Assignment 2
COL362/632

Due date: January 31, 2017, 11:55pm IST

General Instructions

1. Do this assignment individually.
2. Use PostgreSQL 9.5 for your assignments. See `https://www.postgresql.org/download/` for instructions on how to download and install it on your OS. The documentation can be found at `https://www.postgresql.org/docs/9.5/static/index.html`.
3. There are two datasets with their corresponding queries in the assignment. You are required to submit the following files.
   (a) <EntryNumber>-I.sql and <EntryNumber>-II.sql: These files will contain the solutions for the queries in the format mentioned below.
   (b) <EntryNumber>QueryPlan-I and <EntryNumber>QueryPlan-II: These are text files containing the query plans for each of the queries in your solution files. For each query `q` in your sql file, use `EXPLAIN q` to print the query plan in a text format without actual run time information, and concatenate the outputs into a single text file.
   (c) <EntryNumber>RunTime-I and <EntryNumber>RunTime-II: This is a text file that contains a table with 3 columns

   < query no >   < estimated cost >   < actual time >

   This file should contain one line per query and values in a line should be separated by white space. For a query `q`, use `EXPLAIN ANALYZE q` to obtain the estimated cost and actual run time values.
4. The .sql files are run automatically, so please ensure that there are no syntax errors in the file. *If we are unable to run your file, you get an automatic reduction to 0 marks.*
5. The format of the .sql files should be as follows. The solution to each query should be preceeded by the line `--<queryNumber>--`. You may create views for writing queries. All views must be created in a `--PREAMBLE--` section, and then removed in a `--CLEANUP--` section, while the actual query should be preceded by the `--QUERY--` line, as shown below. Note that even if you do not use views, your query should be preceded by the `--QUERY--` line. Leave a blank line after each query. Also, note that the actual query should be *single* SQL statement, whose result will be evaluated.

```
--<queryNumber>--
--PREAMBLE--
Create Views here
--QUERY--
Write the Query here - single SQL statement
--CLEANUP--
Drop any created views here
```

   OR

```
--<queryNumber>--
--QUERY--
Write the Query here - single SQL statement
```

6. Many of the queries below require an ‘ORDER BY’ clause. The default behavior of this clause is to return results in *increasing* order. In this assignment, the phrase ‘order by’ refers to increasing order, unless mentioned otherwise. If you omit the ‘ORDER BY’ clause in a query that requires ordering, your answer will be evaluated as incorrect and zero marks will be awarded.

7. Errors with respect to equality and inequality conditions will also be evaluated as incorrect and zero marks will be awarded.

8. For every question, the number (and meaning) of columns expected in the output are given.

   (a) The sequence of the columns as given here must be maintained in your results.

   (b) You are *NOT* required to follow the column names given in the questions.

9. **No changes are allowed in the i) data, ii) attribute names, iii) table names**

   Note that if you use phppgadmin to interface with Postgres, attribute or table names in your database may be enclosed in quotes. However, while submitting, ensure that your .sql files are quote-free.

1 Dataset1 - ECO

1. The dataset is a subset of the ECO Energy Consumption Dataset, which provides energy consumption data collected from 6 Swiss households over a period of one month - January 2013. Each house is fitted with a Smart meter which measures the overall power consumption of the household. In addition, some selected appliances inside the house are fitted with plugs that measure the consumption of that appliance. CSV files for this dataset can be found at the following:

   1. `http://www.cse.iitd.ac.in/~neha/datasets/ECO/HOUSE.csv`
      This file lists the information of all houses.
   2. `http://www.cse.iitd.ac.in/~neha/datasets/ECO/APPLIANCE.csv`
      This file lists the information of all the appliances in all houses.
   3. `http://www.cse.iitd.ac.in/~neha/datasets/ECO/SMARTMETERDATAsmall.csv`
      This file contains the readings of the smart meters of all houses.
   4. `http://www.cse.iitd.ac.in/~neha/datasets/ECO/PLUGDATAsmall.csv`
      Lists the readings of the appliance level measurements of all appliances in all houses.

2. Your database should include following four tables and you should use only these tables while writing solution of the queries. The Primary Key(s) in each table is underlined.

