

# CSCI 5408

## DATA MANAGEMENT AND WAREHOUSING



## LAB ASSIGNMENT - 1

**Submitted By:** Kenil Shaileshkumar Patel  
(kenil.patel@dal.ca)  
**Banner ID:** B00954251  
**Submitted On:** September 20, 2023

### Gitlab Repository Link

[https://git.cs.dal.ca/kenil/csci5408\\_f23\\_b00954251\\_kenil\\_patel/-/tree/main/Lab1](https://git.cs.dal.ca/kenil/csci5408_f23_b00954251_kenil_patel/-/tree/main/Lab1)

## Table of Contents

Sr. No	Title	Page No.
1.	Query - 1	3
2.	Query - 2	4
3.	Query - 3	5
4.	Query - 4	6
5.	Query - 5	7
6.	Query - 6	8

## Query – 1

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

```
SELECT COUNT(DISTINCT CONCAT(first_name, ' ', last_name)) AS  
Total_No_Of_Unique_Actors FROM lab_1.actors;
```

In this query, first name and last name of actors are concatenated. Then it counts the distinct concatenated names and we get the Total number of unique actors from the actor's table. Total\_No\_Of\_Unique\_Actors is an alias to label the result of this count for better readability.

### Solution:-

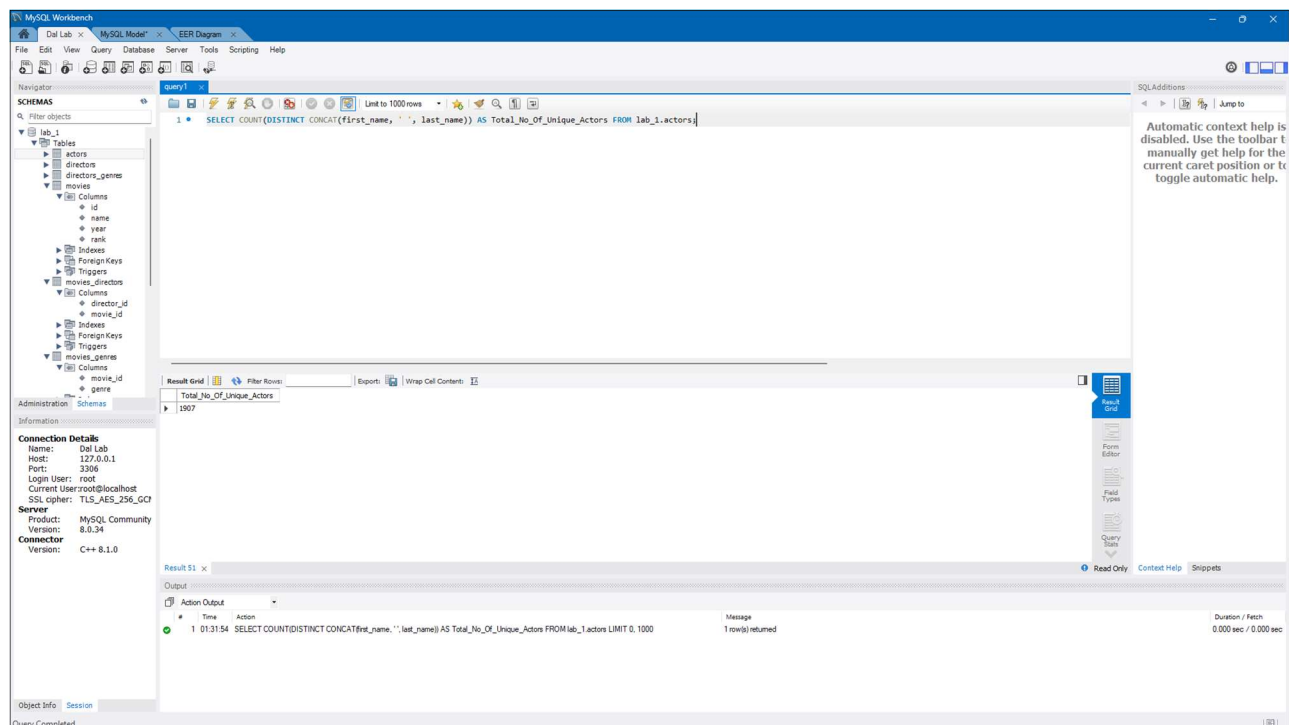


Figure 1 Output of Query-1

## Query – 2

Check how many movies were released between the years 1990s and 2000.

```
SELECT COUNT(*) AS Count_of_movies_bt看_1990_to_2000 FROM lab_1.movies WHERE year BETWEEN 1990 AND 2000;
```

In this query, it counts the total no of movies that were released in a year between 1990 and 2000. And alias to the count is Count\_of\_movies\_bt看\_1990\_to\_2000 for better presentation.

