SI 618 Fall 2019 Homework 2 (100 points)

Part 1 (75 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, and year. Use the template notebook file provided to do the homework. 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 “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 “movies” table, which has four columns: imdb_id, title, year, rating

- 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.

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 movie_genre table and load data into it

3. (5 points) Create the movies table and load data into it

4. (5 points) Create the movie_actor table and load data into it

5. (10 points) Write a SQL query to find top 10 genres in the US by the number of movies in that genre and print out the results.

6. (10 points) Write a SQL query to find the average rating of all movies broken down by year in chronological order.

7. (10 points) Write a SQL query to find all Thriller movies from outside the U.S. ordered by decreasing rating, then by increasing year if ratings are the same.

8. (10 points) Write a SQL query to find the top 10 actors based on average movie rating with at least 2 credits in an after year 2000. For each actor, give their name, average rating of the movies they played in, and the number of movies. Sort the result in the descending order based on average movie rating. In case of ties, sort the rows by actor name.

9. (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 (25 points)

The program should print out the top k actors who played roles in the highest rated movies (on average) in the provided genre.

You should use the sqlite3 database file you created in Part 1.

What to submit:

- uniqname_si618_hw3.db
- uniqname_si618_hw3.ipynb
- uniqname_si618_hw3.html

All your python program files should be able to be run without any error