

# The Bank Database Case

*James William Martinsson Co*

This document outlines the functionalities, structure and rationale behind the development of this banking database. The primary objective of this project was to design a *conceptual*, *logical* and *physical* database model tailored for banking operations.

The *conceptual* and *logical* models were created using **Draw.io**, while the *physical* implementation was developed using **SQL Server Management Studio (SSMS).**

This project represents the final individual assignment for the “***Database Design and Modelling***” course at Nackademin.

While some functionalities may require integration with external systems, the database has been designed to be adaptable to the requirements of most modern banks.

Table of Contents

[The Bank Database Case 1](#_Toc197010884)

[Conceptual Model 3](#_Toc197010885)

[Logical Model 4](#_Toc197010886)

[SQL Script 5](#_Toc197010887)

[Creation of the Database and Schemas 5](#_Toc197010888)

[Creation of the Tables 5](#_Toc197010889)

[Insertion of Fake Data 6](#_Toc197010890)

[Tables 7](#_Toc197010891)

[Table Overview 16](#_Toc197010892)

[Third Normal Form (3NF) 19](#_Toc197010893)

[Long-Term Improvements 19](#_Toc197010894)

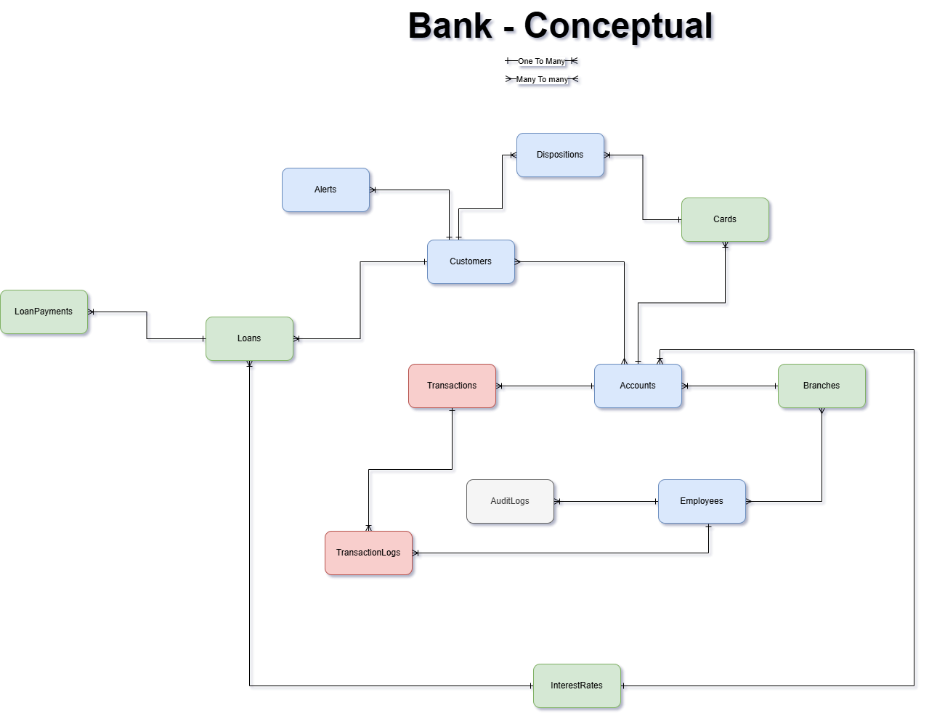
[Security Enhancements 19](#_Toc197010895)

[Additional Tables 19](#_Toc197010896)

[Stored Procedures, Indexes and Functions 19](#_Toc197010897)

[Files 20](#_Toc197010898)

