

LAB 8

HIVE como base de datos analítica (warehouse)

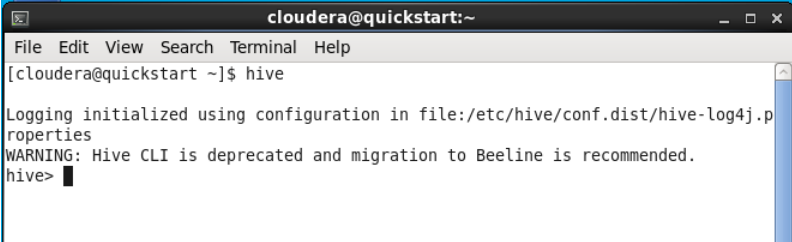
PARTE 1

En este laboratorio vamos a mirar el uso de Apache Hive, como una herramienta warehouse (tipo batch) de análisis de datos.

**** Puede ver la versión de hive con la que está trabajando:**

```
[cloudera@quickstart ~]$ hive --version
Hive 1.1.0-cdh5.13.0
Subversion file:///data/jenkins/workspace/generic-package-rhel64-6-0/topdir/BUILD/hive-1.1.0-cdh5.13.0 -r Unknown
Compiled by jenkins on Wed Oct 4 11:06:55 PDT 2017
From source with checksum 4c9678e964cc1d15a0190a0a1867a837
[cloudera@quickstart ~]$
```

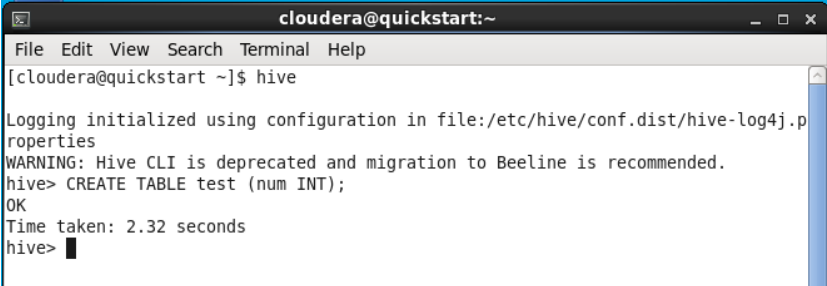
1. Para empezar vamos a crear una tabla y vamos a importar datos desde un archivo que ya está existente en hdfs. Para esto comprobamos que Hive funciona
\$ hive

A terminal window titled 'cloudera@quickstart:~' showing the command 'hive' being executed. The output includes initialization logs and a warning about the deprecated CLI.

```
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive>
```

2. Hacemos una operación de creación de una tabla, en este caso es una tabla vacía que se llama *test* con un solo atributo que es un tipo de número entero.

hive> CREATE TABLE test (num INT);

A terminal window titled 'cloudera@quickstart:~' showing the command 'CREATE TABLE test (num INT);' being executed. The output shows 'OK' and the time taken for the operation.

```
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> CREATE TABLE test (num INT);
OK
Time taken: 2.32 seconds
hive>
```

**** Se puede mirar que la tabla test está en el entorno de hadoop: /user/hive/warehouse**

Browse Directory

/user/hive/warehouse							Go!
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxrwxrwx	cloudera	supergroup	0 B	Tue Mar 09 17:57:35 -0800 2021	0	0 B	test

3. Hacemos un *select*, la tabla está vacía, no queremos que nos devuelva nada, pero sí queremos comprobar que se ha acabado de crear de forma correcta.

```
SELECT * FROM test;
```

```
hive> SELECT * FROM test;
OK
Time taken: 0.578 seconds
hive> █
```

