabel berikut berisi Amazon Keyspaces (untuk Apache Cassandra) kuota dan nilai default. Informasi tentang kuota mana yang dapat disesuaikan tersedia di konsol [Service Quotas](#), di mana Anda juga dapat meminta kenaikan kuota. Untuk informasi lebih lanjut tentang kuota, hubungi AWS Dukungan.

| Kuota                                                               | Deskripsi                                                                                                                                                        | Amazon Keyspaces default |
|---------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Ruang kunci maks per Wilayah AWS                                    | Jumlah maksimum ruang kunci untuk pelanggan ini per Wilayah. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .                | 256                      |
| Tabel maks per Wilayah AWS                                          | Jumlah maksimum tabel di semua ruang kunci untuk pelanggan ini per Wilayah. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> . | 256                      |
| Jumlah maksimum data yang dipulihkan menggunakan ADD REGION operasi | Ukuran maksimum data yang dapat dipulihkan secara bersamaan oleh ADD REGION                                                                                      | 10 TB                    |

| Kuota                                                    | Deskripsi                                                                                                                                                                                                                                                              | Amazon Keyspaces default |
|----------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
|                                                          | operasi. Untuk meningkatkan jumlah data yang akan dipulihkan secara bersamaan, hubungi Dukungan.                                                                                                                                                                       |                          |
| Ukuran skema tabel maks                                  | Ukuran maksimum skema tabel.                                                                                                                                                                                                                                           | 350 KB                   |
| Operasi DDL bersamaan maks                               | Jumlah maksimum operasi DDL bersamaan yang diizinkan untuk pelanggan ini per Wilayah.                                                                                                                                                                                  | 50                       |
| Kueri maks per koneksi                                   | Jumlah maksimum kueri CQL yang dapat diproses oleh koneksi TCP klien tunggal per detik.                                                                                                                                                                                | 3000                     |
| Ukuran baris maksimal                                    | Ukuran maksimum baris, tidak termasuk data kolom statis. Lihat perinciannya di <a href="#">the section called “Perkirakan ukuran baris”</a> .                                                                                                                          | 1 MB                     |
| Jumlah maksimum kolom dalam INSERT dan UPDATE pernyataan | Jumlah maksimum kolom yang diizinkan dalam CQL INSERT atau UPDATE pernyataan. UPDATE Pernyataan INSERT atau mendukung hingga 225 kolom reguler saat Time to Live (TTL) dimatikan. Jika TTL dihidupkan, hingga 166 kolom reguler dapat dimodifikasi dalam satu operasi. | 225/166                  |

| Kuota                                                  | Deskripsi                                                                                                                                                                                                                                                                                           | Amazon Keyspaces default |
|--------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Data statis maks per partisi logis                     | Ukuran agregat maksimum data statis dalam partisi logis. Lihat perinciannya di <a href="#">the section called “Hitung ukuran kolom statis per partisi logis”</a> .                                                                                                                                  | 1 MB                     |
| Subquery maks per pernyataan IN SELECT                 | Jumlah maksimum subquery yang dapat Anda gunakan untuk IN kata kunci dalam sebuah SELECT pernyataan. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .                                                                                                           | 100                      |
| Jumlah maksimum koleksi beku bersarang per Wilayah AWS | Jumlah maksimum koleksi bersarang yang didukung saat Anda menggunakan FROZEN kata kunci untuk kolom dengan tipe data koleksi. Untuk informasi lebih lanjut tentang koleksi beku, lihat <a href="#">the section called “Jenis koleksi”</a> . Untuk meningkatkan tingkat bersarang, hubungi Dukungan. | 8                        |
| Jumlah maksimum pernyataan dalam batch yang dicatat    | Jumlah maksimum pernyataan CQL yang didukung saat Anda menggunakan statment batch yang dicatat. Untuk informasi selengkapnya tentang pernyataan batch, lihat <a href="#">the section called “Gunakan pernyataan batch”</a> .                                                                        | 100                      |

