Manual de usuario de Apache Cassandra

Contenido

[Introducción 3](#_Toc5903698)

[Arquitectura y características de Cassandra 4](#_Toc5903699)

[Componentes de Cassandra 4](#_Toc5903700)

[Seguridad 5](#_Toc5903701)

[Modelo de Datos de Cassandra 5](#_Toc5903702)

[Comparación entre base de datos relacional y Cassandra 9](#_Toc5903703)

[Lenguaje CQL 9](#_Toc5903704)

# Introducción

Cassandra es una base de datos de software libre ya que pertenece a la Apache software foundation. Es una base de datos tipo NoSQL (El concepto de NoSQL se explicará más adelante) distribuida y basada en un modelo de almacenamiento de “Clave-Valor”, está escrita en Java por ello es necesario de su uso para que funcione. Permite manejar con grandes volúmenes de datos de forma distribuida. Por ello su uso en el ámbito de Big Data es muy recomendable ya que Cassandra es capaz de manejar un tamaño de datos máximo de 2 billones de registros y en el caso de tener que manejar con más registros, se tienen que particionar para poder trabajar con ellos.

# Arquitectura y características de Cassandra

Cassandra nos proporciona tolerancia a fallos y disponibilidad, pero a cambio de ser eventualmente consistente, tal y como se define en el teorema CAP (ya que la actividad de inserción de datos no es su fuerte, si no su consulta de información de forma rápida).El nivel de consistencia puede ser configurado, según nos interese incluso a nivel de query, dicha configuración está repartida de la siguiente forma:

* Escala Distribuida: lo que quiere decir que la información está repartida a lo largo de los nodos del cluster. Además ofrece alta disponibilidad, de manera que si alguno de lo nodos se cae el servicio no se degradará.
* Escala linealmente: lo que quiere decir que el rendimiento de forma lineal respecto al número de nodos que añadamos. Por ejemplo, si con 2 nodos soportamos 100.000 operaciones por segundo, con 4 nodos soportaremos 200.000. Esto da mucha predictibilidad a nuestros sistemas.
* Escala de forma horizontal: lo que quiere decir que podemos escalar nuestro sistema añadiendo nuevos nodos basados en hardware commodity de bajo coste.

Implementa una arquitectura Peer-to-Peer, lo que quiere decir es que todos los nodos tienen la misma importancia, por lo cual el patrón de maestro-esclavo es inexistente, así que ningún nodo tendrá alguna información de más con respecto al resto. De esta forma cualquier nodo puede tomar el rol de coordinador de una query. Será el driver el que decida qué nodo quiere que sea el coordenador (esta gestión la realiza internamente cassandra).

# Componentes de Cassandra

Una vez que ya sabemos la arquitectura que tiene Cassandra y como funciona internamente, además de las posibilidades que nos brinda vamos a conocer de forma resumida cómo está compuesta para así ser consciente en todo momento qué componentes tiene esta base de datos.

* Cluster: es un componente que contiene uno o más centro de datos (datacenter).
* Centro de datos (datacenter): se trata de una colección que almacena los nodos relacionados.
* Nodo: es el lugar donde se almacena los datos de Cassandra.
  + Commit log: es un fichero en donde se almacena la información sobre los cambios en los datos. Sirve para recuperar los datos en caso de una fallo en el sistema.
  + MemTable: estructura de almacenamiento en memoria. Contiene los datos que aún no han sido escritos en un SSTable.
  + SSTable: es un fichero que almacenan los datos escritos en disco. Cada fichero SSTable es inmutable (que no cambia) una vez creado.
* About internode communications (gossip): es el protocolo de comunicación peer-to-peer para descubrir y compartir información sobre la localización y estado de los nodos en un cluster de Cassandra.
* Partitioner: este componente determina cómo se distribuyen los datos entre los nodos (las copias entre los nodos).
* Replica placement strategy: define la estrategia a seguir para almacenar las copias de los mismo datos en diferentes nodos, de forma que se aseguren la accesibilidad y la tolerancia a fallos. Se pueden definir diferentes estrategias. Hay que tener en cuenta que no existen copias principales ni secundarias, todas las copias, incluida la primera, con réplicas.
* Snitch: define la topología que utilizan las estrategias de replicación para colocar las réplicas y dirigir las consultas de forma eficiente.