***Note: Shortcut! You can click on one of the titles in the Table of Contents if you want to jump to those sections.***



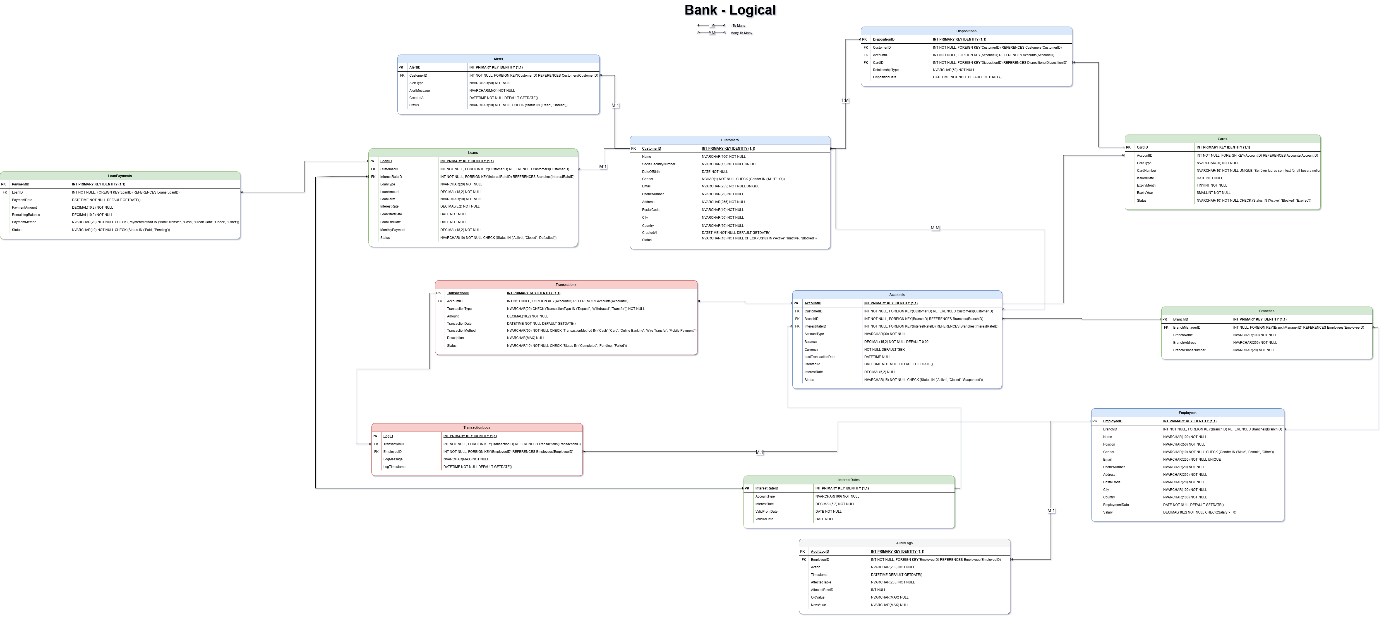
(The original file is provided separately.)

# Conceptual Model

For us to create a conceptual model, we needed to do some research. I decided to do that with the help of my current bank “Handelsbanken”. Having access to a bank as a customer helps me think of ideas and what kind of tables I need to create for a functional and well-structured database model for all kinds of bank companies.

**The tables:**

* **Accounts** = Stores details of each bank account.
* **Alerts** = Stores notifications and alerts for customers (ex. low balance, large withdrawal etc.)
* **AuditLogs** = Stores logs of important system actions for auditing purposes.
* **Branches** = Stores information about the bank branches.
* **Cards** = Stores information about credit and debit cards linked to accounts.
* **Customers** = Stores personal information of customers.
* **Dispositions** = Links customers to cards and accounts, representing relationships.
* **Employees** = Stores information about employees.
* **InterestRates** = Stores information about the interest rates for different account types.
* **LoanPayments** = Tracks payment made towards loan
* **Loans** = Stores details about loans taken by customers.
* **Transactions** = Stores all the transactions made on bank accounts.
* **TransactionLogs** = Tracks transaction activities such as errors and tracking fraud attempts.



(The original file is provided separately.)

