CSCE 5350.001

Fundamentals of Database Systems

Project Part 3

Naga Vara Pradeep Yendluri

11646461

nagavarapradeepyendluri@my.unt.edu

Project Description:

The Movie Producer Management System is an application that is being developed for a movie production company like Universal Studios. The system is designed to store and manage information about the company's movies, artists, songs, employees and various other aspects of the movie production process. The system will store information about the producing site locations, movie-script-inventory, sponsoring companies, employee data, and payroll. It will also store information about the artists and the movies they have worked on, as well as the various aspects of the movie production process, such as soundtracks, awards, and more.

We have Identified the following entities and relations for the movie producer management system.

1. **Movies:** The entity 'Movies' provides information about the different movies produced by the company. It has 6 attributes, including the movie id, movie title, release date, duration, in production and director. This entity is important for keeping track of the different movies produced by the company and the information related to each movie.
2. **Artists:** The entity 'Artists' provides information about the actors involved in the movies. It has 5 attributes, including the actors id, actors name, actors date of birth, address and age. This entity is important for maintaining the information about the actors, their age and date of birth, which is required for casting actors for various roles.
3. **Genre:** The entity 'Genre' provides information about the genre to which the movie belongs. It has 2 attributes, genre id, and genre name. This entity is important for categorizing the movies into different genres, which helps in better management and analysis of the movies.
4. **Sponsoring Companies:** The entity 'Sponsoring Companies' provides information about the companies that sponsor the movies. It has 4 attributes, including sponsor id, sponsor name, movie id and movie sponsored. This entity is important for tracking the sponsorship deals and the companies that sponsor the movies.
5. **Site Locations:** The entity 'Site Locations' provides information about the different producing sites, including their addresses and buildings. It has 4 attributes, including location id, name, address, and building names. This entity is important for tracking

the different producing sites and their details, which is essential for managing the movie production process.

6. **Buildings:** The entity 'Buildings' provides information about the buildings in each producing site. It has 3 attributes, including Building id, name, and type of building. This entity is important for tracking the different types of buildings present in the producing sites, which is essential for managing the resources and maintenance of the buildings.
7. **Movie Script Inventory:** The entity 'Movie Script Inventory' provides information about the movie scripts. It has 5 attributes, including script id, name, movie id, author, and publication date. This entity is important for tracking the different movie scripts and the information related to each script.
8. **Employee:** The entity 'Employee' provides information about the employees of the company. It has 5 attributes, including employe id, name, job title, hourly pay and phone. This entity is important for maintaining the information about the employees, their job title and contact information, which is required for managing the human resources of the company.
9. **Payroll:** The entity 'Payroll' provides information about employee payroll data. It has 5 attributes, including employee id, hours worked, joining date, work date, etc. This entity is important for tracking the payroll information of the employees, which is essential for managing the finances of the company.
10. **Songs:** The entity 'Songs' provides information about different soundtracks used in the movies. It has 4 attributes, including song id, track title, movie id, and singer name. This entity is important for tracking the different soundtracks used in the movies and the information related to each soundtrack.

Binary Relations:

• One to One relation:

1. Every Employee will only one payroll associated to them. So the relation between employee and payroll is one to one
2. Each Movie will have only one script associated with. Hence, the relation between movies and movie script inventory will also be one to one.

• One to Many relation:

1. Each movie can have multiple songs/soundtracks in it and inversely there can be multiple songs in a movie. So, the relation between songs and movies is one to many.
2. Every site location can have multiple buildings in them, and inversely multiple buildings can be located at a single location. So, the relation between these two entities will be one to many relation.

• Many to Many relation:

1. Each movie can have many artists performing in it and inversely many actors can act in many different movies. Hence, the relation between movies and artists will be many to many.
2. Many Sponsoring companies can sponsor for many movies in this system. Inversely, many movies can get sponsorship from different sponsoring companies. Hence, the relation between these two entities is many to many relation.

3. Many movies can be shot at different locations and inversely many locations can concurrently host shootings for many movies in different buildings so the relation between these two entities will also be many to many relation.

Additional Assumptions:

1. Each movie has one director.
2. Each artist has an address.
3. Payroll has number of hours worked per day.
4. Each movie has in_production flag (values 'Y', 'N') to specify if its in production and not yet released.
5. Salary column is changed to hourly_pay and moved to Employees table to decrease redundancy.
6. Hours_worked column in Payroll has constraint of 0 to 12hours per day.
7. Work date in payroll table holds the date information on which day the employee worked.

ER Relations:

Relations transformed to schema.

Movie (**movie_id**, title, rating, date_of_release, duration, script_inventory_id, director, in_production)

Songs (**song_id**, song_name, singer_name, movie_id)

Genre (**genre_id**, genre_name)

TaggedWith (**movie_id**, **genre_id**)

SiteLocation (**location_id**, location_name, address)

ShotAt (**location_id, movie_id**)

Building (**building_id**, building_name, purpose, location_id)

PostProductionDoneIn (**movie_id, building_id**)

Employees (**employee_id**, employee_name, designation, phone_number, hourly_pay)

Manages (**employee_id, location_id**)

Payroll (**payroll_id**, employee_id, hours_worked, date)

SponsoringCompany (**company_id**, company_name)

getsPaidBy (**artist_id, company_id**)

Produces (**company_id, movie_id**)

Artist (**artist_id**, artist_name, date_of_birth, gender, address)

MovieScriptInventory (**script_inventory_id**, script_inventory_name)

ActsIn(**movie_id, artist_id**)

Queries:

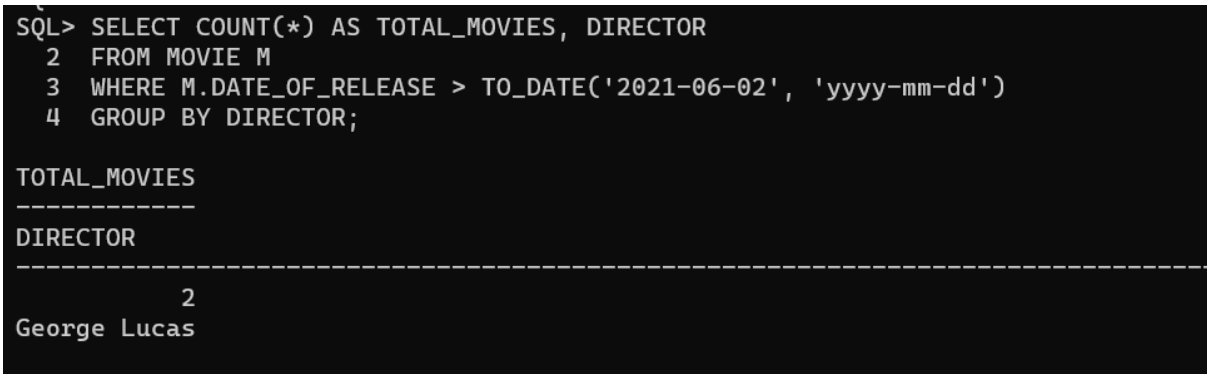
1. List the total number of movies group by director released after June 2nd, 2021.

SELECT COUNT(*) AS TOTAL_MOVIES, DIRECTOR

FROM MOVIE M

WHERE M.DATE_OF_RELEASE > TO_DATE('2021-06-02', 'yyyy-mm-dd')

GROUP BY DIRECTOR;



```
SQL> SELECT COUNT(*) AS TOTAL_MOVIES, DIRECTOR
  2  FROM MOVIE M
  3  WHERE M.DATE_OF_RELEASE > TO_DATE('2021-06-02', 'yyyy-mm-dd')
  4  GROUP BY DIRECTOR;

TOTAL_MOVIES
------------
DIRECTOR
--------------------------------------------------------------------------------
           2
George Lucas
```

2. List movie title(s) that have all artists in their movie with address in Texas.

SELECT m.title

FROM Movie m

INNER JOIN ActsIn ai ON m.movie_id = ai.movie_id

INNER JOIN Artist a ON ai.artist_id = a.artist_id

WHERE a.address LIKE '%Texas%'

GROUP BY m.movie_id, m.title

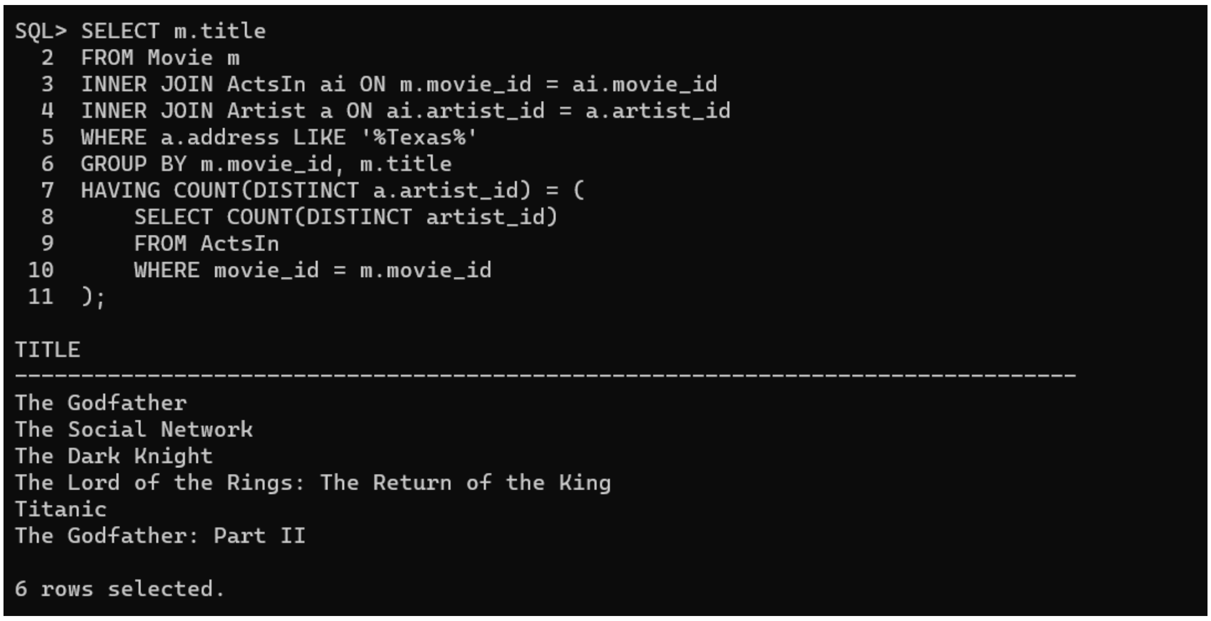
HAVING COUNT(DISTINCT a.artist_id) = (

    SELECT COUNT(DISTINCT artist_id)

    FROM ActsIn

    WHERE movie_id = m.movie_id

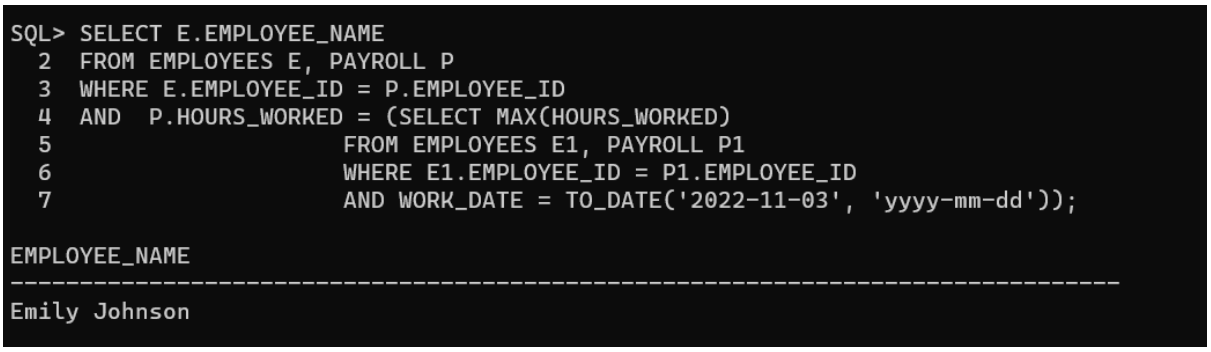
);



```
SQL> SELECT m.title
  FROM Movie m
  INNER JOIN ActsIn ai ON m.movie_id = ai.movie_id
  INNER JOIN Artist a ON ai.artist_id = a.artist_id
  WHERE a.address LIKE '%Texas%'
  GROUP BY m.movie_id, m.title
  HAVING COUNT(DISTINCT a.artist_id) = (
      SELECT COUNT(DISTINCT artist_id)
      FROM ActsIn
      WHERE movie_id = m.movie_id
  );

TITLE
--------------------------------------------------------------------------------
The Godfather
The Social Network
The Dark Knight
The Lord of the Rings: The Return of the King
Titanic
The Godfather: Part II

6 rows selected.
```

