Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavrepalanchowk

A lab report on

"Lab 04"

Sub Code: COMP 232

Submitted by:

Rohan Dhakal (Roll No. 14)

Submitted to:

Dr. Rajani Chulyadyo

Department of Computer Science and Engineering

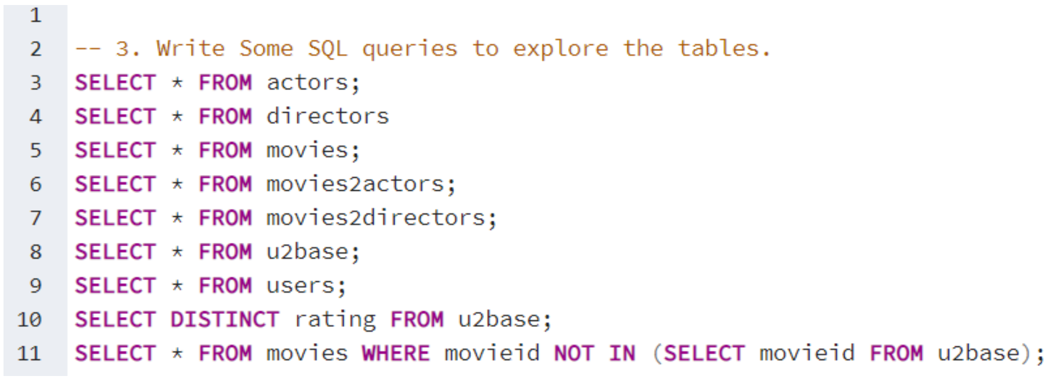
Submission Date: 2022/04/05

Tasks

1. Install PostgreSQL server.
2. Import the given data.
   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 the top 10 highest rated movies and the actors who played in those movies.

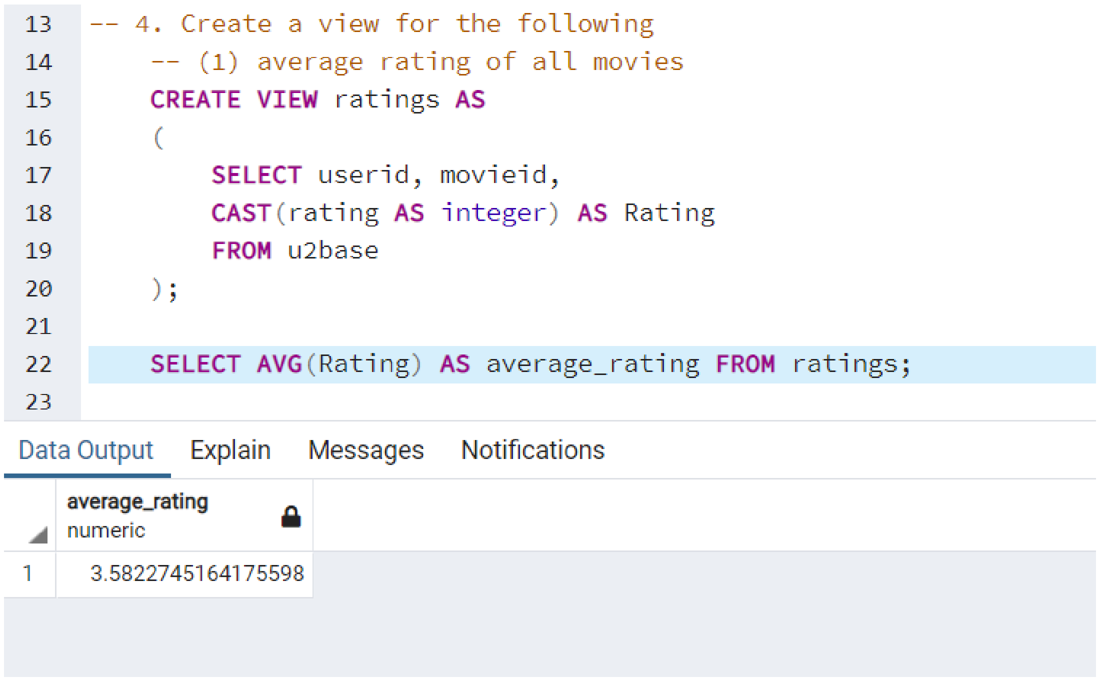
Deliverables

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



```sql

-- 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 rating FROM u2base;
SELECT * FROM movies WHERE movieid NOT IN (SELECT movieid FROM u2base);
```