| Kuota                                                                        | Deskripsi                                                                                                                                                                                                                                       | Amazon Keyspaces default |
|------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Jumlah maksimum pernyataan dalam batch yang dicatat menggunakan kolom statis | Jumlah maksimum pernyataan CQL yang didukung saat Anda menggunakan kolom statis dalam statment batch yang dicatat. Untuk informasi selengkapnya tentang pernyataan batch, lihat <a href="#">the section called “Gunakan pernyataan batch”</a> . | 50                       |
| Ukuran maksimal dari pernyataan batch yang dicatat                           | Ukuran maksimum statment batch yang dicatat. Untuk informasi selengkapnya tentang pernyataan batch, lihat <a href="#">the section called “Gunakan pernyataan batch”</a> .                                                                       | 4MB                      |
| Jumlah maksimum pernyataan dalam batch yang tidak tercatat                   | Jumlah maksimum pernyataan CQL yang didukung saat Anda menggunakan statment batch yang tidak tercatat. Untuk informasi selengkapnya tentang pernyataan batch, lihat <a href="#">the section called “Gunakan pernyataan batch”</a> .             | 30                       |
| Throughput baca maksimal per detik                                           | Throughput baca maksimum per detik—unit permintaan baca (RRU) atau unit kapasitas baca (RCU) — yang dapat dialokasikan ke tabel per Wilayah. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .               | 40.000                   |

| Kuota                                                                           | Deskripsi                                                                                                                                                                                                                                                       | Amazon Keyspaces default |
|---------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Throughput penulisan maks per detik                                             | Throughput tulis maksimum per detik—unit permintaan tulis (WRU) atau unit kapasitas tulis (WCU) —yang dapat dialokasikan ke tabel per Wilayah. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .                             | 40.000                   |
| Throughput penulisan maksimum untuk tabel menggunakan change-data-capture (CDC) | Kapasitas tulis maksimum untuk tabel Amazon Keyspaces dengan CDC diaktifkan. Kuota default ini hanya berlaku untuk tabel dalam mode kapasitas yang disediakan. read/write Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .  | 40.000                   |
| Account-level baca throughput (disediakan)                                      | Jumlah maksimum unit kapasitas baca agregat (RCU) yang dialokasikan untuk akun per Wilayah. Ini hanya berlaku untuk tabel dalam mode read/write kapasitas yang disediakan. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> . | 80.000                   |

| Kuota                                                             | Deskripsi                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | Amazon Keyspaces default |
|-------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Account-level tulis throughput (disediakan)                       | Jumlah maksimum unit kapasitas tulis agregat (WCU) yang dialokasikan untuk akun per Wilayah. Ini hanya berlaku untuk tabel dalam mode read/write kapasitas yang disediakan. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .                                                                                                                                                                                                                           | 80.000                   |
| Jumlah maksimum target yang dapat diskalakan per Wilayah per akun | Jumlah maksimum target yang dapat diskalakan untuk akun per Wilayah. Tabel Amazon Keyspaces dihitung sebagai satu target yang dapat diskalakan jika penskalaan otomatis diaktifkan untuk kapasitas baca, dan sebagai target lain yang dapat diskalakan jika penskalaan otomatis diaktifkan untuk kapasitas tulis. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> untuk Application Auto Scaling dengan memilih target Scalable untuk Amazon Keyspaces. | 1.500                    |

| Kuota                                                                           | Deskripsi                                                                                                                                                               | Amazon Keyspaces default |
|---------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Ukuran kunci partisi maks                                                       | Ukuran maksimum kunci partisi majemuk. Hingga 3 byte penyimpanan tambahan ditambahkan ke ukuran mentah setiap kolom yang disertakan dalam kunci partisi untuk metadata. | 2048 byte                |
| Ukuran kunci clustering maks                                                    | Ukuran gabungan maksimum dari semua kolom pengelompokan. Hingga 4 byte penyimpanan tambahan ditambahkan ke ukuran mentah setiap kolom pengelompokan untuk metadata.     | 850 byte                 |
| Tabel konkuren maks pulih menggunakan Point-in-time Pemulihan (PITR)            | Jumlah maksimum pemulihan tabel bersamaan menggunakan PITR per pelanggan adalah 4. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> . | 4                        |
| Jumlah maksimum data yang dipulihkan menggunakan pemulihan point-in-time (PITR) | Ukuran maksimum data yang dapat dipulihkan menggunakan PITR dalam waktu 24 jam. Anda dapat menyesuaikan nilai default ini di konsol <a href="#">Service Quotas</a> .    | 5 TB                     |

# Meningkatkan atau mengurangi throughput (untuk tabel yang disediakan)

## Meningkatkan throughput yang disediakan

Anda dapat meningkatkan `ReadCapacityUnits` atau `WriteCapacityUnits` sesering yang diperlukan dengan menggunakan konsol atau `ALTER TABLE` pernyataan. Pengaturan baru tidak berlaku sampai operasi `ALTER TABLE` selesai.

