EBook Gratis

APRENDIZAJE cassandra

Free unaffiliated eBook created from
Stack Overflow contributors.

#cassandra

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: cassandra

It is an unofficial and free cassandra ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official cassandra.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con Cassandra

Observaciones

La base de datos de Apache Cassandra es la opción correcta cuando necesita escalabilidad y alta disponibilidad sin comprometer el rendimiento. La escalabilidad lineal y la probada tolerancia a fallas en el hardware básico o la infraestructura en la nube la convierten en la plataforma perfecta para datos de misión crítica. El soporte de Cassandra para la replicación en múltiples centros de datos es el mejor de su clase, brindando una menor latencia para sus usuarios y la tranquilidad de saber que puede sobrevivir a las interrupciones regionales.

PROBADO

Cassandra está en uso en Constant Contact, CERN, Comcast, eBay, GitHub, GoDaddy, Hulu, Instagram, Intuit, Netflix, Reddit, The Weather Channel y más de 1500 compañías más que tienen conjuntos de datos grandes y activos.

TOLERANTE A FALLOS

Los datos se replican automáticamente en varios nodos para tolerancia a fallos. Se admite la replicación en varios centros de datos. Los nodos fallidos se pueden reemplazar sin tiempo de inactividad.

RENDIMIENTO

Cassandra siempre supera las alternativas populares de NoSQL en los puntos de referencia y las aplicaciones reales, principalmente debido a elecciones arquitectónicas fundamentales.

DESCENTRALIZADO

No hay puntos únicos de fracaso. No hay cuellos de botella en la red. Cada nodo en el clúster es idéntico.

ESCALABLE

Algunas de las implementaciones de producción más grandes incluyen Apple, con más de 75,000 nodos que almacenan más de 10 PB de datos, Netflix (2,500 nodos, 420 TB, más de 1 billón de solicitudes por día), el motor de búsqueda chino Easou (270 nodos, 300 TB, más de 800 millones requisitos por día) y eBay (más de 100 nodos, 250 TB).

DURABLE

Cassandra es adecuada para aplicaciones que no pueden permitirse perder datos, incluso cuando un centro de datos completo deja de funcionar.

Estas en control

Elija entre la replicación síncrona o asíncrona para cada actualización. Las operaciones

asíncronas de alta disponibilidad están optimizadas con funciones como Hinted Handoff y Read Repair.

ELÁSTICO

El rendimiento de lectura y escritura aumenta linealmente a medida que se agregan nuevas máquinas, sin interrupciones ni interrupciones en las aplicaciones.

APOYO PROFESIONAL

Los contratos y servicios de soporte de Cassandra están disponibles a través de terceros.

Versiones

Versión	Fecha de lanzamiento
1.1.12	2013-11-19
1.1.9	2013-02-11
1.2.12	2013-11-28
1.2.13	2013-12-19
1.2.15	2014-02-19
1.2.16	2014-04-22
1.2.17	2014-06-25
1.2.18	2014-07-04
1.2.19	2014-11-14
1.2.6	2013-07-02
1.2.8	2013-07-27
2.0.10	2014-08-12
2.0.11	2014-10-17
2.0.12	2015-01-14
2.0.13	2015-03-20
2.0.14	2015-04-01
2.0.15	2015-06-01

Versión	Fecha de lanzamiento
2.0.16	2015-07-08
2.0.17	2015-09-18
2.0.5	2014-02-13
2.0.6	2014-04-02
2.0.7	2014-04-24
2.0.8	2014-06-13
2.0.9	2014-07-22
2.1.11	2015-10-12
2.1.12	2015-10-22
2.1.2	2014-11-20
2.1.3	2015-03-03
2.1.4	2015-04-01
2.1.5	2015-03-31
2.1.6	2015-06-09
2.1.7	2015-06-18
2.1.8	2015-07-03
2.1.9	2015-09-03
2.2.0	2015-05-14
2.2.0-beta1	2015-05-19
2.2.0-rc1	2015-06-04
2.2.0-rc2	2015-06-30
2.2.1	2015-08-25
2.2.2	2015-09-25
2.2.3	2015-10-12
2.2.4	2015-12-02

Versión	Fecha de lanzamiento
3.0.0	2015-01-26
3.0.0-alfa	2015-07-29
3.0.0-alfa1	2015-07-18
3.0.0-beta1	2015-07-10
3.0.0-beta2	2015-09-04
3.0.0-rc1	2015-07-16
3.0.0-rc2	2015-10-16
3.0.1	2015-12-04
3.0.2	2016-01-21
3.0.3	2015-11-24
3.0.4	2016-02-05
3.0.5	2016-04-02
3.0.6	2016-03-31
3.0.7	2016-05-24
3.0.8	2016-05-25
3.2.819	2016-01-05
3.4.950	2016-03-08
3.6.1076	2016-05-02
3.8.1199	2016-06-27
3.10.3004	2016-08-10

