



**Práctica Base de Datos no Convencionales
(Máster Data Science)**

Beatriz Visitación, Natalia Alonso y Susana Albarrán Félix

Contenidos

| | | |
|-----|--|----|
| 1 | Introducción | 3 |
| 2 | Captura y procesamiento de los datos..... | 4 |
| 3 | Almacenamiento de los datos..... | 5 |
| 4 | Análisis de los datos. | 6 |
| 4.1 | Formatos en los que se comercializa el producto 'Como y tronar' | 6 |
| 4.2 | Cambio de nombre del producto 'Son antes' por 'Antes'. | 6 |
| 4.3 | El producto 'Duende y luna' empieza a comercializarse en el formato 'roasted bean' | 8 |
| 4.4 | Referencias asociadas al producto 'Milagros' con formato capsulas (formatt = 'capsules'). ... | 8 |
| 4.5 | Último pedido realizado por el usuario con nickname='cisni'. | 9 |
| 4.6 | Cantidad total gastada por el usuario cuyo nickname es 'naki'. | 9 |
| 4.7 | Inserción de una nueva referencia (barcode= QIO99947091) para el producto 'Como y tronar' con formato cápsulas. | 10 |

1 Introducción

| GESTIÓN DEL DOCUMENTO | | | |
|--|--|----------------|------------|
| Autor | Beatriz Visitación, Natalia Alonso y Susana Albarrán | Fecha | 07/05/2020 |
| Nombre fichero | MDS_Memoria_Visitacion_Alonso_Y_Albarraan.pdf | Versión | 1.0 |
| Descripción: En esta memoria, se incluyen todos los pasos que se han seguido para realizar la práctica de Bases de Datos no Convencionales con Cassandra. Para la realización de la práctica se ha usado Lenguaje de programación Python, SQL y Cassandra (CQL). Las consultas se han realizado en cqlsh. | | | |

En esta práctica se partirá de unas fuentes de datos proporcionadas en sql, con distintas tablas usadas para la gestión de una compañía de comercio justo. A través de estas tablas, se realizan unas vistas y se exportan en csv para su posterior procesamiento. Este procesamiento preparará las tablas para cargarlas en Cassandra y realizar las consultas solicitadas.

La realización de la práctica consta de tres partes fundamentales: captura y procesamiento de datos en sql, almacenamiento y procesamiento del csv, carga y análisis de los datos en Cassandra.

El objetivo de este trabajo es utilizar esta información para realizar consultas en Cassandra y a través de las tablas proporcionadas.

2 Captura y procesamiento de los datos.

Los datos han sido generados mediante SQL-Developer, y se han obtenido las vistas necesarias para cubrir las necesidades de la empresa en relación a las consultas más frecuentes. En este caso se ha considerado que las tablas necesarias para cubrir estas consultas son **ORDERS** y **PRODUCTS**. Adicionalmente, se ha hecho un filtrado de algunas columnas porque no las hemos considerado necesarias.

Para el caso de la tabla **ORDERS** se han seleccionado las columnas **ORDERDATE**, **RETAILPRICE** y **USERR**; y para la tabla **PRODUCTS** se han mantenido todas las columnas, ya que se han considerado todas ellas como “características” asociadas a cada producto. Para obtener las vistas, se han usado las siguientes instrucciones:

- `CREATE VIEW view_products
AS (SELECT * FROM PRODUCTS)`
- `CREATE VIEW view_orders
AS (SELECT ORDERDATE, RETAILPRICE, USERR FROM ORDERS)`

A continuación, las vistas correspondientes se han exportado en formato csv. Una vez creados los csv se ha decidido hacer un pequeño procesamiento para poder obtener los resultados correctos en las consultas.

En primer lugar, se ha añadido un identificador a cada elemento (fila), bien sea producto (**product**) o pedido (**order**), ya que la clave primaria que es necesaria en Cassandra debe ser única (por fila) y con este identificador se aseguraba este requisito. También se ha modificado el campo **RETAILPRICE**, que tenía la forma “15.14 **c**”, y se ha eliminado el carácter **c** para poder realizar operaciones con este campo **numérico**.

El código fuente donde se realiza este procesamiento corresponde al fichero que se denomina ***process_key.py***.

3 Almacenamiento de los datos.

A continuación, se ha realizado la creación de una nueva base de datos en Cassandra que se ha denominado **practica_cassandra**, para ello se han realizado los siguientes pasos:

- `CREATE KEYSPACE practica_cassandra WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };`
- `USE practica_cassandra;`

El siguiente paso es la creación del esquema que tendrán las tablas:

- `CREATE TABLE practica_cassandra.products (
 PRODUCT text,
 BARCODE text,
 FORMATT text,
 MACHINEMAKE text,
 MACHINEMODEL text,
 (...)
 ZIP text,
 TOWN text,
 COUNTRY text,
 id_product text,
 PRIMARY KEY (id_product)
);`
- `CREATE TABLE practica_cassandra.orders (
 ORDERDATE date,
 RETAILPRICE double,
 USERR text,
 id_order text,
 PRIMARY KEY (id_order)
);`

