**CASSANDRA**

**Introducción**

Cassandra es una base de datos NoSQL de tipo clave-valor. Su peculiaridad principal es que se creará una tabla diferente por cada consulta que se quiera realizar.

Sus principales características son:

* Es un sistema descentralizado: todos los nodos tienen el mismo rol, no existe un nodo maestro.
* Replicación: en el clúster y en múltiples datacenters.
* Escalabilidad lineal: las lecturas y escrituras aumentan a medida que se agregan más nodos al clúster.
* Tolerancia a fallas: la información puede estar en más de un nodo (redundancia) y en múltiples centros de datos.
* Consistencia: posibilidad de obtener el registro con la fecha de grabación de un nodo, del clúster o de todos los datacenter.

**Modelo de datos**

**Keyspace**

Un keyspace puede ser visto como el contenedor más exterior para datos en Cassandra. Todos los datos en Cassandra deberían vivir dentro de un keyspace. Esto puede ser como una base de datos en el Sistema de gestión de bases de datos relacionales, que es una colección de tablas. En el caso de Cassandra, un keyspace es una colección de familias de columnas.

**Familia de Columnas**

Una familia de columnas puede ser vista como una colección de filas, y cada fila es una colección de columnas. Es análoga a una tabla en RDBMS pero tiene algunas diferencias. Las familias de columnas son definidas, pero no es necesario para cada fila tener todas las columnas, y las columnas pueden ser añadidas o eliminadas de una fila cuando se requiera.

**Columna**

La columna es la unidad básica de datos en Cassandra.

Una fila puede ser descrita como:

* “Skinny row”: tamaño casi fijo y un número relativamente pequeño de columnas.

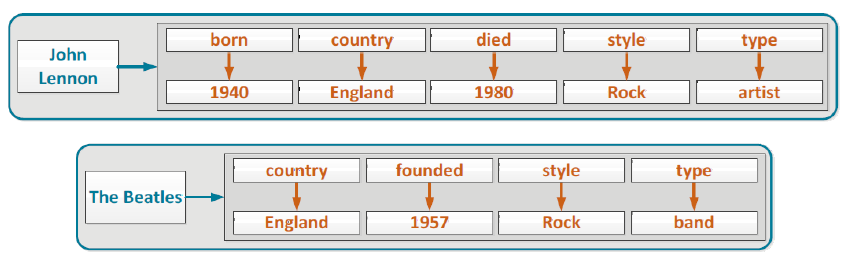


Figura 1. Fila descrita como "Skinny row".

* “Wide row”: tamaño relativamente grande de clave de columnas (100, 1000), este número puede ir aumentando cuando se insertan nuevos datos.

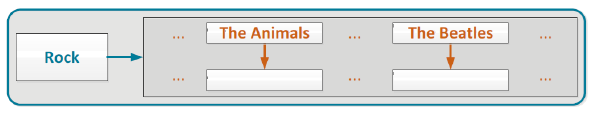


Figura 2. Fila descrita como "Wide row".

La siguiente imagen muestra la estructura de Cassandra.

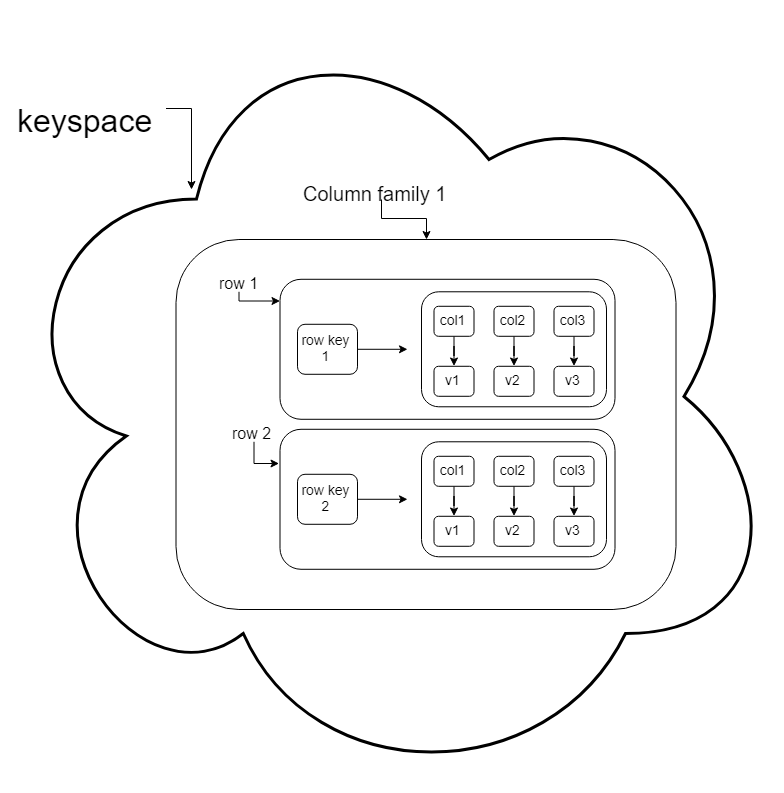


Figura 3. Estructura de Cassandra.

**Clave de partición**

La clave de partición no es otra cosa que la “primera parte” de la clave primaria e identifica de forma unívoca a una partición. Por otro lado, la clave primaria puede ser simple (una única columna) o estar compuesta por más de una columna (clave compuesta). En el caso de una tabla en la que la clave primaria es simple, la clave de partición es la única columna que existe en la clave primaria, por lo tanto, coinciden clave primaria y clave de partición. clave de partición es especialmente importante en Cassandra, ya que es la clave responsable de distribuir los datos a lo largo del total de nodos disponibles.

