**MIDSEM EVALUATION**

Nipun Gupta – 2020089

Aryamann Mishra – 2020035

Anishka Kadiyan – 2020282

Atharva Mehta – 2020038

**DBMS PROJECT**

**Scope of project**

In this project, we aimed to provide users with a convenient and reliable online banking experience.

The project uses the concepts of DBMS to make a database and handle the data required for an online banking system.

**Stakeholders**

* Banking customers
* Employees(Bankers)

**Entities**

1. **USER** (**user ID**, userType, password)
2. **CUSTOMER** (**customer ID**, Customer name, customer address, user ID, Date of birth, credit score)
3. **DOCUMENTS** (customer ID, Document type, Document file)
4. **BANKER** (**Employee ID**, employee name, employee address, year of joining, date of birth, user ID, Adhaar ID, Salary, branchID)
5. **ACCOUNT** (**Account number,** Balance, Category, customer ID, branch ID)
6. **BRANCH** (**Branch ID**, Branch address)
7. **LOAN** (**Loan ID**, Amount, Due date, Interest rate, mortgage, loan type, Customer ID)
8. **CARDS** (**Card number**, cvv, expiry, Account number,cardLimit,cardType)
9. User

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity Constraints |
| userID | INT | Primary key |
| userType | CHAR[50] | Not null, (banker, customer) |
| password | VARCHAR[50] | Not null |

1. Customer

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| customerID | INT | Primary key |
| customerName | VARCHAR[45] | Not Null |
| accountNumber | DOUBLE | Not Null |
| customerAddress | VARCHAR[45] | Not Null |
| userID | INT | foreignKey, Not Null |
| DoB | DATE | Not Null |
|  |  |  |

1. Banker

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| employeeID | INT | Primary key |
| employeeNAME | VARCHAR[45] | Not Null |
| employeeADDRESS | VARCHAR[45] | Not Null |
| userID | INT | foreignKey, Not Null |
| salary | VARCHAR[45] | Not Null |
| aadharID | DOUBLE | Not Null |
| DoB | DATE | Not Null |
| branchID | INT | Not Null |
| joiningYEAR | YEAR | Not Null |

1. Loan

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| loanID | DOUBLE | Primary key |
| amount | VARCHAR[45] | Not Null |
| DueDate | DATE | Not Null |
| InterestRate | VARCHAR[10] | Not Null |
| loanType | VARCHAR[45] | Not Null |
| customerID | INT | foreignKey,Not Null |
|  |  |  |

1. Documents

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| customerID | DOUBLE | ForeignKey referenced from Customer used as PrimaryKey |
| documentTYPE | VARCHAR[45] | Not Null |
| documentFILE | VARCHAR[45] | Not Null |
| customerID | INT | foreignKey,Not Null |

1. Branch

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| branchID | INT | Primary key |
| branchADDRESS | VARCHAR[45] | Not Null |

1. Account

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| accountNUMBER | INT | Primary key |
| balance | INT | Not Null |
| category | VARCHAR[45] | Not Null |
| branchID | INT | Not Null |
| customerID | INT | foreignKey,Not Null |

1. Card

|  |  |  |
| --- | --- | --- |
| Variable | Datatype | Integrity constraints |
| cardNUMBER | INT | Primary key |
| expiry | DATE | Not Null |
| cvv | INT | Not Null |
| cardLimit | INT | Not Null |
| cardType | VARCHAR[20] | Not Null |
| accountNumber | INT | foreignKey,Not Null |

1. Verifies

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| VerificationID | VARCHAR[15] | Primary key |
| Banker | employeeID |  |
| Documents | documentID |  |

1. Issues

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| Banker | employeeID |  |
| Card | cardNumber |  |

1. Submits

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| Banker | employeeID |  |
| Card | cardNumber |  |

1. Gives

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| Branch | branchID |  |
| Loan | loanID |  |

1. Make account

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| Service | serviceID |  |
| Customer | customerID |  |
| Account | accountNUMBER |  |

1. Renew Card

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| Service | bankerID |  |
| Customer | customerID |  |
| Card | cardNUMBER |  |

1. Transaction

|  |  |  |
| --- | --- | --- |
| Variable |  | Integrity Constraints(if any) |
| TransactionID | INT | PrimaryKey |
| Customer | customerID | NOT NULL |
| AccountDebited | accountNumber | NOT NULL |
| AccountCredited | accountNumber | NOT NULL |

**WEAK ENTITY**

In the given entities

* Documents

are the weak entities because they cant independently exist.