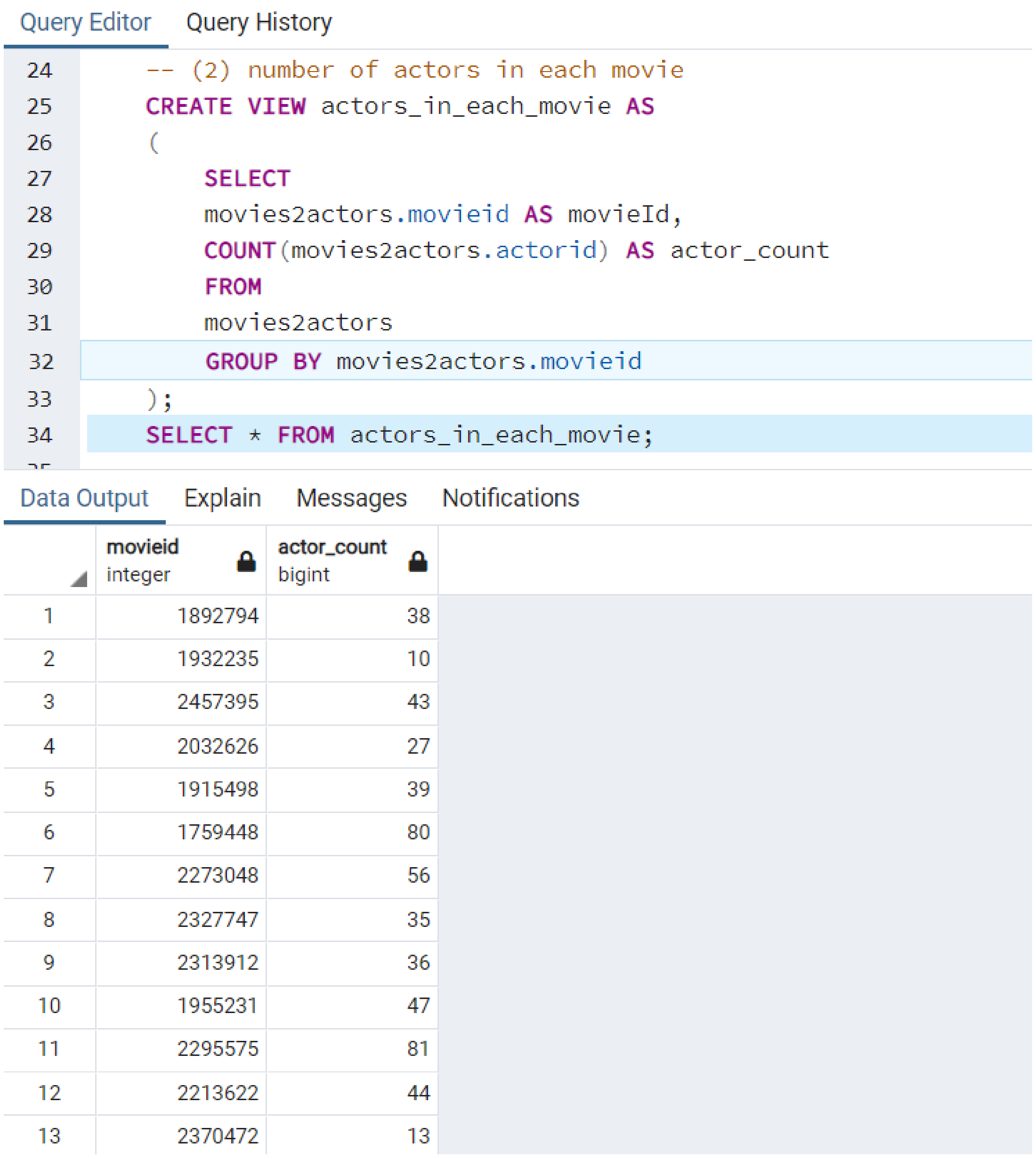
```sql
-- 4. Create a view for the following
    -- (1) average rating of all movies
    CREATE VIEW ratings AS
    (
        SELECT userid, movieid,
        CAST(rating AS integer) AS Rating
        FROM u2base
    );

    SELECT AVG(Rating) AS average_rating FROM ratings;