# Seguridad

La seguridad existente en Cassandra se encuentra a nivel de usuarios como logins con password y permisos de gestión administración vía GRANT/REVOKE, como es utilizada en los sistemas de bases de datos relacionales como por ejemplo ORACLE, a continuación aparece los privilegios que podemos gestionar y a los objetos que son aplicables.

|  |  |
| --- | --- |
| Privilegios | Objetos a los que se le aplican |
| All permissions | All functions |
| Alter | ALL FUNCTIONS IN KEYSPACES “nombre keyspace” |
| Authorize | FUNCTION “nombre de función” |
| Créate | ALL KEYSPACE |
| Describe | KEYSPACE “nombre keyspace” |
| Drop | TABLE “nombre tabla” |
| Execute | ALL ROLES |
| Modify | ROLE “nombre rol” |
| select |  |

A continuación, podremos ver un ejemplo de aplicar /revocar un privilegio:

Asignación de privilegio:

GRANT “privilegio a conceder” ON “objeto al que concede” TO “usuario /rol”;

Revocación de privilegios:

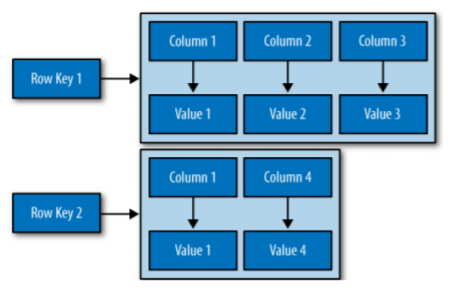
REVOKE "privilegio a revocar" ON "objeto al que revocar" FROM "usuario/rol";

# Modelo de Datos de Cassandra

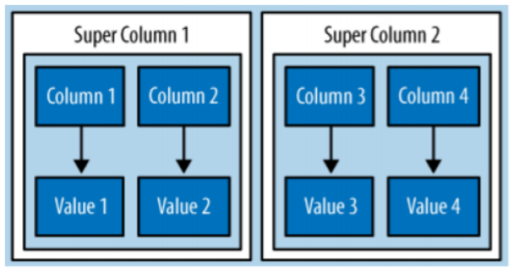
El modelo de datos que se utiliza en Cassandra es algo complejo de entender por cómo se organiza y reparte la información, como ya sabemos, la base de datos de Cassandra se organiza en un sistema de “Clave-Valor” enriquecido, lo que ofrece una visión totalmente diferente a lo que estamos acostumbrado a ver en los sistemas relacionales.

El modelado de datos existente en Cassandra está dividido en diferentes apartados, siendo los siguientes:

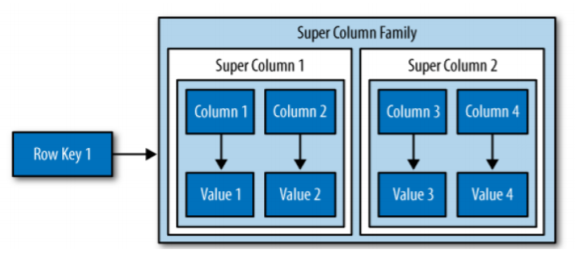
* Columna (Column): es el elemento básico del sistema, es una estructura de tres campos que contienen: el nombre de la columna, el valor de la columna y una marca de tiempo.
  + La clave y el valor se almacenan como datos binarios y el timestamp indica la última vez que se actualizó la columna.
  + Los tres valores son proporcionados por el cliente de alto nivel (por la aplicación), incluido el timestamp. Por ello es necesario tener sincronizado el reloj del sistema con el reloj del clúster.
  + El timestamp se utiliza para la resolución de conflictos y así diferencias cuando se introdujo un dato de otro.
  + El timestamp puede ser de cualquier tipo pero la convención marca que sea el valor en microsegundos.
  + Dicha columna tiene asociada las filas (rows) que es el lugar en donde va a almacenado el valor que hace referencia al nombre de columna.
  + Y la información almacenada que hace referencia a esas columnas se le aplica un hash para generar una key (denominada row-key) que diferencia ese dato con el resto para que en el momento en el que se realice la búsqueda de ese dato, sea capaz de encontrarlo.
  + La row-key es el equivalente a la clave primaria del modelo relacional. y está compuesta por dos partes:
    - Parte del particionado (partition key): el objetivo es identificar en la partición o el nodo en el que se encuentra esa fila/dato
    - Parte del agrupamiento (clustering key): determina el orden físico en el que se almacenan las filas.



