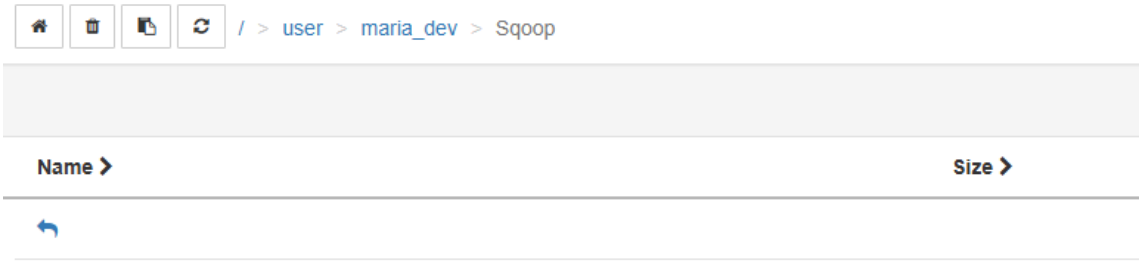
Volcados con SQOOP

1) APARTADO A

1) En la carpeta del usuario maria_dev en HDFS crea una subcarpeta llamada Sqoop donde guardaremos los archivos de esta práctica.



2) Importa con Sqoop las tres tablas que creamos en MySQL en la práctica anterior.

Tabla “emp”

```
[maria_dev@sandbox-hdp ~]$ sqoop import --connect "jdbc:mysql://44.212.173.124:3306/empleados?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --table emp --target-dir /u
ser/maria_dev/Sqoop/emp --num-mappers 1 --split-by id
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/26 08:48:58 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/26 08:48:58 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/26 08:48:58 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/26 08:48:58 INFO tool.CodeGenTool: Beginning code generation
25/11/26 08:49:00 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `emp` AS t LIMIT 1
25/11/26 08:49:01 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `emp` AS t LIMIT 1
```

Tabla “dept”

```
[maria_dev@sandbox-hdp ~]$ sudo sqoop import --connect "jdbc:mysql://44.212.173.124:3306/empleados?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --table dept --target-
dir /user/maria_dev/Sqoop/dept --num-mappers 1 --split-by id
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/26 10:08:12 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/26 10:08:12 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/26 10:08:12 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/26 10:08:12 INFO tool.CodeGenTool: Beginning code generation
25/11/26 10:08:14 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `dept` AS t LIMIT 1
25/11/26 10:08:14 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `dept` AS t LIMIT 1
```

Tabla “salgrade”

```
[maria_dev@sandbox-hdp ~]$ sudo sqoop import --connect "jdbc:mysql://44.212.173.124:3306/empleados?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --table salgrade --tar
get-dir /user/maria_dev/Sqoop/salgrade --num-mappers 1 --split-by id
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/26 10:10:08 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/26 10:10:08 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/26 10:10:08 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/26 10:10:08 INFO tool.CodeGenTool: Beginning code generation
25/11/26 10:10:10 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `salgrade` AS t LIMIT 1
25/11/26 10:10:11 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `salgrade` AS t LIMIT 1
```

3) Importa, dejándola en un solo archivo, todos los datos de los empleados adjuntándoles a cada uno toda la información de su departamento.

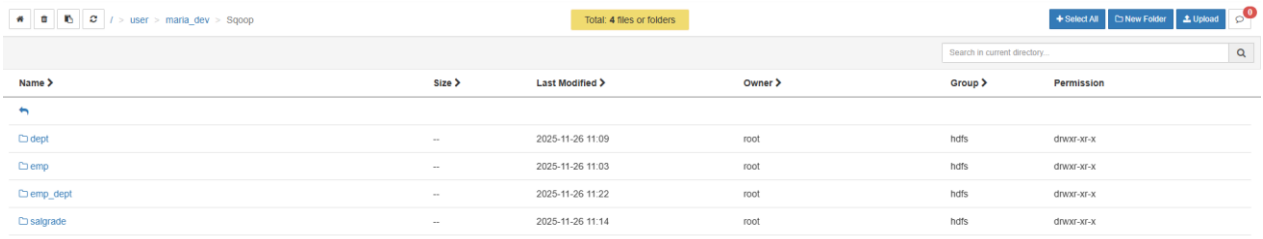
Utilizamos la siguiente query:

```
SELECT emp.*, dept.dname, dept.loc FROM emp
LEFT JOIN dept ON emp.deptno = dept.deptno
       WHERE $CONDITIONS
```