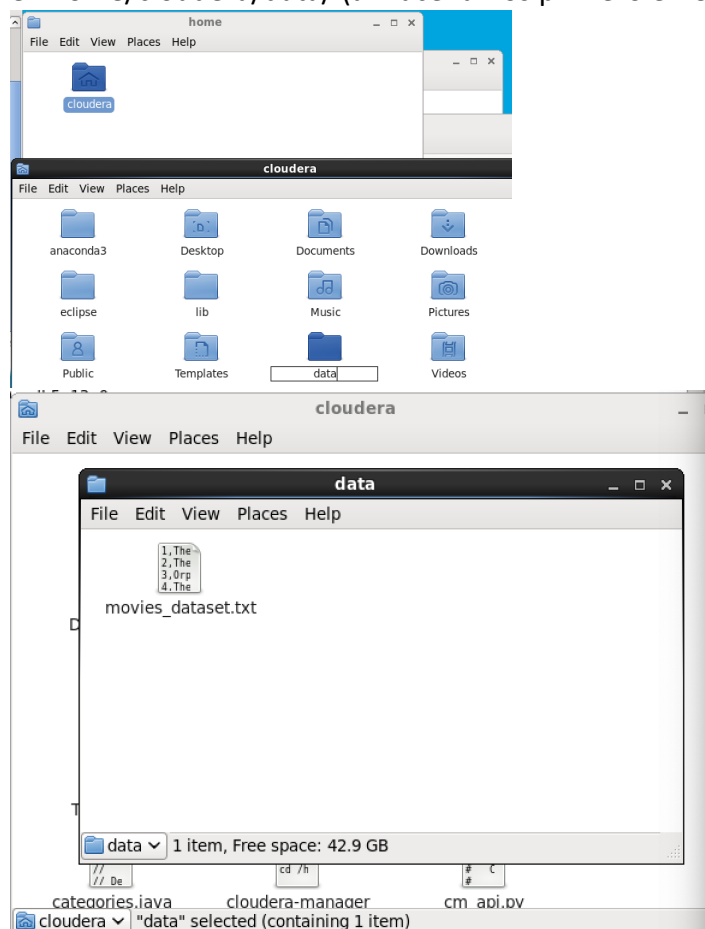
4. Luego vamos a eliminar la tabla.

```
hive> DROP TABLE test;
OK
Time taken: 0.542 seconds
hive> █
```

NOTA: Esas tres operaciones básicas se ejecutan nos permiten comprobar que Hive está funcionando bien.

```
hive> exit
> ;
WARN: The method class org.apache.commons.logging.impl.SLF4JLogFactory#release()
was invoked.
WARN: Please see http://www.slf4j.org/codes.html#release for an explanation.
[cloudera@quickstart ~]$ █
```

4. Guarde el archivo `movies_dataset.txt` que se le ha dado como parte del laboratorio en `home/cloudera/data/` (almacenamos primero en Centos).



5. Vamos a utilizar el archivo de texto, `movies_dataset.txt` donde tenemos información sobre la identificación de una película o serie, su nombre, el año de lanzamiento, el promedio de calificación o valoración (rating) y el número de ratings totales. El archivo tiene 49590 registros.

Abre una nueva terminal

Lo que hacemos es copiar este archivo de datos de ejemplo y lo llevamos al depósito de datos HDFS utilizando el comando `put`.

NOTA: Crea el directorio `hiveFiles` y luego comprueba que efectivamente este archivo `movies_dataset.txt` se ha copiado bien dentro de nuestro sistema de datos distribuido `hdfs`.

```
$ hdfs dfs -mkdir -p /user/hadoop/hiveFiles
```

```
$ hdfs dfs -put data/movies_dataset.txt /user/hadoop/hiveFiles
```

```
[cloudera@quickstart ~]$ hdfs dfs -put data/movies_dataset.txt /user/hadoop/hiveFiles
```

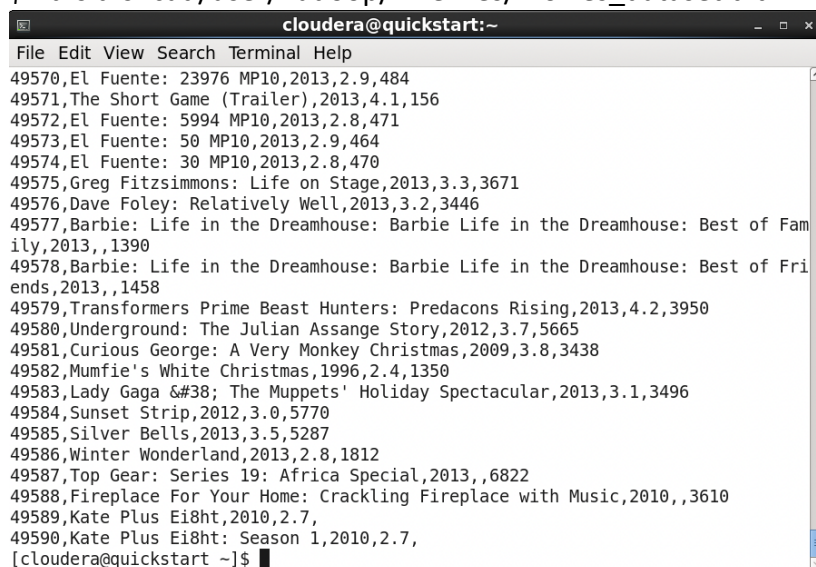
```
[cloudera@quickstart ~]$ hdfs dfs -ls /user/hadoop/hiveFiles
```

```
Found 1 items
```

```
-rw-r--r-- 1 cloudera supergroup 2942575 2021-06-29 08:18 /user/hadoop/hiveFiles/movies_dataset.txt
```

