COEN 280 - Database Systems
Winter 2019
Homework Assignment 2
Due: Friday, Feb 8 @11:59pm

Part 1: Map the EER diagram into Oracle RDBMS model (20 points)
Convert your EER conceptual schema into tables and then implement these tables in the Oracle database. You can change your EER design freely during you conversion since your schema might not optimal. You will get full credit for part 1 and part 2 if your query is working properly.

Note: You are required to populate your database with the given data and test data with the queries in Part2. Use the excel data files for this. The excel file has tabs in the bottom for each data section. Also, go through the queries in part 2 to make reasonable assumptions regarding the attributes unavailable in the files provided and fill them out.

IMPORTANT Notes:

▪ **The following procedure must be followed to access the Oracle database server:**

//$ is the system prompt

$ sqlplus //Use sqlplus to issue sql statements

▪ **Reduction Guidelines for Oracle RDBMS:**

- Do not use triggers.
- Use reference for foreign keys and specify what action should be taken in case of update and/or deletion of the referenced tuple/row (i.e., cascade, reject, or setdefault/null).

▪ **Reference**:

Refer to Oracle SQL reference manual for information on how to create tables, indexes, insert data, etc. (http://docs.oracle.com/cd/E11882_01/server.112/e41084.pdf).

Part 2: Queries on the database (100 points)

1. Print names of the cast of the movie *"The Davinci Code"* in ascending alpha order.
2. Print all information (mid, title, year, num ratings, rating) for the movie(s) with the highest rating (include all that tie for first place). Order by ascending mid.
3. Print all information of the movies that have both a) the highest number of ratings; and b) the lowest average.
4. A decade is any sequence of 10 consecutive years (e.g., 2000, 2001, …, 2010 is a decade). Find the decade with the largest number of films (output only the first year of the decade).
5. Find the film(s) with the largest cast. Return the movie title and the size of the cast. By "cast size" we mean the number of distinct actors that played in that movie: if an actor played multiple roles, or if the actor is simply listed more than once in CASTS, we still count her/him only once.
6. Find actors that played five or more *distinct* roles in the same movie during the year 2010. Write a query that returns the actors' names, the movie name, and the number of distinct roles that they played in that movie (which will be ≥ 5).
7. Print the movie year, title and rating of the highest rated movie for each years in the period 2005-present, inclusive, in ascending year order. In case of a tie, print all, sorted in ascending alpha order on the title.
8. Find out who are the "no flop" actors: we will define a "no flop" actor as one who has played only in movies which have a rating greater than or equal to 8. Split this problem into the following steps.
    a. Create a view called high ratings which contains the distinct names of all actors who have played in movies with a rating greater than or equal to 8. Similarly, create a view called low ratings which contains the distinct names of all actors who have played in movies with a rating less than 8. Print a) the number of rows in the view high ratings and b) the number of rows in the view low ratings
    b. Use the above views to print the number of "no flop" actors in the database.
    c. Finally, use the above view to print the names of these "no flop" actors, along with the number M of movies they have played in, sorted by descending M and then by ascending name, and print only the top 10.
9. Print the names of all actors who have starred in all movies in which *Al pacino* has starred in (it's ok to report the name of *Al pacino* in the result; also, it is ok if these actors have starred in more movies than *Al pacino* has played in).
10. Find out who is the actor with the highest "longevity." Print the name of the actor/actress who has been playing in movies for the longest period of time (i.e. the time interval between their first movie and their last movie is the greatest.

Submission Guidelines

1. Your submission of part 1 and part 2 should include one createdb.sql file, one dropdb.sql file, ten .sql files for queries described in part 2 (named q1.sql to q10.sql), and one readme.txt file.

2. **createdb.sql** file should create required types, tables, indexes if required, generate primary keys, constraints, ... , and populate all the provided data based on the skeleton data provided. There is **60 points penalty** if this file is missing since it is not possible

for us to check your queries without any data.

3. The **dropdb.sql** file should drop all types and tables that are created by createdb.sql. There is **10 points penalty** if this file is missing from your submission or if it does not drop all of your database objects.

4. **q1.sql ~ q10.sql** query files should contain SQL statements for queries Q1 to Q10 described in part 2 respectively. If you need to write two or more SQLs for ONE step, then they should be written after each other in ONE file.

5. Make sure to properly test created.sql, dropdb.sql and the query files (q1.sql…q10.sql) before submission. There will be **penalty** for resubmission if one the mentioned files do not execute properly.

6. The **readme.txt** file must have your name, the name of the database, tables that your createdb.sql file generates and the execution result of query files (q1.sql…q10.sql). There is **10 points penalty** if this file or some of the required information is missing from your submission.

7. You must make a .zip file to include all of your files in one file (<your_name>_hw2.zip:
   Your zip file should contain **createdb.sql dropdb.sql readme.txt q1.sql q2.sql q3.sql q4.sql q5.sql q6.sql q7.sql q8.sql q9.sql q10.sql** files.

8. You need to submit the 1st and 2nd part of your assignment to Camino

9. Start working on your assignment early.