
CSCI 5408

***Data Management, Warehousing, And
Analytics***

Lab 1: Introduction to MySQL

Prepared By

Bhavisha Oza (B00935827)

Summary

The lab-1 is all about getting familiar with MySQL. After refreshing the concepts of it, I did some hands on for the asked queries on given IMDB_dataset. Fetched the results by using DML (Data Manipulation Language) queries with the help of basic SQL functions, operators, sub-queries, and joins. Wanted to use both the ways of retrieving data from multiple related tables, so I've used joins as well as sub-queries.

Steps performed:

- Installed MySQL workbench [1].
- Created new connection for CSCI 5408.
- Created new schema for Lab-1
- Downloaded the IMDB dataset provided for the lab work [2].
- Imported the IMDB.SQL file in MySQL workbench.
- Refreshed the schema and tested the basic Sample select query from movies table.
- Started creating queries for the Lab Exercise [3].
- Executed them, took a screenshot of each output, and added in the report.

Lab exercise:

1. Check how many directors are present in iMDB?

```
SELECT DISTINCT COUNT(*) AS total_directors  
FROM directors;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, with 'lab_1' selected and its tables (actors, directors, directors_genres, movies, movies_directors, movies_genres, roles, Views, Stored Procedures, Functions) listed. The main pane shows the SQL editor with the following code:

```
1 • USE lab_1;  
2  
3 -- 1. Check how many directors are present in iMDB  
4 • SELECT DISTINCT COUNT(*) AS total_directors  
5   FROM directors;  
6
```

The 'Result Grid' pane below shows the result of the query:

total_directors
34

The 'Session' tab at the bottom shows the following log entries:

Action	Time	Response	Duration / Fetch Time
1 16:38:53 USE lab_1		0 row(s) affected	0.00075 sec
2 16:38:57 SELECT DISTINCT COUNT(*) AS total_directors FROM directors LIMIT 0, 1000		1 row(s) returned	0.011 sec / 0.000017...

Figure 1: The result of total directors available in DB.

Explanation: This query retrieves the number of distinct directors from the "directors" table and assigns it to the alias "total_directors."

2. Check how many movies are released post-year 2000?

```
SELECT COUNT(*) AS total_movies  
FROM movies  
WHERE year > 2000;
```

The screenshot shows the Oracle Database SQL Developer interface. The left sidebar displays the schema structure under 'lab_1' (Tables: actors, directors, directors_genres, movies, movies_directors, movies_genres, roles; Views, Stored Procedures, Functions). The main pane shows the SQL editor with the following code:

```
-- 2. Check how many movies are released post-year 2000  
SELECT COUNT(*) AS total_movies  
FROM movies  
WHERE year > 2000;
```

The result grid shows the output:

total_movies
10

The bottom pane shows the history of actions:

Action Output	Time	Action	Response	Duration / Fetch Time
1	16:38:53	USE lab_1	0 row(s) affected	0.00075 sec
2	16:38:57	SELECT DISTINCT COUNT(*) AS total_directors FROM directors LIMIT 0, 1000	1 row(s) returned	0.011 sec / 0.000017...
3	16:46:00	SELECT COUNT(*) AS total_movies FROM movies WHERE year > 2000 LIMIT 0, 1000	1 row(s) returned	0.0048 sec / 0.0000...

Figure 2: The result of total movies released after year 2000.

Explanation: This query retrieves the count the number of movies that were released after the year 2000 and return the result as "total_movies".

3. Find the list of genres of movies directed by Andrew Adamson.

```
SELECT DISTINCT mg.genre  
FROM movies m  
JOIN movies_directors md ON m.id = md.movie_id  
JOIN directors d ON md.director_id = d.id  
JOIN movies_genres mg ON m.id = mg.movie_id  
WHERE d.first_name = 'Andrew' AND d.last_name = 'Adamson';
```

The screenshot shows the Oracle Database SQL Developer interface. The left sidebar displays the schema structure for 'lab_1'. The central pane shows a query editor with the following SQL code:

```

12 -- 3. Find the list of genres of movies directed by Andrew Adamson
13 •   SELECT DISTINCT mg.genre
14   FROM movies m
15   JOIN movies_directors md ON m.id = md.movie_id
16   JOIN directors d ON md.director_id = d.id
17   JOIN movies_genres mg ON m.id = mg.movie_id
18 WHERE d.first_name = 'Andrew' AND d.last_name = 'Adamson';
19
20

```

The result grid shows the output of the query:

genre
Adventure
Animation
Comedy
Family
Fantasy
Romance

The action output pane shows the execution log:

Action	Time	Response	Duration / Fetch Time
USE lab_1	16:38:53	0 row(s) affected	0.00075 sec
SELECT DISTINCT COUNT(*) AS total_directors FROM directors LIMIT 0, 1000	16:38:57	1 row(s) returned	0.011 sec / 0.00017...
SELECT COUNT(*) AS total_movies FROM movies WHERE year > 2000 LIMIT 0, 1000	16:46:00	1 row(s) returned	0.0048 sec / 0.0000...
SELECT DISTINCT dg.genre FROM directors d JOIN directors_genres dg ON d.id = dg.dir...	16:53:47	8 row(s) returned	0.0091 sec / 0.00001...
SELECT CONCAT(first_name, ' ', last_name) AS directors_full_name FROM directors d ...	17:01:17	15 row(s) returned	0.0076 sec / 0.00002...
SELECT r.role FROM roles r WHERE r.actor_id = (SELECT a.id FROM actors a WHERE a ...	17:05:30	1 row(s) returned	0.0069 sec / 0.00002...
SELECT DISTINCT mg.genre FROM movies m JOIN movies_directors md ON m.id = md...	17:11:44	6 row(s) returned	0.0032 sec / 0.00001...

Figure 3: The list of movie genres that Andrew Adamson has directed.

Explanation: This query selects distinct movie genres from the "movies_genres" table for movies directed by Andrew Adamson. It achieves this by joining the "movies" table with the "movies_directors" table and the "directors" table on their respective IDs and then joining the resulting table with the "movies_genres" table. The query filters the results based on the first and last name of the director being "Andrew" and "Adamson," respectively. The DISTINCT keyword ensures that only unique genres are returned [3].

4. List of directors whose movies are ranked between 7 to 8 ranking.

