Database Systems, CSCI 4380-01
Homework # 5
Due Tuesday March 6, 2018 at 11 PM

Introduction.

This homework is worth 4% of your total grade. If you choose to skip it, Midterm #2 will be worth 4% more. Remember, practice is extremely important to do well in this class. I recommend that not only you solve this homework, but also work on homeworks from past semesters. Link to those is provided in the Piazza resources page.

This homework is on SQL. This is time, there are no rules. Use whatever construct you want. However, check for efficiency. We will test the run time of your queries. Always look for simplest and more efficient query you can write. Do not use DISTINCT unless it is necessary. Do not join with relations that are not needed for the query. A lot of the queries in this homework can be quite hard, so break them down into small pieces and build slowly.

I will also provide the correct answers to each query to test against. If we find an error, the correct answers may be updated. So, please remember to check against the latest copy on Piazza.

Data for this homework (different than previous homework)

The data for this homework is slightly different than homework #4. It contains the data for the latest olympics (with a few bugs here and there). If you find the errors, let me know, I will correct them by next homework. Also, I was unable to find the names of team members for team medals, so they will appear as the country name. I am happy to fix it if you are willing to help me.

You can use this data at the database server. In order not to conflict with Homework#4, I will create this data as a new database called `hw5db` on the class server: `http://rpidbclass.info`

If you want to install this data on your own computer, I recommend you create a new database as well and use the data I have provided:

```
psql> create database hw5db;
psql> \c hw5db;
psql> \i hw5_data.dmp
```

Or on Unix shell:

```
$ psql hw5db -f hw5_data.dmp
```

Remember, if you are using the class database for this homework, the rules are the same and even more important as these are considerably more costly and complex queries.

- You can expect a serious slow down near the homework deadline. Please do not wait till the last minute to submit your homeworks.
- Test your queries one at a time.

- Make every effort to read your queries before submitting. A forgotten join condition may mean disaster. Check join conditions first, then run your queries.
- Remember if you have an unresponsive query, it will continue to use system resources even if you quit your browser. So, opening a new browser window will solve your problem but may slow things down for everyone. Queries should terminate after 2 minutes, so if you increased the load with a bad query, then wait for your query to end before submitting another.
- If the server is not responsive, let us know on Piazza. I will see if some jobs need to be killed or whether server needs to be made more powerful. Please be patient.
- Please do not include a query that does not run in your homework submission. I will run all your queries in a batch job and an incomplete query will cause me a great deal of problems.

Homework Description

Ok, you have been hired by NBC to fill any minute of TV coverage of Olympics with commentary, talking about random Olympic facts. We will use the database to compute different statistics that can be used as talking points.

Write the following queries using SQL (in no particular order of difficulty):

Query 1. For each sport name, return the number of times it appeared in winter olympics and the number of times it appeared in the summer olympics (i.e. number of different olympic games). Order by sport name. Hint: This is a great outer join query, though other solutions are possible too.

Query 2. Find average number of medals for each country in Winter Olympic games over all the games they participated in (i.e. total number of medals for that country in each game averaged over all the games). Return country code and the average medals in descending order of medals.

Query 3. Return the first year an athlete from Japan (JPN) won a Gold medal in Individual (event name) Men's (etype) Figure skating (sport discipline).

Query 4. For all winter olympics, find the athletes, the events and the Olympic games in which the athlete shared a gold in that event with another athlete from a different country. Return id, name, country of the athlete, name of the event and its sport discipline, city and year of the games, order by event name, olympic year, city, athlete country and name.

Query 5. We are actually counting medals incorrectly, because team and pair events count as 1 gold only. A table listing the event categories for 2018 events is given in table `winter_eventcategories`. In this case, if etype is `single`, then we count individuals, otherwise we should count individual events.

Return a correct medal count for 2018 olympics for each country. Return country name and total number of medals, ordered by medals first and then by country.

Note there is an error in our data for Germany, but the correct query should produce results that match our results. We will use this method for counting medals only in this query.

Query 6. For each country, return the number of Gold medals by athletes who won a gold medal in 2018 Olympics and never before (use the quick and dirty way of counting medals here, just counting the number of tuples from `winter_medals` including the team and pair events). Order by the count and country. Hint: This is a great query to practice `NOT EXISTS`.

Query 7. Find countries that won a medal in every Winter olympic game since 1900 (including 1900). Order by country.

Query 8. For all winter sports, return the name of sport and the year when USA won a Gold medal in that sport for the first time. Order by sport name. Note that due to the way this database is constructed, the sport with the same name may actually appear more than once with different ID. So, when finding the same sport, use the sport name.

Submission Instructions.

Submit your queries in a PDF on gradescope. You can use the template I posted for Hw#4.

Also, you must submit the text content of your queries formatted in the same way as the last homework on the class server:

Connect to database `submissions`, click on the `SQL command` link on the left.

Submit your queries as text:

```
select hw5($$ ---- $$);
```

by replacing `----` with the full ASCII content of your submission.

Database Schema

```
create table countries(
     name         varchar(255) not null
     , code       char(3) primary key
     , population int
     , gdp        float --gross domestic product
) ;
create table olympics (--all olympic games, winter or summer
      id      int primary key
      , year  int not null
      , city  varchar(255) not null
      , otype char(6) not null --winter or summer
      , check (otype in ('winter','summer'))
);
create table sports (--all sports in winter and summer games
      id          int primary key
      , name      varchar(255) not null
      , discipline varchar(255) not null
      , stype     char(6)     not null  --- winter or summer
) ;
create table events (--all events in winter and summer games
      -- team vs individual events not distinguishable
      id         int primary key
      , sid      int not null -- which sport this event is for
      , name     varchar(255) not null
      , etype    char(1) -- m or f
      , constraint event_fk foreign key (sid)
            references sports(id)
) ;
create table athletes (--all athletes who won a medal in an olympic game
      id        int primary key
      , name    varchar(255) not null
      , country char(3) -- same as countries(code)
      -- but not all countries are present in countries(code)
) ;
create table summer_medals (--medals given in summer olympics
      id      int primary key
      , oid   int not null -- olympic game id
      , aid   int not null -- athlete id
      , eid   int not null -- event id
      , medal varchar(6) not null -- Gold Silver or Bronze
      , constraint sm_fk1 foreign key (oid) references olympics(id)
      , constraint sm_fk2 foreign key (aid) references athletes(id)
      , constraint sm_fk3 foreign key (eid) references events(id)
) ;
create table winter_medals (--medals given in summer olympics
      id      int primary key
      , oid   int not null  -- olympic game id
      , aid   int not null  -- athlete id
      , eid   int not null  -- event id
      , medal varchar(6) not null
      , constraint sm_fk1 foreign key (oid) references olympics(id)
      , constraint sm_fk2 foreign key (aid) references athletes(id)
      , constraint sm_fk3 foreign key (eid) references events(id)
) ;

create table winter_eventcategories(
      eid int primary key
      , etype   char(6)
      , check (etype in ('single','multi','team'))
      , foreign key (eid) references events(id)
) ;
```