

USO DE PIG Y HIVE

Lucian Iacob

2.1 Carga del dataset

1. Crea una carpeta en HDFS en la ruta /hadoop/dataset

Tras haber hecho ya los pasos iniciales para arrancar Hadoop, acceder al hive-server y copiar ahí el archivo con el que vamos a trabajar:

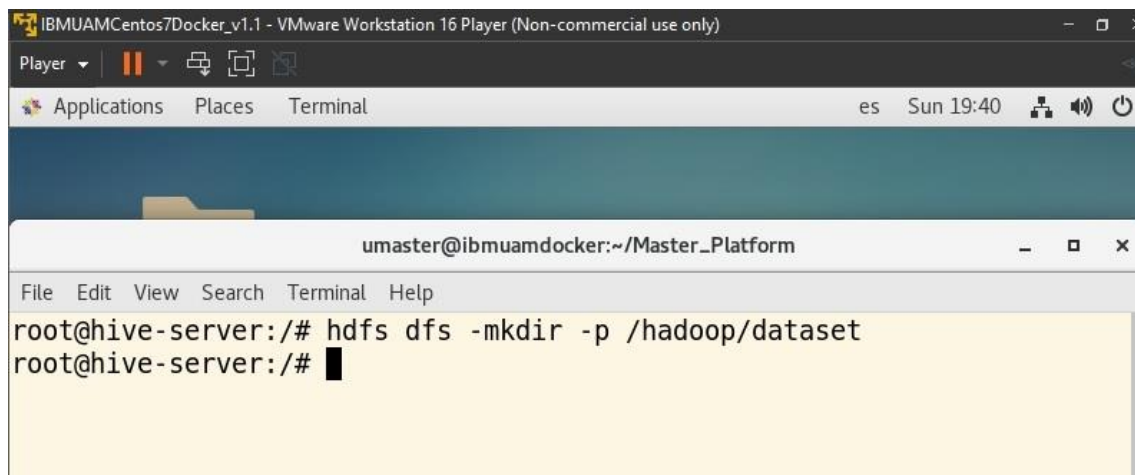
```
>> cd Master_Platform
```

```
>> systemctl start Docker
```

```
>> docker-compose start
```

```
>> docker exec -it hive-server bash
```

```
>> docker cp multas_semaforos_mod.txt hive-server:.
```




The screenshot shows a terminal window titled 'umaster@ibmuamdocker:~/Master_Platform'. The terminal output shows the command 'root@hive-server:/# hdfs dfs -mkdir -p /hadoop/dataset' being executed, followed by a prompt 'root@hive-server:/#'.

