CSCI-5408

DATA MANAGEMENT, WAREHOUSING, & ANALYTICS

LAB ASSIGNMENT - 1

Banner ID: B00977669

GitLab Assignment Link:
https://git.cs.dal.ca/saji/csci5408_w24_b00977669_christin_saji

Table of Contents

Problem Statement 1

Check how many unique **actors** are present in IMDB dataset.

Query:

```
SELECT COUNT(id) AS unique_actor_count

FROM imdb.actors;
```

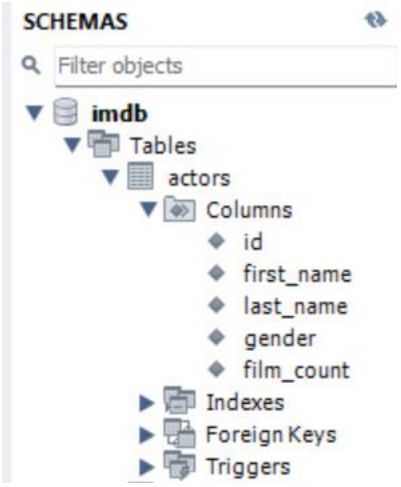

Figure 1 IMDB Schema: Columns of actors Table

Step 1: To obtain the count of unique actors, one can simply count the "id" which serves as the primary key in the "actors" table, even when there are multiple actors with the same name.

Output:

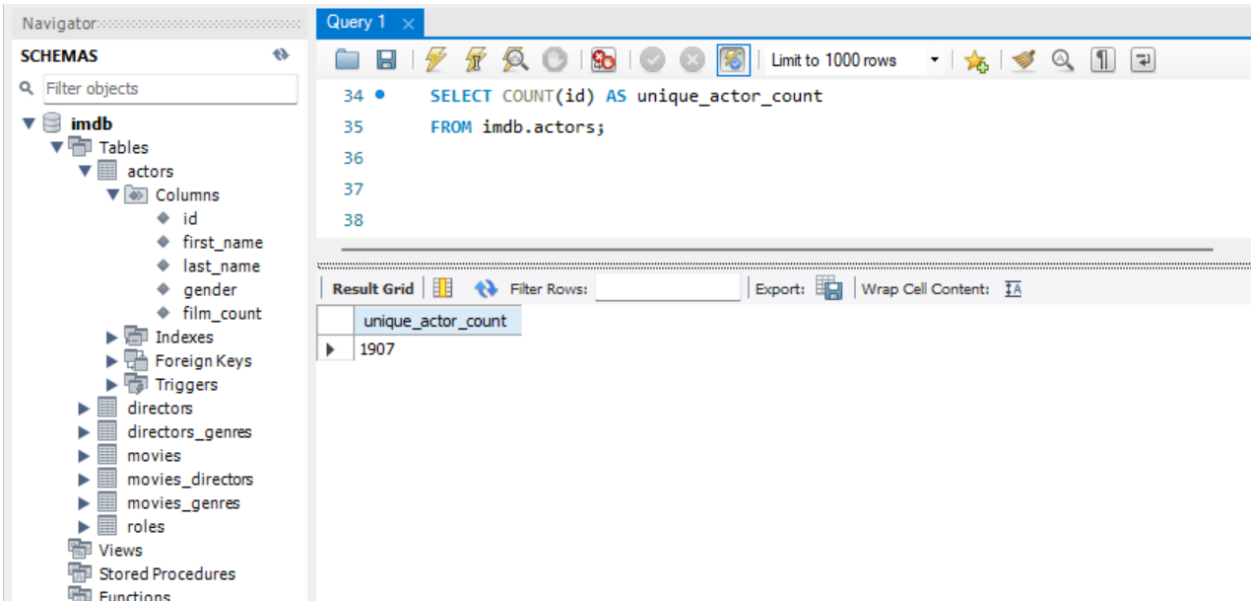

Figure 2 Problem Statement 1 Query with its output

Problem Statement 2

Check how many **movies** are released between the year **1990**s till **2000**.

Query:

```
SELECT COUNT(*) AS movie_count_between_1999_till_2000
FROM imdb.movies
WHERE year BETWEEN 1999 AND 2000;
```

Figure 3 IMDB Schema: Columns of movies Table

Step 1: Using "COUNT(*)", we can determine the number of rows in a table.

