Heaven's Light is Our Guide

Rajshahi University of Engineering and Technology

Department of Electrical & Computer Engineering

Lab Report

Course Code: ECE 2216

Course Title: Data Base System Sessional

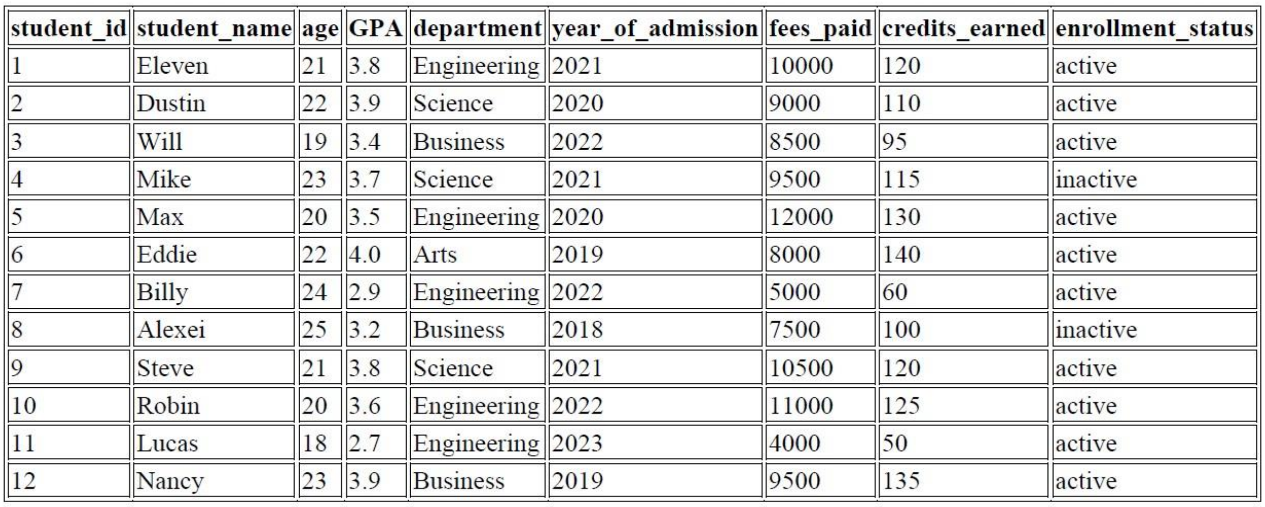
Lab Report no. 02

Submission Date: 30 September, 2024

Submitted To	**Submitted By**
Oishi Jyoti	Name: Mushfiqur Rahaman
Assistant Professor	Roll: 2110031
Department of Electrical and Computer Engineering, RUET	Department of Electrical and Computer Engineering, RUET

Problem Statement:

Students Table



student_id	student_name	age	GPA	department	year_of_admission	fees_paid	credits_earned	enrollment_status
1	Eleven	21	3.8	Engineering	2021	10000	120	active
2	Dustin	22	3.9	Science	2020	9000	110	active
3	Will	19	3.4	Business	2022	8500	95	active
4	Mike	23	3.7	Science	2021	9500	115	inactive
5	Max	20	3.5	Engineering	2020	12000	130	active
6	Eddie	22	4.0	Arts	2019	8000	140	active
7	Billy	24	2.9	Engineering	2022	5000	60	active
8	Alexei	25	3.2	Business	2018	7500	100	inactive
9	Steve	21	3.8	Science	2021	10500	120	active
10	Robin	20	3.6	Engineering	2022	11000	125	active
11	Lucas	18	2.7	Engineering	2023	4000	50	active
12	Nancy	23	3.9	Business	2019	9500	135	active

Tasks:

1. Find students who are older than 20 and have a GPA above the average GPA of all students.
2. Find the top 5 students with the highest fees paid, ordered by GPA (in descending order) as a tiebreaker.
3. List students who belong to the "Engineering" department, have a GPA greater than 3.5, and are enrolled after 2020.
4. Find students who are not active (i.e., enrollment_status = 'inactive') and have not paid any fees (fees_paid = 0).
5. Calculate the total fees paid and average GPA for each department, but only for departments with more than 10 students.

Tools:

1. XAMPP Control Panel
2. Laptop

Task 1: Find students who are older than 20 and have a GPA above the average GPA of all students.

