

BASES DE DATOS NO CONVENCIONALES

MEMORIA PRÁCTICA: Bases de Datos NoSQL – Cassandra

Autores: Verónica Gómez Gómez, Pablo Olmos Martínez,
Carlos Grande Núñez

Table of Contents

Introduccion.....	2
Visualizacion y planificacion de las consultas.....	3
Por cada producto presentar sus características y formatos en los que se comercializa.....	3
De acuerdo a un producto y formato seleccionar las referencias de las que se dispone.....	3
De acuerdo al nick name del usuario (usuario registrado) y una determinada fecha seleccionar la lista de la compra y el precio parcial por cada linea de pedido.....	4
Modificaciones de SQL a Cassandra.....	5
Ejercicios.....	6
I. Formatos en los que se comercializa el producto 'Como y tronar'.....	6
II. Cambio de nombre de producto 'Son antes' por 'Antes'.....	6
III. El producto 'Duende y luna' empieza a comercializarse en el formato 'roasted bean'.....	8
IV. Referencias asociadas al producto 'Milagros' con format='capsules'.....	9
V. Ultimo pedido realizado por el usuario con nickname='cisni'.....	9
VI. Cantidad total gastada por el usuario cuyo nombre es 'naki'.....	10
VII. Insercion de una nueva referencia (barcode=QIO99947O911189 para el producto 'Como y tronar' con formato capsules.....	10
Conclusiones y observaciones.....	10

Introducción

Para esta práctica se han seguido los pasos indicados en la memoria así como diversos consejos y mejores usos en Cassandra.

Esta práctica consta de varias partes:

1. Una visualización previa y planificación de cuales serían las consultas mas habituales en Cassandra para nuestra base de datos.
2. Conocimiento básico de la base de datos SQL contenida en la práctica.
3. Aprendizaje y recuerdo sobre bases de datos SQL y el entorno ofrecido en MyApps.
4. Población de las base de datos conforme a los scripts incluidos.
5. Creación de las vistas SQL en referencia a las necesidades de Cassandra.
6. Exportación de los scripts SQL.
7. Modificaciones básicas sobre dichos scripts para utilizar en Cassandra.
8. Realización de los ejercicios de la práctica sobre Cassandra
9. Conclusiones y resultados

Visualización y planificación de las consultas

En una primera parte y previo a crear tablas en Cassandra o incluso trabajar con SQL teníamos que entender correctamente cuales serian las consultas mas habituales que se harían en Cassandra.

La misma practica nos da estas pistas, por lo que procedimos a crear tres tipos de consulta.

Por cada producto presentar sus características y formatos en los que se comercializa

Para esta primera tabla decidimos utilizar una primary key compuesta por el nombre y el formato, ya que un tipo de café se puede comercializar en varios formatos. Si bien es cierto que este tipo de tabla nos genera una entrada por cada formato para un mismo café.

```
CREATE TABLE practica.PRODUCTS_CHAR_FORMAT
(NAME text,
SPECIES text,
VARIETAL text,
PROVENANCE text,
TOAST text,
PROCESS text,
FORMAT text,
PRIMARY KEY (NAME,FORMAT));
```

La vista asociada en SQL a esta tabla es la siguiente:

```
SELECT a.name, a.species, a.varietal, a.provenance, a.toast, a.process,
b.formatt
FROM PRODUCT a, FORMATS b WHERE a.name = b.product
```

De acuerdo a un producto y formato seleccionar las referencias de las que se dispone

Para esta tabla ya utilizamos una primary key compuesta y su clustering key. Hay que tener en cuenta que barcode es un código único.

```
CREATE TABLE practica.PRODUCTS_FORMAT_REFS  
(PRODUCT text,  
FORMAT text,  
BARCODE text,  
PACK text,  
PRICE text,  
STOCK text,  
MINSTOCK text,  
MAXSTOCK text,  
PRIMARY KEY ((PRODUCT, FORMAT), BARCODE));
```

La vista asociada a SQL es la siguiente:

```
SELECT product, formatt, barCode, pack, price, stock, min_stock, max_stock  
FROM REFS
```

No hay problema por usar nombres como format en la tabla que se usara en Cassandra y los nombres de las variables obtenidos por las vistas asociadas SQL. Ya que antes de usar dichos scripts para Cassandra los datos se modificaran para que funcionen en Cassandra. Mas adelante contare brevemente las modificaciones realizadas y los problemas encontrados.

De acuerdo al nick name del usuario (usuario registrado) y una determinada fecha seleccionar la lista de la compra y el precio parcial por cada linea de pedido

Esta es la tabla/consulta que mas problemas ha dado. Si bien no por la tabla en Cassandra en si, ya que es sencilla si no por como se exporta en SQL.

Los principales problemas han estado relacionados con el formato de la fecha y el trato a los floats entre SQL y Cassandra.

En el momento de la realización de esta memoria aun se estaba trabajando para adaptar ambos valores al formato de Cassandra utilizando un script de Python. En el caso de SQL los floats vienen definidos con '10,10' mientras que en Cassandra se deben declarar como 10.10.

