Carnegie Mellon University
Department of Computer Science
15-415/615 - Database Applications
C. Faloutsos & A. Pavlo, Spring 2014

Homework 2

IMPORTANT - what to hand in:

- Please submit your answers in
  - **hard copy** in class at **1:30pm on Tuesday, February 11th**. For each question, we want both your SQL expression, *as well as* the output of your query on the provided database.
  - And **electronically** on the "blackboard", by the same date/time.

  The electronic submission should be an archive called `[andrew_id].zip`, containing a folder called `queries` and a file for each query, called `q#.sql`, where '#' is the query number. For your convenience, mirror the structure of the file `http://www.cs.cmu.edu/~epapalex/15415S14/db_hw2_s14.zip`, where you can replace the place-holder scripts `/queries/q#.sql` with your real answers.

Reminders:

- **Plagiarism:** Homework may be discussed with other students, but all homework is to be completed **individually**.
- **Typeset** all of your answers whenever possible. Illegible handwriting or ambiguous answers may get no points, at the discretion of the graders.
- For faster grading, please solve each of the 8 questions on a **separate** page, i.e., 8 pages in total for this homework. Type **course-id, hw-id, question-id**, and your **name and Andrew ID on each** of the 8 pages.
- **Late Homeworks**: If you are turning your homework in late, please email it
  - to all TAs
  - with the subject line exactly `15-415 Homework Submission (HW 2)`
  - the count of slip-days you are using, and the count you have left.

For your information:

- Graded out of **100** points;
- **8** questions total

- Rough time estimate: 8 hours (1 hour for each question on average - some questions require less time, some more. Points assigned to each question correlate well with the time needed to solve it.)

Revision : 2014/02/03 12:24

Question	Points	Score
Warm-up queries	5	
Find the star's movies	5	
Popular actors	15	
Most controversial actor	10	
The minions	20	
High productivity	5	
Movies with similar cast	15	
Skyline query	25	
Total:	100	

Introduction

In this homework you will have to write SQL queries to answer questions on a movie dataset, about movies and actors. The database contains two tables:

- `movies`(mid, title, year, num_ratings, rating)
- `play_in`(mid, name, cast_position)

The tables contain the obvious information: which actor played in what movie, at what position; for each movie, we have the title (eg., 'Gone with the wind'), year of production, count of rating reviews it received, and the average score of those ratings (a float in the range 0 to 10, with '10' meaning 'excellent').

We will use Postgres, which is installed in the andrew machines. [1]

You need to do the following set up steps.

1. **Get recommended machine and port number:** On the "blackboard" grades table there are two columns, called `ghc##` and `PGPORT`. The first is the number of the `ghc` machine we recommend for you, and the other is the recommended port number for your Postgres server instance. The goal is to avoid conflicts, by having each of you running on a different machine, and listening on a different port. To log in to the recommended machine, type

   ```
   ssh ghc##.ghc.andrew.cmu.edu
   ```

   replacing `##` by the two digit number you found on "blackboard".

   In the remote case of conflicts, choose a port number close to the one assigned to you and try again. The GHC machines from 02 to 81 are located in GHC 3000, 5201, and 5205 and can be accessed through `ssh`. There are also machines in GHC 5208 which can be accessed physically.

2. **Get and edit setup script**: Download the HW2 .zip from here:

   `http://www.cs.cmu.edu/~epapalex/15415S14/db_hw2_s14.zip`.

   Unzip it and copy the folder `db_hw2_s14` (make sure the name is exactly this one) it on your *home directory* of your andrew account. Edit `setup_db.sh` to modify the line that assigns a value to `PGPORT` to the value that you found on blackboard.

3. **Set up the db:** If your default shell is not bash, type `bash` and then

   ```
   bash setup_db.sh.
   ```

   This will start the Postgres server, create a database with your username, create the schema for this homework, and add data to the database.

4. **Test script:** Run

   ```
   bash run_queries.sh.
   ```

[1] You may develop your queries on your own, local installation of Postgres, but your solutions will be run and graded on the public installation.

This shell script executes all the queries for this homework (which you have to implement). Inside the folder you unzipped, there is a folder called `queries` which contains place-holder .sql files for all the queries that you have to implement. Replace the place-holders with *your* queries, and submit them in a zip file, as we said in the first page.

5. **Optional testing:** If you want to test/debug queries separately, you can also use the script `test.sh`, included in the homework material.

6. **Starting/stopping** the Postgres server: see the instructions at

   `http://www.cs.cmu.edu/~epapalex/15415S14/PostgreSQLReadme.htm`.

The full documentation for Postgres is at http://www.postgresql.org/docs/.

Question 1: Warm-up queries . . . . . [5 points]

Submit on separate page

(a) **[2 points]** Print all actors in the movie `Quantum of Solace`, sorted by cast position. Print only their names.

