Heavens Light is Our Guide

Rajshahi University of Engineering & Technology

Course Title

Data Base Systems Sessional

Course No: ECE 2216

Lab Report: 01

Date of submission: 01 October 2024

Submitted by:	**Submitted to:**
A.S.M Al Mosabbir Roll: 2110011 Reg. No.:1065/2021-2022 Department of ECE	Oishi Jyoti Assistant Professor Department of ECE

Experiment No: 02

Experiment Name: Database query using mysql.

Objective:

1. Study various operation of Data Definition Language (DDL)
2. Study various operation of Data Manipulation Language (DML)

Problem Statement:

Information of 12 students are given below:

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

Query:

Table creation and insertion:

```
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)
);
```

```sql
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');
```

Output:

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	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

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

Query:

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

Output:

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
4	Mike	23	3.7	Science	2021	9500	115	inactive
6	Eddie	22	4	Arts	2019	8000	140	active
9	Steve	21	3.8	Science	2020	10500	120	active
12	Nancy	23	3.9	Business	2019	9500	135	active

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

Query:

```sql
SELECT * FROM student 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	3.5	Engineering	2020	12000	130	active
10	Robin	20	3.6	Engineering	2022	11000	125	active
9	Steve	21	3.8	Science	2020	10500	120	active
1	Eleven	21	3.8	Engineering	2021	10000	120	active
12	Nancy	23	3.9	Business	2019	9500	135	active

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

Query:

```
SELECT * FROM student 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	3.8	Engineering	2021	10000	120	active
10	Robin	20	3.6	Engineering	2022	11000	125	active

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

Query:

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

Output:

student_id	student_name	age	GPA	department	year_of_admission	fees_paid	credits_earned	enrollment_status

Task-5. Calculate the total fees paid and average GPA for each department, but only for departments with more than 10 students

Query:

```
SELECT department,
       SUM(fees_paid) AS total_fees_paid,
       AVG(GPA) AS average_GPA,
       COUNT(*) AS student_count
FROM student
GROUP BY department
HAVING COUNT(*) > 10;
```

Output:

department	total_fees_paid	average_GPA	student_count