# Logical Model

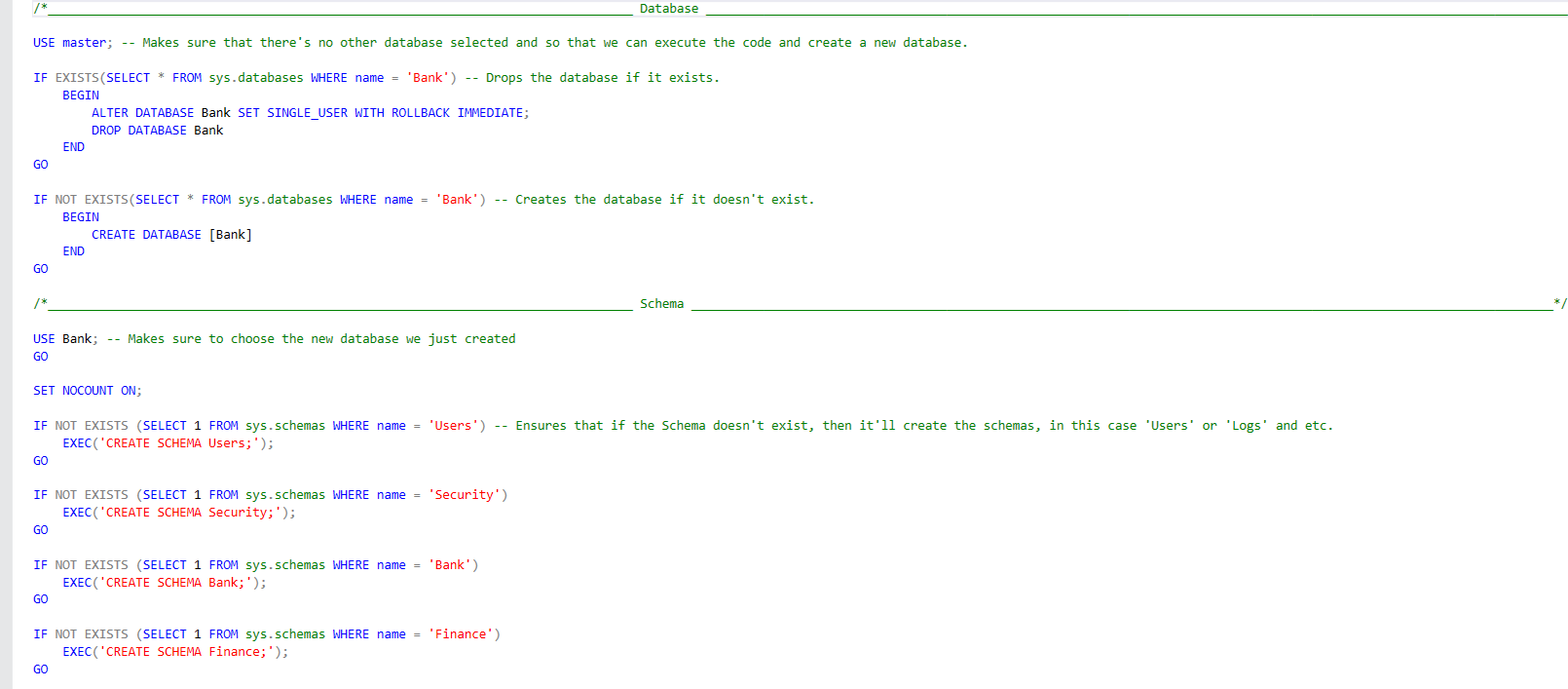
After the conceptual model was finished, it was then time to go further in the project and create and finalize our logical model. We’ve created a model that shows detailed tables with columns and datatypes. As well as using Crow’s foot notation to indicate the different relations of the tables. With this, we can create our SQL Script and finalize our project. We also made sure to follow the 3NF for our model.

Tables and their one-to-many / many-to-many relationships with other tables:

|  |  |  |
| --- | --- | --- |
| Table | One-To-Many (1:M) | Many-To-Many (M:M) |
| Accounts | Branches, Cards, InterestRates, Transactions | Customers |
| Alerts | Customers |  |
| AuditLogs | Employees |  |
| Branches | Accounts | Employes |
| Cards | Accounts, Dispositions |  |
| Customers | Alerts, Dispositions, Loans | Accounts |
| Dispositions | Cards, Customers | Cards 🡨🡪 Dispositions 🡨 🡪 Customers |
| Employees | AuditLogs, TransactionLogs | Branches |
| InterestRates | Accounts, Loans |  |
| Loans | Customers, InterestRates, LoanPayments |  |
| Transactions | Accounts, TransactionLogs |  |
| TransactionLogs | Transactions, Employees |  |