2. Descarga el fichero multas_semaforos.txt desde la plataforma virtual y cópialo en la ruta que acabas de crear.

3. Para comprobar que el archivo se ha copiado bien:

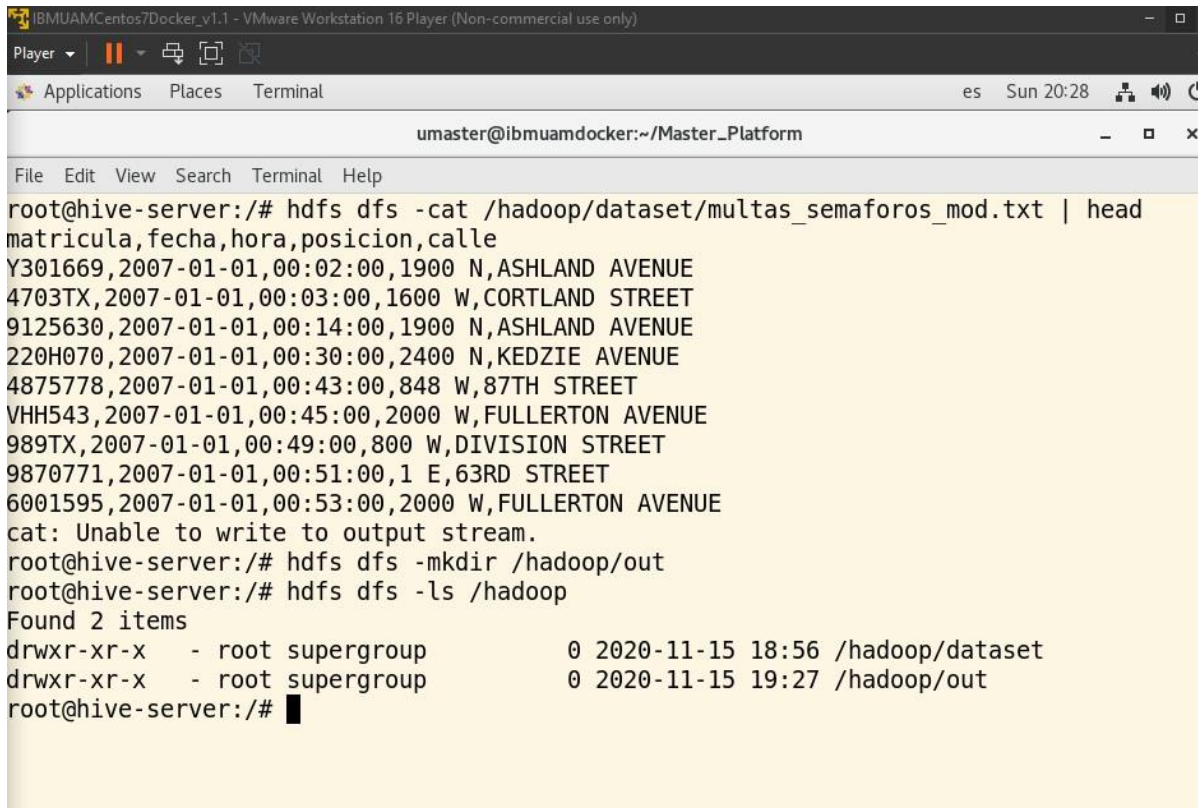
a. (0,25 puntos) Lista el contenido de la carpeta, incluyendo el tamaño del archivo.



The screenshot shows a terminal window titled 'umaster@ibmuamdocker:~/Master_Platform'. The terminal output shows the command 'root@hive-server:/# clear' followed by 'root@hive-server:/# hdfs dfs -put multas_semaforos_mod.txt /hadoop/dataset'. Then, the command 'root@hive-server:/# hdfs dfs -ls -h /hadoop/dataset' is executed, resulting in the output: 'Found 1 items' and a table listing the file 'multas_semaforos_mod.txt' with a size of 200.9 M and a timestamp of 2020-11-15 18:56.

b. (0,25 puntos) Muestra las primeras líneas del fichero, leyendo directamente de HDFS (sin copiar al sistema de ficheros local) en línea de comandos.

4. (0,25 puntos) Crea la carpeta /hadoop/out en HDFS para alojar los resultados de los ejercicios.



```
umaster@ibmuamdocker:~/Master_Platform
File Edit View Search Terminal Help
root@hive-server:/# hdfs dfs -cat /hadoop/dataset/multas_semaforos_mod.txt | head
matricula,fecha,hora,posicion,calle
Y301669,2007-01-01,00:02:00,1900 N,ASHLAND AVENUE
4703TX,2007-01-01,00:03:00,1600 W,CORTLAND STREET
9125630,2007-01-01,00:14:00,1900 N,ASHLAND AVENUE
220H070,2007-01-01,00:30:00,2400 N,KEDZIE AVENUE
4875778,2007-01-01,00:43:00,848 W,87TH STREET
VHH543,2007-01-01,00:45:00,2000 W,FULLERTON AVENUE
989TX,2007-01-01,00:49:00,800 W,DIVISION STREET
9870771,2007-01-01,00:51:00,1 E,63RD STREET
6001595,2007-01-01,00:53:00,2000 W,FULLERTON AVENUE
cat: Unable to write to output stream.
root@hive-server:/# hdfs dfs -mkdir /hadoop/out
root@hive-server:/# hdfs dfs -ls /hadoop
Found 2 items
drwxr-xr-x - root supergroup          0 2020-11-15 18:56 /hadoop/dataset
drwxr-xr-x - root supergroup          0 2020-11-15 19:27 /hadoop/out
root@hive-server:/#
```

2.2 Consultas (9 puntos)

Rellena esta tabla con el tiempo (segundos) que tarda en realizarse cada consulta (0,5 puntos).

