Worksheet 1.1

Student Name: Ayush Yadav **UID: 25IMC13004**
Branch: MCA (AI &ML) **Section/Group: MAM-1(A)**
Semester: 1st **Date of Performance: 21 Aug 2025**
Subject Name: PL/SQL **Subject Code: 25CAP-602**

1. Aim/Objective:

To gain hands-on experience in writing SQL queries for extracting, analyzing, and modifying data from relational databases by using multiple tables, aggregate functions, joins, and conditions to solve practical academic problems.

2. Question:

Table Students:

StudentID	FirstName	LastName	DepartmentID	DOB	Gender	GPA
INT (PK)	VARCHAR	VARCHAR	INT (FK)	DATE	CHAR	DEC

Table Department:

DepartmentID	DepartmentName
INT (PK)	VARCHAR

Table Courses:

CourseID	CourseName	DepartmentID	Credits
INT (PK)	VARCHAR	INT (FK)	INT

Table Enrollement:

EnrollmentID	StudentID	CourseID	Semester	Year	Grade
INT (PK)	INT (FK)	INT (FK)	VARCHAR	INT	CHAR

Table Faculty:

FacultyID	FirstName	LastName	DepartmentID	HireDate	Salary
INT (PK)	VARCHAR	VARCHAR	INT (FK)	DATE	DEC

- Find the **average GPA** of students in each department.
- Retrieve the **total number of courses** offered by each department.
- Find the **highest and lowest salary** among faculty members in the Computer Science department.
- List departments with **more than 5 courses**.
- Retrieve all students whose **GPA is between 3.0 and 3.5**.
- Find courses whose names contain the word **'Data'**.
- List faculty members hired **before 2015** and earning **more than 80,000**.
- Find students **not enrolled** in any course in the current semester.
- Retrieve courses with credits **not equal to 3 or 4**.
- List each student with the name of the department they belong to.
- Retrieve all courses along with the names of faculty teaching them (assuming a **CourseFaculty** mapping table exists).
- Find students along with the **number of courses they have enrolled in**.
- Retrieve all courses along with the **names of students enrolled** in them.
- Find students who have taken **at least one course outside their own department**.
- Find the department(s) with the **highest average faculty salary**.

3. Brief introduction about the main concept:

In this experiment, the goal is to practice working with data stored in a relational database system. Such databases link tables using primary and foreign keys, making it easier to manage complex information. SQL is applied here to search, modify, and analyze the data. Key topics covered include selection of records, filtering with specific conditions, performing aggregate operations, and joining related tables. The academic context, including courses, students, and faculty, is used as the example for applying these techniques.

4. Implemention/ code:

```
CREATE DATABASE AYUSHCU;
USE AYUSHCU;

CREATE TABLE Departments ( DepartmentID INT PRIMARY KEY, DepartmentName VARCHAR(100) NOT NULL);
INSERT INTO Departments (DepartmentID, DepartmentName) VALUES
                        -> (1, 'Computer Science'),
                        -> (2, 'Electrical Engineering'),
                        -> (3, 'Mechanical Engineering'),
                        -> (4, 'Mathematics');

CREATE TABLE Students (StudentID INT PRIMARY KEY, FirstName VARCHAR(50), LastName VARCHAR(50),
DepartmentID INT,  DOB DATE, Gender CHAR(1), GPA DECIMAL(3,2), FOREIGN KEY (DepartmentID)
REFERENCES Departments(DepartmentID);

INSERT INTO Students (StudentID, FirstName, LastName, DepartmentID, DOB, Gender, GPA) VALUES
                        -> (101, 'Ayush', 'Yadav', 1, '2002-05-15', 'M', 8.75),
                        -> (102, 'Priya', 'Sharma', 2, '2001-09-21', 'F', 9.10),
                        -> (103, 'Rohan', 'Verma', 3, '2000-12-11', 'M', 7.80),
                        -> (104, 'Simran', 'Kaur', 4, '2002-03-02', 'F', 8.20);

CREATE TABLE Faculty (FacultyID INT PRIMARY KEY, FirstName VARCHAR(50), LastName VARCHAR(50),
DepartmentID INT, HireDate DATE, Salary DECIMAL(10,2), FOREIGN KEY (DepartmentID) REFERENCES
Departments(DepartmentID));
INSERT INTO Faculty (FacultyID, FirstName, LastName, DepartmentID, HireDate, Salary) VALUES
                        -> (401, 'Arjun', 'Mehta', 1, '2015-08-01', 75000.00),
                        -> (402, 'Neha', 'Kapoor', 2, '2018-07-15', 68000.00),
                        -> (403, 'Rajesh', 'Singh', 3, '2012-06-20', 80000.00),
                        -> (404, 'Pooja', 'Agarwal', 4, '2020-01-10', 62000.00);

CREATE TABLE Courses (CourseID INT PRIMARY KEY, CourseName VARCHAR(100) NOT NULL, DepartmentID
INT, Credits INT, FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID));
INSERT INTO Courses (CourseID, CourseName, DepartmentID, Credits) VALUES
                        -> (201, 'Data Structures', 1, 4),
                        -> (202, 'Circuit Analysis', 2, 3),
                        -> (203, 'Thermodynamics', 3, 3),
                        -> (204, 'Linear Algebra', 4, 4);
```

```
CREATE TABLE Enrollments (EnrollmentID INT PRIMARY KEY, StudentID INT, CourseID INT, Semester
VARCHAR(20), Year INT, Grade CHAR(2), FOREIGN KEY (StudentID) REFERENCES Students(StudentID), FOREIGN
KEY (CourseID) REFERENCES Courses(CourseID));
INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Semester, Year, Grade) VALUES
                        -> (301, 101, 201, 'Fall', 2024, 'A'),
                        -> (302, 102, 202, 'Spring', 2024, 'B+'),
                        -> (303, 103, 203, 'Fall', 2023, 'A-'),
                        -> (304, 104, 204, 'Spring', 2023, 'B');
```