# SQL Script

## Creation of the Database and Schemas



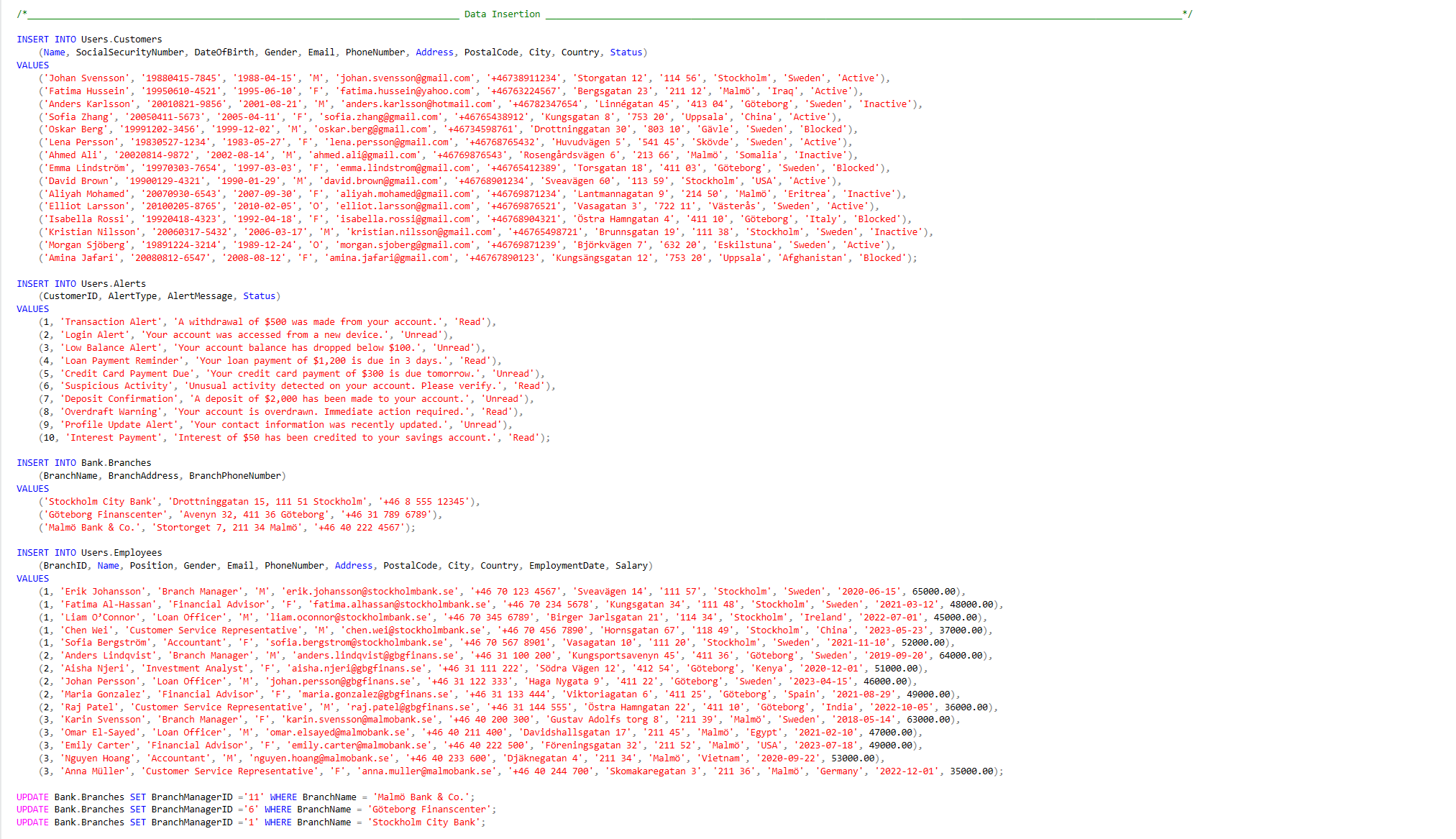
These codes ensure that any database or schema with the same name is removed and then the database and schemas can then be created successfully without any errors.