A continuación, se realiza la carga de las tablas, usando los ficheros csv generados y procesados:

- `COPY practica_cassandra.products (PRODUCT, BARCODE, FORMATT, MACHINEMAKE, MACHINEMODEL, CAFFEA, VARIETAL, ORIGIN, ROASTING, PROCESS, PACKAGING, RETAILPRICE, STOCK, MIN_STOCK, MAX_STOCK, COSTPRICE, PROVIDER, TAXID, SALESPERSON, EMAIL, PHONENUM, BANKACCOUNT, THOROUGHFARE, WAYNAME, GATENUMBER, BLOCKK, STAIRS, FLOORR, DOOR, ZIP, TOWN, COUNTRY, id_product) FROM 'products_process.csv' WITH DELIMITER=',' AND HEADER=TRUE;`
- `COPY practica_cassandra.orders (ORDERDATE, RETAILPRICE, USERR, id_order) FROM 'orders_simplif_process.csv' WITH DELIMITER=',' AND HEADER=TRUE;`

4 Análisis de los datos.

En esta etapa se realizan consultas sobre la base de datos almacenada en Cassandra:

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

```
SELECT formatt FROM practica_cassandra.products WHERE product='Como y tronar' ALLOW FILTERING;
```

Para esta primera consulta se accede a la tabla de los productos, se filtra por el nombre del producto 'Como y tronar' y se muestran los formatos. Nótese que para poder filtrar por un campo distinto de la **primary key** es necesario añadir la cláusula ALLOW FILTERING. A continuación, se muestra el resultado:

```
mysqlsh:practica_cassandra> SELECT formatt FROM practica_cassandra.products WHERE product='Como y tronar' ALLOW FILTERING;
formatt
-----
prepared
capsules
capsules
roasted bean
raw bean
raw bean
roasted bean
prepared
(8 rows)
```

4.2 Cambio de nombre del producto 'Son antes' por 'Antes'.

```
SELECT id_product FROM practica_cassandra.products WHERE product='Son antes' ALLOW FILTERING;
```

```
UPDATE practica_cassandra.products
  SET product = 'Antes'
  WHERE id_product IN ('6', '16', '19', '13', '21', '7', '17', '9', '15', '10', '4',
'3', '5', '18', '14', '8', '20', '2', '22', '12', '23', '24', '11', '1');
```

```
SELECT id_product,product FROM practica_cassandra.products WHERE product='Antes'
ALLOW FILTERING;
```

En este caso se accede a la tabla productos, se busca el producto 'Son antes' y se obtienen sus identificadores. Después se actualiza la tabla cambiando el nombre 'Son antes' por 'Antes' para los identificadores obtenidos. Por último, se comprueba el resultado:

```
cqlsh:practica_cassandra> SELECT id_product FROM practica_cassandra.products WHERE product='Son antes' ALLOW FILTERING;
```

| id_product |
|------------|
| 6 |
| 16 |
| 19 |
| 13 |
| 21 |
| 7 |
| 17 |
| 9 |
| 15 |
| 10 |
| 4 |
| 3 |
| 5 |
| 18 |
| 14 |
| 8 |
| 20 |
| 2 |
| 22 |
| 12 |
| 23 |
| 24 |
| 11 |
| 1 |

(24 rows)

```
cqlsh:practica_cassandra> SELECT id_product,product FROM practica_cassandra.products WHERE product='Antes' ALLOW FILTERING;
```

| id_product | product |
|------------|---------|
| 6 | Antes |
| 16 | Antes |
| 19 | Antes |
| 13 | Antes |
| 21 | Antes |
| 7 | Antes |
| 17 | Antes |
| 9 | Antes |
| 15 | Antes |
| 10 | Antes |
| 4 | Antes |
| 3 | Antes |
| 5 | Antes |
| 18 | Antes |
| 14 | Antes |
| 8 | Antes |
| 20 | Antes |
| 2 | Antes |
| 22 | Antes |
| 12 | Antes |
| 23 | Antes |
| 24 | Antes |
| 11 | Antes |
| 1 | Antes |

(24 rows)

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

```
INSERT INTO practica_cassandra.products (formatt, product, id_product) VALUES ('roasted bean', 'Duende y luna', '6497') IF NOT EXISTS;
```

```
SELECT id_product,product,formatt FROM practica_cassandra.products WHERE product='Duende y luna' ALLOW FILTERING;
```

Para esta consulta se inserta en la tabla products una nueva fila con los campos formatt='roasted bean', product='Duende y luna', id_product= '6497'. El resultado es el siguiente:

```
cqlsh:practica_cassandra> SELECT id_product,product,formatt FROM practica_cassandra.products WHERE product='Duende y luna' ALLOW FILTERING;
```

| id_product | product | formatt |
|------------|---------------|--------------|
| 6497 | Duende y luna | roasted bean |
| 3237 | Duende y luna | capsules |
| 3239 | Duende y luna | capsules |
| 3240 | Duende y luna | capsules |
| 3238 | Duende y luna | capsules |