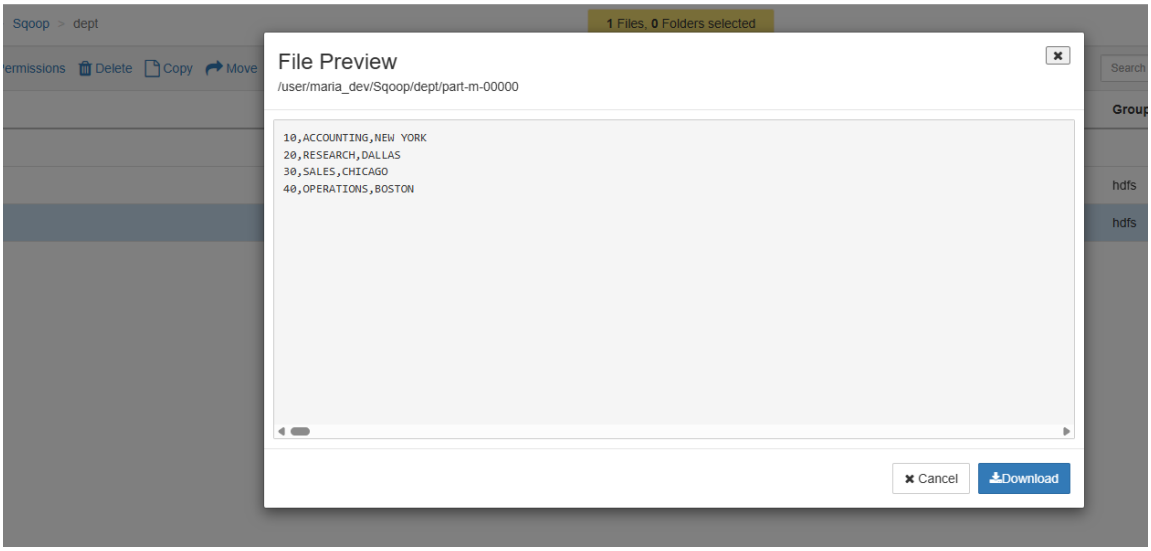
```
[maria_dev@sandbox-hdp ~]$ sudo sqoop import --connect "jdbc:mysql://44.212.173.124:3306/empleados?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --query 'SELECT emp.*,
 dept.dname, dept.loc FROM emp LEFT JOIN dept ON emp.deptno = dept.deptno WHERE $CONDITIONS' --target-dir /user/maria_dev/Sqoop/emp_dept --num-mappers 1 --split-by id
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/26 10:21:28 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/26 10:21:28 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/26 10:21:28 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/26 10:21:28 INFO tool.CodeGenTool: Beginning code generation
25/11/26 10:21:30 INFO manager.SqlManager: Executing SQL statement: SELECT emp.*, dept.dname, dept.loc FROM emp LEFT JOIN dept ON emp.deptno = dept.deptno WHERE  (1 = 0)
25/11/26 10:21:30 INFO manager.SqlManager: Executing SQL statement: SELECT emp.*, dept.dname, dept.loc FROM emp LEFT JOIN dept ON emp.deptno = dept.deptno WHERE  (1 = 0)
25/11/26 10:21:31 INFO manager.SqlManager: Executing SQL statement: SELECT emp.*, dept.dname, dept.loc FROM emp LEFT JOIN dept ON emp.deptno = dept.deptno WHERE  (1 = 0)
```

4) Muestra en HDFS la ubicación y contenido de los ficheros resultantes.

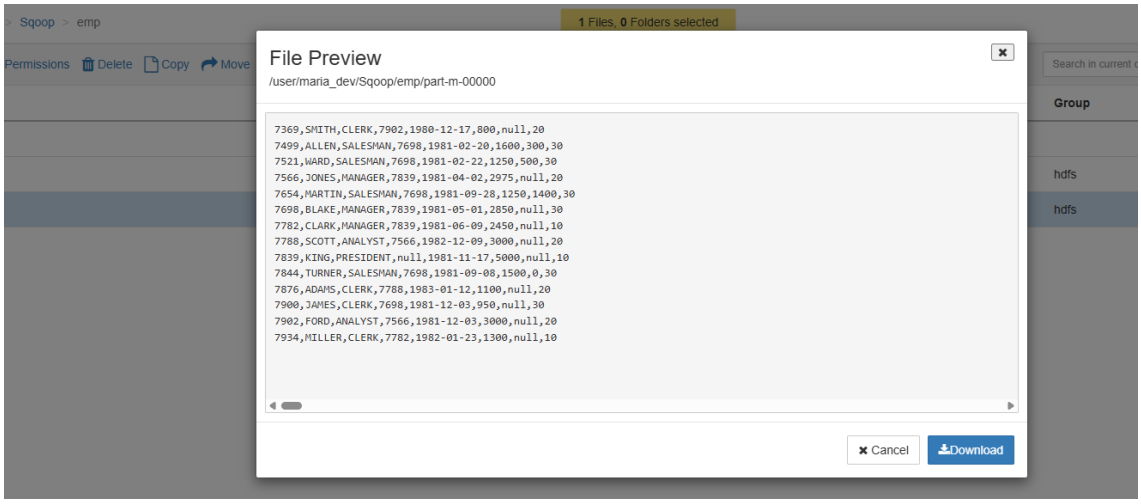
Ubicación



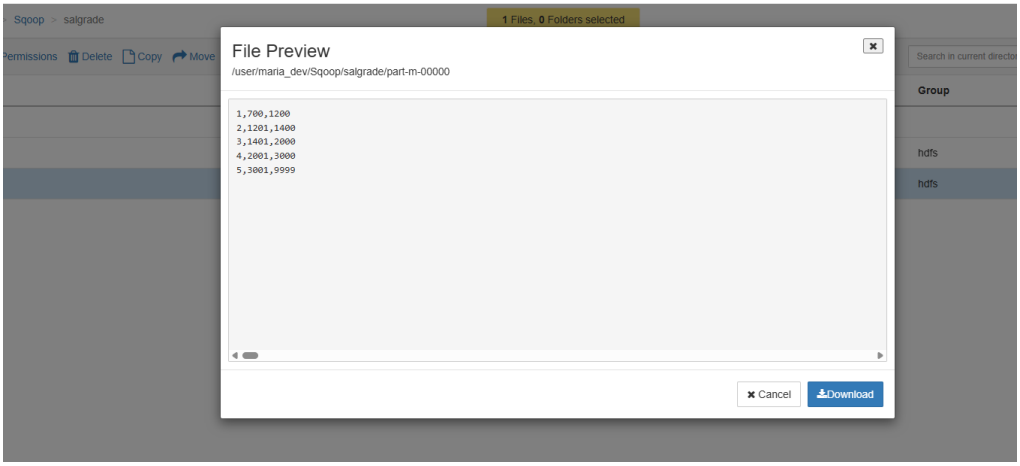
Resultado tabla “dept”.



