

PR_06.3 Dani Gayol Rodríguez

PR_06.3 Dani Gayol Rodríguez.....	1
Apartado A.....	1
1.) Crea las tablas HIVE adecuadas para cargar los datos de cada uno de los archivos. Para mayor eficiencia, crearemos ambas tablas con 5 buckets sobre el id de la película. Tener especial cuidado en el campo que almacene el género de las películas.	2
2.) Mostrar las consultas de creación y carga de las tablas con una captura que muestre los datos cargados.....	3
3.) Mostrar una captura de las tablas en el warehouse de HIVE donde se vean los buckets.....	4
Apartado B.....	6
1.) Mediante una consulta en HIVE, encontrar las cinco películas (código, título y media de votos) mejor valoradas, que hayan sido votadas al menos por 10 usuarios.	7
Apartado C	7
1.) Encontrar las cinco películas más antiguas con una valoración media por encima de 4 puntos.	8
Apartado D	8
1.) Muestra los cinco años en que se editaron más películas indicando el número de ellas para cada año.....	9
Apartado E.....	9
1.) Muestra los diez géneros de películas más frecuentes.	10
2.) Partiendo de la consulta del Apartado B de arriba. ¿Qué géneros son los más frecuentes en dichas películas?.....	10

Apartado A

Antes que nada, nos descargamos los archivos “movies.dat” y “ratings.dat” y los pegamos en el HDFS, primero usamos “WinSCP” para pasarlo a la máquina virtual

/home/maria_dev/				
Nombre	Tamaño	Modificado	Permisos	Propiet...
..		18/06/2018 17:30:38	rwxr-xr-x	root
movies.dat	160 KB	04/12/2023 18:52:20	rw-rw-r--	maria_...
pig_1763396932271.log	2 KB	17/11/2025 17:28:53	rw-rw-r--	maria_...
pig_1763397103224.log	2 KB	17/11/2025 17:31:44	rw-rw-r--	maria_...
quijote.txt	2.174 KB	01/11/2025 16:28:40	rw-rw-r--	maria_...
ratings.dat	21.088 KB	04/12/2023 18:47:10	rw-rw-r--	maria_...
script_quijote.pig	1 KB	17/11/2025 17:31:20	rw-rw-r--	maria_...
u.data	1.933 KB	20/09/2019 4:54:38	rw-rw-r--	maria_...
u.item	231 KB	20/09/2019 4:54:38	rw-rw-r--	maria_...
u.user	23 KB	20/09/2019 4:54:38	rw-rw-r--	maria_...

Una vez en la máquina virtual, pasamos los archivos al HDFS

```
[maria_dev@sandbox-hdp ~]$ hdfs dfs -put /home/maria_dev/movies.dat /user/maria_dev/movielens
[maria_dev@sandbox-hdp ~]$ hdfs dfs -put /home/maria_dev/ratings.dat /user/maria_dev/movielens
[maria_dev@sandbox-hdp ~]$ hdfs dfs -ls /user/maria_dev/movielens
Found 2 items
-rw-r--r--  1 maria_dev hdfs      163666 2025-12-10 08:34 /user/maria_dev/movielens/movies.dat
-rw-r--r--  1 maria_dev hdfs    21593504 2025-12-10 08:34 /user/maria_dev/movielens/ratings.dat
```

1.) Crea las tablas HIVE adecuadas para cargar los datos de cada uno de los archivos. Para mayor eficiencia, crearemos ambas tablas con 5 buckets sobre el id de la película. Tener especial cuidado en el campo que almacene el género de las películas.

Primero creamos el database

```
1 CREATE DATABASE movies;
```

Ahora creamos las tablas

```
1 CREATE TABLE movies (
2     id INT,
3     pelicula STRING,
4     generos STRING
5 )
6 ROW FORMAT DELIMITED
7 FIELDS TERMINATED BY '*'
```

```
1 CREATE TABLE ratings (
2     id INT,
3     usuario INT,
4     voto FLOAT,
5     time_stamp BIGINT
6 )
7 ROW FORMAT DELIMITED
8 FIELDS TERMINATED BY '|'
```

Ahora vamos a cargar los datos en estas tablas

```
1 LOAD DATA INPATH '/user/maria_dev/movielens/movies.dat' INTO TABLE movies;
```

```
1 LOAD DATA INPATH '/user/maria_dev/movielens/ratings.dat' INTO TABLE ratings;
```

2.) Mostrar las consultas de creación y carga de las tablas con una captura que muestre los datos cargados.

Ahora vamos a ver que se cargaron los datos correctamente

```
1 SELECT * FROM ratings LIMIT 5;
```

```
ratings.id  ratings.usuario  ratings.voto  ratings.time_stamp
```

1	1193	5.0	978300760
1	661	3.0	978302109
1	914	3.0	978301968
1	3408	4.0	978300275
1	2355	5.0	978824291

```
1 | SELECT * FROM movies LIMIT 5;
```

```
movies.id  movies.pelicula  movies.generos
```

1	Toy Story (1995)	Animation Children's Comedy
2	Jumanji (1995)	Adventure Children's Fantasy
3	Grumpier Old Men (1995)	Comedy Romance
4	Waiting to Exhale (1995)	Comedy Drama
5	Father of the Bride Part II (1995)	Comedy

3.) Mostrar una captura de las tablas en el warehouse de HIVE donde se vean los buckets.

