Modelado de datos

Apache Cassandra - Bases de datos II

Alberto Díaz Álvarez (<alberto.díaz@upm.es>)
Departamento de Sistemas Informáticos
Escuela Técnica superior de Ingeniería de Sistemas Informáticos

Conceptos básicos

Keyspaces

Las tablas se agrupan en **espacios de claves (*keyspaces*)**

- Entidad lógica que contiene una o más tablas
- La distribución y replicación de los datos es realizada a nivel del espacio de claves
- También define una serie de opciones que se aplican a todas las tablas que estas contienen
    - Por ejemplo, la estrategia de replicación usada por el espacio de claves

En general, se recomienda utilizar un espacio de claves por aplicación

Ejemplo de definición de espacio de claves

```
CREATE KEYSPACE IF NOT EXISTS accounts_vault WITH REPLICATION = {
  'class' : 'SimpleStrategy',
  'datacenter1' : 3,
  'datacenter2' : 2
};

USE accounts_vault;
```

En realidad la sentencia `USE` no es necesaria

- Se puede especificar el espacio de claves como prefijo en las tablas
- Aún así hace más cómodod escribir las consultas

Tablas

Cassandra almacena los datos en **tablas**

- Una tabla es una entidad lógica que organiza el almacenamiento de datos a nivel de nodo y de clúster (de acuerdo a un esquema definido)
- Pueden ser creadas, modificadas y eliminadas sin bloquear operaciones de lectura/escritura

Los datos se organizan en tablas que contienen filas y columnas

- En su creación hay que definir una clave primaria y otras columnas de datos
- Las claves primarias se componen de una o más columnas

Ejemplo de creación de tabla

```
CREATE TABLE IF NOT EXISTS accounts (
  id uuid,
  username text,
  password text,
  service decimal,
  PRIMARY KEY (id)
);
```

Un poco más sobre la clave primaria (I)

Es un subconjunto de las columnas de la tabla

No sólo es obligatoria, sino que además es **no puede cambiar una vez creada**

Tiene dos roles fundamentales:

- Optimizar el rendimiento en las consultas de lectura
  - *Query driven table design*, esto es, las tablas se definen **después** de saber qué consultas queremos hacer (como en casi todos los SGBD NoSQL)
  - La clave primaria se define **según de las consultas que se van a realizar**
- Identificar de forma única cada fila

Un poco más sobre la clave primaria (y II)

La clave primaria se compone de:

- Una *Partition Key*: Obligatoria, y compuesta de una o más columnas
- Una o más *Clustering Keys*: Opcionales

```
...
PRIMARY KEY ((id),  service)
...
```

- En este ejemplo concreto, `id` sería la *Partition Key* y `service` la *Clustering Key*

De acuerdo a esto, se definen dos tipos de tablas:

- Tablas **estáticas**: Aquellas que definen sólo la clave de partición
- Tablas **dinámicas**: Aquellas que definen también la clave de clúster.

Claves de partición (*Partition Keys*)

Cuando se escriben los datos en una tabla, estos se agrupan en **particiones** y se distribuyen en nodos

- Esta operación se basa en la clave de partición (*Partition Key*)
- Al resumen hash de una partición se le denomina *token*
- Cada nodo tiene un rango de tokens, por lo que la clave de partición determina la localidad de los datos (partición) en el clúster

Las particiones son unidades fundamentales en Cassandra

- Cada partición podrá ser encontrada siempre en al menos un nodo
    - Bueno, y también en sus réplicas

Cuando nuestro clúster se compone de cientos de miles de nodos, la clave para mejorar el rendimiento es limitar el número de consultas entre nodos

Claves de clústering (*Clustering Keys*)

Almacena los datos en orden ascendente (defecto) o descendente en la partición

- Ascendente por defecto

```
CREATE TABLE IF NOT EXISTS accounts (
  id uuid,
  username text,
  password text,
  service text,
  PRIMARY KEY ((service),  username)
);
```

En este ejemplo los datos se ordenarán por `service` dentro de cada partición

- Esto optimiza la recuperación de valores cercanos en una misma partición
- Más de una clave de clúster implica órden múltiple

Columnas

Cada tabla está compuesta por filas que contienen **columnas**

Cada columna tiene un **nombre** y un **valor**. Cada columna tiene un **tipo de dato** asociado. Cada columna tiene un **timestamp** asociado.

Modelado de datos

Reglas esenciales para el modelado de datos

Se refieren a la construcción de la clave primaria, que debe optimizar el rendimiento de las consultas:

1. Elegir una clave de partición que responda a la consulta, pero que además distribuya los datos uniformemente por el clúster
    - Por ejemplo, usar `servicio` como *Partition Key* puede ser buena idea si hay muchos servicios y su número de usuarios es similar entre servicios
2. Elegir una clave primaria que minimice el número de particiones a leer
    - Leer muchas particiones, implica (potencialmente) acceder a muchos nodos para obtener los datos requeridos, aunque sean pocos

Y si nos ponemos, ordenar las *Clustering Keys* de acuerdo a las consultas que vamos a realizar

Tipos de Columnas

Espacio de Claves (*keyspaces*)

Tablas

Índices

Triggers

Vistas materializadas

Gracias