Similar sucede con las fechas, para SQL tenemos un formato tipo dd/mm/aa mientras que Cassandra utiliza el siguiente formato aaaa-mm-dd.

```
CREATE TABLE practica.USERS_PURCHASES
(USR text,
ORDERDATE date,
BARCODE text,
PRICE float,
AMOUNT text,
PRIMARY KEY ((USR, ORDERDATE), BARCODE));
```

De momento hay una tabla preliminar sobre la que se ha trabajado por simplificar y poder obtener los resultados de las queries necesarias. Esta tabla temporal hasta que se consiga arreglar los problemas con dichos valores es la siguiente:

```
CREATE TABLE practica.USERS_PURCHASES
(USR text,
ORDERDATE text,
BARCODE text,
PRICE text,
AMOUNT text,
PRIMARY KEY ((USR, ORDERDATE), BARCODE));
```

Su vista asociada para SQL es la siguiente:

```
SELECT b.usr, b.orderDate, b.payment, b.barCode, c.price, b.amount
FROM LINES_USR b, REFS c WHERE b.barCode = c.barCode
```

Modificaciones de SQL a Cassandra

Lo interesante son las pocas modificaciones que han sido necesarias, con un replace all y eliminando las dos primeras lineas del *.sql ha sido suficiente para adaptar. Lo mas complicado esta resultando ser la conversión de floats y el formato de la fecha.

Ejercicios

Mostrare brevemente las consultas y las salidas obtenidas por cada ejercicio realizado. Se parte de una base en la que ya se han poblado las tablas de Cassandra. Las instrucciones sobre como se realizaron los pasos se encuentran en MDS_CQL_OlmosGrandeGomez.zip en un README.md con las instrucciones.

I. Formatos en los que se comercializa el producto 'Como y tronar'.

CONSULTA REALIZADA:

```
SELECT format FROM practica.PRODUCTS_CHAR_FORMAT WHERE NAME='Como y tronar';
```

SALIDA OBTENIDA:

format

capsules

prepared

raw bean

roasted bean

II. Cambio de nombre de producto 'Son antes' por 'Antes'

Para este ejercicio al tratarse 'Son antes' parte de una primary key o una primary key directamente no hemos encontrado una forma efectiva de modificarla. Así que el procedimiento que hemos seguido no es el mas elegante, pero nos ha permitido actualizar el nombre.

Primero se ha realizado una consulta a las dos tablas para ver cuantas entradas existen.

product	format	barcode	maxstock	minstock	pack	price	stock
-----+-----+-----+-----+-----+-----+-----+-----							
Son antes	ground	IOI36746I421868	4340	270	pkg 200 g.	24,15	2533

Son antes	ground	QQI47782I728991	1500	30	pkg 1 Kg.	96,18	300
Son antes	capsules	OQI73010I681542	4210	80	pkg 25 un.	2,9	2387
Son antes	capsules	QII79718I227817	4080	220	pkg 10 un.	1,86	2393
Son antes	roasted bean	OQI67128I901745	2420	410	pkg 200 g.	9,5	900
Son antes	roasted bean	OQQ50976Q123543	4050	320	pkg 1 Kg.	36,03	2124
Son antes	prepared	OQO17188Q460551	4630	260	can 330 ml.	3,99	4088
Son antes	prepared	QIO44629Q878316	4300	210	cup 200 ml.	2,7	2701

(8 rows)

```
cqlsh> SELECT * FROM practica.PRODUCTS_CHAR_FORMAT WHERE NAME='Son antes';
```

name	format	process	provenance	species	toast	varietal
Son antes	capsules	normal	Brasil	Arabica	natural	Maragogype
Son antes	ground	normal	Brasil	Arabica	natural	Maragogype
Son antes	prepared	normal	Brasil	Arabica	natural	Maragogype
Son antes	roasted bean	normal	Brasil	Arabica	natural	Maragogype

A continuación se procedió a insertar una replica de las entradas obtenidas utilizando eso si, el nuevo nombre:

```
INSERT INTO practica.PRODUCTS_CHAR_FORMAT (name, format, process,
provenance, species, toast, varietal) VALUES ('Antes', 'capsules', 'normal',
'Brasil', 'Arabica', 'natural', 'Maragogype');
```

```
INSERT INTO practica.PRODUCTS_CHAR_FORMAT (name, format, process,
provenance, species, toast, varietal) VALUES ('Antes', 'ground', 'normal',
'Brasil', 'Arabica', 'natural', 'Maragogype');
```

El resto de los insert se encuentran en el .zip adjunto con todo el código utilizado. A continuación tenemos una “duplicidad” con el nombre antiguo y el nuevo así que el ultimo paso es eliminar aquellas entradas que cuentan con la primary key obsoleta.

```
DELETE FROM practica.PRODUCTS_CHAR_FORMAT WHERE NAME='Son antes';
```

```
DELETE FROM practica.PRODUCTS_FORMAT_REFS WHERE PRODUCT='Son
antes' and FORMAT='ground';
```

```
DELETE FROM practica.PRODUCTS_FORMAT_REFS WHERE PRODUCT='Son
antes' and FORMAT='capsules';
```

