Department of Computer Science and Engineering

Course Code: CSE 3122	
Course Name: Database Systems	

Lab 03

SQL Subqueries & Aggregate Functions

I. Topic Overview:

In this lab, the students will explore several SQL queries to get acquainted with some advanced SQL features including advanced Select queries, Subqueries & Aggregate functions. Some query-related problems will also be explored to understand how these queries work.

II. Lesson Fit:

This whole lab is a continuation of Lab 02. This lab requires the database folder generated & backed up in the previous lab.

III. Learning Outcome:

After this lecture, the students will be able to:

a. Use Aggregate functions, Group by, Having & Order by clauses in SQL queries
b. Retrieve data using Subqueries /nested queries.
c. Learn about correlated Subqueries.

IV. Anticipated Challenges and Possible Solutions

a. Student may not understand the relation or difference between **where** clause & **having** clause.

Solutions:

Lab instructor will discuss the relation & contrast between where clause & having clause in the lab.

V. Acceptance and Evaluation

Students will show progress as they execute each query successfully in the command window. As this is a practice-and-learn-type lab. Students won't be evaluated in this lab.

VI. Activity Detail

a. **Hour: 1**

Discussion: Give a brief overview of the related queries as well as their syntaxes with some examples stated in the activity list section. The SQL clauses to be covered are – different aggregate functions (count, max, min etc.), group by, having & order by. Explain the mechanism of how Subqueries & correlated Subqueries work with the examples stated in the activity list.

b. **Hour: 2**

Problem Task: Students will complete all Tasks in activity list. They will ask the instructor in case of any issues.

c. **Hour: 3**

Discussion: Instructor will discuss all the tasks again to ensure that students have understood and executed all instructions accurately

Home Task: Complete all remaining tasks in case students couldn't finish within 3 hours.

Lab 03 Activity List

- **All commands are shown in the red boxes .**
- **In the green box write the appropriate query/answer.**
- **All new queries should be typed in command window after `mysql>`**
- **Start by connecting to server using: `mysql -u root -p` [password:`root`]**
- **For more MySQL queries, go to www.w3schools.com/sql or google it!**

We will use the same data as Lab 02. So if you have it with you, simply copy the folder to udrive>data. Then go to command window, login and use dbname. If you don't have it with you, you can borrow it from a friend! The below table is the database state after completing lab 02.

Std_ID	Name	Major	Days_present	Project_marks	CGPA	Sub_date
s001	Abir	CS	10	18.5	3.91	2018-09-15
s002	Nafis	CS	12	20	3.86	2018-08-15

s003	Tasneem	CS	8	18	3.57	2018-09-18
s005	Arafat	CSE	11	20	4.0	2018-09-13
s006	Tasneem	CSE	12	17.5	3.7	2018-08-15
s007	Muhtadi	ECE	10	19	3.67	2018-09-16

Task 1: Aggregate Functions, Group By and Having:

Retrieve the minimum CGPA/Project_marks from the table	Select min(*CGPA*) from ***Lab_Grades***;
Retrieve the total number of students and the average projects marks	Select count(*) as ***total_students***, avg(***Project_marks***) as ***average_project_marks*** from ***Lab_Grades;***
Find the sum of the number of days present.	Select sum(***Days_Present***) from ***Lab_Grades;***

- How will you retrieve the last submission date?

Find Minimum and Maximum CGPA/Project_marks of each major	Select ***major,*** min(***CGPA***) as ***minCGPA,*** max(***CGPA)*** as ***maxCGPA*** *from* ***Lab_Grades*** group by ***major;***
Retrieve total number of students for each major	Select ***major,*** count***(*)*** from ***Lab_Grades*** group by ***major;***

- What is the purpose of the group by keyword? In the above command if we group by sub_date, instead of major, what will be the output?