(Obtuve esto usando un poco de awk: `git log --tags --simplify-by-decoration --pretty="format:%ai %d" |egrep "\(tag: [0-9]"| awk -F" " '{ print $1 " " $5}'|awk -F"." '{print $1 "." $2 "." $3}'| awk -F" " '{print $2 " |" $1}'| sed 's/)//'|sed 's/,//'| sort -n|sort -u -t" " -k1,1 | awk '{print "|" $0 "|"}'`)

Examples

Instalación o configuración

Instalación de un solo nodo

1. Preinstalar NodeJS, Python y Java
2. Seleccione su documento de instalación en función de su plataforma http://docs.datastax.com/en/cassandra/3.x/cassandra/install/installTOC.html
3. Descargue los binarios de Cassandra desde http://cassandra.apache.org/download/
4. Descomprima el archivo descargado en `<installation location>`
5. Inicie el cassandra usando `<installation location>/bin/cassandra` O inicie Cassandra como servicio - `[sudo] service cassandra start`
6. Verifique si Cassandra está en funcionamiento usando `<installation location>/bin/nodetool status` .

Ex:

1. En el entorno de Windows, ejecute el archivo `cassandra.bat` para iniciar el servidor Cassandra y `cqlsh.bat` para abrir el terminal cliente CQL para ejecutar los comandos CQL.

Hay dos formas en que se puede llevar a cabo la instalación de un **solo nodo** .

Debe tener Oracle Java 8 o OpenJDk 8 (preferido para las versiones de Cassandra> 3.0)

1. Instalar un paquete Debian (instala Cassandra como un servicio)

Agregue la versión de Cassandra al repositorio (reemplace el 22x con su propia versión, por ejemplo, para 2.7 use 27x)

```
echo "deb-src http://www.apache.org/dist/cassandra/debian 22x main" | sudo tee -a
/etc/apt/sources.list.d/cassandra.sources.list
# Update the repository
sudo apt-get update
# Then install it
sudo apt-get install cassandra cassandra-tools
```

Ahora Cassandra se puede iniciar y dejar de usar:

```
sudo service cassandra start
sudo service cassandra stop
```

Compruebe el estado utilizando:

```
nodetool status
```

Los directorios de registros y datos son `/var/log/cassandra` y `/var/lib/cassandra` respectivamente.

2. Instalar cualquier versión de Cassandra en forma de archivo binario (instala Cassandra como un proceso independiente)

Descarga la versión Datastax:

```
curl -L  http://downloads.datastax.com/community/dsc-cassandra-version_number-bin.tar.gz | tar xz
```

O Apache Cassandra tarball binario manualmente (desde el sitio http://www.apache.org/dist/cassandra/)

Ahora untar esto:

```
tar -xvzf dsc-cassandra-version_number-bin.tar.gz
```

Cambie el directorio para instalar la ubicación:

```
cd install_location
```

Inicia Cassandra usando:

```
sudo sh ./bin/cassandra
```

Dejar de usar:

```
sudo kill -9 pid
```

Comprobar:

```
./bin/nodetool status
```

Y viola, tienes un clúster de prueba de un solo nodo para Cassandra. Así que solo usa `cqlsh` en el terminal para el shell de Cassandra.

La configuración de Cassandra se puede hacer en `cassandra.yaml` en la carpeta `conf` en `install_location` .

Instalación multi nodo

Instalación de Multi DC Cluster

Lea Empezando con Cassandra en línea:
https://riptutorial.com/es/cassandra/topic/1682/empezando-con-cassandra

Capítulo 2: Casandra como servicio

Introducción

Este tema describe cómo iniciar Apache Cassandra como un servicio en plataformas Windows y Linux. Recuerde que también inicia Cassandra desde el directorio bin ejecutando el script por lotes o shell.

Examples

Windows

1. Descargue el último daemon de apache commons de Apache Commons Project Distributions .
2. Extraiga el demonio común en **<directorio de Cassandra instalado> \ bin** .
3. Renombra la carpeta extraída como daemon.
4. Agregue **<directorio de Cassandra instalado>** como **CASSANDRA_HOME** en la variable de entorno de Windows.
5. Edite el archivo **cassandra.yaml** en **<Directorio de Cassandra instalado> \ conf** y elimine los comentarios de los directorios_de_archivos_de_datos, directorio_inicial de registro, directorio_de_caché guardado y configure las rutas absolutas.
6. Edite **cassandra.bat** en **<directorio de Cassandra instalado> \ bin** y reemplace el valor de PATH_PRUNSRV de la siguiente manera:

```
for 32 bit windows, set PATH_PRUNSRV=%CASSANDRA_HOME%\bin\daemon\
for 64 bit windows, set PATH_PRUNSRV=%CASSANDRA_HOME%\bin\daemon\amd64\
```