**Solution:-**

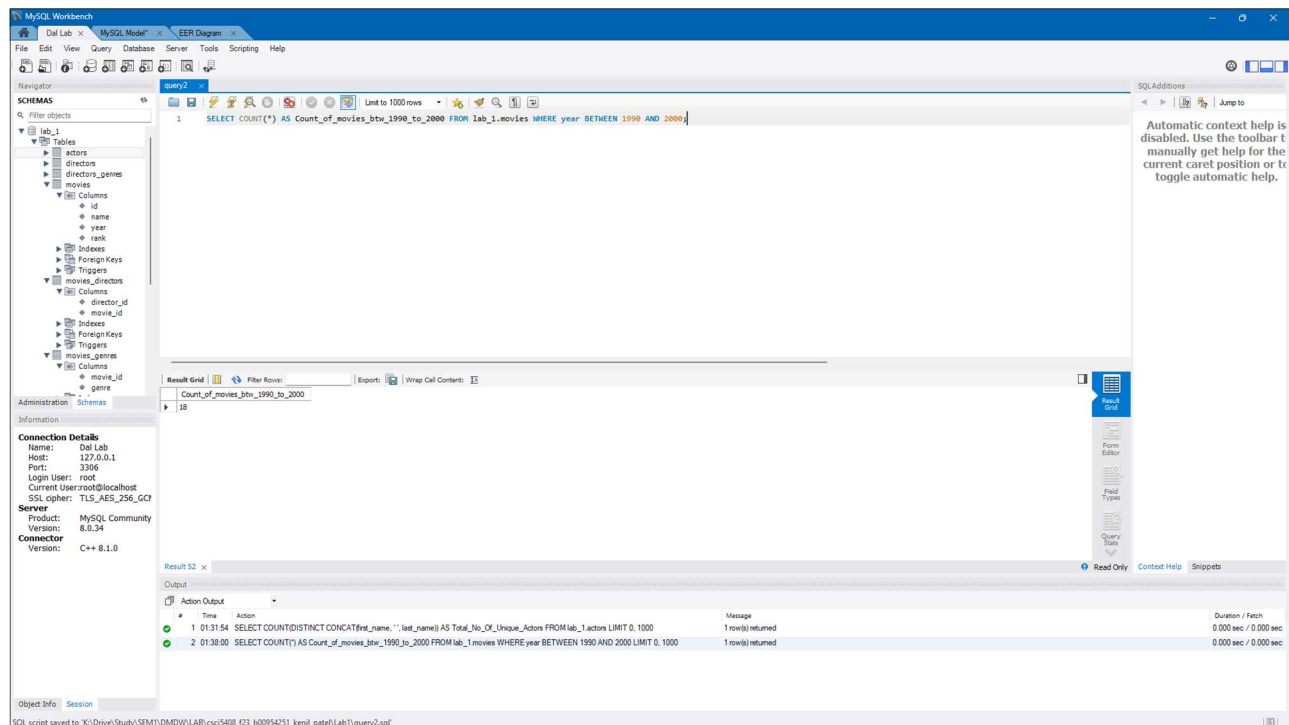


Figure 2 Output of Query-2

## Query – 3

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

```
SELECT DISTINCT lab_1.directors_genres.genre FROM lab_1.directors
RIGHT JOIN lab_1.directors_genres ON lab_1.directors.id = lab_1.directors_genres.director_id
WHERE CONCAT(lab_1.directors.first_name, ' ', lab_1.directors.last_name) = "Christopher
Nolan";
```

This query extracts the distinct genres which are associated with the director "Christopher Nolan". I have done the right join on directors and directors\_genres based on directors IDs and filtering results on the basis of specific directors.