```

Data Output   Explain   Messages   Notifications

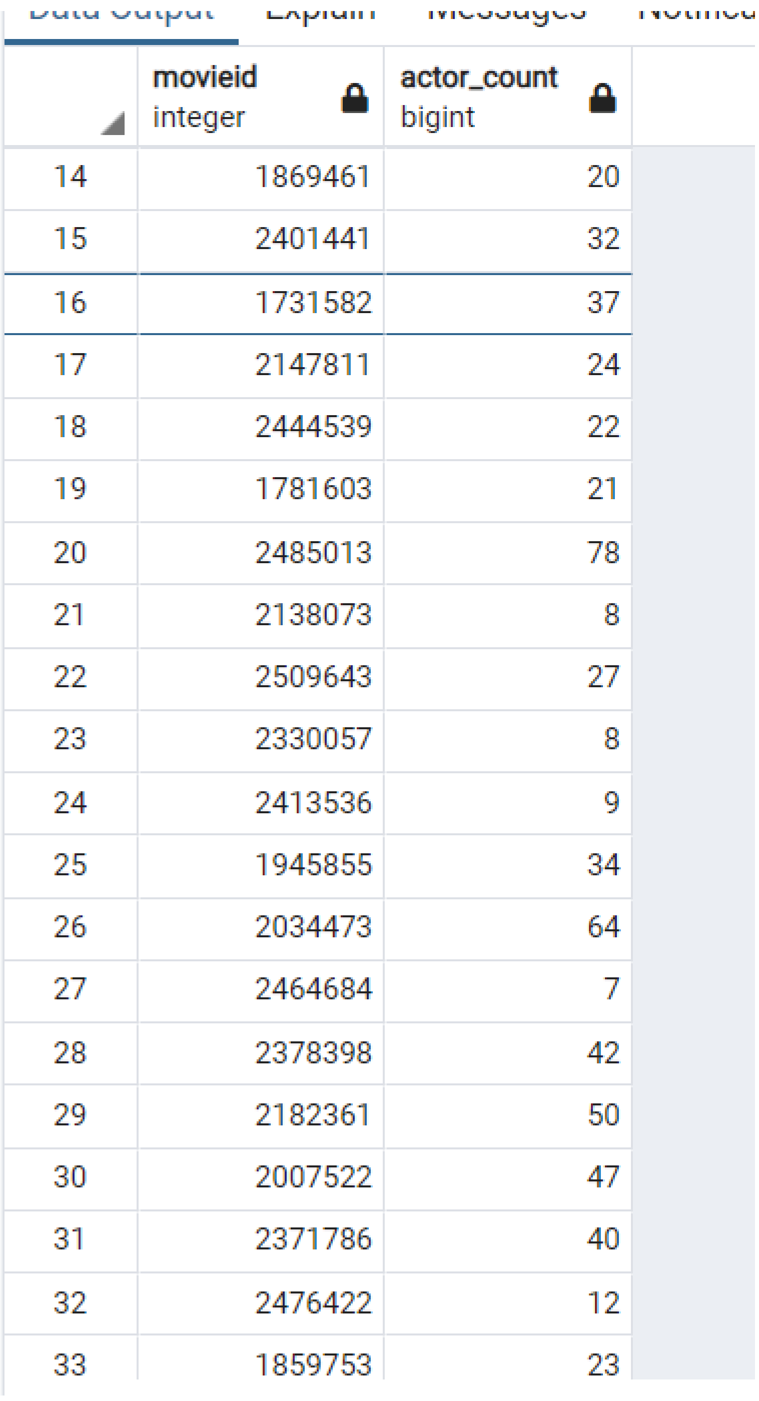
```
   average_rating
   numeric
1  3.5822745164175598
```


