Taller Cassandra

Instalación

1. Pre-requisitos.

- Oracle JDK 8 u OpenJDK 8. Descargue una versión anterior al JDK 8_152. Por ejemplo: JDK 8 121
- Python 2.7

2. Instale Java

- Abra una terminal
- Ingrese a la carpeta donde descargó el archivo

```
$ cd /home/usuario/Descargas
$ tar -xvzf jdk-8u121-linux-x64.tar.gz
```

- Regrese al home del usuario

```
$ cd /home/usuario
```

- En esta misma terminal, vamos a editar un archivo que le indica al sistema donde está Java. Primero abra el archivo `.bashrc` con `nano`

```
$ nano .bashrc
```

Luego añada las siguientes lineas al final del archivo

```
export JAVA_HOME=/home/usuario/Descargas/jdk1.8.0_121
export PATH=$PATH:$JAVA_HOME/bin
```

Guarde el archivo con Ctrl+O y salga con Ctrl+X

- Desde la misma terminal ejecute la siguiente linea con el fin de que la edición que acabamos de hacer se pueda usar:

```
source .bashrc
```

- Valide la instalación. Desde la misma terminal ejecute

```
$ java -version
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)
```

3. Descargue Cassandra.
`http://www.apache.org/dyn/closer.lua/cassandra/3.11.4/apache-cassandra-3.11.4-bin.tar.gz`

4. Descomprima y cambie al directorio creado.

```
$ tar -xvzf apache-cassandra-3.11.4-bin.tar.gz
$ cd apache-cassandra-3.11.4
```

5. Ejecute el servidor de Cassandra (Utilice Ctrl-C para detenerlo).

```
$ ./bin/cassandra -f
```

6. Ejecute el interprete de Comandos de Cassandra.

```
$ ./bin/cqlsh
```

Consultas con CQL

1. Listar keyspaces existentes:

```
> DESC KEYSPACES;
```

2. Crear nuevo keyspace example:

```
> create KEYSPACE example WITH REPLICATION = { 'class' : 'SimpleStrategy',
'replication_factor' : 1};
```

3. Seleccionar keyspace:

```
> USE example;
```

4. Crear una nueva Columnn Family users:

```
> CREATE TABLE users (user_name varchar, password varchar, gender varchar,
last_name
varchar, PRIMARY KEY (user_name));
```

5. Insertar Fila (inserte mas filas por su cuenta):

```
> INSERT INTO users (user_name, password, gender, last_name) VALUES
('pperezc',
'mypass','male', 'perez');
```

6. Insertar más usuarios.
7. Buscar Fila

```
> SELECT * FROM users WHERE last_name = 'perez';
```

El filtrado a través de columnas diferentes a la clave primaria puede arrojar una advertencia de rendimiento. Esto se debe a que Cassandra usa la clave primaria como clave de partición para almacenar los registros en los diferentes nodos. Un filtrado a través de un campo diferente a la clave primaria, obliga a Cassandra a recolectar la información presente en todos los nodos para encontrar aquellos que coinciden con la búsqueda. [1] [2]

```
    > SELECT * FROM users WHERE last_name = 'perez' ALLOW FILTERING;
```

8. Crear un índice:

```
    > CREATE INDEX ON users(last_name);
```

9. Buscar Fila

```
    > SELECT * FROM users WHERE last_name = 'perez';
```

10. Alterar la tabla:

```
    > ALTER TABLE users ADD name varchar;
```

11. Actualizar un valor:

```
    > UPDATE users SET name = 'pedro' WHERE user_name = 'pperezc';
```

12. Seleccionar:

```
    > SELECT * FROM users;
    > SELECT * FROM users WHERE last_name = 'perez';
    > SELECT * FROM users LIMIT 1;
```

13. Colección SET:

```
    > ALTER TABLE users ADD emails set<text>;
    > INSERT INTO users(user_name, password, gender, last_name, emails) VALUES
('Laura',
    '5m8svvd', 'female', 'Paez', {'lau@uno.com'});
    > SELECT * FROM users;
    > UPDATE users SET emails = emails+{'lau1234@uno.com'} WHERE user_name =
'Laura';
    > SELECT * FROM users;
```

14. Colección LIST:

```
    > ALTER TABLE users ADD top_places list<text>;
    > UPDATE users SET top_places = ['rivendell', 'rohan'] WHERE user_name =
'Laura';
    > SELECT * FROM users;
    > UPDATE users SET top_places = top_places+[ 'the shire' ] WHERE user_name =
'Laura';
    > SELECT * FROM users;
    > UPDATE users SET top_places[2] = 'riddermark' WHERE user_name ='Laura';
    > SELECT * FROM users;
    > DELETE top_places[1] FROM users WHERE user_name = 'Laura';
    > SELECT * FROM users;
```

15. Colección MAP:

```
    > ALTER TABLE users ADD telefonos map<text, text>;
    > UPDATE users SET telefonos = {'casa':'123456', 'oficina':'1236'} WHERE
user_name =
    'Laura';
    > SELECT * FROM users;
    > UPDATE users SET telefonos ['celular'] = '3012356' WHERE user_name =
'Laura';
    > SELECT * FROM users;
```

Otros recursos

- Guía de referencia de CQL. CQL
- Juego de parámetros de Cassandra: Cassandra calculator

Taller Entregable

Descripción

La red social Twitter puede ser entendida como una comunidad de usuarios que siguen y son seguidos por otros usuarios. La relación de seguimiento no es simétrica, es decir, si un usuario `A` sigue a un usuario `B`, no implica que el usuario `B` siga al usuario `A`. Como parte de su sistema, Twitter Inc. permite al usuario seguir o dejar de seguir usuarios, consultar la lista de usuarios que sigue, consultar la lista de usuarios que lo siguen.

Últimamente Twitter está contratando personal experto para mejorar su sistema de base de datos y como parte del proceso de entrevista te ha encargado la tarea de diseñar un modelo de datos en Cassandra que permita de manera eficiente realizar los siguientes patrones de acceso:

Lecturas:

- Listar usuarios que sigue el usuario `X`.
- Determinar si un usuario `X` sigue al usuario `Y`.
- Listar los usuarios que siguen al usuario `X`.

Escrituras:

- Usuario `X` sigue un nuevo usuario `Y`.
- Usuario `X` deja de seguir a un usuario `Y`.

Entregables

- Diagrama del modelo de datos
- Script de creación de datos
- Script con las operaciones de lectura y escritura.

Comprima los archivos en un zip y use el siguiente link para cargar el desarrollo del taller:
https://www.dropbox.com/request/ch4hTKzwkDwyeFMmgI4Q
Deadline: martes 9 de julio se 2019.