Documents cannot exist without customer and the banker. Customer is needed to provide the documents and the banker needs to verify these documents.

**TERNARY RELATIONSHIP**

Ternary Relationships in these relationships are make account, renew cards.

These are ternary relationships because they have more than one entities involved in the relationship.

Make Account : Banker,Customer,Account

Renew Card : Banker,Customer,Card

**ENTITY PARTICIPATION TYPE**

Total Participation : **Documents, BankCards, Loan, Customer**

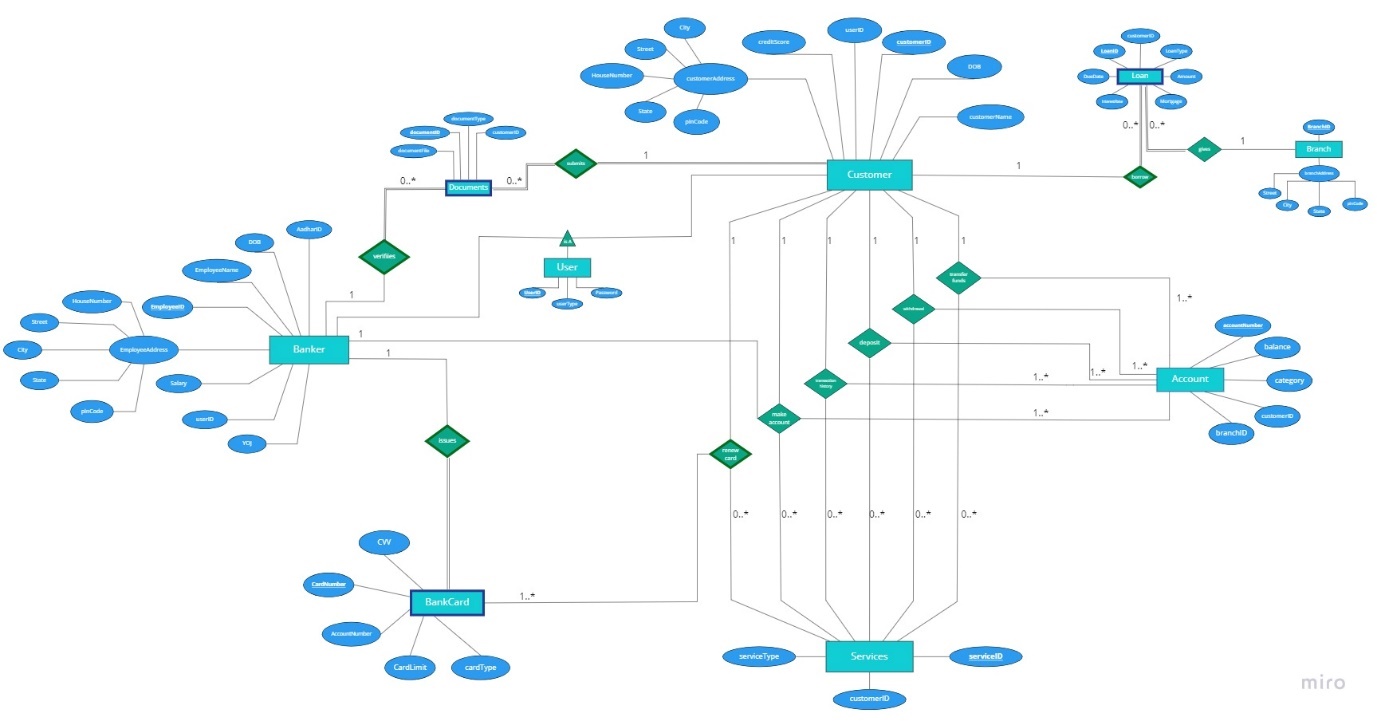
Documents participate completely in submits and verifies relationships because they have to be part of these relationships.

BankCards participate completely in issues and renew cards as they have to be a part of these relationships.

Customers have total participation in submit as without submitting they cant proceed to any service.