### Solution:-

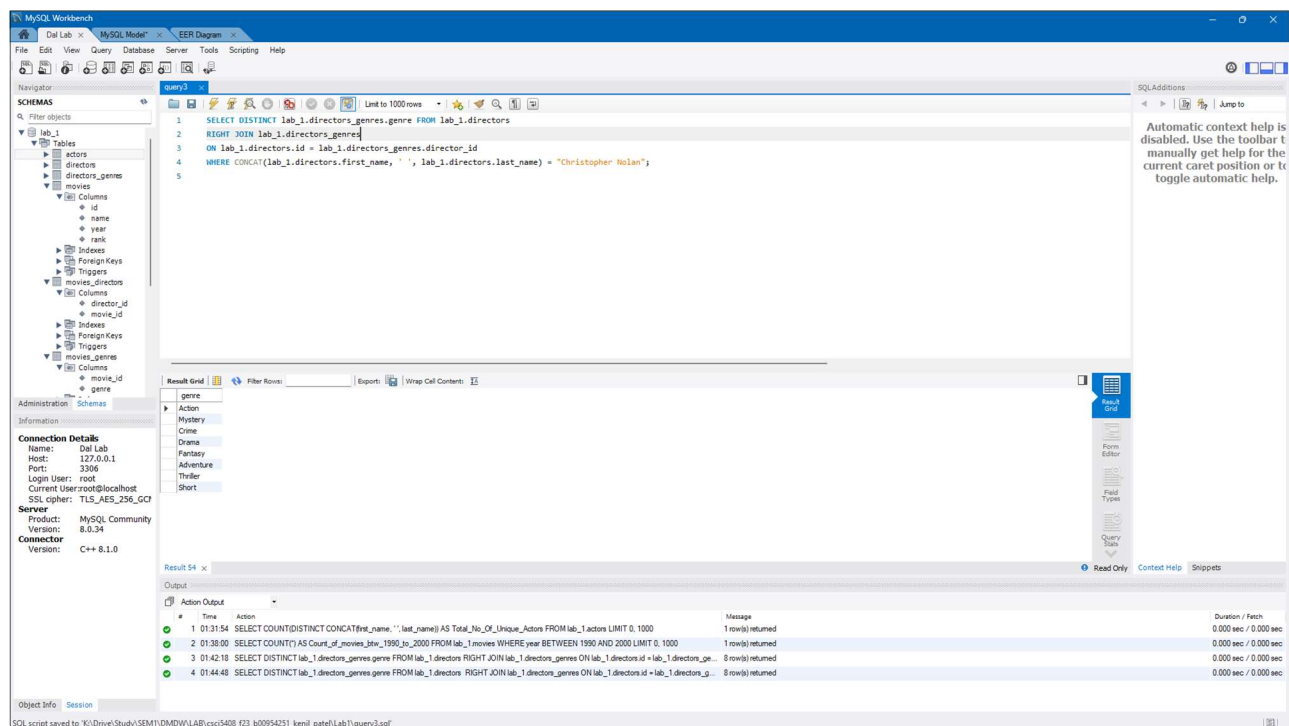


Figure 3 Output of Query-3

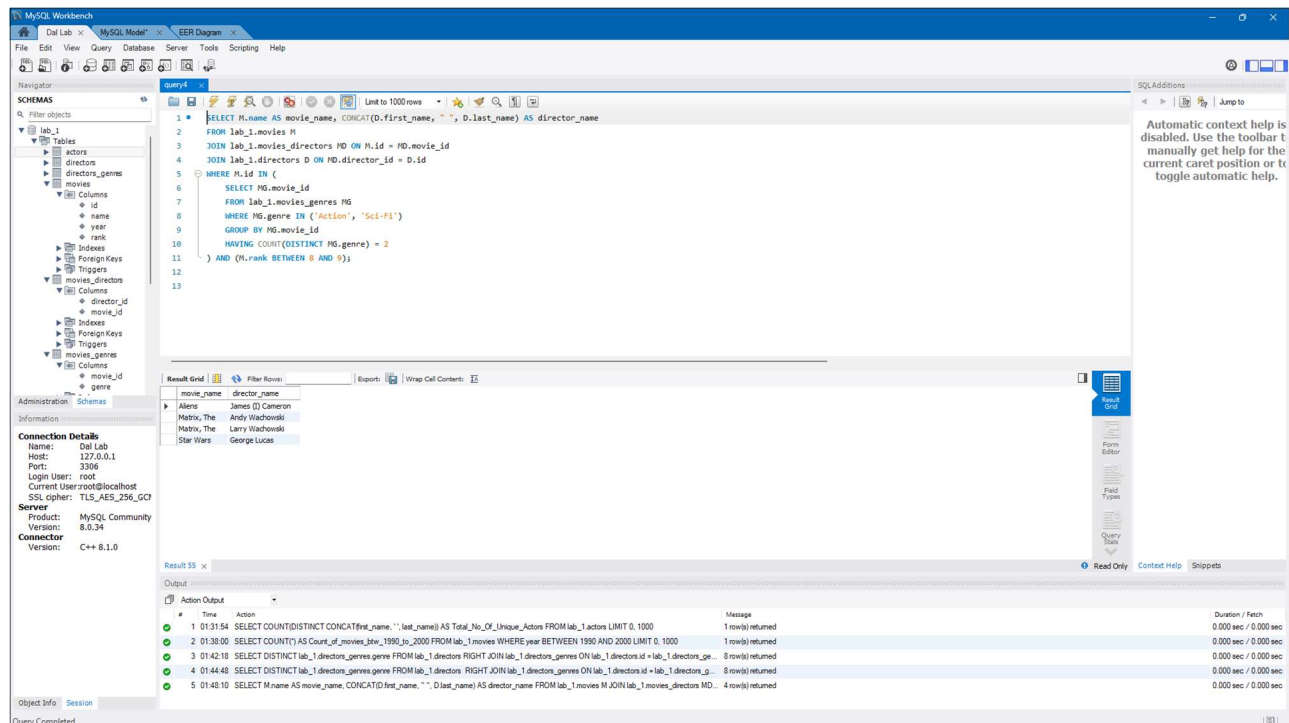
## Query – 4

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.

```
SELECT M.name AS movie_name, CONCAT(D.first_name, " ", D.last_name) AS director_name
FROM lab_1.movies M
JOIN lab_1.movies_directors MD ON M.id = MD.movie_id
JOIN lab_1.directors D ON MD.director_id = D.id
WHERE M.id IN (
    SELECT MG.movie_id
    FROM lab_1.movies_genres MG
    WHERE MG.genre IN ('Action', 'Sci-Fi')
    GROUP BY MG.movie_id
    HAVING COUNT(DISTINCT MG.genre) = 2
) AND (M.rank BETWEEN 8 AND 9);
```

First, in the sub-query, extract all movie\_id from the movies\_genres table which has action and sci-fi genre both in a movie. After, getting movie\_id, I joined on movies\_directors table and got the directors\_id and another joined the director's table to get the director's name.

### Solution:-



The screenshot shows the MySQL Workbench interface. The central pane displays the following SQL query:

```
1 SELECT M.name AS movie_name, CONCAT(D.first_name, " ", D.last_name) AS director_name
2 FROM lab_1.movies M
3 JOIN lab_1.movies_directors MD ON M.id = MD.movie_id
4 JOIN lab_1.directors D ON MD.director_id = D.id
5 WHERE M.id IN (
6     SELECT MG.movie_id
7     FROM lab_1.movies_genres MG
8     WHERE MG.genre IN ('Action', 'Sci-Fi')
9     GROUP BY MG.movie_id
10    HAVING COUNT(DISTINCT MG.genre) = 2
11 ) AND (M.rank BETWEEN 8 AND 9);
```

The bottom pane shows the results of the query in a table with two columns: 'movie\_name' and 'director\_name'. The results are as follows:

movie_name	director_name
Aliens	James (J) Cameron
Matrix, The	Andy Wachowski
Matrix, The	Larry Wachowski
Star Wars	George Lucas

