Lab 4: Simple Queries

Due date: Thursday, October 25, **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 for 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. However, each information need **must be met** with a **single** SQL statement.

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.

NOTE: Please provide a comment in front of each SQL statement in each of your files. The simplest comment can just state the query number (e.g.,

"--- Q3.") for this particular database. This is very useful for the situations when for one reason or another you elected not to implement a query.

STUDENTS dataset

For the STUDENTS dataset, write an SQL script `STUDENTS-info.sql` containing SQL statements answering the following information requests.

1. Find all students who study in classroom 104. For each student list first and last name.
2. For each student in the sixth grade, report the name of the student (first, last) and the name of their teacher (first, last). Sort the output in alphabetical order by the last name of the teacher, then in alphabetical order by the last name of the student.
3. Find all students whose first or last name starts with a `T`. Return the first name, the last name and the grade for each student discovered. Sort output in ascending order by grade.
4. Find all classrooms in which fourth grade is taught. Report each classroom exactly once. Report classrooms in descending order.
5. Find all students taught by `JEROME COVIN`. Output first and last names of students sorted in alphabetical order by their last name.
6. For each classroom, report the grade taught in it. Order the output by grade in descending order. Each classroom–grade pair should be reported exactly once.
7. For each teacher, report the grade (s)he teaches. Each name has to be reported exactly once. Sort the output alphabetically by teacher's last name.
8. Find all fourth grade students who are NOT taught by GEORGETTA SUMPTON. Report their first and last names in alphabetical order by last name.

BAKERY dataset

Write an SQL script `BAKERY-info.sql` containing SQL statements answering the following information requests.

Note: Your queries must match exactly the wording of the information need. For example, if you are asked to find the price of an `Apricot Tart`, the following query

```
SELECT price
FROM goods
WHERE CODE = '90-APR-PF';
```

is considered to be incorrect because nowhere in the query was the code `'90-APR-PF'` mentioned. (This is especially important when you are expected to produce a join of two or more tables, but instead look up the foreign key value and use it verbatim in the query. Such queries will be marked as incorrect on the spot).

1. Find all tarts on the menu. For each item output the flavor, the name (food type) of the item, and the price. Sort your output in descending order by price.

2. Find all pastries that are not eclairs, not cakes whose price is between \$12.00 and \$20.00 (inclusive), not vanilla-flavored and cost more than \$3.00. Output the pastry code, its flavor, the name of the pastry and the price. Sort your output in ascending order by price.

   (note: anything vanilla is excluded; any eclairs are out and any cakes that fall in the specified price range are out).

3. Find all customers who purchased at least one `Opera Cake`. Output only the first and the last names of the customers. Sort by the last name of the customer.

4. Report all purchases of an `Opera Cake` that took place in the first two weeks of October, 2007[1] For each purchase, report its date and the name of the customer. Output purchases sorted by date. Sort purchases on the same date by the last name of the customer.

5. Find all different pastries purchased on October 31, 2007. Each pastry type (flavor, food) is to be listed once. Order output by the type of pastry (`food`, then `flavor`).

6. Find all purchases in which the customer bought multiples of a single type of a cookie (e.g., two `Lemon Cookies`). Report receipt number, date of purchase, purchased cookie (flavor, food) and the name of the customer. Sort the output by purchase date, and by the receipt number for purchases made on the same day. Each item must be reported once.

7. Find all dates in the second half of October of 2007 (October 16 to October 31 inclusive) on which one customer made multiple purchases. Report each date exactly once, output dates sorted in ascending order.

8. Find all customers who purchased, during the same trip to the bakery, two different cakes. Report first and last names of the customers.

9. Find all customers who did not make a purchase on October 09, 2007. Report their first and last names.

10. Find all items purchased by `JULIET LOGAN` on October 31, 2007. Report the name of the pastry (flavor, food) and its price. Report the

[1] Recall, the dataset contains data for the month of October, 2007.

purchases sorted by receipt number. For each receipt, sort the purchases in the order in which they appear on the receipt.

CARS dataset

1. Find all Japanese car makers in the database. Report the full names of the car makers sorted in alphabetical order.
2. Find all models[2] produced by GM (`'gm'`) stored in the database. Report just the names of the models in alphabetical order.
3. Find all Cadillacs in the database. For each, report the make and the year. Sort output by year.
4. Find all cars produced by `Renault` between 1971 and 1977 (inclusive). Report the make of the car and the year it was produced, sort output in ascending order by the year.
5. Report all car makers which produced in 1971 a car with more than 115 horsepowers and over 300 engine displacement. Report the full and the abbreviated names of each car maker in alphabetical order (by full name) and report each car maker once.
6. Find all light (lighter than 3000 lb) 6-cylinder cars produced in the 1980s, and report their make, weight and acceleration. Output in descending order by weight.
7. Find all cars that have that are more powerful (more horsepowers) than the 1970 `dodge d200`. Report the makes, the years, the horsepower of the car and the short name and the country of the maker. Output in ascending order by horsepower.
8. Find all non-European car makers which produced at least one light (weight less than 2000lbs) car between 1979 and 1981 (inclusively). Output the full name of the company and its home country. Each company should be reported just once.

