**Case Study: Loan Payment Tracking System**

**Problem Statement:**

Design and implement a Loan Payment Tracking System for a bank using Oracle SQL and PL/SQL. The system will be used to manage loan applications, track payments, and handle customer information. Your task is to create the necessary database schema, populate the database with sample data, and develop PL/SQL procedures to handle loan application processing, payment tracking, and generating loan statements.

**Requirements:**

1. **Customer Management**:
   * Implement the functionality to add, update, delete, and search for customers.
   * Ensure that each customer has attributes such as CUSTOMER\_ID, FIRST\_NAME, LAST\_NAME, DOB, EMAIL, PHONE\_NUMBER, and ADDRESS.
2. **Loan Application Processing**:
   * Implement the functionality to manage loan applications.
   * Ensure that each loan application has attributes such as LOAN\_ID, CUSTOMER\_ID, LOAN\_AMOUNT, INTEREST\_RATE, LOAN\_TERM, APPLICATION\_DATE, and STATUS.
3. **Payment Tracking**:
   * Implement the functionality to track loan payments.
   * Ensure that each payment has attributes such as PAYMENT\_ID, LOAN\_ID, PAYMENT\_DATE, AMOUNT\_PAID, and PAYMENT\_METHOD.

**Tasks:**

1. **Design the Database Schema**:
   * Create the Customers, Loans, and Payments tables with the appropriate fields and constraints.
   * Define primary keys and foreign keys to maintain data integrity.
2. **Populate the Database with Sample Data**:
   * Insert sample records into the Customers, Loans, and Payments tables to facilitate testing of the system.
3. **Develop PL/SQL Procedures**:
   * Create a procedure to handle loan application processing. The procedure should insert, update, and delete loan application records.
   * Create a procedure to handle loan payments. The procedure should insert new payment records and update loan balances accordingly.
   * Create a procedure to generate loan statements, including details such as payment history, remaining balance, and due dates.

**Expected Outcomes:**

1. **Customers Table**:
   * Contains all information about the bank's customers.
2. **Loans Table**:
   * Stores loan application details and statuses.
3. **Payments Table**:
   * Tracks all loan payments made by customers.
4. **PL/SQL Procedures**:
   * Efficiently manage loan applications, payments, and loan statements, maintaining accurate records in the database.

**Deliverables:**

1. SQL scripts to create the Customers, Loans, and Payments tables.
2. SQL scripts to insert sample data into the tables.
3. PL/SQL scripts for the procedures to handle loan application processing, payment tracking, and generating loan statements.
4. Documentation explaining how to set up and use the system, including how to run the PL/SQL procedures.

**Database Schema:**

1. **Customers Table**:
   * **CUSTOMER\_ID**: Number, Primary Key
   * **FIRST\_NAME**: Varchar2(50)
   * **LAST\_NAME**: Varchar2(50)
   * **DOB**: Date
   * **EMAIL**: Varchar2(100)
   * **PHONE\_NUMBER**: Varchar2(15)
   * **ADDRESS**: Varchar2(255)
2. **Loans Table**:
   * **LOAN\_ID**: Number, Primary Key
   * **CUSTOMER\_ID**: Number, Foreign Key References Customers(CUSTOMER\_ID)
   * **LOAN\_AMOUNT**: Number
   * **INTEREST\_RATE**: Number
   * **LOAN\_TERM**: Number -- in months
   * **APPLICATION\_DATE**: Date
   * **STATUS**: Varchar2(20) -- ('Pending', 'Approved', 'Rejected')
3. **Payments Table**:
   * **PAYMENT\_ID**: Number, Primary Key
   * **LOAN\_ID**: Number, Foreign Key References Loans(LOAN\_ID)
   * **PAYMENT\_DATE**: Date
   * **AMOUNT\_PAID**: Number
   * **PAYMENT\_METHOD**: Varchar2(20) -- ('Cash', 'Cheque', 'Online')

**Case Study Task:**

* **Design**: Create the database schema as provided.
* **Implement**: Insert sample data into the Customers, Loans, and Payments tables.
* **Develop**: Write PL/SQL procedures for handling loan application processing, payment tracking, and generating loan statements.
* **Test**: Test the procedures with various scenarios (e.g., applying for a loan, making payments, generating loan statements, ensuring proper updates).