Lab7

Motivation for Database Management

1) In your future career, your company/team may use various databases, hence familiarity with SQL would be helpful to understand better the nature of those databases, how to use them better, how to optimize them, etc.

2) Hardware engineers are responsible for the servers that manage databases. Understanding how a database works, helps you with design and optimization steps of those servers

3) Databases, cloud centers and datacenters are very much related.

4) Large scale distributed DBMS (database management systems) used for big data of machine learning and cloud computing applications depend on efficient DBMS.

References

There are tons of books and tutorials out there for SQL, Relational database models and RDBM. Here is just some of them. Feel free to pick any one of them and try to read as much as you can while working on the lab. If you have no preferences, pick the first one on this list:

1) Database Systems: Design, Implementation, & Management - Carlos Coronel, Steven Morris

https://www.amazon.com/Database-Systems-Design-Implementation-Management/dp/1305627482/ref=sr_1_3?s=books&ie=UTF8&qid=1522859414&sr=1-3&keywords=Database+Systems%3A+Design%2C+Implementation%2C+%26+Management&dpID=5

2) Database System Concepts, 6E
Abraham Silberschatz / Henry F.Korth /S.Sudarshan
https://www.amazon.com/Database-Concepts-Abraham-Silberschatz-Professor/dp/0073523321

3) Pro SQL Server 2012 RDB and implementation
https://www.amazon.com/Relational-Database-Implementation-1-Jul-2012-Paperback/dp/B013J8XC4A/ref=sr_1_1?s=books&ie=UTF8&qid=1522859217&sr=1-1&keywords=pro+2012+sql+server+relational

4) Introducing Microsoft SQL Server 2016 Mission-Critical Applications, Deeper Insights, Hyperscale Cloud
Stacia Varga, Denny Cherry, Joseph D'Antoni

5) Fundamentals of database systems, Ramaz
https://www.amazon.com/Fundamentals-Database-Systems-Ramez-Elmasri/dp/0133970779

6) An in troduction to relational database theory by H. Darwen.
https://bookboon.com/en/an-introduction-to-relational-database-theory-ebook

Lab7

EE 599 Lab7
Spring 2019 Nazarian

Assigned: Wednesday Feb. 20
Due: Friday Mar. 1 at 11:59pm
Late submissions will be accepted for two days after the deadline with a maximum penalty of 15
between 12am and 1am: 2% penalty, between 1 and 2am: 4%, 2-3am: 8%, and after 3am: 15
March 3rd, after 11:59pm.

Notes:

This assignment is based on individual work. No collaboration is allowed (Discussing the technical details of each problem with other students before submitting your work, copying from students of current or previous semesters is not permitted in our lab and HW assignments). You may post your questions on the discussion forums and/or use the office hours. We may pick some students in random to demonstrate their design and simulations. Please refer to the syllabus and the first lecture regarding the USC and our policies, including the penalties for any violation. If you have any doubts about what is allowed or prohibited in this course, please contact the instructor.

What You Will Practice

In this lab, you will code solutions to the several SQL problems given below. For all SQL questions, you are not allowed to make any modifications on the table itself. Changing data records is not allowed.

Q1: Student course registration (30%)

A university uses a table ***Student_registration*** to save students' course registration information. The table is shown in below:

student_name	stu_id	course	status
Amy	60000	EE504	C
Amy	60000	EE501	C
Amy	60000	EE505	C
Joe	70000	EE500	C
Joe	70000	EE501	IP
Joe	70000	EE504	C

Henry	80000	EE500	C
Henry	80000	EE502	C
Henry	80000	EE503	C
Sam	62000	EE501	C
Max	90000	EE502	IP
Max	90000	EE503	C

Column **status** indicates whether the student completed ('C') the course or is still taking the course in progress ('IP').

Write a SQL query to find the students that already complete all the courses. i.e., no course is incomplete. Sort the result ascendingly by their student ID.