7. Vamos a mirar el contenido del fichero: id de la película, nombre de la película, año de estreno, rating, y número de ratings (campos separados por coma).

```
$ hdfs dfs -cat /user/hadoop/hiveFiles/movies_dataset.txt
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
49570,El Fuente: 23976 MP10,2013,2.9,484  
49571,The Short Game (Trailer),2013,4.1,156  
49572,El Fuente: 5994 MP10,2013,2.8,471  
49573,El Fuente: 50 MP10,2013,2.9,464  
49574,El Fuente: 30 MP10,2013,2.8,470  
49575,Greg Fitzsimmons: Life on Stage,2013,3.3,3671  
49576,Dave Foley: Relatively Well,2013,3.2,3446  
49577,Barbie: Life in the Dreamhouse: Barbie Life in the Dreamhouse: Best of Family,2013,,1390  
49578,Barbie: Life in the Dreamhouse: Barbie Life in the Dreamhouse: Best of Friends,2013,,1458  
49579,Transformers Prime Beast Hunters: Predacons Rising,2013,4.2,3950  
49580,Underground: The Julian Assange Story,2012,3.7,5665  
49581,Curious George: A Very Monkey Christmas,2009,3.8,3438  
49582,Mumfie's White Christmas,1996,2.4,1350  
49583,Lady Gaga &#38; The Muppets' Holiday Spectacular,2013,3.1,3496  
49584,Sunset Strip,2012,3.0,5770  
49585,Silver Bells,2013,3.5,5287  
49586,Winter Wonderland,2013,2.8,1812  
49587,Top Gear: Series 19: Africa Special,2013,,6822  
49588,Fireplace For Your Home: Crackling Fireplace with Music,2010,,3610  
49589,Kate Plus Ei8ht,2010,2.7,  
49590,Kate Plus Ei8ht: Season 1,2010,2.7,  
[cloudera@quickstart ~]$
```

****** Vamos a **regresar a la terminal de Hive** y vamos a hacer el ciclo de creación, consulta y eliminación de tablas dentro de Hive.

```
[cloudera@quickstart ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.p
roperties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive>
```

8. Lo primero que vamos a crear es una tabla externa donde tendremos cinco atributos: un identificador que sea un entero, un nombre que sea una cadena, el año que será entero, la valoración que es float y el número total de ratings de la película, que será un entero.

La tabla del warehouse se llamará *movies* y la vamos a diseñar de forma que esté esperando datos separados por comas, como hemos visto que está compuesto nuestro archivo de texto.

Podemos poner un comentario para describir qué contiene la tabla.

Indicamos que *el archivo tiene las tuplas en cada fila* y también que el separador de valores es una coma. E indicamos que la fuente es un archivo de texto.

La tabla buscará la data en la ruta */user/hadoop/hiveFiles*.

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS movies (id INT, nombre STRING, anio INT,
rating FLOAT, numRatings INT)
COMMENT 'data de peliculas'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/hadoop/hiveFiles'
;
```

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS movies (id INT, nombre STRING, anio IN
T, rating FLOAT, numRatings INT)
> COMMENT 'data de peliculas'
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY ','
> STORED AS TEXTFILE
> LOCATION '/user/hadoop/hiveFiles'
> ;
OK
Time taken: 0.152 seconds
hive>
```

Ahora ya está creada la estructura, Hive ha ido a la carpeta */user/hadoop/hiveFiles* en búsqueda del archivo *movies_dataset.txt* para copiar la data en una tabla del warehouse que hemos llamado *movies*.

