CIND719-DK0T
Assignment 2:

Ramello Peralta 500519802

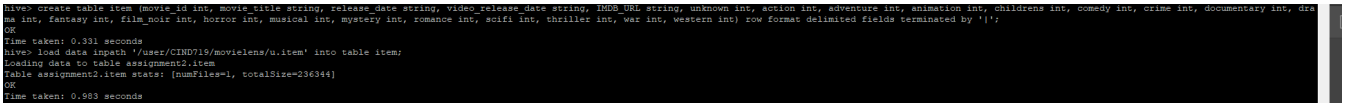
1. Store the data in a Hive database ml as table userratings (u.data), users (u.user) (1 pt)

```
hive> create database assignment2
    > ;
OK
Time taken: 0.051 seconds
```

```
hive> create table assignment2.userratings (user_id int, item_id int, rating int
, timestamp int) row format delimited fields terminated by '\t';
OK
Time taken: 0.489 seconds
```

```
hive> load data inpath '/user/CIND719/movielens/u.data' into table assignment2.u
serratings;
Loading data to table assignment2.userratings
Table assignment2.userratings stats: [numFiles=1, totalSize=1979173]
OK
Time taken: 1.129 seconds
hive> use assignment2;
OK
Time taken: 0.279 seconds
hive> select * from userratings limit 5;
OK
196     242     3       881250949
186     302     3       891717742
22      377     1       878887116
244     51      2       880606923
166     346     1       886397596
Time taken: 0.413 seconds, Fetched: 5 row(s)
```

```
hive> create table users (user_id int, age int, gender string, occupation string
, zip_code int) row format delimited fields terminated by '|';
OK
Time taken: 0.302 seconds
hive> load data inpath '/user/CIND719/movielens/u.user/' into table users;
Loading data to table assignment2.users
Table assignment2.users stats: [numFiles=1, totalSize=22628]
OK
Time taken: 1.095 seconds
hive> select * from users limit 5;
OK
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
Time taken: 0.144 seconds, Fetched: 5 row(s)
hive>
```



- Creating the item table as it will be used in later questions

Create table item (movie_id int, movie_title string, release_date string, video_release_date string, IMDB_URL string, unknown int, adventure int, animation int, childrens int, comedy int, crime int, documentary int, drama int, fantasy int, film_noir int, horror int, musical int, mystery int, romance int, scifi int, thriller int, war int, western int) row format delimited fields terminated by ‘|’;

Load data inpath ‘/user/CIND719/movielens/u.item’ into table item;

2. Write HiveQL queries to confirm the number of records in both tables.

```
hive> select count(*) from userratings;
Query ID = root_20210314212525_e54c16a7-dcef-4923-ab35-ed965c0a09bc
Total jobs = 1
Launching Job 1 out of 1


Status: Running (Executing on YARN cluster with App id application_1615755931817_0002)

----------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 4.25 s
----------------------------------------------------------------------------------
OK
100000
Time taken: 4.947 seconds, Fetched: 1 row(s)
hive> select count(*) from users;
Query ID = root_20210314212525_f8055af5-c065-4d0b-a73a-de25c2bc0a61
Total jobs = 1
Launching Job 1 out of 1


Status: Running (Executing on YARN cluster with App id application_1615755931817_0002)

----------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 4.26 s
----------------------------------------------------------------------------------
OK
943
Time taken: 4.974 seconds, Fetched: 1 row(s)
hive>
```

3. Extract the list of top 10 items (movies) that received the most ratings (not necessarily highest rating) from female educators. (2 pts)

```
hive>
    > select movie_title, count(rating) as c from item right join userratings on item.movie_id = userratings.item_id join u
sers on users.user_id = userratings.user_id where users.gender = 'F' and users.occupation = 'educator' group by movie_title
 order by c desc limit 10;
Query ID = root_20210314220404_539536f1-f4f8-4479-8ae4-203beba2da35
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.


Status: Running (Executing on YARN cluster with App id application_1615755931817_0004)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 2 ..........   SUCCEEDED      1          1        0        0       0       0
Map 5 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 4 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 05/05  [==========================>>] 100%  ELAPSED TIME: 9.99 s
--------------------------------------------------------------------------------
OK
English Patient, The (1996)     23
Contact (1997)  15
Full Monty, The (1997)  14
In & Out (1997) 13
Star Wars (1977)        13
Jerry Maguire (1996)    12
Emma (1996)     12
Schindler's List (1993) 12
Liar Liar (1997)        12
Sense and Sensibility (1995)    12
Time taken: 18.533 seconds, Fetched: 10 row(s)
hive>
```

