Rajshahi University of Engineering & Technology

Course Title

Data Base Systems Sessional

Course No: ECE 2216

Lab Report: 02

Date of submission: October 02, 2024

Submitted by:	**Submitted to:**
Md Sajjad Hosen Siam Roll: 2110013 Reg. No.:1067/2021-2022 Department of ECE	Oishi Jyoti Assistant Professor Department of ECE

Experiment No: 02

Experiment Name: Database query using mysql

Objectives:

The objective of these errands is to analyze a dataset containing understudy records to extricate significant experiences and unravel particular inquiries. The assignments include distinguishing understudies based on different criteria such as age, GPA, enrollment status, and expenses paid. This center on sifting information to discover understudies with tall scholarly execution, especially those having a place to the Designing division, and recognizing designs in understudy enrollment. Additionally, it incorporates the computation of departmental measurements, such as add up to expenses paid and normal GPA, to supply a clearer understanding of understudy dissemination over diverse disciplines. The point is to apply information questioning procedures to draw conclusions that might bolster scholastic and authoritative decision making.

Lab Task: Create the following table:

```
CREATE DATABASE students_db;
USE students_db;

```

```
CREATE TABLE student (
    student_id INT(100),
    student_name VARCHAR(255),
    age INT(100),
    GPA DOUBLE,
    department VARCHAR(255),
    year_of_admission INT(100),
    fees_paid INT(100),
    credits_earned INT(100),
    enrollment_status VARCHAR(255)
);
```

```
INSERT INTO student (student_id, student_name, age, GPA, department, year_of_admission, fees_paid, credits_earned,
enrollment_status)
VALUES
(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', 2019, 5000, 60, 'active'),
(8, 'Alexei', 25, 3.2, 'Business', 2018, 7500, 100, 'inactive'),
(9, 'Steve', 21, 3.8, 'Science', 2020, 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');
```

St_id	St_name	Age	Department	GPA	Admission_ year	Fees_paid	Earned_ credit	Enroll_ Status
1	Eleven	21	Engineering	3.8	2021	10000	120	Active
2	Dustin	22	Science	3.9	2020	9000	110	Active
3	Will	19	Business	3.4	2022	8500	95	Active
4	Mike	23	Science	3.7	2021	9500	115	Inactive
5	Max	20	Engineering	3.5	2020	12000	130	Active
6	Eddie	22	Arts	4.0	2019	8000	140	Active
7	Billy	24	Engineering	2.9	2022	5000	60	Active
8	Alexei	25	Business	3.2	2018	7500	100	Inactive
9	Steve	21	Science	3.8	2021	10500	120	Active
10	Robin	20	Engineering	3.6	2022	11000	125	Active
11	Lucas	18	Engineering	2.7	2023	4000	50	Active
12	Nancy	23	Business	3.9	2019	9500	135	Active

Table 1: Students information table.

Problems:

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., enroll_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

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

Code:

```
SELECT *
FROM students
WHERE age > 20
AND GPA > (SELECT AVG(GPA) FROM students);

```

Output:

	student_id	student_name	age	GPA	2	department	year_of_admission	fees_paid	1	credits_earned	enrollment_status
Edit Copy Delete	5	Max	20	3.50		Engineering	2020	12000.00		130	active
Edit Copy Delete	10	Robin	20	3.60		Engineering	2022	11000.00		125	active
Edit Copy Delete	9	Steve	21	3.80		Science	2021	10500.00		120	active
Edit Copy Delete	1	Eleven	21	3.80		Engineering	2021	10000.00		120	active
Edit Copy Delete	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 (as a tiebreaker)

Code:

```
SELECT *
FROM students
ORDER BY fees_paid DESC, GPA DESC
LIMIT 5;
```

Output:

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

Task 3: List students from the "Engineering" department with a GPA greater than 3.5 and enrolled after 2020

