CQL (*Cassandra Query Language*)

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

Cassandra Query Language (CQL)

Es el lenguaje principal de comunicación con clústers de Cassandra

- Su sintaxis es muy intuitiva, similar a SQL
- Pero no tiene gramática para caracteristicas relacionales, como los `join`

```
CREATE KEYSPACE nombre_del_keyspace ...
DROP KEYSPACE nombre_del_keyspace;
CREATE TABLE nombre_de_la_tabla ...
TRUNCATE TABLE nombre_de_la_tabla;
DROP TABLE nombre_de_la_tabla;
INSERT INTO nombre_de_la_tabla ...
UPDATE nombre_de_la_tabla SET ...
DELETE FROM nombre_de_la_tabla WHERE ...
// Y más ...
```

Características de CQL

Es case-insensitive; por ejemplo, `SELECT` es lo mismo que `select`

- Lo mismo con los indicadores
- De hecho da igual cómo los creemos, CQL los almacena en minúsculas
- **Excepto** en los identificadores que se enmarcan entre dobles comillas
  - `tabla` es igual que `TABLA`, pero `"tabla"` es diferente de `"TABLA"`

Los comentarios se preceden de `//`

¿Cómo ejecutamos queries?

Podemos ejecutarlas programáticamente desde un *driver* cliente

- Por ejemplo, desde Python o Java (Datastax Java Driver, el de por defecto)

También desde su *shell*, denominada cqlsh (*Cassandra Query Language Shell*)

- Viene de serie con cualquier paquete que instalemos de Cassandra
- Está desarrollado en Python
- Se conecta a un nodo del clúster y ejecuta las queries que le pasemos
  - Si no se especifica, se conecta al nodo local (el de por defecto)

Usar editores propietarios o de terceros

- Por ejemplo, DataStax Studio

`cqlsh`

```
$ cqlsh [options] [host [port]]
```

- `options` son opciones que incluyen, entre otras:
  - `--help`: La ayuda, que incluye todas las opciones disponibles
  - `--version`: La versión de `cqlsh`
  - `-u` y `-p`: El usuario y la contraseña
  - `-k`: El `keyspace` a usar
  - `-f`: El fichero de queries a ejecutar
  - `--request-timeout`: El timeout de las queries

Comandos especiales

- `CAPTURE`: Captura la salida de la query en un fichero (no sobrescribe, añade)
- `CONSISTENCY`: Muestra el nivel de consistencia y permite cambiarlo
- `COPY`: Importación y exportación de datos
- `DESCRIBE`: Muestra información sobre el clúster, `keyspace`, tablas, etc.
- `EXIT`: Sale de `cqlsh`
- `PAGING`: Activa o desactiva la paginación en los resultados de las consultas
- `TRACING`: Activa o desactiva las trazas de las consultas

Consistencia "ajustable"

`CONSISTENCY` permite "ajustar" el nivel de consistencia por operación

- **¿Cuántas réplicas** deben responder para aceptar una operación?
  - `ONE`, `TWO` y `THREE`: Al menos una, dos y tres réplicas, respectivamente
  - `QUORUM`: La mayoría de las réplicas del clúster
  - `ALL`: Todas las réplicas
  - `LOCAL_*`: Limita la respuesta a las réplicas del datacenter local

¿Y si existen inconsistencias entre réplicas? Se resuelven durante las lecturas

- **CUIDADO**: Una mayor consistencia afectará significativamente al rendimiento

Un proceso similar se realiza durante las **operaciones de escritura**

- **CUIDADO**: Si no se puede escribir en todas las réplicas, la operación fallará

Comando `COPY`

Permite importar datos desde ficheros CSV

```
COPY tabla (columna1, columna2, ...) FROM 'fichero.csv' WITH DELIMITER = ',' AND HEADER = TRUE;
```

También permite la exportación de datos a ficheros CSV

```
COPY tabla (columna1, columna2, ...) TO 'fichero.csv' WITH DELIMITER = ',' AND HEADER = TRUE;
```

CUIDADO: No es lo más recomendable para la carga o volcado de datos masivos

Identificadores y palabras clave

Comandos shell

Manipulación de datos CQL

Tipos de datos CQL

Funciones

Funciones definidas por el usuario

Colecciones

Tipos de datos definidos por el usuario

Seguridad y roles

Gracias