**CONSOLE BASED BANKING APPLICATION**

**OBJECTIVE:**

Developing a console based banking application using java and DBMS to query to store the information. Using JDBC for the connectivity.

**INDENTIFICATION OF NEED**: (where it is used and needed)

It can used to process the transaction and all other banking operation through the console based application. To make the process easier and efficient.

**PLATFORM SPECIFICATION:**

1. HARDWARE:
   * + Intel core i3, i5, i7 or mac
     + minimum 4GB Ram
2. SOFTWARE:
   * + Java
     + SQL
     + JDBC
     + JIRA

**FUNCTIONAL REQUIREMENT:**

1.User Account Management

* Savings current account
* View account type, balance,
* Update account such as contact details

2.Transaction Management

* Deposit, withdrawal from account (do not conflict)
* Transfer funds b/w account
* Ensure thread-safe transaction using multithreading
* Transaction History

3.Transaction History

* Save a log of all transaction to a file, allowing users to view a history of their account activities.
* Retrieve and display transaction history for specific account from the saved file.

4.Database Operation (using JDBC)

* Store account and transaction details in relation database
* Perform curd operation for account management
* Generic reports based on the data stored in the DB

5.Reports

Generate various report such as:

* Account detail for each customer
* Total balance across all account
* Number of accounts by types (Saving or current account)
* Daily summaries of deposit, withdrawal, transfer.

**SCHEMA DESIGN:**

**1.BANK: store information about the bank**

COLUMNS:

bank\_id INT UNIQUE(PK)

bank\_Name VARCHAR (100)

Bank\_branch VARCHAR (100)

**2.ACCOUNT: stores account detail, association with a particular bank**

COLUMNS:

Account\_id INT UNIQUE (PK)

Customer\_id INT

Bank\_id INT (FK)

Account\_type VARCHAR (50)

**3.SAVINGS ACCOUNT AND CURRENT ACCOUNT: specialized table for different account types, inheriting from the Account table**

COLUMNS(SAVINGSACOUNT):

Account\_id INT(FK)

Interset\_rate DECIMAL (5,2)

COLUMNS (CURRENTACCOUNT):

Account\_id INT(FK)

Overdraft\_limit DECIMAL (15,2)

**4.TRANSACTION: records all transaction (deposit, withdraw, transfer) linked to account**

COLUMNS:

Transaction\_id INT(PK)

Account\_id INT(FK)

Transaction\_type VARCHAR (50)

Amount DECIMAL (15,2)

Transaction\_date TIMESTAMP

**5.DEPOSITTRANSACTION, WITHDRAWTRANSACTION: specialized table for different transaction type, inheriting from the transaction table**

COLUMNS(WITHDRAWTRANSACTION):

Transaction\_id INT (FK)

Withdrawal\_method VARCHAR (50)

COLUMNS(DEPOSITTRANSACTION):

Transaction\_id INT (FK)

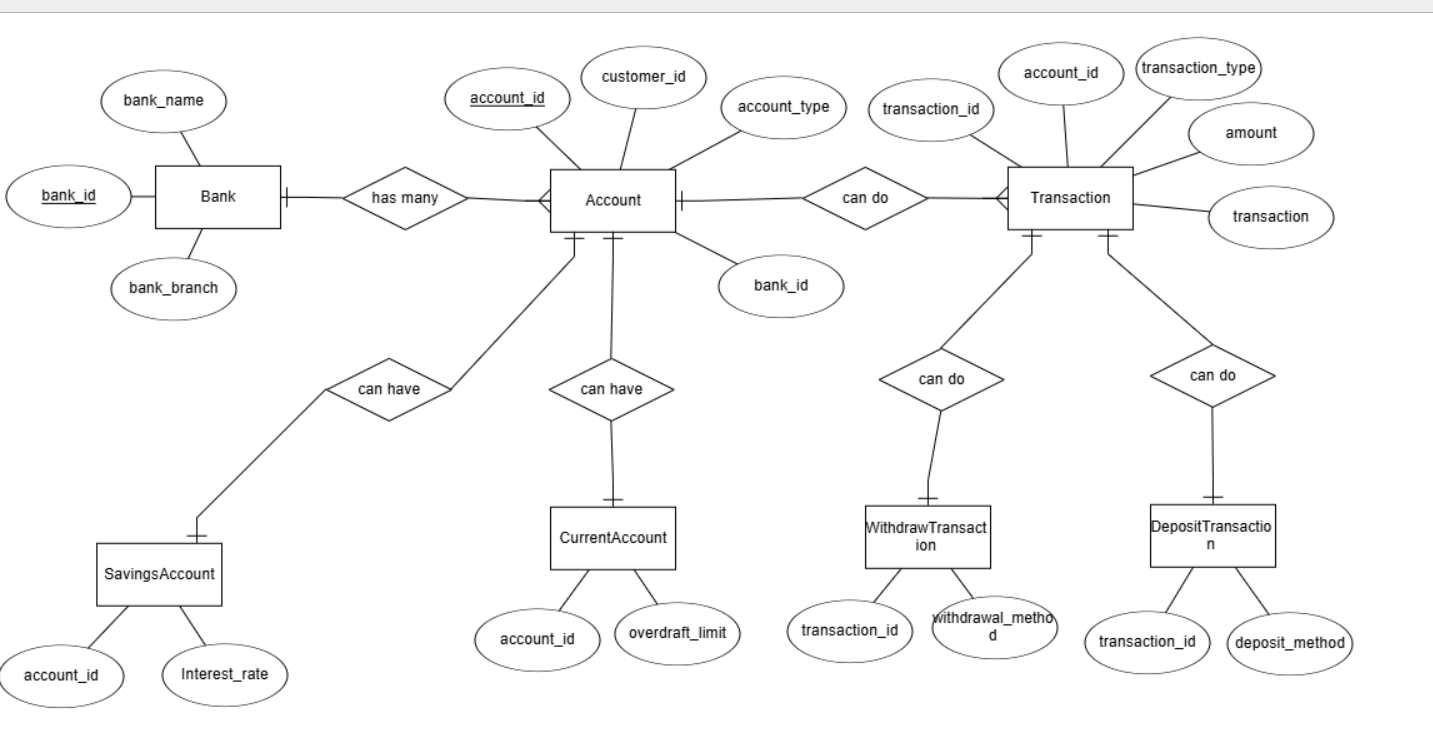
Deposit\_method VARCHAR (50)

