

Assignment 1: SQL-Based Conversion from 3NF to 2NF

Objective: Identify and eliminate partial dependencies in a SQL schema that is in 3NF and convert it to 2NF.

Scenario: Consider the following table that is in 3NF:

```
CREATE TABLE Employee_Courses (  
    EmpID INT,  
    CourseID INT,  
    EmpName VARCHAR(50),  
    CourseName VARCHAR(50),  
    InstructorName VARCHAR(50),  
    PRIMARY KEY (EmpID, CourseID)  
);
```

Where:

- EmpID is the Employee ID.
- CourseID is the Course ID.
- EmpName is the name of the employee.
- CourseName is the name of the course.
- InstructorName is the name of the course instructor.

Instructions:

1. Identify the functional dependencies and determine the partial dependencies.
2. Explain why this table is in 3NF but violates 2NF.
3. Using SQL, decompose this table into smaller tables that eliminate the partial dependency and make the schema comply with 2NF.
4. Provide the SQL commands to create the decomposed tables.
5. Illustrate the SQL commands needed to insert data into these decomposed tables.
6. Provide the SQL commands to query data from these new tables.

Assignment 2: Converting a Multi-Table 3NF Schema to 2NF Using SQL

Objective: Decompose a multi-table schema in 3NF into a schema that adheres to 2NF.

Scenario: Consider the following set of tables in 3NF:

```
sql  
Copy code  
CREATE TABLE Orders (  
    OrderID INT PRIMARY KEY,  
    CustomerID INT,  
    OrderDate DATE  
);  
  
CREATE TABLE Products (  
    ProductID INT PRIMARY KEY,  
    ProductName VARCHAR(50),
```

```

        Price DECIMAL(10, 2)
    );

CREATE TABLE OrderDetails (
    OrderID INT,
    ProductID INT,
    Quantity INT,
    PRIMARY KEY (OrderID, ProductID),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

```

Instructions:

1. Identify the functional dependencies and explain why this schema is in 3NF.
2. Determine if there are any partial dependencies in the `OrderDetails` table.
3. Convert the `OrderDetails` table to a 2NF-compliant schema by eliminating partial dependencies.
4. Provide the SQL commands to modify the schema and create the new tables after decomposition.
5. Provide sample SQL INSERT statements for the new tables.
6. Provide SQL SELECT queries to retrieve order details, showing how the new structure is beneficial.

Assignment 3: SQL Query-Based Approach to Detect and Remove Partial Dependencies

Objective: Write SQL queries that help identify and resolve partial dependencies in a given schema.

Scenario: Given a table in 3NF that contains partial dependencies:

```

sql
Copy code
CREATE TABLE Student_Courses (
    StudentID INT,
    CourseID INT,
    InstructorName VARCHAR(50),
    InstructorEmail VARCHAR(50),
    StudentName VARCHAR(50),
    PRIMARY KEY (StudentID, CourseID)
);

```

Instructions:

1. Write a SQL query to detect partial dependencies in the `Student_Courses` table. Explain the concept of partial dependency and identify where it occurs.
2. Decompose the table into two or more smaller tables to remove partial dependencies.
3. Write SQL commands to:
 - o Create the new tables.
 - o Insert data into the decomposed tables.

4. Demonstrate how the new structure eliminates partial dependencies and results in a more efficient schema by querying the data.

Assignment 4: Hands-On SQL Decomposition to 2NF

Objective: Apply normalization techniques by decomposing a schema in 3NF into 2NF using SQL.

Scenario: Consider the following table in 3NF:

```
sql
Copy code
CREATE TABLE Library (
    BookID INT,
    ISBN VARCHAR(20),
    Title VARCHAR(100),
    AuthorName VARCHAR(50),
    PublisherName VARCHAR(50),
    Price DECIMAL(10, 2),
    PRIMARY KEY (BookID)
);
```

Where:

- BookID is the primary key.
- ISBN is the unique identifier for books.
- Title is the name of the book.
- AuthorName is the name of the author.
- PublisherName is the name of the publisher.
- Price is the price of the book.

Instructions:

1. Identify the functional dependencies and explain the partial dependencies that exist in this table.
2. Decompose the table into smaller tables in SQL that remove partial dependencies and make the schema 2NF-compliant.
3. Provide the SQL code to create these new tables.
4. Write SQL INSERT statements to populate the new tables.
5. Show how to query the normalized schema for data about books, authors, and publishers.

Assignment 5: Decomposing Complex SQL Schema to 2NF

Objective: Convert a complex schema in 3NF to 2NF while using SQL.

Scenario: Consider the following set of tables in 3NF:

```

sql
Copy code
CREATE TABLE Employee (
    EmpID INT PRIMARY KEY,
    EmpName VARCHAR(50),
    EmpRole VARCHAR(50)
);

CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50)
);

CREATE TABLE Assignment (
    EmpID INT,
    DeptID INT,
    AssignmentDate DATE,
    PRIMARY KEY (EmpID, DeptID),
    FOREIGN KEY (EmpID) REFERENCES Employee(EmpID),
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

```

Instructions:

1. Identify any partial dependencies present in the schema.
2. Decompose the tables in SQL to remove partial dependencies and move the schema to 2NF.
3. Provide the SQL code to create new tables after decomposition.
4. Provide SQL INSERT statements to insert data into the new tables.
5. Demonstrate how the decomposition results in improved schema design by running SQL SELECT queries to retrieve data.

Assignment 6: Advanced SQL Normalization: Identifying and Correcting Partial Dependencies

Objective: Demonstrate the ability to analyze, identify, and correct partial dependencies in a complex schema using SQL.

Scenario: You are given the following database schema in 3NF:

```

sql
Copy code
CREATE TABLE Sales (
    SaleID INT PRIMARY KEY,
    ProductID INT,
    SaleDate DATE,
    SalePrice DECIMAL(10, 2),
    CustomerName VARCHAR(50),
    CustomerEmail VARCHAR(50),
    PRIMARY KEY (SaleID)
);

CREATE TABLE Products (
    ProductID INT PRIMARY KEY,

```

```
    ProductName VARCHAR(50),  
    Category VARCHAR(50)  
);
```

Instructions:

1. Identify the functional dependencies in the given schema.
2. Check for partial dependencies and explain why they violate 2NF.
3. Write the SQL code to decompose the `Sales` table to eliminate partial dependencies, ensuring it follows 2NF.
4. Show the updated SQL schema and provide examples of INSERT and SELECT statements to demonstrate how the new schema works.