COEN 280 - Database Systems
Fall 2018
Homework Assignment 3
Due: Friday, Nov 16
@11:59pm

In your course project you would develop a data analysis application for imdb.com's user review data. The emphasis would be on the database infrastructure of the application.

The dataset used for the project is an extension of MovieLens10M dataset, published by GroupLeans research group, http://www.grouplens.org. The datadset links the movies of MovieLens dataset with their corresponding web pages at Internet Movie Database (IMDb) and Rotten Tomatoes movie review systems.
http://www.imdb.com
http://www.rottentomatoes.com

The dataset includes 2113 users, 10197 movies, 855598 ratings and 13222 tags. The dataset files that you will use in this project are available on Camino.
(Note: Please make sure to use the dataset available on Camin, not the one from the imdb.com website or grouplens,org)

Overview & Requirements:
You would develop a target application which runs queries on the MovieLens/imdb data and extracts useful information. The primary users for this application will be users seeking for movies and their ratings that match their search criteria. Your application will have a user interface that provides the user the available movie attributes such as genre, country, cast, rating, year and user's tags and ratings. Using this application, the user will search for the movies from various categories that have the properties (attributes) the user is looking for.

The user can filter the search results by available movie attributes such as movie title, genre, year, country, all filming locations, average of Rotten Tomato All Critics rating and Rotten Tomato top critics rating and Rotten tomato audience rating, average of Rotten Tomato all Critics number of reviews and Rotten Tomato Top critics number of reviews and Rotten Tomato Audience number of ratings.

You will be designing your application a standalone Java application.

Example screenshots of a possible application are available in Appendix-B. In evaluating your work, instructor's primary focus will be primarily on how you design your database and how efficiently you can search the database and pull out the information. However, your GUI should provide the basic functionality for easy browsing of the movie categories and attributes (as illustrated in Appendix-B). Creativity is encouraged!

Project Details:

0. Part 0

- Install Oracle Database 11gR2 or later. Consult the instructions provided on Camino under Assignment 3. If you are using a MAC laptop, you can install a virtualization software such as Virtual Box, and install a Windows or Linux guest operating system. You can then install Oracle Database on this environment.

I. Part 1

- Download the MovieLens dataset from Camino. Look at each data file and understand what information the data objects provide. Pay attention to the data items in data objects that you will need for your application (For example, movie attributes, etc.)
- You may have to modify your database design from Homework 2 to model the database for the described application scenario on page-1. Your database schema doesn't necessarily need to include all the data items provided in the data files. Your schema should be precise but yet complete. It should be designed in such a way that all queries/data retrievals on/from the database run efficiently and effectively.
- Produce DDL SQL statements for creating the corresponding tables in a relational DBMS. Note the constraints, including key constraints, referential integrity constraints, not NULL constraints, etc. needed for the relational schema to capture and enforce the semantics of your ER design.
- Populate your database with the dataset. Genrete INSERT statements for your tables and run those to insert data into your DB.
- After you populated your database, created indexes on frequently accessed columns of its tables using CREATE INDEX statement. This will help speed up query execution times. You have some flexibility about which indexes to choose.

II. Part 2

Implement the application for searching movies as explained in section "Overview & Requirements". In this milestone you would:

- Write the SQL queries to search your database.
- Establish connectivity with the DBMS.
- Embed/execute queries in/from the code. Retrieve query results and parse the returned results to genrete the output that will be displayed on the GUI.
- Implement a GUI where the user can search for movies that match the criteria given.
    - Browse through attributes for the movies (See Appendix C); select the movie attributes that user wants to search for;
    - The usage flow of the GUI is as follows:

      1) Once the application is loaded, Genres attribute values are loaded from the backend database.

      2) The user is **required** to select desired genres attribute values. Assume that use selects *Drama* as the genres attribute value.

      3) The Countries matching the genres selections will be listed under the Country attribute panel. Since user selected *Drama* in previous step, only country values that their movie genre is Drama should appear in the Country attribute panel. Assume that use selects *USA* as the country attribute value.

