University Database Management System
1. Student Management: Store student details such as StudentID, Name, Age, Gender, Department, and Email.
2. Course Management: Maintain course details including CourseID, CourseName, Credits, and
Department.
3. Enrollment System: Allow students to enroll in multiple courses, tracking StudentID,
CourseID, EnrollmentDate, and Grade.
4. Professor Management: Store professor details like ProfessorID, Name, Department, and
Email
Write queries for the following questions:
1. Calculate percentage of students in each department
2. Detect duplicate enrollments (same student enrolled in same course in the same semester)
3. Find the semester with the highest average enrollments per course
4. List students with more than 3 enrollments
5. List all courses and the number of students enrolled in each

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100),
    Age INT,
    Gender VARCHAR(10),
    Department VARCHAR(50),
    Email VARCHAR(100)
);

CREATE TABLE Courses (
    CourseID INT PRIMARY KEY AUTO_INCREMENT,
    CourseName VARCHAR(100),
    Credits INT,
    Department VARCHAR(50)
);

CREATE TABLE Professors (
    ProfessorID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100),
    Department VARCHAR(50),
    Email VARCHAR(100)
);

CREATE TABLE Enrollments (
    StudentID INT,
    CourseID INT,
    Semester VARCHAR(20),
```

```
    EnrollmentDate DATE,
    Grade VARCHAR(5),
    PRIMARY KEY (StudentID, CourseID, Semester),
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);

-- Students
INSERT INTO Students (Name, Age, Gender, Department, Email) VALUES
('Aman Singh', 20, 'Male', 'Computer Science', 'aman@uni.edu'),
('Riya Mehta', 22, 'Female', 'Electronics', 'riya@uni.edu'),
('Karan Patel', 21, 'Male', 'Mechanical', 'karan@uni.edu'),
('Priya Sharma', 23, 'Female', 'Computer Science', 'priya@uni.edu');

-- Courses
INSERT INTO Courses (CourseName, Credits, Department) VALUES
('Data Structures', 4, 'Computer Science'),
('Digital Circuits', 3, 'Electronics'),
('Thermodynamics', 4, 'Mechanical'),
('Algorithms', 4, 'Computer Science');

-- Professors
INSERT INTO Professors (Name, Department, Email) VALUES
('Dr. Verma', 'Computer Science', 'verma@uni.edu'),
('Dr. Iyer', 'Electronics', 'iyer@uni.edu'),
('Dr. Naik', 'Mechanical', 'naik@uni.edu');

-- Enrollments
INSERT INTO Enrollments (StudentID, CourseID, Semester, EnrollmentDate, Grade)
VALUES
(1, 1, 'Spring2025', '2025-01-10', 'A'),
(1, 4, 'Spring2025', '2025-01-11', 'B'),
(2, 2, 'Spring2025', '2025-01-12', 'A'),
(3, 3, 'Spring2025', '2025-01-13', 'B+'),
(4, 1, 'Spring2025', '2025-01-14', 'A'),
(4, 4, 'Spring2025', '2025-01-15', 'A');

-- 1. Percentage of Students in Each Department
SELECT Department, ROUND(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM Students), 2) AS
Percentage
FROM Students
GROUP BY Department;

-- 2. Detect Duplicate Enrollments
SELECT StudentID, CourseID, Semester, COUNT(*) AS Count
FROM Enrollments
GROUP BY StudentID, CourseID, Semester
HAVING COUNT(*) > 1;

-- 3. Semester with Highest Average Enrollments per Course
```

```
SELECT Semester, ROUND(COUNT(*) * 1.0 / (SELECT COUNT(*) FROM Courses), 2) AS
AvgEnrollmentsPerCourse
FROM Enrollments
GROUP BY Semester
ORDER BY AvgEnrollmentsPerCourse DESC
LIMIT 1;

-- 4. Students with More Than 3 Enrollments
SELECT s.Name, COUNT(e.CourseID) AS TotalEnrollments
FROM Enrollments e
JOIN Students s ON e.StudentId = s.StudentID
GROUP BY e.StudentID
HAVING COUNT(e.CourseID) > 3;

-- 5. Courses and Number of Students Enrolled
SELECT c.CourseName, COUNT(e.StudentID) AS NumberOfStudents
FROM Courses c
LEFT JOIN Enrollments e ON c.CourseID = e.CourseId
GROUP BY c.CourseID, c.CourseName;
```