(b) **[3 points]** Print all movie titles that were released in 2002, with `rating` larger than 8 and with more than one rating (`num_ratings` > 1).

Question 2: Find the star's movies . . . . . [5 points]

Submit on separate page

(a) **[5 points]** Print movie titles where `Sean Connery` was the star (i.e. he had position 1 in the cast). Sort the movie titles alphabetically.

Question 3: Popular actors ..... [15 points]

Submit on separate page

(a) **[8 points]** We want to find the actors of the highest quality. We define their quality as the weighted average of the ratings of the movies they have played in (regardless of cast position), using the number of ratings for each movie as the weight. In other words, we define the quality for a particular actor as

$$\frac{\sum_{all\ movies\ of\ actor}(\texttt{num\_ratings} \times \texttt{rating})}{\sum_{all\ movies\ of\ actor} \texttt{num\_ratings}}$$

Print the names of the top 5 actors, according to the above metric. Break ties alphabetically.

(b) **[7 points]** Now we want to find the 5 most popular actors, in terms of number of ratings (regardless of positive or negative popularity). I.e, if actor 'Smith' played in 2 movies, with `num_ratings` 10 and 15, then Smith's popularity is 25 (=10+15). Print the top 5 actor names according to popularity. Again, break ties alphabetically.

Question 4: Most controversial actor ..... [10 points]

Submit on separate page

(a) **[10 points]** We want to find the most controversial actor. As a measure of controversy, we define the maximum difference between the ratings of two movies that an actor has played in (regardless of cast position). That is, if actor 'Smith' played in a movie that got `rating`=1.2, and another that got `rating`=9.5, and all the other movies he played in, obtained scores within that range, then Smith's contoversy score is 9.5-1.2= 8.3. Print the name of the top-most controversial actor - again, if there is a tie in first place, break it alphabetically.

Question 5: The minions ..... [20 points]

Submit on separate page

(a) **[20 points]** Find the "minions" of `Annette Nicole`: Print the names of actors who only played in movies with her and never without her. The answer *should not* contain the name of `Annette Nicole`. Order the names alphabetically.

Question 6: High productivity . . . . . [5 points]

Submit on separate page

(a) **[5 points]** Find the top 2 most productive years (by number of movies produced). Solve ties by preferring chronologically older years, and print only the years.

Question 7: Movies with similar cast . . . . . [15 points]

Submit on separate page

(a) **[8 points]** Print the count of distinct pairs of movies that have at least one actor in common (ignoring cast position). Exclude self-pairs, and mirror-pairs.

(b) **[7 points]** Print the count of distinct pairs of moves that have at least two actors in common (again, ignoring cast position). Again, exclude self-pairs, and mirror-pairs.

Question 8: Skyline query ..... [25 points]

Submit on separate page

(a) **[25 points]** We want to find a set of movies that have both high popularity (ie, high `num_ratings`) as well as high quality (`rating`). No single movie may achieve both - in which case, we want the so-called *Skyline* query[2]. More specifically, we want all movies that are not "dominated" by any other movie:

Definition of domination : *Movie "A"* dominates *movie "B" if movie "A" wins over movie "B", on both criteria, or wins on one, and ties on the rest.*

Figure 1 gives a pictorial example: the solid dots ('A', 'D', 'F') are not dominated by any other dot, and thus form the skyline. All other dots are dominated by at least one other dot: e.g., dot 'B' is dominated by dot 'A', being inside the shaded rectangle that has 'A' as the upper-right corner.

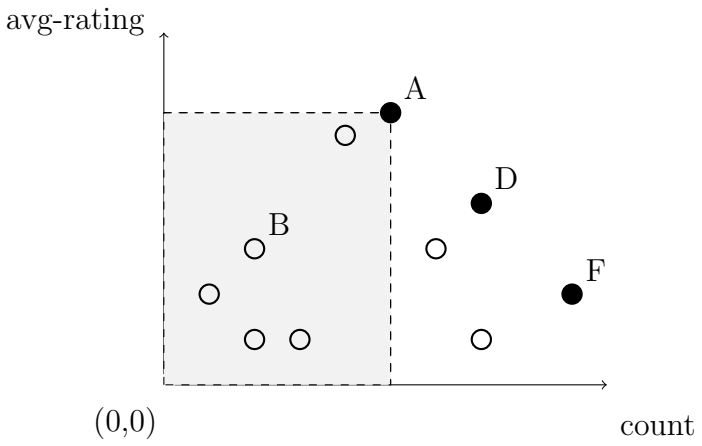

Figure 1: Illustration of Skyline and domination : 'A' dominates all points in the shaded rectangle; 'A', 'D' and 'F' form the skyline of this cloud of points

Given the above description, print the title of all the movies on the skyline, along with the rating and the number of ratings.

[2]FYI, this is also related to the *multi-objective optimization*, since we want movies that optimize two different criteria, at the same time.