Query Editor | Query History

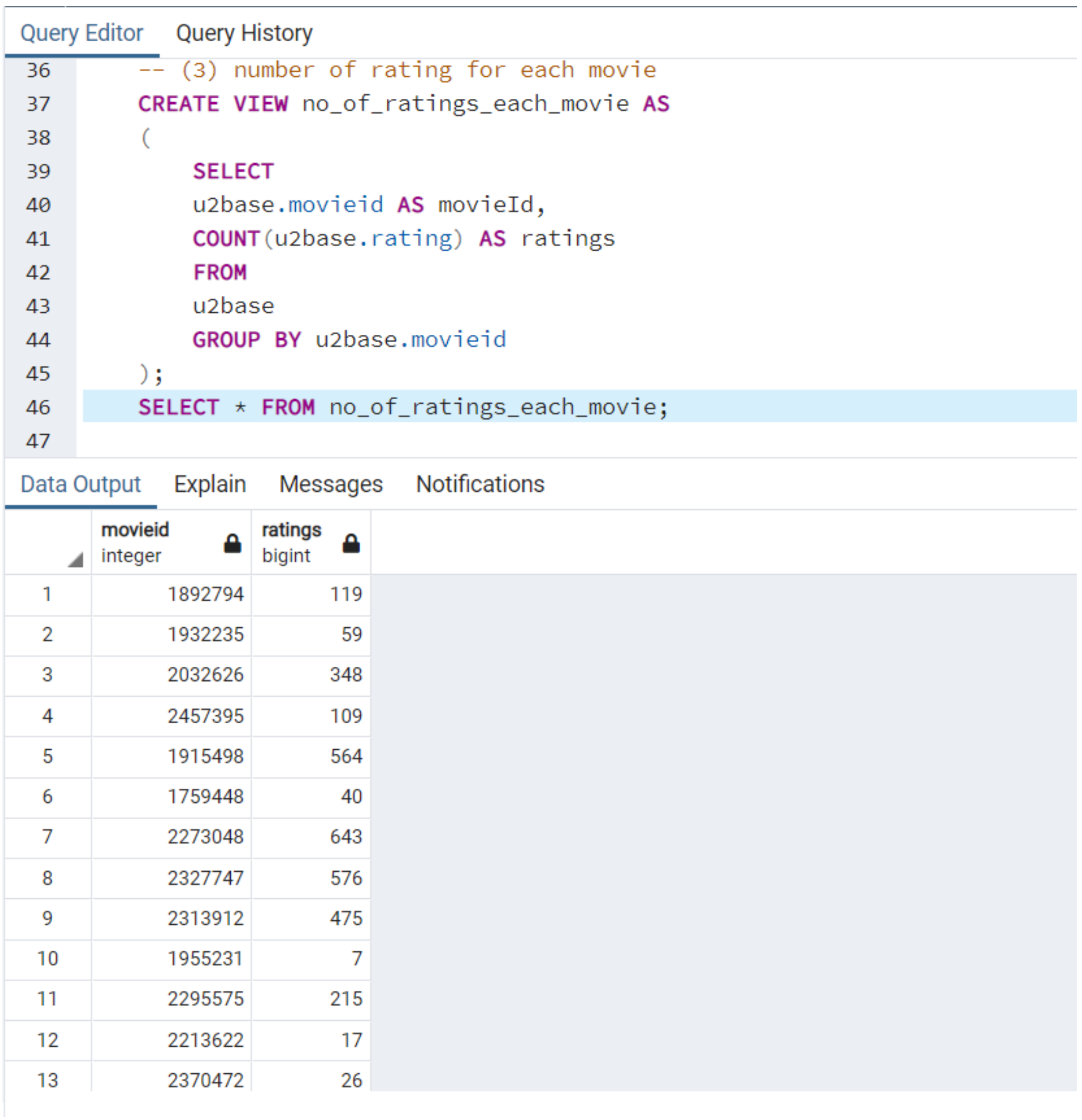
```
-- (2) number of actors in each movie
CREATE VIEW actors_in_each_movie AS
(
    SELECT
    movies2actors.movieid AS movieId,
    COUNT(movies2actors.actorid) AS actor_count
    FROM
    movies2actors
    GROUP BY movies2actors.movieid
);
SELECT * FROM actors_in_each_movie;
```

Data Output | Explain | Messages | Notifications

	movieid 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
12	2213622	44
13	2370472	13

