

FUNDAMENTOS DE PROYECTOS BIG DATA

Uso de PIG y HIVE

José Manuel Bustos Muñoz

- **Carga de datos.**

Vamos a trabajar con los ficheros del archivo “sampleData.tar”, así que lo copiamos a la carpeta de usuario del cluster: “**scp sampleData.tar uamibm104@150.244.65.33:/home/uamibm104**”.

Nos conectamos via ssh al cluster, y podemos ver con un ls que se ha copiado el archivo con los ficheros. Descomprimimos el archivo .tar: “**tar -xvf sampleData.tar**”.

Nos posicionamos en la carpeta de los ficheros y con la instrucción “**hdfs dfs -put nombreFichero.csv**” los copiamos al sistema de ficheros HDFS y ya podemos trabajar con ellos con Hive y Pig.

```
Customer.csv  Product.csv  Sales.csv
[[uamibm104@nodogestion001d WithRowHeaders]$ hdfs dfs -ls
Found 15 items
drwx-----  - uamibm104 hdfs          0 2017-11-11 19:00 .Trash
drwx-----  - uamibm104 hdfs          0 2017-12-02 11:47 .staging
-rw-r--r--  2 uamibm104 hdfs  406697 2017-11-25 13:45 AtletasOlimpicos.csv
-rw-r--r--  2 uamibm104 hdfs   2513 2017-12-02 17:26 Customer.csv
-rw-r--r--  2 uamibm104 hdfs  281111 2017-11-29 23:14 Players.csv
-rw-r--r--  2 uamibm104 hdfs   1725 2017-12-02 17:26 Product.csv
-rw-r--r--  2 uamibm104 hdfs   1146 2017-12-02 17:26 Sales.csv
-rw-r--r--  2 uamibm104 hdfs  5117407 2017-11-29 23:15 Seasons_stats.csv
drwxr-xr-x  - uamibm104 hdfs          0 2017-11-29 23:17 WordCount
drwxr-xr-x  - uamibm104 hdfs          0 2017-10-21 13:30 myTestDir
drwxr-xr-x  - uamibm104 hdfs          0 2017-10-21 13:32 myTestDir2
-rw-r--r--  2 uamibm104 hdfs  80373 2017-11-29 23:15 players_stats.csv
-rw-r--r--  2 uamibm104 hdfs  317618 2017-11-04 13:42 quijote.txt
drwxrwxrwx  - uamibm104 hdfs          0 2017-11-11 12:11 salida
drwxr-xr-x  - uamibm104 hdfs          0 2017-11-11 13:28 salida_prueba
```

HIVE

1. Enumera y explica los pasos que has dado para la carga de datos.

Creamos una tabla para cada archivo: customer_bustos, product_bustos, sales_bustos. Las tablas se crean con la instrucción **CREATE TABLE nombreTabla**, y una vez creada se cargan los datos desde el fichero correspondiente con la instrucción **LOAD DATA INPATH nombreFichero**.

Ejemplo creación y carga de la tabla customer:

```
CREATE TABLE customer_bustos (name string, apellido string, status string, telefono string, custID int, domicilio string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE TBLPROPERTIES ("skip.header.line.count"="1");
```

```
LOAD DATA INPATH 'Customer.csv' OVERWRITE INTO TABLE customer_bustos;
```

Haríamos el mismo proceso para la creación y carga de las tablas de productos y de ventas:

Crear y cargar tabla customer_bustos

```
[hive> CREATE TABLE customer_bustos (name string, apellido string, status string, telefono string, cu  
stID int, domicilio string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE TBLPROP  
ERTIES ("skip.header.line.count"="1");  
OK  
Time taken: 0.176 seconds  
[hive> LOAD DATA INPATH 'Customer.csv' OVERWRITE INTO TABLE customer_bustos;  
Loading data to table default.customer_bustos  
Table default.customer_bustos stats: [numFiles=1, numRows=0, totalSize=2561, rawDataSize=0]  
OK  
Time taken: 0.311 seconds  
hive> _
```

Crear y cargar tabla product_bustos

```
[hive> CREATE TABLE product_bustos (name string, descrip string, categoria string, qty int, number in  
t, packaged string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE TBLPROPERTIES ( "  
skip.header.line.count"="1");  
OK  
Time taken: 0.087 seconds  
[hive> LOAD DATA INPATH 'Product.csv' OVERWRITE INTO TABLE product_bustos;  
Loading data to table default.product_bustos  
Table default.product_bustos stats: [numFiles=1, numRows=0, totalSize=1792, rawDataSize=0]  
OK  
Time taken: 0.24 seconds  
hive> _
```