7. Edite **cassandra.bat** y configure SERVICE_JVM para el nombre del servicio requerido.

```
SERVICE_JVM="cassandra"
```

8. Con privilegios de administrador, ejecute cassandra.bat install desde el símbolo del sistema.

Linux

1. Cree el script de inicio /etc/init.d/cassandra.
2. Editar el contenido del archivo:

```
#!/bin/sh
#
```

```
# chkconfig: - 80 45
# description: Starts and stops Cassandra
# update daemon path to point to the cassandra executable
DAEMON=<Cassandra installed directory>/bin/cassandra
start() {
        echo -n "Starting Cassandra... "
        $DAEMON -p /var/run/cassandra.pid
        echo "OK"
        return 0
}
stop() {
        echo -n "Stopping Cassandra... "
        kill $(cat /var/run/cassandra.pid)
        echo "OK"
        return 0
}
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart)
        stop
        start
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart}"
        exit 1
esac
exit $?
```

3. Hacer el archivo ejecutable:

   sudo chmod + x /etc/init.d/cassandra

4. Agregue el nuevo servicio a la lista:

   sudo chkconfig --add cassandra

5. Ahora puedes administrar el servicio desde la línea de comando:

```
sudo /etc/init.d/cassandra start
sudo /etc/init.d/cassandra stop
sudo /etc/init.d/cassandra restart
```

Lea Casandra como servicio en línea: https://riptutorial.com/es/cassandra/topic/10544/casandra-como-servicio

Capítulo 3: Cassandra - PHP

Examples

Aplicación de consola simple

Descargue el controlador Datastax PHP para Apache Cassandra desde el sitio del proyecto Git y siga las instrucciones de instalación.

Utilizaremos la tabla de **"usuarios"** de la aplicación **KillrVideo** y el controlador Datastax de PHP. Una vez que tengas a Cassandra en funcionamiento, crea el siguiente espacio de teclas y tabla:

```
//Create the keyspace
    CREATE KEYSPACE killrvideo WITH REPLICATION =
    { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };

//Use the keyspace
USE killrvideo;

// Users keyed by id
CREATE TABLE users (
    userid uuid,
    firstname text,
    lastname text,
    email text,
    created_date timestamp,
    PRIMARY KEY (userid)
);
```

Crea un archivo PHP con tu editor favorito. Primero, debemos conectarnos a nuestro **espacio de** claves y clúster **"killrvideo"** :

```
build();
$keyspace  = 'killrvideo';
$session  = $cluster->connect($keyspace);
```

Insertemos un usuario en la tabla de **"usuarios"** :

```
execute(new Cassandra\SimpleStatement(
    "INSERT INTO users (userid, created_date, email, firstname, lastname)
      VALUES (14c532ac-f5ae-479a-9d0a-36604732e01d, '2013-01-01 00:00:00',
      'luke@example.com','Luke','Tillman')"
  ));
```

Usando el controlador Datastax PHP Cassandra, podemos consultar al usuario por ID de usuario:

```
execute(new Cassandra\SimpleStatement
          ("SELECT firstname, lastname, email FROM killrvideo.users
            WHERE userid=14c532ac-f5ae-479a-9d0a-36604732e01d"));

foreach ($result as $row) {
```

```
    printf("user: \"%s\" \"%s\" email: \"%s\" \n", $row['firstname'],
    $row['lastname'], $row['email']);
  }
```

Para que un usuario actualice su dirección de correo electrónico en el sistema:

```
execute(new Cassandra\SimpleStatement
          ("UPDATE users SET email = 'language_evangelist@example.com'
            WHERE userid = 14c532ac-f5ae-479a-9d0a-36604732e01d"));

execute(new Cassandra\SimpleStatement
          ("SELECT firstname, lastname, email FROM killrvideo.users
            WHERE userid=14c532ac-f5ae-479a-9d0a-36604732e01d"));

foreach ($result as $row) {
    printf("user: \"%s\" \"%s\" email: \"%s\" \n", $row['firstname'],
    $row['lastname'], $row['email']);
  }
```

Eliminar el usuario de la tabla y generar todas las filas. Notarás que la información del usuario ya no vuelve después de ser eliminada:

```
execute(new Cassandra\SimpleStatement
          ("DELETE FROM users WHERE userid = 14c532ac-f5ae-479a-9d0a-36604732e01d"));

execute(new Cassandra\SimpleStatement
          ("SELECT firstname, lastname, email FROM killrvideo.users
            WHERE userid=14c532ac-f5ae-479a-9d0a-36604732e01d"));

foreach ($result as $row) {
    printf("user: \"%s\" \"%s\" email: \"%s\" \n", $row['firstname'],
    $row['lastname'], $row['email']);
  }
```

La salida debe verse algo como esto:

```
user: "Luke" "Tillman" email: "luke@example.com"
user: "Luke" "Tillman" email: "language_evangelist@example.com"
```

Referencias :

Comenzando con Apache Cassandra y PHP , DataStax Academy

Lea Cassandra - PHP en línea: https://riptutorial.com/es/cassandra/topic/2866/cassandra---php

Capítulo 4: Conectando a Cassandra

Observaciones

El controlador Cassandra de Datastax refleja en gran medida el controlador Java JDBC MySQL.

