Lab 4 Part II: The Return of Simple Queries

Due date: Tuesday, October 23, **midnight**!.

Note: This is the due date for the entire Lab 4 assignment.

Lab Assignment

The Task

In addition to the queries you wrote for part 1 of the Lab assignment, write, debug and test the SQL statements for the following information needs.

The SQL statements can be simply included at the end of the SQL script files you prepared for Part 1 of the lab.

STUDENT dataset

1. Find teachers of all students whose first name is `'THEO'`. Output first and last names of the teachers. Ensure that each teacher's name does not get reported more than once.
2. For each classroom, report the grade it is used for. Sort the output by grade level, then by classroom number.

BAKERY dataset

1. Find all purchases made on October 21. For each purchase output the name of the cusomter and the reciept number.
2. Find all items sold on October 3. For each item list the reciept number, the name of the item (flavor, food), and the price. Do not eliminate duplicate tuples. Output the result sorted by flavor, food.

3. Find all people who purchased `Apple Pie` between October 10 and October 20 (inclusive). Report the date of purchase, the name of the customer (first, last). Output the result sorted by date of purchase, then by last name of the customer.

4. Find all purchases (reciepts) which had at least five items in them. Report the Reciept number, date and the name (first, last) of the customer. Sort the output by date, then by last name of the customer[1]

5. Find all purchases of items that cost more than $8. For each item, output the date, the item name (flavor, food) and price. Sort the output by date, then by item name (flavor, then food).

6. Find all purchases of items that cost more than $8. For each item, output the date, the item name (flavor, food) and price and the name of the customer who bought it. Sort the output by customer name and item price.

CARS dataset

1. Find the engine displacements for all cars produced by `Nissan Motors`. Output the full model name, year, and engine displacement, sorted by engine displacement.

2. Find all 4-cylinder models made in 1970 that accelerate better than *some* 8-cylinder car made in 1971. Output the full name of the 4-cylinder model, its acceleration time, and the acceleration time of the 8-cylinder model that it bettered.

3. Find any Volvo cars that have better (i.e., higher) gas milage than a 1978 `amc concord`. Output the full name of the car, the year it was built and the gas milage.

4. Find all Japanese cars built between 1973 and 1976 which have more horsepower than one General Motors car built in the same year. Output the full name of the Japanese model, the year it was built and the horsepower. Sort the output by year, then by horsepower.

CSU dataset

Here are the queries for the CSU dataset.

1. For each campus report the number of degrees granted, the number of enrolled students, and the number of full-time faculty lines for the years 2002 and 2003. Ouput full name of the campus, the year, and the requested information.

[1] **do not** use aggregation operations for this query.

2. For all campuses in `Los Angeles` county report all disciplines where total enrollment (undergraduate+graduate) in 2004 has exceeded 1000 students. Output the full name of the campus, the year it was founded, and the full name of the discipline. Exclude `'Undeclared'` majors from the output.

3. For each campus report the percentage of students studying `Engineering` in 2004. (Use the `TotalEnrollment_AY` column for the total enrollment numbers and the undergraduate enrollment for the discipline). Output the full name of the campus, and the computer percentage presented in the column named `"Eng_Percentage"`. Sort output by percentage.

4. For each campus report the percentage of students studying `Engineering` or `Computer and Info. Sciences` in 2004. (Use the `TotalEnrollment_AY` column for the total enrollment numbers). Output the full name of the campus, and the computer percentage presented in the column named `"Eng_CS_Percentage"`. Sort output by percentage.

   (Note: it is ok, if the result of this query contains fewer tuples than the result of the previous query.)

5. Find all campuses where in 2004 enrollment in `Business and Management` discipline has exceeded enrollment in `Engineering` discipline. Output the full name of the campus and the difference in enrollments. Sort the result by the difference.

6. For each campus report the total number of campus student fees collected for year 2000. Report the full name of campus and the total amount. Sort the result by the amount.

MARATHON dataset

For this dataset, all times must be output in the same format as in the original dataset (in the file `marathon.csv`).

1. Find the results of all athletes from Rhode Island (`RI`). For each athlete output their full name (first, last), the town they are from, their time and overall place. Sort by the overall place.

2. List all female runners whose result was better than the result of the 30th man in the `40-49` age category. Report the full name of each runner, her time and pace, overall place, age group and the place in the age group. Sort by age group, the by place within the age group.

3. For each gender-age group[2] report the names of the three top finishers. The format for the output should be the age group, the gender, the name of the first place runner, the name of the second-place runner,

[2]...with at least three runners in the race

the name of the third-place runner. All names must be full (first, last).Sort the output by age group, then by gender.

4. List all locations, whose representatives placed in top 40 in **both** men's and women's competititions (by age-gender group). Output the name of the town and the state.

5. List all locations which sent participants to the competition in the 20-39 age group, but did not send a participant in the 40-49 age group.

Submission Instructions

Note: this is a repetition of the instructions from Part 1. Only one submission for both parts is needed!

You must submit all your files in a single archive. Accepted formats are `gzipped tar` (`.tar.gz`) or `zip` (`.zip`). The file you are submitting must be named `lab4-ilastname.ext`, where *i* stands for the initial of your first name, and *lastname* is your last name. E.g., if I were submitting this file, the name would be `lab4-adekhtyar.zip` or `lab4-adekhtyar.tar.gz`.

The archive shall contain five directories: `CARS`, `CSU`, `BAKERY`, `STUDENTS` and `MARATHON`.

Each directory shall contain the following SQL scripts:

- Database creation script. (e.g., `CARS-setup.sql`). Use the scripts from Lab 2 and (for MARATHON) Lab 3 submissions.
- Table creation scripts (e.g.,`CARS-countries.sql`). Use the scripts from Lab 2/Lab 3. You are welcome to `cat` all scripts together into one big script. If you do, name it `<DATASET>-insert.sql` (e.g., `CARS-insert.sql`).
- The cleanup script (e.g., `CARS-cleanup.sql`). Use the scripts from Lab 2 and Lab 3.
- **NEW script.** One script per database, containing all SQL statements and any `SQL*plus` statements needed for formatting. Name the script `<DATASET>-info.sql` (e.g., `CARS-info.sql`.