      4) The user can select desired Country attribute values. This attribute is optional in building the query. User might not select a country at all.

      5) The Filming Location Country attribute values corresponding to the Country selection will be listed. This attribute is optional in building the query. Since user selected *Drama and USA* in previous step, only filming location values that are USA production **AND** their movie genre is Drama, should appear in the Filming Location Country attribute panel. Assume that use selects *India* as the filming location country.

6) The user specifies Critic's ratings, Number of ratings, and Movie year. These attributes are optional in building the query. Assume that use specifies critic's rating to be equal 7, number of reviews to be greater than 10,000 and movie year to be between 2000 and 2010.

7) The movie tag values corresponding to the previous selections will be listed in the Movie Tag panel. This attribute is optional in building the query. Based on previous selections, tag values corresponding to movies that are USA production **AND** Drama genre **AND** filmed in India **AND** critic rating=7 **AND** number of reviews>10,000 **AND** 2000 < movie year < 2010, should appear in the Movie Tag panel.

- The application should be able to search for the movies that have either all the specified attribute values (AND condition) or that have any of the attribute values specified (OR condition).
  For example, if user selected AND condition, and selected Drama and Family as genre, movies with Drama **AND** Family genres should be listed.
  If user selected OR condition, and selected Drama and Family as genre, movies with Drama **OR** Family genre should be listed.

- Select a certain movie in the search results and list the following for that movie(s): movie title, genre, year, country, all filming locations, average of Rotten Tomato All Critics rating and Rotten Tomato top critics rating and Rotten tomato audience rating, average of Rotten Tomato all Critics number of reviews and Rotten Tomato Top critics number of reviews and Rotten Tomato Audience number of ratings.

Please note that all data displayed on the GUI should be kept in the database and should be retrieved from it when needed. You are not allowed to create internal data structures to store data.

Required .sql files:

You are required to create two .sql files:

1. createdb.sql: This file should create all required tables. In addition, it should include constraints, indexes, and any other DDL statements you might need for your application.
2. dropdb.sql: This file should drop all tables and the other objects once created by your createdb.sql file.

Required Java Programs:

You are required to implement two Java programs:

1. populate.java: This program should get the names of the input files as command line parameters and populate them into your database. It should be executed as:
   "> java populate <filename1.dat> <filename2.dat>…..<filename.dat>".
   Note that every time you run this program, it should remove the previous data in your tables; otherwise the tables will have redundant data.
2. hw3.java: This program should provide a GUI, similar to figure 1, to query your database. The GUI should include:
   a. List of movie genres.
   b. Countries where the movies are produced.
   c. Filming location country where movies are filmed
   d. Critic's rating which is Rotten Tomato all critics rating (rtAllCrtiticsRating)
   e. No. of Reviews, which is the Rotten Tomatoes all critics' number of reviews
   f. Movie year.
   g. Movie tags values
   h. List of results
      i. Results should include movie title, genre, year, country, all filming locations, average of Rotten Tomato All Critics rating and Rotten Tomato top critics rating and Rotten tomato audience rating, average of Rotten Tomato all Critics number of reviews and Rotten Tomato Top critics number of reviews and Rotten Tomato Audience number of ratings.

Appendix-A

MovieLens+ IMDB+ Rotten Tomatoes Dataset

This dataset is an extension of MovieLens10M dataset, published by GroupLeans[1] research group. It links the movies of MovieLens dataset with their corresponding web pages at Internet Movie Database (IMDb)[2] and Rotten Tomatoes movie review systems[3]. From the original dataset, only those users with both rating and tagging information have been maintained.