-- Find the average GPA of students in each department.

```
SELECT
   d.DepartmentName,
   AVG(s.GPA) AS AverageGPA
FROM
   Students s
JOIN
   Departments d ON s.DepartmentID = d.DepartmentID
GROUP BY
   d.DepartmentName;
```

-- Retrieve the total number of courses offered by each department.

```
SELECT
   d.DepartmentName,
   COUNT(c.CourseID) AS TotalCourses
FROM
   Departments d
LEFT JOIN
   Courses c ON d.DepartmentID = c.DepartmentID
GROUP BY
   d.DepartmentName;
```

-- Find the highest and lowest salary among faculty members in the Computer Science department.

```
SELECT
   d.DepartmentName,
   MAX(f.Salary) AS HighestSalary,
   MIN(f.Salary) AS LowestSalary
```

```
FROM
   Faculty f
JOIN
   Departments d ON f.DepartmentID = d.DepartmentID
WHERE
   d.DepartmentName = 'Computer Science'
GROUP BY
   d.DepartmentName;
```

-- List departments with more than 5 courses.

```
SELECT d.DepartmentName, COUNT(c.CourseID) AS CourseCount
FROM courses c
JOIN departments d ON c.DepartmentID = d.DepartmentID
GROUP BY d.DepartmentName
HAVING COUNT(c.CourseID) > 5;
```

-- Retrieve all students whose GPA is between 3.0 and 3.5.

```
SELECT * FROM students WHERE GPA BETWEEN 3.0 AND 3.5;
```

-- Find courses whose names contain the word ‘Data’.

```
SELECT * FROM courses WHERE CourseName LIKE '%Data%';
```

-- List faculty members hired before 2015 and earning more than 80,000.

```
SELECT * FROM faculty WHERE HireDate < '2015-01-01' AND Salary > 80000;
```

-- Find students not enrolled in any course in the current semester.

```
SELECT
   s.StudentID,
   s.FirstName,
   s.LastName,
   s.GPA
FROM
   Students s
LEFT JOIN Enrollments e
   ON s.StudentID = e.StudentID
   AND e.Semester = 'Fall'
   AND e.Year = 2024
WHERE
   e.EnrollmentID IS NULL;
```

-- Retrieve courses with credits not equal to 3 or 4.
SELECT * FROM courses WHERE Credits NOT IN (3, 4);

-- List each student with the name of the department they belong to.
SELECT s.StudentID, s.FirstName, s.LastName, d.DepartmentName
FROM students s
JOIN departments d ON s.DepartmentID = d.DepartmentID;

-- Retrieve all courses along with the names of faculty teaching them (assuming a CourseFaculty mapping table exists).
SELECT c.CourseName, f.FirstName, f.LastName
FROM courses c
JOIN CourseFaculty cf ON c.CourseID = cf.CourseID
JOIN faculty f ON cf.FacultyID = f.FacultyID;