Anda tidak dapat melebihi kuota per akun saat menambahkan kapasitas yang disediakan. Dan Anda dapat meningkatkan kapasitas yang disediakan untuk tabel Anda sebanyak yang Anda butuhkan. Untuk informasi selengkapnya tentang kuota per akun, lihat bagian sebelumnya, [the section called “Kuota layanan Amazon Keyspaces”](#).

## Menurunkan throughput yang disediakan

Untuk setiap tabel dalam `ALTER TABLE` pernyataan, Anda dapat mengurangi `ReadCapacityUnits` atau `WriteCapacityUnits` (atau keduanya). Pengaturan baru tidak berlaku sampai operasi `ALTER TABLE` selesai.

Penurunan diizinkan sampai empat kali, kapan saja per hari. Satu hari ditentukan berdasarkan Waktu Universal Terkoordinasi (UTC). Selain itu, jika tidak ada penurunan dalam satu jam terakhir, penurunan tambahan diperbolehkan. Ini secara efektif membawa jumlah maksimum penurunan dalam sehari menjadi 27 (4 menurun pada jam pertama, dan 1 penurunan untuk masing-masing jendela 1 jam berikutnya dalam sehari).

## Enkripsi Amazon Keyspaces saat istirahat

Anda dapat mengubah opsi enkripsi antara AWS KMS kunci yang AWS dimiliki dan AWS KMS kunci yang dikelola pelanggan hingga empat kali dalam jendela 24 jam, berdasarkan per tabel, mulai dari saat tabel dibuat. Jika tidak ada perubahan dalam enam jam terakhir, perubahan tambahan diperbolehkan. Ini secara efektif membawa jumlah maksimum perubahan dalam sehari menjadi delapan (empat perubahan dalam enam jam pertama, dan satu perubahan untuk masing-masing jendela enam jam berikutnya dalam sehari).

Anda dapat mengubah opsi enkripsi untuk menggunakan AWS KMS kunci yang AWS dimiliki sesering yang diperlukan, bahkan jika kuota sebelumnya telah habis.



Ini adalah kuota kecuali jika Anda meminta jumlah yang lebih tinggi. Untuk meminta peningkatan kuota layanan, lihat [Dukungan](#).

## Kuota dan nilai default untuk tipe yang ditentukan pengguna (UDT) di Amazon Keyspaces

### Kuota UDT Amazon Keyspaces dan nilai default

Tabel berikut berisi kuota dan nilai default yang terkait dengan UDT di Amazon Keyspaces. Informasi tentang kuota mana yang dapat disesuaikan tersedia di konsol [Service Quotas](#), di mana Anda juga dapat meminta kenaikan kuota. Untuk informasi lebih lanjut tentang kuota ini, hubungi AWS Dukungan.

| Kuota                                      | Deskripsi                                                                 | Amazon Keyspaces default |
|--------------------------------------------|---------------------------------------------------------------------------|--------------------------|
| Jumlah maksimal UDT per Wilayah AWS        | Jumlah maksimum UDT di semua ruang kunci untuk pelanggan ini per Wilayah. | 256                      |
| Jumlah maksimum tabel per UDT              | Jumlah maksimum tabel yang dapat mereferensikan UDT yang sama.            | 100                      |
| Jumlah maksimal UDT per tabel              | Jumlah maksimum UDT yang dapat dirujuk tabel.                             | 50                       |
| Tingkat maksimum penyangkutan UDT          | Kedalaman bersarang maksimum yang didukung untuk UDT.                     | 8                        |
| Jumlah maksimal UDT anak langsung per UDT  | Jumlah maksimum UDT anak yang didukung untuk UDT.                         | 10                       |
| Jumlah maksimum UDT induk langsung per UDT | Jumlah maksimum UDT induk yang didukung untuk UDT.                        | 10                       |
| Ukuran skema UDT maks                      | Ukuran maksimum skema untuk UDT.                                          | 25 KB                    |

| Kuota                        | Deskripsi                                       | Amazon Keyspaces default |
|------------------------------|-------------------------------------------------|--------------------------|
| Panjang nama Max UDT         | Jumlah maksimum karakter dalam nama UDT.        | 48                       |
| Panjang nama bidang UDT maks | Jumlah maksimum karakter dalam nama bidang UDT. | 128                      |