9. Ahora lo que hacemos es comprobar que realmente se han insertado para ver todos los datos:

```
hive> SELECT * FROM movies;
```

```

cloudera@quickstart:~
File Edit View Search Terminal Help
49574 El Fuente: 30 MP10 2013 2.8 470
49575 Greg Fitzsimmons: Life on Stage 2013 3.3 3671
49576 Dave Foley: Relatively Well 2013 3.2 3446
49577 Barbie: Life in the Dreamhouse: Barbie Life in the Dreamhouse: Best of F
amily 2013 NULL 1390
49578 Barbie: Life in the Dreamhouse: Barbie Life in the Dreamhouse: Best of F
riends 2013 NULL 1458
49579 Transformers Prime Beast Hunters: Predacons Rising 2013 4.2 3
950
49580 Underground: The Julian Assange Story 2012 3.7 5665
49581 Curious George: A Very Monkey Christmas 2009 3.8 3438
49582 Mumfie's White Christmas 1996 2.4 1350
49583 Lady Gaga &#38; The Muppets' Holiday Spectacular 2013 3.1 3
496
49584 Sunset Strip 2012 3.0 5770
49585 Silver Bells 2013 3.5 5287
49586 Winter Wonderland 2013 2.8 1812
49587 Top Gear: Series 19: Africa Special 2013 NULL 6822
49588 Fireplace For Your Home: Crackling Fireplace with Music 2010 NULL 3
610
49589 Kate Plus Ei8ht 2010 2.7 NULL
49590 Kate Plus Ei8ht: Season 1 2010 2.7 NULL
Time taken: 0.049 seconds, Fetched: 49590 row(s)
hive>

```

A partir de este momento los datos tienen una estructura semejante a una tabla, podemos consultarlos como si tuviéramos tabla o vista de datos relacional y, por ejemplo, podríamos hacer consultas sencillas, como extraer todas las películas cuyo rating sea mayor o igual que 4.5.

```

hive> SELECT nombre FROM movies;
hive> SELECT * FROM movies WHERE rating>=4.5;

```

```

hive> SELECT * FROM movies WHERE rating>=4.5;
OK
6997 Breaking Bad: Season 1 2008 4.5 NULL
8041 Breaking Bad: Season 2 2009 4.5 NULL
12079 Breaking Bad: Season 3 2010 4.5 NULL
13315 Breaking Bad 2008 4.5 NULL
14721 The Walking Dead: Season 1 2010 4.5 NULL
14955 Breaking Bad: Season 4 2011 4.5 NULL
14957 Sherlock: Series 1 2010 4.5 NULL
19400 The Walking Dead: Season 2 2011 4.5 NULL
21911 The Walking Dead 2010 4.5 NULL
26484 Sherlock 2010 4.5 NULL
30426 The Avengers 2012 4.5 8575
33523 Sherlock: Series 2 2012 4.5 NULL
36259 Breaking Bad: Season 5 2012 4.5 NULL
37138 Orange Is the New Black: Season 1 2013 4.5 NULL
37141 Orange Is the New Black 2013 4.5 NULL
37897 The Walking Dead: Season 3 2012 4.5 NULL
41846 Fairy Tail: Season 1 2009 4.5 NULL
43071 Fairy Tail 2009 4.5 NULL
43444 Blackfish 2013 4.5 4985
44098 Arrested Development (Trailer) 2013 4.5 97
46388 The Fosters 2013 4.5 NULL
46401 The Fosters: Season 1 2013 4.5 NULL
49504 Lilyhammer: Season 2 (Trailer) 2013 4.5 106
Time taken: 0.048 seconds, Fetched: 23 row(s)
hive>