Figure 4 Output of Query-4

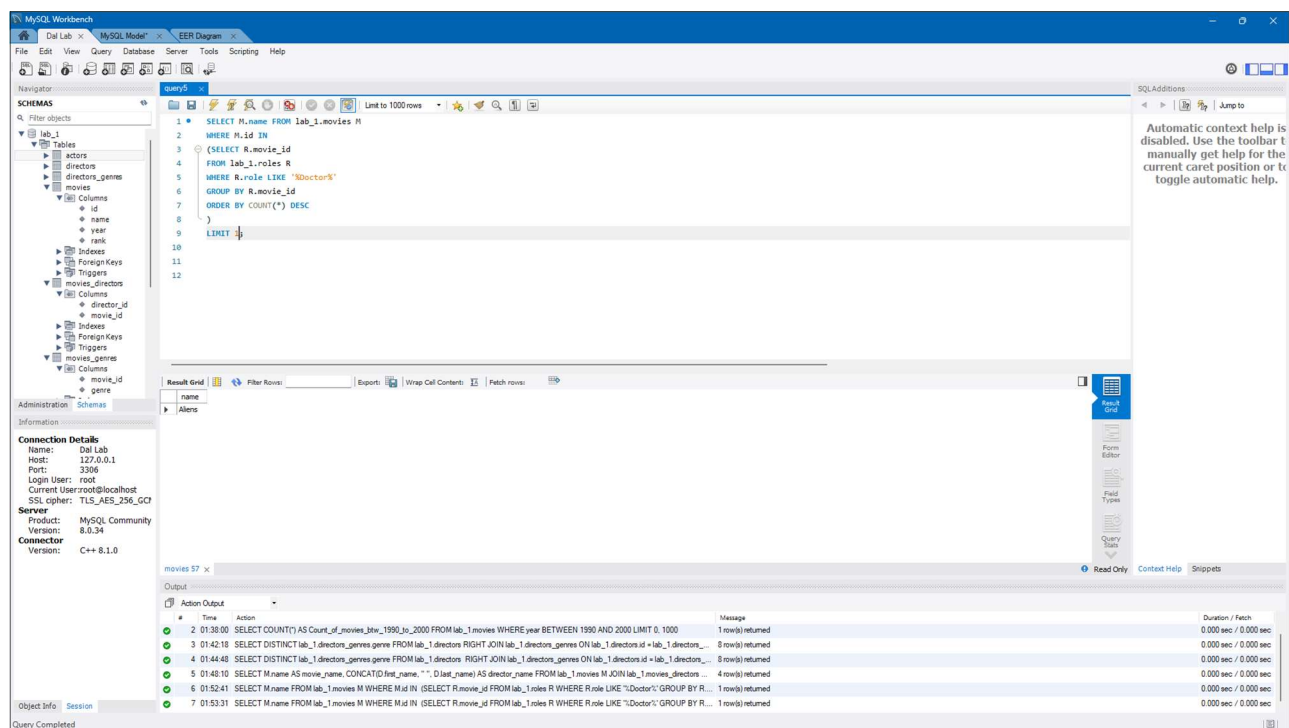
## Query – 5

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 a doctor.

```
SELECT M.name FROM lab_1.movies M
WHERE M.id IN
(SELECT R.movie_id
FROM lab_1.roles R
WHERE R.role LIKE '%Doctor%'
GROUP BY R.movie_id
ORDER BY COUNT(*) DESC
)
LIMIT 1;
```

This query retrieves the movie name for the movie with the highest count of characters that have doctors' roles. And subquery counts the total doctor roles in a movie and returns a movie\_id. The outer query selects the movie name from the top row which has the highest count.