# Riwayat dokumen untuk Amazon Keyspaces (untuk Apache Cassandra)

Tabel berikut menjelaskan perubahan penting pada dokumentasi sejak rilis terakhir Amazon Keyspaces (untuk Apache Cassandra). Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke umpan RSS.

- Pembaruan dokumentasi terbaru: 28 Januari 2026

| Perubahan                                                                                                     | Deskripsi                                                                                                                                                                                      | Tanggal           |
|---------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <a href="#">Anda sekarang dapat melakukan pra-pemanasan tabel baru dan yang sudah ada di Amazon Keyspaces</a> | Anda sekarang dapat mengonfigurasi warm-throughput tabel baru dan yang sudah ada di Amazon Keyspaces untuk menghangatkannya terlebih dahulu untuk lonjakan kapasitas puncak yang diantisipasi. | Januari 28, 2026  |
| <a href="#">Amazon Keyspaces sekarang mendukung pengalaman terintegrasi AWS CloudShell</a>                    | Anda sekarang dapat meluncurkan shell pra-otentikasi langsung dari konsol Amazon Keyspaces.                                                                                                    | Januari 26, 2026  |
| <a href="#">Amazon Keyspaces sekarang mendukung batch yang dicatat</a>                                        | Anda sekarang dapat menggabungkan beberapa pernyataan ke dalam batch yang dicatat di Amazon Keyspaces.                                                                                         | November 11, 2025 |
| <a href="#">Perubahan pada sertifikat Amazon Keyspaces TLS</a>                                                | Instruksi yang diperbarui untuk menyertakan sertifikat yang dikeluarkan di bawah rantai Amazon Trust Services (ATS).                                                                           | Oktober 25, 2025  |

|                                                                                            |                                                                                                                                                                                                                                                                                                                                                                                           |                    |
|--------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|
| <a href="#">Amazon Keyspaces sekarang mendukung permintaan IPv4 dan IPv6</a>               | Titik akhir tumpukan ganda Amazon Keyspaces mendukung konfigurasi IPv4 dan IPv6.                                                                                                                                                                                                                                                                                                          | September 30, 2025 |
| <a href="#">NOW dan sekarang UUID didukung di semua pernyataan CQL di Amazon Keyspaces</a> | Anda sekarang dapat menggunakan NOW dan UUID fungsi di semua pernyataan CQL di Amazon Keyspaces.                                                                                                                                                                                                                                                                                          | September 8, 2025  |
| <a href="#">Adaptor Kinesis Streams Amazon Keyspaces sekarang tersedia</a>                 | Sekarang Anda dapat memproses aliran CDC Amazon Keyspaces dengan Kinesis Client Library (KCL).                                                                                                                                                                                                                                                                                            | Juli 2, 2025       |
| <a href="#">Amazon Keyspaces kebijakan terkelola baru</a>                                  | Amazon Keyspaces menambahkan kebijakan terkelola baru yang memberikan izin Keyspaces CDCServiceRolePolicy yang diperlukan ke peran terkait layanan untuk AWSServiceRoleForAmazonKeyspacesCDC mempublikasikan data metrik aliran CDC Amazon Keyspaces atas nama Anda. CloudWatch Untuk informasi selengkapnya, lihat <a href="#">Menggunakan peran untuk aliran CDC Amazon Keyspaces</a> . | Juli 2, 2025       |

[Pembaruan kebijakan terkelola Amazon Keyspaces](#)

Amazon Keyspaces menambahkan izin baru ke `AmazonKeyspacesReadOnlyAccess_v2` dan kebijakan `AmazonKeyspacesReadOnlyAccess` terkelola untuk memungkinkan prinsipal IAM melihat aliran. Amazon Keyspaces memperbarui kebijakan `AmazonKeyspacesFullAccess` terkelola dengan izin untuk membuat peran terkait layanan `AWSServiceRoleForAmazonKeyspacesCDC` yang menerbitkan `CloudWatch` metrik atas nama prinsipal.