**Clave compuesta**

Como ya hemos visto, la clave primaria puede tener más de una columna, en estos casos hablamos de clave compuesta. Una característica interesante de la clave compuesta es que únicamente la “primera parte” es considerada como la clave de partición.

**Clave clúster**

Cuando alguna de las consultas que vamos a realizar implica obtener los datos ordenados ascendentemente o descendentemente por alguna de las columnas entra en juego la clave clúster. La clave clúster la conforman el resto de las columnas que forman parte de la clave primaria pero no de la clave de partición, es decir, la “segunda parte” de la clave primaria.

**Ejemplo práctico**

A continuación, se va a proceder a desarrollar un ejemplo práctico para entender mejor los conceptos. Por ejemplo, supongamos un clúster de 10 nodos utilizado para almacenar 1GB de información cada día. Nuestra aplicación es una base de datos de incidencias del departamento de policía de la ciudad de San Francisco. En primer lugar, se crea el keyspace llamado *incidencias:*

*CREATE KEYSPACE personas*

*WITH REPLICATION = {*

*'class' : 'SimpleStrategy',*

*'replication\_factor' : 2*

*} ;*

Para este primer ejemplo, se particionará los datos dependiendo de la categoría de las incidencias.

*CREATE TABLE incidencias.incidenciaporcategoria (*

*pdid text,*

*category text,*

*date date,*

*descript text,*

*PRIMARY KEY (category,pdid)*

*) WITH CLUSTERING ORDER BY (pdid DESC);*

En la tabla anterior la categoría de la incidencia será la clave partición, esto significa que cuando un nodo reciba un registro para escribir aplicará la siguiente función hash(category) = {1, 2, 3, …, 10}, es decir que los posibles valores serán del 1 al 10 (nodos en el clúster). De esta manera todos los registros de una misma categoría estarán en un solo nodo.

La clave de clúster (pdid) es la que se encarga de ordenar los registros en el nodo, en este caso de manera descendiente.

Cada nodo Cassandra almacenará los registros pertenecientes a diferentes claves de partición (en nuestro ejemplo puede haber 10 nodos, pero 100 categorías).

Nuestra tabla anterior quedaría así:

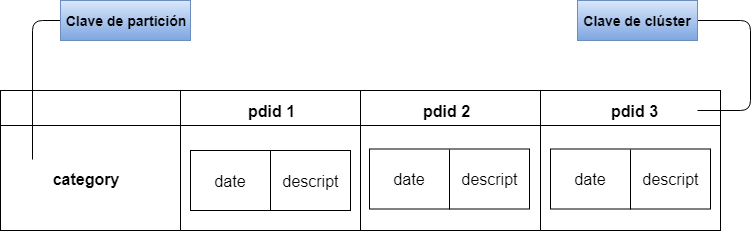


Figura 4. Diseño de la tabla ejemplo ‘incidenciaporcategoria’.

En el ejemplo anterior observamos que para cada clave partición (category) se creará una fila con las incidencias de la misma categoría, ordenados de forma descendiente definidos por la clave clúster (pdid). En este caso los valores de cada celda serán los campos *date*, *descript*.

Cada tabla tendrá entones una fila con N registros para cada categoría. La capacidad máxima de registros por fila teóricamente es de 2 billones.

Para optimizar Cassandra, es necesario agregar más valores a la clave de partición, esto lo podemos hacer separando también por distrito:

*CREATE TABLE incidencias.incidenciaporcategoriadistrito (*

*pdid text,*

*category text*

*pddistrict text,*

*date date,*

*descript text,*

*PRIMARY KEY ((category,pddistrict),pdid)*

*) WITH CLUSTERING ORDER BY (pdid DESC);*

La clave de partición (category,pddistrict) crea diferentes filas de registros por cada categoria/distrito. La tabla ahora será de la siguiente forma:

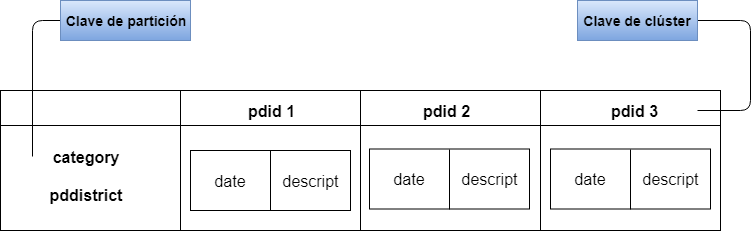


Figura 5. Diseño de la tabla ejemplo ‘incidenciaporcategoriadistrito’.

Para poder consultar la información es obligatorio definir las claves partición. Por ejemplo, para obtener todas las incidencias según la zona y el distrito se haría de la siguiente forma:

*SELECT \* FROM incidencias.incidenciaporcategoriadistrito WHERE category='SUICIDE' AND pddistrict='BAYVIEW';*

**Preparación del fichero de datos**

Para poder utilizar el fichero de datos proporcionado ha sido necesario modificar el formato de los campos “Date” y “Time”. Se ha utilizado el siguiente script escrito en Python:



Figura . Script Python para modificar el formato de los campos 'Date' y 'Time'.

El resultado obtenido es un nuevo csv con las mismas columnas, pero con el formato correcto.

Como se ha comentado anteriormente, para cada consulta es necesario la creación de una tabla con los campos que sean requeridos para dicha consulta. Por este motivo, se ha creado un script Python para facilitar la creación de los distintos csv de donde se importarán los datos para cada tabla.



Figura 7. Script Python para la creación de los distintos csv para importar datos en cada tabla.

Para importar estos csv a una tabla se utiliza el siguiente comando:

*COPY keyspace.name\_table(col1,col2…colN) FROM '../archivo.csv' WITH HEADER = TRUE ;*