Crear y cargar tabla sales_bustos

```
[hive> CREATE TABLE sales_bustos (custID int, number int, qty int, fecha string, salesID int) ROW FOR  
MAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE TBLPROPERTIES ("skip.header.line.count"="1  
");  
OK  
Time taken: 0.139 seconds  
[hive> LOAD DATA INPATH 'Sales.csv' OVERWRITE INTO TABLE sales_bustos;  
]
```

2. Mostrar todos los productos, cuyo campo **CATEGORY** sea **Video**.

```
select * from product_bustos where categoria="Video";
```

```
[hive> select * from product_bustos where categoria=="Video";
OK
93G Video      Video Card Jargon Brand 93G      Video    80      99202  dvd:manual:game
84G1 Video     Video Card Jargon Brand 84F1     Video    14      99207  dvd:manual:hdmicable
09K Video      Video Card Tiger Brand 84F1     Video     5      98243  manual:game
Time taken: 0.388 seconds, Fetched: 3 row(s)
hive> _
```

Como se ve en la imagen se obtienen 3 productos para la categoría buscada.

3. Una vez mostrados los productos que tienen **CATEGORY = Video**, ¿qué productos tienen el primer elemento del vector **PACKAGED_WITH = dvd**?

```
select * from product_bustos where categoria="Video" and packaged like "dvd%";
```

```
[hive> select * from product_bustos where categoria=="Video" and packaged like "dvd%";
OK
93G Video      Video Card Jargon Brand 93G      Video    80      99202  dvd:manual:game
84G1 Video     Video Card Jargon Brand 84F1     Video    14      99207  dvd:manual:hdmicable
Time taken: 0.119 seconds, Fetched: 2 row(s)
hive> _
```

De los productos de la categoría video que como se vio antes son 3, si filtramos también para ver que el primer elemento del campo **PACKAGED_WITH** sea dvd, se obtienen 2 productos que lo cumplen.

4. Averigua cuántos productos de cada categoría hay. Pista: usa **GROUP BY**.

```
select categoria, count(*) from product_bustos group by categoria;
```

```
[hive> select categoria,count(*) from product_bustos group by categoria;
Query ID = uamibm104_20171203120822_08144104-55d3-4bf5-91f1-7191df5138cf
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
To control the number of reducers, use -R, --reducers (for e.g. horton)
```

Se obtiene lista de categorías y el número de productos de cada una. Se ve que Optical es la que más productos tiene con 4, y el resto tienen 3.

```
[hive> select categoria, count(*) from product_bustos group by categoria;
OK
CPU      3
Case     3
HD       3
MB       3
Optical  4
Power    3
Ram      3
Video    3
Time taken: 27.548 seconds, Fetched: 8 row(s)
hive> _
```

5. Lista la categoría (**CATEGORY**) que tiene más de cuatro productos.

Con la sentencia: “**select categoria, count(*) as contador from product_bustos group by categoria order by contador desc limit 1;**” obtenemos la categoría con más número de productos, que en este caso no supera el número de 4.

```
Total mapreduce jobs: 0 Time spent: 0 seconds (0.00 ms)
OK
Optical 4
Time taken: 51.917 seconds, Fetched: 1 row(s)
```

6. ¿Cuántas ventas se realizaron el 24 de enero de 2012?. Contabiliza el total y ordena los resultados por el campo **SALE_ID**.

Con la sentencia: “**select count(*) from sales_bustos where fecha=“1/24/2012”;**” se obtiene el número de ventas ocurridas en esa fecha, que serían 19.

```
[hive> select count(*) from sales_bustos where fecha=="1/24/2012";
Query ID = uamibm104_20171203131140_05de94a8-c28f-4782-8324-b7cccd2eaee95
Total jobs = 1
Launching Job 1 out of 1
```