Step 2: By incorporating the "WHERE" clause, we can retrieve only the desired rows, i.e., those between the 1990s and 2000.

Output:

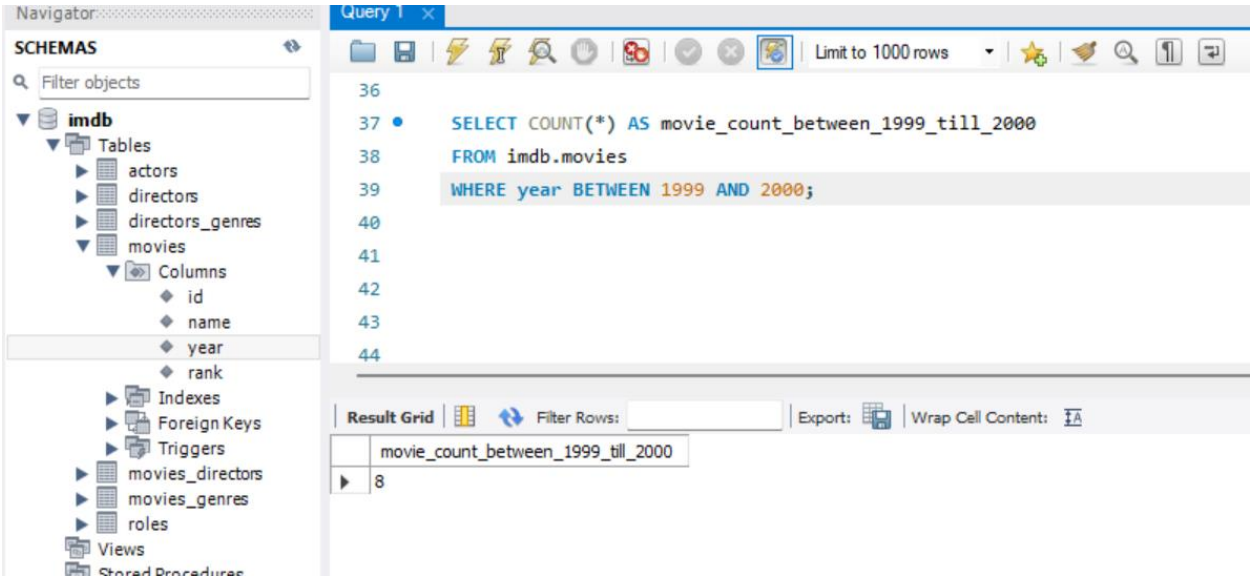

Figure 4 Problem Statement 2 Query with its output

Problem Statement 3

Find the list of **genres** of movies directed by **Christopher Nolan**.

Query:

```
SELECT DISTINCT genre
FROM imdb.directors_genres dg
JOIN imdb.directors d ON dg.director_id = d.id
WHERE d.first_name = 'Christopher' AND d.last_name = 'Nolan';
```

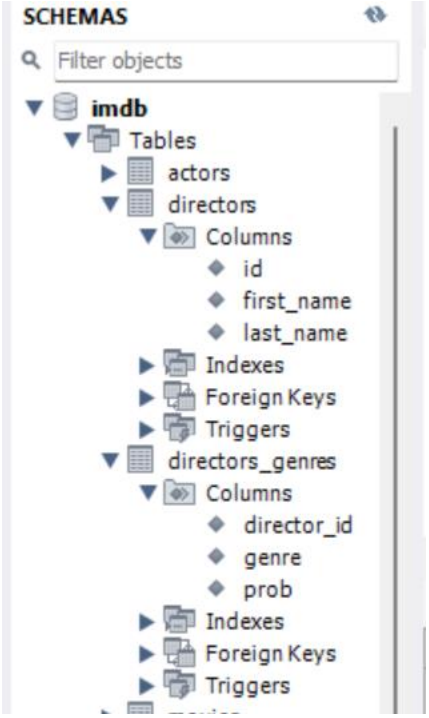

Figure 5 IMDB Schema: Columns of directors and directors_genres Table

Step 1: To obtain the list of genres for movies directed by Christopher Nolan, we need to join the two tables, “directors” and “directors_genres”. The “directors” table contains details about the director’s name, while the genre details are stored in the “directors_genres” table.

