Project Assignment: Banking System Database with MySQL

Overview

In this project, you will design and implement a MySQL database for a fictional banking system. The goal is to create a robust and efficient database that can handle typical banking operations, such as managing customers, accounts, transactions, and employee records. This project will help you understand the practical aspects of database design, SQL queries, and database management in a real-world scenario.

Objectives

Design a relational database schema suitable for a banking system.

Implement the database schema using MySQL.

Populate the database with sample data.

Write SQL queries to perform common banking operations.

Requirements

Database Design:

Entities and Attributes:

Customer: CustomerID, FirstName, LastName, DateOfBirth, Address, Phone, Email.

Account: AccountID, AccountNumber, CustomerID, AccountType (Checking, Savings), Balance, OpenDate.

Transaction: TransactionID, AccountID, TransactionType (Deposit, Withdrawal, Transfer), Amount, Date, Description.

Employee: EmployeeID, FirstName, LastName, Position, BranchID, HireDate.

Branch: BranchID, BranchName, Location.

Relationships:

Each customer can have multiple accounts.

Each account can have multiple transactions.

Each employee works at one branch.

Each branch can have multiple employees.

Schema Implementation:

Use MySQL to create tables for the entities defined above.

Define appropriate data types and constraints (e.g., primary keys, foreign keys, NOT NULL).

Sample Data:

Insert at least 5 customers, 10 accounts, 20 transactions, 5 employees, and 2 branches.

SQL Queries:

Write SQL queries to perform the following operations:

Retrieve all customers who have a balance greater than $10,000.

List all transactions for a specific account.

Find the total number of accounts for each branch.

Calculate the total balance for each customer.

List all employees who have been working for more than 5 years.

Find the branch with the highest number of employees.

Advanced Operations (Optional):

Implement a stored procedure to transfer money between two accounts.

Create a view that shows customer details along with their account balances.

Submission Guidelines

Database Schema:

Submit the SQL script used to create the database schema.

Include comments in the script to explain the purpose of each table and key constraints.

Sample Data:

Submit the SQL script used to insert sample data into the database.

Ensure that the data covers various scenarios to test the database's functionality.

SQL Queries:

Submit a document containing the SQL queries for the operations listed above.

Include a brief explanation of each query and its expected result.

Optional Components:

If you implement the advanced operations, submit the SQL scripts for the stored procedures and views, along with explanations of their functionality.

Evaluation Criteria

Database Design (40%):

Correctness of the schema design (tables, relationships, constraints).

Proper use of data types and constraints.

Normalization is not required, but the schema should be efficient and logical.

Implementation (30%):

Accuracy of the SQL scripts used to create tables and insert data.

Proper handling of primary and foreign keys.

SQL Queries (20%):

Correctness and efficiency of the SQL queries.

Clarity and completeness of the explanations.

Optional Components (10%):

Functionality and correctness of stored procedures and views (if implemented).

Tips for Success

Plan your schema carefully before implementation to avoid major changes later.

Test your SQL queries thoroughly to ensure they return correct and expected results.

Use MySQL Workbench or any other preferred tool to manage your database efficiently.

Document your work clearly to make it easy to understand and evaluate.