Con la sentencia: “**select salesID, fecha from sales_bustos where fecha=‘1/24/2012’ order by salesID;**” se obtiene el listado con cada id de venta y fecha en la que ocurrió la venta, ordenado por el id.

```
[hive> select salesID, fecha from sales_bustos where fecha=="1/24/2012" order by salesID;
Query ID = uamibm104_20171203130520_c8bbdd45-f0dc-41e0-9512-b87abe96ebc9
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
```

```
Total MapReduce CPU Time Spent: 3 seconds 650 msec
OK
34842 1/24/2012
34843 1/24/2012
34844 1/24/2012
34845 1/24/2012
34846 1/24/2012
34847 1/24/2012
34848 1/24/2012
34849 1/24/2012
34850 1/24/2012
34851 1/24/2012
34852 1/24/2012
34853 1/24/2012
34854 1/24/2012
34855 1/24/2012
34856 1/24/2012
34857 1/24/2012
34858 1/24/2012
34859 1/24/2012
34860 1/24/2012
Time taken: 26.152 seconds, Fetched: 19 row(s)
```

En la imagen se aprecia el resultado, que son 19 registros como se dijo antes, y se ve los ids de las ventas que ocurrieron el día solicitado.

7. Muestra las ventas de la categoría (**CATEGORY**) **Optical** se realizaron el día 9 de enero de 2012.

“select sa.salesID, pr.categoría, sa.fecha from sales_bustos as sa, product_bustos as pr where sa.number=pr.number and sa.fecha=“1/9/2012” and pr.categoría=“Optical”;”

```
[hive> select sa.salesID, pr.categoría, sa.fecha from sales_bustos as sa, product_bustos as pr where
sa.number==pr.number and sa.fecha=="1/9/2012" and pr.categoría=="Optical";
Query ID = uamibm104_20171203132235_fd154579-423d-4f0c-a1c0-ea526e189ca4
Total jobs = 1
```

Al mostrar el resultado vemos el id de cada venta, la categoría que se puede ver que es optical como se dice en el enunciado, y la fecha requerida. Las ventas que cumplen esto son 3.

```
Total MapReduce CPU Time Spent: 2 seconds 0 msec
OK
34824  Optical 1/9/2012
34828  Optical 1/9/2012
34839  Optical 1/9/2012
Time taken: 25.687 seconds, Fetched: 3 row(s)
```

8. ¿Cuál es el cliente que más compras ha realizado? Lista su nombre, los productos que ha comprado y las fechas en las que ha realizado cada compra.

Con la sentencia: “**select custID, count(sales_bustos.salesID) as contador from sales_bustos group by custID order by contador desc limit 1;**” se obtiene el id del cliente que más compras ha realizado.

Sería el cliente con id 64, que ha realizado 3 compras.

```
STAGE 1: map: 1 REDUCE: 1   Cumulative CPU: 2.75 SEC    HDFS Read: 0.415 HDFS Write: 0.000 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 760 msec
OK
64      3
Time taken: 53.167 seconds, Fetched: 1 row(s)
```

Con la sentencia: “**select c.name, p.name, s.fecha from customer_bustos as c, product_bustos as p, sales_bustos as s where c.custID=64 and c.custID=s.custID and p.number=s.number order by s.fecha desc;**” se obtienen las compras de ese cliente con la información requerida.

```
OK
Mello  J Case 1501      1/9/2012
Mello  09K Video       1/9/2012
Mello  J Power 300W     1/24/2012
Time taken: 35.078 seconds, Fetched: 3 row(s)
hive> □
```

PIG

1. Enumera y explica los pasos que has dado para la carga de datos.

Para la creación y carga de los datos desde los ficheros en PIG se utiliza la fórmula:

```
customer_bustos = LOAD 'Customer.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage('','','YES_MULTILINE','NOCHANGE','SKIP_INPUT_HEADER') AS (name:chararray, apellido:chararray, status:chararray, telefono:chararray, custID:int, domicilio:chararray);
```

Se realiza el proceso para los tres ficheros, y después de cargar cada uno se hace un “DUMP” para mostrar los datos cargados.