Step 2: Using the “WHERE” clause, we can retrieve only the rows directed by “Christopher Nolan”. Employing “DISTINCT” helps avoid any duplicate genres that may be present in the database.

Output:

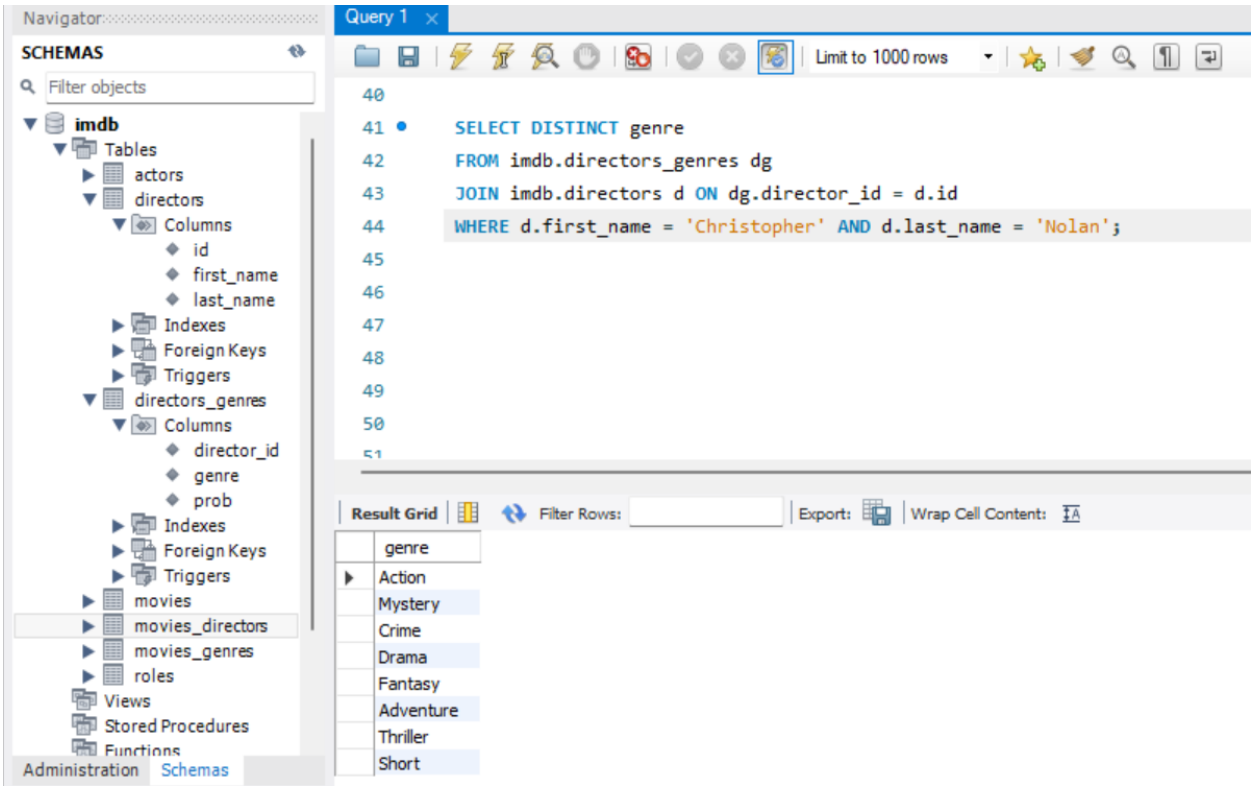

Figure 6 Problem Statement 3 Query with its output

Problem Statement 4

Find the list of all **directors**, and the **movie name** which are **ranked** between **8 to 9** and have a genre of **Sci-Fi** and **Action**.

Query:

```
SELECT d.first_name, d.last_name, m.name AS movie_name

FROM imdb.directors d

JOIN imdb.movies_directors md ON d.id = md.director_id

JOIN imdb.movies m ON md.movie_id = m.id

JOIN imdb.movies_genres mg ON m.id = mg.movie_id

WHERE m.rank BETWEEN 8 AND 9

        AND mg.genre IN ('Sci-fi', 'Action')

GROUP BY d.first_name, d.last_name, movie_name

HAVING COUNT(mg.genre) = 2;
```

Figure 7 IMDB Schema: Columns of directors, movies, movies_director and movies_genres Table

Step 1: By joining the tables “directors”, “movies_directors”, “movies”, and “movies_genres”, we can retrieve information about directors, movies, ratings, and genres.