Data statistics

2113	users
10197	movies
20	movie genres
20809	movie genre assignments (avg. 2.040 genres per movie)
4060	directors
95321	actors (avg. 22.778 actors per movie)
72	countries
10197	country assignments (avg. 1.000 countries per movie)
47899	location assignments (avg. 5.350 locations per movie)
13222	tags
47957	tag assignments (tags), i.e. tuples [user, tag, movie] (avg. 22.696 tags per user, avg. 8.117 tags per movie)
855598	ratings (avg. 404.921 ratings per user, avg. 84.637 ratings per movie)

The dataset includes 10 types of data objects: *movies*, *movie_genres*, *movie_directors*, *movie_actors*, *movie_countries*, *movie_locations*, *tags*, *movie_tags*, *user_taggedmovies*, and *user_ratedmovies*.

The fields of objects are given below:

Movies Objects
Movies objects contain basic information about the movies. The original movie information -title and year- available at MovieLens10M dataset has been extended with public data provided in IMDb and Rotten Tomatoes websites.

```
{
      Movie id
      Title in Spanish
      IMDb movie id
      IMDb picture URL
      Year
      Rotten Tomatoes movie id
      Rotten Tomatoes all critics' rating
      Rotten Tomatoes all critics' number of reviews
      Rotten Tomatoes all critics' number of fresh score
      Rotten Tomatoes all critics' number of rotten score
      Rotten Tomatoes all critics' avg. score
      Rotten Tomatoes top critics' rating
      Rotten Tomatoes top critics' number of reviews
      Rotten Tomatoes top critics' number of fresh score
      Rotten Tomatoes top critics' number of rotten score
      Rotten Tomatoes top critics' score,
      Rotten Tomatoes audience' rating,
```

[1] http://www.grouplens.org
[2] http://www.imdb.com
[3] http://www.rottentomatoes.com

```
	Rotten Tomatoes audience' number of ratings,
	Rotten Tomatoes audience' avg. scores,
	Rotten Tomatoes picture URL
}
```

Movie_genres Objects

This object contains the genres of the movies.

```
{
	movieID
	Genres
}
```

Movie_directors Objects

This object contains the directors of the movies.

```
{
	movieID
	directorID
	directorName
}
```

Movie_actors Objects

This object contains the main actors and actresses of the movies. A ranking is given to the actors of each movie according to the order in which they appear on the movie IMDb cast web page.

```
{
	movieID
	actorID
	actorName
	ranking
}
```

Movie_countries Objects

This object contains the countries of origin of the movies.

```
{
	movieID
	country
}
```

Movie_locations Objects

This object contains filming locations of the movies.

```
{
	movieID
	location1(country)
	location2(state)
	location3(city)
	location4(street)
}
```

Tags Objects

This object contains the set of tags available in the dataset.

```
{
	tagID
	tagText
}
```

Movie_tags Objects

This object contains the tags assigned to the movies, and the number of times the tags were assigned to each movie.

```
{
	movieID
```

```
	tagID
	tagWeight
}
```

User_taggedmovies (and User_taggedmovies-timestamps) Objects

These objects contain the tag assignments of the movies provided by each particular user. They also contain the timestamps when the tag assignments were done.

```
{
	userID
	movieID
	tagID
	timestamp
}
```

User_ratedmovies (and User_ ratedmovies-timestamps) Objects

These objects contain the ratings of the movies provided by each particular user. They also contain the timestamps when the ratings were provided.

```
{
	userID
	movieID
	rating
	timestamp
}
```

Usage of this dataset is governed by the Academic Dataset Terms of Use.