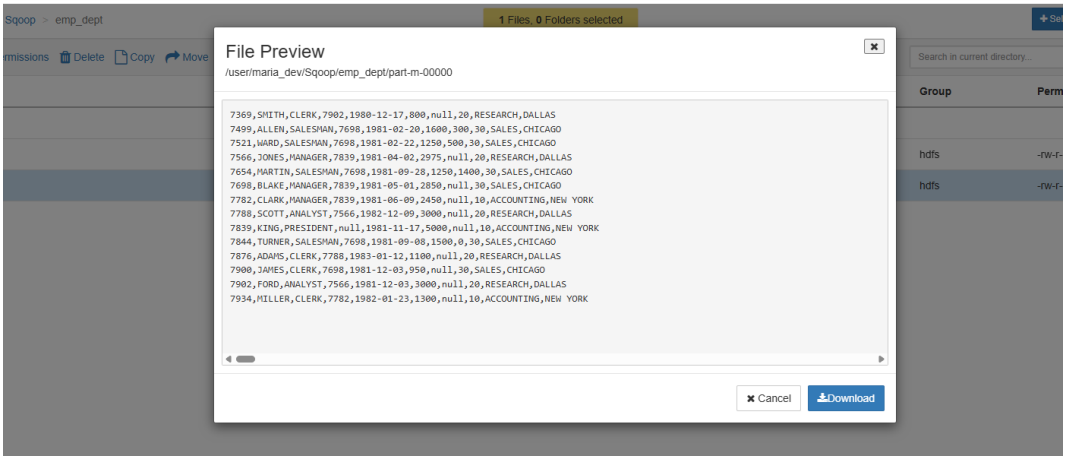
Resultado tabla “emp”.



Resultado tabla “salgrade”.

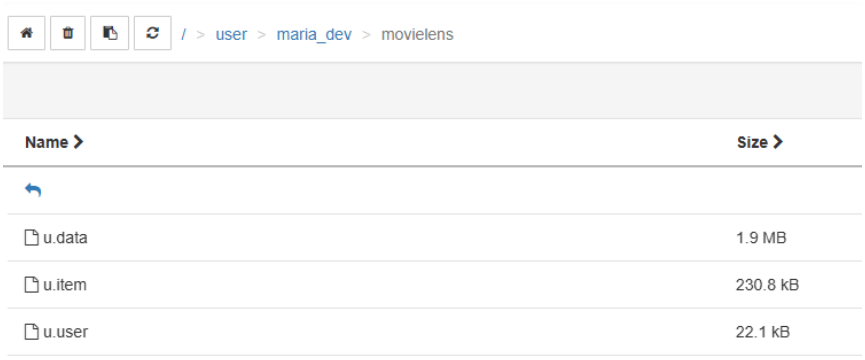


Resultado tabla “emp” unida a la tabla “dept”.



2) APARTADO B

1) En la carpeta del usuario maria_dev en HDFS crea una subcarpeta llamada movielens donde guardaremos los archivos u.data, u.user y u.item.



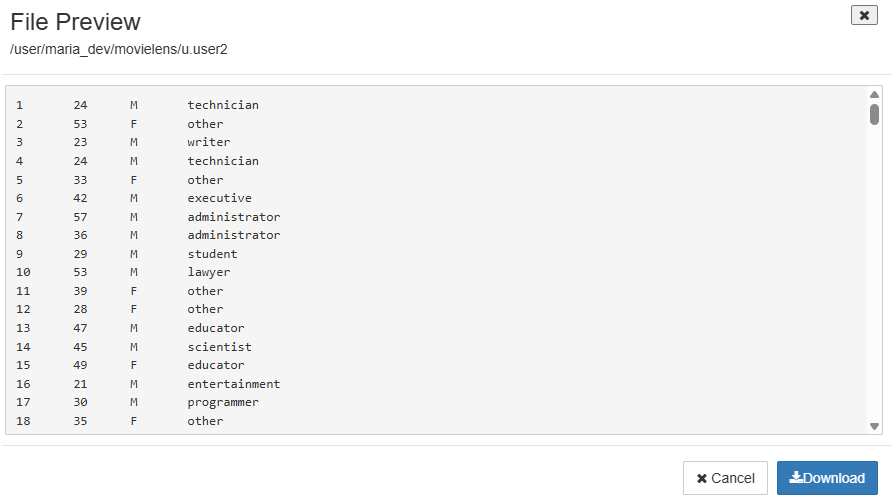
2) Utilizando PIG, al archivo u.user quítale la última columna con el código postal. Guarda el resultado en el archivo u.user2.

Utilizaremos este script:

```
users = LOAD '/user/maria_dev/movielens/u.user'
```

```
    USING PigStorage('|')
    AS (
        user_id:int,
        age:int,
        gender:chararray,
        occupation:chararray,
        zip:chararray
    );

result = FOREACH users GENERATE
        user_id AS user_id,
        age AS age,
        gender AS gender,
        occupation AS occupation;

STORE result INTO '/user/maria_dev/movielens/u.user2' USING PigStorage('\t');
```

El resultado es:



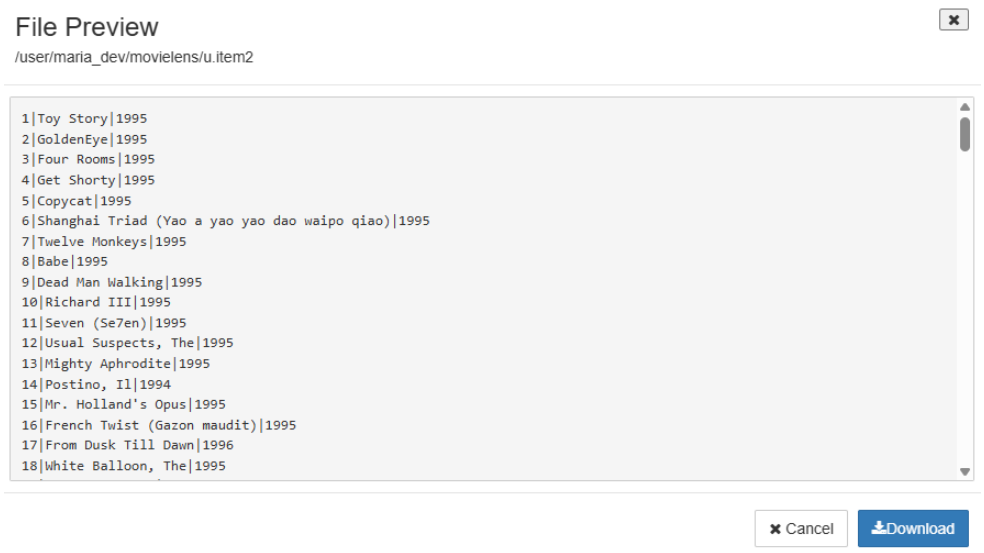
3) Utilizando PIG, del archivo u.item quédate solamente con las dos primeras columnas (id y título). Posteriormente de la columna título extrae el año y guárdala en una nueva columna (anio). En la columna título ha de quedar exclusivamente el título sin el año. Guarda el resultado en un archivo llamado u.item2