Solving Procedure:

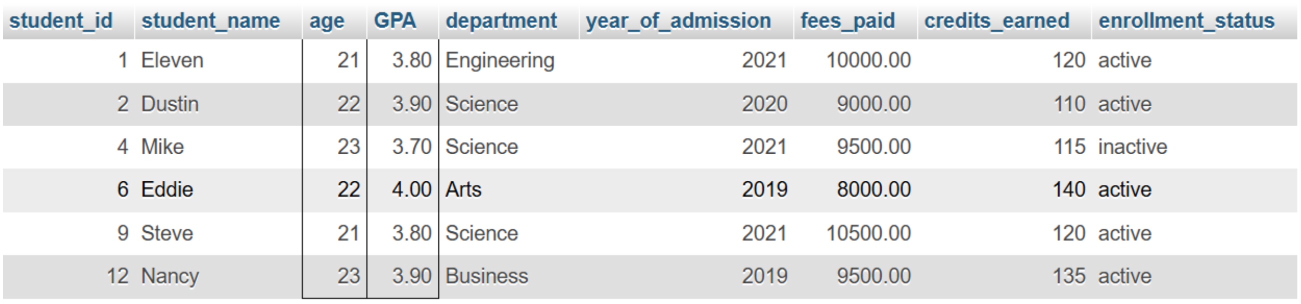
- Define table structure.
- Insert multiple rows of student data with corresponding values.
- Execute the SQL query.
- Calculate the average GPA of all students.
- Select students older than 20 and having a GPA above the calculated average GPA.

Query:

```
SET @avg_gpa = (SELECT AVG(GPA) FROM Students);

SELECT  student_id,  student_name,  age,  GPA,  department,  year_of_admission,
fees_paid, credits_earned, enrollment_status
FROM Students
WHERE age > 20 AND GPA > @avg_gpa;
```

Output:



student_id	student_name	age	GPA	department	year_of_admission	fees_paid	credits_earned	enrollment_status
1	Eleven	21	3.80	Engineering	2021	10000.00	120	active
2	Dustin	22	3.90	Science	2020	9000.00	110	active
4	Mike	23	3.70	Science	2021	9500.00	115	inactive
6	Eddie	22	4.00	Arts	2019	8000.00	140	active
9	Steve	21	3.80	Science	2021	10500.00	120	active
12	Nancy	23	3.90	Business	2019	9500.00	135	active

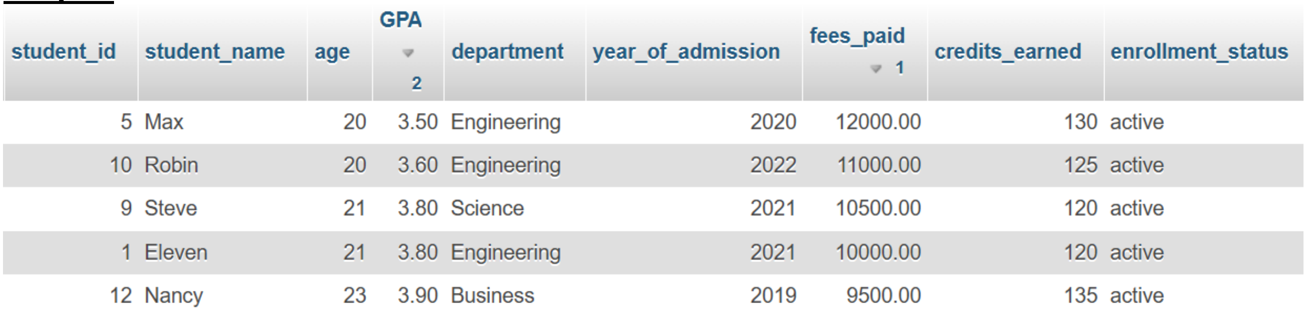
Task 2: Find the top 5 students with the highest fees paid, ordered by GPA (in descending order) as a tiebreaker

Solving Procedure:

- The SELECT statement retrieves columns of interest from the Students table.
- The ORDER BY clause sorts the results by fees_paid in descending order and then by GPA in descending order for tie-breaking.
- The LIMIT 5 clause restricts the output to the top 5 students.

Query:
```
SELECT  student_id,  student_name,  age,  GPA,  department,  year_of_admission,
fees_paid, credits_earned, enrollment_status
FROM Students
ORDER BY fees_paid DESC, GPA DESC
LIMIT 5;
```

Output:



student_id	student_name	age	GPA 2	department	year_of_admission	fees_paid 1	credits_earned	enrollment_status
5	Max	20	3.50	Engineering	2020	12000.00	130	active
10	Robin	20	3.60	Engineering	2022	11000.00	125	active
9	Steve	21	3.80	Science	2021	10500.00	120	active
1	Eleven	21	3.80	Engineering	2021	10000.00	120	active
12	Nancy	23	3.90	Business	2019	9500.00	135	active

Task 3: List students who belong to the "Engineering" department, have a GPA greater than 3.5, and are enrolled after 2020.