-- Find students along with the number of courses they have enrolled in.
SELECT s.StudentID, s.FirstName, s.LastName, COUNT(e.CourseID) AS TotalCourses
FROM students s
LEFT JOIN enrollments e ON s.StudentID = e.StudentID
GROUP BY s.StudentID, s.FirstName, s.LastName;

-- Retrieve all courses along with the names of students enrolled in them.
SELECT c.CourseName, s.FirstName, s.LastName
FROM courses c
JOIN enrollments e ON c.CourseID = e.CourseID
JOIN students s ON e.StudentID = s.StudentID;

-- Find students who have taken at least one course outside their own department.
SELECT DISTINCT s.StudentID, s.FirstName, s.LastName
FROM students s
JOIN enrollments e ON s.StudentID = e.StudentID
JOIN courses c ON e.CourseID = c.CourseID
WHERE s.DepartmentID <> c.DepartmentID;

--Find the department(s) with the highest average faculty salary.

```
SELECT d.DepartmentName, AVG(f.Salary) AS AvgSalary
FROM faculty f
JOIN departments d ON f.DepartmentID = d.DepartmentID
GROUP BY d.DepartmentName
HAVING AVG(f.Salary) = (
SELECT MAX(AvgSalary)
FROM (
        SELECT AVG(Salary) AS AvgSalary
        FROM faculty
         GROUP BY DepartmentID
        ) AS dept_avg
);
```

5. OUTPUT:

C:\Program Files\MySQL\MySQL Shell 8.0\bin\mysqlsh.exe

```
Query OK, 0 rows affected (0.0181 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > CREATE DATABASE AYUSHCU;
Query OK, 1 row affected (0.0161 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > USE AYUSHCU;
Default schema set to `AYUSHCU`.
Fetching global names, object names from `ayushcu` for auto-completion... Press ^C to stop.
 MySQL  localhost:33060+ ssl  ayushcu  SQL > CREATE TABLE Departments (
                                        ->     DepartmentID INT PRIMARY KEY,
                                        ->     DepartmentName VARCHAR(100) NOT NULL
                                        -> );
Query OK, 0 rows affected (0.0710 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > INSERT INTO Departments (DepartmentID, DepartmentName) VALUES
                                        -> (1, 'Computer Science'),
                                        -> (2, 'Electrical Engineering'),
                                        -> (3, 'Mechanical Engineering'),
                                        -> (4, 'Mathematics');
Query OK, 4 rows affected (0.0083 sec)

Records: 4  Duplicates: 0  Warnings: 0
 MySQL  localhost:33060+ ssl  ayushcu  SQL > CREATE TABLE Students (
                                       ->     StudentID INT PRIMARY KEY,
                                       ->     FirstName VARCHAR(50),
                                       ->     LastName VARCHAR(50),
                                       ->     DepartmentID INT,
                                       ->     DOB DATE,
                                       ->     Gender CHAR(1),
                                       ->     GPA DECIMAL(3,2),
                                       ->     FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
                                       -> );
Query OK, 0 rows affected (0.2009 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > INSERT INTO Students (StudentID, FirstName, LastName, DepartmentID, DOB, Gender, GPA) VALUES
                                       -> (101, 'Ayush', 'Yadav', 1, '2002-05-15', 'M', 8.75),
                                       -> (102, 'Priya', 'Sharma', 2, '2001-09-21', 'F', 9.10),
                                       -> (103, 'Rohan', 'Verma', 3, '2000-12-11', 'M', 7.80),
                                       -> (104, 'Simran', 'Kaur', 4, '2002-03-02', 'F', 8.20);
Query OK, 4 rows affected (0.0158 sec)
```

```
 MySQL  localhost:33060+ ssl  ayushcu  SQL > INSERT INTO Faculty (FacultyID, FirstName, LastName, DepartmentID, HireDate, Salary) VALUES
                                       -> (401, 'Arjun', 'Mehta', 1, '2015-08-01', 75000.00),
                                       -> (402, 'Neha', 'Kapoor', 2, '2018-07-15', 68000.00),
                                       -> (403, 'Rajesh', 'Singh', 3, '2012-06-20', 80000.00),
                                       -> (404, 'Pooja', 'Agarwal', 4, '2020-01-10', 62000.00);
Query OK, 4 rows affected (0.0147 sec)

Records: 4  Duplicates: 0  Warnings: 0
 MySQL  localhost:33060+ ssl  ayushcu  SQL > CREATE TABLE Courses (
                                       ->     CourseID INT PRIMARY KEY,
                                       ->     CourseName VARCHAR(100) NOT NULL,
                                       ->     DepartmentID INT,
                                       ->     Credits INT,
                                       ->     FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
                                       -> );
Query OK, 0 rows affected (0.0961 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > INSERT INTO Courses (CourseID, CourseName, DepartmentID, Credits) VALUES
                                       -> (201, 'Data Structures', 1, 4),
                                       -> (202, 'Circuit Analysis', 2, 3),
                                       -> (203, 'Thermodynamics', 3, 3),
                                       -> (204, 'Linear Algebra', 4, 4);
Query OK, 4 rows affected (0.0203 sec)

Records: 4  Duplicates: 0  Warnings: 0
 MySQL  localhost:33060+ ssl  ayushcu  SQL > CREATE TABLE Enrollments (
                                       ->     EnrollmentID INT PRIMARY KEY,
                                       ->     StudentID INT,
                                       ->     CourseID INT,
                                       ->     Semester VARCHAR(20),
                                       ->     Year INT,
                                       ->     Grade CHAR(2),
                                       ->     FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
                                       ->     FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
                                       -> );
Query OK, 0 rows affected (0.0926 sec)
```

