CSE 4308

DATABASE MANAGEMENT SYSTEMS LAB

NAME: Nazia Karim Khan Oishee
Student ID: 200042137

Introduction

In the fourth lab of Database Management Systems our goal was to learn and have a grip over advanced data manipulation. We were to write all the given SQL statements in an editor first and save them with .sql extension. Then execute the script.

Task

We were given 6 interrelated tables named Actor ,Director, Direction, Casts, Movie, Reviewer, Rating. We had to access data from the table using DML.

1. Find the name of the actors/actresses that are also directors (with and without 'intersect' clause).

Solution: Without 'intersect'

```
SELECT DISTINCT ACTOR.ACT_FIRSTNAME,ACTOR.ACT_LASTNAME

FROM ACTOR,DIRECTOR

WHERE ACT_FIRSTNAME=DIR_FIRSTNAME AND ACT_LASTNAME=DIR_LASTNAME ;
```

With 'intersect'

```
(SELECT ACT_FIRSTNAME,ACT_LASTNAME

FROM ACTOR)

INTERSECT

(SELECT DIR_FIRSTNAME, DIR_LASTNAME

FROM DIRECTOR) ;
```

2. Find the list of all the first names stored in the database.

Solution:

```
(SELECT ACT_FIRSTNAME FROM ACTOR)
```

```
UNION (SELECT DIR_FIRSTNAME FROM DIRECTOR);
```

3. Find the movie titles that did not receive any ratings (with and without ‘minus’ clause).

Solution: Without ‘minus’ clause

```
SELECT MOV_TITLE

FROM MOVIE

WHERE MOV_ID NOT IN (SELECT MOV_ID FROM RATING);
```

With ‘minus’ clause

```
SELECT MOV_TITLE

FROM MOVIE

WHERE MOV_ID IN

(SELECT MOV_ID FROM MOVIE MINUS SELECT MOV_ID FROM RATING);
```

4. Find the average rating of all movies.

Solution:

```
SELECT AVG(REV_STARS) FROM RATING;
```

5. Find the minimum rating for each movie and display them in descending order of rating.

Solution:

```
SELECT MOV_ID,MIN(REV_STARS)

FROM RATING

GROUP BY MOV_ID

ORDER BY MIN(REV_STARS) DESC;
```

6. Find the last name of actors/actresses and the number of ratings received by the movies that they played a role in.

Solution:

```
SELECT DISTINCT ACTOR.ACT_LASTNAME,RATING.REV_STARS
```

```
FROM CASTS,RATING,ACTOR

WHERE CASTS.MOV_ID = RATING.MOV_ID AND CASTS.ACT_ID=ACTOR.ACT_ID;
```

7. Find the last name and average runtime of movies of different actors/actresses. Do not include any actor/actress who worked with 'James Cameron' (with and without 'having' clause).

Solution:Without 'having' clause

```
SELECT ACT_LASTNAME,avg(MOV_TIME)
FROM((((ACTOR NATURAL JOIN CASTS) NATURAL JOIN MOVIE) NATURAL JOIN (DIRECTOR NATURAL JOIN
DIRECTION))
WHERE DIR_FIRSTNAME<>'James' AND DIR_LASTNAME<>'Cameron'
GROUP BY ACT_LASTNAME;
```

With 'having' clause

```
SELECT distinct ACT_LASTNAME,avg(MOV_TIME)

FROM(((ACTOR NATURAL JOIN CASTS) NATURAL JOIN MOVIE) NATURAL JOIN (DIRECTOR
NATURAL JOIN DIRECTION))

GROUP BY ACT_LASTNAME

HAVING DIR_FIRSTNAME<>'James' AND DIR_LASTNAME<>'Cameron';
```

8. Find the first name and last name of the director of the movie having the highest average rating (with and without 'all' clause).

