Lab 6: Full Power of SQL

Due date: Tuesday, November 20, before lab.

Note: Lab 7 may be assigned *before* Lab 6 is officially due.

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 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 may be quite complex and to address them, the use of aggregation, grouping, nested queries or their combinations may be required.

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.

Each information needs needs to be addressed with a *single SQL statement*, but the statement can have multiple levels of nesting, and use `MINUS`, `UNION` and `INTERSECT` operations.

Note: In this lab, we use only seven databases. There are no queries for the AIRLINES database.

STUDENT database

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

1. Find the grade(s) with the largest number of classrooms. Report the grade and the number of classrooms in it.
2. Find the teacher with the fewest students. Report the teacher (first name, last name) and the classroom.
3. Find how many classrooms have the number of students that is below the average number of students per class. Report just the number.
4. Find all grades in which the number of students is smaller than the number of students in fourth grade. Report just the grades.
5. Find all pairs of grades with the same number of students in them. Report each pair only once. Report both grades and the number of students.

BAKERY database

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

1. Find all customers who purchased neither `Danish`es nor `Chocolate Meringue`s from the bakery. Report first and last name of each customer. Sort output in alphabetical order by customer's last name.
2. For each customer of the bakery who made more than the average number of purchases, find the total number of purchases (i.e., reciepts associated with that customer) and the total amount of money spent. Report the first and the last name of the customer, the number of purchases and the total amount; sort output in descending order by the total amount of money spent.
3. Find the customer(s) who spent the most money at the bakery in October of 2007. Report first and last name.
4. Find the type of baked good responsible for lowest total revenue. Report the name of the pastry (flavor, food) and the total revenue it generated.
5. Find the most popular (by number of purchases) item. Report the item (food, flavor) and the number of times it was purchased.

6. Find the day of the lowest revenue in the month of October (of 2007). Report the date and the revenue.

7. For every customer who DID NOT make a purchase on the day of the **highest revenue**, report the total number of purchases (overall) the customer made, and the total amount of those purchases. Order the output by the total amount of purchases.

8. For every customer report the item they purchased most often as long they have purchased their favorite item four times or more in the month of October of 2007. Report customer name (first, last) and the name (flavor, food) of the item, as well as the total number of purchases. Sort in alphabetical order by last name of the customer. (Note: some customers may have purchaased more than one item the same number of times. All such items shall be displayed).

9. Output the names of all customers who made multiple purchases (more than one receipt) on the earliest day in October on which they made a purchase. Report names (first, last) of the customers and the earliest day in October on which they made a purchase, sorted in chronological order.

10. For each customer output the number of different pastry types they never bought in October 2007. Report the first and the last name of the customer, the number of pastry types. Sort output in descending order by the number of pastry types.

CARS database

1. Report the most powerful (in terms of horsepowers) vehicles in the database. For each vehicle, report its full name and the year of production.

2. Among the vehicles with the best acceleration, report the most powerful (horsepower) one. Report full name and the year of production.

3. Find the automaker that produced multiple vehicles in 1979, whose 1979 vehicles had the worst average gas milage. Report the automaker, the number of vehicle models it produced in 1970 and the average gas milage. (Note: exclude automakers with a single vehicle in 1979 from consideration completely).

4. For each year from 1970 to 1975 inclusively find the automakers whose models for that year had the best average acceleration. Report the year, the automaker, the number of models produced that year and the acceleration. Present the output in chronological order. (There may be some years when multiple automakers had that achievement. Report all such cases.)

5. Find the least fuel-efficient 4-cylinder model. Report the full name of the car, the year it was produced and the home country of its maker.

6. Find the difference in gas milage between the most fuel-efficient 8-cylinder model and the least fuel-efficient 4-cylinder model. Report just the number.

7. For each country report the number of 4-cylinder models its companies have produced in the 1970s which have higher horsepower than some 8cylinder model also produced in the 1970s. (note, the 8-cylinder model can come from any country and any company).

CSU database

Here are the queries for the CSU dataset.

1. For each campus with 2004 enrollment over 20,000 students 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 percentage of engineering students presented in the column named `"Eng_Percentage"`. Sort output in descending order by percentage.

2. For each campus with 2004 enrollment over 20,000 students 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 in descending order by percentage.

3. Find the campus with the largest undergraduate enrollment in 1965. Output the name of the campus and the total undergraduate enrollment.

4. Find the university that granted the largest total number of degrees over the entire recorded history. Report the name of the university and the total number of degrees.

5. Find the university with the best student-to-faculty ratio in 2004. Report the name of the campus, total undergraduate enrollment, faculty FTE and the student-to-faculty ratio. Recall that you want *fewer* students per faculty.

6. Find the university with the largest percentage of the undergraduate student body in the `'Computer and Info. Sciences'` discipline in 2004. Output the name of the campus and the percent of the engineering students on campus.

7. For each year between 2000 and 2004 (inclusive) report the campus with the highest absolute increase in enrollement from previous year.

Output the year, the campus name and the increase. Sort output in chronological order.

Note: if a university started accepting students in year $n \geq 2000$ for the first time, information about this university need not be captured in the process of determining the campus with the best increase in enrollment for year n. That is: only consider a campus in year n if it enrolled students in year $n-1$.

8. For each year between 2000 and 2004 (inclusive) find the university with the best (highest) total degrees granted to total enrollment (use enrollment numbers) ratio. Report the years, the names of the campuses and the ratios in chronological order.

