CS450 - Database Concepts

Fall 2020

Instructor: Dr. Jessica Lin

Project Assignment

General. Your project is to design and implement a database for an online movie streaming company like Netflix. At the back-end, the database manages all of its data in the Oracle database management system. The database stores information about customers, movies, actors, and customer viewing history. At the front-end, you are required to implement a command-line interface to its users (DBMS users) using Java and JDBC (although if you love challenge and want to build a web interface or a GUI, you can do so for extra credit).

The following is the list of "basic" information that the database should store. Note the organization of the list does not necessarily imply that this is how you should design the database. For example, for each movie, you have a list of items that needs to be stored. It is up to you to decide how you should store and organize these items. You will need to come up with a design that makes sense, given the descriptions.

Everyone should have a partner and work in a team of 2.

1. Members. Store the following for each member:
    a. Unique member ID
    b. Full name (first name, last name)
    c. Credit card on file
    d. Profiles (see below)

2. Movies. For each movie:
    a. Unique movie ID
    b. Movie name
    c. Year it was produced
    d. Cast (i.e. actors in the movie)
    e. Producer
    f. Genre(s)
    g. Average member rating

3. Actors/Actresses. For each actor/actress:
    a. Unique ID
    b. Name (first, last)

4. Each member can have up to 5 profiles in the account for different members in the household[1]. For example, an account owner "princeharry" may have a profile for himself "Harry", another profile "Meghan" for his wife, and a separate profile "Kid." For each profile, we have:
    a. Profile name (not unique across all members, but unique within an account)
    b. Favorite movie genre(s)
    c. Viewing history (see below)

5. There is a separate viewing history for each "profile" created by a member. For the history, we want to store
    a. Movies watched
    b. Ratings given by this profile, if available. Rating scale is from 1 to 5, 1 being "Do not like" and 5 being "Love the movie"

The descriptions above summarize the minimal information your database should contain. As mentioned, it does not necessarily imply the organization of the relations. You may re-arrange the attributes or add more attributes (exception: you should not add any more global unique identifiers unless you have a good reason to). The design decision is yours, but your design should be reasonably efficient, and well justified. <u>You will be graded on both correctness and quality of the design.</u>

Your interface should have minimal functionality including a variety of queries and browsing capabilities over the movies (e.g. the database user should be able to search by genre, movie name, movie star, etc.), profiles, viewing history, etc. The user should be able to update the database by inserting or deleting tuples.

You'll be graded for each phase. The project accounts for 20% of your total grade.

Phase 1: Conceptual Design. (5%) You need to submit an ER diagram for your database according to the project description and the assumption you make. (See HW1 assignment for an example).

Phase 2: Schema Design. (5%) After your conceptual design is finished, you need to translate the ER diagram into relation schema. Submit a copy of your relation schema, the SQL script that you use to implement your database, and insert enough tuples in **each** table, e.g. at least 5 accounts (with at least 2 profiles in each account), 10 movies, 20 actors/actresses, etc.

[1] Separate profiles and rental history allows more accurate recommendation to its members.

Phase 3: JDBC Implementation. (7%) Write a program in JDBC that will allow users to access and query your database. Command-line menu is acceptable.

Extra credit (2%): Instead of command-line interface, you can create a graphical user interface (GUI) or a web app for extra credit, e.g. have a login page, search bar/menu, etc.

Putting it together. (3%) Prepare a final report and a demo video.

Your report should contain the following:

- Materials from all phases
- README describing how to run your program
- Screenshots on program/query execution
- Write-up: Discuss and justify your design decisions and challenges you encountered. List any part(s), if any, that do not work properly or you did not implement.
- Contribution of each member in the team. Do both members contribute equally on the project?

The demo video should demonstrate the features of your program: insertion, deletion, update, search. Max 5 minutes.

Timeline:

October 2: project released

October 19: Phase 1 (ER diagram) due

November 6: Phase 2 (schema) due

December 7: Phase 3 (including final report and video) due

Information for Project Phase 3:

To create and populate your database, run your SQL script from Phase 2. Make sure that in the beginning of the script, you **delete all tables that you are about to create**.

Instead of having command-line menu, you can create a GUI for extra credit (2%).

Your program should allow the user to do the following. Note, the "user" here means the DBMS user, not the website user.

Optional: If you want to be extra fancy, you can create a separate interface for the streaming service user (think about Netflix interface, which allows a member to specify a profile, browser for movies, view streaming history, watch a movie, and rate a movie he/she has watched). Note the streaming service user does not directly alter the database. Instead, his/her actions indirectly update the database (e.g. by watching or rating a movie, a record will be inserted into the appropriate table(s). The streaming service user would not be able to alter movie-related tables.

You're only required to make the interface for the DBMS user, who has access to manipulate and query from all relations.

- View table content: Give user a list of existing tables so that he/she can select one to view the tuples.
- Add records: Enter information for new members, movies, actors, etc. Update/delete information.
- Search database: Allow users to search for movies based on title or actor; display all matching movies (movie name, year, and average rating) with their average ratings. You should use partial matching instead of exact matching for strings.
- Show rental history for a given profile (you'll need to ask for account info first, and then profile).
- Exit the program

Here is a suggestion on what your (command-line) menu should look like:

When the program starts, prompt the user for his/her login and password. This is the Oracle login/password. Please do not hard-code your own login information in the code. Once the program connects to the database successfully, display the following menu options. After each selection is executed, your program should go back to the main menu.

1. View table content – list your tables for the user to choose, then upon user selection, display the content (tuples) in the table
2. Insert new record into… (the exact attributes depend on your database design) The design is up to you – you could follow the attribute-by-attribute steps as described below once the user selects a table to insert into, or you could list all the attributes and accepts a string containing all attribute values (delimited by commas). When inserting a new rating, your trigger should update the average rating for the given movie. Also make sure that the number of profiles per account does not exceed 5.
    a. Account
        i. Member ID
        ii. First name
        iii. Last name
        iv. Etc.
    b. Movie
        i. Movie ID
        ii. Movie name

        iii. Etc.
    c. Actor
        i. Actor ID
        ii. First name
        iii. Last name
        iv. Etc.
    d. Profile
        i. Member ID
        ii. Profile name
    e. History
        i. Member ID
        ii. Profile name
        iii. Movie ID
        iv. rating
        v. Etc.

    There might be more tables depending on your design.

3. Update record
    a. Update (then ask for information to update)
    b. Delete (then ask for record to delete)
4. Search for movies
    a. Movie name
    b. Actor name
5. Show information for a member's profile.
    a. Prompt for member ID and then profile name
    b. See rental history
6. Exit