### Solution:-



The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1 SELECT M.name FROM lab_1.movies M
2 WHERE M.id IN
3 (SELECT R.movie_id
4 FROM lab_1.roles R
5 WHERE R.role LIKE '%Doctor%'
6 GROUP BY R.movie_id
7 ORDER BY COUNT(*) DESC
8 )
9
10 LIMIT 1;
```

The Results grid at the bottom shows the output of the query. The first column is 'name' and the second column is 'movie\_id'. The results are as follows:

name	movie_id
Avatar	1

The output also includes a detailed log of the query execution, showing the time taken for each step and the number of rows returned.

Figure 5 Output of Query-5

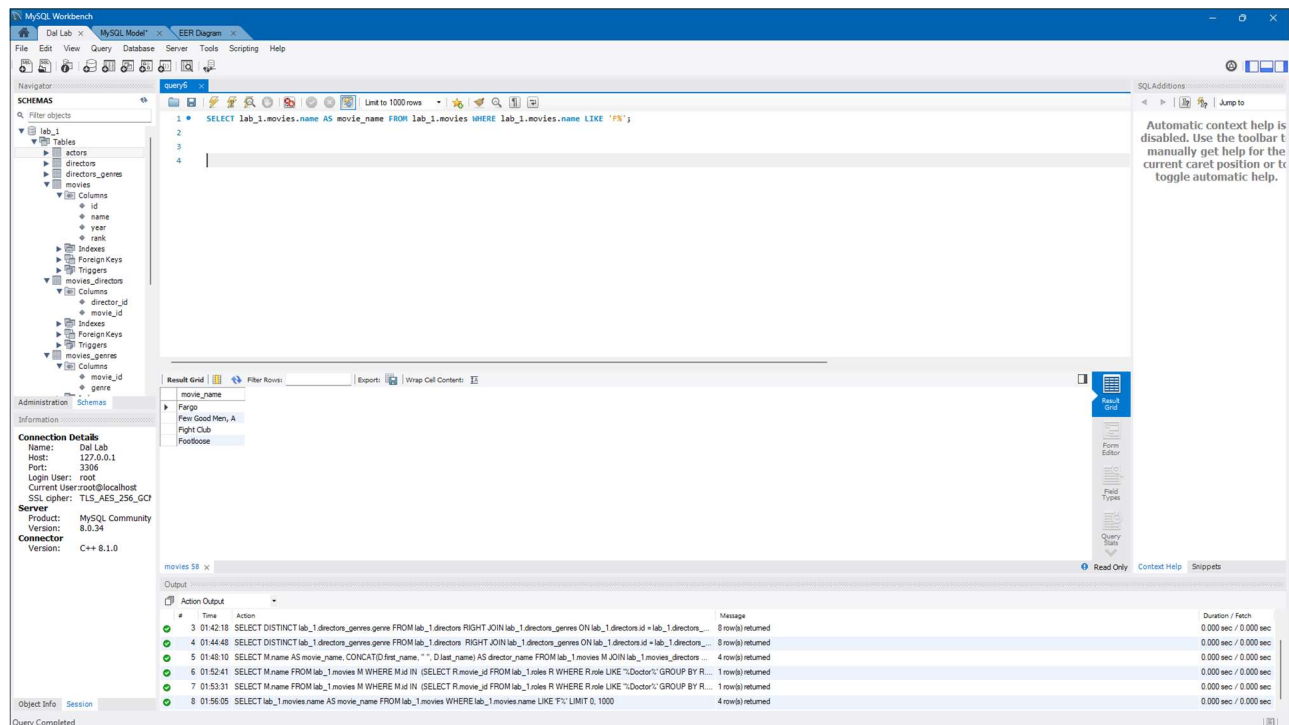
## Query – 6

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

```
SELECT lab_1.movies.name AS movie_name FROM lab_1.movies WHERE lab_1.movies.name LIKE 'F%';
```

This query finds the names of the movies that start with F. So in LIKE it checks that the first letter is capital F and the rest can be anything.

**Solution:-**



The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL query:

```
1 SELECT lab_1.movies.name AS movie_name FROM lab_1.movies WHERE lab_1.movies.name LIKE 'F%';
```

The left sidebar shows the Schemas tree with the following structure:

- lab\_1
  - actors
  - directors
  - directors\_genres
  - movies
    - id
    - name
    - year
    - rank
  - Indexes
  - ForeignKeys
  - Triggers
  - movies\_directors
    - director\_id
    - movie\_id
  - Indexes
  - ForeignKeys
  - Triggers
  - movies\_genres
    - movie\_id
    - genre

The bottom pane shows the Output window with the following results:

#	Time	Action	Message	Duration / Fetch
3	01:42:18	SELECT DISTINCT lab_1.directors_genres.genre FROM lab_1.directors RIGHT JOIN lab_1.directors_genres ON lab_1.directors.id = lab_1.directors_genres.director_id	8 row(s) returned	0.000 sec / 0.000 sec
4	01:44:48	SELECT DISTINCT lab_1.directors_genres.genre FROM lab_1.directors RIGHT JOIN lab_1.directors_genres ON lab_1.directors.id = lab_1.directors_genres.director_id	8 row(s) returned	0.000 sec / 0.000 sec
5	01:48:10	SELECT M.name AS movie_name, CONCAT(D.first_name, ' ', D.last_name) AS director_name FROM lab_1.movies M JOIN lab_1.movies_directors M_directors ON M.id = M_directors.movie_id	4 row(s) returned	0.000 sec / 0.000 sec
6	01:52:41	SELECT M.name FROM lab_1.movies M WHERE M.id IN (SELECT R.movie_id FROM lab_1.roles R WHERE R.role LIKE '%Doctor%') GROUP BY M.name	1 row(s) returned	0.000 sec / 0.000 sec
7	01:53:31	SELECT M.name FROM lab_1.movies M WHERE M.id IN (SELECT R.movie_id FROM lab_1.roles R WHERE R.role LIKE '%Doctor%') GROUP BY M.name	1 row(s) returned	0.000 sec / 0.000 sec
8	01:56:05	SELECT lab_1.movies.name AS movie_name FROM lab_1.movies WHERE lab_1.movies.name LIKE 'F%' LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

Figure 6 Output of Query-6