Juli 2, 2025

[Aliran pengambilan data perubahan Amazon Keyspaces \(CDC\) sekarang tersedia](#)

Aktifkan aliran CDC untuk merekam perubahan di tabel Amazon Keyspaces Anda.

Juli 2, 2025

[Amazon Keyspaces Replikasi multi-wilayah sekarang mendukung Wilayah Afrika \(Cape Town\).](#)

Anda sekarang dapat menentukan Wilayah Afrika (Cape Town) untuk ruang kunci Multi-wilayah.

Maret 31, 2025

[Amazon Keyspaces Replikasi multi-wilayah sekarang mendukung unlimited. Wilayah AWS](#)

Anda sekarang dapat menentukan Wilayah sebanyak yang Anda inginkan untuk ruang kunci Multi-wilayah.

25 Maret 2025

[Amazon Keyspaces mendukung tipe yang ditentukan pengguna \(UDT\) di ruang kunci Multi-wilayah](#)

Anda sekarang dapat membuat dan menggunakan UDT di ruang kunci Multi-wilayah.

Maret 24, 2025

[Pembaruan kebijakan terkelola Amazon Keyspaces](#)

Amazon Keyspaces menambahkan izin baru ke kebijakan AmazonKey spacesFullAccess terkelola untuk memungkinkan prinsipal IAM menambahkan Wilayah baru ke ruang kunci yang ada. Ini termasuk pembaruan untuk peran terkait layanan. [AWSServiceRoleForAmazonKeyspacesReplication](#)

November 19, 2024

[Tambahkan replika ke tabel Multi-wilayah di Amazon Keyspaces](#)

Anda sekarang dapat menambahkan baru Wilayah AWS ke ruang kunci tunggal dan Multi-wilayah yang ada.

November 19, 2024

[Dukungan untuk tipe yang ditentukan pengguna \(UDT\) di Amazon Keyspaces](#)

Dengan dukungan native untuk UDT di Amazon Keyspaces, Anda sekarang dapat menentukan struktur data dalam aplikasi Anda yang mewakili hierarki data dunia nyata.

Oktober 30, 2024

[ADD COLUMNdukungan untuk replikasi Multi-wilayah Amazon Keyspaces](#)

Amazon Keyspaces sekarang mendukung perubahan skema untuk tabel Multi-region.

September 17, 2024

[Panduan migrasi yang diperbarui untuk Amazon Keyspaces](#)

Panduan migrasi yang diperbarui menguraikan cara membuat rencana migrasi agar berhasil bermigrasi dari Apache Cassandra ke Amazon Keyspaces, termasuk berbagai strategi untuk migrasi offline dan online.

Juni 28, 2024

[Connect ke Amazon Keyspaces dari Amazon Elastic Kubernetes Service](#)

Anda sekarang dapat mengikuti tutorial langkah demi langkah untuk terhubung ke Amazon Keyspaces dari Amazon EKS.

Februari 7, 2024

[Amazon Keyspaces Dukungan replikasi multi-wilayah untuk tabel yang disediakan](#)

Amazon Keyspaces sekarang mendukung mode kapasitas yang disediakan untuk tabel Multi-wilayah.

23 Januari 2024

[API penskalaan otomatis Amazon Keyspaces untuk tabel yang disediakan](#)

Amazon Keyspaces sekarang menawarkan CQL dan dukungan AWS API untuk menyiapkan penskalaan otomatis dengan mode kapasitas yang disediakan.

23 Januari 2024

[Aktivitas DML Amazon Keyspaces disertakan dalam log CloudTrail](#)

Anda sekarang dapat mengaudit panggilan API Amazon Keyspaces Data Manipulation Language (DHTML). AWS CloudTrail

20 Desember 2023

[Dukungan Amazon Keyspaces untuk kata kunci FROZEN](#)

Amazon Keyspaces sekarang mendukung FROZEN kata kunci untuk tipe data pengumpulan.

15 November 2023

[Pembaruan kebijakan terkelola Amazon Keyspaces](#)

Amazon Keyspaces menambahkan izin baru ke kebijakan AmazonKey spacesFullAccess terkelola untuk memungkinkan klien terhubung ke Amazon Keyspaces melalui akses titik akhir VPC antarmuka ke instans Amazon EC2 untuk memperbarui tabel Amazon system.peers Keyspaces dengan informasi jaringan dari VPC.