customer_bustos

```
[grunt> customer_bustos = LOAD 'Customer.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage('','','YES_MULTILINE','NOCHANGE','SKIP_INPUT_HEADER') AS (name:chararray, apellido:chararray, status:chararray, telefono:chararray, custID:int, domicilio:chararray);
[grunt> DUMP customer_bustos;
2017-12-02 17:25:59 [main] INFO org.apache.pig.Main - Starting Pig at Fri Dec 02 17:25:59 EST 2017
Total input paths to process : 1
(2 GB Memory E,2 GB Memory ECC,Ram,3000,87655,manual:heatsink)
(4 GB Memory E,4 GB Memory ECC,Ram,1000,87659,manual:heatsink)
(16 GB Memory E,16 GB Memory ECC,Ram,238,87634,manual)
(500 GB HD J,500 GB HD Panther Brand,HD,200,45628,atacable:manual)
(500 GB HD T,500 GB HD Tiger Brand,HD,498,45641,satacable:manual)
(1 TB HD J,1 TB HD Jargon Brand,HD,231,45691,)
(4 Core CPU J3,4 Core CPU Jargon Brand 3 GHZ ,CPU,50,98820,thermalpaste:heatsink:manual)
(2 Core CPU J2,2 Core CPU Jargon Brand 2 GHZ ,CPU,118,98838,thermalpaste:heatsink)
(1 Core CPU J2,1 Core CPU Jargon Brand 2 GHZ ,CPU,203,98792,thermalpaste:heatsink)
(94F991 MB,Motherboard F991 CPU,MB,19,282299,)
(94G822 MB,Motherboard G822 CPU,MB,30,282109,)
(93H772 MB,Motherboard H772 CPU,MB,15,282009,cables:screws)
(93G Video,Video Card Jargon Brand 93G,Video,80,99202,dvd:manual:game)
(84G1 Video,Video Card Jargon Brand 84F1,Video,14,99207,dvd:manual:hdmi:cable)
(09K Video,Video Card Tiger Brand 84F1,Video,5,98243,manual:game)
(J Case 1500,Computer Case Jargon Brand Style 1500,Case,20,77623,fans:manual:screws)
(J Case 1501,Computer Case Jargon Brand Style 1501,Case,18,77624,fans:manual:screws)
(T Case 4332,Computer Case Tiger Brand Style 4332,Case,7,88211,fans:manual:screws:watercooler)
(J Power 300W,Power Supply Jargon Brand 300 Watts,Power,28,92387,cables:screws)
(J Power 500W,Power Supply Jargon Brand 500 Watts,Power,17,92373,cables:screws)
(T Power 300W,Power Supply Tiger Brand 300 Watts,Power,8,93347,cables:screws)
(DVD J INT,DVD Jargon Brand Internal,Optical,23,88734,manual)
(DVD J EXT,DVD Jargon Brand External,Optical,45,88821,)
(DVD T INT,DVD Tiger Brand Internal,Optical,19,82331,satacable:manual)
(DVD T EXT,DVD Tiger Brand External,Optical,17,82337,satacable:manual)
[grunt>
```

product_bustos

```
[grunt> product_bustos = LOAD 'Product.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage('','','YES_MULTILINE','NOCHANGE','SKIP_INPUT_HEADER') AS (name:chararray, descrip:chararray, categoria:chararray, qty:int, number:int, packaged:chararray);
[grunt> DUMP product_bustos;_
```

sales_bustos

```
[grunt> sales_bustos = LOAD 'Sales.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage('','','YES_MULTILINE','NOCHANGE','SKIP_INPUT_HEADER') AS (custID:int, number:int, qty:int, fecha:chararray, salesID:int);
[grunt> DUMP sales_bustos;_
```

2. Mostrar todos los productos, cuyo campo **CATEGORY** sea **Video**.

“**videos_product = FILTER product_bustos BY categoria==‘Video’;**”

Con “**dump videos_product;**” se muestran los resultados que son 3 registros.

```
[grunt> cont_cat = FILTER product_bustos BY categoria==‘Video’;
Total input paths to process : 1
(93G Video,Video Card Jargon Brand 93G,Video,80,99202,dvd:manual:game)
(84G1 Video,Video Card Jargon Brand 84F1,Video,14,99207,dvd:manual:hdmiicable)
(09K Video,Video Card Tiger Brand 84F1,Video,5,98243,manual:game)
grunt> ]
```

3. Una vez mostrados los productos que tienen **CATEGORY = Video**, ¿qué productos tienen el primer elemento del vector **PACKAGED_WITH = dvd**?