Script:

```
items = LOAD '/user/maria_dev/movielens/u.item'
    USING PigStorage('|')
    AS (
        id:int,
        titulo:chararray,
        release_date:long,
        video_release_date:long,
        IMDb_URL:chararray,
```

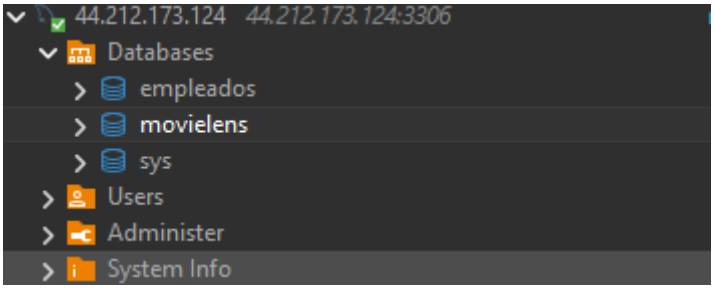
```
        unknown:boolean,
        action:boolean,
        adventure:boolean,
        animation:boolean,
        childrens:boolean,
        comedy:boolean,
        crime:boolean,
        documentary:boolean,
        drama:boolean,
        fantasy:boolean,
        film_noir:boolean,
         horror:boolean,
        musical:boolean,
        mystery:boolean,
        romance:boolean,
        sci_fi:boolean,
        thriller:boolean,
        war:boolean,
        western:boolean
    );

u2 = FOREACH items GENERATE
        id,
        titulo;

u3 = FOREACH u2 GENERATE
        id,
        REGEX_EXTRACT(titulo, '(.*) \\([0-9]{4}\\)', 1) AS titulo_limpio,
        REGEX_EXTRACT(titulo, '.*\\(([0-9]{4})\\)', 1) AS anio;

STORE u3 INTO '/user/maria_dev/movielens/u.item2' USING PigStorage('|');
```

Resultado:



3) APARTADO C

1) Crea en tu servidor MySQL de AWS una nueva base de datos llamada movielens.



2) En dicha base de datos crea tres tablas (usuarios, votos y películas) con la estructura adecuada para almacenar los ficheros u.data, u.user2 y u.item2. Utiliza sentencias CREATE TABLE. Crea los índices y relaciones entre las tres tablas.

Creamos las tablas usando este script:

```
CREATE DATABASE IF NOT EXISTS movielens;

USE movielens;

CREATE TABLE usuario (
	id int PRIMARY KEY NOT NULL AUTO_INCREMENT,
	edad int,
	genero varchar(255),
	ocupacion varchar(255)
);

CREATE TABLE pelicula (
	id int PRIMARY KEY NOT NULL AUTO_INCREMENT,
	titulo varchar(255),
	anyo int
);

CREATE TABLE voto (
	usuario_id int,
	pelicula_id int,
	valoracion varchar(255),
	fecha timestamp,

	FOREIGN KEY (usuario_id) REFERENCES usuario(id),
	FOREIGN KEY (pelicula_id) REFERENCES pelicula(id),

	PRIMARY KEY (usuario_id, pelicula_id)

);
```

3) Utilizando SQOOP, exporta los tres ficheros de HDFS a sus tablas.

Tabla “usuarios”.

```
[maria_dev@sandbox-hdp ~]$ sudo sqoop export --connect "jdbc:mysql://44.212.173.124:3306/movielens?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --table usuario --expo
rt-dir /user/maria_dev/movielens/u.user2 --fields-terminated-by '\t' --batch
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/27 08:26:10 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/27 08:26:10 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/27 08:26:10 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/27 08:26:10 INFO tool.CodeGenTool: Beginning code generation
25/11/27 08:26:12 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `usuario` AS t LIMIT 1
25/11/27 08:26:12 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `usuario` AS t LIMIT 1
```

Tabla “peliculas”.

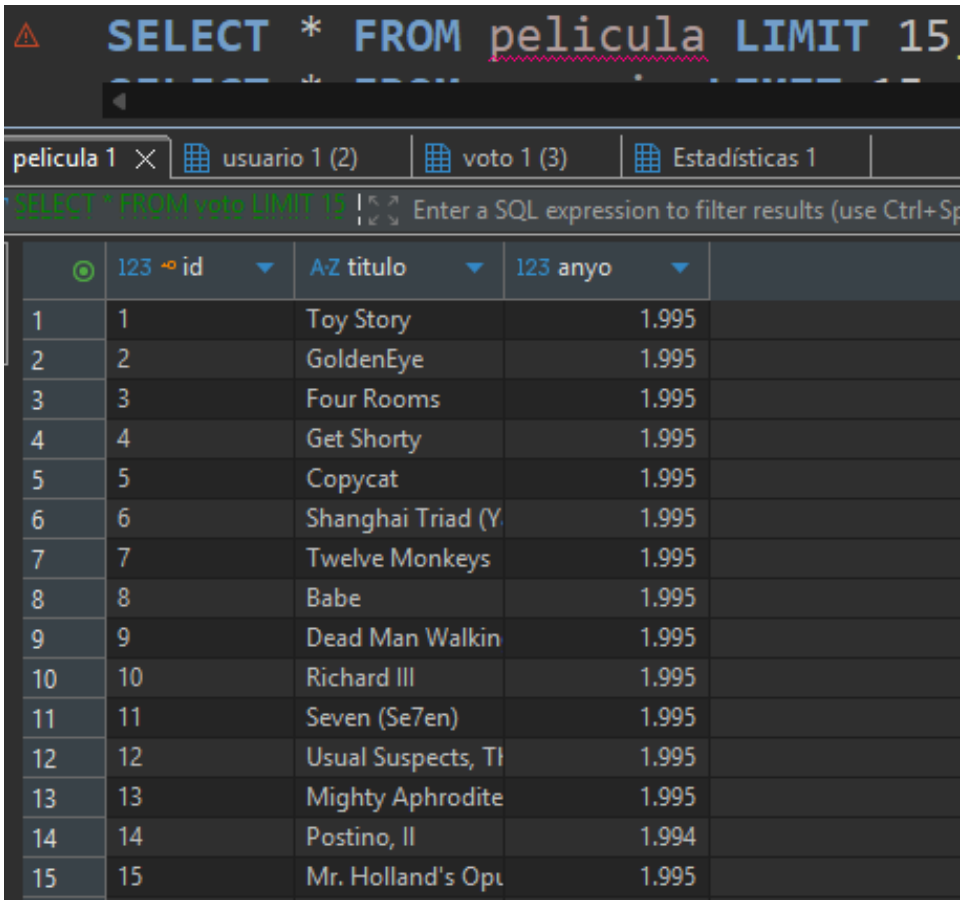
```
[maria_dev@sandbox-hdp ~]$ sudo sqoop export --connect "jdbc:mysql://44.212.173.124:3306/movielens?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --table pelicula --exp
ort-dir /user/maria_dev/movielens/u.item2 --fields-terminated-by '|' --batch
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/27 08:30:30 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/27 08:30:30 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/27 08:30:30 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/27 08:30:30 INFO tool.CodeGenTool: Beginning code generation
25/11/27 08:30:31 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `pelicula` AS t LIMIT 1
25/11/27 08:30:32 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `pelicula` AS t LIMIT 1
```

