CS 34800 Project

Due Date: Monday, 27th March, 2017 (submit before 11:59 PM)

Part A: Database Design (30 points)

1. **(E-R diagram – 15 points)** Design an E-R diagram for the following scenario. State any additional assumptions that you make. Be sure to indicate any key and participation constraints.

   A company wants to design a system to manage the information about recruiting, and you have been selected for this job. Your task is to design the database that captures all the information that needs to be maintained. The relevant information is as follows:

   A. The database should keep the information about candidates, positions, groups, and interviews.
   B. Each candidate can only apply to one position at a time, and can re-apply in the future if he/she fails. Each application and its outcome must be recorded.
   C. Each group can have several open positions, and the open positions will be closed when candidates pass the interviews and accept the offers.
   D. Each candidate can be interviewed by several interviewers, and the interviewers and their feedbacks need to be recorded.
   E. Employees can refer a candidate to an open position, they will get bonus if the candidate gets hired. Such information needs to be recorded, if the candidates are referred by an employee.

2. **(E-R to Relational – 15 points)** Convert the E-R model designed in 1. to its corresponding relational model, and provide SQL statements to create the tables in the database (be sure to include primary key/foreign key and any other constraints).

Part B: SQL Queries, Integrity Constraints (70 points)

Introduction

Each student has to do this project individually. You are going to use Oracle to perform some queries and create views for a database. The schema and sample data of the database are provided. The project should be run on CS sun workstations.

Your login information can be found on my.cs.purdue.edu, under the *Local Accounts* page. Information about general initial configuration is available in: https://www.cs.purdue.edu/resources/facilities/oracle.html

Your assignment

In this project you will use the file: db.sql. Copy this file into your working directory. Create and populate tables by the following command:

```
SQL>@db
```

File db.sql will create the tables needed for this assignment. It will also fill the tables with some sample data. This will help you test your queries. The current data does not cover all the possible scenarios that the queries address and they are not completely matching the examples shown in this handout. So, feel free to add additional tuples to test some corner cases. **Note that during grading, the TAs will use the same schema with different data.**

The following section shows the schema of the database.

Consider the following relations pertaining to an online movie-booking database system. The underlined attributes indicate the primary key for each relation.

Users (userID, email, age, numberBookedSoFar)

Theaters (theaterID, name, address)

Movies (movieID, title, genre, year, runtime)

Showtimes (showID, movieID, theaterID, starttime, endtime, hall, max_occupancy)

Tickets (userID, showID)

– Relation Users contains information about each user. The *age* attribute is either ‘Teen’, ‘Young Adult’, ‘Adult’, ‘Middle-Aged’ or ‘Senior’ (their ages are <18, 18-25, 25-40, 40-60, >60 respectively). The *numberBookedSoFar* attribute indicates the number of movies a user has booked through the system since joining.

– Relation Theaters contains information about the theaters.

– Relation Movies contains information about each movie in the system. The runtime of the movie indicates its length.

– Relation Showtimes contains information on show timings of movies across all theaters. The *movieID* and *theaterID* attributes are foreign keys referencing Movies and Theaters respectively. The *starttime* and *endtime* attributes indicate the start time and end time of each show. The *hall* attribute indicates the hall in a theater that a movie is currently being screened on. The *max_occupancy* attribute indicates the limit on the number of people that can be present in the hall at any time.

– Relation Tickets contains information about which users booked which shows. The *userID* and *theaterID* attributes are foreign keys referencing Users and Theaters.

Answer the following based on the above schema:

1. **(SQL Queries – 30 points)** Write SQL queries (one SQL statement per question) that answer the questions below:

   1.1. Find the names of the movies that are currently running house-full (i.e., the halls screening the movies have reached their full occupancy)
   1.2. Display the names of theaters screening 3 or more shows; also show the number of shows each such theater is screening.
   1.3. Find the name and id of the theaters which have at-least one show for every movie.
   1.4. Find the email ids and age groups of users who have booked tickets for only one show currently, i.e., of all movies being screened currently, these users are going to only one movie. Also, for each such user, display the name of the movie they have booked their tickets for.
   1.5. Find the names of the movies where there exists at least one show that only has users belonging to one age group.
   1.6. Find the name(s) of the theater(s) that have a show with the largest number of bookings by users. If there are ties, display all theaters that qualify.
   1.7. For each user, display their name, email id and the theater in which he/she has watched most shows.
   1.8. For each movie, display its name as well as the average number of users who have currently booked tickets for a show screening the movie.
   1.9. For every show, find the number of seats left in the show.
   1.10. Find the name of the users and the genre for the users who only watch movies from one genre.

