Lab 4: Simple Queries

Due date: Tuesday, October 16, **midnight**!.

Lab Assignment

Assignment Preparation

This is an individual lab. Each student has to complete all work required in the lab, and submit all required materials **exactly as specified** in this assignment.

The assignment will involve writing SQL queries for different information needs (quesions asked in English) for each of the five course datasets.

The Task

You are to write and debug (to ensure correct output) the SQL queries that return information as requested in each of the information needs outlined below. The information needs can be addressed with a simple SELECT statement (i.e., a SELECT statement without grouping, aggregation and nested subqueries) and/or with UNION, MINUS and INTERSECT statements.

For this assignment, you will prepare one SQL script for each database. In addition to SQL statements you may need to include some SQL*plus formatting instructions to ensure that your output looks good. In particular, every row of every resulting table must be printed in a single line. If that means changing the size of the line - do it. Similarly, there should not be awkward pagination of the answers - change page size as needed.

STUDENT dataset

For STUDENT dataset, write an SQL script containing SQL statements answering the following information requests.

1. Find all students who attend 3d grade. For each student list first and last name and the classroom number.

2. For each student in 4th grade, report their teacher. Output first and last name of the student and first and last name of the teacher.

3. For each students whose first name is `'JODY'` report their first and last name, grade, classroom and the last name of the teacher.

4. Find all students taught by `'ALFREDA SUGAI'`. Output first and last names of students.

5. Find all students who study in classrooms with numbers between 103 and 108 (inclusive). Output the first and last name of the student, their grade and classroom. Output the list ordered by classroom and within each classroom – by student's last name.

BAKERY dataset

Write an SQL script containing SQL statements answering the following information requests.

1. Find all chocolate-flavored goods on the menu. For each item output the flavor, the name (food type) of the item, and the price.

2. Find all customers who purchased `'Chocolate Tart'`. Output only the first and the last names of the customers. Each customer name must be reported once.

3. Find all customers who purchases any danishes. For each customer output the name and the date of the purchase.

4. Find all purchases (i.e., reciepts) which included the same item purchased more than once (e.g., a reciept that contains more than one `'Lemon Cake'`). For each purchase, output the date, the item purchased (flavor, food type), its price and the name (first, last) of the customer. Output only one row per purchase per item (e.g., if a customer purchases three `'Apricot Tart'`s, you should only report this fact once).

5. Find all items ever purchased by `'SHARRON TOUSSAND'`. Report the name of the item (flavor, food type) and the price. Each type of purchases items must be reported only once.

6. Find all customers who purchased (at some point in time, possibly during different trips to the bakery) both `'Napoleon Cake'` and a `'Casino Cake'`. Output first and last names of each customer. (each name must be reported once).

7. Find all customers who purchased at least once either `'Casino Cake'` or a `'Napoleon Cake'` but never purchased a `'Lemon Cookie'`. Output names (first, last) of customers. (each name must be reported once).

8. Find all dates on which the bakery sold at least one `'Opera Cake'` but sold not a single `'Tuile Cookie'`. Output the dates, ordered from earliest to latest.

CARS dataset

1. Find all 8-cylinder cars. For each car report its full name (e.g., `'plymouth fury iii'`), year it was produced and milage per gallon. The list should be displayed in ascending order by milage per gallon.

2. Find all cars that accelerate from 0 to 60mph in less than 12.5 seconds. For each car report its full name and year, county of the maker, acceleration time and horspower.

3. Find all cars built in or before 1975 with better gas milage than the 1976 'ford pinto'. Output the full name and year, country of origin and gas milage.

4. For each 1980 model, output its full name, full name of the manufacturer (e.g., `'Ford Motor Company'`, manufacturer's home country and continent.

5. Find all car makers which produced at least one light (weight less than 2000lbs) car between 1974 and 1979 (inclusively). Output the full name of the company and its home country. Each company should be reported at most once.

6. Find all models produced by `'gm'`. Output the model name (e.g., `'pontiac'`, full car (make) name (e.g., `'pontiac firebird'`) and year of production. Output the list sorted by the name of the model.

CSU dataset

Here are the queries for the CSU dataset.

1. For each year between 1992 and 2002 (inclusive) report the number of graduates from Cal Poly (San Luis Obispo). Output the year and the number of graduates.

2. Find all years in which the number of graduates from CSU San Marcos exceeded the number of graduates from CSU Stanislaus. Out put the years, and the respective numbers of degrees granted for each of the two campuses. (here we assume "graduates" = "degrees granted").

3. Report enrollments in `'Computer and Info. Sciences'` disciplines for each campus in 2004. Output the full name of the campus and the enrollment. Sort the output in ascending order by the enrollment.

4. Find campuses where the faculty FTE has decreased in two consecutive years. Report the full campus name, and the two consecutive years and the FTEs for each of them. (Note, if such decrease in FTEs happened on a single campus more than once, report all such occurrences). Sort the output in ascending order by the first of the pair of years.

5. List all campuses where campus fees in 2006 were greater than those in San Diego State University. For each campus, list its full name and the fee amount. Sort the output by the fee amount.

6. For each campus, report the student to faculty ratio in 2004. (Use the student enrollment FTE and the faculty FTE for the year.) Output the full name of the campus, student enrollment, faculty size and the student-to-faculty ratio. Output in ascending order by the computed ratio.

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 male runners from the `'20-39'` age group representing the state of Massachussets. Output first and last name, town/city name, their place in the group and their time.

2. Find all female runners whose pace at the marathon was better than 7:30. Output their names, age category and the overall place each runner took in the race.

3. Find the results for all 44-year old runners. For each runner, output name (first, last), town, state, time running and gender. Sort by gender, then by time.

4. Find all women who ran the race faster than the race participant with Bib number 284.

5. List all runners who took first place in their respective age/gender groups. For each runner, output name (first, last), age, gender,

Submission Instructions

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`.