3. Find the name of the employee(s) that had worked the most hours on November 3, 2022

```
SELECT E.EMPLOYEE_NAME
FROM EMPLOYEES E, PAYROLL P
WHERE E.EMPLOYEE_ID = P.EMPLOYEE_ID
AND  P.HOURS_WORKED = (SELECT MAX(HOURS_WORKED)
                    FROM EMPLOYEES E1, PAYROLL P1
                    WHERE E1.EMPLOYEE_ID = P1.EMPLOYEE_ID
                    AND WORK_DATE = TO_DATE('2022-11-03', 'yyyy-mm-dd'));
```

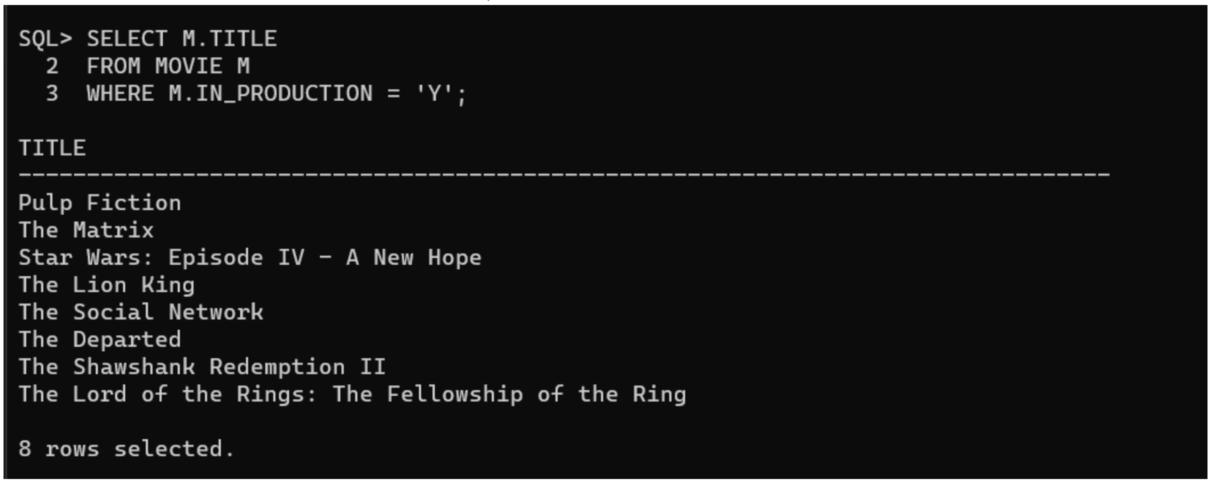


```
SQL> SELECT E.EMPLOYEE_NAME
  FROM EMPLOYEES E, PAYROLL P
  WHERE E.EMPLOYEE_ID = P.EMPLOYEE_ID
  AND  P.HOURS_WORKED = (SELECT MAX(HOURS_WORKED)
                     FROM EMPLOYEES E1, PAYROLL P1
                     WHERE E1.EMPLOYEE_ID = P1.EMPLOYEE_ID
                     AND WORK_DATE = TO_DATE('2022-11-03', 'yyyy-mm-dd'));

EMPLOYEE_NAME
--------------------------------------------------------------------------------
Emily Johnson
```

4. List the movies that currently are in production.

```
SELECT M.TITLE
FROM MOVIE M
```

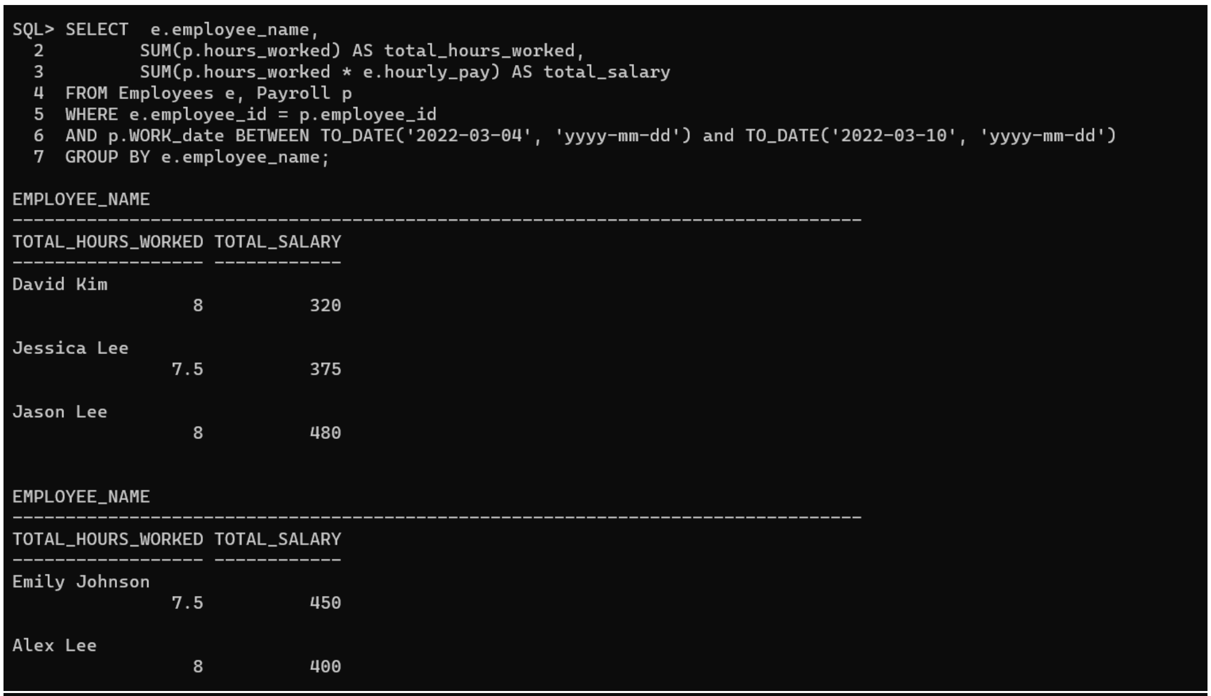
WHERE M.IN_PRODUCTION = 'Y';



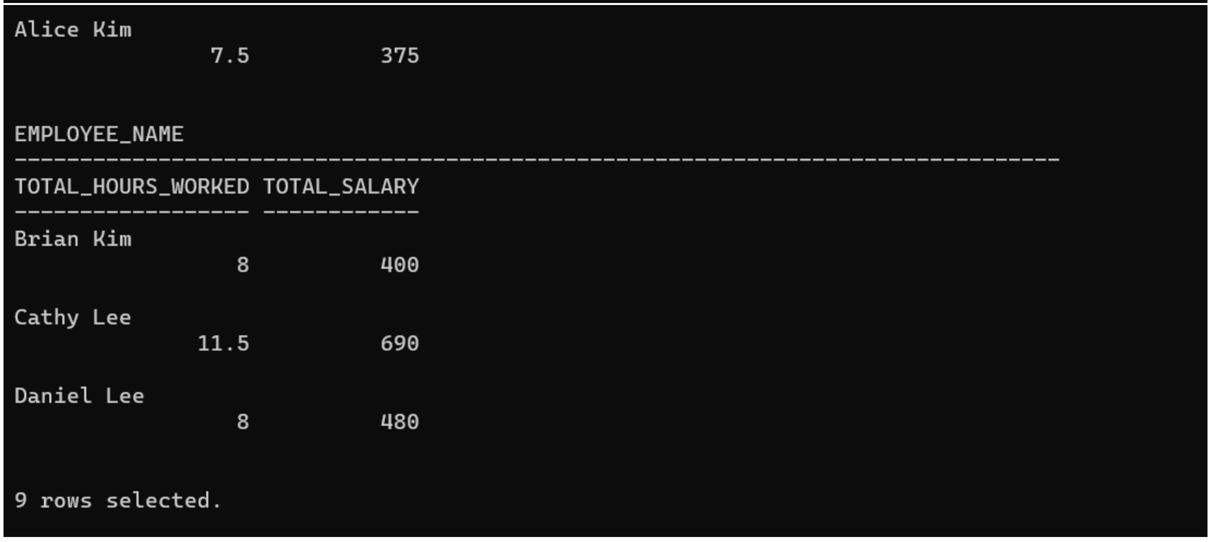
```
SQL> SELECT M.TITLE
  2  FROM MOVIE M
  3  WHERE M.IN_PRODUCTION = 'Y';

TITLE
--------------------------------------------------------------------------------
Pulp Fiction
The Matrix
Star Wars: Episode IV - A New Hope
The Lion King
The Social Network
The Departed
The Shawshank Redemption II
The Lord of the Rings: The Fellowship of the Ring

8 rows selected.
```

5. Print the payroll from March 4, 2022 to March 10, 2022 displaying employee name, hours worked and total salary for all employees

```
SELECT  e.employee_name,
       SUM(p.hours_worked) AS total_hours_worked,
       SUM(p.hours_worked * e.hourly_pay) AS total_salary
FROM Employees e, Payroll p
WHERE e.employee_id = p.employee_id
AND p.WORK_date BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and
TO_DATE('2022-03-10', 'yyyy-mm-dd')
GROUP BY e.employee_name;
```



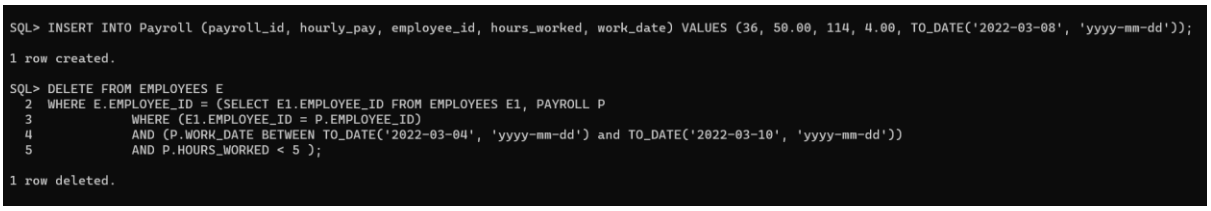
```
SQL> SELECT  e.employee_name,
  2        SUM(p.hours_worked) AS total_hours_worked,
  3        SUM(p.hours_worked * e.hourly_pay) AS total_salary
  4  FROM Employees e, Payroll p
  5  WHERE e.employee_id = p.employee_id
  6  AND p.WORK_date BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd')
  7  GROUP BY e.employee_name;

EMPLOYEE_NAME
--------------------------------------------------------------------------------
TOTAL_HOURS_WORKED TOTAL_SALARY
------------------ ------------
David Kim
                 8          320

Jessica Lee
               7.5          375

Jason Lee
                 8          480


EMPLOYEE_NAME
--------------------------------------------------------------------------------
TOTAL_HOURS_WORKED TOTAL_SALARY
------------------ ------------
Emily Johnson
               7.5          450

Alex Lee
                 8          400
```



```
Alice Kim
               7.5          375


EMPLOYEE_NAME
--------------------------------------------------------------------------------
TOTAL_HOURS_WORKED TOTAL_SALARY
------------------ ------------
Brian Kim
                 8          400

Cathy Lee
              11.5          690

Daniel Lee
                 8          480


9 rows selected.
```