Step 2: Filtering the results based on rating and genre range will list all the movies with either “Sci-Fi” or “Action” genre.

Step 3: Grouping the result by director's name and movie name, we can then count the genres that occur together. If the count is 2, it indicates that movie has both the "Sci-Fi" and "Action" genres. If the count is 1, it means only one of the genres was present.

Output:

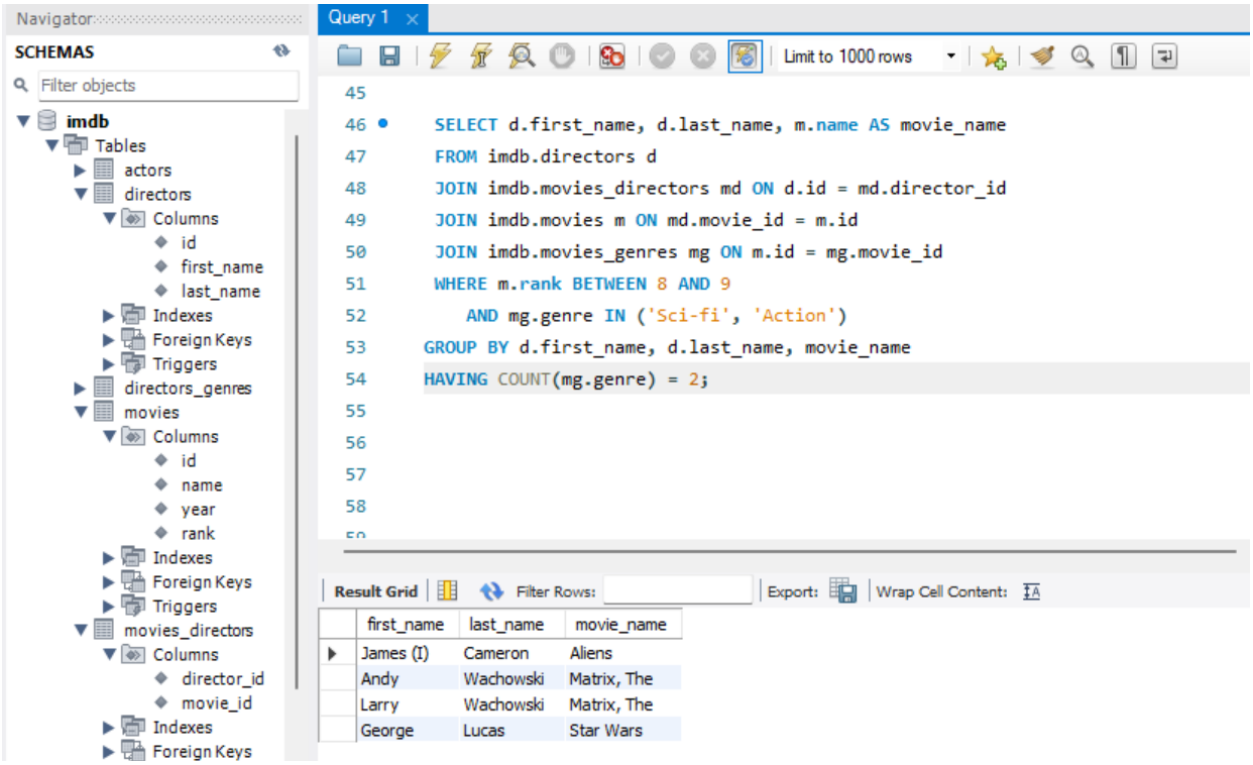

Figure 8 Problem Statement 4 Query with its output

Problem Statement 5

Find the **name of the movie** in which the actor's **role is any doctor**, and the movie has the **highest number of roles of doctor**.