9. For each university with an undergraduate engineering program in 2004 (i.e., with a non-zero number of engineering undergraduates) report the year of the lowest student-to-faculty ratio (use enrollment FTE and faculty FTE numbers). Output campus name, year and the ratio in alphabetical order by campus name.

INN database

1. Find the most popular month in the hotel. The most popular month is the month during which the largest number of reservations originated. Report the month and the total number of reservations in it. (Month can be reported in any readable format, e.g., `'JUL'` or `'OCTOBER'`).

2. Find the room that has been occupied the least based on the reservations in the database[1]. Report the room name, room code and the number of days it was occupied.

3. Find the most expensive reservation(s) made. Report the room name (full), dates of stay, last name of the person who made the reservation, daily rate and the total amount paid.

4. For each month, report the most expensive reservation that originated in it. Report the month, full room name, dates of stay, last name of the person who made the reservation, daily rate and the total amount paid. Sort output in *chronological* order by month.

5. For each room, report the total revenue the room has generated off of the reservations and the percentage of the overall hotel revenue the room reservations account for. Sort the rooms in descending order by the percentage.

6. Find the room that housed the largest number of people. Report the full name of the room and the total number of people who stayed in it.

[1] No need to limit the number of occupied days to 2010.

best month (i.e., month with the highest total revenue). Report the month, the total number of reservations and the revenue. For the purposes of the query, count the entire revenue of a stay that commenced in one month and ended in another towards the earlier month. (e.g., a September 29 - October 3 stay is counted as September stay for the purpose of revenue computation).

7. For each room report whether it is occupied or unoccupied on October 22, 2010. Report the full name of the room, the room code, and put either `'Occupied'` or `'Empty'` depending on whether the room is occupied on that day. (the room is occupied if there is someone staying the night of May 19, 2010. It is NOT occupied if there is a checkout on this day, but no checkin). Output in alphabetical order by room code.

8. For each room report how many reservations were made for the least expensive rate for that room. Report full room name and the appropriate number of reservations. Sort the output in ascending order by the number of reservations.

9. For each month report the room that generated the largest revenue for that month. Count a stay towards the revenue of the month if the stay originated on that month (e.g., a Jan 30 to Feb 10 stay counts as January revenue). Report the months in chronological order and the full names of the rooms.

MARATHON database

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 state with the largest number of participants.

2. Find all towns in Rhode Island (`RI`) which fielded more female runners than male runners for the race. Report the names of towns.

3. Find all towns in New Hampshire (`'NH'`) which had at least one runner in the 20-39 age group who was slower than at least one runner from `'QUECHEL'`, `'VT'`. Report just the names of the towns.

4. Find all towns in New Hampshire (`'NH'`) which had **exactly one** runner in the 20-39 age group who was slower than at least one runner from `'QUECHEL'`, `'VT'`. Report just the names of the towns.

WINE dataset

1. Find the grape(s) that grow(s) in the largest number of appellations. Report grape name, color and the number of appellations it grows in.

2. Find the most popular red grape (i.e., the grape that is used to make the largest number of red wines in the database). Report the name of the grape.

3. Find the highest-scoring Zinfandels from Sonoma county. Report the vintage, winery, name of the wine and its score.

4. Report the grape with the largest number of high-ranked wines (score of 94 and above).

5. Report the appellation responsible for the largest number of high-ranked wines (score of 94 and above). Report just the name of the appellation.

6. Find the high-ranked wine (score of 94 or above) responsible for highest sales revenue. Report the vintage year, winery, wine name, score and the computed revenue.

7. Find the highest-ranked cheap (price does not exceed $18) red wine from Napa couny. Report grape, winery, wine name, appelation, score and price.

8. Find if there are any 2008 Chardonnays that scored better than any 2005 Pinot Noir. Report winery, wine name, appellation, score and price.

9. Two California AVAs, Carneros and Dry Creek Valley have a bragging rights contest every year: the AVA that produces more highly-ranked (92 points or above) wines wins. Based on the data in the database, output (as a single tuple) the number of vintage years each AVA has won between 2005 and 2009. (It is OK if the query reports the results only for the years when BOTH AVAs produced highly-ranked wines.)

10. Find how many cases were produced of the most expensive red wine from Central Coast county. (Note: it's ok for this query to return multiple rows)

11. Find the winery with the largest total number of wines in the database and report the name of the winery, total number of wines and the total sales revenue that their wines generate.

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 `lab6.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 `lab5-adekhtyar.zip` or `lab4-adekhtyar.tar.gz`.

The archive shall contain seven directories: `CARS`, `CSU`, `BAKERY`, `INN`, `STUDENTS`, `WINE` and `MARATHON`. (note: `AIRLINES` directory is not required, but it is ok to submitted. It's just not graded, since we don't have any queries for the AIRLINES dataset.)

Each directory shall contain the following SQL scripts:

- Database creation (<DATABASE>-setup.sql), database population (<DATABASE>-insert.sql) and database cleanup (<DATABASE>-cleanup.sql) scripts from Lab 4.
- **NEW script.** One script per database, containing all SQL statements and any SQL*plus statements needed for formatting. Name the script <DATASET>-queries.sql (e.g., CARS-queries.sql.

Note: Please do not use any spool commands in your scripts.

Submit:

```
$handin dekhtyar lab06 <file>
```