Lab 6: Counting with SQL

Due date: Friday, May 26, **midnight**.

Note: Lab 7, the last SQL lab, will be assigned on Friday, May 26 in class.

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.

You will continue using instructor's databases you used in Labs 4 and 5. The read-only versions of these databases are available on the MySQL server as databases with the following names:

```
airlines
bakery
cars
csu
inn
katzenjammer
marathon
students
wine
```

Each of you has been granted the `SELECT` priveleges on each of these databases (contact the instructor if this is not the case). Your queries must be written for the tables in these databases and must run properly on them and produce correct output.

You will prepare one SQL script for each database. Please note: every row of every resulting table must be printed in a single line.

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. Each information need in this lab can (and must) be represented by either a single SELECT statement (possibly including aggregate operations, GROUP BY and HAVING clauses), or by a number of SELECT statements combined using the UNION operator. For this assignment, you will prepare one SQL script for each database.

General Note: In queries that start with the phrase "For each YYY report ...", you are expected to include the column representing YYY in your output. For example, the query "For each grade report the sum of all classrooms" should result in a query that outputs two columns: `GRADE` and `SUM(CLASSROOM)`. This applies to all datasets and all upcoming labs as well.

Filenames. For this lab, name the SQL scripts containing your queries `<DATASET>-count.sql`. E.g., for the `CARS` dataset, the script file name is `CARS-count.sql`.

STUDENTS dataset

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

1. Report the names of teachers who have between seven and eight (inclusive) students in their classes. Sort output in alphabetical order by teacher's last name.
2. For each grade, report the number of classrooms in which it is taught and the toal number of students in the grade. Sort the output by the number of classrooms in descending order, then by grade in ascending order.
3. For each kindergarden classroom, report the total number of students. Sort output in the descending order by the number of students.
4. For each fourth grade classroom, report the student (last name) who is the last (alphabetically) on the class roster. Sort output by classroom.

BAKERY dataset

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

1. For each pastry flavor which is found in more than three types of pastries sold by the bakery, report the average price of an item of this flavor and the total number of different pastrties of this flavor on the menu. Sort the output in ascending order by the average price.

2. Find the total amount of money the bakery earned in October 2007 from selling eclairs. Report just the amount.

3. For each purchase made by `NATACHA STENZ` output the receipt number, the date of purchase, the total number of items purchased and the amount paid. Sort in descending order by the amount paid[1].

4. For each day of the week of October 8 (Monday to Sunday) report the total number of purchases (receipts), the total number of pastries purchased and the overall daily revenue. Report results in chronological order and include both the day of the week and the date. [2].

5. Report all days on which more than ten tarts were purchased, sorted in chronological order.

CARS dataset

1. For each Japanese car maker (reported by their short name) report the best mileage per gallon of a car produced by it and the average acceleration. Sort output in ascending order by the best mileage.

2. For each US car maker (reported by their short name), report the number of 4-cylinder cars that are lighter than 4000 lbs with 0 to 60 mph acceleration better than 14 seconds. Sort the output in descending order by the number of cars reported.

3. For each year in which honda produced more than 2 models, report the best, the worst and the average gas milage of a toyota vehicle. Report results in chronological order.

4. For each year when US-manufactured cars averaged less than 100 horsepowers, report the highest and the lowest engine displacement number. Sort in chronological order.

CSU dataset

Here are the queries for the CSU dataset.

1. For each campus that averaged more than $2500 in fees between 2000 and 2005, report the total cost of fees for this five year period. Sort in ascending order by fee.

[1]The total amounts paid may look strange, if you are using floating points for prices.
[2]The total amounts paid may look strange, if you are using floating points for prices.

2. For each campus for which data exists for more than 60 years, report the average, the maximum and the minimum enrollment (for all years). Sort your output by average enrollment.

3. For each campus in LA and Orange counties report the total number of degrees granted between 1998 and 2002 (inclusively). Sort the output in descending order by the number of degrees.

4. For each campus that had more than 20000 enrolled students in 2004 report the number of disciplines for which the campus had non-zero graduate enrollment. Sort the output in alphabetical order by the name of the campus. (This query should exclude campuses that had no graduate enrollment at all).

MARATHON dataset

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

Note: please remember that the **best**, i.e., the **fastest** time is the smallest one!

1. For each gender/age group, report total number of runners in the group, the overall place of the best runner in the group and the overall place of the worst runner in the group. Output result sorted by age group and sorted by gender (F followed by M) within each age group.

2. Report the total number of gender/age groups for which both the first and the second place runners (within the group) hail from the same state.

