USE CASE #1: List all the movies and the number of ratings

hive> select title, count(*) from movies right outer join ratings on movies.movie_id=ratings.movie_id group by movies.movie_id, title;

Store ratings data into new table

CREATE TABLE mov_rating_count(movie_id bigint, title string, rating_count bigint);

INSERT OVERWRITE TABLE mov_rating_count

SELECT movie_id, title, count(*)

FROM movies

RIGHT OUTER JOIN ratings

ON movies.movie_id=ratings.movie_id

GROUP BY movies.movie_id, title;

hive> INSERT OVERWRITE TABLE mov_rating_count

> SELECT movie_id, title, count(*)

> FROM movies

> RIGHT OUTER JOIN ratings

> ON movies.movie_id=ratings.movie_id

> GROUP BY movies.movie_id, title;

Loading data to table movie_lens_data.mov_rating_count

Table movie_lens_data.mov_rating_count stats: [numFiles=2, numRows=33670, totalSize=1134891, rawDataSize=1101221]

OK

Time taken: 31.771 seconds

hive>

```
hive> select movie_id, title, rating_count from mov_rating_count;
OK
2       Jumanji (1995)  23950
3       Grumpier Old Men (1995) 15267
5       Father of the Bride Part II (1995)      14769
6       Heat (1995)     26593
10      GoldenEye (1995)        31357
13      Balto (1995)    1648
14      Nixon (1995)    6750
18      Four Rooms (1995)       5781
19      Ace Ventura: When Nature Calls (1995)   22877
23      Assassins (1995)        4636
Time taken: 0.182 seconds, Fetched: 10 row(s)
hive>
```

USE CASE #2: List all the users and the number of ratings they have done for a movie

```
hive> SELECT u.user_id, u.name, COUNT(r.rating)
 FROM users u, ratings r WHERE u.user_id=r.user_id
 GROUP BY u.user_id, u.name;
```

Store users data into new table

```
CREATE TABLE IF NOT EXISTS user_rating_count ( user_id bigint, name String, rating_count int) COMMENT 'Details how many movies a user rated.' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' STORED AS TEXTFILE;

INSERT OVERWRITE TABLE user_rating_count
 SELECT u.user_id, u.name, COUNT(r.rating)
 FROM users u, ratings r WHERE u.user_id=r.user_id
 GROUP BY u.user_id, u.name;

hive> select user_id, name, rating_count from user_rating_count;
OK
1       Test User 1     3
2       Test User 2     4
3       Test User 3     4
4       Test User 4     183
5       Test User 5     25
6       Test User 6     18
7       Test User 7     20
8       Test User 8     15
9       Test User 9     16
10      Test User 10    30
```

USE CASE #3: List all the Movie IDs which have been rated (Movie Id with at least one user rating it)

select movie_id, title from mov_rating_count;

or

select DISTINCT ratings.movie_id,movies.title from ratings LEFT JOIN movies where ratings.movie_id = movies.movie_id ;

hive> select movie_id, title from mov_rating_count LIMIT 10;

OK

2 Jumanji (1995)

3 Grumpier Old Men (1995)

5 Father of the Bride Part II (1995)

6 Heat (1995)

10 GoldenEye (1995)

13 Balto (1995)

14 Nixon (1995)

18 Four Rooms (1995)

19 Ace Ventura: When Nature Calls (1995)

23 Assassins (1995)

USE CASE #4: List all the Users who have rated the movies (Users who have rated atleast one movie)

hive> select user_id, name from user_rating_count;

OK

1 Test User 1

2 Test User 2

3 Test User 3

4 Test User 4

5 Test User 5

6 Test User 6

7 Test User 7

8 Test User 8

9 Test User 9

10 Test User 10

Time taken: 0.134 seconds, Fetched: 10 row(s)

hive>

USE CASE #5: List of all the User with the max, min, average ratings they have given against any movie

hive> select user_id, max(rating), min(rating), round(avg(rating),2) from ratings group by user_id;

Query ID = hive_20160714172915_16fd4bee-4dfa-4e05-8d3d-ba6769d7001d

Total jobs = 1

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1468446400470_0021)

--

VERTICES STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED

--

Map 1 ..... SUCCEEDED 9 9 0 0 0 0

Reducer 2 ..... SUCCEEDED 2 2 0 0 0 0

--

VERTICES: 02/02 [==========>>] 100% ELAPSED TIME: 22.99 s

--

OK

1 5.0 2.5 3.5

6 5.0 1.0 3.64

7 5.0 1.5 4.25

9 5.0 1.0 3.44

12 5.0 1.0 4.08

13 5.0 1.0 2.55

14 5.0 1.0 2.94

19 5.0 3.5 4.37

42483 5.0 1.0 3.86

42484 5.0 3.0 3.83

Time taken: 23.487 seconds, Fetched: 10 row(s)

hive>

USE CASE #6: List all the Movies with the max, min, average ratings given by any user

```
SELECT m.movie_id, m.title, MAX(r.rating),
 AVG(r.rating), MIN(r.rating) FROM movies m,
 ratings r WHERE m.movie_id=r.movie_id
 GROUP BY m.movie_id,m.title;
```

Store movies data into new table

```
CREATE TABLE IF NOT EXISTS movie_ratings ( movie_id bigint,
 title String,
 max_rating float,
 avg_rating float,
 min_rating float)
 COMMENT 'Max min avg rating of any movie.'
 ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n' STORED AS TEXTFILE;

INSERT OVERWRITE TABLE movie_ratings
 SELECT m.movie_id, m.title, MAX(r.rating),
 AVG(r.rating), MIN(r.rating) FROM movies m,
 ratings r WHERE m.movie_id=r.movie_id
 GROUP BY m.movie_id,m.title;
```

hive> select movie_id, title, max_rating, avg_rating, min_rating from movie_ratings LIMIT 20;

OK

1 Toy Story (1995) 5.0 3.8948016 0.5

2 Jumanji (1995) 5.0 3.2210855 0.5

3 Grumpier Old Men (1995) 5.0 3.1800942 0.5

4 Waiting to Exhale (1995) 5.0 2.8797274 0.5

5 Father of the Bride Part II (1995) 5.0 3.0808113 0.5

6 Heat (1995) 5.0 3.836536 0.5

7 Sabrina (1995) 5.0 3.3733666 0.5

8 Tom and Huck (1995) 5.0 3.139661 0.5

9 Sudden Death (1995) 5.0 3.015246 0.5

10 GoldenEye (1995) 5.0 3.436888 0.5

11 "American President 5.0 3.6641243 0.5

12 Dracula: Dead and Loving It (1995) 5.0 2.670864 0.5

13 Balto (1995) 5.0 3.2976334 0.5

14 Nixon (1995) 5.0 3.4313333 0.5

15 Cutthroat Island (1995) 5.0 2.7282789 0.5

16 Casino (1995) 5.0 3.7851126 0.5

17 Sense and Sensibility (1995) 5.0 3.9575002 0.5

18 Four Rooms (1995) 5.0 3.4020066 0.5

19 Ace Ventura: When Nature Calls (1995) 5.0 2.6226342 0.5

20 Money Train (1995) 5.0 2.8992693 0.5

Time taken: 0.106 seconds, Fetched: 20 row(s)

hive>