Tabla “valoraciones”.

```
[maria_dev@sandbox-hdp ~]$ sudo sqoop export --connect "jdbc:mysql://44.212.173.124:3306/movielens?useSSL=false&allowPublicKeyRetrieval=true" --username admin_remote --password '12345_Sql' --table voto --export-
dir /user/maria_dev/movielens/u.data --fields-terminated-by '\t' --batch
Warning: /usr/hdp/2.6.5.0-292/accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/11/27 08:35:16 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.5.0-292
25/11/27 08:35:16 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
25/11/27 08:35:16 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
25/11/27 08:35:16 INFO tool.CodeGenTool: Beginning code generation
25/11/27 08:35:18 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `voto` AS t LIMIT 1
25/11/27 08:35:18 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `voto` AS t LIMIT 1
```

4) Comprueba con sentencias SELECT que los ficheros se importaron correctamente.

Tabla “pelicula”



	123 id	A-Z titulo	123 anyo
1	1	Toy Story	1.995
2	2	GoldenEye	1.995
3	3	Four Rooms	1.995
4	4	Get Shorty	1.995
5	5	Copycat	1.995
6	6	Shanghai Triad (Y	1.995
7	7	Twelve Monkeys	1.995
8	8	Babe	1.995
9	9	Dead Man Walkin	1.995
10	10	Richard III	1.995
11	11	Seven (Se7en)	1.995
12	12	Usual Suspects, Tl	1.995
13	13	Mighty Aphrodite	1.995
14	14	Postino, Il	1.994
15	15	Mr. Holland's Opu	1.995

Tabla “usuarios”

```
SELECT * FROM usuario LIMIT 15;
```

usuario 1 (2) | voto 1 (3) | Estadísticas 1

Enter a SQL expression to filter results (use Ctrl+Sp

	123 id	123 edad	A-Z genero	A-Z ocupacion
1	1	24	M	technician
2	2	53	F	other
3	3	23	M	writer
4	4	24	M	technician
5	5	33	F	other
6	6	42	M	executive
7	7	57	M	administrator
8	8	36	M	administrator
9	9	29	M	student
10	10	53	M	lawyer
11	11	39	F	other
12	12	28	F	other
13	13	47	M	educator
14	14	45	M	scientist
15	15	49	F	educator

Tabla “usuarios”

```
SELECT * FROM voto LIMIT 15;
```

elicula 1 | usuario 1 (2) | voto 1 (3) | Estadísticas 1

Enter a SQL expression to filter results (use Ctrl+Space)

	123 usuario_id	123 pelicula_id	A-Z valoracion	123 fecha
1	1	1	5	874.965.758
2	1	2	3	876.893.171
3	1	3	4	878.542.960
4	1	4	3	876.893.119
5	1	5	3	889.751.712
6	1	6	5	887.431.973
7	1	7	4	875.071.561
8	1	8	1	875.072.484
9	1	9	5	878.543.541
10	1	10	3	875.693.118
11	1	11	2	875.072.262
12	1	12	5	878.542.960
13	1	13	5	875.071.805
14	1	14	5	874.965.706
15	1	15	5	875.071.608

4) APARTADO D

1) Top 10 películas más votadas de todos los tiempos (número de votos, no el valor de este).

```
SELECT p.titulo, count(v.valoracion) AS 'votos' FROM voto AS v
 LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
     GROUP BY v.pelicula_id
     ORDER BY count(v.valoracion) DESC
 LIMIT 10;
```

Connection: 44.212.173.124
Time: 2025-12-01 12:10:56.910
Query: -- 4. Año con más películas votadas (por número total de votos).

```
SELECT p.anyo, count(v.valoracion) AS 'votos' FROM voto AS v
LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
        GROUP BY p.anyo
        ORDER BY count(v.valoracion) DESC
LIMIT 1
```

titulo	votos
Star Wars	583
Contact	509
Fargo	508
Return of the Jedi	507
Liar Liar	485
English Patient, Tl	481
Scream	478
Toy Story	452
Air Force One	431
Independence Da	429

2) Películas con nota media ≥ 4.5 y al menos 100 valoraciones.

```
SELECT p.titulo, avg(v.valoracion) AS 'media valoraciones' FROM voto AS v
 LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
     GROUP BY v.pelicula_id
     HAVING avg(v.valoracion) >= 4.5
     AND count(v.valoracion) >= 100;
```

Enter a SQL expression to filter results (use Ctrl+Space)

titulo	media valoraciones

3) Usuarios que han dado más de 300 valoraciones y su nota media.

