PR_06.2 Dani Gayol Rodríguez

Gestion de tablas y formatos

1. Carga de Datos Delimitados (Básico):

```
CREATE EXTERNAL TABLE ml_ratings_ext (
    user_id INT,
    movie_id INT,
    rating INT,
    time_stamp BIGINT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
LOCATION '/user/hive/movielens/u.data';
```

```
LOAD DATA INPATH '/user/maria_dev/movielens/u.data' INTO TABLE ml_ratings_ext;
```

ml_ratings_ext.user_id	ml_ratings_ext.movie_id	ml_ratings_ext.rating	ml_ratings_ext.time_stamp
196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596
298	474	4	884182806

2. Definición de Esquema y Comentarios:

```
CREATE TABLE ml_users_managed (
    user_id INT,
    age INT,
    gender STRING,
    occupation STRING COMMENT 'Tipo de ocupación puesta por el usuario',
    zipcode STRING
)
COMMENT 'Tabla gestionada con información de usuarios'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

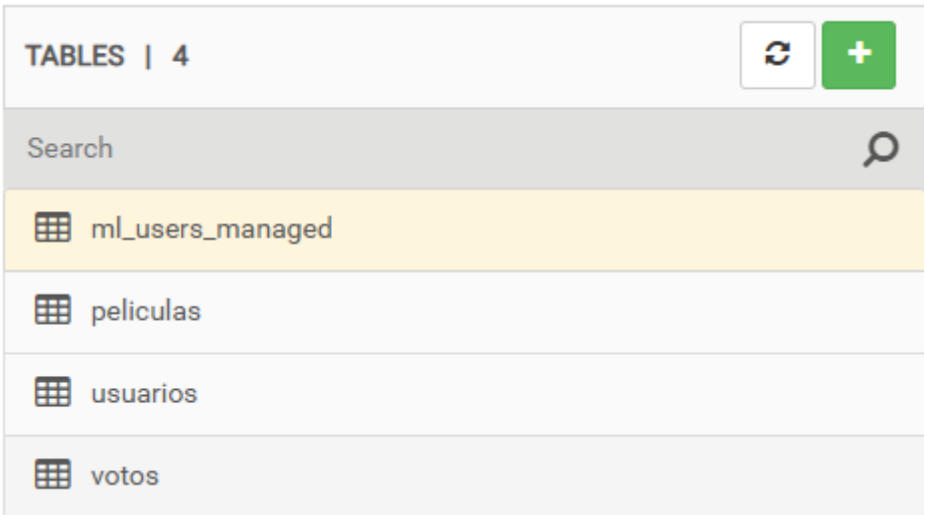
```
LOAD DATA INPATH '/user/maria_dev/movielens/u.user' INTO TABLE ml_users_managed;
```

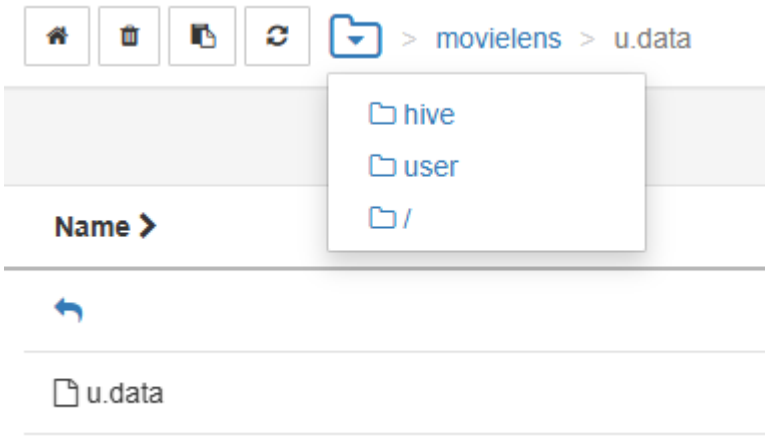
ml_users_managed.user_id	ml_users_managed.age	ml_users_managed.gender	ml_users_managed.occupation	ml_users_managed.zipcode
1	24	M	technician	85711
2	53	F	other	94043
3	23	M	writer	32067
4	24	M	technician	43537
5	33	F	other	15213

3. Simulación de Pérdida de Datos (DROP TABLE):

Para la tabla “ml_ratings_ext”, al ser “external”, Hive borra el metadata pero no elimina los archivos del directorio asociado