```


Entonces, para acceder a los datos que están prácticamente en un archivo de texto hemos usado un lenguaje muy parecido a SQL.

```
hive> SELECT nombre, anio FROM movies WHERE nombre LIKE '%CSI%' AND anio>2010;
```

```
cloudera@quickstart:~
File Edit View Search Terminal Help
hive> SELECT nombre, anio FROM movies WHERE nombre LIKE '%CSI%' AND anio>2010;
OK
CSI: NY: Season 8      2011
CSI: Miami: Season 10 2011
CSI: NY: Season 9      2012
CSI: Miami: Season 10: Countermeasures 2011
CSI: Miami: Season 10: By The Book 2011
CSI: Miami: Season 10: Sinner Takes All 2011
CSI: Miami: Season 10: Stiff 2011
CSI: Miami: Season 10: Blown Away 2011
CSI: Miami: Season 10: Look Who's Taunting 2011
CSI: Miami: Season 10: Killer Regrets 2011
CSI: Miami: Season 10: Dead Ringer 2011
CSI: Miami: Season 10: A Few Dead Men 2011
CSI: Miami: Season 10: Terminal Velocity 2011
CSI: Miami: Season 10: Friendly Fire 2011
CSI: Miami: Season 10: Crowned 2011
CSI: Miami: Season 10: Long Gone 2011
CSI: Miami: Season 10: At Risk 2011
CSI: Miami: Season 10: Rest in Pieces 2011
CSI: Miami: Season 10: No Good Deed 2011
CSI: Miami: Season 10: Last Straw 2011
CSI: Miami: Season 10: Habeas Corpse 2011
CSI: Miami: Season 10: Law & Disorder 2011
CSI: NY: Season 8: Who's There? 2011
CSI: NY: Season 8: Brooklyn Til I Die 2011
CSI: NY: Season 8: Means To An End 2011
CSI: NY: Season 8: Clean Sweep 2011
CSI: NY: Season 8: Crushed 2011
CSI: NY: Season 8: Crossroads 2011
CSI: NY: Season 8: Air Apparent 2011
CSI: NY: Season 8: Get Me Out Of Here! 2011
CSI: NY: Season 8: Cavallino Rampante 2011
CSI: NY: Season 8: Officer Involved 2011
CSI: NY: Season 8: Indelible 2011
CSI: NY: Season 8: Keep It Real 2011
```

10. Podemos hacer consultas un poco más complejas, por ejemplo, seleccionar el nombre, número de ratings y el identificador de las películas con más de 7500 ratings y ordenarlas por orden decreciente de numRatings.

```
hive> SELECT id, nombre, numRatings FROM movies WHERE numRatings>7500 ORDER BY numRatings DESC;
```

```
cloudera@quickstart:~
Query ID = cloudera_20210629084848_9779d12b-ee53-47a8-9d64-e1666ab518b2
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1624977209257_0004, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1624977209257_0004/
Kill Command = /usr/lib/hadoop/bin/hadoop job -kill job_1624977209257_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-06-29 08:48:59,305 Stage-1 map = 0%, reduce = 0%
2021-06-29 08:49:05,550 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.82 sec
2021-06-29 08:49:11,778 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 3.15 sec
```

**** Nota los procesos map reduce al ejecutar esta consulta**

```

cloudera@quickstart:~
File Edit View Search Terminal Help
38745 KT 7600
2042 As It Is in Heaven 7598
38767 The Tulse Luper Suitcases: The Moab Story 7596
19234 Here 7593
2437 Daddy Long Legs 7593
1406 The Passion of the Christ 7589
3793 Jhoom Barabar Jhoom 7587
31000 Do Rahain 7586
13993 King of Paper Chasin' 7585
26279 Hugo 7582
14327 Aisha 7581
15955 Lafangey Parindey 7578
5433 Beaufort 7577
8620 A Heavenly Vintage 7575
481 The Lovers on the Bridge 7573
12357 Guzaarish 7572
7269 A Woman in Berlin 7571
1552 The Motorcycle Diaries 7562
1921 The Sea Inside 7559
30999 Chak Jawaana 7559
40936 A Werewolf Boy 7552
1135 American Me 7552
86 The Hindenburg 7546
9159 Confucius 7544
8506 Ajami 7541
827 The Return of a Man Called Horse 7540
1518 Flesh + Blood 7531
27775 Without a Father 7528
930 The Piano Teacher 7528
36003 WWE's Greatest Rivalries: Shawn Michaels vs Bret Hart 7525
24759 House of Pleasures 7523
430 The Cider House Rules 7523
1227 The Private Life of Sherlock Holmes 7519
37428 The Host 7514
630 Caravans 7512
9420 13 Assassins 7509
39346 Kai po che! 7506
6566 The Chaser 7503
12011 Ken Burns: The War: A World Without War 7501
Time taken: 18.667 seconds, Fetched: 492 row(s)
hive>