   1. house

Column Name	Data Type	Description
<u>houseid</u>	int	Unique identifier for the house, primary key
housename	varchar	Name of the house

   2. appliance

Column Name	Data Type	Description
<u>applianceid</u>	int	Unique identifier for appliance, primary key
houseid	int	Foreign key, references house(houseid)
appname	varchar	Name of the appliance
devicetype	varchar	For example, Kitchen or Electronics

   3. smartmeterdata

Column Name	Data Type	Description
houseid	int	Foreign key, references house(houseid)
readingtime	timestamp	Format is YYYY-MM-DD HH:MI:SS.
powerallphases	real	Total power on all phases

4. plugdata

Column Name	Data Type	Description
applianceid	int	Foreign key, references appliance(applianceid)
readingtime	timestamp	Format same as in smartmeterdata
power	real	Power consumption of appliance

1.1 Queries

Note:

- The dataset contains entries with a power reading of -1. These correspond to timestamps when the sensor was unable to record a reading. Your queries MUST suitably ignore these readings when computing the total, average, minimum, or maximum consumption.
- The sensors record readings at 1 minute intervals. The average power consumption of a device from timestamps t_1 to t_2 is the average of all (valid) power readings within the interval $[t_1, t_2]$. The total power consumption, similarly, is the sum of all (valid) power readings within the interval $[t_1, t_2]$. Observe that the closed interval $[t_1, t_2]$ *includes* the timestamps t_1 and t_2.
- Wherever asked for the aggregate power consumption of a *house*, use the smartmeterdata relation.
- Follow the order by requirement in each query.
- Report all power consumption results rounded to 2 digits of precision.

1. Count the number of device types per house. If a house has no device types, 0 should be reported for it. Output (housename, number of device types) order by housename.
2. List all appliances in each household. Output (housename, appname, devicetype) ordered first by house name, then by appliance name.
3. Print a list of all appliances in House2 along with their average power consumption. Output (appname, avgpower) ordered by appliance name.
4. For each appliance in House 4, list the maximum, minimum, and average power consumption. Output (appname, maxpower, minpower, avgpower) ordered by appliance name.
5. For each device type, get the house that has the maximum total power consumption on any appliance in that type. Output (devicetype, housename) ordered by device type.

   In case of a tie between two different houses, report the one with the smaller housename.
6. Get the average power consumption across all appliances and all homes for each device type. Output (devicetype, avgpower) ordered by device type.
7. For each house, print the average appliance power consumption (across all appliances) and the average aggregate power consumption. Output (housename, avgappliancepower, avgoverallpower) ordered by house name.
8. Get the minimum, maximum, and average aggregate power consumption of House1 for each hour in the day. Output (hour[01-23], minpower, maxpower, avgpower) ordered by hour.

9. For each house, get the name of the appliance that has been used for the longest duration within a given day. Output (housename, appname) ordered by house name.

   Note that an appliance can be interpreted as OFF *only* when it has a power reading of 0. A duration of use is a period between two different, consecutive OFF states observed within the same day.

   In case of a tie between two different appliances, report the one with the smaller appname.

10. Count the number of days when House6 had the least aggregate average power consumption of all houses. Output (numdays).

11. For each pair of houses (a,b), count the number of hours when the average power consumption of house a is greater than that of house b. Since there are 6 houses in this dataset, the result of this query must have 30 rows. Output (housea, houseb, numhours) ordered by housea first, then houseb. Here, housea and houseb are houseIds.

12. Get the 'coverage' of each appliance in each house, rounded to 2 places of decimal. The coverage of an appliance is defined as the ratio of the number of valid (≥ 0) power readings of the appliance to the total number of power readings for the appliance. Output (appname, housename, coverage) ordered by housename first, then appname.

13. Get the weekly average of total aggregate power consumption of House3. Output (avgpower).

2 Dataset2 - Basketball

1. The Mens Basketball dataset contains individual and team statistics from professional basketball leagues including the National Basketball Association through the 2011-12 NBA season. CSV files for this dataset can be found at the following:

   1. `http://www.cse.iitd.ac.in/~neha/datasets/Basketball/basketball_abbrev.csv`
      This file lists useful abbreviations in the database and the full name.
   2. `http://www.cse.iitd.ac.in/~neha/datasets/Basketball/basketball_coaches.csv`
      Contains information related to the coaches, the teams they have coached, the year in which they were associated with different teams etc.
   3. `http://www.cse.iitd.ac.in/~neha/datasets/Basketball/basketball_master.csv`
      This file contains general information of all the people in the database(coaches and players) like place of birth, height, college etc.
   4. `http://www.cse.iitd.ac.in/~neha/datasets/Basketball/basketball_player.csv`
      This file contains information about all the players. The information includes the teams they have been associated with, games played, points earned etc.
   5. `http://www.cse.iitd.ac.in/~neha/datasets/Basketball/basketball_series.csv`
      This file contains information about all the matches ever played, the teams that won/lost and the points scored by both the teams.
   6. `http://www.cse.iitd.ac.in/~neha/datasets/Basketball/basketball_team.csv`
      This file has details of all the teams, the franchise they belong to, division, rank, wins and loses etc.

