

**Kathmandu University**  
**Department of Computer Science and Engineering**  
**Dhulikhel, Kavre**



A Report on 'Lab Work 4' [COMP 232]

**Submitted by:**

Chandan Kumar Mahato (31)

II-year, II semester

**Submitted to:**

Dr. Rajani Chulyadyo

Department of Computer Science and Engineering

**Submission Date:** March 31, 2022

## *Qlab4*

### Tasks

1. Install PostgreSQL server.
2. Import the given data.  
[https://drive.google.com/drive/folders/1DZgVkk\\_mZzfEyty5q4Zj3i1VPhU9FTD\\_](https://drive.google.com/drive/folders/1DZgVkk_mZzfEyty5q4Zj3i1VPhU9FTD_)
3. Write some SQL queries to explore the tables.
4. Create a view for the following
  - I. average rating of all movies
  - II. number of actors in each movie
  - III. number of ratings for each movie
  - IV. number of ratings by each user
5. Find the number of users who have rated at least one movie.
6. Find the number of unrated movies.
7. Find top 10 highest rated movies and the actors who played in those movies.

### Deliverables

SQL script for Tasks 3 - 7 and results obtained in Tasks 5 - 7

### *Solution:*

Queries SQL (structured query language) for the following questions:

3. Write Some SQL queries to explore the tables.

```
SELECT * FROM actors;
SELECT * FROM directors
SELECT * FROM movies;
SELECT * FROM movies2actors;
SELECT * FROM movies2directors;
SELECT * FROM u2base;
SELECT * FROM users;
SELECT DISTINCT a_gender FROM actors;
SELECT DISTINCT rating FROM u2base;
```

#### 4. Create a View for the following

##### 4.1. average rating of all movies.

```
CREATE VIEW rating_integer AS
(
    SELECT userid, movieid,
    CAST(rating AS integer) AS Rating
    FROM u2base
);
SELECT avg(Rating) AS average_rating FROM rating_integer;
```

```
20 -- 4. Create a view for the following
21 -- 4 (1) average rating of all movies
22 CREATE VIEW rating_integer AS
23 (
24     SELECT userid, movieid,
25     CAST(rating AS integer) AS Rating
26     FROM u2base
27 );
28 SELECT avg(Rating) AS average_rating FROM rating_integer;
29
```

Data Output Explain Messages Notifications

	average_rating numeric	
1	3.5822745164175598	

## 4.2. number of actors in each movie.

```
CREATE VIEW no_of_actors AS
(
    SELECT
        movies2actors.movieid AS movie,
        COUNT(movies2actors.actorid) AS actor_count
    FROM
        movies2actors
    GROUP BY movies2actors.movieid
);
SELECT * FROM no_of_actors;
```

```
29
30 -- 4 (2) number of actors in each movie
31 CREATE VIEW no_of_actors AS
32 (
33     SELECT
34         movies2actors.movieid AS movie,
35         COUNT(movies2actors.actorid) AS actor_count
36     FROM
37         movies2actors
38     GROUP BY movies2actors.movieid
39 );
40 SELECT * FROM no_of_actors;
41
```

Data Output Explain Messages Notifications

	movie integer	actor_count bigint
1	1892794	38
2	1932235	10
3	2457395	43
4	2032626	27
5	1915498	39
6	1759448	80
7	2273048	56
8	2327747	35
9	2313912	36
10	1955231	47
11	2295575	81

### 4.3. number of ratings for each movie.

```
CREATE VIEW ratings_for_each_movie AS
(
    SELECT
        u2base.movieid AS movie,
        COUNT(u2base.rating) AS rating_count
    FROM
        u2base
    GROUP BY u2base.movieid
);
SELECT * FROM ratings_for_each_movie;
```

```
42 -- 4 (3) number of rating for each movie
43 CREATE VIEW ratings_for_each_movie AS
44 (
45     SELECT
46         u2base.movieid AS movie,
47         COUNT(u2base.rating) AS rating_count
48     FROM
49         u2base
50     GROUP BY u2base.movieid
51 );
52 SELECT * FROM ratings_for_each_movie;
53
```

Data Output Explain Messages Notifications




	movie integer	rating_count bigint
1	1892794	119
2	1932235	59
3	2032626	348
4	2457395	109
5	1915498	564
6	1759448	40
7	2273048	643
8	2327747	576
9	2313912	475
10	1955231	7
11	2295575	215