4. Find the highest rated Fantasy movie.

- Highest rating means the highest average so the avg function is used

```
hive> select movie_title, avg(rating) as avgRating from item join userratings on
 movie_id=item_id where fantasy = 1 group by movie_title order by avgRating desc
 limit 1;
Query ID = root_20210324011717_6cbe7e1c-ab90-446c-98b5-d54276757388
Total jobs = 1
Launching Job 1 out of 1


Status: Running (Executing on YARN cluster with App id application_1616548340612
_0001)

----------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 8.86 s
----------------------------------------------------------------------------------
OK
Star Kid (1997) 5.0
Time taken: 12.347 seconds, Fetched: 1 row(s)
hive>
```

5. Load the u.data, and u.user files into Apache Spark as DataFrames named df_udata and df_uuser. Apply following queries. (2 pts)

```
cu@spark3cuvm:~/data$ cd movielens
cu@spark3cuvm:~/data/movielens$ ls
u.data  u.item  u.user
cu@spark3cuvm:~/data/movielens$
```

```
>>> df_uuser = spark.read.format('csv').option('delimiter','|').option('header','false').schema('user
_id int, age int, gender string, occupation string, zip_code int').load('/home/cu/data/movielens/u.us
er')
>>> df_uuser.show(5)
+-------+---+------+----------+--------+
|user_id|age|gender|occupation|zip_code|
+-------+---+------+----------+--------+
|      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|
+-------+---+------+----------+--------+
only showing top 5 rows
```

```
>>> df_udata = spark.read.format('csv').option('delimiter','\t').option('header','false').schema('use
r_id int, item_id int, rating int, timestamp string').load('/home/cu/data/movielens/u.data')
>>> df_udata.show(5)
+-------+-------+------+---------+
|user_id|item_id|rating|timestamp|
+-------+-------+------+---------+
|    196|    242|     3|881250949|
|    186|    302|     3|891717742|
|     22|    377|     1|878887116|
|    244|     51|     2|880606923|
|    166|    346|     1|886397596|
+-------+-------+------+---------+
only showing top 5 rows
```

a. How many unique occupations are in the data and what is the frequency of each occupation? (2 pts)

```
>>> df_uuser.registerTempTable('uuser_table')
```

```
>>> spark.sql('select distinct occupation from uuser_table').show()
+--------------+
|    occupation|
+--------------+
|     librarian|
|       retired|
|        lawyer|
|          none|
|        writer|
|    programmer|
|     marketing|
|         other|
|     executive|
|     scientist|
|       student|
|      salesman|
|        artist|
|    technician|
|administrator|
|      engineer|
|    healthcare|
|      educator|
|entertainment|
|     homemaker|
+--------------+
only showing top 20 rows
```

```
>>> spark.sql('select occupation, count(*) as f from uuser_table group by occupation').show()
+-------------+---+
|   occupation|  f|
+-------------+---+
|    librarian| 51|
|      retired| 14|
|       lawyer| 12|
|         none|  9|
|       writer| 45|
|   programmer| 66|
|    marketing| 26|
|        other|105|
|    executive| 32|
|    scientist| 31|
|      student|196|
|     salesman| 12|
|       artist| 28|
|   technician| 27|
|administrator| 79|
|     engineer| 67|
|   healthcare| 16|
|     educator| 95|
|entertainment| 18|
|    homemaker|  7|
+-------------+---+
only showing top 20 rows
```

b. Find the number of recommendations corresponding to each occupation. (3 pts)

```
>>> df_udata.registerTempTable('udata_table')
```

```
>>> spark.sql('select occupation, count(*) as no_of_recs from uuser_table left join udata_table on uu
ser_table.user_id=udata_table.user_id group by occupation').show()
+-------------+----------+
|   occupation|no_of_recs|
+-------------+----------+
|    librarian|      5273|
|      retired|      1609|
|       lawyer|      1345|
|         none|       901|
|       writer|      5536|
|   programmer|      7801|
|    marketing|      1950|
|        other|     10663|
|    executive|      3403|
|    scientist|      2058|
|      student|     21957|
|     salesman|       856|
|       artist|      2308|
|   technician|      3506|
|administrator|      7479|
|     engineer|      8175|
|   healthcare|      2804|
|     educator|      9442|
|entertainment|      2095|
|    homemaker|       299|
+-------------+----------+
only showing top 20 rows
```

End of Assignment 2.