### **Assignment: Bills Management System**

#### **Introduction**

As part of your learning journey into databases and data models, you will collaboratively design and implement a database for a **Bills Management System**. This system will help users track and manage their bills, including due dates, payment history, and bill categories. The assignment focuses on conceptualizing, designing, and applying relational data modeling principles.

### **Learning Objectives**

1. Create an Entity-Relationship Diagram for a database.
2. Define database schemas using SQL.
3. Populate tables with sample data using SQL INSERT statements.
4. Write SQL queries to retrieve and analyze data effectively.
5. Demonstrate the working database with practical queries.

### **What You'll Need**

* Access to a computer with MySQL Workbench or SQLite.
* A code editor (e.g., Visual Studio Code).
* Familiarity with SQL commands, data modeling, and relational database principles.

### **Scenario**

You are tasked with creating a **Bills Management System** database. You will design its structure, populate it with sample data, and write queries to interact with the system. Your output will showcase your understanding of database concepts and SQL.

### 

### **Submission Instructions**

1. **Clone or Create Project Files:**
   * Use a tool like MySQL Workbench or SQLite to create your database.
2. **Prepare an SQL File:**
   * Name the file bills\_management.sql.
   * Include SQL statements for all tasks, formatted with comments.
3. **Sample Comments Format:**

****

1. **Push the File to GitHub:**
   * Ensure all queries are functional before submission.

### **Assignment Tasks**

#### **Part 1: Creating an Entity-Relationship Diagram**

**Task 1.1:** Design an ER diagram for a Bills Management System. Include key entities:

* **Users** (Attributes: UserID, Name, Email, Phone).
* **Bills** (Attributes: BillID, UserID, Amount, DueDate, Category, Status).
* **Payments** (Attributes: PaymentID, BillID, PaymentDate, PaymentAmount).

Clearly label primary and foreign keys and relationships between entities.

#### 

#### **Part 2: Defining the Database Schema**

**Task 2.1:** Write SQL statements to define the database schema, including:

* Users table.
* Bills table with foreign key reference to Users.
* Payments table with foreign key reference to Bills.

#### **Part 3: Populating Tables with Sample Data**

**Task 3.1:** Populate the Users table with at least five records. Include:

* UserID, Name, Email, and Phone.

**Task 3.2:** Populate the Bills table with at least ten records. Include:

* BillID, UserID, Amount, DueDate, Category (e.g., Utilities, Subscriptions), and Status (e.g., Paid, Unpaid).

**Task 3.3:** Populate the Payments table with at least five records. Include:

* PaymentID, BillID, PaymentDate, and PaymentAmount. Ensure valid BillID references.

#### **Part 4: Writing SQL Queries**

**Task 4.1:** Retrieve all unpaid bills from the Bills table. Display columns:

* BillID, Amount, DueDate, and Status.

**Task 4.2:** Group bills by category in the Bills table and calculate the total amount for each category. Display:

* Category, TotalAmount.

**Task 4.3:** Calculate the total unpaid amount due for the current month. Display:

* TotalDue.

**Task 4.4:** Retrieve the payment history for a user with UserID = 1. Display:

* PaymentID, BillID, PaymentDate, and PaymentAmount.

#### **Part 5: Demonstrating the Database**

* Create a demo of your database in MySQL Workbench or SQLite.

### **Bonus Challenge: Count the Number of Unpaid Bills per Category**

**Objective**: Write a simple query to count the number of unpaid bills for each category.

**Instructions**:

1. **Write a Query**:  
   Write an SQL query to count the number of unpaid bills for each category.  
   Display the following columns:
   * Category
   * UnpaidBillsCount (number of unpaid bills per category)