#### 4.4. number of ratings by each user

```
CREATE VIEW ratings_by_each_user AS
(
    SELECT
        u2base.userid AS USER,
        COUNT (u2base.rating) AS rating_count
    FROM
        u2base
    GROUP BY u2base.userid
);
SELECT * FROM ratings_by_each_user;
```

```
54 -- 4 (4) number of rating by each user
55 CREATE VIEW ratings_by_each_user AS
56 (
57     SELECT
58         u2base.userid AS USER,
59         COUNT (u2base.rating) AS rating_count
60     FROM
61         u2base
62     GROUP BY u2base.userid
63 );
64 SELECT * FROM ratings_by_each_user;
```

Data Output Explain Messages Notifications

	 user integer	 rating_count bigint	
1	1489	488	
2	4790	377	
3	273	187	
4	3936	34	
5	2574	32	
6	951	20	
7	5761	223	
8	5843	219	
9	5729	62	
10	5468	383	
11	4326	127	
12	5259	34	

5. Find the number of users who have rated at least one movie.

```
CREATE VIEW user_count AS(
SELECT rating_integer.userid AS users
  FROM
    rating_integer
  INNER JOIN
    users
  ON rating_integer.userid = users.userid
 WHERE rating_integer.rating != 0
)
SELECT Count(users) FROM user_count;
```

```
65
66 -- 5 Find the number of users who have rated at least one movie.
67 CREATE VIEW user_count AS(
68 SELECT rating_integer.userid AS users
69     FROM
70     rating_integer
71   INNER JOIN
72     users
73   ON rating_integer.userid = users.userid
74   WHERE rating_integer.rating != 0
75 )
76 SELECT Count(users) FROM user_count;
77
```

Data Output Explain Messages Notifications

	count bigint
1	996159

6. Find the number of unrated movies

```
CREATE VIEW rating_integer AS
(
  SELECT userid, movieid,
  CAST(rating AS integer) AS Rating
  FROM u2base
);
SELECT count(rating_integer.movieid) FROM
rating_integer WHERE rating_integer.rating = 0;
```

```
78 -- 6 Find the number of unrated movies
79 CREATE VIEW rating_integer AS
80 (
81   SELECT userid, movieid,
82   CAST(rating AS integer) AS Rating
83   FROM u2base
84 );
85 SELECT count(rating_integer.movieid) FROM
86 rating_integer WHERE rating_integer.rating = 0;
87
```

Data Output Explain Messages Notifications

	count bigint
1	0

7. Find top 10 highest rated movies and the actors who played in those movies.

```
CREATE VIEW highly_rated_movies AS (SELECT rating_integer.movieid,  
COUNT(rating_integer.rating) AS rating_count  
FROM  
rating_integer  
GROUP BY(rating_integer.movieid)  
ORDER BY rating_count DESC LIMIT 10);  
  
SELECT * FROM highly_rated_movies;  
  
SELECT  
highly_rated_movies.movieid, movies2actors.actorid  
FROM  
movies2actors  
INNER JOIN  
highly_rated_movies  
ON highly_rated_movies.movieid = movies2actors.movieid;
```

```
88 -- 7 Find top 10 highest rated movies and the actors who played in those movies.  
89 CREATE VIEW highly_rated_movies AS (SELECT rating_integer.movieid,  
90 COUNT(rating_integer.rating) AS rating_count  
91 FROM  
92 rating_integer  
93 GROUP BY(rating_integer.movieid)  
94 ORDER BY rating_count DESC LIMIT 10);  
95  
96 SELECT * FROM highly_rated_movies;
```

Data Output Explain Messages Notifications

	movieid integer	rating_count bigint
1	1721568	3427
2	2371726	2990
3	2371786	2989
4	2371787	2882
5	2058680	2671
6	2324123	2652
7	2401750	2649
8	2457464	2590
9	1749731	2583
10	2478414	2577



```
95
96 SELECT * FROM highly_rated_movies;
97
98 SELECT
99 highly_rated_movies.movieid, movies2actors.actorid
100 FROM
101 movies2actors
102 INNER JOIN
103 highly_rated_movies
104 ON highly_rated_movies.movieid = movies2actors.movieid;
```

Data Output Explain Messages Notifications

	<div>movieid</div> <div>integer</div>	<div>actorid</div> <div>integer</div>	
1	1721568	42050	
2	1721568	69157	
3	1721568	86411	
4	1721568	129334	
5	1721568	282670	
6	1721568	317484	
7	1721568	384945	
8	1721568	482454	

✓ Successfully run. To