Consultas	Pig Local	Pig MapReduce	Hive
C1	1min 7s	6min 15s	2min 39s
C2	1min 28 s	2min 56 s	1min 11s
C3	1min 20 s	4min 55s	4min 50s
C4	2 min 13s	8 min 13s	4min 55s

Analiza los resultados obtenidos e intenta justificar porqué se obtienen esos resultados. ¿Cuál es más rápido Pig o Hive? ¿Por qué? (1 punto)

De primeras se ve que la opción más rápida en este caso es la local. Los tiempos de más reduce son claramente superiores. No obstante, si nos fijamos en las trazas de las queries vemos que el tiempo consumido por la CPU es mucho menor que en el caso local. Esto me lleva a pensar que el archivo es demasiado pequeño para que de verdad se explote la capacidad de cómputo que nos da Hadoop y su HDFS. El tiempo perdido en gestionar qué nodo hace qué y mover los cálculos a través de la red supone mucho más tiempo que el cálculo en sí. Supongo que conforme aumenta de tamaño el fichero este balance se equilibra.

Por otro lado, se ve que, dentro del ambiente HDFS Hive es más rápido que Pig por bastante. Igual es por no haber hecho las órdenes de la manera más eficiente al no tener suficiente conocimiento del lenguaje, pero la diferencia es notable.

2.2.1. Pig (4 puntos)

Abre la shell de Pig en modo local y ejecuta las siguientes operaciones sobre el dataset.

Para iniciar como se nos ha pedido:

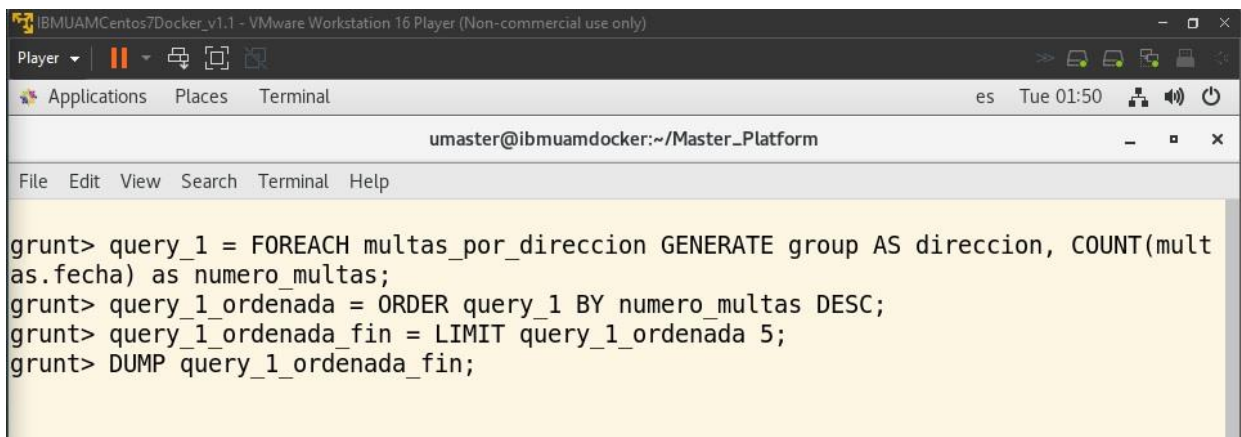
```
>> pig -x local
```

Y luego cargamos el dataset de la siguiente forma:

```
>>      multas      =      LOAD      '/multas_semaforos_mod.txt'      USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',',      'YES_MULTILINE',      'NOCHANGE',
'SKIP_INPUT_HEADER') AS (matricula:chararray, fecha:chararray, hora:chararray, posicion:chararray,
calle:chararray);
```