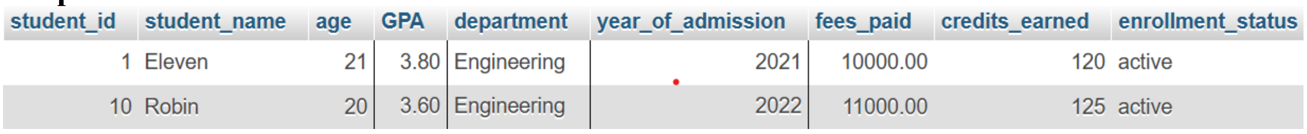
Solving Procedure:

- The SELECT statement retrieves columns of interest from the Students table.
- The WHERE clause filters the students based on the following conditions:
  - department = 'Engineering': The student must belong to the "Engineering" department.
  - GPA > 3.5: The student must have a GPA greater than 3.5.
  - year_of_admission > 2020: The student must have been enrolled after the year 2020.

Query:
```
SELECT  student_id,  student_name,  age,  GPA,  department,  year_of_admission,
fees_paid, credits_earned, enrollment_status
FROM Students
```

```
WHERE department = 'Engineering' AND GPA > 3.5 AND year_of_admission > 2020;
```

Output:



student_id	student_name	age	GPA	department	year_of_admission	fees_paid	credits_earned	enrollment_status
1	Eleven	21	3.80	Engineering	2021	10000.00	120	active
10	Robin	20	3.60	Engineering	2022	11000.00	125	active

<u>Task 4:</u> Find students who are not active (i.e., enrollment_status = 'inactive') and have not paid any fees (fees_paid = 0).

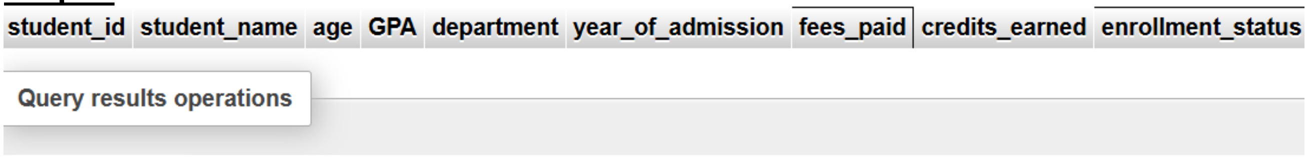
<u>Solving Procedure:</u>

- The SELECT statement retrieves columns of interest from the Students table.
- The WHERE clause filters the students to only those who are inactive (enrollment_status = 'inactive') and have not paid any fees (fees_paid = 0).

<u>Query:</u>

```
SELECT  student_id,  student_name,  age,  GPA,  department,  year_of_admission,
fees_paid, credits_earned, enrollment_status
FROM Students
WHERE enrollment_status = 'inactive' AND fees_paid = 0;
```

<u>Output:</u>



student_id	student_name	age	GPA	department	year_of_admission	fees_paid	credits_earned	enrollment_status

<u>Task 5:</u> Calculate the total fees paid and average GPA for each department, but only for departments with more than 10 students.

<u>Solving Procedure:</u>

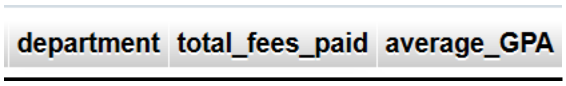
- First, create a temporary table to count the number of students in each department.
- Then, calculate the total fees and average GPA for departments that have more than 10 students.

<u>Query:</u>

```
CREATE TEMPORARY TABLE DepartmentCount AS
SELECT department, COUNT(*) as student_count
FROM Students
GROUP BY department;

SELECT s.department,
       SUM(s.fees_paid) AS total_fees_paid,
       AVG(s.GPA) AS average_GPA
FROM Students s
JOIN DepartmentCount dc ON s.department = dc.department
WHERE dc.student_count > 10
GROUP BY s.department;
```

<u>Output:</u>



department	total_fees_paid	average_GPA

<u>Discussion:</u>

The primary objective of this lab report was to analyze student data using MySQL queries, providing insights into various aspects of student performance and financial status. In Task 1, we identified students older than 20 with a GPA above the average, which highlights high-achieving students who may be potential candidates for advanced academic programs. This information could assist academic advisors in targeting support and resources effectively. Task 2 involved determining the top 5 students based on fees paid, revealing a correlation between financial investment and student performance. Understanding this relationship is crucial for the financial department as it informs budgeting strategies and fee structures. Furthermore, the analysis of departments in Task 5, where we calculated total fees and average GPA, provides essential data for academic resource allocation. The results indicate that certain departments may require additional funding or support to improve student outcomes. It is important to note the limitations of our dataset, which may not represent the entire student body. Future analyses could benefit from a more extensive dataset and an exploration of additional factors such as retention rates and student engagement levels. Overall, the findings from this report underscore the importance of data-driven decision-making in higher education, allowing institutions to enhance student success and optimize resource allocation.

References:

[1] W3Schools.com, "SQL Syntax," *W3Schools*, https://www.w3schools.com/sql/sql_syntax.asp

[2] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 7th ed. Boston, MA, USA: Pearson, 2016, pp. 235-240.