## Creation of the Tables



We created the tables and ensured that the tables are in the right orders to prevent errors with foreign keys. (We created Bank.Branches without the foreign key from Employees and added it afterwards with the Alter code as seen in the screenshot above.)

## Insertion of Fake Data



We inserted data that is of course fake to showcase how our tables would look like in production with data. (Because Employees table was created after the Branches table, we had to add the Branch Managers for each branch after adding our Employees into the Employees Table as seen in the screenshot above.)

This script is ready from the get-go, just execute it and you’ll get a message saying that the script was executed successfully without errors:



# Tables

**Schema** Users **Primary Key:** 🔑

**Tables** Customers **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | CustomerID | INT | No | PRIMARY KEY  IDENTITY(1,1) | - | Unique identifier for customers |
| 2. | - | Name | NVARCHAR(100) | No | - | - | Customer’s name |
| 3. | - | SocialSecurityNumber | NVARCHAR(13) | No | UNIQUE | - | Customer’s social security number (must be unique) |
| 4. | - | DateOfBirth | DATE | No | - | - | Customer’s date of birth |
| 5. | - | Gender | NCHAR(1) | No | (Gender IN ('M', 'F', 'O')) | - | Customer’s gender (M = Male, F = Female, O = Others) |
| 6. | - | Email | NVARCHAR(255) | No | UNIQUE | - | Customer’s email (must be unique) |
| 7. | - | PhoneNumber | NVARCHAR(20) | No | - | - | Customer’s phone number |
| 8. | - | Address | NVARCHAR(255) | No | - | - | Customer’s address |
| 9. | - | PostalCode | NVARCHAR(10) | No | - | - | Postal code |
| 10. | - | City | NVARCHAR(50) | No | - | - | City name |
| 11. | - | Country | NVARCHAR(60) | No | - | - | Country name |
| 12. | - | CreatedAt | DATETIME | No | DEFAULT GETDATE() | - | Timestamp when the customer was created |
| 13. | - | Status | NVARCHAR(10) | No | CHECK (Status IN ('Active', 'Inactive', 'Blocked')) | - | Customer status (Active, Inactive or Blocked) |

**Schema** Users **Primary Key:** 🔑

**Tables** Alerts **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | AlertID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for alerts |
| 2. | **🔑➡** | CustomerID | INT | No | FOREIGN KEY | Users.Customers(CustomerID) | References the customer triggering an alert |
| 3. | - | AlertType | NVARCHAR(50) | No | - | - | Declares the type of alert |
| 4. | - | AlertMessage | NVARCHAR(MAX) | No | - | - | Alert messages |
| 5. | - | CreatedAt | DATETIME | No | DEFAULT GETDATE() | - | Timestamp when the alert was created |
| 6. | - | Status | NVARCHAR(10) | No | CHECK (Status IN ('Read', 'Unread')) | - | Alerts status (Read or Unread) |

**Schema** Users **Primary Key:** 🔑

**Tables** Employees **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | EmployeeID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for employees |
| 2. | **🔑➡** | BranchID | INT | No | FOREIGN KEY | Bank.Branches(BranchID) | References the branches the employees works for |
| 3. | - | Name | NVARCHAR(100) | No | - | - | Employee’s name |
| 4. | - | Position | NVARCHAR(50) | No | - | - | Employee’s position |
| 5. | - | Gender | NVARCHAR(10) | No | CHECK (Gender IN ('M', 'F', 'O')) | - | Customer’s gender (M = Male, F = Female, O = Others) |
| 6. | - | Email | NVARCHAR(255) | No | UNIQUE | - | Employee’s email (must be unique) |
| 7. | - | PhoneNumber | NVARCHAR(20) | No | - | - | Employee’s phone number |
| 8. | - | Address | NVARCHAR(255) | No | - | - | Employee’s address |
| 9. | - | PostalCode | NVARCHAR(10) | No | - | - | Postal code |
| 10. | - | City | NVARCHAR(100) | No | - | - | City name |
| 12. | - | Country | NVARCHAR(100) | No | - | - | Country name |
| 13. | - | EmploymentDate | DATE | No | DEFAULT GETDATE() | - | Timestamp when the employee was employed |
| 14. | - | Salary | DECIMAL(18,2) | No | CHECK(Salary >= 0) | - | Employee salary (Ensures that salary is 0 or higher) |