1. (C1. 0,5 puntos) Lista las 5 cámaras que más multas han generado.

```
grunt> multas_por_direccion = GROUP multas BY (posicion, calle);
```



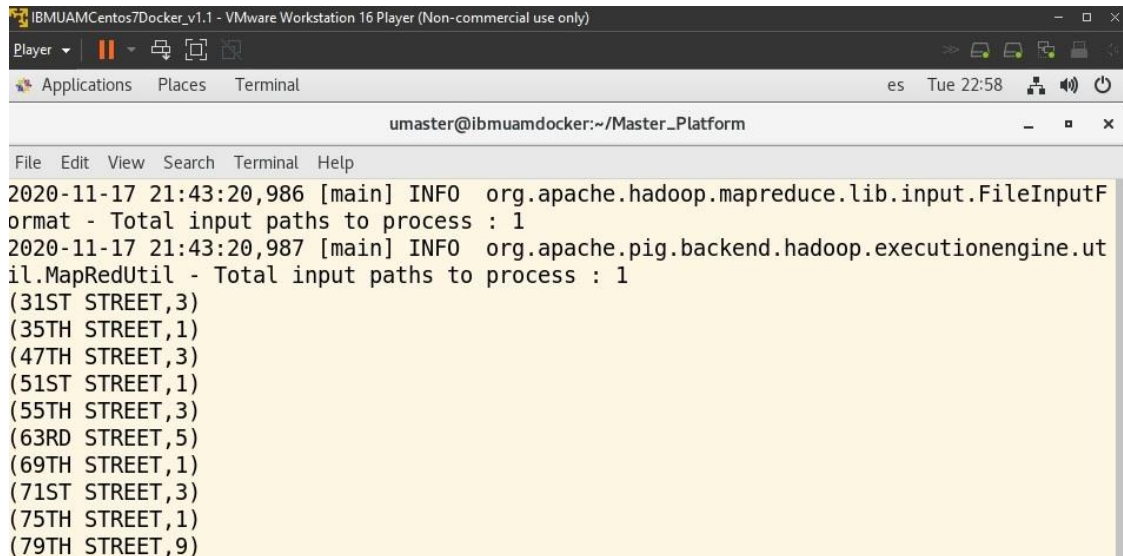
```
root@hive-server:/output_pig_local/query_1# cat part-r-00000
(400 W,BELMONT AVENUE)*94466
(4200 S,CICERO AVENUE)*78265
(1000 W,HOLLYWOOD AVENUE)*65977
(400 S,WESTERN AVENUE)*63825
(8900 S,STONY ISLAND)*60656
root@hive-server:/output_pig_local/query_1#
```

En la primera imagen muestro el group by que hice para tener cada cámara única y en la siguiente los distintos comandos para obtener la query. Las pongo por separado básicamente porque se me olvidó mostrarla al hacer la captura de pantalla de todo el proceso lógico. Por último, muestro el resultado de la query aunque luego lo adjunte en la carpeta.

2. (C2. 0,5 puntos) Indica el número de cámaras que hay por cada una de las calles.