Solution:Without 'all' clause

```
SELECT DIRECTOR.DIR_FIRSTNAME, DIRECTOR.DIR_LASTNAME

FROM ((DIRECTOR NATURAL JOIN DIRECTION) NATURAL JOIN MOVIE)

WHERE MOV_ID IN

(SELECT TOP (MOV_ID), AVG(REV_STARS) AS AVERAGE_REVIEW

FROM RATING

GROUP BY MOV_ID

ORDER BY AVG(REV_STARS) DESC);
```

With ‘all’ clause

```
SELECT DIR_FIRSTNAME, DIR_LASTNAME

FROM ((DIRECTOR NATURAL JOIN DIRECTION) NATURAL JOIN MOVIE)

WHERE MOV_ID IN

(SELECT MOV_ID, AVG(REV_STARS)

FROM RATING

HAVING AVG(REV_STARS) >= ALL(SELECT MOV_ID, AVG(REV_STARS)

FROM RATING

GROUP BY MOV_ID));
```

9. Find all the movie related information of movies acted and directed by the same person.

Solution:

```
SELECT DISTINCT
MOVIE.MOV_ID,MOV_TITLE,MOV_YEAR,MOV_LANGUAGE,MOV_RELEASEDATE,MOV_COUNTRY

FROM MOVIE,DIRECTION

WHERE DIR_ID IN

(SELECT DIR_ID FROM DIRECTOR,ACTOR WHERE DIR_FIRSTNAME=ACT_FIRSTNAME);
```

10 Find the title and average rating of the movies that have average rating more than 7 (with and without using ‘having’ clause).

Solution: With ‘having’ clause

```
SELECT MOV_TITLE,REVIEW

FROM MOVIE NATURAL JOIN (SELECT MOV_ID,avg(REV_STARS)REVIEW FROM MOVIE NATURAL
JOIN RATING GROUP BY MOV_ID)

GROUP BY REVIEW,MOV_TITLE HAVING REVIEW>7;
```

Without ‘having’ clause

```
SELECT MOV_TITLE,REVIEW FROM
MOVIE NATURAL JOIN (SELECT MOV_ID,avg(REV_STARS)REVIEW FROM MOVIE NATURAL JOIN
RATING GROUP BY MOV_ID) WHERE REVIEW>7;
```

11.Find the title of the movies having average rating higher than the average rating of all the movies.

Solution:

```
SELECT MOV_TITLE,avg(REV_STARS)REVIEW

FROM MOVIE NATURAL JOIN RATING

GROUP BY MOV_ID,MOV_TITLE HAVING avg(REV_STARS)>(SELECT avg(REV_STARS) FROM MOVIE
NATURAL JOIN RATING);
```

12. Find the title and average rating of the movies without using the group by statement.

Solution:

```
SELECT MOV_TITLE,MOV_ID

FROM MOVIE NATURAL JOIN (SELECT MOV_ID, AVG(REV_STARS)

FROM RATING GROUP BY MOV_ID);
```

13.Find the actresses with the same first name.

Solution:

```
SELECT ACT_FIRSTNAME
FROM ACTOR WHERE ACT_GENDER='F'
GROUP BY ACT_FIRSTNAME HAVING count(*)>1;
```

14.Find the title and maximum rating of the movies that have at least 10 reviews and has a female actress. One of the reviewers of the movie should be ‘Neal Wruck’. Do not include any movie that received less than 4 stars rating or any movies from directors that have did not direct more than one movie.

Solution:

```
SELECT MOV_ID,MOV_TITLE,MAX(REV_STARS) FROM ( ((SELECT * FROM ACTOR WHERE
ACT_GENDER='F') NATURAL JOIN CASTS) NATURAL JOIN (
(SELECT MOV_TITLE,MOV_ID FROM MOVIE NATURAL JOIN RATING GROUP BY MOV_ID,MOV_TITLE
HAVING COUNT(*)>=10)

NATURAL JOIN(  DIRECTION NATURAL JOIN
(SELECT DIR_ID FROM DIRECTOR NATURAL JOIN DIRECTION GROUP BY DIR_ID HAVING
COUNT(DIR_ID)>=1)  )) NATURAL JOIN (
(SELECT * FROM RATING NATURAL JOIN REVIEWER WHERE REV_NAME='Neal Wruck' AND
REV_STARS>4)) )

GROUP BY MOV_ID,MOV_TITLE;
```

Conclusion

It required a great depth of knowledge on data manipulation to complete all the tasks. I faced many errors and it took a lot of time to complete all the tasks. By completing the tasks I learned a lot about DML.