6. Design a delete statement to delete employees working less than 5 hours from March 4, 2023 to March 10, 2023.

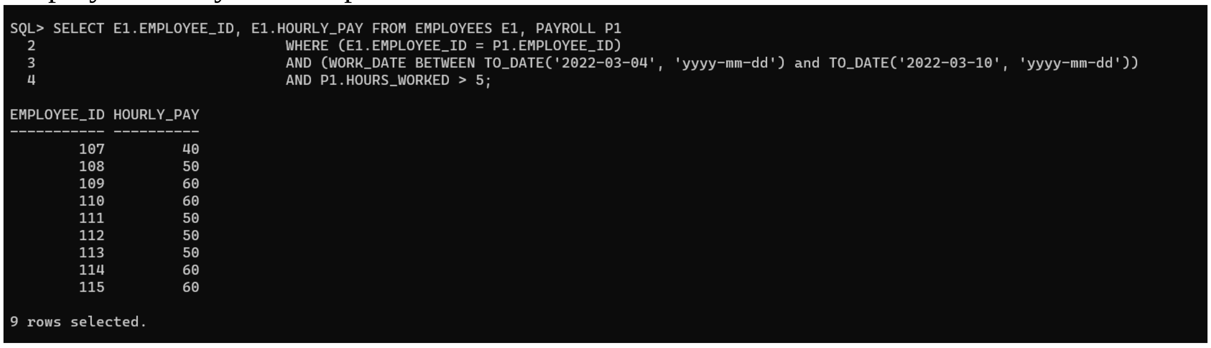
```
DELETE FROM EMPLOYEES E

WHERE E.EMPLOYEE_ID = (SELECT E1.EMPLOYEE_ID FROM EMPLOYEES E1, PAYROLL P

                WHERE (E1.EMPLOYEE_ID = P.EMPLOYEE_ID)

                AND (P.WORK_DATE BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd'))

                AND P.HOURS_WORKED < 5 );
```



```
SQL> INSERT INTO Payroll (payroll_id, hourly_pay, employee_id, hours_worked, work_date) VALUES (36, 50.00, 114, 4.00, TO_DATE('2022-03-08', 'yyyy-mm-dd'));

1 row created.

SQL> DELETE FROM EMPLOYEES E
     WHERE E.EMPLOYEE_ID = (SELECT E1.EMPLOYEE_ID FROM EMPLOYEES E1, PAYROLL P
                 WHERE (E1.EMPLOYEE_ID = P.EMPLOYEE_ID)
                 AND (P.WORK_DATE BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd'))
                 AND P.HOURS_WORKED < 5 );

1 row deleted.
```

7. Design an update statement to give a 23% salary raise to employees working more than 5 hours from March 4, 2023 to March 10, 2023.

Employees salary before update:



```
SQL> SELECT E1.EMPLOYEE_ID, E1.HOURLY_PAY FROM EMPLOYEES E1, PAYROLL P1
                                WHERE (E1.EMPLOYEE_ID = P1.EMPLOYEE_ID)
                                AND (WORK_DATE BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd'))
                                AND P1.HOURS_WORKED > 5;

EMPLOYEE_ID HOURLY_PAY
----------- ----------
        107         40
        108         50
        109         60
        110         60
        111         50
        112         50
        113         50
        114         60
        115         60

9 rows selected.
```

Update:

UPDATE EMPLOYEES E

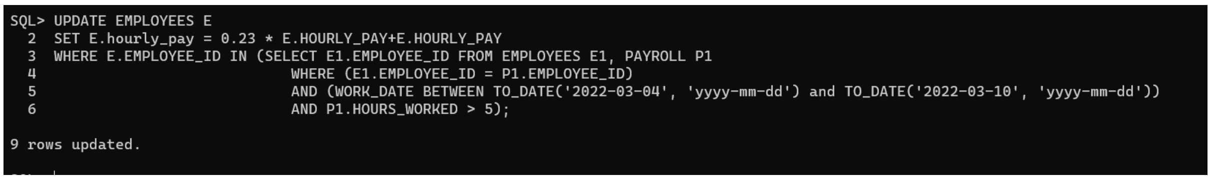
SET E.hourly_pay = 0.23 * E.HOURLY_PAY+E.HOURLY_PAY

WHERE E.EMPLOYEE_ID IN (SELECT E1.EMPLOYEE_ID FROM EMPLOYEES E1, PAYROLL P1

WHERE (E1.EMPLOYEE_ID = P1.EMPLOYEE_ID)

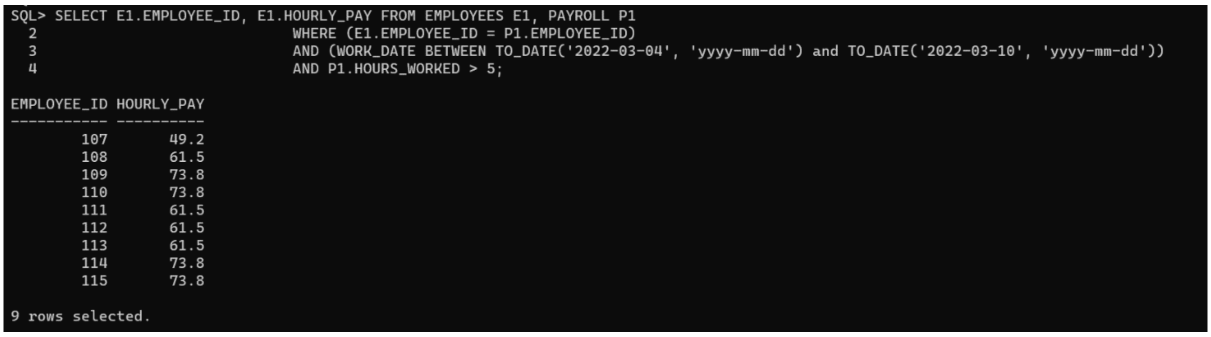
AND (WORK_DATE BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd'))

AND P1.HOURS_WORKED > 5);



```
SQL> UPDATE EMPLOYEES E
     SET E.hourly_pay = 0.23 * E.HOURLY_PAY+E.HOURLY_PAY
     WHERE E.EMPLOYEE_ID IN (SELECT E1.EMPLOYEE_ID FROM EMPLOYEES E1, PAYROLL P1
                                WHERE (E1.EMPLOYEE_ID = P1.EMPLOYEE_ID)
                                AND (WORK_DATE BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd'))
                                AND P1.HOURS_WORKED > 5);

9 rows updated.
```

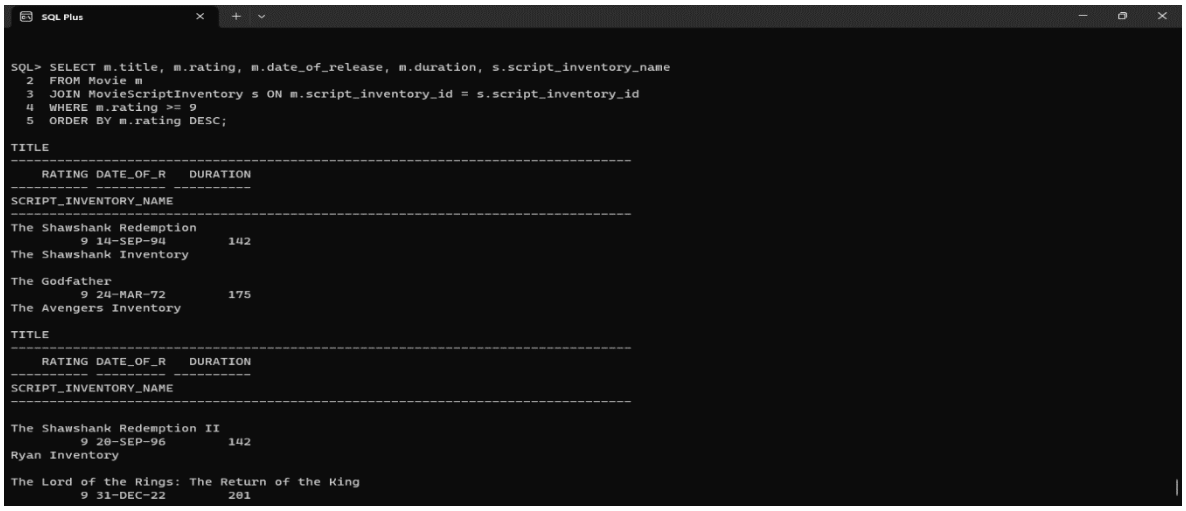
After Update:



```
SQL> SELECT E1.EMPLOYEE_ID, E1.HOURLY_PAY FROM EMPLOYEES E1, PAYROLL P1
                                WHERE (E1.EMPLOYEE_ID = P1.EMPLOYEE_ID)
                                AND (WORK_DATE BETWEEN TO_DATE('2022-03-04', 'yyyy-mm-dd') and TO_DATE('2022-03-10', 'yyyy-mm-dd'))
                                AND P1.HOURS_WORKED > 5;

EMPLOYEE_ID HOURLY_PAY
----------- ----------
        107       49.2
        108       61.5
        109       73.8
        110       73.8
        111       61.5
        112       61.5
        113       61.5
        114       73.8
        115       73.8

9 rows selected.
```

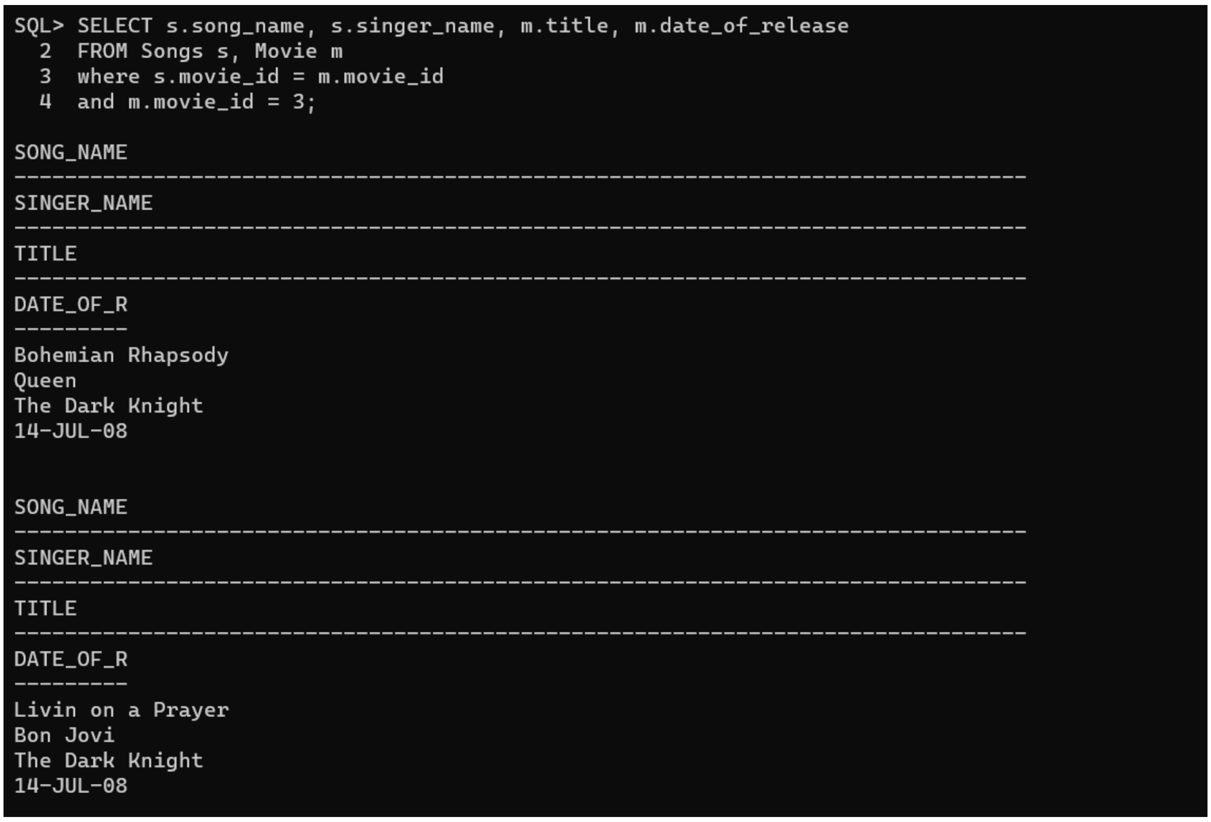
10 Queries:

1) Query to retrieve script name and movie name of films, duration, and date of release with rating greater than or equal to 9 and displays result in descending order.