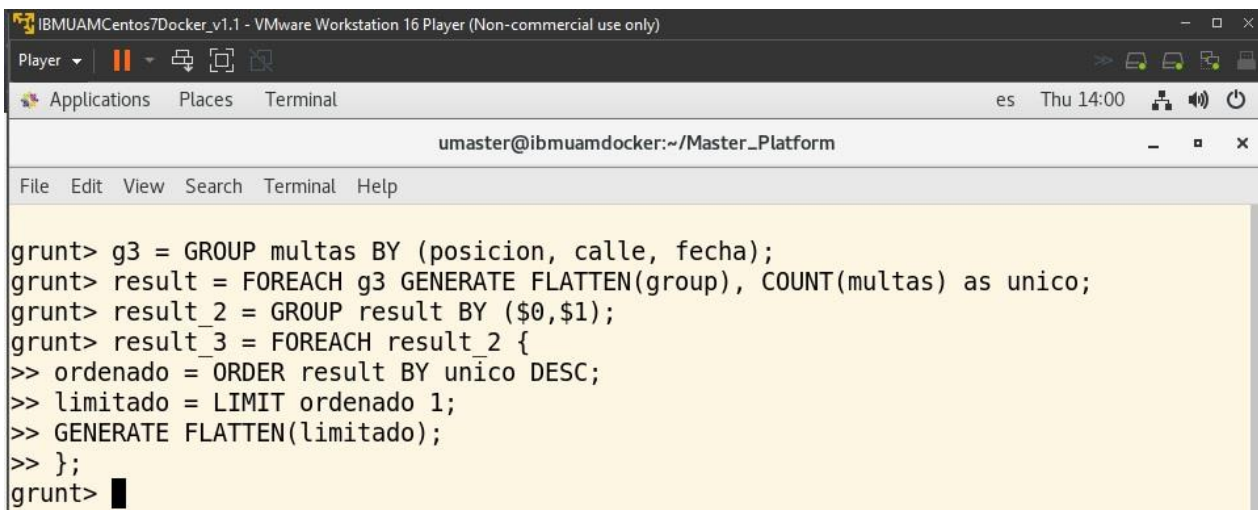
```
grunt> por_calle = GROUP multas BY calle;  
grunt> q2 = FOREACH por_calle {  
>> posiciones = DISTINCT multas.posicion;  
>> GENERATE group AS calle, COUNT(posiciones) as pos;  
>> };
```

Agrupo por calle, y para cada calle miro las posiciones únicas que hay para luego contarlas. Hay un corte de por medio porque las instrucciones intermedias estaban mal. A continuación, incluyo una imagen para hacernos a la idea del resultado de la query aunque solo muestre las primeras líneas. El resultado completo irá adjuntado.



```
2020-11-17 21:43:20,986 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1  
2020-11-17 21:43:20,987 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1  
(31ST STREET,3)  
(35TH STREET,1)  
(47TH STREET,3)  
(51ST STREET,1)  
(55TH STREET,3)  
(63RD STREET,5)  
(69TH STREET,1)  
(71ST STREET,3)  
(75TH STREET,1)  
(79TH STREET,9)
```

3. (C3. 1 punto) De cada cámara, indica qué día ha registrado más multas y cuántas multas.



```
grunt> g3 = GROUP multas BY (posicion, calle, fecha);  
grunt> result = FOREACH g3 GENERATE FLATTEN(group), COUNT(multas) as unico;  
grunt> result_2 = GROUP result BY ($0,$1);  
grunt> result_3 = FOREACH result_2 {  
>> ordenado = ORDER result BY unico DESC;  
>> limitado = LIMIT ordenado 1;  
>> GENERATE FLATTEN(limitado);  
>> };  
grunt>
```

Agrupando por cámara y fecha contamos cuántas multas ha puesto cada día. Y luego para cada cámara nos quedamos con la fecha en la que más multas ha puesto limitando a 1 línea por grupo tras ordenar de manera descendente, y cuántas ha puesto. No muestro los resultados por no recargar más esto, pero claramente lo adjunto.