A la relación obtenida en el anterior ejercicio de los productos de categoría video, filtramos por aquellos que empiezan por ‘dvd’: “**resul_dvd = FILTER videos_product BY packaged matches ‘dvd.*’;**”

Son 2 productos los que cumple este nuevo filtro.

```
[grunt> resul_dvd = FILTER videos_product BY packaged matches ‘dvd.*’;
Total input paths to process : 1
(93G Video,Video Card Jargon Brand 93G,Video,80,99202,dvd:manual:game)
(84G1 Video,Video Card Jargon Brand 84F1,Video,14,99207,dvd:manual:hdmiicable)
grunt> ]
```

4. Averigua cuántos productos de cada categoría hay. Pista: usa **GROUP BY**.

Agrupamos los productos por categoría: “**b = GROUP product_bustos BY categoria;**”

Hacemos un recuento de los productos por categoría: “**cont_cat = FOREACH b GENERATE group AS categoria, COUNT(product_bustos.categoria);**”

```
[grunt> cont_cat = FOREACH b GENERATE group AS categoria, COUNT(product_bustos.categoria);
[grunt> dump cont_cat;
```

Mostramos el resultado con “**dump cont_cat;**”. Se ven las categorías y el número de productos de cada una. óptical es la que más con 4 productos, el resto tienen 3.

```
[grunt> Total input paths to process : 1
(HD,3)
(MB,3)
(CPU,3)
(Ram,3)
(Case,3)
(Power,3)
(Video,3)
(Optical,4)
grunt> ]
```

5. Lista la categoría (**CATEGORY**) que tiene más de cuatro productos.

A la agrupación por categorías se realiza un contador por productos: “**contador_categorias = FOREACH b GENERATE group AS cat, COUNT(product_bustos.categoría) AS cont;**”

```
[grunt> contador_categorias = FOREACH b GENERATE group AS cat, COUNT(product_bustos.categoría) AS cont;
```

Se filtra por aquellas categorías que tengan más de 4 productos: “**filter_contador = FILTER contador_categorias BY cont > 4;**”

Se obtienen 0 registros con más de 4 productos.

```
Output(s):
Successfully stored 0 records in: "hdfs://nodogestion001d.iu.uam.es:8020/tmp/temp127029102/tmp-33645
6431"
```

Se ordena el contador de categorías de forma descendente, y se limita a 1 el resultado para ver la categoría con más productos. Se ve que dicha categoría es “Optical” con 4 productos, por lo tanto ninguna categoría cumple el filtro de más de 4.

“**cont_categ_orden = ORDER contador_categorias BY cont DESC;**”

“**limit cont_categ = LIMIT cont_categ_orden 1;**”

```
Total input paths to process : 1
(Optical,4)
grunt>
```

6. ¿Cuántas ventas se realizaron el 24 de enero de 2012?. Contabiliza el total y ordena los resultados por el campo **SALE_ID**.

Se filtran las ventas por la fecha indicada: “**ventas_fecha = FILTER sales_bustos BY fecha == ‘1/24/2012’;**”

```
[grunt> ventas_fecha = FILTER sales_bustos BY fecha == '1/24/2012';
2017-12-03 19:01:12,256 [main] WARN  org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I
```

Se hace un recuento de las ventas filtradas por fecha anteriormente: “**contador_ventas = FOREACH (GROUP ventas_fecha ALL) GENERATE group AS venta, COUNT(ventas_fecha) AS cont;**”

```
[grunt> contador_ventas = FOREACH (GROUP ventas_fecha ALL) GENERATE group AS venta, COUNT(ventas_fecha) AS cont;
2017-12-03 19:13:19,838 [main] WARN  org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I
MPLICIT_CAST_TO_LONG 1 time(s).
[grunt> DUMP contador_ventas;
]
```

Se obtienen 19 ventas en dicha fecha.

```
2017-12-03 19:14:03,784 [main] INFO  org.apache.pig.b
Total input paths to process : 1
(all,19)
grunt> _
```

7. Muestra las ventas de la categoría (**CATEGORY**) Optical se realizaron el día 9 de enero de 2012.

Se realiza un JOIN de los productos y las ventas por el id de producto o number: “**ventas_optical = JOIN product_bustos BY number, sales_bustos BY number;**”

Se filtra este join por la categoria Optical requerida: “**filter_ventas_optical = FILTER ventas_optical BY categoría=='Optical';**”

```
[grunt> filter_ventas_optical = FILTER ventas_optical BY categoria=='Optical';
2017-12-03 19:40:01,183 [main] WARN  org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning IMPLICIT_CAST_TO_LONG 1 time(s).
[grunt> dump filter_ventas_optical;
```