3 Oktober 2023

[Pembaruan kebijakan terkelola Amazon Keyspaces](#)

Amazon Keyspaces membuat kebijakan AmazonKey spacesReadOnlyAccess\_v2 terkelola baru untuk memungkinkan klien terhubung ke Amazon Keyspaces melalui akses titik akhir VPC antarmuka ke instans Amazon EC2 untuk memperbarui tabel Amazon system.peers Keyspaces dengan informasi jaringan dari VPC.

12 September 2023

[Praktik terbaik untuk membuat koneksi di Amazon Keyspaces](#)

Pelajari cara meningkatkan dan mengoptimalkan konfigurasi driver klien di Amazon Keyspaces.

Juni 30, 2023



|                                                                                          |                                                                                                                                                                                                                  |                |
|------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|
| <a href="#">Ruang kunci sistem sekarang didokumentasikan untuk Amazon Keyspaces</a>      | Pelajari apa yang disimpan di ruang kunci sistem dan cara menanyainya untuk informasi berguna di Amazon Keyspaces.                                                                                               | Juni 21, 2023  |
| <a href="#">Amazon Keyspaces sekarang mendukung replikasi Multi-wilayah</a>              | Amazon Keyspaces Replikasi multi-wilayah membantu Anda memelihara aplikasi yang didistribusikan secara global dengan memberi Anda toleransi kesalahan, stabilitas, dan ketahanan yang lebih baik.                | Juni 5, 2023   |
| <a href="#">Pembaruan kebijakan terkelola Amazon Keyspaces</a>                           | Amazon Keyspaces menambahkan izin baru ke kebijakan AmazonKey spacesFullAccess terkelola untuk memungkinkan Amazon Keyspaces membuat peran terkait layanan saat administrator membuat ruang kunci Multi-wilayah. | Juni 5, 2023   |
| <a href="#">Dukungan Amazon Keyspaces untuk kata kunci IN</a>                            | Amazon Keyspaces sekarang mendukung IN kata kunci dalam SELECT pernyataan.                                                                                                                                       | April 25, 2023 |
| <a href="#">Cross-account akses untuk Amazon Keyspaces dan titik akhir VPC antarmuka</a> | Pelajari cara menerapkan akses lintas akun untuk Amazon Keyspaces dengan titik akhir VPC.                                                                                                                        | 20 April 2023  |

[Dukungan Amazon Keyspaces untuk stempel waktu sisi klien](#)

Stempel waktu sisi klien Amazon Keyspaces adalah stempel waktu Cassandra-compatible tingkat sel yang membantu aplikasi terdistribusi untuk menentukan urutan operasi penulisan ketika klien yang berbeda membuat perubahan pada data yang sama.

14 Maret 2023

[Memulai dengan Amazon Keyspaces dan titik akhir VPC antarmuka](#)

Dalam tutorial langkah demi langkah ini, pelajari cara terhubung ke Amazon Keyspaces dari VPC.

1 Maret 2023

[Mengoptimalkan biaya tabel Amazon Keyspaces](#)

Praktik dan panduan terbaik tersedia untuk membantu Anda mengidentifikasi strategi untuk mengoptimalkan biaya tabel Amazon Keyspaces yang ada.

17 Februari 2023

[Sekarang default Murmur3Partitioner](#)

Sekarang partisi default di Amazon Keyspaces. `Murmur3Partitioner`

17 November 2022

[Amazon Keyspaces sekarang mendukung Murmur3Partitioner](#)

Sekarang `Murmur3Partitioner` tersedia di Amazon Keyspaces.

9 November 2022

[Support update untuk string kosong dan nilai blob](#)

Amazon Keyspaces sekarang juga mendukung string kosong dan nilai gumpalan sebagai nilai kolom pengelompokan.

Oktober 19, 2022

[Amazon Keyspaces sekarang tersedia di AWS GovCloud \(US\)](#)

Amazon Keyspaces sekarang tersedia di AWS GovCloud (US) Region dan berada dalam lingkup kepatuhan . FedRAMP-High Untuk informasi tentang titik akhir yang tersedia, lihat titik akhir [AWS GovCloud \(US\) Region FIPS](#).