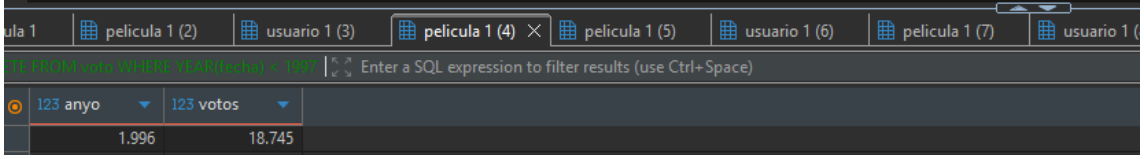
```
SELECT u.id, avg(v.valoracion) AS 'media valoraciones' FROM voto AS v
 LEFT JOIN usuario AS u ON v.usuario_id = u.id
     GROUP BY u.id
     HAVING count(v.valoracion) > 300;
```

Enter a SQL expression to filter results (use Ctrl+Space)

id	media valoraciones
7	3,9652605459
13	3,0974842767
59	3,9345549738
92	3,2448453608
94	3,6575
130	3,9858356941
145	3,335443038
151	3,996742671
181	1,491954023
194	2,9639344262

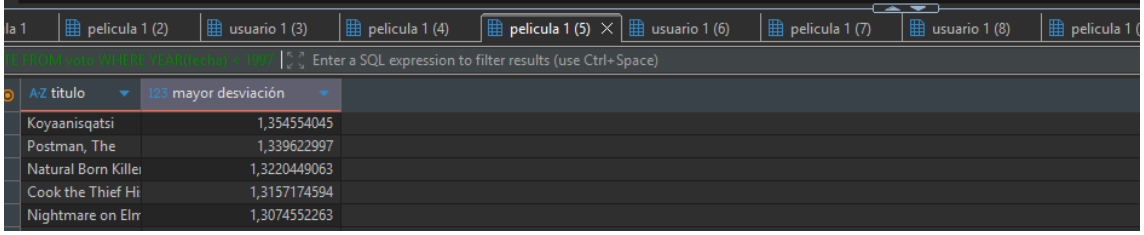
4) Año con más películas votadas (por número total de votos).

```sql
SELECT p.anyo, count(v.valoracion) AS 'votos' FROM voto AS v
 LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
     GROUP BY p.anyo
     ORDER BY count(v.valoracion) DESC
 LIMIT 1;
```



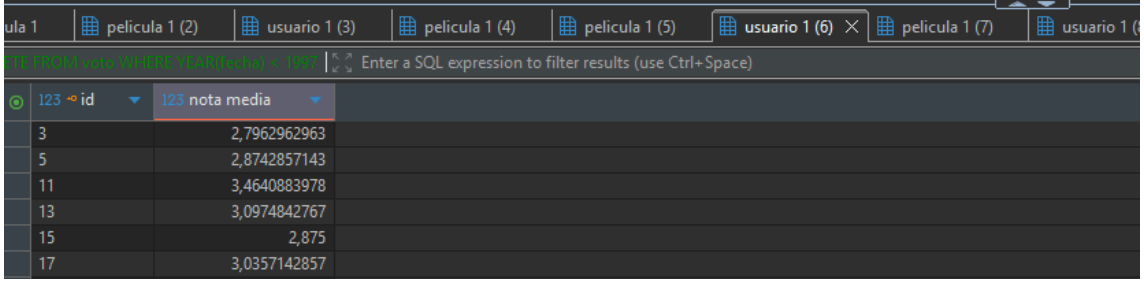
5) Las 5 películas más "polarizadas" (mayor desviación estándar, con valoraciones muy extremas) con al menos 50 votos. (Investiga qué función de SQL de da la desviación estándar).

```sql
SELECT p.titulo, STD(v.valoracion) AS 'mayor desviación' FROM voto AS v
 LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
    GROUP BY v.pelicula_id
    HAVING count(v.valoracion) >= 50
    ORDER BY STD(v.valoracion) DESC
 LIMIT 5;
```



6) Usuarios cuya nota media es menor que la nota media global.

```sql
SELECT u.id, avg(v.valoracion) AS 'nota media' FROM voto AS v
 LEFT JOIN usuario AS u ON v.usuario_id = u.id
     GROUP BY u.id
     HAVING avg(v.valoracion) < (
         SELECT avg(valoracion) FROM voto
     );
```



7) Películas que han recibido al menos una valoración de 1 y una de 5 (las más divididas).

```sql
SELECT * FROM pelicula AS p
    WHERE p.id IN (
        SELECT v.usuario_id FROM voto AS v
            WHERE v.valoracion = 1
    )
    AND p.id IN (
        SELECT v.usuario_id FROM voto AS v
            WHERE v.valoracion = 5
    );
```

pelicula 1 (2) | usuario 1 (3) | pelicula 1 (4) | pelicula 1 (5) | usuario

Enter a SQL expression to filter results (use Ctrl+Space)

id	titulo	anyo
1	Toy Story	1.995
2	GoldenEye	1.995
3	Four Rooms	1.995
5	Copycat	1.995
6	Shanghai Triad (Y	1.995
7	Twelve Monkeys	1.995
8	Babe	1.995
9	Dead Man Walkin	1.995
11	Seven (Se7en)	1.995
12	Usual Suspects, TI	1.995
13	Mighty Aphrodite	1.995

8) Top 10 usuarios más activos en 1997 (por número de valoraciones ese año).

```sql
SELECT u.*, count(v.valoracion) AS 'total valoraciones' FROM voto AS v
 LEFT JOIN usuario AS u ON u.id = v.usuario_id
    WHERE YEAR(v.fecha) = 1997
 GROUP BY v.usuario_id
    ORDER BY count(v.valoracion) DESC
 LIMIT 10;
```

pelicula 1 (2) | usuario 1 (3) | pelicula 1 (4) | pelicula 1 (5) | usuario 1 (6) | pelicula 1 (7) | usuario 1 (8) | pelicula 1