`Session` , `Statement` , `PreparedStatement` están presentes en ambos controladores.

La conexión Singleton es de esta pregunta y respuesta:
http://stackoverflow.com/a/24691456/671896

En cuanto a las características, Cassandra 2 y 3 son idénticas. Cassandra 3 introdujo una reescritura completa del sistema de almacenamiento de datos.

Examples

Java: incluye el controlador DSE de Cassandra

En su proyecto Maven, agregue lo siguiente a su archivo `pom.xml` . Las siguientes versiones son para Cassandra 3.x.

```
    <dependency>
        <groupId>com.datastax.cassandra</groupId>
        <artifactId>cassandra-driver-core</artifactId>
        <version>3.1.0</version>
    </dependency>
    <dependency>
        <groupId>com.datastax.cassandra</groupId>
        <artifactId>cassandra-driver-mapping</artifactId>
        <version>3.1.0</version>
    </dependency>
    <dependency>
        <groupId>com.datastax.cassandra</groupId>
        <artifactId>cassandra-driver-extras</artifactId>
        <version>3.1.0</version>
    </dependency>
    <dependency>
        <groupId>com.datastax.cassandra</groupId>
        <artifactId>dse-driver</artifactId>
        <version>1.1.0</version>
    </dependency>
```

Java: conectarse a una instancia local de Cassandra

Conectarse a Cassandra es muy similar a conectarse a otras fuentes de datos. Con Cassandra, no se requieren credenciales.

```
String cassandraIPAddress = "127.0.0.1";
String cassandraKeyspace = "myKeyspace";
String username = "foo";
```

```
String password = "bar";

com.datastax.driver.core.Cluster cluster = Cluster.builder()
    .addContactPoint(cassandraIPAddress)
    .withCredentials(username, password) // If you have setup a username and password for your
node.
    .build();

com.datastax.driver.core.Session session = cluster.connect(cassandraKeyspace);

com.datastax.driver.core.Metadata metadata = cluster.getMetadata();

// Output Cassandra connection status
System.out.println("Connected to Cassandra cluster: " + metadata.getClusterName() + " with
Partitioner: " + metadata.getPartitioner());

// Loop through your entire Cluster.
for (Host host : metadata.getAllHosts()) {
    System.out.println("Cassandra Host Address: " + host.getAddress() + " | Is Up = " +
host.isUp());
}
```

Java: conectar utilizando un Singleton

```
public enum Cassandra {

    DB;

    private Session session;
    private Cluster cluster;
    private static final Logger LOGGER = LoggerFactory.getLogger(Cassandra.class);

    /**
     * Connect to the cassandra database based on the connection configuration provided.
     * Multiple call to this method will have no effects if a connection is already
established
     * @param conf the configuration for the connection
     */
    public void connect(ConnectionCfg conf) {
        if (cluster == null && session == null) {
            cluster =
Cluster.builder().withPort(conf.getPort()).withCredentials(conf.getUsername(),
conf.getPassword()).addContactPoints(conf.getSeeds()).build();
            session = cluster.connect(conf.getKeyspace());
        }
        Metadata metadata = cluster.getMetadata();
        LOGGER.info("Connected to cluster: " + metadata.getClusterName() + " with partitioner:
" + metadata.getPartitioner());
        metadata.getAllHosts().stream().forEach((host) -> {
            LOGGER.info("Cassandra datacenter: " + host.getDatacenter() + " | address: " +
host.getAddress() + " | rack: " + host.getRack());
        });
    }

    /**
     * Invalidate and close the session and connection to the cassandra database
     */
    public void shutdown() {
        LOGGER.info("Shutting down the whole cassandra cluster");
```

```
        if (null != session) {
            session.close();
        }
        if (null != cluster) {
            cluster.close();
        }
    }

    public Session getSession() {
        if (session == null) {
            throw new IllegalStateException("No connection initialized");
        }
        return session;
    }
}
```

Usando la conexión Singleton

```
public void cassandra() throws Exception {
    Cassandra.DB.connect();
    Cassandra.DB.getSession().execute(/* CQL | Statement | PreparedStatement */)
    Cassandra.DB.close();
}
```

Lea Conectando a Cassandra en línea:
https://riptutorial.com/es/cassandra/topic/7845/conectando-a-cassandra