4. (C4. 1 punto) Indica qué matrícula ha sido la más multada, mostrando la fecha/hora y la calle de cada una de sus multas.

```

umaster@ibmuamdocker:~/Master_Platform
File Edit View Search Terminal Help

grunt> por_mat = GROUP multas BY matricula;
grunt> result = FOREACH por_mat GENERATE FLATTEN (group) as matricula, COUNT(multas)
as num_multas;
grunt> result_2 = ORDER result BY num_multas DESC;
grunt> result_2_lim = LIMIT result_2 1;
grunt> filtrado = FILTER multas BY (matricula == result_2_lim.matricula);
grunt>

```

Agrupamos por matrícula para poder contar cuántas multas lleva cada coche y usar la que tenga más como filtro para sacar la info del archivo original.

Para guardar los archivos usamos el comando:

```
>> STORE [query_x] INTO [fichero local del hive-server] USING PigStorage(',');
```

Ahora cierra la shell de Pig y vuelve a arrancarlo en modo mapreduce. Ejecuta las consultas anteriores comprobando cuántos trabajos de Hadoop son necesarios para cada una (1 punto).

Para cargar los datos ahora hay que usar:

```

multas = LOAD 'hdfs:///hadoop/dataset/multas_semaforos_mod.txt' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'YES_MULTILINE', 'NOCHANGE',
'SKIP_INPUT_HEADER') AS (matricula:chararray, fecha:chararray, hora:chararray, posicion:chararray,
calle:chararray);

```

En las siguientes capturas vemos la parte de Job Stats del log de la consulta. La primera columna corresponde al nombre de la tarea, la 2ª es el número de tareas Map y la 3ª, las Reduce. Así mismo podemos ver el tiempo de ejecución de cada tarea arriba del todo.

- C1

```

HadoopVersion  PigVersion  UserId  StartedAt          FinishedAt          Features
2.7.4    0.17.0    root    2020-11-20 23:26:00  2020-11-20 23:32:15  GROUP_BY,ORDER_BY,LIMIT

Success!

Job Stats (time in seconds):
JobId  Maps  Reduces  MaxMapTime  MinMapTime  AvgMapTime  MedianMapTimeMaxReduceTim
Outputs
job_1605914551937_0001  2      1      n/a      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0002  1      1      n/a      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0003  1      1      n/a      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0004  1      1      n/a      n/a      n/a      n/a      n/a      n/a      n/a
461946,

Input(s):
Successfully read 0 records from: "hdfs:///hadoop/dataset/multas_semaforos_mod.txt"

```

- C2

```
HadoopVersion  PigVersion      UserId  StartedAt      FinishedAt      Features
2.7.4   0.17.0   root    2020-11-20 23:44:42    2020-11-20 23:47:38    GROUP_BY

Success!

Job Stats (time in seconds):
JobId  Maps    Reduces MaxMapTime    MinMapTime    AvgMapTime    MedianMapTime
re Outputs
job_1605914551937_0005  2      1      n/a      n/a      n/a      n/a      n/a      n/a
9000/tmp/temp-1737353072/tmp-1066973692,

Input(s):
Successfully read 0 records from: "hdfs:///hadoop/dataset/multas_semaforos_mod.txt"
```

- C3

```
HadoopVersion  PigVersion      UserId  StartedAt      FinishedAt      Features
2.7.4   0.17.0   root    2020-11-21 00:32:42    2020-11-21 00:37:37    GROUP_BY

Success!

Job Stats (time in seconds):
JobId  Maps    Reduces MaxMapTime    MinMapTime    AvgMapTime    MedianMapTime
re Outputs
job_1605914551937_0011  2      1      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0012  1      1      n/a      n/a      n/a      n/a      n/a      n/a
tmp/temp-1737353072/tmp-1513800911,

Input(s):
Successfully read 0 records from: "hdfs:///hadoop/dataset/multas_semaforos_mod.txt"
```

