```
-- Create tables
CREATE TABLE STUDENTS(
    StudentID INT PRIMARY KEY,
    Name VARCHAR(50),
    Department VARCHAR(10),
    Year INT,
    GPA DECIMAL(3,1)
);

CREATE TABLE COURSES(
    CourseID VARCHAR(10) PRIMARY KEY,
    CourseName VARCHAR(50),
    Department VARCHAR(10),
    Credits INT
);

CREATE TABLE ENROLLMENTS(
    EnrollID INT PRIMARY KEY,
    StudentID INT,
    CourseID VARCHAR(10),
    Semester VARCHAR(10),
    Marks INT,
    FOREIGN KEY (StudentID) REFERENCES STUDENTS(StudentID),
    FOREIGN KEY (CourseID) REFERENCES COURSES(CourseID)
);

CREATE TABLE INSTRUCTORS(
    InstructorID INT PRIMARY KEY,
    Name VARCHAR(50),
    Department VARCHAR(10),
    Experience INT
);

CREATE TABLE COURSE_ASSIGNMENT(
    CourseID VARCHAR(10),
    InstructorID INT,
    Year INT,
    PRIMARY KEY (CourseID, InstructorID),
    FOREIGN KEY (CourseID) REFERENCES COURSES(CourseID),
    FOREIGN KEY (InstructorID) REFERENCES INSTRUCTORS(InstructorID)
);

-- Insert sample data
INSERT INTO STUDENTS VALUES
```

```
(101, 'Aarav Sharma', 'CSE', 3, 8.5),
(102, 'Riya Verma', 'CSE', 2, 9.1),
(103, 'Kabir Singh', 'ECE', 4, 7.8),
(104, 'Meera Nair', 'EEE', 3, 8.0),
(105, 'Ananya Gupta', 'CSE', 4, 8.9);

INSERT INTO COURSES VALUES
('C101', 'DBMS', 'CSE', 4),
('C102', 'Operating Systems', 'CSE', 3),
('C103', 'Digital Electronics', 'ECE', 3),
('C104', 'Power Systems', 'EEE', 3),
('C105', 'AI and ML', 'CSE', 4);

INSERT INTO ENROLLMENTS VALUES
(1, 101, 'C101', 'Sem5', 87),
(2, 101, 'C102', 'Sem5', 78),
(3, 102, 'C101', 'Sem3', 91),
(4, 103, 'C103', 'Sem7', 67),
(5, 105, 'C105', 'Sem8', 94),
(6, 105, 'C101', 'Sem8', 89),
(7, 102, 'C105', 'Sem4', 92);

INSERT INTO INSTRUCTORS VALUES
(1, 'Dr. R.K. Rao', 'CSE', 15),
(2, 'Dr. S. Patel', 'ECE', 10),
(3, 'Dr. Neha Joshi', 'EEE', 8),
(4, 'Prof. A. Mehta', 'CSE', 12);

INSERT INTO COURSE_ASSIGNMENT VALUES
('C101', 1, 2024),
('C102', 1, 2024),
('C103', 2, 2024),
('C104', 3, 2024),
('C105', 4, 2024);


CREATE VIEW CSE_COURSE_DETAILS AS
SELECT
    c.CourseID,
    c.CourseName,
    c.Credits,
    i.Name AS InstructorName,
    i.Experience,
    COUNT(e.StudentID) AS StudentsEnrolled
```

```
FROM COURSES c
LEFT JOIN COURSE_ASSIGNMENT ca ON c.CourseID = ca.CourseID
LEFT JOIN INSTRUCTORS i ON ca.InstructorID = i.InstructorID
LEFT JOIN ENROLLMENTS e ON c.CourseID = e.CourseID
WHERE c.Department = 'CSE'
GROUP BY c.CourseID, c.CourseName, c.Credits, i.Name, i.Experience;


CREATE VIEW STUDENT_COURSE_PERFORMANCE AS
SELECT
   s.StudentID,
   s.Name AS StudentName,
   s.Department AS StudentDepartment,
   c.CourseID,
   c.CourseName,
   e.Marks,
   e.Semester
FROM STUDENTS s
JOIN ENROLLMENTS e ON s.StudentID = e.StudentID
JOIN COURSES c ON e.CourseID = c.CourseID;



CREATE VIEW TOP_PERFORMERS_VIEW AS
SELECT
   s.StudentID,
   s.Name AS StudentName,
   s.Department,
   c.CourseID,
   c.CourseName,
   e.Marks
FROM STUDENTS s
JOIN ENROLLMENTS e ON s.StudentID = e.StudentID
JOIN COURSES c ON e.CourseID = c.CourseID
WHERE e.Marks > (
   SELECT AVG(e2.Marks)
   FROM ENROLLMENTS e2
   WHERE e2.CourseID = e.CourseID
);


CREATE VIEW DEPARTMENT_AVG_VIEW AS
SELECT
   s.Department,
```

```
    AVG(s.GPA) AS AvgGPA,
    AVG(e.Marks) AS AvgMarks
FROM STUDENTS s
LEFT JOIN ENROLLMENTS e ON s.StudentID = e.StudentID
GROUP BY s.Department;


-- First drop the existing view
DROP VIEW IF EXISTS CSE_COURSE_DETAILS;

-- Recreate with average marks
CREATE VIEW CSE_COURSE_DETAILS AS
SELECT
    c.CourseID,
    c.CourseName,
    c.Credits,
    i.Name AS InstructorName,
    i.Experience,
    COUNT(e.StudentID) AS StudentsEnrolled,
    AVG(e.Marks) AS AvgMarks
FROM COURSES c
LEFT JOIN COURSE_ASSIGNMENT ca ON c.CourseID = ca.CourseID
LEFT JOIN INSTRUCTORS i ON ca.InstructorID = i.InstructorID
LEFT JOIN ENROLLMENTS e ON c.CourseID = e.CourseID
WHERE c.Department = 'CSE'
GROUP BY c.CourseID, c.CourseName, c.Credits, i.Name, i.Experience;


WITH DepartmentStudentAvg AS (
    SELECT
        StudentID,
        StudentName,
        StudentDepartment,
        AVG(Marks) AS AvgMarks
    FROM STUDENT_COURSE_PERFORMANCE
    GROUP BY StudentID, StudentName, StudentDepartment
),
RankedStudents AS (
    SELECT
        *,
        ROW_NUMBER() OVER (PARTITION BY StudentDepartment ORDER BY AvgMarks
DESC) as DeptRank
    FROM DepartmentStudentAvg
)
```

```
SELECT
    StudentID,
    StudentName,
    StudentDepartment,
    AvgMarks
FROM RankedStudents
WHERE DeptRank <= 3
ORDER BY StudentDepartment, DeptRank;


-- Check all views
SELECT * FROM CSE_COURSE_DETAILS;
SELECT * FROM STUDENT_COURSE_PERFORMANCE;
SELECT * FROM TOP_PERFORMERS_VIEW;
SELECT * FROM DEPARTMENT_AVG_VIEW;
```