Capítulo 5: Correr Reparación en Cassandra

Sintaxis

- **Sinopsis**
- `nodetool [node-options] repair [other-options]`
- **Opciones de nodo**
- `[(-h <host> | --host <host>)]`
- `[(-p <port> | --port <port>)]`
- `[(-pw <password> | --password <password>)]`
- `[(-pwf <passwordFilePath> | --password-file <passwordFilePath>)]`
- `[(-u <username> | --username <username>)]`
- **Otras opciones**
- `[(-dc <specific_dc> | --in-dc <specific_dc>)...]`
- `[(-dcpar | --dc-parallel)]`
- `[(-et <end_token> | --end-token <end_token>)]`
- `[(-full | --full)]`
- `[(-hosts <specific_host> | --in-hosts <specific_host>)...]`
- `[(-j <job_threads> | --job-threads <job_threads>)]`
- `[(-local | --in-local-dc)]`
- `[(-pl | --pull)]`
- `[(-pr | --partitioner-range)]`
- `[(-seq | --sequential)]`
- `[(-st <start_token> | --start-token <start_token>)]`
- `[(-tr | --trace)]`
- `[--]`
- `[<keyspace> <tables>...]`

Parámetros

Parámetro	Detalles
`-dc <specific_dc>` , `--in-dc <specific_dc>`	Use `-dc` para reparar centros de datos específicos
`-dcpar` , `--dc-parallel`	Utilice `-dcpar` para reparar centros de datos en paralelo.
`-et <end_token>` , `--end-token <end_token>`	Utilice `-et` para especificar un token en el que finaliza el rango de reparación
`-full` , `--full`	Utilice `-full` para emitir una reparación completa.
`-h <host>` , `--host <host>`	Nombre de host del nodo o dirección IP
`-hosts <specific_host>` , `--in-hosts <specific_host>`	Utilice `-hosts` para reparar hosts específicos
`-j <job_threads>` , `--job-threads <job_threads>`	Número de hilos para ejecutar trabajos de reparación. Por lo general, esto significa el número de CF para reparar simultáneamente. ADVERTENCIA: aumentar esto aumenta la carga de los nodos de reparación, así que tenga cuidado. (por defecto: 1, max: 4)
`-local` , - en `--in-local-dc`	Use `-local` para reparar solo contra nodos en el mismo centro de datos
`-p <port>` , `--port <port>`	Número de puerto del agente jmx remoto
`-pl` , `--pull`	Use `--pull` para realizar una reparación en un solo sentido donde los datos solo se transmiten desde un nodo remoto a este nodo.
`-pr` , `--partitioner-range`	Use `-pr` para reparar solo el primer rango devuelto por el particionador
`-pw <password>` , `--password <password>`	Contraseña del agente jmx remoto
`-pwf <passwordFilePath>` , `--password-file <passwordFilePath>`	Ruta al archivo de contraseñas JMX
`-seq` , `--sequential`	Utilice `-seq` para realizar una reparación secuencial.
`-st <start_token>` , `--start-token <start_token>`	Utilice `-st` para especificar un token en el que comienza el rango de reparación
`-tr` , `--trace`	Utilice `-tr` para rastrear la reparación. Las trazas se registran en `system_traces.events` .
`-u <username>` , `--username <username>`	Nombre de usuario de agente jmx remoto

Parámetro	Detalles
`--`	Esta opción se puede usar para separar las opciones de la línea de comandos de la lista de argumentos (útil cuando los argumentos pueden confundirse con las opciones de la línea de comandos
`[<keyspace> <tables>...]`	El espacio de teclas seguido por una o varias tablas.

Examples

Correr reparación en Cassandra

Ejecutar reparación en un rango de partición particular.

```
nodetool repair -pr
```

Ejecutar reparación en todo el clúster.

```
nodetool repair
```

Ejecutar reparación en modo paralelo.

```
nodetool repair -par
```

Lea Correr Reparación en Cassandra en línea:
https://riptutorial.com/es/cassandra/topic/6556/correr-reparacion-en-cassandra

Capítulo 6: Llaves de casandra

Examples

Clave de partición, clave de clustering, clave principal

Cassandra usa dos tipos de llaves:

- Las **claves de partición** son responsables de la distribución de datos entre nodos
- La **clave de agrupación** es responsable de la clasificación de datos dentro de una partición

Una **clave principal** es una combinación de los tipos. El vocabulario depende de la combinación:

- **clave principal simple** : solo la clave de partición, compuesta de una columna
- **clave de partición compuesta** : solo la clave de partición, compuesta de varias columnas
- **clave principal compuesta** : una clave de partición con una o más claves de agrupamiento.
- **clave principal compuesta y compuesta** : una clave de partición compuesta de varias columnas y múltiples claves de agrupamiento.