**Schema** Users **Primary Key:** 🔑

**Tables** Accounts **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | AccountID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for accounts |
| 2. | **🔑➡** | CustomerID | INT | No | FOREIGN KEY | Users.Customers(CustomerID) | References which customers owns the account |
| 3. | **🔑➡** | BranchID | INT | No | FOREIGN KEY | Bank.Branches(BranchID) | References which branch the accounts are created for |
| 4. | **🔑➡** | InterestRateID | INT | No | FOREIGN KEY | Bank.InterestRates(InterestRateID) | References the interest rate applied to the account |
| 5. | - | AccountType | NVARCHAR(30) | No | - | - | Specifies the type of account |
| 6. | - | Balance | DECIMAL(15,2) | No | DEFAULT 0.00 | - | Displays the current total balance in the account |
| 7. | - | Currency | NVARCHAR(3) | No | DEFAULT ‘SEK’ | - | Specifies the currency (Default SEK) |

**Schema** Users **Primary Key:** 🔑

**Tables** Dispositions **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | DispositionID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for dispositions |
| 2. | **🔑➡** | CustomerID | INT | No | FOREIGN KEY | Users.Customers(CustomerID) | |  | | --- | | References which customer is associated with the disposition |  |  | | --- | |  | |
| 3. | **🔑➡** | AccountID | INT | No | FOREIGN KEY | Users.Accounts(AccountID) | |  | | --- | | References which account is associated with the disposition |  |  | | --- | |  | |
| 4. | **🔑➡** | CardID | INT | No | FOREIGN KEY | Bank.Cards(CardID) | References which card is associated with the disposition |
| 5. | - | RelationshipType | NVARCHAR(50) | No | - | - | Specifies the type of relationship between the customer and the account or card |
| 6. | - | DispositionDate | DATETIME | No | DEFAULT GETDATE() | - | Timestamp when the disposition was created |

**Schema** Bank **Primary Key:** 🔑

**Tables** Branches **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | BranchID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for branches |
| 2. | **🔑➡** | BranchManagerID | INT | No | FOREIGN KEY | Users.Employees(EmployeeID) | References this branch’s manager |
| 3. | - | BranchName | NVARCHAR(100) | No | - | - | Specifies the name of the branch |
| 4. | - | BranchAddress | NVARCHAR(255) | No | - | - | Branch address |
| 5. | - | BranchPhoneNumber | NVARCHAR(20) | No | - | - | Branch phone number |

**Schema** Bank **Primary Key:** 🔑

**Tables** InterestRates **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | InterestRateID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for interest rates |
| 2. | - | AccountType | NVARCHAR(100) | No | - | - | Specifies the type of account |
| 3. | - | InterestRate | DECIMAL(5,2) | No | - | - | Specifies the interest rate |
| 4. | - | ValidFromDate | DATE | No | - | - | Specifies the date the interest rate is valid from |
| 5. | - | ValidToDate | DATE | Yes | - | - | Specifies the date the interest rate is valid to |

**Schema** Bank **Primary Key:** 🔑

**Tables** Cards **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | CardID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for cards |
| 2. | **🔑➡** | AccountID | INT | No | FOREIGN KEY | Users.Accounts(AccountID) | Referenches which account owns this card |
| 3. | - | CardType | NVARCHAR(20) | No | - | - | Specifies the type of card |
| 4. | - | CardNumber | NVARCHAR(16) | No | UNIQUE | - | Customer’s own unique card number (must be unique) |
| 5. | - | IssuedDate | DATE | No | - | - | Specifies the date this card was issued |
| 6. | - | ExpiryYear | SMALLINT | No | - | - | Specifies the date the card expires |
| 7. | - | Status | NVARCHAR(10) | No | CHECK (Status IN ('Active', 'Blocked', 'Expired')) | - | Cards status (Active, Blocked or Expired) |