CSU dataset

Here are the queries for the CSU dataset. Name the SQL scrips `CSU-info.sql`

1. Report all counties in which CSU campuses are located. Report each country once and sort the in alphabetical order by county name.
2. Report all counties in which more than one CSU campus is located. Report the name of the county (just once). Sort output in alphabetical order by the county name.

[2] Just models, not makes.

3. For each year between 1997 and 2002 (inclusive) report the number of students who graduated from `California State University-East Bay` Output the year and the number of degrees granted. Sort output by year.

4. Report all years when the number of degrees granted by `San Francisco State University` was less than the number of degrees granted by `San Jose State University`. Output years (in chronological order) and the number of degrees granted for each campus.

5. Report total (both undergraduate and graduate) enrollments in `'Mathematics'` disciplines for each San Diego, Los Angeles, San Bernardino, and Kern county campus in 2004. Output the full name of the campus, the county it resides in and the enrollment. Sort the output in descending order by the enrollment.

6. For each university with over 20,000 students enrolled in 2004, report the difference and the ratio between the student fees in 1996 and 2004. Sort the output in descending order by the ratio.

7. Report all disciplines in which graduate enrollment in 2004 in California State Polytechnic University-Pomona was less than the enrollment in California Polytechnic State University-San Luis Obispo. Report the discipline names and the enrollments in both universities. Output the discipline names in alphabetical order.

8. Find all campuses in which the number of faculty full-time equivalent positions (FTEs) increased on two consecutive years. Report campus, the three years involved (year1→increase→ year2 → increase → year3) and the FTE numbers for each year.

9. For each campus with enrollment (use student enrollment FTE) under 12,000, 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.

10. Find all campuses which in 2004 had more `Engineering` than `Education` students in the graduate program. Report campus names and the number of graduate students in each of the two disciplines.

INN dataset

For the INN dataset, create a SQL script file `INN-info.sql` with SQL queries for the following information needs. (When no year is supplied in the query descriptions below, assume 2010).

1. Find all rooms whose decor is something other than `'traditional'`. For each room report its code, the full name of the room, max occupancy and the base rate. Sort the output in ascending order by the base rate.

2. Find all August reservations (a.k.a., all reservations that both start AND end in August) for the 'Frugal not apropos' room. For each reservation report the last name of the person who reserved it, checkin and checkout dates, the total number of people staying and the daily rate. Output reservations in chronological order.

3. Find all rooms occupied on January 21, 2010. Report full name of the room, number of beds, the check-in and checkout dates of the reservation and total number of people staying. Sort output in alphabetical order by room name.

4. Find all rooms that are not occupied on September 19 and 20, 2010. Report room code, full name of the room and the base rate. Sort output in alpabetical order by room name.

5. Find all customers who stayed at the inn more than once during the year, with one of the stays falling completely inside the month of July. Report customer names (first, last) in alphabetical order. Each name shall be reported once.

6. For each reservation that starts on February 1, 2010 report the room name, nightly rate, number of nights spent and the total amount of money paid. Sort output in descending order by the number of nights stayed.

7. Find the names of all people[3] staying at the inn at the same time as `LENA LEDOUX`. Sort the output in alphabetical order by last name.

8. Find all occupants of `'rustic'` rooms on the night of the Halloween. Report names of occupants (last, first), full name of the room they occupy, total number of occupants and the checkin date. Sort the output in chronological order by checkin date.

9. Find all reservations for April (both checkin and checkout dates) where two adults are staying with two children. Report reservation code, full room name, checkin and checkout dates and the number of kids. Sort in chronological order by checkin date, then by checkout date.

10. Order all reservations that commenced on a Friday of the month of May[4] by the total amount paid for the reservation. For each reservation report the room name, the last name of the person staying, checkin date, number of nights and the total paid.

MARATHON dataset

For this dataset, all times must be outputed in the same format as in the original dataset (in the file `marathon.csv`). The information needs are below. Name the file `MARATHON-info.sql`.

[3] We only know the names of the people who made the reservations, so only those names are subject to the query.

[4] look up the dates in the calendar

1. Find all towns not in Massachussetts, Rhode Island or Connecticut, which sent a female runner in the 20-39 age group. Report town name and state name. Sort the output in alphabetical order by state, then by town names. Each town needs to be reported only once.