```

SELECT count(nombre) FROM movies WHERE nombre LIKE "%Woman%";
 -> total de 142 películas

```

cloudera@quickstart:~
File Edit View Search Terminal Help
2021-06-29 09:19:27,673 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 2.72
sec
MapReduce Total cumulative CPU time: 2 seconds 720 msec
Ended Job = job_1624977209257_0007
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 2.72 sec HDFS Read: 2949899
HDFS Write: 2 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 720 msec
OK
0
Time taken: 18.125 seconds, Fetched: 1 row(s)
hive> SELECT count(nombre) FROM movies WHERE nombre LIKE "%Woman%";
Query ID = cloudera_20210629092020_8573e4d4-a3ee-4279-9b77-e94995a7def7
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1624977209257_0008, Tracking URL = http://quickstart.cloudera
:8088/proxy/application_1624977209257_0008/
Kill Command = /usr/lib/hadoop/bin/hadoop job -kill job_1624977209257_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-06-29 09:20:36,254 Stage-1 map = 0%, reduce = 0%
2021-06-29 09:20:41,407 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.77 se
c
2021-06-29 09:20:47,607 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 2.82
sec
MapReduce Total cumulative CPU time: 2 seconds 820 msec
Ended Job = job_1624977209257_0008
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 2.82 sec HDFS Read: 2949904
HDFS Write: 4 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 820 msec
OK
142
Time taken: 18.323 seconds, Fetched: 1 row(s)
hive>

```

Como estas consultas son un poco más complejas, Hive va a necesitar lanzar varios trabajos de tipo MapReduce, a medida que se van haciendo más complejas las queries, la cantidad de trabajo realizada por Hive será más complejo y tardará algo más de tiempo. Sin embargo, en este caso los jobs se lanzan de forma automática (no nos dedicamos a programar) y de lo que nos preocupamos es de que los datos ayuden a resolver problemas de consulta.

PARTE 2: TRABAJO GRUPAL

1. **Usa la tabla movies** para generar las siguientes consultas que nos permitan explorar el comportamiento de la data, por ejemplo, ¿hay películas con muy pocos ratings? ¿Cuáles son las 5 películas más antiguas en los registros? ¿Hay películas con valoración menor a 1.5? ¿Cuáles son las 5 peores películas?

****** Cuando termines, como dejaremos de usar la tabla movies a partir de aquí, puedes eliminarla usando `hive> DROP TABLE movies;`

2. Vamos a descargar un archivo de datos que representará nuestro conjunto de datos de análisis. Antes de empezar el ejercicio debes entrar en la máquina virtual y descargar el archivo de datos en una carpeta local.

Primero, debes descargar un archivo de datos "**puntuaciones.csv**" para resolver una serie de consultas. Para ello:

- En la máquina virtual MV_Cloudera abre el navegador web, y,
- Descarga los datos desde esta dirección: <https://bit.ly/2TVtzUF>
- Descomprime y cambia el nombre del archivo a "puntuaciones.csv"

Una vez que el archivo de datos está descargado:

1. Comprueba el contenido del archivo de datos "puntuaciones.csv" del repositorio de datos y cómo los campos están separados
2. Copia el archivo descargado a una carpeta **vacía** en HDFS
3. Crea una tabla en el warehouse de Hive para importar los datos del archivo recientemente copiado a HDFS. Los campos esperados serán los siguientes: Ciudad, número de comentarios, nombre del establecimiento, tipo de establecimiento. Estos campos son del tipo: STRING, INT, STRING y STRING

Vía *queries* SQL a la tabla, contesta a las siguientes preguntas:

1. ¿Cuántos **restaurantes** de la tabla están en la ciudad de Madrid? (note que no todas las observaciones corresponden a *restaurantes*)
2. ¿Qué establecimiento tiene el mayor número de puntuaciones?
3. ¿Cuántos **tipos** de establecimientos hay en el conjunto de datos?