**Schema** Bank **Primary Key:** 🔑

**Tables** LoanPayments **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | PaymentID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for loan payments |
| 2. | **🔑➡** | LoanID | INT | No | FOREIGN KEY | Bank.Loans(LoanID) | References which loan the loan payment is for |
| 3. | - | PaymentDate | DATETIME | No | DEFAULT GETDATE() | - | Specifies when the payment was made |
| 4. | - | PaymentAmount | DECIMAL(10,2) | No | - | - | Specifies the total payment amount |
| 5. | - | RemainingBalance | DECIMAL(10,2) | No | - | - | Specifies the remaining balance to be paid |
| 6. | - | PaymentMethod | NVARCHAR(20) | No | CHECK (PaymentMethod IN ('Bank Transfer', 'Cash', 'Credit Card', 'Check', 'Other')) | - | Payment method (Bank Transfer, Cash, Credit Card, Check or Other) |

**Schema** Bank **Primary Key:** 🔑

**Tables** Loans **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | LoanID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for loans |
| 2. | **🔑➡** | CustomerID | INT | No | FOREIGN KEY | Users.Customers(CustomerID) | References which customer has a loan |
| 3. | **🔑➡** | InterestRateID | INT | No | FOREIGN KEY | Bank.InterestRates(InterestRateID) | References which interest rate of the loan |
| 4. | - | LoanType | NVARCHAR(20) | No | - | - | Specifies the loan type |
| 5. | - | LoanAmount | DECIMAL(18,2) | No | - | - | Specifies the total loan amount |
| 6. | - | LoanTerm | NVARCHAR(10) | No | - | - | Specifies the loans length of time to be repaid |
| 7. | - | InterestRate | DECIMAL(5,2) | No | - | - | Specifies the interest rate for the loan |
| 8. |  | LoanStartDate | DATE | No | - | - | Specifies when the customer took the loan |
| 9. |  | LoanEndDate | DATE | No | - | - | Specifies when the loan is paid off |
| 10. |  | MonthlyPayment | DECIMAL(18,2) | No | - | - | Specifies the monthly payment for the loan |
| 11. |  | Status | NVARCHAR(10) | No | CHECK (Status IN ('Active', 'Closed', 'Defaulted')) | - | Loan status (Active, Closed or Defaulted) |

**Schema** Security **Primary Key:** 🔑

**Tables** AuditLogs **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | AuditLogID | INT | No | PRIMARY KEY IDENTITY(1,1) | . | Unique identifiers for audit logs |
| 2. | **🔑➡** | EmployeeID | INT | No | FOREIGN KEY | Users.Employees(EmployeeID) | References which employee made an action in the database |
| 3. | - | Action | NVARCHAR(255) | No | - | - | Specifies what type of action was made |
| 4. | - | Timestamp | DATETIME | No | DEFAULT GETDATE() | - | Timestamp when the audit was logged |
| 5. | - | AffectedTable | NVARCHAR(255) | No | - | - | Specifies which table was changed |
| 6. | - | AffectedRowID | INT | Yes | - | - | Specifies the rows id that was changed |
| 7. | - | OldValue | NVARCHAR(MAX) | Yes | - | - | Specifies the old value of the column/table |
| 8. | - | NewValue | NVARCHAR(MAX) | Yes | - | - | Specifies the new value of the column/table |

**Schema** Finance **Primary Key:** 🔑

**Tables** Transactions **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | TransactionID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for transactions |
| 2. | **🔑➡** | AccountID | INT | No | FOREIGN KEY | Users.Accounts(AccountID) | References which account made the transaction |
| 3. | - | TransactionType | NVARCHAR(20) | No | CHECK (TransactionType IN ('Deposit', 'Withdrawal', 'Transfer')) | - | Specifies the transaction type |
| 4. | - | Amount | DECIMAL(18,2) | No | - | - | Specifies the amount of the transaction |
| 5. | - | TransactionDate | DATETIME | No | DEFAULT GETDATE() | - | Timestamp when the transaction was made |
| 6. | - | TransactionMethod | NVARCHAR(50) | No | CHECK (TransactionMethod IN ('Cash', 'Card', 'Online Banking', 'Wire Transfer', 'Mobile Payment')) | - | Transaction method (Cash, Card, Online Banking, Wire Transfer or Mobile Payment) |
| 7. | - | Description | NVARCHAR(MAX) | Yes | - | - | Description of the transaction |

