Task 2: Advanced Queries with Joins and Filtering

1. Database Creation

Query:

```
CREATE DATABASE StudentManagement;
USE StudentManagement;
CREATE TABLE Students (
    StudentID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(50),
    Gender VARCHAR(1),
    Age INT,
    Grade VARCHAR(10),
    MathScore INT,
    ScienceScore INT,
    EnglishScore INT
);
CREATE TABLE Courses (
    CourseID INT AUTO_INCREMENT PRIMARY KEY,
    CourseName VARCHAR(100),
    CourseDescription TEXT
);
CREATE TABLE Enrolments (
    EnrolmentID INT AUTO_INCREMENT PRIMARY KEY,
    StudentID INT,
    CourseID INT,
    EnrolmentDate DATE,
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);
```

Purpose:

This query creates the StudentManagement database with three tables: Students, Courses, and Enrolments. The Enrolments table establishes relationships between students and their courses.

2. Data Insertion

Query:

```
26 •  INSERT INTO Students (Name, Gender, Age, Grade, MathScore, ScienceScore, EnglishScore)
27    VALUES
28    ('RIYA', 'F', 16, 'A', 95, 90, 88),
29    ('RAM', 'M', 17, 'B', 80, 85, 82),
30    ('Charlie', 'M', 15, 'A', 92, 87, 91);
31
32 •  INSERT INTO Courses (CourseName, CourseDescription)
33    VALUES
34    ('Mathematics', 'Advanced math course'),
35    ('Science', 'Fundamentals of science'),
36    ('English', 'Comprehensive English course');
37
38 •  INSERT INTO Enrolments (StudentID, CourseID, EnrolmentDate)
39    VALUES
40    (1, 1, '2023-01-10'),
41    (1, 2, '2023-02-15'),
42    (2, 1, '2023-03-20'),
43    (3, 3, '2023-04-05');
```

Purpose:

Inserts sample records into Students, Courses, and Enrolments tables to simulate a real-world dataset.

3. Data Analysis Tasks

3.1 LIST ALL STUDENTS AND THEIR COURSES

Query:

```
44    -- QUERY NO 1
45    -- Task 1: List all students and the courses they are enrolled in.
46 •  SELECT S.NAME,C.CourseName FROM STUDENTS S
47    INNER JOIN Enrolments E ON E.StudentID=S.StudentID
48    INNER JOIN Courses C ON C.CourseID=E.CourseID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

NAME	CourseName
RIYA	Mathematics
RIYA	Science
RAM	Mathematics
Charlie	English

Purpose: Lists all students along with the courses they are enrolled in.

3.2 NUMBER OF STUDENTS ENROLLED IN EACH COURSE

```
49    -- QUERY NO 2
50    -- Task 2: Find the number of students enrolled in each course.
51 •  SELECT c.CourseName, COUNT(e.StudentID) AS EnrolledStudents
52    FROM Courses c
53    LEFT JOIN Enrolments e ON c.CourseID = e.CourseID
54    GROUP BY c.CourseID, c.CourseName;
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

CourseName	EnrolledStudents
Mathematics	2
Science	1
English	1

Purpose: Shows the number of students enrolled in each course, including courses with zero enrollments.

3.3 STUDENTS ENROLLED IN MORE THAN ONE COURSE

Query:

```
5     -- QUERY NO 3
5     -- Students Enrolled in More Than One Course.
7 •   SELECT s.Name, COUNT(e.CourseID) AS CourseCount
3     FROM Students s
9     INNER JOIN Enrolments e ON s.StudentID = e.StudentID
9     GROUP BY s.Name
L     HAVING COUNT(e.CourseID) > 1;
```

ult Grid | Filter Rows: | Export: | Wrap Cell Content:

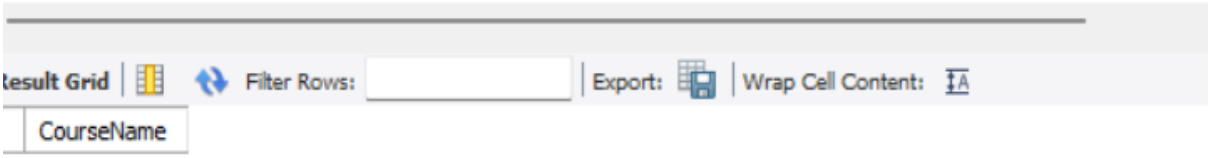
Name	CourseCount
RIYA	2

Purpose: Identifies students who are enrolled in more than one course

3.4 COURSES WITH NO ENROLLED STUDENTS

Query:

```
62    -- QUERY NO 4
63    --  Courses with No Enrolled Students.
64    SELECT c.CourseName
65    FROM Courses c
66    LEFT JOIN Enrolments e ON c.CourseID = e.CourseID
67    WHERE e.EnrolmentID IS NULL;
```



Purpose: Lists courses that have no students enrolled.