(5 rows)

4.4 Referencias asociadas al producto 'Milagros' con formato capsulas (formatt = 'capsules').

```
SELECT barcode FROM practica_cassandra.products WHERE product='Milagros' AND formatt='capsules' ALLOW FILTERING;
```

```
cqlsh:practica_cassandra> SELECT barcode FROM practica_cassandra.products WHERE product='Milagros' AND formatt='capsules' ALLOW FILTERING;
```

| barcode |
|-----------------|
| Q000771I980323 |
| Q0I11851I601451 |

(2 rows)

4.5 Último pedido realizado por el usuario con nickname='cisni'.

```
SELECT MAX(ORDERDATE),USERR,id_order FROM practica_cassandra.orders WHERE USERR
='cisni' ALLOW FILTERING;
```

```
SELECT ORDERDATE,USERR,id_order FROM practica_cassandra.orders WHERE USERR ='cisni'
AND ORDERDATE='2014-12-17' ALLOW FILTERING;
```

Para esta consulta se ha obtenido la última fecha asociada al usuario *cisni* y a continuación se han buscado todos los pedidos realizados por el mismo usuario en dicha fecha, ya que se ha observado que si hay varias fechas coincidentes sólo se muestra una de ellas.

```
cqlsh:practica_cassandra> SELECT MAX(ORDERDATE),USERR,id_order FROM practica_cassandra.orders WHERE USERR ='cisni' ALLOW FILTERING;
```

| system.max(orderdate) | userr | id_order |
|-----------------------|-------|----------|
| 2014-12-17 | cisni | 38804 |

```
(1 rows)

Warnings :
Aggregation query used without partition key

Read 45605 live rows and 2759 tombstone cells for query SELECT * FROM practica_cassandra.orders WHERE userr = cisni AND LIMIT 100 (see tombstone_warn_threshold)

cqlsh:practica_cassandra> SELECT ORDERDATE,USERR,id_order FROM practica_cassandra.orders WHERE USERR ='cisni' AND ORDERDATE='2014-12-17' ALLOW FILTERING;
```

| orderdate | userr | id_order |
|------------|-------|----------|
| 2014-12-17 | cisni | 38767 |
| 2014-12-17 | cisni | 38766 |
| 2014-12-17 | cisni | 38812 |
| 2014-12-17 | cisni | 38765 |

```
(4 rows)
```

4.6 Cantidad total gastada por el usuario cuyo nickname es 'naki'.

```
SELECT SUM(RETAILPRICE),USERR,id_order FROM practica_cassandra.orders WHERE USERR
='naki' ALLOW FILTERING;
```

Mediante la función **SUM** se ha calculado la cantidad total gastada por el usuario naki y el resultado es el siguiente:

```
cqlsh:practica_cassandra> SELECT SUM(RETAILPRICE),USERR FROM practica_cassandra.orders WHERE USERR ='naki' ALLOW FILTERING;
```

| system.sum(retailprice) | userr |
|-------------------------|-------|
| 3640.11011 | naki |

```
(1 rows)
```

4.7 Inserción de una nueva referencia (barcode= QIO99947091) para el producto 'Como y tronar' con formato cápsulas.

```
INSERT INTO practica_cassandra.products (barcode, formatt, product, id_product)
VALUES ('QIO99947091', 'capsules', 'Como y tronar', '6498') IF NOT EXISTS;
```

```
SELECT id_product,product,formatt,barcode FROM practica_cassandra.products WHERE
product='Como y tronar' ALLOW FILTERING;
```

Para esta consulta se inserta en la tabla products una nueva fila con los campos barcode='QIO99947091', formatt='capsules', producto='Como y tronar', id_product='6498'. El resultado es el siguiente:

```
cqlsh:practica_cassandra> INSERT INTO practica_cassandra.products (barcode, formatt, product, id_product) VALUES ('QIO99947091', 'capsules', 'Como y tronar', '6498') IF NOT EXISTS;

[applied]
-----
      True

cqlsh:practica_cassandra> SELECT id_product,product,formatt,barcode FROM practica_cassandra.products WHERE product='Como y tronar' ALLOW FILTERING;
```

| id_product | product | formatt | barcode |
|------------|---------------|--------------|-----------------|
| 1442 | Como y tronar | prepared | IQI566801433916 |
| 1444 | Como y tronar | capsules | III500390920868 |
| 6498 | Como y tronar | capsules | QIO99947091 |
| 1443 | Como y tronar | capsules | 0IQ87311Q947313 |
| 1446 | Como y tronar | roasted bean | QQ0605451206885 |
| 1447 | Como y tronar | raw bean | IQ0448870834761 |
| 1448 | Como y tronar | raw bean | QQ0850791315077 |
| 1445 | Como y tronar | roasted bean | III896630382888 |
| 1441 | Como y tronar | prepared | QQ089712Q822661 |

(9 rows)