Agustus 4, 2022

[Pantau biaya penyimpanan tabel Amazon Keyspaces dengan Amazon CloudWatch](#)

Amazon Keyspaces sekarang membantu Anda memantau dan melacak biaya penyimpanan tabel dari waktu ke waktu dengan metrik. `BillableTableSizeInBytes` CloudWatch

14 Juni 2022

[Amazon Keyspaces sekarang mendukung Terraform](#)

Anda sekarang dapat menggunakan Terraform untuk melakukan operasi bahasa definisi data (DDL) di Amazon Keyspaces.

9 Juni 2022

[Dukungan fungsi Amazon Keyspaces token](#)

Amazon Keyspaces sekarang membantu Anda mengoptimalkan kueri aplikasi dengan menggunakan fungsi. `token`

19 April 2022

[Integrasi Amazon Keyspaces dengan Apache Spark](#)

[Amazon Keyspaces sekarang membantu Anda membaca dan menulis data di Apache Spark dengan lebih mudah dengan menggunakan Spark Cassandra Connector open-source.](#)

19 April 2022

[Referensi API Amazon Keyspaces](#)

Amazon Keyspaces mendukung operasi bidang kontrol untuk mengelola ruang kunci dan tabel menggunakan SDK dan. AWS CLI Panduan referensi API menjelaskan operasi bidang kontrol yang didukung secara rinci.

2 Maret 2022

[Cara memecahkan masalah konfigurasi umum saat menggunakan Amazon Keyspaces.](#)

Pelajari lebih lanjut tentang cara mengatasi masalah konfigurasi umum yang mungkin Anda temui saat menggunakan Amazon Keyspaces.

22 November 2021

[Amazon Keyspaces mendukung Time to Live \(TTL\).](#)

Amazon Keyspaces Time to Live (TTL) membantu Anda menyederhanakan logika aplikasi dan mengoptimalkan harga penyimpanan dengan kedaluwarsa data dari tabel secara otomatis.

Oktober 18, 2021

[Memigrasi data ke Amazon Keyspaces menggunakan DSBulk.](#)

Step-by-step tutorial untuk memigrasi data dari Apache Cassandra ke Amazon Keyspaces menggunakan Bulk Loader (DSBulk) DataStax .

9 Agustus 2021

[Amazon Keyspaces mendukung entri Endpoint VPC dalam tabel. system.peers](#)

Amazon Keyspaces memungkinkan Anda mengisi system.peers tabel dengan informasi titik akhir VPC antarmuka yang tersedia untuk meningkatkan penyeimbangan beban dan meningkatkan throughput. read/write

29 Juli 2021

[Perbarui ke kebijakan terkelola IAM untuk mendukung AWS KMS kunci terkelola pelanggan.](#)

Kebijakan terkelola IAM untuk Amazon Keyspaces sekarang menyertakan izin untuk membuat daftar dan melihat kunci AWS KMS terkelola pelanggan yang tersedia yang disimpan. AWS KMS

1 Juni 2021

[Dukungan Amazon Keyspaces untuk kunci yang dikelola AWS KMS pelanggan.](#)

Amazon Keyspaces memungkinkan Anda mengendalikan AWS KMS kunci terkelola pelanggan yang disimpan AWS KMS untuk enkripsi saat istirahat.

1 Juni 2021

[Dukungan Amazon Keyspaces untuk sintaks JSON](#)

Amazon Keyspaces membantu Anda membaca dan menulis dokumen JSON dengan lebih mudah dengan mendukung sintaks JSON untuk operasi INSERT dan SELECT.

21 Januari 2021

[Dukungan Amazon Keyspaces untuk kolom statis](#)

Amazon Keyspaces sekarang membantu Anda memperbarui dan menyimpan data umum di antara beberapa baris secara efisien dengan menggunakan kolom statis.

9 November 2020

[Rilis GA dari dukungan NoSQL Workbench untuk Amazon Keyspaces](#)

NoSQL Workbench adalah aplikasi sisi klien yang membantu Anda merancang dan memvisualisasikan model data nonrelasional untuk Amazon Keyspaces dengan lebih mudah. Klien NoSQL Workbench tersedia untuk Windows, macOS, dan Linux.