Code:

```
SELECT *
FROM students
WHERE department = 'Engineering'
AND GPA > 3.5
AND year_of_admission > 2020;
```

Output:

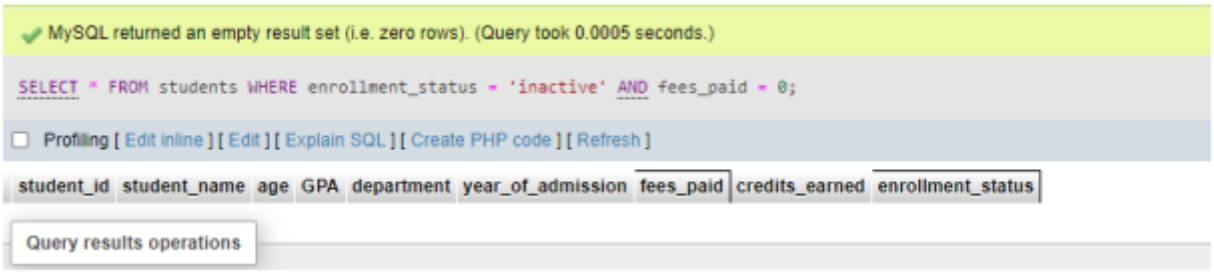
	student_id	student_name	age	GPA	department	year_of_admission	fees_paid	credits_earned	enrollment_status
Edit Copy Delete	1	Eleven	21	3.80	Engineering	2021	10000.00	120	active
Edit Copy Delete	10	Robin	20	3.60	Engineering	2022	11000.00	125	active

Task 4: Find students who are not active and have not paid any fees (fees_paid = 0)

Code:

```
SELECT *
FROM students
WHERE enrollment_status = 'inactive'
AND fees_paid = 0;
```

Output:

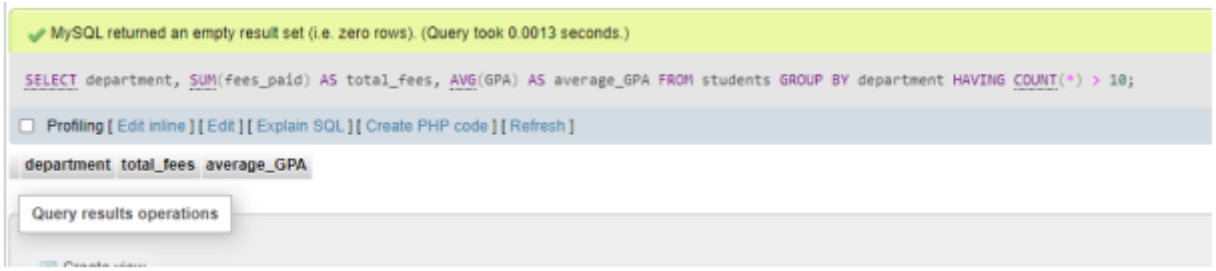


Task 5: Calculate the total fees paid and average GPA for each department with more than 10 students

Code:

```sql
SELECT department, SUM(fees_paid) AS total_fees, AVG(GPA) AS average_GPA
FROM students
GROUP BY department
HAVING COUNT(*) > 10;
```

Output:



Discussion: This experiment looked at basic database tasks using MySQL in XAMPP. We started by creating a database called “student_db” and a table named “students.” We changed a column name from “favorite_subject” to “major” to show how to manage changes in the table.[3] We deleted records for students who scored below 30 marks to keep the data relevant. We also added a new column called “log” and filled it with values based on the semester. Overall, this experiment helped us understand important tasks like creating, changing, and managing data in a simple way, making the database more useful and accurate.

References

[1] C. J. Date, An Introduction to Database Systems, 8th ed. Boston: Addison-Wesley, 2003.

[2] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 7th ed. Boston: Pearson, 2016.

[3] A. Silberschatz, H. F. Korth, and S. Sudarshan, Database System Concepts, 6th ed. New York: McGraw-Hill, 2010.