```
DROP TABLE ml_ratings_ext;
```





Para la tabla “ml_users_managed”, al ser “managed“, Hive controla el directorio y borra la carpeta y su contenido

```
1 DROP TABLE ml_users_managed;
```

Tipos de Datos Complejos

1. Uso de STRUCT (Información Personal):

```
CREATE TABLE ml_user_info (
    user_id INT,
    personal_data STRUCT<
        age: INT,
        gender: STRING
    >,
    occupation STRING,
    zipcode STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|';
```

```
SELECT user_id, personal_data.gender
FROM ml_user_info
WHERE personal_data.gender = 'F';
```

2. Uso de ARRAY (Géneros de Película):

Primera consulta

```
SELECT movie_id, size(genres_list) AS num_genres
FROM ml_movies_array;
```

Segunda consulta

```
SELECT genres_list[1]
FROM ml_movies_array
WHERE movie_id = 50;
```

3. Uso de MAP (Puntuación de Usuarios por Ocupación):

```
SELECT map_keys(avg_rating_by_occupation)
FROM ml_occupation_ratings;
```

Consultas Básicas y JOINS

1. Agregación Básica:

Como eliminamos esta tabla anteriormente, solo basta con crearla otra vez ya que los datos siguen guardados

```
CREATE EXTERNAL TABLE ml_ratings_ext (
    user_id INT,
    movie_id INT,
    rating INT,
    time_stamp BIGINT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
LOCATION '/user/hive/movielens/u.data';
```

Ahora hacemos la consulta

```
SELECT AVG(rating) AS avg_rating
FROM ml_ratings_ext;
```

avg_rating
3.52986

2. Filtrado y Agrupación:

Como eliminamos esta tabla anteriormente, aquí si hay que crear la tabla y insertarle los datos otra vez

```sql
CREATE TABLE ml_users_managed (
    user_id INT,
    age INT,
    gender STRING,
    occupation STRING COMMENT 'Tipo de ocupación declarada por el usuario',
    zipcode STRING
)
COMMENT 'Tabla gestionada con información de usuarios MovieLens'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

```sql
LOAD DATA INPATH '/user/maria_dev/movielens/u.user' INTO TABLE ml_users_managed;
```

Ahora ya podemos hacer la consulta

```sql
SELECT gender, occupation, COUNT(*) AS total_users
FROM ml_users_managed
GROUP BY gender, occupation
ORDER BY total_users DESC;
```

gender	occupation	total_users
M	student	136
M	other	69
M	educator	69
M	engineer	65
F	student	60

3. INNER JOIN:

Si quieres que la media no salga con muchas decimanles puedes usar “ROUND” para redondear al número de decimales que quieras

```sql
SELECT u.occupation, AVG(r.rating) AS avg_rating
FROM ml_ratings_ext r
INNER JOIN ml_users_managed u
    ON r.user_id = u.user_id
GROUP BY u.occupation
ORDER BY avg_rating DESC;
```

u.occupation	avg_rating
none	3.779134295227525
lawyer	3.7353159851301116
doctor	3.688888888888889
educator	3.6706206312221985
artist	3.653379549393414

4. LEFT OUTER JOIN:

```sql
CREATE TABLE unrated_users (user_id INT);

INSERT INTO unrated_users VALUES (1), (2), (3), (4), (5);
```

```sql
SELECT u.user_id, r.rating
FROM ml_users_managed u
LEFT OUTER JOIN ml_ratings_ext r
    ON u.user_id = r.user_id
WHERE r.rating IS NULL;
```

5. LEFT SEMI JOIN (Simulación de IN/EXISTS):

```sql
SELECT u.user_id
FROM ml_users_managed u
LEFT SEMI JOIN ml_ratings_ext r
    ON u.user_id = r.user_id
WHERE u.occupation = 'programmer';
```

u.user_id
17
29
45
53
55
58
82