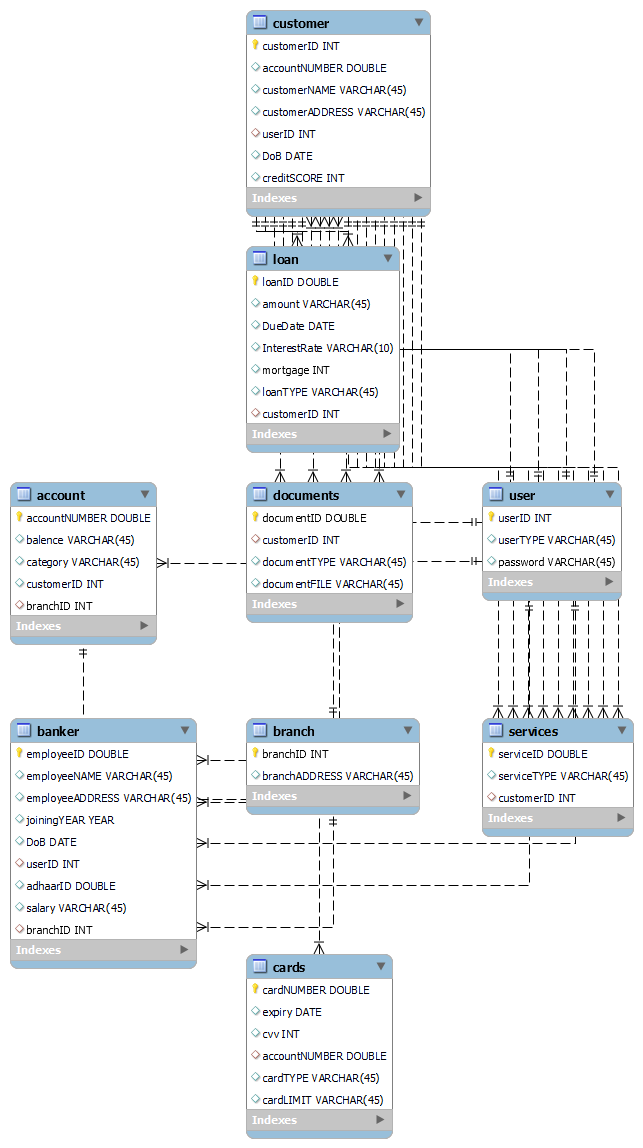
Loan participates completely in gives and borrow relationships. Without loan there is not concept of give and borrow.

**ER DIAGRAM**



**Link to the ER Diagram** <https://miro.com/welcomeonboard/ZFZ2UDJJRzBmWDVpZDBtR2tRb2tGbWJXaGw4ejZ0WmtZYzJJcUtrS2dXTm1DMlJTQTZObklnbmI2ZkpZeFpNTXwzMDc0NDU3MzU0MjY2MjY2OTg4?invite_link_id=486978681630>

**Relationship Schema**

****

**SQL Queries**

1. **Withdrawing money from bank account only if the account has sufficient balance**

Update account set balance = balance-1000 where customerID = ID and balance >= 1000

1. **Listing all the customer ID’s and name of all the customers who have taken loan**

SELECT customer.customerID, customer.customerNAME FROM loan INNER JOIN customer ON loan.customerID = customer.customerID

1. **Grouping all the employee’s according to joining Year in decreasing order**

SELECT joiningYEAR, COUNT(employeeID) from banker group by joiningYEAR order by joiningYEAR desc

1. **Listing all the customer’s ID’s and names who have a creditScore > 300 and account net balance > $31624.49**

select customerID, customerNAME from customer where creditSCORE > 300 and customerID in (select customerID from account where balence > '$31624.49')

1. **List all the accountNumbers which have more than $31624.49 in their accounts and have a credit or debit card**

select accountNUMBER from cards where accountNUMBER in (select accountNUMBER from account where balence > '$31624.49')

1. **Finding all the customers who have a credit score<200 but still have a loan sanctioned by the bank**

select customerID, customerNAME from customer where exists(select customer.customerID from customer inner join loan on customer.customerID=loan.customerID where customer.creditSCORE < 200)

1. **List all the customers who have availed the service of depositing money in their bank account**

select customerNAME, customerID from customer where customerID in (select customerID from services where serviceTYPE = 'deposit')

1. **Listing all the customers who have more than $61112.05 in their accounts and have their name beginning with “kip”.**

select customerNAME, customer.customerID, balence from customer, account where customerNAME like 'kip%' and customer.customerID = account.customerID and customer.customerID in(select customerID from account where balence > '$61112.05')

1. **If the bank employee has more than 20 years of experience then we give him/her a raise.**

update banker set salary = '$10000' where year(curdate()) - joiningYEAR >= 20

1. **Deleting all the cards that have expired(expiry date of tha card has passed)**

delete from cards where expiry = curdate()