Department of Computer Engineering
University of Peradeniya
Lab 05

CO226 - Database Systems

December 11, 2018

1 Aim

The aim of this lab is to provide students hands on experience to write sql queries in MySQL.

2 Objectives

At the end of this lab, students should be able to write nested sql queries in MySQL and retrieve some useful information about the data in the database.

3 Lab Task

Using the movie-rating database you have created in the previous lab, you have to retrieve some useful information using select, update and delete queries.
Figure01 shows a certain instance of the populated database.
Log into MySQL server and use the database you have created in the previous lab named as E15XXXlab04 .

MOVIE

Movie ID	Title	Year	Director
101	Gone with the Wind	1939	Victor Fleming
102	Star Wars	1977	George Lucas
103	The Sound of Music	1965	Robert Wise
104	E.T.	1982	Steven Spielberg
105	Titanic	1997	James Cameron
106	Snow White	1937	NULL
107	Avatar	2009	James Cameron
108	Raiders of the Lost Ark	1981	Steven Spielberg

REVIEWER

Reviewer ID	Reviewer Name
201	Sarah Martinez
202	Daniel Lewis
203	Brittany Harris
204	Mike Anderson
205	Chris Jackson
206	Elizabeth Thomas
207	James Cameron
208	Ashley White

RATING

Reviewer ID	Movie ID	Stars	Rating Date
201	101	2	2011-01-22
201	101	4	2011-01-27
202	106	4	null
203	103	2	2011-01-20
203	108	4	2011-01-12
203	108	2	2011-01-30
204	101	3	2011-01-09
205	103	3	2011-01-27
205	104	2	2011-01-22
205	108	4	null
206	107	3	2011-01-15
206	106	5	2011-01-19
207	107	5	2011-01-20
208	104	3	2011-01-02

Figure 1: An instance of ' Movie Rating' database

Write the following SQL queries using MySQL, to retrieve the data from the database you created in lab04 .

1. Write a nested query to list the details of the movies directed by a director,

   (a) who is also a reviewer. (1 mark)

   (b) who is not a reviewer. (1 mark)

2. Write a nested query to list the details of the movie ratings,
   (a) reviewed by the reviewer Sarah Martinez. (1 mark)
   (b) not reviewed by the reviewer Sarah Martinez. (1 mark)
3. Write a nested query to list the **movie ids** where each movie has some rating
   (a) less than to any of the ratings received by the movie which has a **movie id** equal to **103**. (1 mark)
   (b) less than or equal to any of the ratings received by the movie which has a **movie id** equal to **103**. (1 mark)
   (c) equal to any of the ratings received by the movie which has a **movie id** equal to **103**. (1 mark)
   (d) greater than to any of the ratings received by the movie which has a **movie id** equal to **103**. (1 mark)
   (e) greater than or equal to any of the ratings received by the movie which has a **movie id** equal to **103**. (1 mark)
   (f) not equal to any of the ratings received by the movie which has a **movie id** equal to **103**. (1 mark)
4. Write a nested query to list the reviewer ids who has the same (**movie id**, stars) combination on some movie which has a rating date equal to **2011-01-12**. (5 marks)
5. Find all the years that have a movie that received a rating of 4 or 5 and sort them in increasing order of the year. Write,
   (a) a non-nested query. (5 marks)
   (b) a non-correlated nested query. (5 marks)
6. Find the titles of all movies that have no ratings. Write,
   (a) non-correlated nested query. (5 marks)
   (b) a correlated nested query. (5 marks)
7. Some reviewers did not provide a date with their rating. Find the names of all reviewers who have a NULL value for the date. Write,
   (a) a non-nested query. (5 marks)
   (b) a non-correlated nested query. (5 marks)
   (c) a correlated nested query. (5 marks)
8. For each movie that has some rating, find
   (a) the highest **stars** value received. (2 marks)
   (b) the least **stars** value received. (2 marks)
   (c) the average value of **stars** received. (2 marks)
   (d) the sum of all the **stars** received. (2 marks)
   (e) the number of times each movie was rated. (2marks)

In each of the above cases, return the movie title and asked stars value. Sort the results by movie title.

9. Find the names of all the reviewers who have contributed three or more ratings. Write,

   (a) a non-nested query. (5 marks)

   (b) a non-correlated nested query. (5 marks)

   (c) a correlated nested query. (5 marks)

10. Select **movie ids** of each movie with its **title**, **reviewer id** and **stars** received.

11. List the **movie titles** and average **ratings**, from the highest-rated to lowest-rated. If two or more movies have the same average rating, list them in alphabetical order. (5 marks)

12. Remove all ratings where the movie's year is before 1970 or after 2000. (5 marks)

13. Remove all ratings where the rating date is NULL. (5 marks)

14. Insert 5-star ratings by James Cameron for all movies in the database. Leave the review date as NULL. (5 marks)

15. For all movies that have an average rating of 4 stars or higher, add 25 to the release year. (Update the existing tuples. Do not insert new tuples). (5 marks)

4 Submission

Submit the queries and results of lab task in a text file named E15XXXLab05.txt

5 Deadline

The deadline for the submission is Wednesday(19th of December) 23:55h.