Appendix-B

Sample Application

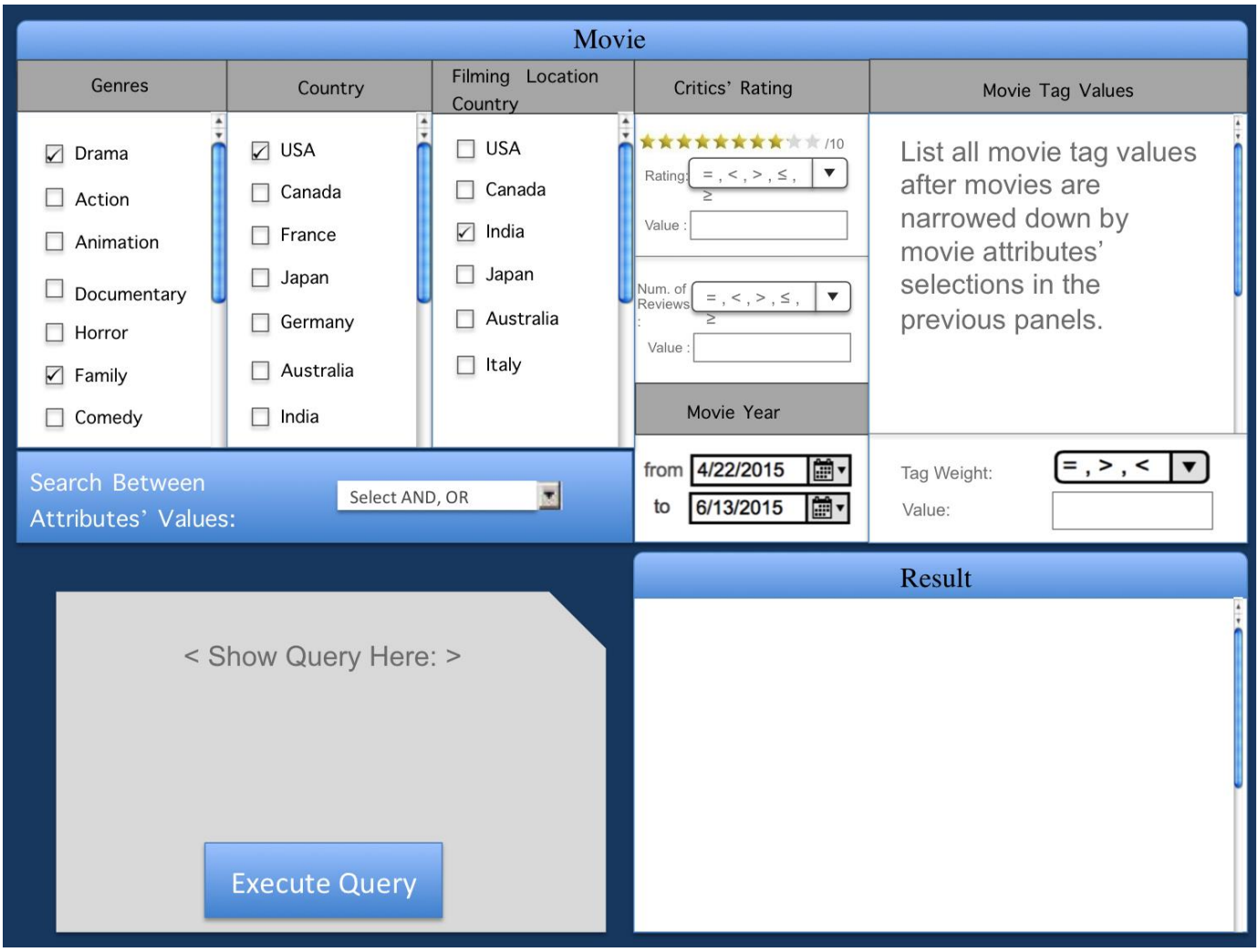

Figure 1- Movie Application Main UI (Movie Search)

Appendix-C

Movie Attributes:

1. Genera
2. Country
3. Filming Location Country
4. Critics' Rating
5. Year
6. Movie Tags

Sample Genera Values:

1. Drama
2. Action
3. Animation
4. Documentary
5. Horror
6. Family
7. …

Sample Country Values:

1. USA
2. Canada
3. Japan
4. Italy
5. India
6. China
7. …

Sample Movie Tag Values:

1. Earth
2. Whale
3. Police
4. Computers
5. Fun
6. Strange
7. Bad acting
8. Bad ending
9. …