StudentID	StudentName	Courset1	Course2	Course3
101	Alice	Math	History	Physics
102	Bob	Physics	NULL	NULL
103	Carol	Chemistry	Biology	NULL

The original "StudentCourses" Table

Task 1: Identify Normalization Issues

The original "StudentCourses" table violates several normalization principles, particularly the following:

1. Repeating Groups/Multivalued Columns: Columns like Course1, Course2, and Course3 violate First Normal Form (1NF) because they represent multiple courses in one table instead of having separate records for each course.

2. NULL Values: Since some students have enrolled in fewer courses, there are NULL values present in the course columns, which results in wasted storage and inefficient queries.

3. Data Redundancy: If multiple courses are added to each student, you would need to add more columns (e.g., Course4, Course5), leading to poor scalability and repetitive course data, violating Second Normal Form (2NF) and Third Normal Form (3NF).

Task 2: Normalize the Table

To normalize the table, we can break it into multiple tables to ensure that it follows the rules of 1NF, 2NF, and 3NF.

Step 1: Create a Students Table

StudentID	StudentName
101	Alice
102	Bob
103	Carol

Step 2: Create a Courses Table

CourseID	CourseName
1	Math
2	History
3	Physics
4	Chemistry
5	Biology

Step 3: Create a StudentCourses Table

This table will create a relationship between students and the courses they are enrolled in. It resolves the many-to-many relationship between students and courses.

StudentID	CourseID	CourseName
101	1	Math
101	2	History
101	3	Physics
102	3	Physics
103	4	Chemistry
103	5	Biology

Explanation:

1NF is achieved by removing multivalued attributes (repeating groups of courses) and representing each CourseID in a separate row.

2NF is achieved by eliminating partial dependencies. Each student-course relationship is represented uniquely, and course information is stored in a separate Courses table.

3NF is achieved because there are no transitive dependencies; all non-key attributes in each table are fully dependent on the primary key.