La sintaxis de PRIMARY KEY

Declarando una clave

La declaración de creación de la tabla debe contener una expresión `PRIMARY KEY` . La forma en que lo declaras es muy importante. En una palabra:

```
PRIMARY KEY(partition key)
PRIMARY KEY(partition key, clustering key)
```

Los paréntesis adicionales agrupan varios campos en una clave de partición compuesta o declaran una clave compuesta compuesta.

Ejemplos

Clave primaria **simple** :

```
PRIMARY KEY (key)
```

`key` se llama la **clave de partición** .

(para una clave principal simple, también es posible colocar la expresión `PRIMARY KEY` después del campo, es decir, la `key int PRIMARY KEY,` por ejemplo).

Clave primaria **compuesta** :

```
PRIMARY KEY (key_part_1, key_part_2)
```

Contrariamente a SQL, esto no crea exactamente una clave primaria compuesta. En su lugar, declara `key_part_1` como la **clave de partición** y `key_part_2` como la **clave de agrupación** . Cualquier otro campo también se considerará parte de la clave de agrupación.

Compuesto + claves primarias **compuestas** :

```
PRIMARY KEY ((part_key_1, ..., part_key_n), (clust_key_1, ..., clust_key_n))
```

El primer paréntesis define la **clave de partición compuesta** , las otras columnas son las claves de agrupamiento.

Resumen de sintaxis

- `(part_key)`
- `(part_key, clust_key)`
- `(part_key, clust_key_1, clust_key_2)`
- `(part_key, (clust_key_1, clust_key_2))`
- `((part_key_1, part_key_2), clust_key)`
- `((part_key_1, part_key_2), (clust_key_1, clust_key_2))`

Ordenamiento de claves y consultas permitidas.

La **clave de partición** es el **especificador mínimo** necesario para realizar una consulta utilizando una cláusula where.

Si declara una **clave de agrupación compuesta** , el orden importa.

Digamos que tienes la siguiente clave principal:

```
PRIMARY KEY((part_key1, part_key_2), (clust_key_1, clust_key_2, clust_key_3))
```

Entonces, las *únicas consultas válidas* usan los siguientes campos en la cláusula `where` :

- `part_key_1` , `part_key_2`
- `part_key_1` , `part_key_2` , `clust_key_1`
- `part_key_1` , `part_key_2` , `clust_key_1` , `clust_key_2`
- `part_key_1` , `part_key_2` , `clust_key_1` , `clust_key_2` , `clust_key_3`

Ejemplos de consultas inválidas son:

- `part_key_1` , `part_key_2` , `clust_key_2`
- Cualquier cosa que no contenga `part_key_1` , `part_key_2`
- ...

Si desea utilizar `clust_key_2` , también debe especificar `clust_key_1` , y así sucesivamente.

Por lo tanto, el orden en el que declara sus claves de agrupamiento tendrá un impacto en el tipo

de consultas que puede hacer. En el sentido contrario, el orden de los campos de clave de partición no es importante, ya que siempre debe especificarlos todos en una consulta.

Lea Llaves de casandra en línea: https://riptutorial.com/es/cassandra/topic/9071/llaves-de-casandra

Capítulo 7: Reparaciones en casandra

Parámetros

Opción / Bandera	Descripción
opción	*descripción de la opción*
-h	nombre de host El valor predeterminado es "localhost". Si no especifica una reparación de host se ejecuta en el mismo host desde el que se ejecuta el comando.
-pag	Puerto JMX. El valor predeterminado es 7199.
-u	nombre de usuario Solo es necesario si la seguridad JMX está habilitada.
-pw	contraseña. Solo es necesario si la seguridad JMX está habilitada.
bandera	*descripción de la bandera*
-local	Solo compare y transmita datos desde nodos en el centro de datos "local".
-pr	Reparación de la "gama de particiones". Sólo repara el rango de token primario para una réplica. Más rápido que reparar todos los rangos de sus réplicas, ya que evita la reparación de los mismos datos varias veces. Tenga en cuenta que si usa esta opción para reparar un nodo, también debe usarla para el resto de su grupo.
-par	Ejecutar reparaciones en paralelo. Realiza las reparaciones más rápido, pero restringe significativamente la capacidad del clúster para manejar las solicitudes.
-Hospedadores	Le permite especificar una lista de nodos delimitados por comas para transmitir sus datos desde. Útil si tiene nodos que se sabe que son "buenos". Si bien está documentado como una opción válida para Cassandra 2.1+, también funciona con Cassandra 2.0.

Observaciones

Reparación Casandra Anti-Entropía:

La reparación anti-entropía en Cassandra tiene dos fases distintas. Para ejecutar reparaciones exitosas y exitosas, es importante comprender ambas.

- **Cálculos del árbol Merkle** : esto calcula las diferencias entre los nodos y sus réplicas.

- **Transmisión de datos** : según el resultado de los cálculos del Árbol Merkle, los datos se programan para ser transmitidos de un nodo a otro. Este es un intento de sincronizar los datos entre réplicas.

