Department of Computer Science and Engineering

Course Code: CSE 370	Credits: 3.0
Course Name: Database Systems	Semester: Spring 24

Lab 02: SQL Subqueries & Aggregate Functions

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 the command window after mysql>**
- **Start by connecting to the server using: mysql –u root –p [password:root]**
- **For more MySQL queries, go to www.w3schools.com/sql or google it!**

Initial Table: It's a bit different than Lab 01!

std_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.5	3.91	2018-09-15
s002	Nafis	CSE	1	12	20	3.86	2018-08-15
s003	Tasneem	CS	1	8	18	3.57	2018-09-18
s004	Nahid	ECE	2	7	16.5	3.25	2018-08-20
s005	Arafat	CS	2	11	20	4.0	2018-09-13
s006	Tasneem	CSE	1	12	17.5	3.7	2018-08-15
s007	Muhtadi	ECE	1	10	19	3.67	2018-09-16

Link for Table Data: https://docs.google.com/document/d/1ZFFMN863k9GOjTG6ibbCAEEdqF3ExJzug-ymPON6ofA/

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 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 submission_date<='2018-09-1' GROUP BY major;
```

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

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

- Think about how you can retrieve the names 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);
```

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

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 names 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

```
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 the highest and the other didn’t]

Retrieve the total number of students who obtained the maximum marks. There are many ways of solving one task; a few ways for this one are 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 keyword (marked in blue) is used for.

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

Task 4: Take a Quiz

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