* Súper Columna (SuperColumn): es un elemento que se encuentra compuesto por varias columnas (descrito anteriormente). El aspecto negativo que tiene la súper columna es que no aconsejan el uso de ella ya que afecta al rendimiento en la búsqueda de información ya que “desglosar” las columnas por lo cual tarda más en mostrar el resultado, así que por ello no se recomienda su usabilidad.



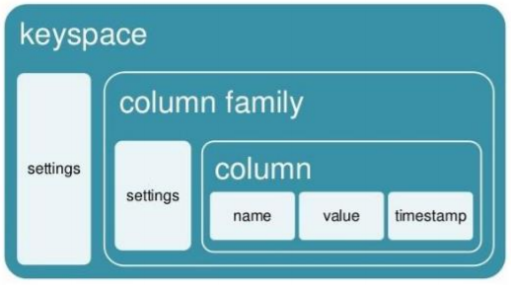
* Familia de Columnas (ColumnFamily): sería un equivalente a una tabla en un esquema de datos relacional. Se trata de un contenedor para una colección ordenada de columnas. Cada entrada (row) se identifica y accede a ella mediante una row-KEY .
  + Cada Familia de Columna es guardada en un fichero separado y es ordenado por la KEY.
  + Los tipos de columnas que almacena la Familia de Columnas es la siguiente:
    - Standard Columns
    - Counter Columns
    - Collection Columns: SET, LIST, MAP
    - User-defined type
    - Tuple-type
    - Timestamp type



* Espacio de clave: es un contenedor para la familia de columnas, más o menos el equivalente a una base de datos (schema) en el modelo relacional. Es una colección ordenada de una familia de columnas o columnas.
  + Los atributos básicos de un Keyspace o Espacio de Clave son:
    - Replication factor: cuánto quieres asignar en rendimiento a favor de consistencia.
    - Replica placement strategy: indica cómo se colocan las réplicas en el anillo:
      * SimpleStrategy (solo un centro de datos).
      * NetworkTopologyStrategy (varios centro de datos).
* Cassandra ofrece soporte para particionado distribuido de datos.
  + RandomPartitioner: ofrece un correcto balanceo de carga.
  + OrderPreservingPartitioner: nos permite ejecutar consultas de rangos, pero exista más trabajo eligiendo node tokens.

○ Además de ello Cassandra tiene consistencia reconfigurable.

* En el momento de escritura, consistency level determina en cuántas réplicas se debe escribir para confirmar a la aplicación cliente.
* Y al leer, también se especifica cuántas réplicas deben responder para retornar los datos a la aplicación cliente.



* Cluster: es un conjunto de máquinas o nodos que ejecutan Cassandra y proporcionan al sistema escalabilidad horizontal, de manera transparente para el cliente.

○ Los datos en Cassandra se guardan en el Clúster (“Ring”) donde se asignan datos a los nodos dentro de un anillo.

○ Un nodo tiene réplicas para diferentes rangos de datos.

○ Si un nodo se cae, su réplica puede responder.

○ El protocolo utilizado en Cassandra es P2P que hace que los datos se repliquen entre nodos acorde a un factor de replicación definido.

# Comparación entre base de datos relacional y Cassandra

A continuación, aparece una pequeña tabla en la cual se muestra las posibles similitudes entre Cassandra y los sistemas de bases de datos relacionales del mercado.

|  |  |
| --- | --- |
| Base de datos relacional | Cassandra |
| Base de datos | Keyspace(Espacio clave) |
| Tabla | Columna Family(Familia de columnas) |
| Clave primaria | Row key |

# Lenguaje CQL

Para poder interactuar con Cassandra mediante CQL tendremos que introducir a través de la sh de la propia base de datos el comando cqlsh, aunque también podemos utilizar herramientas gráficas o a través de los driver soportados por múltiples lenguajes de programación.