Enter a SQL expression to filter results (use Ctrl+Space)

id	edad	genero	ocupacion	total valoraciones
13	47	M	educator	523
303	19	M	student	484
276	21	M	student	459
181	26	M	executive	435
450	35	F	educator	426
429	27	M	student	414
222	29	M	programmer	373
417	27	F	other	359
378	35	M	student	351
592	18	M	student	346

9) Películas estrenadas después de 1995 con mejor nota media que Toy Story (1995).

```
SELECT p.titulo, avg(v.valoracion) FROM voto AS v
LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
    WHERE p.anyo > 1995
GROUP BY v.pelicula_id
    HAVING avg(v.valoracion) > (
        SELECT avg(v.valoracion) FROM voto AS v
            LEFT JOIN pelicula AS p ON v.pelicula_id = p.id
                WHERE p.titulo = 'Toy Story'
                GROUP BY p.id
    )
;
```

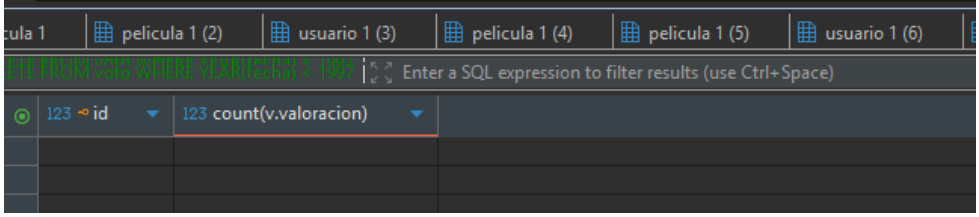
pelicula 1 (2) | usuario 1 (3) | pelicula 1 (4) | pelicula 1 (5) | usuario 1 (6) | pelicula 1 (7)

Enter a SQL expression to filter results (use Ctrl+Space)

titulo	avg(v.valoracion)
Fargo	4,155511811
Wallace & Gromit	4,447761194
Lone Star	4,0534759358
Big Night	3,9005847953
Sling Blade	4,1985294118
Kolya	3,9914529915
Chasing Amy	3,935483871
Shall We Dance?	4,2608695652
Full Monty, The	3,926984127

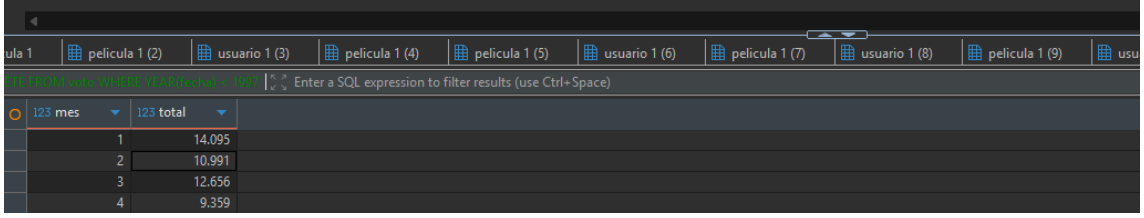
10) Usuarios que han valorado todas las películas estrenadas en 1993.

```sql
SELECT u.id, count(v.valoracion) FROM voto AS v
 LEFT JOIN usuario AS u ON v.usuario_id = u.id

    WHERE v.pelicula_id IN (
        SELECT p.id FROM pelicula AS p
            WHERE p.anyo = 1993
    )

 GROUP BY v.usuario_id

    HAVING count(v.valoracion) = (
        SELECT count(*) FROM pelicula AS p
            WHERE p.anyo = 1993
    )
;
```



11) Evolución mensual del número de valoraciones en 1998.

```sql
SELECT MONTH(v.fecha) AS 'mes', count(v.valoracion) AS 'total' FROM voto AS v
    WHERE CAST(v.fecha AS DATE) BETWEEN '1998-01-01' AND '1998-12-31'
 GROUP BY MONTH(v.fecha)
 ORDER BY MONTH(v.fecha) ASC;
```



12) Las 5 películas con mayor aumento de popularidad (comparar 1997 vs 1998).

```sql
SELECT p.titulo, count(v97.valoracion) - count(v98.valoracion) AS 'comparación de popularidad'
FROM (
    SELECT * FROM voto
        WHERE CAST(fecha AS DATE) BETWEEN '1997-01-01' AND '1997-12-31'
    ) AS v97
INNER JOIN (
    SELECT * FROM voto
        WHERE CAST(fecha AS DATE) BETWEEN '1998-01-01' AND '1998-12-31'
    ) AS v98
    ON v97.pelicula_id = v98.pelicula_id
LEFT JOIN pelicula AS p ON p.id = v97.pelicula_id

GROUP BY v97.pelicula_id
ORDER BY count(v97.valoracion) - count(v98.valoracion) DESC
LIMIT 5
;
```

titulo	comparación de popularidad
Toy Story	0
GoldenEye	0
Four Rooms	0
Get Shorty	0
Copycat	0

13) Usuarios que han valorado más películas que la media de su género.

```sql
SELECT u.id, u.genero, avg(v.valoracion) AS 'Media votos' FROM voto AS v
LEFT JOIN usuario AS u ON v.usuario_id = u.id
    GROUP BY v.usuario_id
    HAVING IF(
        u.genero = 'F',
        avg(v.valoracion) > (
            SELECT avg(valoracion) FROM voto
                WHERE usuario_id IN (
                    SELECT id FROM usuario
                        WHERE genero = 'F'
                )
        ),
        avg(v.valoracion) > (
            SELECT avg(valoracion) FROM voto
                WHERE usuario_id IN (
                    SELECT id FROM usuario
                        WHERE genero = 'M'
                )
        )
    )
;
```

id	genero	Media votos
1	M	3,6102941176
2	F	3,7096774194
4	M	4,3333333333
6	M	3,63507109
7	M	3,9652605459