**USE CASE DIAGRAM**

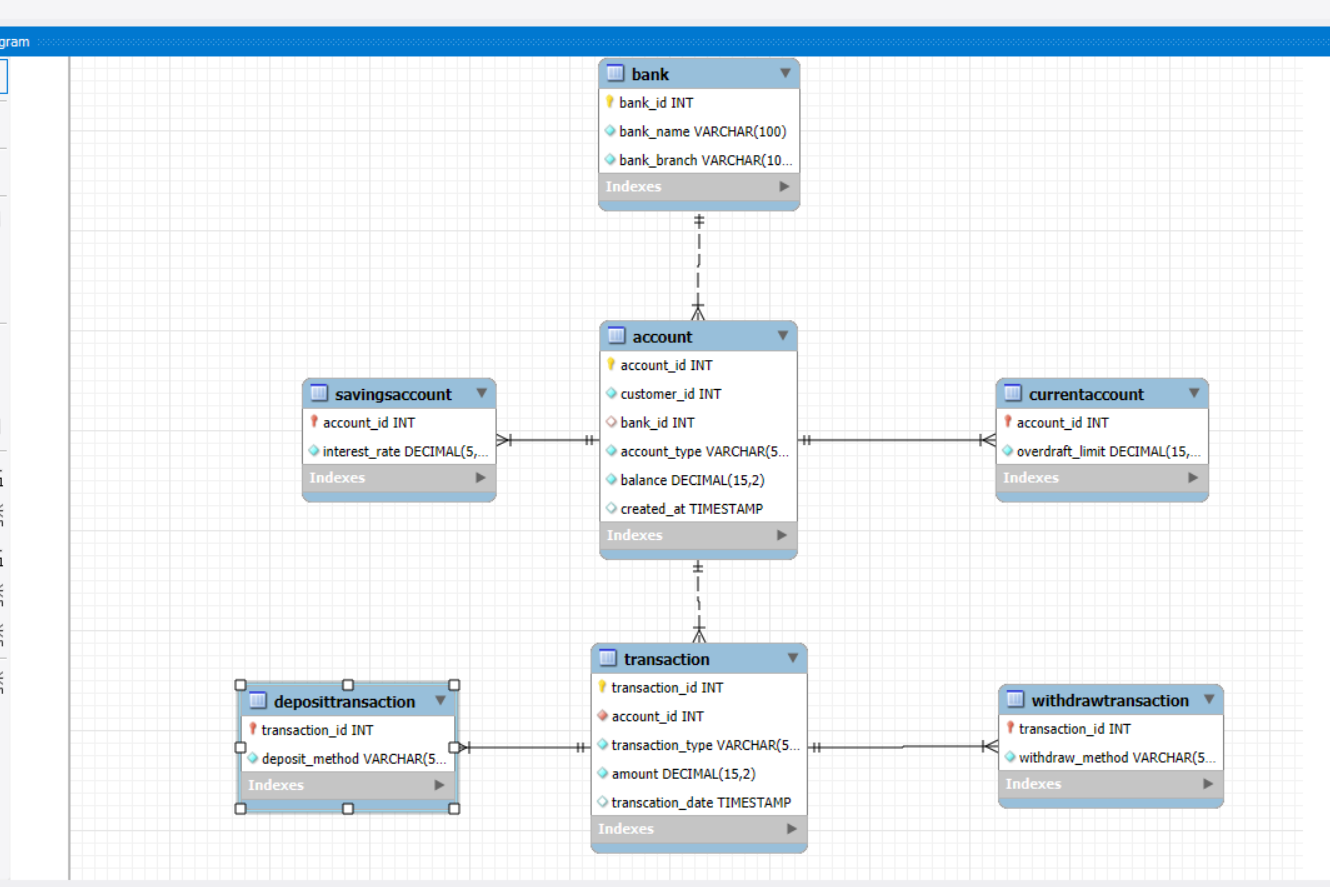
BANKING APPLICATION

USER

**ER RELATIONAL DIAGRAM**



**SCHEMA DIAGRAM**

****

|  |  |  |
| --- | --- | --- |
| **RELATIONSHIP** | **TYPE** | **DESCRIPTION** |
| Bank -> Account | One To Many | Links Each Bank To Its Accounts, Allowing Each Account To Be Associated With A Specific Bank |
| Account -> SavingsAccount | One To One | Extends Account With Specific Attributes For Saving Accounts, Using Savings Accounts For Savings Specific Data Like Interest Rate |
| Account -> CurrentAccount | One To One | Extends Account With Specific Attributes For Current Accounts, Using Current Account For Data Like Overdraft Limit |
| Account -> Transaction | One To Many | Associates Multiple Transactions With A Single Account, Tracking All Financial Activities Related To Each Account |
| Transaction -> depositTransaction | One To One | Add Details Specific To Deposit Transactions, Such As Deposit Methods, To The Basic Transaction Data. |
| Transaction -> withdrawalTransaction | One To One | Add Details Specific To Withdrawal Transactions, Such As Withdrawal Method, To The Base Transactions Data. |