C:\Program Files\MySQL\MySQL Shell 8.0\bin\mysqlsh.exe

```
                                    -> );
Query OK, 0 rows affected (0.0926 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Semester, Year, Grade) VALUES
                                    -> (301, 101, 201, 'Fall', 2024, 'A'),
                                    -> (302, 102, 202, 'Spring', 2024, 'B+'),
                                    -> (303, 103, 203, 'Fall', 2023, 'A-'),
                                    -> (304, 104, 204, 'Spring', 2023, 'B');
Query OK, 4 rows affected (0.0143 sec)

Records: 4  Duplicates: 0  Warnings: 0
 MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM DEPARTMENTS;
+--------------+------------------------+
| DepartmentID | DepartmentName         |
+--------------+------------------------+
|            1 | Computer Science       |
|            2 | Electrical Engineering |
|            3 | Mechanical Engineering |
|            4 | Mathematics            |
+--------------+------------------------+
4 rows in set (0.0018 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM STUDENTS;
+-----------+-----------+----------+--------------+------------+--------+------+
| StudentID | FirstName | LastName | DepartmentID | DOB        | Gender | GPA  |
+-----------+-----------+----------+--------------+------------+--------+------+
|       101 | Ayush     | Yadav    |            1 | 2002-05-15 | M      | 8.75 |
|       102 | Priya     | Sharma   |            2 | 2001-09-21 | F      | 9.10 |
|       103 | Rohan     | Verma    |            3 | 2000-12-11 | M      | 7.80 |
|       104 | Simran    | Kaur     |            4 | 2002-03-02 | F      | 8.20 |
+-----------+-----------+----------+--------------+------------+--------+------+
4 rows in set (0.0024 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM FACULTY;
+-----------+-----------+----------+--------------+------------+----------+
| FacultyID | FirstName | LastName | DepartmentID | HireDate   | Salary   |
+-----------+-----------+----------+--------------+------------+----------+
|       401 | Arjun     | Mehta    |            1 | 2015-08-01 | 75000.00 |
|       402 | Neha      | Kapoor   |            2 | 2018-07-15 | 68000.00 |
|       403 | Rajesh    | Singh    |            3 | 2012-06-20 | 80000.00 |
|       404 | Pooja     | Agarwal  |            4 | 2020-01-10 | 62000.00 |
+-----------+-----------+----------+--------------+------------+----------+
```

```
 MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM COURSE;
ERROR: 1146: Table 'ayushcu.course' doesn't exist
 MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM COURSES;
+----------+------------------+--------------+---------+
| CourseID | CourseName       | DepartmentID | Credits |
+----------+------------------+--------------+---------+
|      201 | Data Structures  |            1 |       4 |
|      202 | Circuit Analysis |            2 |       3 |
|      203 | Thermodynamics   |            3 |       3 |
|      204 | Linear Algebra   |            4 |       4 |
+----------+------------------+--------------+---------+
4 rows in set (0.0021 sec)
 MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM ENROLLMENTS;
+--------------+-----------+----------+----------+------+-------+
| EnrollmentID | StudentID | CourseID | Semester | Year | Grade |
+--------------+-----------+----------+----------+------+-------+
|          301 |       101 |      201 | Fall     | 2024 | A     |
|          302 |       102 |      202 | Spring   | 2024 | B+    |
|          303 |       103 |      203 | Fall     | 2023 | A-    |
|          304 |       104 |      204 | Spring   | 2023 | B     |
+--------------+-----------+----------+----------+------+-------+
4 rows in set (0.0007 sec)
```

```
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT
                                        ->     d.DepartmentName,
                                        ->     AVG(s.GPA) AS AverageGPA
                                        -> FROM
                                        ->     Students s
                                        -> JOIN
                                        ->     Departments d ON s.DepartmentID = d.DepartmentID
                                        -> GROUP BY
                                        ->     d.DepartmentName;
+------------------------+------------+
| DepartmentName         | AverageGPA |
+------------------------+------------+
| Computer Science       |   8.750000 |
| Electrical Engineering |   9.100000 |
| Mechanical Engineering |   7.800000 |
| Mathematics            |   8.200000 |
+------------------------+------------+
4 rows in set (0.0019 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT
                                        ->     d.DepartmentName,