For each major find the minimum and maximum CGPA/Project_marks, but only if there were at least 2 students in the major	Select ***major,*** min(***CGPA***) as ***minCGPA,*** max(***CGPA)*** as ***maxCGPA*** *from* ***Lab_Grades*** group by ***major*** having count(*)>=2***;***
For each major find the minimum and maximum CGPA/Project_marks, but consider only students who submitted before or on 15th sep	Select ***major,*** min(***CGPA***) as ***minCGPA,*** max(***CGPA)*** as ***maxCGPA*** *from* ***Lab_Grades*** where ***sub_date<=***'2018-09-15' group by ***major;***

- The having and where clause both are used to specify a condition when selecting rows. What is the different between them?

Task 2: Sub Queries/Nested Queries, Any and All:

- Think about how you can retrieve the name of students who got the highest project marks. Try out your query, did you get the "correct" response according to the table?

Now, try the nested/sub query on the right

Select *Name* from *Lab_Grades* where *Project_marks*=(Select max(*Project_marks*) from *Lab_Grades*);

For each major find the name of the student who has the lowest attendance

Select *Major, Name, Days_present* from *Lab_Grades* where *(Major, Days_present)* in (Select *Major,* min(*Days_present*) from *Lab_Grades* group by *Major*);

- Why is the "in" operator used in the above command instead of "="?

Retrieve the CSE students whose CGPA/Project_marks is higher than at least 1 CS students

Select * from *Lab_Grades* where *Major* = 'CSE' and *CGPA*>any (Select *CGPA* from *Lab_Grades* where *Major* = 'CS');

Retrieve the CSE students whose CGPA/Project_marks is higher than all CS students

Select * from *Lab_Grades* where *Major* = 'CSE' and *CGPA*>all (Select *CGPA* from *Lab_Grades* where *Major* = 'CS');

- Did you understand the role of "any" and "all" in the above queries? Explain below.

- Retrieve the name of the students who have received marks greater than at least 1 student doing the same major as them.[Hint: see next command]

Task 3: Correlated Subqueries and Exists:

Select those majors for which at least 1 student has CGPA lower than 3.7/project_marks < 18

Select distinct ***Major*** from ***Lab_Grades L1*** where exists (Select * from ***Lab_Grades L2*** where ***L2.Major=L1.Major*** and ***L2.CGPA<3.7***);

- L1 and L2 are temporary aliases and create two separate instances for Lab_grades, why are they required?

Retrieve the name of student who has obtained maximum marks in project using exists

Select ***Name*** from ***Lab_Grades L1*** where not exists (Select * from ***Lab_Grades L2*** where ***L2.Std_ID!=L1.Std_ID*** and ***L2.Project_marks>L1.Project_marks);***

Retrieve the name of student who has obtained maximum marks in project and who is unique using exists

Select ***Name*** from ***Lab_Grades L1*** where not exists (Select * from ***Lab_Grades L2*** where ***L2.Std_ID!=L1.Std_ID*** and ***L2.Project_marks>=L1.Project_marks);***

- Please identify the difference between the above two queries. [Hint: 1 asks for unique-only 1 student got highest and the other doesn't]

Retrieve the total number of students who obtained the maximum marks. There are many ways of achieving one task, a few ways for this one is shown below

Select Count(*) from ***Lab_Grades L1*** where not exists (Select * from ***Lab_Grades L2*** where ***L2.Std_ID!=L1.Std_ID*** and ***L2.Project_marks>L1.Project_marks);***

Select Count(*) from ***Lab_Grades*** where***Project_marks*** = (Select max(***Project_marks)*** from ***Lab_Grades***);

Select Count(*) from ***Lab_Grades*** where ***Project_marks*** > all (Select ***Project_marks*** from ***Lab_Grades***);

Retrieve the major which has the highest number of students enrolled.

Select ***Major*** from ***Lab_Grades*** group by ***Major*** having Count(*) >= all (Select Count(*) from ***Lab_Grades*** group by ***Major***);

- The statement below is the general format for a "Select" statement. State what each of the keywords (marked in blue) are used for.

SELECT *column_name(s)*
FROM *table_name(s)*
WHERE *conditions*
GROUP BY *column_name(s)*
HAVING *conditions*
ORDER BY *column_name(s);*

- Write down the general format for "Insert", "Delete" and "Update" statements.

Task 4: Take a Quiz

Go to https://sqlzoo.net/wiki/Nested_SELECT_Quiz to test your understanding of the queries taught in class.