Deteniendo una reparación :

Puede detener una reparación emitiendo un comando STOP VALIDATION desde nodetool:

```
$ nodetool stop validation
```

¿Cómo sé cuándo se completa la reparación?

Puede verificar la primera fase de reparación (cálculos de Merkle Tree) revisando `nodetool compactionstats` .

Puede verificar las corrientes de reparación utilizando `nodetool netstats` . Los flujos de reparación también serán visibles en sus registros. Puede `grep` para ellos en los registros de su sistema de esta manera:

```
$ grep Entropy system.log

INFO [AntiEntropyStage:1] 2016-07-25 07:32:47,077 RepairSession.java (line 164) [repair
#70c35af0-526e-11e6-8646-8102d8573519] Received merkle tree for test_users from /192.168.14.3
INFO [AntiEntropyStage:1] 2016-07-25 07:32:47,081 RepairSession.java (line 164) [repair
#70c35af0-526e-11e6-8646-8102d8573519] Received merkle tree for test_users from /192.168.16.5
INFO [AntiEntropyStage:1] 2016-07-25 07:32:47,091 RepairSession.java (line 221) [repair
#70c35af0-526e-11e6-8646-8102d8573519] test_users is fully synced
INFO [AntiEntropySessions:4] 2016-07-25 07:32:47,091 RepairSession.java (line 282) [repair
#70c35af0-526e-11e6-8646-8102d8573519] session completed successfully
```

Los flujos de reparación activos también se pueden monitorear con este comando (Bash):

```
$ while true; do date; diff <(nodetool -h 192.168.0.1 netstats) <(sleep 5 && nodetool -h
192.168.0.1 netstats); done
```

ref: ¿cómo sé si la reparación nodetool está terminada

¿Cómo comprobar las corrientes de reparación atascadas o huérfanas?

En cada nodo, puede monitorear esto con `nodetool tpstats` , y verificar si hay algo "bloqueado" en las líneas "AntiEntropy".

```
$ nodetool tpstats
Pool Name                    Active   Pending      Completed   Blocked  All time blocked
...
AntiEntropyStage                  0         0         854866         0                 0
...
AntiEntropySessions               0         0           2576         0                 0
...
```

Examples

Ejemplos para ejecutar la reparación de Nodetool

Uso:

```
$ nodetool repair [-h | -p | -pw | -u] <flags> [ -- keyspace_name [table_name]]
```

Opción de reparación predeterminada

```
$ nodetool repair
```

Este comando reparará el rango de token primario del nodo actual (es decir, el rango que posee) junto con las réplicas de otros rangos de token que tiene en todas las tablas y todos los espacios de claves en el nodo actual:

Por ejemplo, si tiene un factor de replicación de 3, entonces habrá un total de 5 nodos involucrados en la reparación: 2 nodos repararán 1 rango de partición 2 nodos repararán 2 rangos de partición 1 nodo reparará 3 rangos de partición. (El comando se ejecutó en este nodo)

Reparación en paralelo

```
$ nodetool repair -par
```

Este comando se ejecutará, realice la misma tarea que la reparación predeterminada, pero ejecutando la reparación en paralelo en los nodos que contienen réplicas.

Reparar rango de fichas primarias

Este comando repara solo el rango de token primario del nodo en todas las tablas y todos los espacios de teclas en el nodo actual:

```
$ nodetool repair -pr
```

Repare solo el centro de datos local en el que reside el nodo:

```
$ nodetool repair -pr -local
```

Repare solo el rango primario para todas las réplicas en todas las tablas y todos los espacios clave en el nodo actual, solo transmitiendo desde los nodos listados:

```
$ nodetool repair -pr -hosts 192.168.0.2, 192.168.0.3, 192.168.0.4
```

Repare solo el rango primario para todas las réplicas en el espacio de claves de stackoverflow en el nodo actual:

```
$ nodetool repair -pr -- stackoverflow
```

Repare solo el rango primario para todas las réplicas en la tabla test_users del espacio de claves

de stackoverflow en el nodo actual:

```
$ nodetool repair -pr -- stackoverflow test_users
```

Lea Reparaciones en casandra en línea:
https://riptutorial.com/es/cassandra/topic/4873/reparaciones-en-casandra

Capítulo 8: Seguridad

Observaciones

Recursos de seguridad de casandra

- CQL: definición de sintaxis de roles de base de datos
- CQL: Lista de permisos de objetos
- Documentación DataStax: autenticación interna
- Documentación DataStax: Autorización interna

Examples

Configurando autenticacion interna

Cassandra no requerirá que los usuarios inicien sesión usando la configuración predeterminada. En cambio, sin contraseña, se permiten inicios de sesión anónimos para cualquier persona que pueda conectarse a `native_transport_port` . Este comportamiento se puede cambiar editando la configuración de `cassandra.yaml` para usar un autenticador diferente:

```
# Allow anonymous logins without authentication
# authenticator: AllowAllAuthenticator

# Use username/password based logins
authenticator: PasswordAuthenticator
```

Las credenciales de inicio de sesión validadas por `PasswordAuthenticator` se almacenarán en el espacio de claves interno de `system_auth` . De forma predeterminada, el espacio de claves no se replicará en todos los nodos. Tendrá que cambiar la configuración de la replicación para asegurarse de que Cassandra aún pueda leer las credenciales de usuario del almacenamiento local en caso de que no se pueda acceder a otros nodos en el clúster, ¡de lo contrario no podrá iniciar sesión!