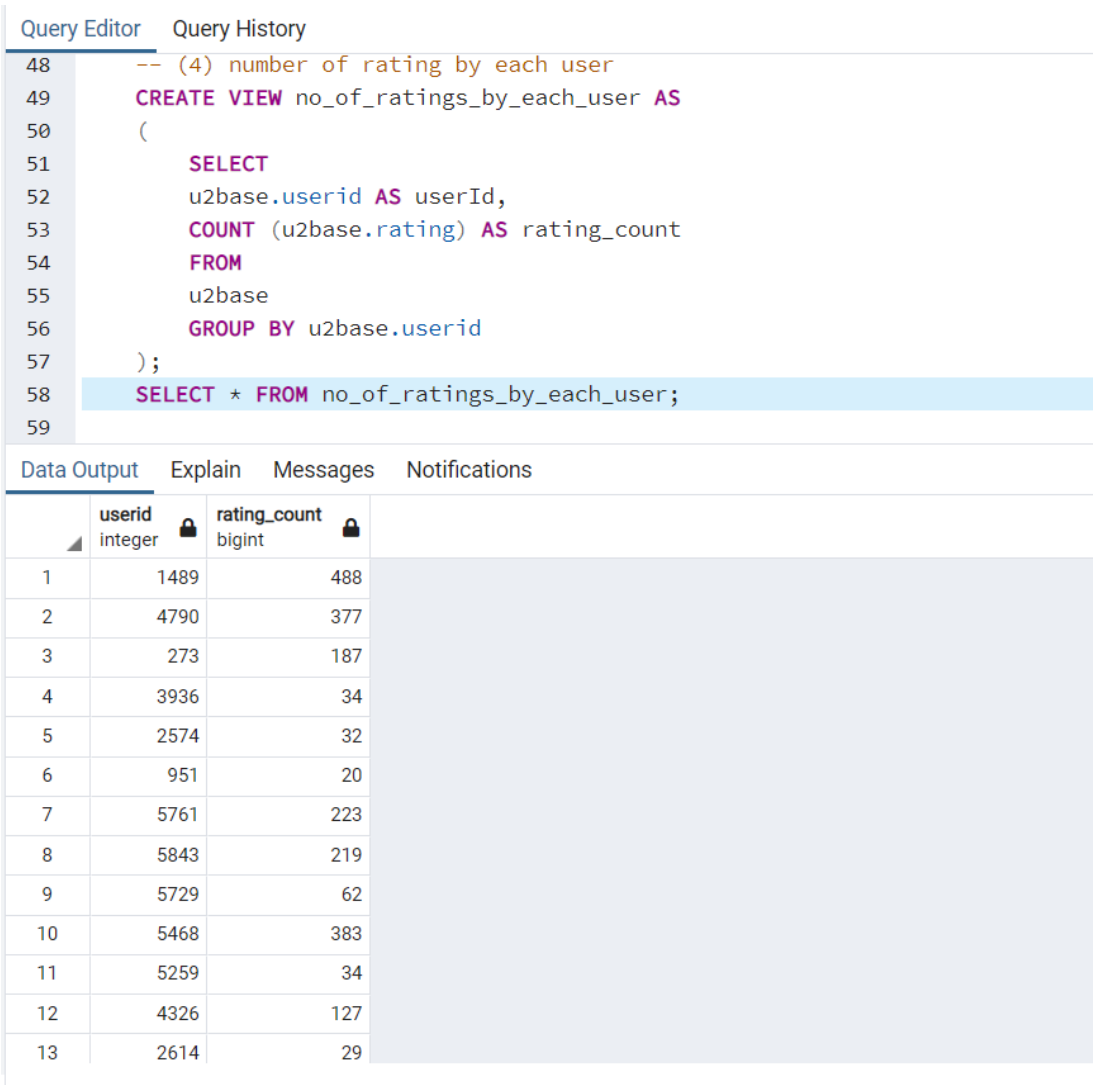


	movieid integer	actor_count bigint
14	1869461	20
15	2401441	32
16	1731582	37
17	2147811	24
18	2444539	22
19	1781603	21
20	2485013	78
21	2138073	8
22	2509643	27
23	2330057	8
24	2413536	9
25	1945855	34
26	2034473	64
27	2464684	7
28	2378398	42
29	2182361	50
30	2007522	47
31	2371786	40
32	2476422	12
33	1859753	23



```
-- (3) number of rating for each movie
CREATE VIEW no_of_ratings_each_movie AS
(
    SELECT
    u2base.movieid AS movieId,
    COUNT(u2base.rating) AS ratings
    FROM
    u2base
    GROUP BY u2base.movieid
);
SELECT * FROM no_of_ratings_each_movie;
```

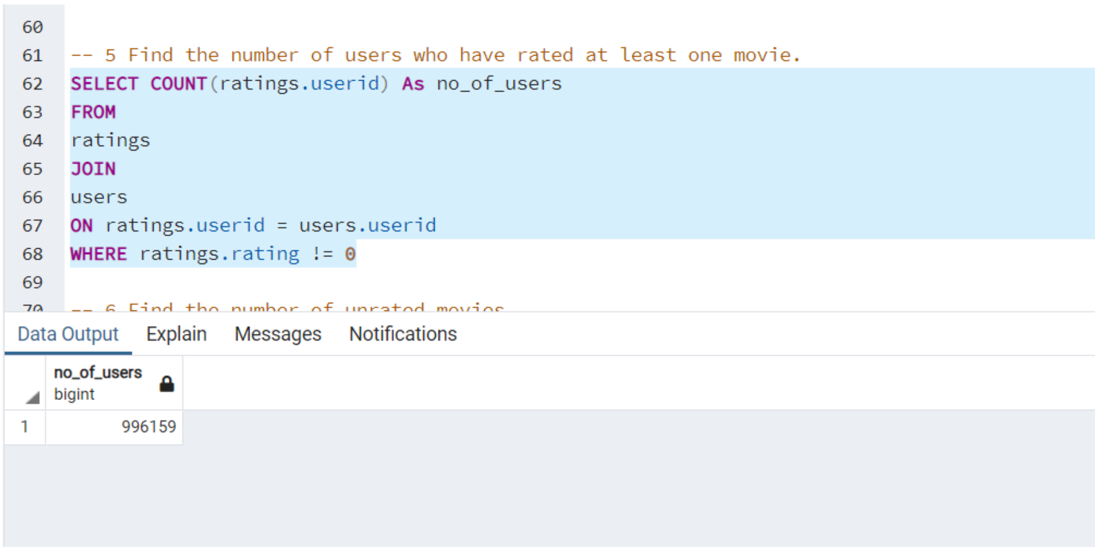
	movieid integer	ratings bigint
1	1892794	119
2	1932235	59
3	2032626	348
4	2457395	109
5	1915498	564
6	1759448	40
7	2273048	643
8	2327747	576
9	2313912	475
10	1955231	7
11	2295575	215
12	2213622	17
13	2370472	26