```

SELECT CONCAT(first_name, ' ', last_name) AS directors_full_name
FROM directors d
JOIN movies_directors md ON d.id = md.director_id
JOIN movies m ON md.movie_id = m.id
WHERE m.rank BETWEEN 7 AND 8;

```

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'Administration' and 'Schemas' are selected. The schema 'lab_1' is chosen. The main area displays a query editor with the following SQL code:

```

18 -- 4. List of directors whose movies are ranked between 7 to 8 ranking
19 • SELECT CONCAT(first_name, ' ', last_name) AS directors_full_name
20 FROM directors d
21 JOIN movies_directors md ON d.id = md.director_id
22 JOIN movies m ON md.movie_id = m.id
23 WHERE m.rank BETWEEN 7 AND 8;

```

The result grid shows the output of the query, listing 13 directors with their full names:

directors_full_name
John (I) Landis
Ron Howard
Rob Reiner
Oliver (I) Stone
Ron Clements
John Musker
Sofia Coppola
Ethan Coen
Joel Coen
Steven Soderbergh
Mike (I) Judge
Daren Aronofsky
John (I) Hughes
Guy Ritchie
David Koepf

Below the result grid, it says 'Result 6'. On the right side of the interface, there is a vertical toolbar with icons for 'Result Grid', 'Form Editor', 'Field Types', and 'Query Stats'.

Figure 4: The list of directors whose movies are in the seventh to eighth ranks.

Explanation: This query retrieves the full names of directors who have directed movies with a rank between 7 and 8. It combines data from the 'directors' table, 'movies_directors' table, and 'movies' table using joins, and concatenates the first name and last name of the directors into a single column called 'directors_full_name' [3].

5. Find the role of Juliette Akinyi in Lost in Translation movie.

```

SELECT r.role
FROM roles r
WHERE r.actor_id = (
    SELECT a.id
    FROM actors a
    WHERE a.first_name = 'Juliette' AND a.last_name = 'Akinyi'
)
AND r.movie_id = (
    SELECT m.id
    FROM movies m
    WHERE m.name = 'Lost in Translation'
);

```

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays the schema structure under 'lab_1'. The central pane contains the following SQL code:

```

25 -- 5. Find the role of Julliet Akinyi in Lost in Translation movie
26 • SELECT r.role
27   FROM roles r
28 WHERE r.actor_id = ( SELECT a.id FROM actors a WHERE a.first_name = 'Julliet' AND a.last_name = 'Akinyi')
29   AND r.movie_id = ( SELECT m.id FROM movies m WHERE m.name = 'Lost in Translation');
30

```

The 'Result Grid' pane shows the output:

role
Hans

There are 7 rows in the result set.

Figure 5: The result of Julliet Akinyi's role in the movie Lost in Translation.

Explanation: This query retrieves the role of an actor named Julliet Akinyi in the movie 'Lost in Translation'. It does so by selecting the role from the 'roles' table, filtering for rows where the actor's ID matches the ID of Julliet Akinyi (obtained from the 'actors' table) and the movie's ID matches the ID of the movie 'Lost in Translation' (obtained from the 'movies' table). The result will be the role played by Julliet Akinyi in that particular movie [3].

6. List of the movies that contain the letter 'o' in any position.

```

SELECT name
FROM movies
WHERE name LIKE '%o%';

```

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays the schema structure under 'lab_1'. The central pane contains the following SQL code:

```

34 -- 6. List of the movies that contain the letter 'o' in any position
35 • SELECT name
36   FROM movies
37 WHERE name LIKE '%o%';

```

The 'Result Grid' pane shows the output:

name
Animal House
Apollo 13
Fargo
Few Good Men, A
Footloose
Godfather, The
Hollow Man
Kill Bill: Vol. 1
Kill Bill: Vol. 2
Lost in Translation
Memento
O Brother, Where...
Ocean's Eleven
Office Space
Pirates of the Ca...
Planes, Trains &...
Pulp Fiction
Reservoir Dogs
Shawshank Red...
Star of Echoes

There are 9 movies listed in the result set.

Figure 6: The result of movies with the letter o in any place.

Explanation: This query selects the names of movies from a table called "movies" where the name contains the letter "o" anywhere in it.

References:

- [1] “MySQL Community Downloads,” *MySQL* [Online]. Available: <https://dev.mysql.com/downloads/workbench/> [Accessed: May 10, 2023].
- [2] “IMDB_dataset,” *Brightspace Dalhousie University* [Online]. Available: <https://dal.brightspace.com/d2l/le/content/271677/viewContent/3623394/View> [Accessed: May 10, 2023].
- [3] “Lab-1,” *Brightspace Dalhousie University* [Online]. Available: <https://dal.brightspace.com/d2l/le/content/271677/viewContent/3623368/View> [Accessed: May 10, 2023].