                                        ->     COUNT(c.CourseID) AS TotalCourses
                                        -> FROM
                                        ->     Departments d
                                        -> LEFT JOIN
                                        ->     Courses c ON d.DepartmentID = c.DepartmentID
                                        -> GROUP BY
                                        ->     d.DepartmentName;
+------------------------+--------------+
| DepartmentName         | TotalCourses |
+------------------------+--------------+
| Computer Science       |            1 |
| Electrical Engineering |            1 |
| Mechanical Engineering |            1 |
| Mathematics            |            1 |
+------------------------+--------------+
4 rows in set (0.0011 sec)
```

```
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT
                                       ->     d.DepartmentName,
                                       ->     MAX(f.Salary) AS HighestSalary,
                                       ->     MIN(f.Salary) AS LowestSalary
                                       -> FROM
                                       ->     Faculty f
                                       -> JOIN
                                       ->     Departments d ON f.DepartmentID = d.DepartmentID
                                       -> WHERE
                                       ->     d.DepartmentName = 'Computer Science'
                                       -> GROUP BY
                                       ->     d.DepartmentName;
+-----------------+--------------+-------------+
| DepartmentName  | HighestSalary | LowestSalary |
+-----------------+--------------+-------------+
| Computer Science |      75000.00 |     75000.00 |
+-----------------+--------------+-------------+
1 row in set (0.0015 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT
                                       ->     d.DepartmentName,
                                       ->     COUNT(c.CourseID) AS TotalCourses
                                       -> FROM
                                       ->     Departments d
                                       -> JOIN
                                       ->     Courses c ON d.DepartmentID = c.DepartmentID
                                       -> GROUP BY
                                       ->     d.DepartmentName
                                       -> HAVING
                                       ->     COUNT(c.CourseID) > 5;
Empty set (0.0014 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM students WHERE GPA BETWEEN 3.0 AND 3.5;
Empty set (0.0009 sec)
```

```
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM courses WHERE CourseName LIKE '%Data%';
+----------+-----------------+--------------+---------+
| CourseID | CourseName      | DepartmentID | Credits |
+----------+-----------------+--------------+---------+
|      201 | Data Structures |            1 |       4 |
+----------+-----------------+--------------+---------+
1 row in set (0.0010 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT * FROM faculty WHERE HireDate < '2015-01-01' AND Salary > 80000;
Empty set (0.0008 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT
                                        ->     s.StudentID,
                                        ->     s.FirstName,
                                        ->     s.LastName,
                                        ->     s.GPA
                                        -> FROM
                                        ->     Students s
                                        -> LEFT JOIN Enrollments e
                                        ->     ON s.StudentID = e.StudentID
                                        ->     AND e.Semester = 'Fall'
                                        ->     AND e.Year = 2024
                                        -> WHERE
                                        ->     e.EnrollmentID IS NULL;
+-----------+-----------+----------+------+
| StudentID | FirstName | LastName | GPA  |
+-----------+-----------+----------+------+
|       102 | Priya     | Sharma   | 9.10 |
|       103 | Rohan     | Verma    | 7.80 |
|       104 | Simran    | Kaur     | 8.20 |
+-----------+-----------+----------+------+
3 rows in set (0.0018 sec)
```

```
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT c.CourseName, f.FirstName, f.LastName
                                        -> FROM courses c
                                        -> JOIN CourseFaculty cf ON c.CourseID = cf.CourseID
                                        -> JOIN faculty f ON cf.FacultyID = f.FacultyID;
ERROR: 1146: Table 'ayushcu.coursefaculty' doesn't exist