Una vez tenemos las ventas de la categoria, se filtran por la fecha y ya tenemos el resultado requerido que serían 3 ventas: “**filter_ventas_fecha = FILTER filter_ventas_optical BY fecha=='1/9/2012';**”

```
2017-12-03 19:44:20,702 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReducePlanner
Total input paths to process : 1
(DVD J INT,DVD Jargon Brand Internal,Optical,23,88734,manual,922,88734,24,1/9/2012,34828)
(DVD J INT,DVD Jargon Brand Internal,Optical,23,88734,manual,128,88734,4,1/9/2012,34839)
(DVD J INT,DVD Jargon Brand Internal,Optical,23,88734,manual,922,88734,1,1/9/2012,34824)
grunt> _
```

8. ¿Cuál es el cliente que más compras ha realizado? Lista su nombre, los productos que ha comprado y las fechas en las que ha realizado cada compra.

Agrupamos las ventas por id de cliente, se realiza un recuento de las ventas por cliente, se ordena de forma descendente y se limita a 1 para acabar obteniendo el id del cliente con más ventas así como el número de ventas:

```
"clientes_ventas = GROUP sales_bustos BY custID;"  
"cont_ventas = FOREACH clientes_ventas GENERATE group AS cliente,  
COUNT(sales_bustos.custID) as contador;"  
"order_ventas = ORDER cont_ventas BY contador DESC;"  
"limit_ventas = LIMIT order_ventas 1;
```

```
[grunt> clientes_ventas = GROUP sales_bustos BY custID;  
2017-12-03 19:56:43,111 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> cont_ventas = FOREACH clientes_ventas GENERATE group AS cliente, COUNT(sales_bustos.custID) a]  
s contador;  
2017-12-03 19:57:48,311 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> order_ventas = ORDER cont_ventas BY contador DESC;  
2017-12-03 19:58:35,340 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> limit_ventas = LIMIT order_ventas 1;  
2017-12-03 19:58:53,110 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I  
MPLICIT_CAST_TO_LONG 1 time(s).  
grunt> dump limit_ventas;
```

```
2017-12-03 20:01:03,521 [main] INFO o  
Total input paths to process : 1  
(64,3)  
grunt> _
```

El id del cliente con más ventas sería el 64.

Generamos relaciones de customer, product y sales con los datos que nos interesan de cada uno:

```
"a = foreach customer_bustos generate name, custID;"  
"b = foreach product_bustos generate name, number;"  
"c = foreach sales_bustos generate fecha, salesID, custID, number;"
```

```
[grunt> a = foreach customer_bustos generate name, custID;  
2017-12-03 20:03:49,763 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Enc  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> b = foreach product_bustos generate name, number;  
2017-12-03 20:04:06,788 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Enc  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> c = foreach sales_bustos generate fecha, salesID, custID, number;  
2017-12-03 20:04:16,847 [main] WARN org.apache.pig.newplan.BaseOperatorPlan - Enc
```

Luego generamos dos join uno entre clientes y ventas, y otro entre productos y ventas, para terminar haciendo un JOIN de ambos y obtener una relación de las 3 relaciones origen con los datos que las unen y requerimos.

```
"d = JOIN a BY custID, c BY custID;"  
"e = JOIN b BY number, c BY number;"  
"f = JOIN d BY salesID, e BY salesID;"
```

```
[grunt> f = JOIN d BY salesID, e BY salesID;  
2017-12-03 20:11:12,141 [main] WARN  org.apache.pig.  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> dump f;
```

Filtramos la relación resultante por el id del cliente que se obtuvo como el cliente con más ventas:
"f_filter = FILTER f BY d::c::custID==64;"

```
[8  
[grunt> f_filter = FILTER f BY d::c::custID==64;  
2017-12-03 20:19:20,338 [main] WARN  org.apache.pig.newplan.BaseOperatorPlan - Encountered Warning I  
MPLICIT_CAST_TO_LONG 1 time(s).  
[grunt> dump f_filter;
```

El resultado obtenido son las 3 compras realizadas por ese cliente.

```
2017-12-03 20:20:27,071 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapred.MapReducePl  
Total input paths to process : 1  
(Mello,64,1/9/2012,34826,64,98243,09K Video,98243,1/9/2012,34826,64,98243)  
(Mello,64,1/9/2012,34833,64,77624,J Case 1501,77624,1/9/2012,34833,64,77624)  
(Mello,64,1/24/2012,34847,64,92387,J Power 300W,92387,1/24/2012,34847,64,92387)  
[grunt> resul_filter = FOREACH (GROUP f_filter ALL) GENERATE f_filter.name;
```