Para `SimpleStrategy` (donde `N` es el número de nodos en su clúster):

```
ALTER KEYSPACE system_auth WITH replication = {'class': 'SimpleStrategy',
'replication_factor': N};
```

Para `NetworkTopologyStrategy` (donde `N` es el número de nodos en el centro de datos correspondiente):

```
ALTER KEYSPACE system_auth WITH replication =  { 'class' : 'NetworkTopologyStrategy',
'datacenter1' : N };
```

Reinicie cada nodo después de los cambios descritos anteriormente. Ahora solo podrás iniciar

sesión con el superusuario predeterminado:

`cqlsh -u cassandra -p cassandra`

(Opcional) Reemplazar el superusuario predeterminado con un usuario personalizado

El uso de un superusuario predeterminado con una contraseña estándar no es mucho más seguro que no usar ningún usuario. Debe crear su propio usuario en lugar de usar una contraseña segura y única:

```
CREATE ROLE myadminuser WITH PASSWORD = 'admin123' AND LOGIN = true AND SUPERUSER = true;
```

Inicie sesión con su nuevo usuario: `cqlsh -u myadminuser -p admin123`

Ahora deshabilite el inicio de sesión para el usuario estándar de Cassandra y elimine el estado de superusuario:

```
ALTER ROLE cassandra WITH LOGIN = false AND SUPERUSER = false;
```

Configurando autorizacion interna

Por defecto, cada usuario podrá acceder a todos los datos en Cassandra. Tendrá que configurar un autorizador diferente en su `cassandra.yaml` para otorgar permisos de objetos individuales a sus usuarios.

```
# Grant all permissions to all users
# authorizer: AllowAllAuthorizer

# Use object permissions managed internally by Cassandra
authorizer: CassandraAuthorizer
```

Los permisos para usuarios individuales se almacenarán en el espacio de claves interno de `system_auth` . Debe cambiar la configuración de replicación en caso de que aún no lo haya hecho al habilitar la autenticación basada en contraseña.

Para `SimpleStrategy` (donde `N` es el número de nodos en su clúster):

```
ALTER KEYSPACE system_auth WITH replication = {'class': 'SimpleStrategy',
'replication_factor': N};
```

Para `NetworkTopologyStrategy` (donde `N` es el número de nodos en el centro de datos correspondiente):

```
ALTER KEYSPACE system_auth WITH replication =  { 'class' : 'NetworkTopologyStrategy',
'datacenter1' : N };
```

Reinicie cada nodo después de los cambios descritos anteriormente. Ahora podrá establecer permisos utilizando, por ejemplo, los siguientes comandos.

Otorga todos los permisos para el espacio de claves y rol especificados:

```
GRANT ALL ON KEYSPACE keyspace_name TO role_name;
```

Otorgar permisos de lectura en todos los espacios de claves:

```
GRANT SELECT ON ALL KEYSPACES TO role_name;
```

Permitir la ejecución de las instrucciones INSERT, UPDATE, DELETE y TRUNCATE en un cierto espacio de teclas:

```
GRANT MODIFY ON KEYSPACE keyspace_name TO role_name;
```

Permite cambiar espacios de teclas, tablas e índices para ciertos espacios de teclas:

```
GRANT ALTER ON KEYSPACE keyspace_name TO role_name;
```

Tenga en cuenta que los permisos se almacenarán en caché para `permissions_validity_in_ms` (`cassandra.yaml`) y que los cambios podrían no ser efectivos al instante.

Lea Seguridad en línea: https://riptutorial.com/es/cassandra/topic/5646/seguridad

Creditos

S. No	Capítulos	Contributors
1	Empezando con Cassandra	Community, muru, perennial_noob, phact, Ravinder Matte, sourav, Stephen Leppik
2	Casandra como servicio	Shoban Sundar
3	Cassandra - PHP	Aaron, ethrbunny, Renjith VR
4	Conectando a Cassandra	geeves
5	Correr Reparación en Cassandra	muru, Ravinder Matte
6	Llaves de casandra	Derlin
7	Reparaciones en casandra	Aaron, Akshay
8	Seguridad	Stefan Podkowinski