**Schema** Finance **Primary Key:** 🔑

**Tables** TransactionLogs **Foreign Key: 🔑➡**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Key** | **Name** | **Data Type** | **Null** | **Attributes** | **References** | **Description** |
| 1. | 🔑 | LogID | INT | No | PRIMARY KEY IDENTITY(1,1) | - | Unique identifier for transaction logs |
| 2. | **🔑➡** | TransactionID | INT | No | FOREIGN KEY | Finance.Transactions(TransactionID) | References the transaction that was logged |
| 3. | **🔑➡** | EmployeeID | INT | No | FOREIGN KEY | Users.Employees(EmployeeID) | References the employee that administered this transaction |
| 4. | - | LogMessage | NVARCHAR(MAX) | No | - | - | Description of the log |
| 5. | - | LogTimestamp | DATETIME | No | DEFAULT GETDATE() | - | Timestamp when the transaction was logged |

# Table Overview

We will now display how each table looks with the fake data generated by ChatGPT.

Users.Customers A screenshot of a computer

AI-generated content may be incorrect.

Users.AlertsA screenshot of a message

AI-generated content may be incorrect.

Users.EmployeesA screenshot of a computer

AI-generated content may be incorrect.

Users.Accounts

A table with numbers and text

AI-generated content may be incorrect.

Users.DispositionsA table with numbers and text

AI-generated content may be incorrect.

Bank.BranchesA screenshot of a computer

AI-generated content may be incorrect.

Bank.InterestRate

A screenshot of a computer screen

AI-generated content may be incorrect.

Bank.Cards

A screenshot of a card number

AI-generated content may be incorrect.

Bank.LoansA screenshot of a computer screen

AI-generated content may be incorrect.

Bank.LoanPaymentsA screenshot of a number

AI-generated content may be incorrect.

Security.AuditLogs A screenshot of a computer

AI-generated content may be incorrect.

Finance.TransacitonsA screenshot of a number of numbers

AI-generated content may be incorrect.

Finance.TransactionLogsA screenshot of a phone

AI-generated content may be incorrect.

# Third Normal Form (3NF)

We designed this database to follow the rules of Third Normal Form (3NF), which means the structure avoids unnecessary repetition and keeps the data organized in a smart way.

* All fields in the tables store only **one piece of information** (1NF).
* Every field that isn’t a primary key depends **completely** on the primary key of the table (2NF).
* There are **no fields that depend on other non-key fields** (3NF), meaning each piece of information is stored in the right place and not repeated across different tables.

# Long-Term Improvements

We were tasked with a relatively simple task, to create a database for a bank. While we think that we’ve managed to create a database that fits a bank, there are of course still things that could be improved but here are some of them:

## Security Enhancements

* Encrypting sensitive data such as Cardnumber, SocialSecurityNumber by using hashing or encryption functions
* Adding UserID, IP Address for more detailed tracking in the Audit log

## Additional Tables

These are the tables we think would work for the customers, in a realistic setting where the customer may want to have more or fewer tables, the customer would be involved during the process and add or remove tables depending on their needs. With closer contact with the customer, creating this model and database to it’s completed state would be more time efficient.

## Stored Procedures, Indexes and Functions

The next step is creating a fully functional script with different procedures and functions that is needed for a bank database to work. Creating indexes to make reoccurring columns/texts load faster.

# Files

These are the files used for this project:

* Bank\_ER\_Diagram\_Conceptual – The conceptual model of this database
* Bank\_ER\_Diagram – The more detailed logical model of this database
* Bank\_Script – The main code, just execute this file into your SSMS and you’re good to go.
* Bank\_Dokumentation – Documentation for this project (This document)

*This concludes the documentation. We appreciate your time and hope this project meets your expectations. If you have any questions or require further assistance, please feel free to reach out.*