- C4

```
HadoopVersion  PigVersion      UserId  StartedAt      FinishedAt      Features
2.7.4   0.17.0   root    2020-11-21 00:49:07    2020-11-21 00:57:20    GROUP_BY,ORI

Success!

Job Stats (time in seconds):
JobId  Maps    Reduces MaxMapTime    MinMapTime    AvgMapTime    MedianMapTime
re Outputs
job_1605914551937_0015  2      1      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0016  1      1      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0017  1      1      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0018  1      1      n/a      n/a      n/a      n/a      n/a      n/a
job_1605914551937_0019  2      0      n/a      n/a      n/a      n/a      0      00
1541877070,

Input(s):
Successfully read 0 records from: "hdfs:///hadoop/dataset/multas_semaforos_mod.txt"
```

Es un poco inquietante que en la última línea salga que se han leído cero registros de nuestro dataset, pero luego mirando el contenido de la salida se ve que los registros están. Así que no sabría decir el origen de esa línea, porque los datos se han cargado bien usando el comando que hemos indicado al principio de esta sección.

2.2.2. Hive (3,5 puntos)

Abre la shell de Hive y ejecuta las siguientes operaciones.

1. (0,5 puntos) Realiza lo siguiente:

a. Crea una base de datos que se llame prácticas y actívala.

```
umaster@ibmuamdocker:~/Master_Platform
File Edit View Search Terminal Help

hive> CREATE DATABASE practicas;
OK
Time taken: 0.134 seconds
hive> USE practicas;
OK
Time taken: 0.041 seconds
hive>
```

b. Crea una tabla para alojar los datos del dataset. Debes tener en cuenta el formato del fichero y los campos que tiene.

```
umaster@ibmuamdocker:~/Master_Platform
File Edit View Search Terminal Help

hive> CREATE TABLE multas (matricula string, fecha string, hora string, posicion
string, calle string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS T
EXTFILE TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.215 seconds
hive>
```

c. Carga el fichero en la tabla y haz una primera selección de todos los campos con un máximo de 10 registros.

```
umaster@ibmuamdocker:~/Master_Platform
File Edit View Search Terminal Help

hive> LOAD DATA INPATH 'hdfs:///hadoop/dataset/multas_semaforos_mod.txt' OVERWRI
TE INTO TABLE multas;
Loading data to table practicas.multas
OK
Time taken: 1.695 seconds
hive> SELECT * FROM multas LIMIT 10;
OK
Y301669 2007-01-01      00:02:00      1900 N  ASHLAND AVENUE
4703TX  2007-01-01      00:03:00      1600 W  CORTLAND STREET
9125630 2007-01-01      00:14:00      1900 N  ASHLAND AVENUE
220H070 2007-01-01      00:30:00      2400 N  KEDZIE AVENUE
4875778 2007-01-01      00:43:00      848 W   87TH STREET
VHH543  2007-01-01      00:45:00      2000 W  FULLERTON AVENUE
989TX   2007-01-01      00:49:00      800 W   DIVISION STREET
9870771 2007-01-01      00:51:00      1 E     63RD STREET
6001595 2007-01-01      00:53:00      2000 W  FULLERTON AVENUE
8623536 2007-01-01      00:55:00      11100 S HALSTED STREET
Time taken: 0.515 seconds, Fetched: 10 row(s)
hive> █
```

Vamos ahora a repetir cada consulta que hemos realizado con Pig para así comprobar la expresividad de ambos lenguajes. Entra en la consola de Hive y obtén:

1. (C1. 0,5 puntos) Lista las 5 cámaras que más multas han generado.

```
hive> SELECT posicion, calle, count(*) num_multas
> FROM multas
> GROUP BY posicion, calle
> ORDER BY num_multas DESC
> LIMIT 5;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = root_20201121165904_ebf37e27-1a25-4ccb-9d37-da7e065e1e84
Total jobs = 2
```

```
Total MapReduce CPU Time Spent: 21 seconds 400 msec
OK
400 W    BELMONT AVENUE    94466
4200 S    CICERO AVENUE    78265
1000 W    HOLLYWOOD AVENUE    65977
400 S    WESTERN AVENUE    63825
8900 S    STONY ISLAND    60656
Time taken: 159.137 seconds, Fetched: 5 row(s)
```