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT s.StudentID, s.FirstName, s.LastName, COUNT(e.CourseID) AS TotalCourses
                                        -> FROM students s
                                        -> LEFT JOIN enrollments e ON s.StudentID = e.StudentID
                                        -> GROUP BY s.StudentID, s.FirstName, s.LastName;
+-----------+-----------+----------+--------------+
| StudentID | FirstName | LastName | TotalCourses |
+-----------+-----------+----------+--------------+
|       101 | Ayush     | Yadav    |            1 |
|       102 | Priya     | Sharma   |            1 |
|       103 | Rohan     | Verma    |            1 |
|       104 | Simran    | Kaur     |            1 |
+-----------+-----------+----------+--------------+
4 rows in set (0.0011 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT c.CourseName, s.FirstName, s.LastName
                                        -> FROM courses c
                                        -> JOIN enrollments e ON c.CourseID = e.CourseID
                                        -> JOIN students s ON e.StudentID = s.StudentID;
+-----------------+-----------+----------+
| CourseName      | FirstName | LastName |
+-----------------+-----------+----------+
| Data Structures | Ayush     | Yadav    |
| Circuit Analysis | Priya     | Sharma   |
| Thermodynamics  | Rohan     | Verma    |
| Linear Algebra  | Simran    | Kaur     |
+-----------------+-----------+----------+
4 rows in set (0.0015 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT DISTINCT s.StudentID, s.FirstName, s.LastName
                                        -> FROM students s
                                        -> JOIN enrollments e ON s.StudentID = e.StudentID
                                        -> JOIN courses c ON e.CourseID = c.CourseID
                                        -> WHERE s.DepartmentID <> c.DepartmentID;
Empty set (0.0015 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL >
```

```
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT c.CourseName, s.FirstName, s.LastName
                                         -> FROM courses c
                                         -> JOIN enrollments e ON c.CourseID = e.CourseID
                                         -> JOIN students s ON e.StudentID = s.StudentID;
+------------------+-----------+----------+
| CourseName       | FirstName | LastName |
+------------------+-----------+----------+
| Data Structures  | Ayush     | Yadav    |
| Circuit Analysis | Priya     | Sharma   |
| Thermodynamics   | Rohan     | Verma    |
| Linear Algebra   | Simran    | Kaur     |
+------------------+-----------+----------+
4 rows in set (0.0015 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT DISTINCT s.StudentID, s.FirstName, s.LastName
                                         -> FROM students s
                                         -> JOIN enrollments e ON s.StudentID = e.StudentID
                                         -> JOIN courses c ON e.CourseID = c.CourseID
                                         -> WHERE s.DepartmentID <> c.DepartmentID;
Empty set (0.0015 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL > SELECT d.DepartmentName, AVG(f.Salary) AS AvgSalary
                                         -> FROM faculty f
                                         -> JOIN departments d ON f.DepartmentID = d.DepartmentID
                                         -> GROUP BY d.DepartmentName
                                         -> HAVING AVG(f.Salary) = (
                                         -> SELECT MAX(AvgSalary)
                                         -> FROM (
                                         -> SELECT AVG(Salary) AS AvgSalary
                                         -> FROM faculty
                                         ->  GROUP BY DepartmentID
                                         -> ) AS dept_avg
                                         -> );
+-----------------------+--------------+
| DepartmentName        | AvgSalary    |
+-----------------------+--------------+
| Mechanical Engineering | 80000.000000 |
+-----------------------+--------------+
1 row in set (0.0021 sec)
MySQL  localhost:33060+ ssl  ayushcu  SQL >
```

6. Learning Outcomes:

- Recognize how relational databases are organized and how tables are connected.
- Write SQL queries that use aggregate functions such as COUNT, AVG, MIN, and MAX.
- Retrieve records by applying conditions and using pattern-based searches.
- Use different join operations to merge information from several tables.
- Interpret the results of queries to solve academic-style problems.
- Strengthen logical thinking and query-building skills for handling and modifying data.