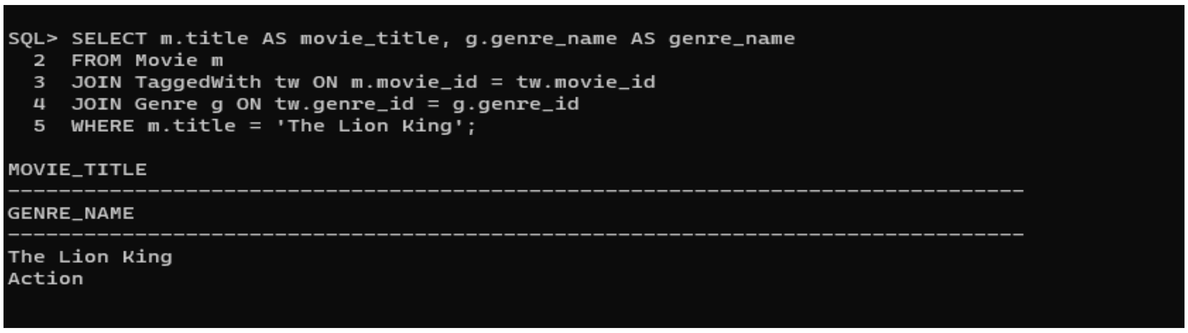
```
SQL> SELECT m.title, m.rating, m.date_of_release, m.duration, s.script_inventory_name
     FROM Movie m
     JOIN MovieScriptInventory s ON m.script_inventory_id = s.script_inventory_id
     WHERE m.rating >= 9
     ORDER BY m.rating DESC;

TITLE
--------------------------------------------------------------------------------
    RATING DATE_OF_R   DURATION
---------- --------- ----------
SCRIPT_INVENTORY_NAME
--------------------------------------------------------------------------------
The Shawshank Redemption
         9 14-SEP-94        142
The Shawshank Inventory

The Godfather
         9 24-MAR-72        175
The Avengers Inventory

TITLE
--------------------------------------------------------------------------------
    RATING DATE_OF_R   DURATION
---------- --------- ----------
SCRIPT_INVENTORY_NAME
--------------------------------------------------------------------------------

The Shawshank Redemption II
         9 20-SEP-96        142
Ryan Inventory

The Lord of the Rings: The Return of the King
         9 31-DEC-22        201
```

2) Query to display the song details like song name, movie name, movie release date and singer name of movie with id = 3.



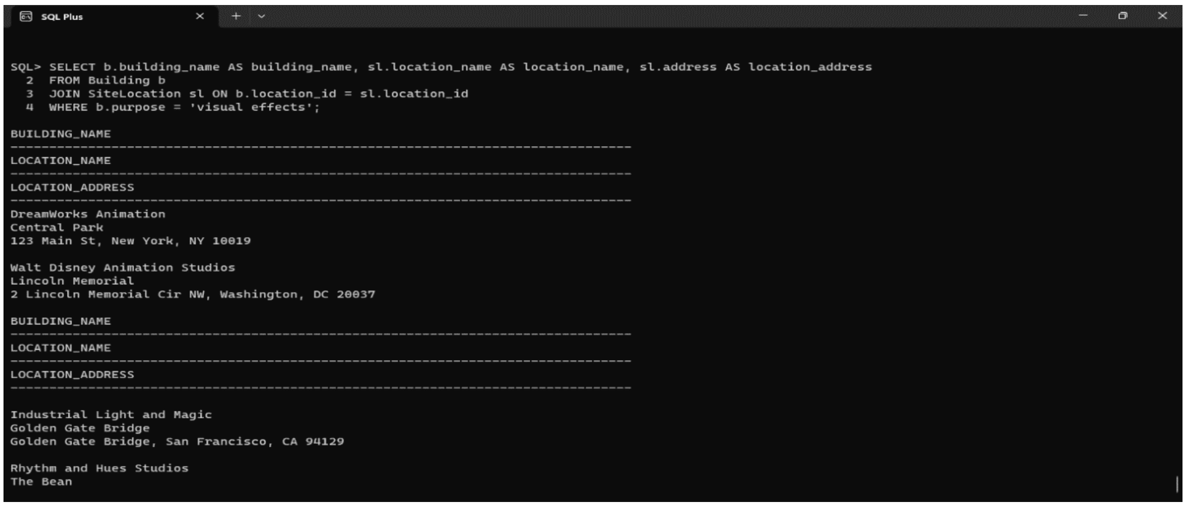
```
SQL> SELECT s.song_name, s.singer_name, m.title, m.date_of_release
     FROM Songs s, Movie m
     where s.movie_id = m.movie_id
     and m.movie_id = 3;

SONG_NAME
--------------------------------------------------------------------------------
SINGER_NAME
--------------------------------------------------------------------------------
TITLE
--------------------------------------------------------------------------------
DATE_OF_R
---------
Bohemian Rhapsody
Queen
The Dark Knight
14-JUL-08


SONG_NAME
--------------------------------------------------------------------------------
SINGER_NAME
--------------------------------------------------------------------------------
TITLE
--------------------------------------------------------------------------------
DATE_OF_R
---------
Livin on a Prayer
Bon Jovi
The Dark Knight
14-JUL-08
```

3) Query to Display the genre associated with the movie titled “The Lion King”



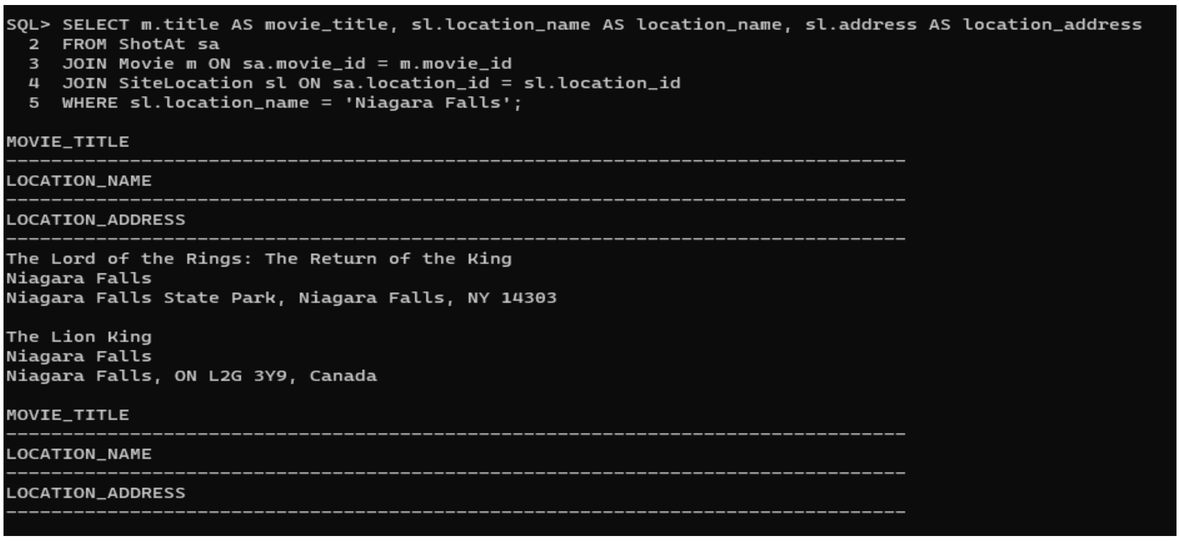
```
SQL> SELECT m.title AS movie_title, g.genre_name AS genre_name
     FROM Movie m
     JOIN TaggedWith tw ON m.movie_id = tw.movie_id
     JOIN Genre g ON tw.genre_id = g.genre_id
     WHERE m.title = 'The Lion King';

MOVIE_TITLE
--------------------------------------------------------------------------------
GENRE_NAME
--------------------------------------------------------------------------------
The Lion King
Action
```

4) Query to display the building name, Location name and address of building that is used for Visual effects purpose



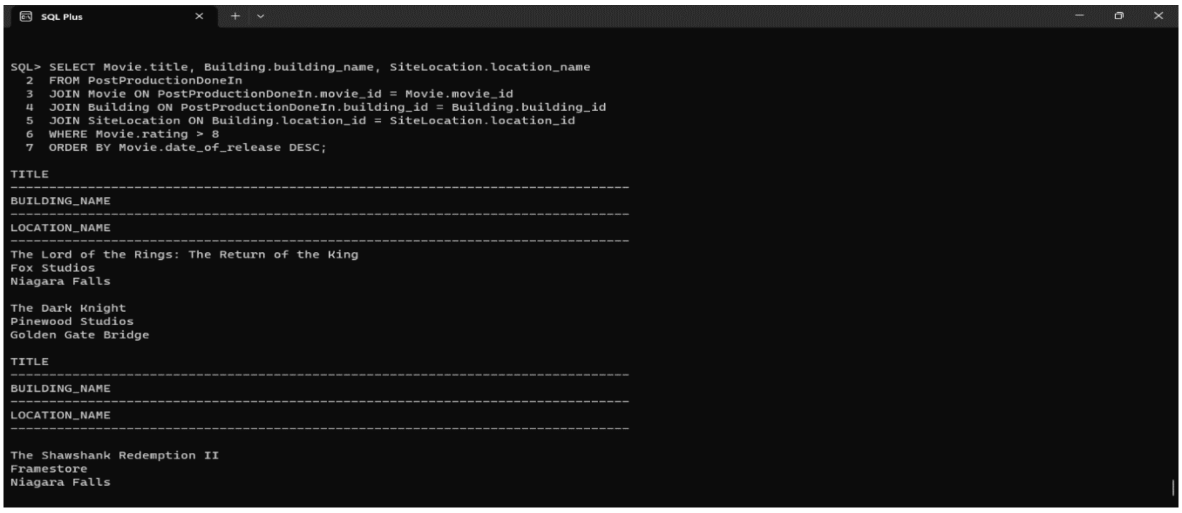
```
SQL> SELECT b.building_name AS building_name, sl.location_name AS location_name, sl.address AS location_address
     FROM Building b
     JOIN SiteLocation sl ON b.location_id = sl.location_id
     WHERE b.purpose = 'visual effects';

BUILDING_NAME
--------------------------------------------------------------------------------
LOCATION_NAME
--------------------------------------------------------------------------------
LOCATION_ADDRESS
--------------------------------------------------------------------------------
DreamWorks Animation
Central Park
123 Main St, New York, NY 10019

Walt Disney Animation Studios
Lincoln Memorial
2 Lincoln Memorial Cir NW, Washington, DC 20037

BUILDING_NAME
--------------------------------------------------------------------------------
LOCATION_NAME
--------------------------------------------------------------------------------
LOCATION_ADDRESS
--------------------------------------------------------------------------------

Industrial Light and Magic
Golden Gate Bridge
Golden Gate Bridge, San Francisco, CA 94129

Rhythm and Hues Studios
The Bean
```

5) Query to display movie name, shooting location and address of movies which were shot at location Niagara falls