2. (C2. 0,5 puntos) Indica el número de cámaras que hay por cada una de las calles

```
hive> SELECT result.calle, COUNT(*) FROM (SELECT DISTINCT posicion,calle FROM multas)
result GROUP BY result.calle;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = root_20201121181332_386d0c3e-2c37-422a-b3c7-7505c289b309
```

Primero sacamos las combinaciones únicas de posición y calle, que nos dará las cámaras que tenemos. Esto lo agrupamos por calle y finalmente seleccionamos la calle y el número de registros para cada uno de ellos, que será el número de cámaras. El resultado lo adjuntaré aparte.

3. (C3. 1 punto) De cada cámara, indica qué día ha registrado más multas y cuántas multas.

```
hive> CREATE TEMPORARY TABLE multas_por_camara_fecha AS
> SELECT posicion, calle, fecha, COUNT(*) multas_dia
> FROM multas
> GROUP BY posicion, calle, fecha;
```

```
hive> SELECT A.posicion, A.calle, A.fecha, A.multas_dia
> FROM multas_por_camara_fecha A
> INNER JOIN (
> SELECT posicion, calle, MAX(multas_dia) multas_dia
> FROM multas_por_camara_fecha
> GROUP BY posicion, calle) B
> ON A.posicion = B.posicion
> AND A.calle = B.calle
> AND A.multas_dia = B.multas_dia;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = root_20201122122830_661ac8f6-1b10-43d6-a37d-68ee14b4db46
```


Creamos una tabla temporal en la que sacamos el número de multas que ha puesto cada cámara cada día y luego hacemos un INNER JOIN para con una tabla en la que seleccionamos el máximo de multas en un día para obtener lo que nos pide la query. El tiempo de ejecución será el de la suma de ambos procesos.

4. (C4. 1 punto) Indica qué matrícula ha sido la más multada, mostrando la fecha/hora y la calle de cada una de sus multas.

```
hive> CREATE TEMPORARY TABLE result AS
> SELECT matricula, COUNT(*) AS num_multas
> FROM multas
> GROUP BY matricula
> ORDER BY num_multas DESC
> LIMIT 1;
```

```
hive> CREATE TABLE query_4 AS
> SELECT A.* FROM multas AS A,
> result AS B
> WHERE A.matricula = B.matricula;
```

```
hive> SELECT * FROM query_4;
OK
SCHLARS 2007-08-31      19:20:00      7900 S  STONEY ISLAND
SCHLARS 2008-02-27      16:22:00      7900 S  COTTAGE GROVE
SCHLARS 2008-04-26      15:35:00      7900 S  STONEY ISLAND
SCHLARS 2008-05-17      01:37:00      8300 S  STONY ISLAND AVENUE
SCHLARS 2008-08-11      17:36:00      7100 S  COTTAGE GROVE
SCHLARS 2008-09-27      08:12:00      6700 S  STONY ISLAND AVENUE
```

Creamos una tabla temporal que contenga la matrícula que más multas tenga (similar a lo que hicimos en la query 1). Luego hacemos un filtrado de la tabla original para obtener una tabla en la que solo esté la matrícula en cuestión y por último la mostramos.

Para obtener los archivos de HDFS al contenedor ejecutamos:

```
>> hdfs dfs -get /hadoop/out
```

Y nos lo mandará a local del contenedor en el que estamos, hive-server. Para traerlo de ahí a nuestra máquina ejecutamos:

```
>> docker cp hive-server:out .
```

Y ya la tendremos en la carpeta Master_Platform que es desde la que partimos inicialmente.

Como detalle de lo que hablamos de si en pig pusieron grunt por lo de llamarse pig. Parece ser que sí. Incluso encontré por ahí que, al parecer, según Ben Reed (investigador de Yahoo que estuvo ayudando en la creación), quería poner “oink” pero la primera implementación era tan limitada que no se merecía ese nombre y se quedó en “grunt”.