Como al intentar crear las tablas con “buckets” me daba error, voy a tener que crear dos nuevas y luego pasarle los datos ahí

```
1 | CREATE TABLE movies_bucket (
2 |   id INT,
3 |   pelicula STRING,
4 |   generos STRING
5 |
6 | CLUSTERED BY (id) INTO 5 BUCKETS
7 | STORED AS TEXTFILE
8 | TBLPROPERTIES ("bucketing_version"="2");
```

```
1 CREATE TABLE ratings_bucket (
2     id INT,
3     usuario INT,
4     voto FLOAT,
5     time_stamp BIGINT
6 )
7 CLUSTERED BY (id) INTO 5 BUCKETS
8 STORED AS TEXTFILE
9 TBLPROPERTIES ("bucketing_version"="2");
```

Ahora le vamos a pasar los datos

```
1 INSERT INTO TABLE movies_bucket
2 SELECT id, pelicula, generos
3 FROM movies;
```

```
1 INSERT INTO TABLE ratings_bucket
2 SELECT id, usuario, voto, time_stamp
3 FROM ratings;
```

Finalmente, en el “files view” de Ambari nos vamos a esta ruta para comprobarlo

The screenshot shows the Ambari 'files view' interface. At the top, there is a breadcrumb navigation bar: a folder icon followed by '> warehouse > movies.db'. Below this, there is a sidebar with three items: 'hive', 'apps', and a folder icon with a '/' symbol. The main content area displays a file tree. At the top of the tree is a folder icon followed by '.hive-staging_hive_2025-12-10_09-05-5...'. Below it are two files: '000000_0' and another folder icon followed by '.hive-staging_hive_2025-12-10_09-06-2...'. This structure represents the bucketed data in the Hive metastore.

Apartado B

1.) Mediante una consulta en HIVE, encontrar las cinco películas (código, título y media de votos) mejor valoradas, que hayan sido votadas al menos por 10 usuarios.

```
1 SELECT m.id,
2       m.pelicula,
3       AVG(r.voto) AS avg_rating,
4       COUNT(r.id) AS num_votes
5 FROM ratings r
6 JOIN movies m ON r.usuario = m.id
7 GROUP BY m.id, m.pelicula
8 HAVING COUNT(r.id) >= 10
9 ORDER BY avg_rating DESC
10 LIMIT 5;
```

m.id	m.pelicula	avg_rating	num_votes
2905	Sanjuro (1962)	4.608695652173913	69
2019	Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)	4.560509554140127	628
318	Shawshank Redemption, The (1994)	4.554557700942973	2227
858	Godfather, The (1972)	4.524966261808367	2223
745	Close Shave, A (1995)	4.52054794520548	657

Apartado C

1.) Encontrar las cinco películas más antiguas con una valoración media por encima de 4 puntos.

```
1 SELECT m.id,
2      m.pelicula,
3      REGEXP_EXTRACT(m.pelicula, '.*\\((\\d{4})\\))', 1) AS year,
4      AVG(r.voto) AS avg_rating
5 FROM ratings r
6 JOIN movies m ON r.usuario = m.id
7 GROUP BY m.id, m.pelicula
8 HAVING avg_rating > 4
9 ORDER BY year ASC
10 LIMIT 5;
```

m.id	m.pelicula	year	avg_rating
1294	M	""	4.124658780709736
3629	Gold Rush, The (1925)	1925	4.189090909090909
3517	Bells, The (1926)	1926	4.5
2010	Metropolis (1926)	1926	4.082474226804123
3022	General, The (1927)	1927	4.368932038834951

Apartado D

1.) Muestra los cinco años en que se editaron más películas indicando el número de ellas para cada año.

```
1 SELECT
2     REGEXP_EXTRACT(pelicula, '.*\\((\\d{4})\\)'). AS year,
3     COUNT(*) AS num_movies
4 FROM movies
5 GROUP BY REGEXP_EXTRACT(pelicula, '.*\\((\\d{4})\\)'). AS year
6 ORDER BY num_movies DESC
7 LIMIT 5;
```

year	num_movies
1996	345
1995	342
1998	337
1997	314
1999	283

Apartado E

1.) Muestra los diez géneros de películas más frecuentes.

```
1 SELECT generos, COUNT(*) AS count_genero
2 FROM (
3     SELECT EXPLODE(SPLIT(generos, '\|')) AS generos
4     FROM movies
5 ) t
6 GROUP BY generos
7 ORDER BY count_genero DESC
8 LIMIT 10;
```

generos	count_genero
Drama	1603
Comedy	1198
Action	503
Thriller	492
Romance	471
Horror	343
Adventure	283
Sci-Fi	276
Children's	251
Crime	211

2.) Partiendo de la consulta del Apartado B de arriba. ¿Qué géneros son los más frecuentes en dichas películas?

Partiendo de la consulta anterior, los géneros más frecuentes, por ejemplo, el top 3 son los siguientes;

```
1 SELECT generos, COUNT(*) AS top_3
2 FROM (
3     SELECT EXPLODE(SPLIT(generos, '\|\')) AS generos
4     FROM movies
5 ) t
6 GROUP BY generos
7 ORDER BY top_3 DESC
8 LIMIT 3;
```

generos top_3

Drama	1603
-------	------

Comedy	1198
--------	------

Action	503
--------	-----