Query:

```
SELECT m.name AS movie_name

FROM imdb.roles r

JOIN imdb.movies m ON r.movie_id = m.id

WHERE r.role LIKE 'Dr.%' OR r.role LIKE '%doctor%'

GROUP BY m.id, movie_name

ORDER BY COUNT(*) DESC

LIMIT 1;
```

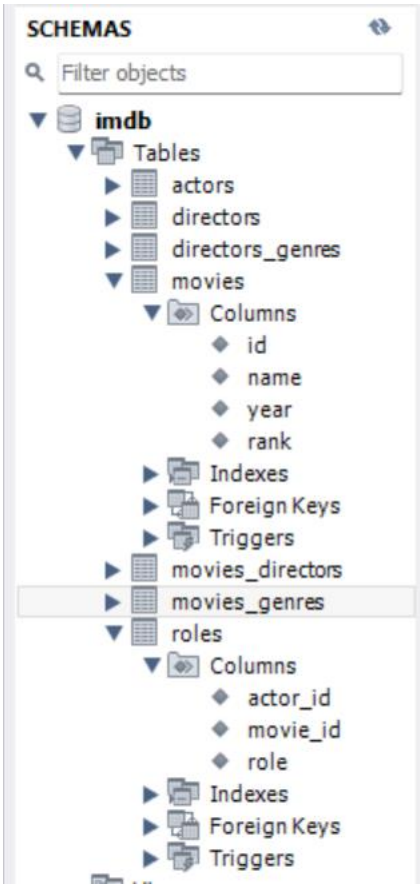

Figure 9 IMDB Schema: Columns of movies and roles Table

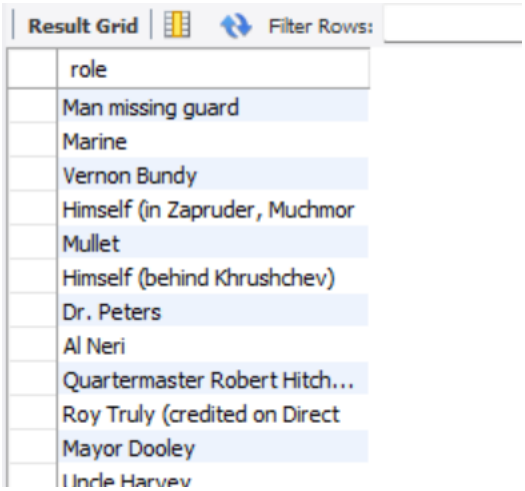

Figure 10 Output of role data

Step 1: Since doctors in the role database are mentioned as “Dr.”, we need to filter roles that start with “Dr.”

Step 2: We join the “roles” with the “movies” to obtain movie and role details.

Step 3: Using the “LIKE” operator, we can retrieve role names that start with “Dr.”, and in case any role contains the term “doctor”, we can add “%doctor%” for additional coverage.

Step 4: We then group the results by movie ID and movie name, ordering the groups in descending order based on the count of roles (doctors) in each movie. This ensures that the top result will have the highest number of doctor roles.

Step 5: Finally, by using “LIMIT 1”, we retrieve only the movie with the highest count.

Output:

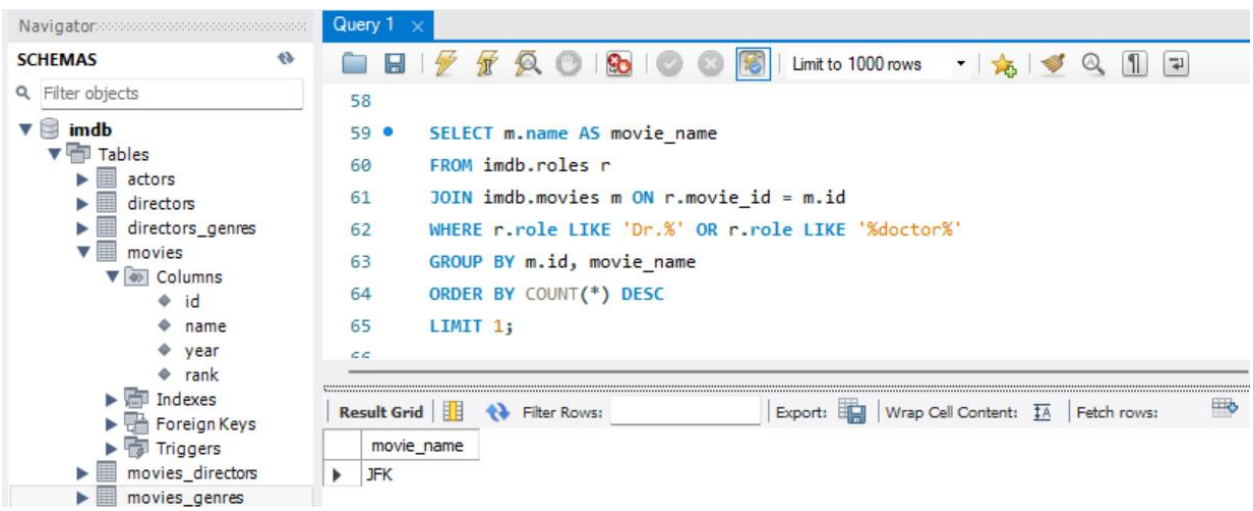

Figure 11 Problem Statement 5 Query with its output

Problem Statement 6

Find the list of the movies that start with the letter **'f'**.

Query:

```
SELECT name AS movie_name
FROM imdb.movies
WHERE name LIKE 'F%';
```

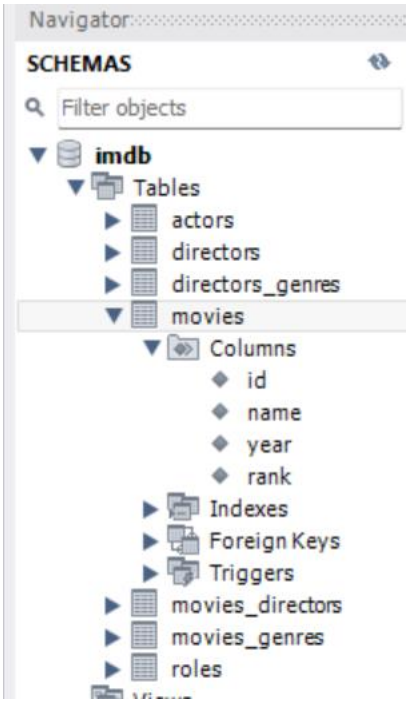

Figure 12 IMDB Schema: Columns of movies Table

Step 1: Simply by checking the names that start with "F" using the "LIKE" operator in the movies table.

Output:

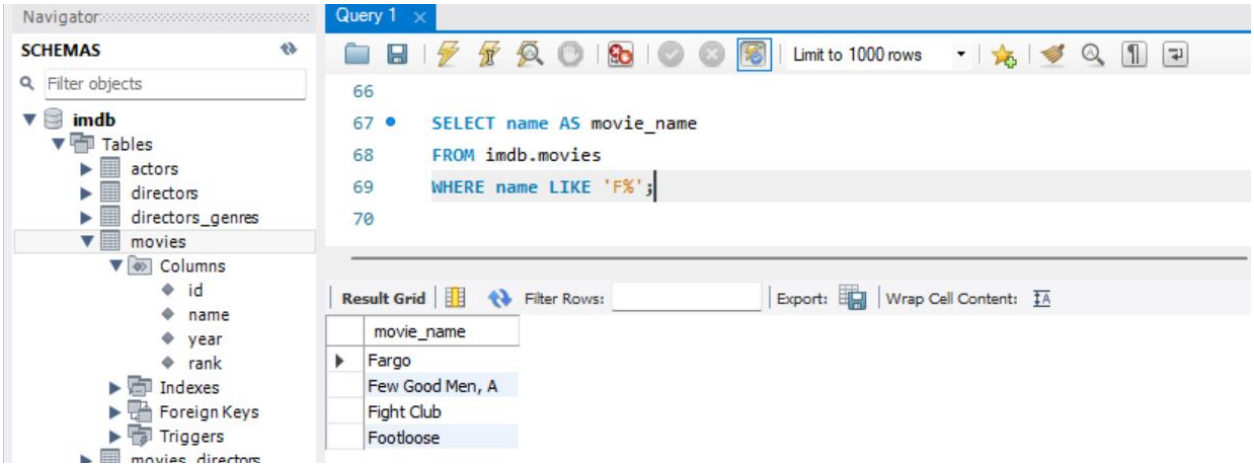

Figure 13 Problem Statement 6 Query with its output