Query Editor Query History

```
-- (4) number of rating by each user
CREATE VIEW no_of_ratings_by_each_user AS
(
    SELECT
    u2base.userid AS userId,
    COUNT (u2base.rating) AS rating_count
    FROM
    u2base
    GROUP BY u2base.userid
);
SELECT * FROM no_of_ratings_by_each_user;
```

Data Output Explain Messages Notifications

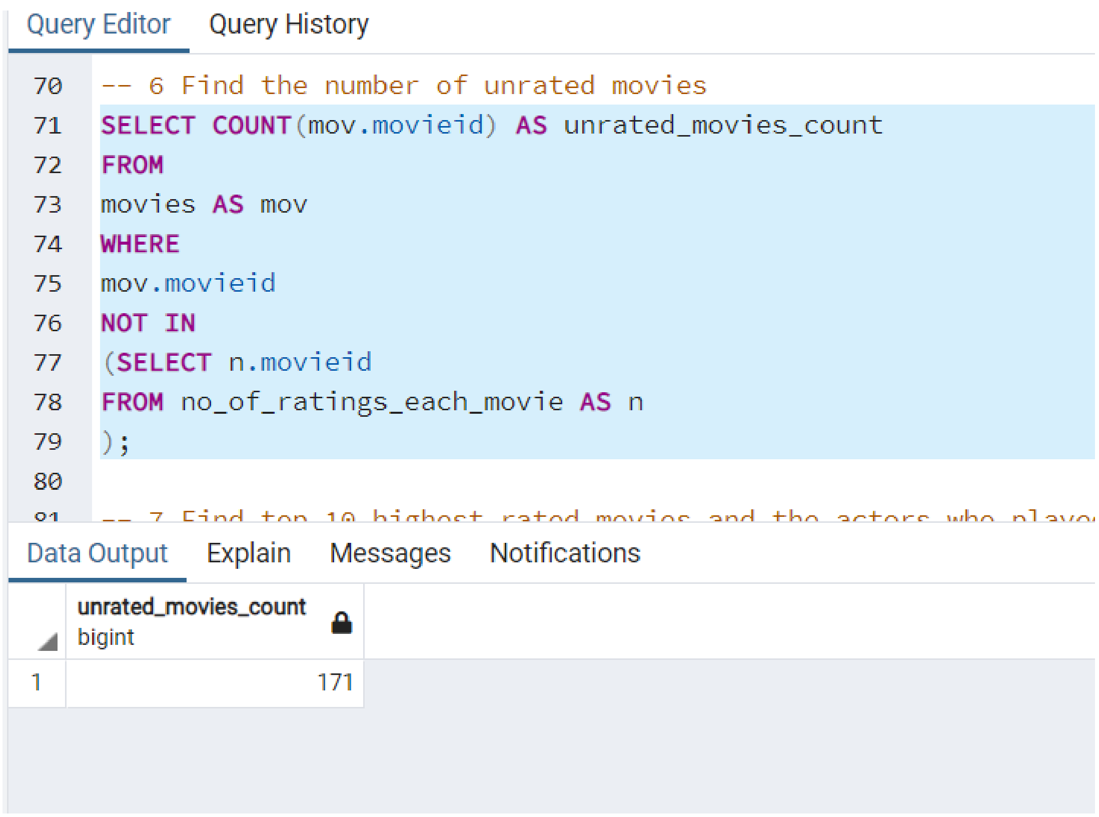
	userid integer	rating_count bigint
1	1489	488
2	4790	377
3	273	187
4	3936	34
5	2574	32
6	951	20
7	5761	223
8	5843	219
9	5729	62
10	5468	383
11	5259	34
12	4326	127
13	2614	29



```
-- 5 Find the number of users who have rated at least one movie.
SELECT COUNT(ratings.userid) As no_of_users
FROM
ratings
JOIN
users
ON ratings.userid = users.userid
WHERE ratings.rating != 0
```