2. Your database should include following five tables and you should use only these tables while writing solution of the queries. The Primary Key(s) in each table is underlined.

   1. basketball_abbrev

Column Name	Data Type	Description
<u>id</u>	int	Unique identifier, primary key
abbrev_type	varchar	Type of abbreviation
code	varchar	The acronym
full_name	varchar	Full form

   2. basketball_coaches

Column Name	Data Type	Description
<u>id</u>	int	Unique identifier, primary key
coachid	varchar	Id given to the coach. There can be multiple rows with the same coach id
year	int	Year in which the coach was associated with a team
tmid	varchar	team ID of the team coached
lgid	varchar	Id of the league
won	int	Number of matches won
lost	int	Number of matches lost

3. basketball_master

Column Name	Data Type	Description
<u>id</u>	int	Primary key
bioid	varchar	Unique id of user (player or coach)
firstname	varchar	First name of player/ coach
middlename	varchar	Middle name of player/ coach
lastname	varchar	Lastname of player/ coach
height	double precision	height of user
weight	double precision	weight of user
college	varchar	university/ college of user
birthdate	date	the birthdate of the user
birthcity	varchar	Native place of user
deathdate	date	deathdate of user
race	varchar	Race of user (W - White, B - Black)

4. basketball_player

Column Name	Data Type	Description
<u>id</u>	int	Primary key
playerid	varchar	Unique id of player
year	int	Year in which he played
tmid	varchar	Team id
lgid	varchar	League id
gp	int	Games Played
points	int	Total points scored by the player
fgmade	int	Field goals made
ftmade	int	Free throws made

5. basketball_series

Column Name	Data Type	Description
<u>id</u>	int	Primary key
year	int	Year the match was played
round	varchar	Round information for the match (QF: QuarterFinal, F: Final etc)
series	varchar	Id of the series
tmidwinner	varchar	Team ID of the winner
tmidloser	varchar	Team ID of the loser
w	int	points of the winning team
l	int	points of the losing team

6. basketball_team

Column Name	Data Type	Description
id	int	Primary key
year	int	The year for which the team statistics are given
lgid	varcar	League id
tmid	varchar	Team id
franchid	varchar	Id of the franchise that owns the team
divid	varchar	Id of the division
rank	int	Rank of the team that year
name	varchar	Name of the team
won	int	Number of matches won
lost	int	Number of matches lost
games	int	Number of games played
arena	varchar	team arena name
attendance	int	arena capacity

2.1 Queries

1. Find the coach(es) with the maximum win percentage. If more than one found, order by the coachID. Output(CoachID).
2. Report the Success rates of the all coaches, ordered by decreasing order of Success rate. The Success rate of a coach is defined as the ratio of number of matches won to number of total matches played. Output(CoachID,Success Rate)
   Note: NULL values should be considered 0.
3. Report the teams which have been ranked '1' each year. The output should be ordered by year. Output(year).
4. Find the maximum number of matches that were played in any year. Output(year, number of matches).
5. For each team, display the player(s) with the maximum points to GP(games played) ratio. The output should be ordered by increasing order of teamId. Output(teamID,playerID)
6. Report the team(s) that have qualified for the semi-finals and beyond most number of times. The output should be ordered by teamID. Output(teamID, count).
7. Report the player(s) that were a part of the teams that played the championship finals in the years '1946','1947', and '1948' .The output should be order by the teamID and playerID. Output(teamID, playerID).
8. Report the player(s) from the college 'Duke' who have played for the team with the teamID 'NYK' .The output should be order by the playerID. Output(playerID).
9. Report all the players who have been coached by the coach with coachID 'olseneha'. The result should be ordered according to the playerID. Output(playerID).
10. Report the player(s) who have scored between 200 and 500(both included) field goals (column 'fgmade') in sorted order of playerIDs. Output(playerID).
11. For all the teams that players with first name "mark" (ignore case) have been a part of, find the first names of the coaches that those teams have had. The result should be sorted in increasing order of the name. Output(team id, first name of the coach)
12. List the first names of all the players born in the same year as any player who was born in the city of "Detroit" and is white. The output should be sorted by name in decreasing order. Output(firstname).
13. For each team belonging to the west division, find the first names of the players who have played and scored more than 700 points (700 included), in all, for the team. Order by increasing order of name. Output(firstname,playerid, total points).
    Note: If the player has been on and off the team, consider the sum total of the points.

14. For all the losing teams who have suffered a loss margin (W-L in series table) greater than or equal to 2 (we call them 'Y'), find the one which has the highest win to loss ratio (we call it 'X'). Then count the number of teams which have a higher win to loss ratio compared to 'X' for each division in which the teams from 'Y' belong. Output (Division, count) sorted by the division.