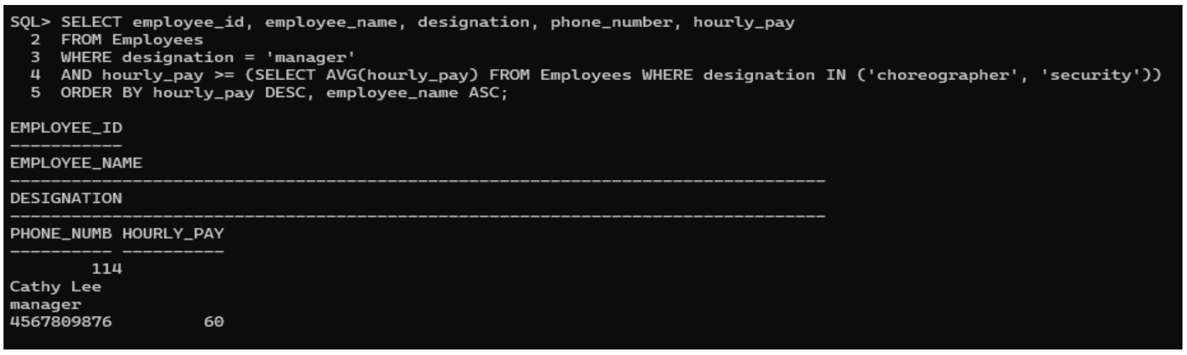
```
SQL> SELECT m.title AS movie_title, sl.location_name AS location_name, sl.address AS location_address
     FROM ShotAt sa
     JOIN Movie m ON sa.movie_id = m.movie_id
     JOIN SiteLocation sl ON sa.location_id = sl.location_id
     WHERE sl.location_name = 'Niagara Falls';

MOVIE_TITLE
--------------------------------------------------------------------------------
LOCATION_NAME
--------------------------------------------------------------------------------
LOCATION_ADDRESS
--------------------------------------------------------------------------------
The Lord of the Rings: The Return of the King
Niagara Falls
Niagara Falls State Park, Niagara Falls, NY 14303

The Lion King
Niagara Falls
Niagara Falls, ON L2G 3Y9, Canada

MOVIE_TITLE
--------------------------------------------------------------------------------
LOCATION_NAME
--------------------------------------------------------------------------------
LOCATION_ADDRESS
--------------------------------------------------------------------------------
```

6) Query to display the Movie title, Post production building and address of movies with rating above 8 and sorted in descending order via release dates



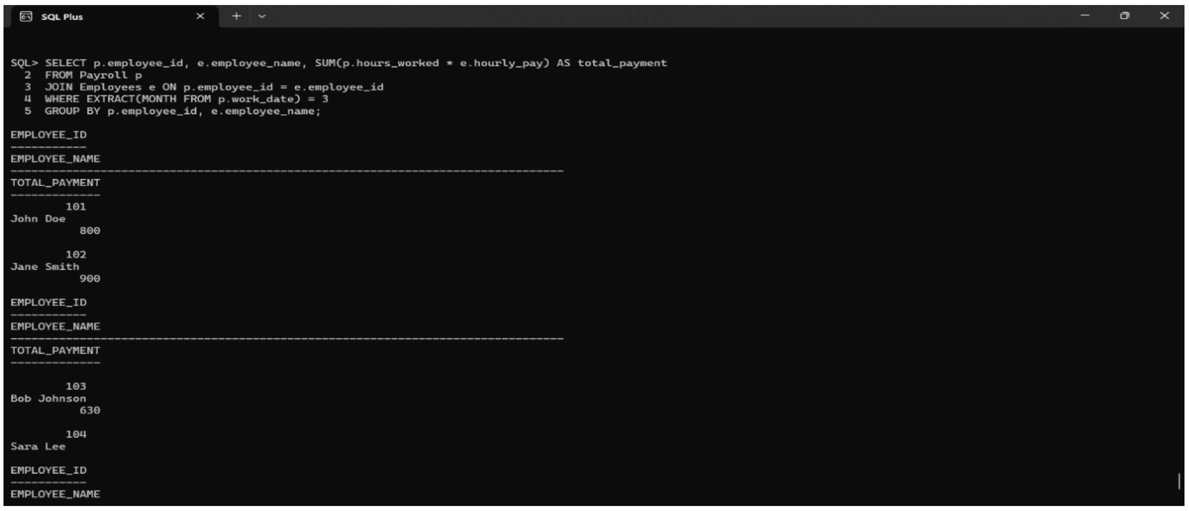
```
SQL> SELECT Movie.title, Building.building_name, SiteLocation.location_name
     FROM PostProductionDoneIn
     JOIN Movie ON PostProductionDoneIn.movie_id = Movie.movie_id
     JOIN Building ON PostProductionDoneIn.building_id = Building.building_id
     JOIN SiteLocation ON Building.location_id = SiteLocation.location_id
     WHERE Movie.rating > 8
     ORDER BY Movie.date_of_release DESC;

TITLE
--------------------------------------------------------------------------------
BUILDING_NAME
--------------------------------------------------------------------------------
LOCATION_NAME
--------------------------------------------------------------------------------
The Lord of the Rings: The Return of the King
Fox Studios
Niagara Falls

The Dark Knight
Pinewood Studios
Golden Gate Bridge

TITLE
--------------------------------------------------------------------------------
BUILDING_NAME
--------------------------------------------------------------------------------
LOCATION_NAME
--------------------------------------------------------------------------------

The Shawshank Redemption II
Framestore
Niagara Falls
```

7) Query to display the details of employee with designation of manager whose hourly pay is greater than the average pay of employee with designation choreographer and security



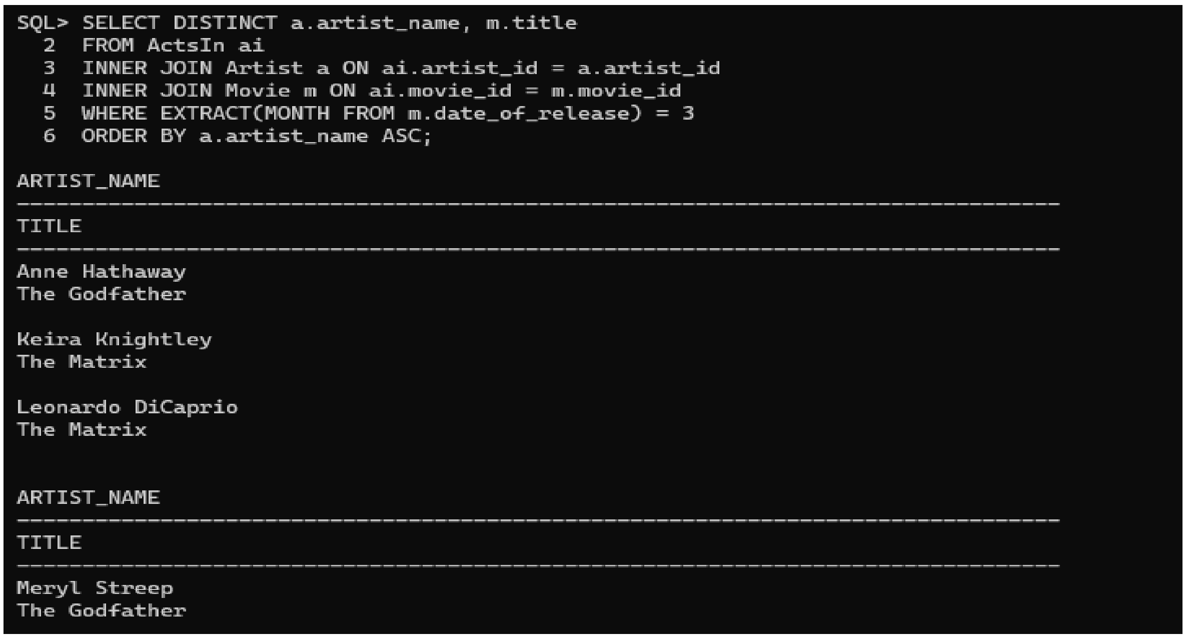
```
SQL> SELECT employee_id, employee_name, designation, phone_number, hourly_pay
     FROM Employees
     WHERE designation = 'manager'
     AND hourly_pay >= (SELECT AVG(hourly_pay) FROM Employees WHERE designation IN ('choreographer', 'security'))
     ORDER BY hourly_pay DESC, employee_name ASC;

EMPLOYEE_ID
-----------
EMPLOYEE_NAME
--------------------------------------------------------------------------------
DESIGNATION
--------------------------------------------------------------------------------
PHONE_NUMB HOURLY_PAY
---------- ----------
        114
Cathy Lee
manager
4567809876         60
```

8) Query to display the total payroll amount for each employee for the month of march



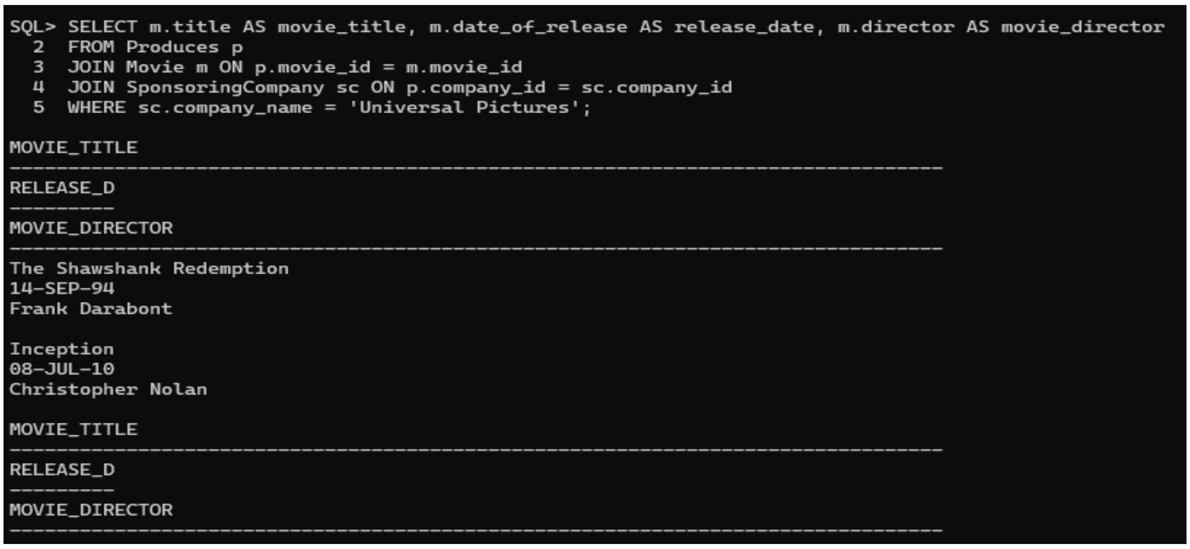
```
SQL> SELECT p.employee_id, e.employee_name, SUM(p.hours_worked * e.hourly_pay) AS total_payment
     FROM Payroll p
     JOIN Employees e ON p.employee_id = e.employee_id
     WHERE EXTRACT(MONTH FROM p.work_date) = 3
     GROUP BY p.employee_id, e.employee_name;

EMPLOYEE_ID
-----------
EMPLOYEE_NAME
--------------------------------------------------------------------------------
TOTAL_PAYMENT
-------------
        101
John Doe
          800

        102
Jane Smith
          900

EMPLOYEE_ID
-----------
EMPLOYEE_NAME
--------------------------------------------------------------------------------
TOTAL_PAYMENT
-------------

        103
Bob Johnson
          630

        104
Sara Lee

EMPLOYEE_ID
-----------
EMPLOYEE_NAME
```

9) Query to display the Artists names and movie title of films released in the month of march



```
SQL> SELECT DISTINCT a.artist_name, m.title
     FROM ActsIn ai
     INNER JOIN Artist a ON ai.artist_id = a.artist_id
     INNER JOIN Movie m ON ai.movie_id = m.movie_id
     WHERE EXTRACT(MONTH FROM m.date_of_release) = 3
     ORDER BY a.artist_name ASC;

ARTIST_NAME
--------------------------------------------------------------------------------
TITLE
--------------------------------------------------------------------------------
Anne Hathaway
The Godfather

Keira Knightley
The Matrix

Leonardo DiCaprio
The Matrix


ARTIST_NAME
--------------------------------------------------------------------------------
TITLE
--------------------------------------------------------------------------------
Meryl Streep
The Godfather
```

10) Query to display the movie title, date of release and director name of films that were produced by the Universal Pictures.