2. Find the results of all male runners from the from `W ATTL, MA`. Output first and last name of the runner, their overall place in the race and their time. Sort the output in ascending order by the overall place.

3. Find the results for all 33-year old female runners from Rhode Island. For each runner, output name (first, last), town, state, and the running time. Sort by time.

4. Find all women who ran the race faster than the race participant with Bib number 399 but slower than `KEN HAEDER`. For each runner report first and last name, age group, overall place in the race, place within the age group and running time. Sort by the overall place.

5. List all runners who took first place and second place in their respective age/gender groups. For age group, output name (first, last) and age for both the winner and the runner up (in a single row). Order the output by gender, then by age group.

6. Report the paces, in (`'MI:SS'` format) of all male runners from Connecticut whose overall time was better than 1 hour, 29 minutes and 22 seconds. Report names of the runners and their pace, age group and the overall place in the race. Sort by pace in ascending order.

AIRLINES dataset

1. Find all airlines that have at least one flight out of `NDV` airport. Report the full name and the abbreviation of each airline. Report each name only once. Sort the airlines in alphabetical order.

2. Find all airlines that have at least one flight to `CVO` airport. Report the full name and the abbreviation of each airline. Report each name only once. Sort the airlines in alphabetical order.

3. Find all airlines that have at least one flight from `NDV` airport and at least one flight to `CVO` airport. Report the full name and the abbreviation of each airline. Report each name only once. Sort the airlines in alphabetical order.

4. Find all airlines that have at least one flight between `NDS` and `CVO` airports. Report the full name and the abbreviation of each airline. Report each name only once. Sort the airlines in alphabetical order.

5. Find all airports with a direct flight from `NDV` airport. For each airport report its full name and the airport code. Sort output by the airport code.

6. Find all airports wih no direct flight from `NDV` airport. For each airport report its full name and the airport code. Sort output by the airport code.

7. We want to fly from `NDV` airport to `KKI` airport with just a single connection. Find the list of connecting airports (i.e. airports that have a flight from `NDV` and a flight to `KKI`). For each airport report its full name and the airport code. Report each airport only once.

8. We want to fly from `NDV` airport to `KKI` airport with just a single connection. For each such connection, specify the names of the airlines for each of the flights.

9. Report all pairs of airports served by both `Frontier` and `JetBlue`. Each pair must be reported exactly once (if a pair X,Y is reported, than a pair Y,X is redundant and should not be reported).

WINE dataset

Create a SQL script `WINE-info.sql` containing SQL statements representing the following information needs.

1. List all AVAs located in San Luis Obispo County. Output just the names of the AVA appellations and sort them in alphabetical order.

2. List all `Central Coast` appellations that are not located in San Luis Obispo county. List the appellation name, county and whether it is an AVA. Sort output by county name, and by appellation name within the county.

3. List all AVAs located in the same area as the `Shenandoah Valley` AVA. List the name of the AVA and the county it is in. Do not list `Livermore Valley`.

4. List all red grape varieties for which at least one wine of the 2009 vintage is rated in the database. Each grape variety needs to be reported once. Sort the output in alphabetical order.

5. List all Sonoma county appellations for which the database contains at least one rating for a `'Grenache'`. For each appellation list its name and county. Sort output in alphabetical order by county, then by appellation name. Report each appellation once.

6. List all vintage years in which at least one `Chardonnay` from Napa County (any appellation) scored above 94. Each year needs to be reported once. Sort in chronological order.

7. List all `'Pinot Noir'` wines with scores above 94 sorted in descending order by production quantity. For each wine, list the winery, the name, vintage, score and price.

8. A case of wine is 12 bottles. For each Kosta Browne (name of the winery) Pinot Noir compute the total revenue assuming that all the wine sold at the specified price. Report the name of the wine, its vintage wine score and overall revenue. Sort in descending order by revenue. Exclude NULL values.

9. List all 2006 vintage wines from Napa (any appellation within the county) whose total revenue exceeds that of the 2006 'Appelation Series'[5] Paso Robles Zinfandel from 'Rosenblum' winery. For each wine report grape, winery and name, score and revenue. Order by revenue.

10. Find all wines produced in the same vintage year as the Tor Chardonnay, which have both the higher score and the higher production.

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.ext where ext is one of the extensions above. The archive shall contain eight directories: AIRLINES, CARS, CSU, INN BAKERY, STUDENTS, MARATHON and WINE.

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 script. cat all <DATASET>-build-<table>.sql scripts together into one big script. 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 (as specified above) <DATASET>-info.sql (e.g., CARS-info.sql.

Submit using handin:

```
$ handin dekhtyar lab04 <file>
```

[5] There is a typo there. Let it be for now.