SQL TASK – 2 – StudentManagement Database

1. Create Table – Courses, Enrollments

SQL Query:

Courses

```
CREATE TABLE Courses(course_id INT AUTO_INCREMENT PRIMARY KEY,
                     course_name VARCHAR(100) NOT NULL,
                     course_description VARCHAR(255));
```

Enrollments

```
CREATE TABLE Enrolments(enrolment_id INT AUTO_INCREMENT PRIMARY KEY,
                        Student_id INT, course_id INT,
                        enrolment_date DATE,
                        FOREIGN KEY(Student_id)REFERENCES Students(StudentId),
                        FOREIGN KEY(course_id)REFERENCES Courses(course_id));
```

Purpose: Creates a new table for storing Course and enrollment data in the existing database.

Observation: Establishes the structure to store detailed Course and enrollment information.

2. Insert Sample Records

SQL Query:

Courses

```
INSERT INTO Courses(course_name, course_description) VALUES
                    ('Computer Science','Covers Maths,Science, English, and basic computing'),
                    ('Physics & Applied Science','Covers physics and chemistry principles and practical'),
                    ('Accountancy','Covers financial recording, involves basic math calculations and
                    English communication'),
```

('Psychology','Covers the scientific study of mind and behavior');

Enrollments

INSERT INTO Enrolments (student_id, course_id, enrolment_date) VALUES

(1, 1, '2025-06-01'),

(2, 2, '2025-06-01'), (2, 3, '2025-06-02'),

(3, 1, '2025-06-03'),

(4, 2, '2025-06-04'), (4, 3, '2025-06-04'),

(5, 1, '2025-06-05'),

(6, 1, '2025-06-06'),

(7, 2, '2025-06-07'), (7, 3, '2025-06-07'),

(8, 1, '2025-06-08'),

(9, 1, '2025-06-09'),

(10, 2, '2025-06-10'), (10, 3, '2025-06-10');

Purpose: Populates the Course and Enrollment table with Sample records for analysis.

Observation: The Courses and Enrolments tables show diverse academic choices. Some students enroll in comprehensive courses, while others opt for specialized subjects, reflecting varied enrollment patterns.

3. Listing all students and the courses they are enrolled in

SQL Query:

SELECT s.Name AS StudentName, c.course_name AS CourseName FROM Enrolments e

INNER JOIN Students s ON e.student_id = s.StudentID

INNER JOIN Courses c ON e.course_id = c.course_id;

Purpose: Display all enrolled students along with the courses they have registered for by combining data from Students, Courses, and Enrolments tables using an INNER JOIN.

Observation: Useful for verifying enrolment data and understanding which students are registered in which courses.

StudentName	CourseName
Alice Johnson	Computer Science
Catherine	Computer Science
Eva Mariam	Computer Science
Fadalu	Computer Science
Harry	Computer Science
Isabella	Computer Science
Bablu	Physics & Applied Science
David	Physics & Applied Science
Grace	Physics & Applied Science
Jis Thomas	Physics & Applied Science
Bablu	Accountancy
David	Accountancy
Grace	Accountancy
Jis Thomas	Accountancy

4. Number of students enrolled in each course

SQL Query:

```
SELECT c.course_id,c.course_name, COUNT(e.student_id) AS NumberOfStudents
FROM Courses c
LEFT JOIN Enrolments e ON c.course_id = e.course_id
GROUP BY c.course_id, c.course_name;
```

Purpose: Shows number students enrolled in each courses.

Observation: Shows how many students are enrolled in each course, helping to analyze course popularity and student distribution.

course_id	course_name	NumberOfStudents
1	Computer Science	6
2	Physics & Applied Science	4
3	Accountancy	4
4	Psychology	0

5. Students who have enrolled in more than one course

SQL Query:

```
SELECT s.StudentID, s.Name, COUNT(e.course_id) AS NumberOfCourses
FROM Enrolments e
INNER JOIN Students s ON e.student_id = s.StudentID
GROUP BY s.StudentID, s.Name
HAVING COUNT(e.course_id) > 1;
```

Purpose: Finds students who are enrolled in multiple courses.

Observation: Helps identify students with broader academic engagement.

StudentID	Name	NumberOfCourses
2	Bablu	2
4	David	2
7	Grace	2
10	Jis Thomas	2

6. Courses with no enrolled students

SQL Query:

```
SELECT c.course_id, c.course_name FROM Courses c
LEFT JOIN Enrolments e ON c.course_id = e.course_id
WHERE e.enrolment_id IS NULL;
```

Purpose: Identifies courses that currently have no student enrolments.

Observation: Useful for recognizing less popular or unselected courses for potential review or promotion.

course_id	course_name
4	Psychology