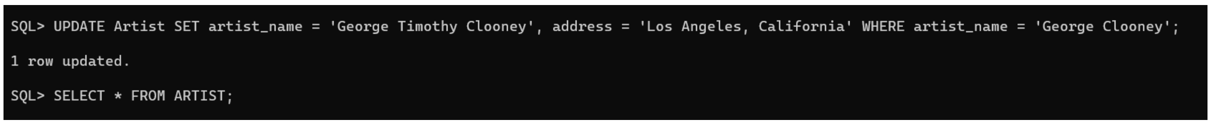


```
SQL> SELECT m.title AS movie_title, m.date_of_release AS release_date, m.director AS movie_director
     FROM Produces p
     JOIN Movie m ON p.movie_id = m.movie_id
     JOIN SponsoringCompany sc ON p.company_id = sc.company_id
     WHERE sc.company_name = 'Universal Pictures';

MOVIE_TITLE
--------------------------------------------------------------------------------
RELEASE_D
---------
MOVIE_DIRECTOR
--------------------------------------------------------------------------------
The Shawshank Redemption
14-SEP-94
Frank Darabont

Inception
08-JUL-10
Christopher Nolan

MOVIE_TITLE
--------------------------------------------------------------------------------
RELEASE_D
---------
MOVIE_DIRECTOR
--------------------------------------------------------------------------------
```

Updates:

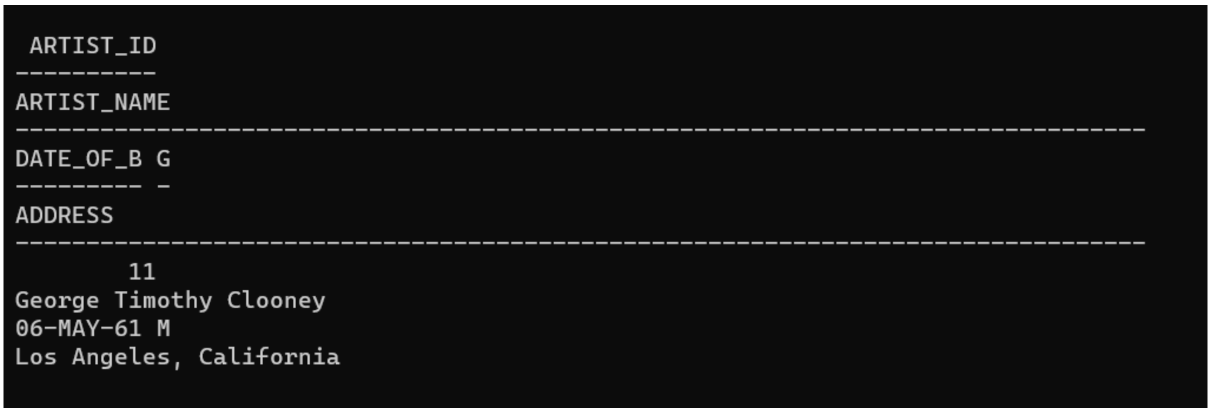
1)Artist table

UPDATE Artist SET artist_name = 'George Timothy Clooney', address = 'Los Angeles, California' WHERE artist_name = 'George Clooney';



```
SQL> UPDATE Artist SET artist_name = 'George Timothy Clooney', address = 'Los Angeles, California' WHERE artist_name = 'George Clooney';

1 row updated.

SQL> SELECT * FROM ARTIST;
```

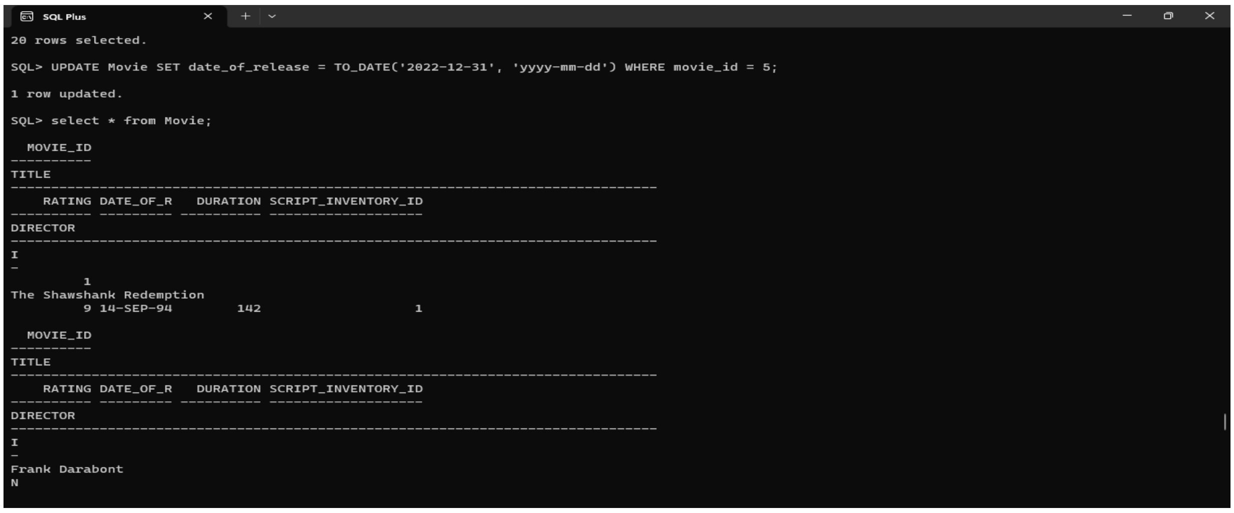
Updated artist name and address



```
 ARTIST_ID
----------
ARTIST_NAME
--------------------------------------------------------------------------------
DATE_OF_B G
--------- -
ADDRESS
--------------------------------------------------------------------------------
        11
George Timothy Clooney
06-MAY-61 M
Los Angeles, California
```

2)Movie table:

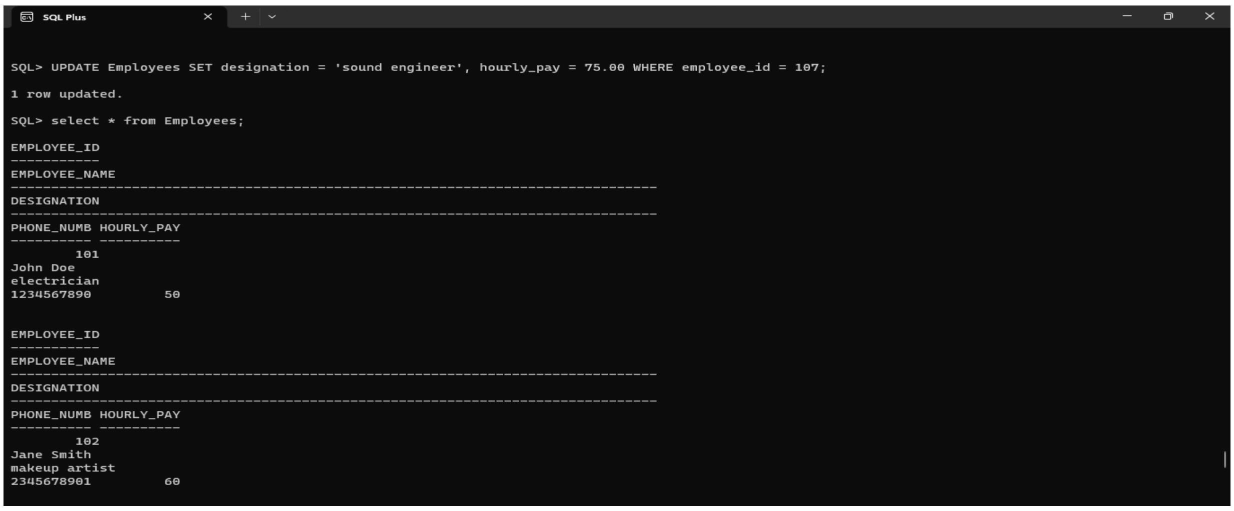
UPDATE Movie SET date_of_release = TO_DATE('2022-12-31', 'yyyy-mm-dd') WHERE movie_id = 5;



```
20 rows selected.

SQL> UPDATE Movie SET date_of_release = TO_DATE('2022-12-31', 'yyyy-mm-dd') WHERE movie_id = 5;

1 row updated.

SQL> select * from Movie;

  MOVIE_ID
----------
TITLE
--------------------------------------------------------------------------------
    RATING DATE_OF_R   DURATION SCRIPT_INVENTORY_ID
---------- --------- ---------- -------------------
DIRECTOR
--------------------------------------------------------------------------------
I
-
         1
The Shawshank Redemption
         9 14-SEP-94        142                   1

  MOVIE_ID
----------
TITLE
--------------------------------------------------------------------------------
    RATING DATE_OF_R   DURATION SCRIPT_INVENTORY_ID
---------- --------- ---------- -------------------
DIRECTOR
--------------------------------------------------------------------------------
I
-
Frank Darabont
N
```

3) Employees table:

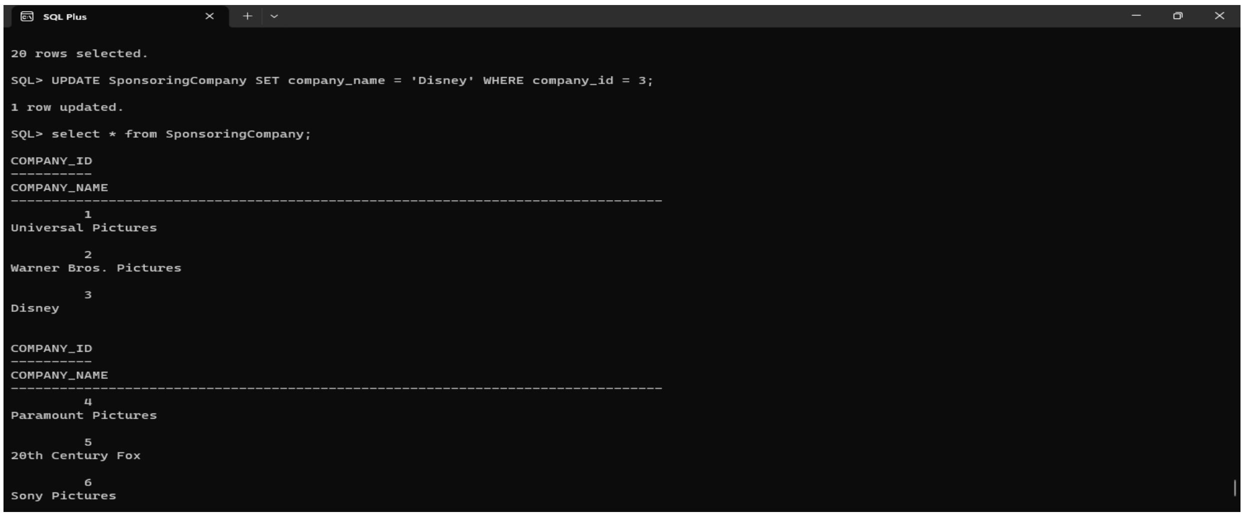
UPDATE Employees SET designation = 'sound engineer', hourly_pay = 75.00 WHERE employee_id = 107;



```
SQL> UPDATE Employees SET designation = 'sound engineer', hourly_pay = 75.00 WHERE employee_id = 107;

1 row updated.

SQL> select * from Employees;

EMPLOYEE_ID
-----------
EMPLOYEE_NAME
--------------------------------------------------------------------------------
DESIGNATION
--------------------------------------------------------------------------------
PHONE_NUMB HOURLY_PAY
---------- ----------
        101
John Doe
electrician
1234567890         50


EMPLOYEE_ID
-----------
EMPLOYEE_NAME
--------------------------------------------------------------------------------
DESIGNATION
--------------------------------------------------------------------------------
PHONE_NUMB HOURLY_PAY
---------- ----------
        102
Jane Smith
makeup artist
2345678901         60
```

4)SponsoringCompany Table:

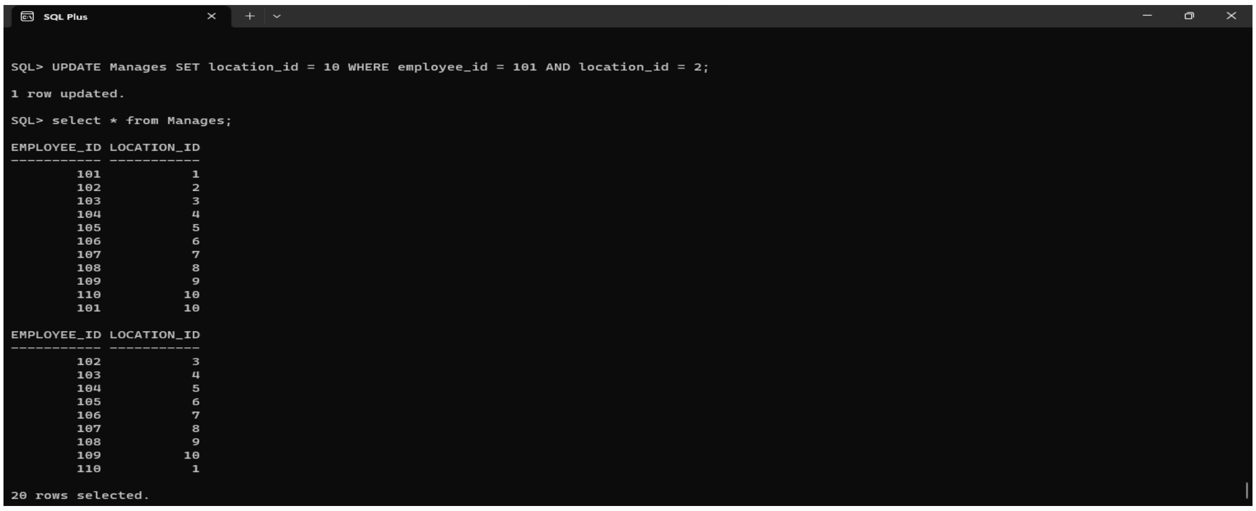
UPDATE SponsoringCompany SET company_name = 'Disney' WHERE company_id = 3;



```
20 rows selected.

SQL> UPDATE SponsoringCompany SET company_name = 'Disney' WHERE company_id = 3;

1 row updated.

SQL> select * from SponsoringCompany;

COMPANY_ID
----------
COMPANY_NAME
--------------------------------------------------------------------------------
         1
Universal Pictures

         2
Warner Bros. Pictures

         3
Disney


COMPANY_ID
----------
COMPANY_NAME
--------------------------------------------------------------------------------
         4
Paramount Pictures

         5
20th Century Fox

         6
Sony Pictures
```

5)Manages table:

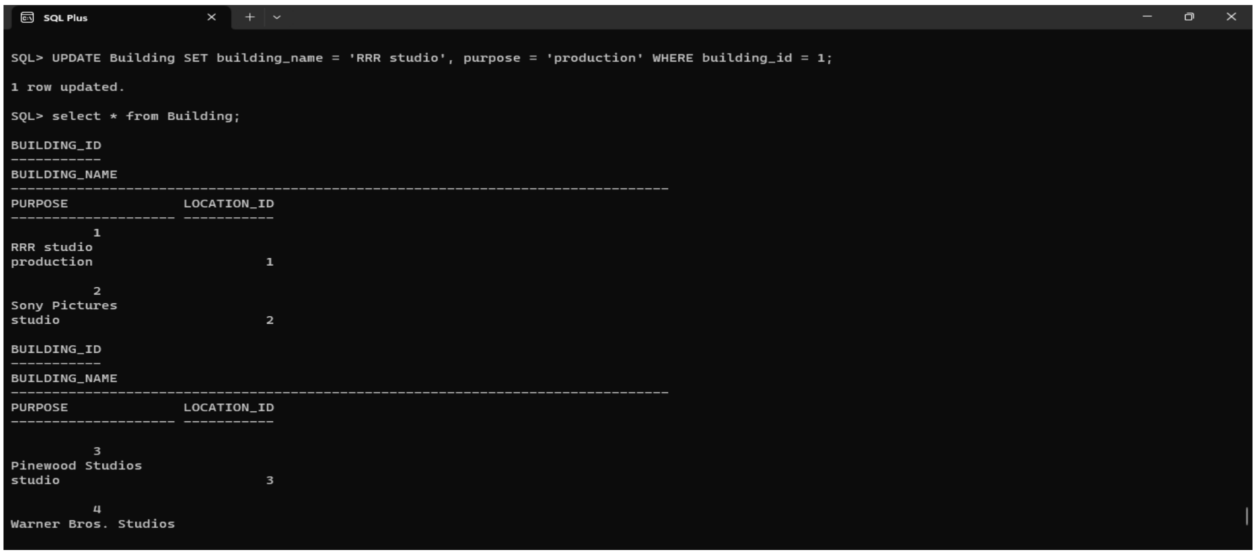
UPDATE Manages SET location_id = 10 WHERE employee_id = 101 AND location_id = 2;



```
SQL> UPDATE Manages SET location_id = 10 WHERE employee_id = 101 AND location_id = 2;

1 row updated.

SQL> select * from Manages;

EMPLOYEE_ID LOCATION_ID
----------- -----------
        101           1
        102           2
        103           3
        104           4
        105           5
        106           6
        107           7
        108           8
        109           9
        110          10
        101          10

EMPLOYEE_ID LOCATION_ID
----------- -----------
        102           3
        103           4
        104           5
        105           6
        106           7
        107           8
        108           9
        109          10
        110           1

20 rows selected.
```

6)Buildings table:

UPDATE Building SET building_name = 'RRR studio', purpose = 'production' WHERE building_id = 1;

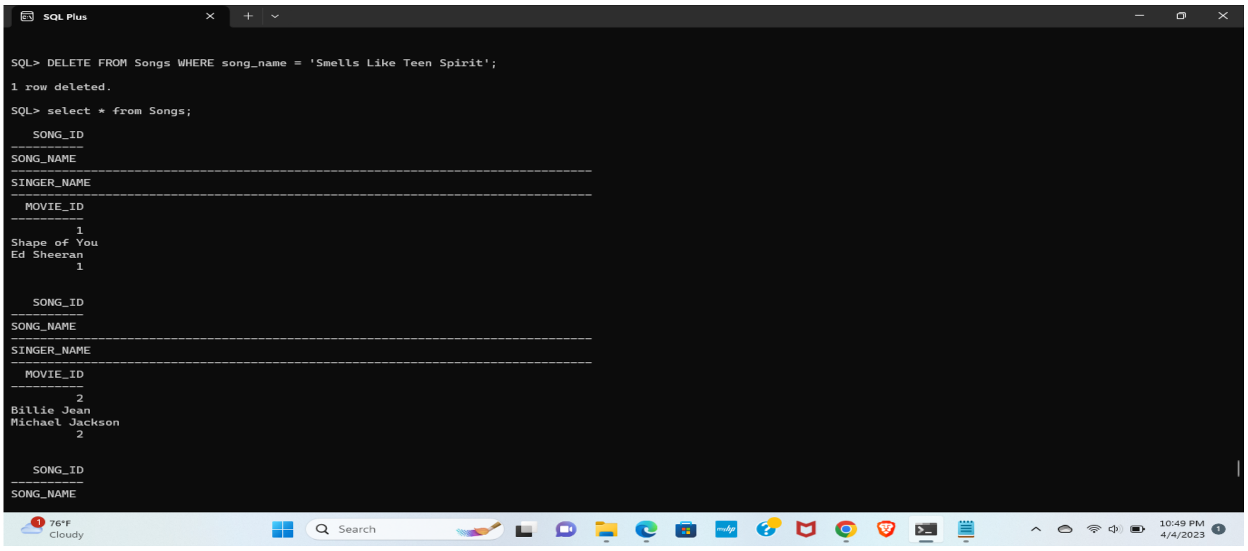


```
SQL> UPDATE Building SET building_name = 'RRR studio', purpose = 'production' WHERE building_id = 1;

1 row updated.

SQL> select * from Building;

BUILDING_ID
-----------
BUILDING_NAME
--------------------------------------------------------------------------------
PURPOSE              LOCATION_ID
-------------------- -----------
          1
RRR studio
production                     1

          2
Sony Pictures
studio                         2

BUILDING_ID
-----------
BUILDING_NAME
--------------------------------------------------------------------------------
PURPOSE              LOCATION_ID
-------------------- -----------

          3
Pinewood Studios
studio                         3

          4
Warner Bros. Studios
```

Deletion:

1)Songs table:

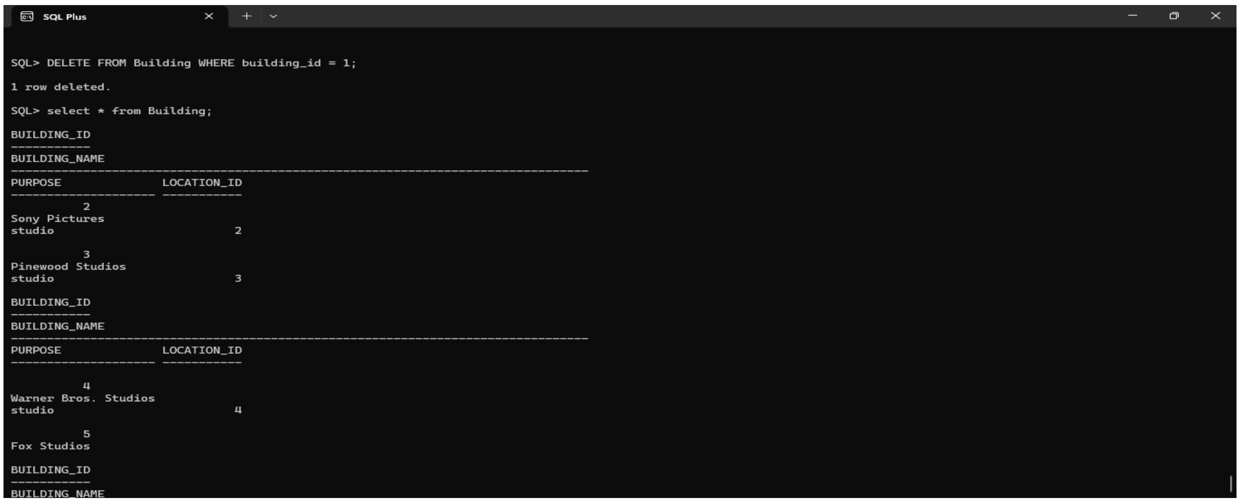
DELETE FROM Songs WHERE song_name = ‘Smells Like Teen Spirit’;



```
SQL> DELETE FROM Songs WHERE song_name = 'Smells Like Teen Spirit';

1 row deleted.

SQL> select * from Songs;

   SONG_ID
----------
SONG_NAME
--------------------------------------------------------------------------------
SINGER_NAME
--------------------------------------------------------------------------------
  MOVIE_ID
----------
         1
Shape of You
Ed Sheeran
         1


   SONG_ID
----------
SONG_NAME
--------------------------------------------------------------------------------
SINGER_NAME
--------------------------------------------------------------------------------
  MOVIE_ID
----------
         2
Billie Jean
Michael Jackson
         2


   SONG_ID
----------
SONG_NAME
```

2)Buildings table:

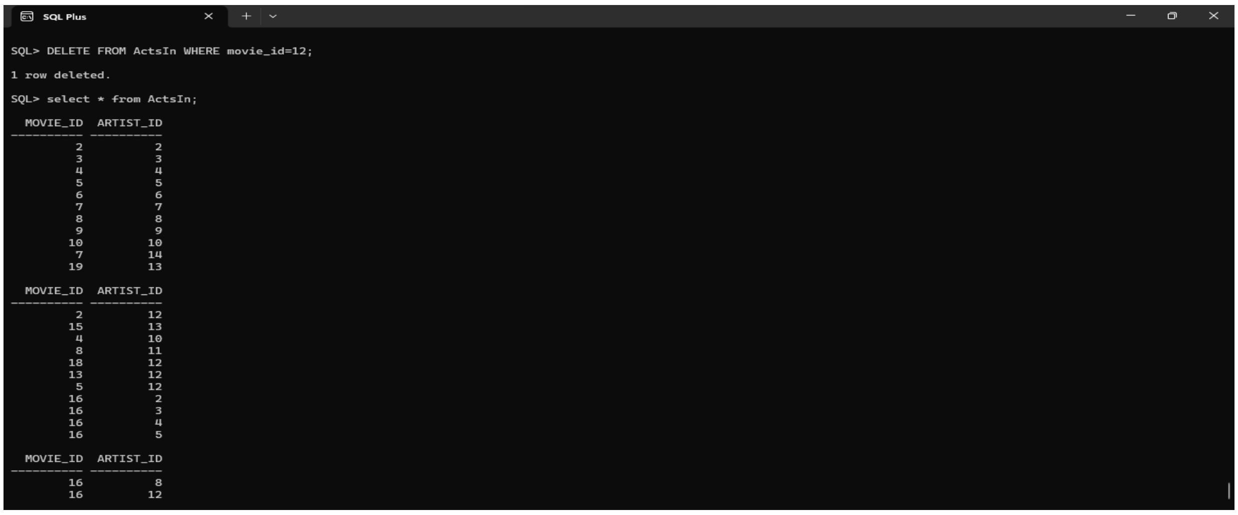
DELETE FROM Building WHERE building_id = 1;



```
SQL> DELETE FROM Building WHERE building_id = 1;

1 row deleted.

SQL> select * from Building;

BUILDING_ID
-----------
BUILDING_NAME
--------------------------------------------------------------------------------
PURPOSE              LOCATION_ID
-------------------- -----------
          2
Sony Pictures
studio                         2

          3
Pinewood Studios
studio                         3

BUILDING_ID
-----------
BUILDING_NAME
--------------------------------------------------------------------------------
PURPOSE              LOCATION_ID
-------------------- -----------

          4
Warner Bros. Studios
studio                         4

          5
Fox Studios

BUILDING_ID
-----------
BUILDING_NAME
```

3)ActsIn Table:

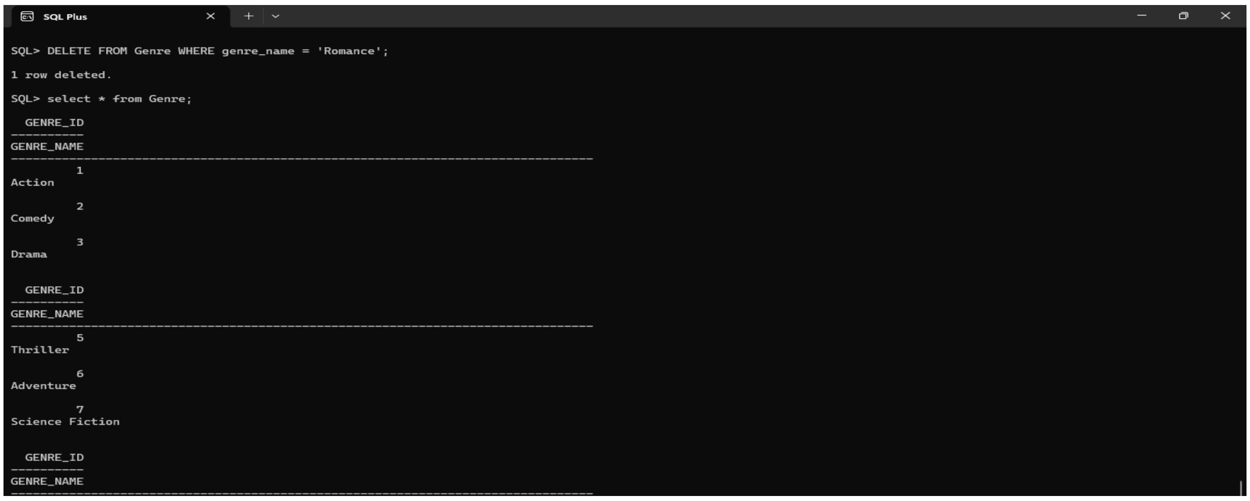
DELETE FROM ActsIn WHERE movie_id = 12;



```
SQL> DELETE FROM ActsIn WHERE movie_id=12;

1 row deleted.

SQL> select * from ActsIn;

  MOVIE_ID  ARTIST_ID
---------- ----------
         2          2
         3          3
         4          4
         5          5
         6          6
         7          7
         8          8
         9          9
        10         10
         7         14
        19         13

  MOVIE_ID  ARTIST_ID
---------- ----------
         2         12
        15         13
         4         10
         8         11
        18         12
        13         12
         5         12
        16          2
        16          3
        16          4
        16          5

  MOVIE_ID  ARTIST_ID
---------- ----------
        16          8
        16         12
```

4)Genre table:

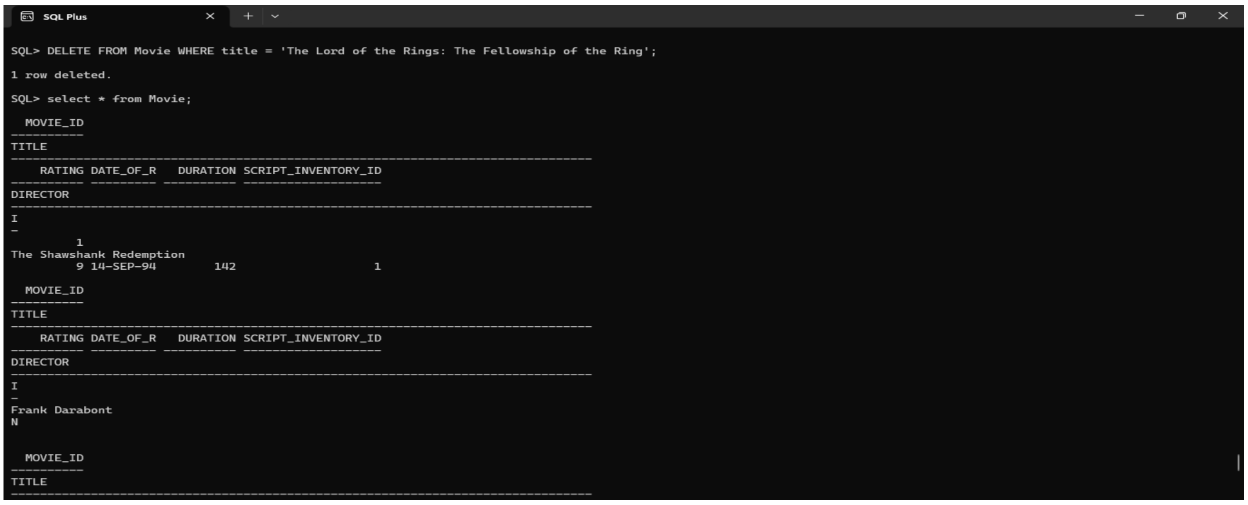
DELETE FROM Genre WHERE genre_name = 'Romance';



```
SQL> DELETE FROM Genre WHERE genre_name = 'Romance';

1 row deleted.

SQL> select * from Genre;

  GENRE_ID
----------
GENRE_NAME
--------------------------------------------------------------------------------
         1
Action

         2
Comedy

         3
Drama


  GENRE_ID
----------
GENRE_NAME
--------------------------------------------------------------------------------
         5
Thriller

         6
Adventure

         7
Science Fiction


  GENRE_ID
----------
GENRE_NAME
--------------------------------------------------------------------------------
```

5)Movie table:

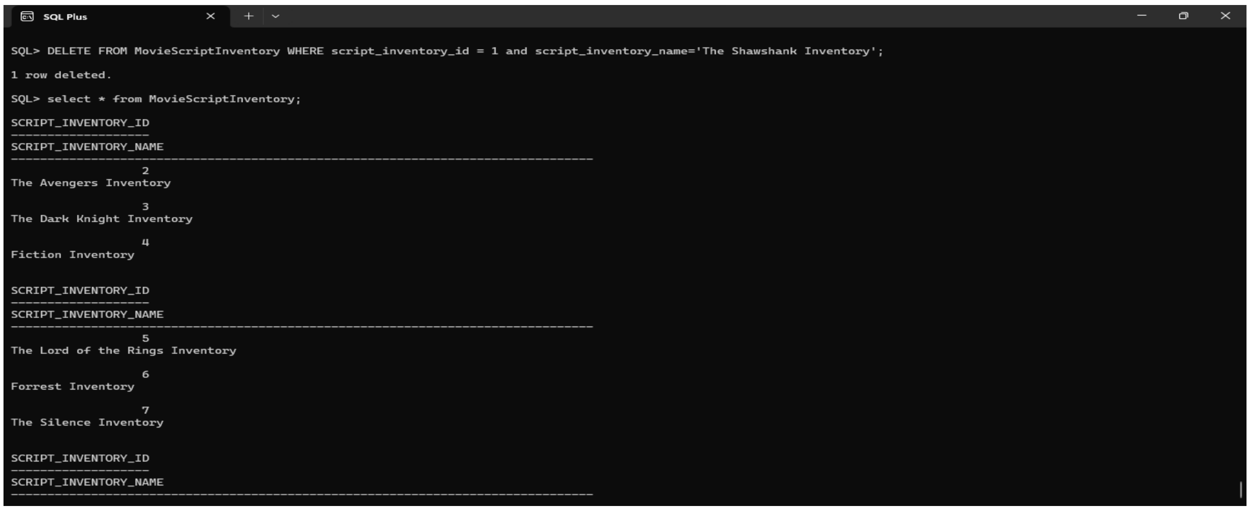
DELETE FROM Movie WHERE title = ‘The Lord of the Rings: The Fellowship of the Ring’;



```
SQL> DELETE FROM Movie WHERE title = 'The Lord of the Rings: The Fellowship of the Ring';

1 row deleted.

SQL> select * from Movie;

  MOVIE_ID
----------
TITLE
--------------------------------------------------------------------------------
    RATING DATE_OF_R   DURATION SCRIPT_INVENTORY_ID
---------- --------- ---------- -------------------
DIRECTOR
--------------------------------------------------------------------------------
I
-
         1
The Shawshank Redemption
         9 14-SEP-94        142                   1

  MOVIE_ID
----------
TITLE
--------------------------------------------------------------------------------
    RATING DATE_OF_R   DURATION SCRIPT_INVENTORY_ID
---------- --------- ---------- -------------------
DIRECTOR
--------------------------------------------------------------------------------
I
-
Frank Darabont
N


  MOVIE_ID
----------
TITLE
--------------------------------------------------------------------------------
```

6)MovieScriptInventory:

DELETE FROM MovieScriptInventory WHERE script_inventory_id = 1 and script_inventory_name = ‘The Shawshank Inventory’;



```
SQL> DELETE FROM MovieScriptInventory WHERE script_inventory_id = 1 and script_inventory_name='The Shawshank Inventory';

1 row deleted.

SQL> select * from MovieScriptInventory;

SCRIPT_INVENTORY_ID
-------------------
SCRIPT_INVENTORY_NAME
--------------------------------------------------------------------------------
                  2
The Avengers Inventory

                  3
The Dark Knight Inventory

                  4
Fiction Inventory


SCRIPT_INVENTORY_ID
-------------------
SCRIPT_INVENTORY_NAME
--------------------------------------------------------------------------------
                  5
The Lord of the Rings Inventory

                  6
Forrest Inventory

                  7
The Silence Inventory


SCRIPT_INVENTORY_ID
-------------------
SCRIPT_INVENTORY_NAME
--------------------------------------------------------------------------------
```

Individual Contribution

1. Added director attribute for Movie entity.

2. Wrote query to find the total number of movies group by director released after June 2nd, 2021.

3. Designed and solved query to retrieve script name and movie name of films with rating greater than or equal to 9 and display result in descending order.

4. Designed and solved query to retrieve the song and singer name of movie with id 3.

5. Wrote and solved query for updating artist name, address using artist_name.

6. Wrote and solved query for deleting song from Songs table using song name.

7. Added hours_worked attribute to Payroll to track hours worked during each date.

8. Added Work date attribute, that tracks hours worked on the recorded date in Payroll table