3. For each full minute, report the total number of runners whose pace was between that number of minutes and the next. (That is, how many runners ran the marathon at a pace between 5 and 6 mins, how many - at a pace between 6 and 7 mins, and so on).

4. For each state, whose representatives participated in the marathon report the number of runners from it who finished in top 10 in their gender-age group (if a state did not have runners in top 10s, do not output information about the state). Output in descending order by the computed number.

5. For each `CT` town with 3 or more participants in the race, report the average time of its resident runners in the race *computed in seconds*. Output the results sorted by the average time (best average time first).

AIRLINES dataset

1. Find all airports with exactly 17 outgoing flights. Report airport code and the full name of the airport sorted in alphabetical order by the code.

2. Find the number of airports from which airport `ANP` can be reached with exactly one transfer. (make sure to exclude `ANP` itself from the count). Report just the number.

3. Find the number of airports from which airport `ATE` can be reached with *at most* one transfer. (make sure to exclude `ATE` itself from the count). Report just the number.

4. For each airline report the total number of airports from which it has at least one outgoing flight. Report the full name of the airline and the number of airports computed. Report the results sorted by the number of airports in descending order.

INN dataset

1. For each room report the total revenue for all stays and the average revenue per stay generated by stays in the room that originated in the months of September, October and November. Sort output in descending order by total revenue. (Output full room names).

2. Report the total number of reservations that commenced on Fridays and the total revenue they brought in. (*Hint:* look up the date of the *first* Friday on the calendar).

3. For each day of the week, report the total number of reservations commenced on it and the total revenue these reservations brought. Report days of week as `Monday`, `Tuesday`, etc.

4. For each room report the highest markup against the base price and the smallest markup (i.e., largest markdown). Report markups and markdowns in absolute terms (absolute difference between the base price and the rate). Sort output in descending order by the absolute value of the largest markup. Report full names of the rooms.

5. For each room report how many nights in 2010 the room was occupied. Report the room code, the full name of the room and the number of occupied nights. Sort in descending order by occupied nights. (Note: it has to be *number of nights in 2010* - the last reservation in each room *may* and *will* can go beyond December 31, 2010, so the "extra" nights in 2011 need to be deducted).

   Note/Hint: This is almost an extra credit problem. While multiple solutions are possible, my solution uses SQL's `SIGN()` built-in function which returns -1 for negative numbers, +1 for positive numbers and 0 for 0.

WINE dataset

1. For each wine score value above 88, report average price, the cheapest price and the most expensive price for a bottle of wine with that score

(for all vintage years combined), the total number of wines with that score and the total number of cases produced. Sort by the wine score.

2. For each year, report the total number of red Sonoma County wines whose scores are 90 or above. Output in chronological order.

3. For each appellation that produced more than two Cabernet Sauvingnon wines in 2007 report its name and county, the total number of Cabernet Sauvingnon wines produced in 2008, the average price of a bottle of Cabernet Sauvingnon from that vintage, and the total (known) number of bottles produced[3]. Sort output in descending order by the number of wines.

4. For each appellation inside Central Coast compute the total (known)[4] sales volume that it can generate for the wines produced in 2008. Sort the output in descending order by the total sales volume. (Note: recall what a case of wine is).

5. For each county in the database, report the score of the highest ranked 2009 red wine. Exclude wines that do not have a county of origin (`'N/A'`). Sort output in descending order by the best score.

KATZENJAMMER dataset

1. For each performer (use first name) report how many times she sang lead vocals on a song. Sort output in descending order by the number of leads.

2. Report how many different unique instruments each performer plays on songs from `'Le Pop'`. Sort the output by the first name of the performers.

3. Report the number of times Turid stood at each stage position when performing live. Sort output in ascending order of the number of times she performed in each position.

4. Report how many times each of the remaining peformers played `bass balalaika` on the songs where Anne-Marit was positioned on the left side of the stage. Sort output alphabetically by the name of the performer.

5. Report all instruments (in alphabetical order) that were played by three or more people.

6. For each performer, report the number of times they played more than one instrument on the same song. Sort output in alphabetical order by first name of the performer[5].

[3]Recall, one case is 12 bottles.
[4]Recall, that information about production volumes for some wines is not available.
[5]Yes, you can do it without using nested queries!

Submission Instructions

You must submit nine `<DATASET>-count.sql` files. In addition, submit a simple `README` file with your name and email address.

There is no need to submit any other files.

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 `ext` is one of the extensions above.

When unpacked, your archive must place the nine `<DATASET>-count.sql` files into the current directory (i.e., in the root of your `handin` directory for Lab 6 submission). No other subdirectories are needed.

Submit using handin:

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