Data Output Explain Messages Notifications

```
  no_of_users
  bigint
1      996159
```

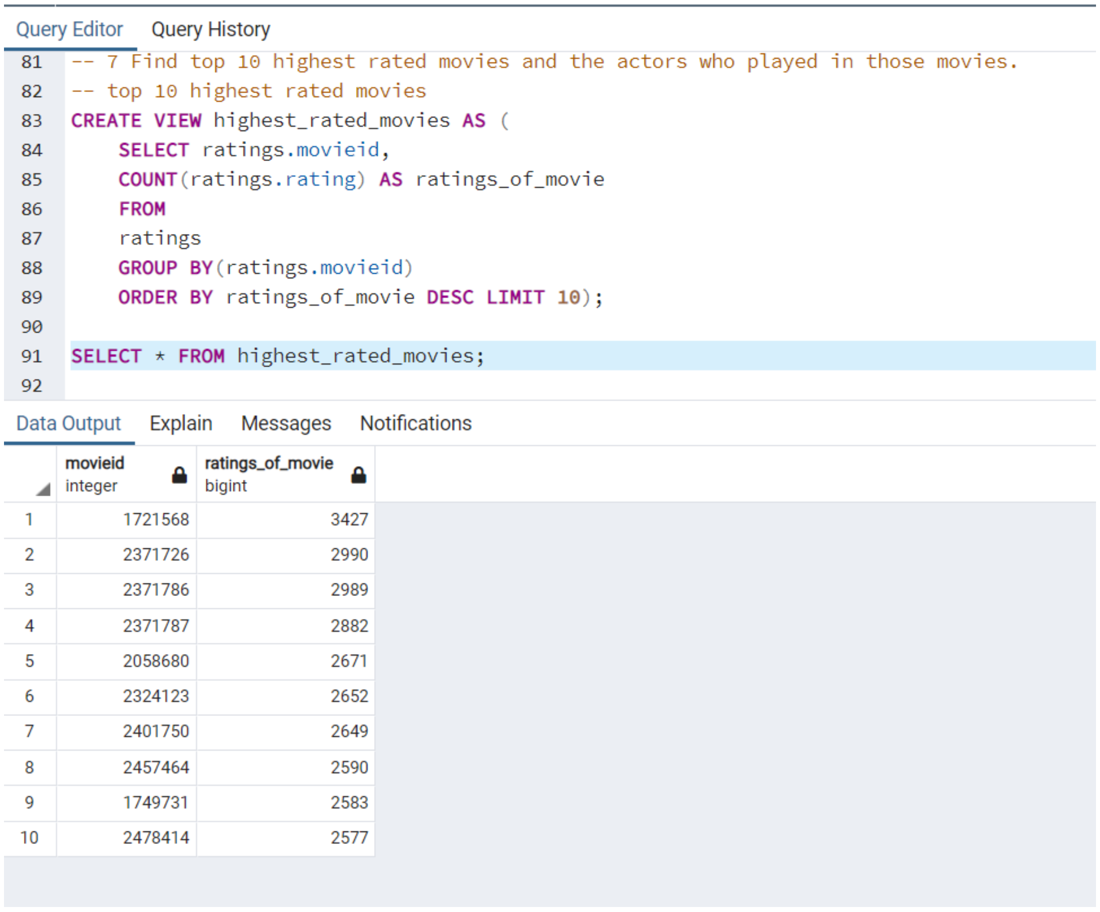


Query Editor Query History

```
-- 6 Find the number of unrated movies
SELECT COUNT(mov.movieid) AS unrated_movies_count
FROM
movies AS mov
WHERE
mov.movieid
NOT IN
(SELECT n.movieid
FROM no_of_ratings_each_movie AS n
);
```