Como ultimo paso comprobamos que se hayan eliminado las entradas

```
cqlsh> SELECT * FROM practica.PRODUCTS_CHAR_FORMAT WHERE NAME='Son
antes';
```

name	format	process	provenance	species	toast	varietal
-----+	-----+	-----+	-----+	-----+	-----+	-----

(0 rows)

III. El producto 'Duende y luna' empieza a comercializarse en el formato 'roasted bean'

QUERY

```
INSERT INTO practica.PRODUCTS_CHAR_FORMAT (name, format, process, provenance, species,
toast, varietal) VALUES ('Duende y luna', 'roasted bean', 'decaf', 'Camerun', 'Arabica', 'natural',
'Charrieriana');
```

RESULTADO

```
cqlsh> SELECT * FROM practica.PRODUCTS_CHAR_FORMAT WHERE NAME='Duende y luna' ;
```

name	format	process	provenance	species	toast	varietal
-----+	-----+	-----+	-----+	-----+	-----+	-----
Duende y luna	capsules	decaf	Camerun	Arabica	natural	Charrieriana
Duende y luna	roasted bean	decaf	Camerun	Arabica	natural	Charrieriana

En este caso al tratarse de las características de un café que se comercializa con un nuevo formato hemos considerado utilizar las mismas características de Duende y luna en versión capsules.

IV. Referencias asociadas al producto 'Milagros' con format='capsules'

QUERY

```
cqlsh> SELECT * FROM practica.PRODUCTS_FORMAT_REFS WHERE PRODUCT='Milagros' AND  
FORMAT='capsules' ;
```

RESULTADO

product	format	barcode	maxstock	minstock	pack	price	stock
Milagros	capsules	OQI11851I601451	2210	130	pkg 25 un.	12,15	2046
Milagros	capsules	QOO07711I980323	4530	110	pkg 10 un.	6,1	2709

V. Ultimo pedido realizado por el usuario con nickname='cisni'

Estamos trabajando de momento en el formateo adecuado de la variable fecha. Hay una primera aproximación a como seria la query, si bien el resultado obtenido no es real debido a que la fecha esta mal formateada.

El formato usado en la población de la tabla es de: dd-mm-yy cuando para Cassandra es yyyy-mm-dd.

Aunque este mal, de momento utilizamos la query simplemente por saber que tipo de búsqueda habría que realizar. Si da tiempo se mejorara el SQL obtenido para que la fecha sea la real.

QUERY

```
cqlsh> SELECT max(orderdate) FROM practica.USERS_PURCHASES WHERE USR='cisni' allow  
filtering;
```

RESULTADO

```
system.max(orderdate)
```

```
-----  
0029-12-07
```

(1 rows)

VI. Cantidad total gastada por el usuario cuyo nombre es 'naki'

Este ejercicio se encuentra todavía en proceso. Entendemos que la consulta a realizar una vez consigamos transformar correctamente de text a float seria el siguiente:

```
SELECT sum(price) FROM practica.USERS_PURCHASES WHERE USR='naki' allow filtering;
```

VII. Inserción de una nueva referencia (barcode=QIO999470911189 para el producto 'Como y tronar' con formato capsules

QUERY

```
INSERT INTO practica.PRODUCTS_FORMAT_REFS (product, format, barcode) VALUES ('Como y tronar', 'capsules', 'QIO999470911189');
```

RESULTADOS

```
cqlsh> SELECT * FROM practica.PRODUCTS_FORMAT_REFS WHERE PRODUCT='Como y tronar' AND FORMAT='capsules' ;
```

product	format	barcode	maxstock	minstock	pack	price	stock
Como y tronar	capsules	III50039Q920868	2920	50	pkg 10 un.	4,1	691
Como y tronar	capsules	OIQ87311Q947313	3910	70	pkg 500 g.	8,65	1263
Como y tronar	capsules	QIO999470911189	null	null	null	null	null

Conclusiones y observaciones

Ha sido una practica bastante interesante que nos ha permitido entender mejor la lógica que hay que aplicar para crear tablas en Cassandra. Es cierto que es un cambio en la forma de crear tablas a la que estamos acostumbrados, como puede ser un SQL, MYSQL y similares.

Vemos que se debe hacer un trabajo previo bastante importante que consiste en saber cuales serán las consultas mas frecuentes y actuar en conforme a ellas.

Los mayores problemas que nos hemos encontrado esta relacionado con las conversiones de valores y formatos que hay que hacer para una correcta realización de varios de los ejercicios. Es algo sobre lo que nos gustaría indagar mas, el potencial que hay detrás de Cassandra para dichas conversiones o si por el contrario se debe apoyar en otras herramientas como Python para realizar un ajuste de valores y formatos.