14) Películas que nadie ha valorado con 3 estrellas (solo 1,2,4,5).

```
SELECT * FROM pelicula
 WHERE id NOT IN (
     SELECT pelicula_id FROM voto
         WHERE valoracion = 3
 );
```

uario 1 (3) | pelicula 1 (4) | pelicula 1 (5) | usuario 1 (6) | pelicula

ELETE FROM voto WHERE YEAR(fecha) < 1997 | Enter a SQL expression to filter results (us

	123 id	AZ titulo	123 anyo
1	75	Brother Minister: The Assassination of Malcolm X	1.994
2	247	Turbo: A Power Rangers Movie	1.997
3	314	3 Ninjas: High Noon At Mega Mountain	1.998
4	437	Amityville 1992: It's About Time	1.992
5	438	Amityville 3-D	1.983
6	439	Amityville: A New Generation	1.993
7	442	Amityville Curse, The	1.990

15) Ranking de días de la semana con más actividad (lunes, martes...).

```
SELECT  count(valoracion) 'contado', DAYOFWEEK(fecha) AS 'Dia' FROM voto
    GROUP BY DAYOFWEEK(fecha)
 ORDER BY count(valoracion) DESC ;
```

cula 1 (4) | pelicula 1 (5) | usuario 1 (6) | pelicula 1 (7) | usuario 1 (8) | pelicula 1 (9) | usuario 1 (10) | Resultados 1 (11) | Resul

Enter a SQL expression to filter results (use Ctrl+Space)

	123 contado	123 Dia
1	16.621	4
2	15.918	6
3	15.077	7
4	13.933	3
5	13.364	5
6	13.174	2
7	11.913	1

16) Usuarios que han dado su primera y última valoración con diferencia > 6 meses.

```sql
SELECT * FROM usuario AS u
 LEFT JOIN voto AS v ON u.id = v.usuario_id
 GROUP BY v.usuario_id
     HAVING MONTH(max(v.fecha) - min(v.fecha)) > 6;
```

usuario 1 (6) | pelicula 1 (7) | usuario 1 (8) | pelicula 1 (9) | usuario 1 (10) | Resultados 1 (11) | Resultados 1 (12)

Enter a SQL expression to filter results (use Ctrl+Space)

id	edad	genero	ocupacion	usuario_id	pelicula_id	valoracion	fecha
3	24	M	writer	3	181	4	1998-03-07 02:24:42
16	22	M	entertainment	16	1	5	1997-10-24 18:30:33
27	41	F	librarian	27	9	4	1998-04-02 18:49:02
39	42	M	entertainment	39	258	4	1998-04-01 03:11:20
111	58	M	engineer	111	242	4	1998-04-04 08:51:41
133	54	M	engineer	133	243	3	1998-03-22 17:50:35
139	21	M	student	139	100	5	1997-11-14 20:09:59
142	14	M	other	142	7	4	1998-02-28 04:34:49
166	48	M	educator	166	243	3	1998-02-02 05:37:07
179	16	M	entertainment	179	258	5	1998-04-09 19:47:50
206	15	F	student	206	242	3	1998-02-22 20:40:49
237	50	M	administrator	237	9	4	1997-11-12 23:18:50
240	24	F	educator	240	242	5	1998-01-26 00:48:03
241	27	F	student	241	268	4	1998-02-12 02:12:56
265	27	M	executive	265	1	5	1997-09-27 00:30:47

17) Las 10 películas con mayor ratio 5-estrellas / total valoraciones.

```sql
SELECT p.titulo, count(v5.valoracion) / count(v.valoracion) AS mayor_ratio FROM pelicula AS p
LEFT JOIN voto AS v ON p.id = v.pelicula_id
LEFT JOIN (
    SELECT * FROM voto
        WHERE valoracion = 5
) AS v5 ON p.id = v5.pelicula_id

GROUP BY p.id

LIMIT 10;
```

titulo	mayor ratio
Toy Story	1
GoldenEye	1
Four Rooms	1
Get Shorty	1
Copycat	1
Shanghai Triad (Y	1
Twelve Monkeys	1
Babe	1
Dead Man Walkin	1
Richard III	1

18) UPDATE: Aumenta en 1 año la edad de todos los usuarios (simulación de paso del tiempo).

```sql
UPDATE usuario SET edad = edad + 1;
```

	id	edad	genero	ocupacion
1	1	26	M	technician
2	2	55	F	other
3	3	25	M	writer
4	4	26	M	technician
5	5	35	F	other
6	6	44	M	executive
7	7	59	M	administrator
8	8	38	M	administrator
9	9	31	M	student
10	10	55	M	lawyer
11	11	41	F	other
12	12	30	F	other
13	13	49	M	educator
14	14	47	M	scientist
15	15	51	F	educator
16	16	23	M	entertainment
17	17	32	M	programmer
18	18	37	F	other
19	19	42	M	librarian
20	20	44	F	homemaker

19) INSERT: Añade una nueva película ficticia estrenada hoy.

```
INSERT INTO pelicula(titulo, anyo)
VALUES ('Mickey Mouse y Alberto Chicote, un amor que desenamora', 2025);
```

	id	titulo	anyo
1	1.686	Mickey Mouse y Alberto Chicote, un amor que desenamora	2.025

20) DELETE: Elimina todas las valoraciones anteriores a 1997.

```
DELETE FROM voto WHERE YEAR(fecha) < 1997;
```

usuario_id	pelicula_id	valoracion	fecha
259	255	4	1997-09-20 03:05:10
259	286	4	1997-09-20 03:05:27
259	298	4	1997-09-20 03:05:54
259	185	4	1997-09-20 03:06:21
259	173	4	1997-09-20 03:07:23
259	772	4	1997-09-20 03:08:02
259	108	4	1997-09-20 03:08:02
259	288	3	1997-09-20 03:08:25
259	928	4	1997-09-20 03:08:57
259	117	4	1997-09-20 03:09:48
259	200	4	1997-09-20 03:11:21
259	405	3	1997-09-20 03:12:00
259	1.074	3	1997-09-20 03:14:24
259	176	4	1997-09-20 03:16:26
259	357	5	1997-09-20 03:18:05
259	210	4	1997-09-20 03:18:05