2. **(PL/SQL – 15 points)** Write the following procedures in PL/SQL. (In sqlplus command mode, type “SET SERVEROUTPUT ON SIZE 32000” to enable printing).

   2.1. (5 points) Write a procedure that generates information about the users of the database:

```
AgeGroup      NumberOfUsers      AvgBooked
------------------------------------------------------------------------------------------
Teen           25                1.42
Young Adult    300               5.19
Adult          …                 …
Middle-aged    …                 …
Senior         …                 …
```

   This procedure gives a list of information of each age group in the database. The NumOfUsers indicates the number of users of each age group. AvgBooked indicates the average number of all shows booked by users in that age group.

2.2. (10 points) Write a procedure that generates the following report about the showtimes and booking information:

```
Theater: Wabash Landing
MovieTitle     StartTime     EndTime       Hall
-------------------------------------------------------------------------------------------
Zootopia       9:30 AM       11:00 AM      5

               UserID        Email                         AgeGroup
               ----------------------------------------------------------------------
               1             mwayne@gmail.com              Young Adult
               …
               …
               …

MovieTitle     StartTime     EndTime       Hall
-------------------------------------------------------------------------------------------
La La Land     12:00 PM      14:00 PM      4

               UserID        Email                         AgeGroup
               ----------------------------------------------------------------------
               9             kathy@purdue.edu              Adult
               …
               …

Theater: Goodrich Eastside
…
………
```

This procedure generates a report of all theaters, the shows they screen, and the users who have booked those shows.

3. **(Triggers – 15 points)** Implement triggers to enforce the following constraints:
   3.1. (7.5 points) No two movies can be screened in the same hall at the same time.
   3.2. (7.5 points) A hall screening a movie cannot have more than the maximum number of people allowed into the hall at a time.

4. **(JDBC – 10 points)** Write a Java application to access your database using JDBC API. Your application should take the movie id (e.g. MV123) as an input and output the following:
   4.1. (5 points) Information on show times (movie title, theater name, theater address, hall, start time, end time.) for the movie specified in the input.
   4.2. (5 points) Number of users from each age group who have booked tickets for the movie specified in the input

Drop

Use the command DROP to drop all procedures and tables. Use statement “select * from user_catalog;” to make sure that all the objects are dropped.

What to submit:

You are required to submit **5** files. Detailed instructions on each file are below:

Part A:

1. E-R diagram: Submit a pdf file named **a1_your_career_login.pdf**. Document all the design decisions you made, the E-R diagram and the relations.
2. E-R to relational: Submit a sql script **a2_your_career_login.sql**. This file should contain all the SQL statements to create the relations in part A-1.

Part B:

1. SQL Queries: Submit a sql script **b1_your_career_login.sql**. Clearly mention the query number in comments.
2. PL/SQL: Submit ONLY ONE file, **b2_your_career_login.sql**. This file should contain all the function and procedure definitions, and also the execute statements to execute the procedures in order.
3. Triggers: Submit ONLY ONE file, **b3_your_career_login.sql**. This file should contain both the triggers, and the execute statements to enable the triggers.
4. JDBC: Submit a Java program, **b4_your_career_login.java**. Include comments on how to compile, run and use your program.

Turning in your files

Please create a *single* zip file that contains the 5 files as mentioned. Please turn in the project by uploading the zip file on Blackboard before the deadline.

We prefer a typed/typeset document for Part A. If you (clearly and readably) handwrite your answer, then turn in a (clear, readable) scan as a PDF. If you do not know of a way to generate a PDF, bring it up to one of the TAs. Do not forget to write your name on your assignment document, otherwise you may lose points.