**SI 618 Fall 2022 Homework 3 (100 points)**

**Part 1 (80 points)**

The provided ‘movie\_actors\_data.txt’ file contains a JSON string on each line. For example, the first line is:

{"rating": 9.3, "genres": ["Crime", "Drama"], "rated": "R", "filming\_locations": "Ashland, Ohio, USA", "language": ["English"], "title": "The Shawshank Redemption", "runtime": ["142 min"], "poster": "http://img3.douban.com/lpic/s1311361.jpg", "imdb\_url": "http://www.imdb.com/title/tt0111161/", "writers": ["Stephen King", "Frank Darabont"], "imdb\_id": "tt0111161", "directors": ["Frank Darabont"], "rating\_count": 894012, "actors": ["Tim Robbins", "Morgan Freeman", "Bob Gunton", "William Sadler", "Clancy Brown", "Gil Bellows", "Mark Rolston", "James Whitmore", "Jeffrey DeMunn", "Larry Brandenburg", "Neil Giuntoli", "Brian Libby", "David Proval", "Joseph Ragno", "Jude Ciccolella"], "plot\_simple": "Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.", "year": 1994, "country": ["USA"], "type": "M", "release\_date": 19941014, "also\_known\_as": ["Die Verurteilten"]}

The fields we are interested in are imdb\_id, title, rating, genres, actors, directors, year, and country. You will parse the JSON strings, and load the data into three tables in SQLite, and then write SQL queries to retrieve the data specified.

You will create three tables:

* The “movies” table, which has four columns: imdb\_id, title, year, rating, country
* The “movie\_genre” table, which has two columns: imdb\_id and genre. A movie typically has multiple genres, and in this case, there should be one row for each genre. If some movie does not have any genre, ignore that movie.
* The “movie\_actor” table, which has two columns imdb\_id and actor. A movie typically has multiple actors, and in this case, there should be one row for each actor.
* The “movie\_director” table, which has two columns imdb\_id and director. A movie typically has multiple directors, and in this case, there should be one row for each director.

1. (10 points) Parse input file to get needed data for the three tables and load them into appropriate Python data structure.

2. (5 points) Create the movies table and load data into it. Note that you only need to consider the first country if there are multiple countries associated with a movie. Read and show the first 5 rows of the movies table in your SQLite database.

3. (5 points) Create the movie\_genre table and load data into it.

4. (5 points) Create the movie\_actor table and load data into it

5. (5 points) Create the movie\_director table and load data into it

6. (10 points) Write a SQL query to find the number of released movies by decade in chronological order. (Note: you will need to round down years to get the corresponding decade. e.g. 2009 belongs to the 2000s)

7. (10 points) Write a SQL query to find the top 5 genres in the US by the average ratings of movies in that genre. Print out the results

8. (10 points) Write a SQL query to find the top 10 actors who have worked with the director Christopher Nolan in most movies. The actors are ordered in decreasing number of movies they worked with the director. In case of ties, the rows are ordered by actors’ names in alphabetical order.

9. (10 points) Write a SQL query to find the top 10 directors based on average ratings of the movies they directed between the years 1980 and 2010 (begin and end years are included). Only consider those who directed at least 2 movies in this time range. For each director, show their name, average rating of the movies, and the number of movies they directed. Sort the result by descending average movie rating, then by director name in alphabetical order.

10 (10 points) Write a SQL query for finding pairs of actors who co-starred in at least 2 highly rated (rating > 9) movies together. The pairs of names must be unique. This means that ‘actor A, actor B’ and ‘actor B, actor A’ are the same pair, so only one of them should appear.

For each pair of actors you print out, the two actors must be ordered alphabetically. The pairs are ordered in decreasing number of movies they co-stared in. In case of ties, the rows are ordered by actors’ names.

You will need to join the movie\_actor table with itself to get this data (in addition to another required join). It is a bit tricky. If you cannot do it with SQL statement, you can also write some Python code that works on the Python data structure that you used to create the movie\_actor table. That’ll mean much more lines of code, and if you do it that way, you’ll get 5 points instead of 10 points. You will only get 10 points if you solve it with pure SQL.

When you run your Python code, it should print out exactly the output provided in the template notebook.

**Part 2 (20 points)**

Write a python function that takes two arguments: genre and k

The function should print out the top k directors who made the highest-rated movies (on average) in the provided genre. You should use the sqlite3 database file you created in Part 1. Some example runs of my function are shown below. Your program should produce the same output when provided with the same command line arguments.

**What to submit:**

* uniqname\_si618\_hw3.db
* uniqname\_si618\_hw3.ipynb
* uniqname\_si618\_hw3.html