Data Output Explain Messages Notifications

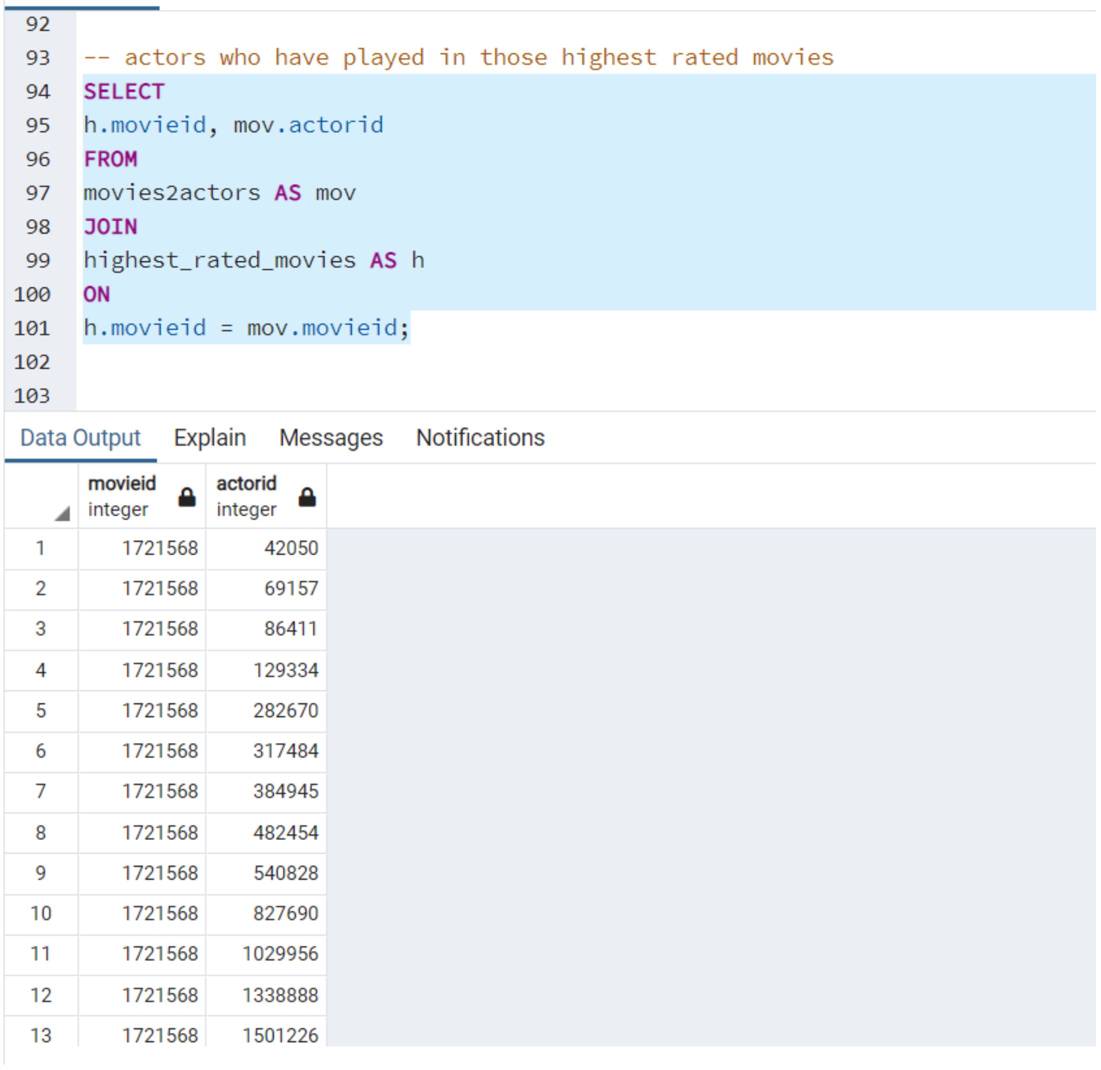
```
  unrated_movies_count
  bigint
1                  171
```



```
-- 7 Find top 10 highest rated movies and the actors who played in those movies.
-- top 10 highest rated movies
CREATE VIEW highest_rated_movies AS (
    SELECT ratings.movieid,
    COUNT(ratings.rating) AS ratings_of_movie
    FROM
    ratings
    GROUP BY(ratings.movieid)
    ORDER BY ratings_of_movie DESC LIMIT 10);

SELECT * FROM highest_rated_movies;
```

	movieid integer	ratings_of_movie 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



```sql

-- actors who have played in those highest rated movies
SELECT
h.movieid, mov.actorid
FROM
movies2actors AS mov
JOIN
highest_rated_movies AS h
ON
h.movieid = mov.movieid;


```

Data Output Explain Messages Notifications

	movieid integer	actorid integer
1	1721568	42050
2	1721568	69157
3	1721568	86411
4	1721568	129334
5	1721568	282670
6	1721568	317484
7	1721568	384945
8	1721568	482454
9	1721568	540828
10	1721568	827690
11	1721568	1029956
12	1721568	1338888
13	1721568	1501226