Example output:

name	**stu_id**
Amy	60000
Sam	62000
Henry	80000

Q2: Course Units (30%)

Apart from **Student_registrition** table, the university uses another table **Courses** to save the number of units for each course. Shown below:

course	**units**
EE500	3
EE501	4
EE502	3
EE503	7
EE504	4
EE505	4
EE506	3

Write a SQL query to find the students who have completed all the courses (no IP) and the total completed units is > 10.

Example output:

Name	**Units_complete**

Amy	12
Henry	13

Q3: Employee Bonus (15%)

Given the tables for employee and the bonus they received in the last year:

Table: Employee

empId	name	supervisor	salary
1	John	3	1000
2	Dan	3	2000
3	Brad	null	4000
4	Thomas	3	4000

(empId is the primary key column for this table.)

Table: Bonus

empId	bonus
2	500
4	2000

(empId is the primary key column for this table.)

Write a query that selects all employee's name and bonus whose bonus is < 1000.

Example output:

name	bonus
John	null
Dan	500
Brad	null

Q4: Yummy Desserts (25% + 10%)

Below is a table of chefs and the yummy desserts they know to make:

Table: ChefSkill

Chef	Dish
A	Mint chocolate brownie

B	Upside down pineapple cake
B	Creme brulee
B	Mint chocolate brownie
C	Upside down pineapple cake
C	Creme brulee
D	Apple pie
D	Upside down pineapple cake
D	Creme brulee
E	Apple pie
E	Upside down pineapple cake
E	Creme brulee
E	Bananas Foster

Write a query that will pick out all those chefs who can make every item in the table below (maybe someone wants us to cater a weekend party and requests these dishes, and we'd like to pay just one chef from our query result, overtime wages to make the dishes):

Table: Menu

Dish
Upside down pineapple cake
Apple pie
Creme brulee

For example, with the above data, the query would output

Chef
D

E

The menu items are just samples, and your query should work for ANY such table. For this problem, you are required to formulate at least two different queries. (submit two files, eg. q4_v1.sql and q4_v2.sql). Note that the approaches do have to be totally distinct, eg. you can't use NOT to invert an existing solution, or use IN() instead of OR, etc.
If you come up with YET ANOTHER very different way, you can get 10 more bonus point (submit yet another file, eg. q4_v3.sql).

Notes

Attached Files: q4.sql (785 B)
q12.sql (833 B)
q3.sql (481 B)

Notes

- It is recommended to use online tool http://www.sqlfiddle.com/ to test your SQL commands. You may also choose to install the database locally and test your code.
- DDL code to define the SQL schema (written in MySQL 5.6) for each problem is given for your reference, which you can use create the table. Note that the DDL code for question 1 and 2 are provided together as q12.sql.
- You are allowed to use any database (eg. MySQL, Oracle, SQLite, PostgreSQL). PLEASE MENTION AT THE TOP OF EACH FILE, which database you used for that question. If you are using MySQL, you can just copy and paste the DDL code provided, otherwise you may need to modify the syntax accordingly.
- What you need to submit are text files with the **DDL code to define SQL schema** (q12.sql, q3.sql, q4.sql) as well as the **SQL DML query commands** that you come up with, one file for each solution (q1_sol.sql, q2_sol.sql, q3_sol.sql, q4_sol_v1.sql and q4_sol_v2.sql).

Submission

Github Link to accept the assignment invitation:
https://classroom.github.com/a/61tne_7r

Please push all .sql files you wrote into the GitHub repo.

Don’t forget to submit a readme file:

- **Your readme file should be named as readme_<STUDENT-ID>.txt, please replace <STUDENT-ID> with your own 10-digit USC-ID.**
- Any non-working part should be clearly stated.
- The citations should be done carefully and clearly, e.g.: “to write my code, lines 27 to 65, I used the Djkstra's shortest path algorithm C++

code from the following website: www.SampleWebsite.com/...”

You'll lose points if you:

- don't name your files properly
- submit a .zip (you need to submit individual files instead)
- neglect to mention what DB software you used

Main TAs for this lab

Junting, Suofei, and Ananya