Problem 1: Programming Tasks (Oracle) [40 points]

The goal of this problem is to write SQL queries and execute them on a real database using Oracle.

Consider the following partial schema for the (real) Internet Movie Database (IMDB.com) where keys are underlined:

- **actors**
  - id: int
  - first_name: string
  - last_name: string
  - gender: string

- **directors**
  - id: int
  - first_name: string
  - last_name: string

- **director_genre**
  - director_id: int
  - genre: string
  - prob: double

- **movies**
  - id: int
  - name: string
  - year: int
  - rank: double

- **movies_directors**
  - director_id: int
  - movie_id: int

- **movies_genres**
  - movie_id: int
  - genre: string

- **roles**
  - actor_id: int
  - movie_id: int
  - role: string

Using the Amazon RDS handout, connect to the Oracle instance we have set up on the Cloud (Amazon’s Relational Data Store) via sqlplus on eniac. You can look at the content of the real tables with queries as follows:

```sql
select table_name from roles where rownum < 10;
```

You can similarly repeat for the tables **movies_directors**, **movies_genres**, **movies**, **director_genre**, **directors**, and **actors**.

Write the following queries in SQL. Please put each answer in a separate script file named **hw1-1a.sql**, **hw1-1b.sql**, etc., and use Canvas to submit. **Your script must compile. Your grades will be based on correctness, implementation and style. You are not allowed to hard code the answers to the query.**

For queries asking for movies, directors, etc., please return all attributes corresponding to the associated source table.

1. (8 points) Find all movies directed by a director whose last name ends with the letter “n”. Hint: the LIKE predicate allows for partial string comparisons, with the percent sign (“%”) used as a wildcard. Google if you need more details.

2. (8 points) Find all genres directed by director whose last name is “Lucas,” with probability greater than 0.5.

3. (8 points) Find all actors who acted in “Comedy” but not in “Drama”.

...
4. (8 points) Find for each actor, the directors who have collaborated on the highest number of movies with him or her. The output should be a list of (actor, director, count) triples.

5. (8 points) List the highest score of each genre type in which each actor acts. The output should be a list of (actor, genre, score) triples.

6. (Extra Credit 8 points) Let $Pr(d, g)$ be the probability that a director $d$ directs genre $g$, given by the relation director_genre. If a director never directs genre $g$, then $Pr(d, g) = 0$. Define the similarity score between two directors $d_1$ and $d_2$ as $\sum_g |Pr(d_1, g) - Pr(d_2, g)|$. Find the 5 director pair with the highest similarity scores, for directors with at least 2 movies.

Problem 2: SQL Queries [50 points]

Consider the following partial schema for a Pinterest-style photo-sharing website, PENNterest.

\[
\begin{align*}
\text{Users} & (userID: \text{int}, \text{name: string, email: string, age: int, gender: character}) \\
\text{Board} & (userID: \text{int}, \text{boardName: string}) \\
\text{PhotoTag} & (photoID: \text{int}, \text{tag: string}) \\
\text{PinInBoard} & (photoID: \text{int}, userID: \text{int}, boardOwnerID: \text{int}, boardName: string) \\
\text{Follower} & (userID: \text{int}, following: \text{int}) \\
\text{Rating} & (photoID: \text{int}, userID: \text{int}, score: float)
\end{align*}
\]

Please note the following:

- Each underlined attribute is (part of) the primary key of its corresponding relation.
- If attribute $A$ is the key of relation $R$, then each appearance of $A$ outside of $R$ is a foreign key referencing $R$.
- In relation Follower, the attributes userID and following refer to users’ IDs.
- Each board has a unique name defined by its creator. Different users may have the same board names.
- Relation Rating consists of scores that a user gives to photos.

Write SQL for the following, and collect into a file for each sub-answer, hw1-2a.sql, hw1-2b.sql, ..., to be submitted on Canvas. Note that this portion of the assignment will NOT work on the Oracle instance for Question 1, as you don’t have permissions to create tables here. Exit sqlplus for now. See www.cis.upenn.edu/~zives/cis550/oracle-faq.html for information on how to relaunch sqlplus using Oracle on eniac, or the read handout on creating your own Amazon RDS instance, .

1. (10 points) Using SQL DDL, create the relational tables corresponding to the schema above.

2. (8 points) Return the full user information of all female users from Penn (email ending in “upenn.edu”), lexicographically sorted by name. You can use an ORDER BY clause (with the names of the sort attributes and an optional DESCENDING specifier) to sort rows in output.

3. (8 points) Find the tags of photos that have not been pinned.

4. (8 points) Find all pairs of users (by name) who have created at least 2 pins.
5. (8 points) Find the photoIDs of the photos pinned by at least 3 users, and also have the tag “cat.”

6. (8 points) Find the photoIDs of all the photos with an average rating of at least 3, unioned with all photos that have no rating.

Problem 3: Short Questions [10 points]

Answer the following questions and type your solutions up in one file, hw1-3.txt. Submit via Canvas.

1. (5 points) How do we capture set-valued attributes in the relational model?

2. (5 points) What is meant by the “open world” assumption?