28 Oktober 2020

[Pratinjau rilis dukungan NoSQL Workbench untuk Amazon Keyspaces](#)

NoSQL Workbench adalah aplikasi sisi klien yang membantu Anda merancang dan memvisualisasikan model data nonrelasional untuk Amazon Keyspaces dengan lebih mudah. Klien NoSQL Workbench tersedia untuk Windows, macOS, dan Linux.

5 Oktober 2020

[Contoh kode baru untuk akses terprogram ke Amazon Keyspaces](#)

Kami terus menambahkan contoh kode untuk akses terprogram ke Amazon Keyspaces. Sampel sekarang tersedia untuk driver Java, Python, Go, C #, dan Perl Cassandra yang mendukung Apache Cassandra versi 3.11.2.

17 Juli 2020

[Pemulihan point-in-time](#)[Amazon Keyspaces \(PITR\)](#)

Amazon Keyspaces sekarang menawarkan pemulihan point-in-time (PITR) untuk membantu melindungi tabel Anda dari operasi tulis atau penghapusan yang tidak disengaja dengan menyediakan pencadangan data tabel Anda secara terus menerus.

9 Juli 2020

[Ketersediaan umum Amazon Keyspaces](#)

Dengan Amazon Keyspaces, yang sebelumnya dikenal selama pratinjau sebagai Amazon Managed Apache Cassandra Service (MCS), Anda dapat menggunakan kode Cassandra Query Language (CQL), driver Cassandra berlisensi Apache 2.0, dan alat pengembang yang sudah Anda gunakan saat ini.

23 April 2020

[Penskalaan otomatis Amazon Keyspaces](#)

Amazon Keyspaces (untuk Apache Cassandra) terintegrasi dengan Application Auto Scaling untuk membantu Anda menyediakan kapasitas throughput secara efisien untuk beban kerja variabel sebagai respons terhadap lalu lintas aplikasi aktual dengan menyesuaikan kapasitas throughput secara otomatis.

23 April 2020

|                                                                                          |                                                                                                                                              |               |
|------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| <a href="#">Antarmuka titik akhir cloud pribadi virtual (VPC) untuk Amazon Keyspaces</a> | Amazon Keyspaces menawarkan komunikasi pribadi antara layanan dan VPC Anda sehingga lalu lintas jaringan tidak meninggalkan jaringan Amazon. | 16 April 2020 |
| <a href="#">Tag-based kebijakan akses</a>                                                | Anda sekarang dapat menggunakan tag sumber daya dalam kebijakan IAM untuk mengelola akses ke Amazon Keyspaces.                               | 8 April 2020  |
| <a href="#">Kontra tipe data</a>                                                         | Amazon Keyspaces sekarang membantu Anda mengoordinasikan penambahan dan penurunan nilai kolom dengan menggunakan penghitung.                 | 7 April 2020  |
| <a href="#">Sumber daya penandaan</a>                                                    | Amazon Keyspaces sekarang memungkinkan Anda memberi label dan mengkategorikan sumber daya dengan menggunakan tag.                            | 31 Maret 2020 |
| <a href="#">CloudFormation dukungan</a>                                                  | Amazon Keyspaces sekarang membantu Anda mengotomatiskan pembuatan dan pengelolaan sumber daya dengan menggunakan CloudFormation              | 25 Maret 2020 |



### [Support untuk peran dan kebijakan IAM dan otentikasi SiGv4](#)

Menambahkan informasi tentang bagaimana Anda dapat menggunakan [AWS Identity and Access Management \(IAM\)](#) untuk mengelola izin akses dan menerapkan kebijakan keamanan untuk Amazon Keyspaces dan cara menggunakan plugin otentikasi untuk Driver DataStax Java untuk Cassandra untuk mengakses Amazon Keyspaces secara terprogram menggunakan peran IAM dan identitas gabungan.

17 Maret 2020

### [Read/write modus kapasitas](#)

Amazon Keyspaces sekarang mendukung dua mode read/write kapasitas throughput. Mode read/write kapasitas mengontrol cara Anda dikenakan biaya untuk throughput baca dan tulis dan bagaimana kapasitas throughput tabel dikelola.

20 Februari 2020

### [Rilis awal](#)

Dokumentasi ini mencakup rilis awal Amazon Keyspaces (untuk Apache Cassandra).

3 Desember 2019

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.