

BANK MANAGEMENT SYSTEM

Prepared By:

Priyanshu Gupta (112001033)

Vikas Saini (112001049)

Aaryuman Sawai (112001002)

Submitted To:

Dr. Koninika Pal

RELATIONAL DATABASE DESIGN

1. Branch(

branch_id serial NOT NULL,
name varchar(15) NOT NULL,
address varchar(15) NOT NULL,
PRIMARY KEY (branch_id)

)

2. Account

(

account_number serial NOT NULL,
branch_id int NOT NULL,
isOpen BOOLEAN NOT NULL,
balance INT NOT NULL,
primary key(account_number),
FOREIGN KEY (branch_id) REFERENCES branch(branch_id)

)

3. customer

(

Aadhar_number int not null check(Aadhar_number > 0),
cust_name char(50) not null,
cust_address char(50) not null,
pan_number int not null check(pan_number > 0),
income int not null check(income > 0),
primary key (Aadhar_number)

)

4. Holds

```
(  
    Aadhar_number int not null,  
    account_number int NOT NULL check (account_number > 0),  
    foreign key(Aadhar_number)  
    references customer(Aadhar_number),  
    foreign key(account_number)  
    references Account(account_number),  
    primary key (Aadhar_number,account_number)  
);
```

```
5. Atm_card(  
    issue_date date not null ,  
    account_number int NOT NULL check (account_number > 0),  
    atm_card_number int not null check(atm_card_number > 0),  
    foreign key(account_number) references  
    Account(account_number) ,  
    primary key (atm_card_number));
```

```
6. Supports (  
    account_number int NOT NULL check (account_number > 0),  
    atm_card_number int not null check(atm_card_number > 0),  
    foreign key(account_number) references  
    Account(account_number),  
    foreign key(atm_card_number) references  
    atm_card(atm_card_number),  
    primary key(atm_card_number));
```

```
7. Credit_card (
```

```
account_number int NOT NULL check (account_number > 0),
card_limit int not null check (card_limit >= 0 and card_limit <=
500000),
color varchar not null ,
expiry_date date not null,
credit_card_issue_date date not null,
card_id int not null,
foreign key (account_number) references
Account(account_number) ,
primary key (card_id));
```

```
8. fixed_deposit (
    open_date date NOT NULL,
    fd_id int NOT NULL,
    account_number int NOT NULL,
    interest float not null check (interest > 0),
    duration int not null,
    primary key (fd_id),
    foreign key (account_number) references
Account(account_number)
);
```

```
9. Locker (
    locker_id int NOT NULL,
    fd_id int,
    issue_date date,
    availability int NOT NULL,
    size int NOT NULL,
    foreign key (fd_id) references fixed_deposit(fd_id),
    primary key (locker_id));
```

```
10. transaction (  
    transaction_id serial NOT NULL,  
    sender_id int NULL,  
    receiver_id int NULL,  
    amount float NULL,  
    date varchar(15),  
    PRIMARY KEY (transaction_id),  
    FOREIGN KEY (sender_id) REFERENCES  
Account(account_number),  
    FOREIGN KEY (receiver_id) REFERENCES  
Account(account_number)  
);
```

```
11. has(  
    branch_id int not null,  
    account_number int not null,  
    foreign key (branch_id) REFERENCES branch(branch_id),  
    foreign key (account_number) references  
account(account_number),  
    primary key (branch_id, account_number)  
);
```

```
12. employee (  
    employee_id serial NOT NULL,  
    name varchar(15) NULL,  
    address varchar(15) NULL,  
    salary int NULL,  
    branch_id int NOT NULL,  
    dob varchar(15) NULL,
```

```
PRIMARY KEY (employee_id),  
FOREIGN KEY (branch_id) REFERENCES branch(branch_id)  
);
```

```
13. loan (  
    loan_id int NOT NULL,  
    account_number int NOT NULL,  
    rate FLOAT check (rate >= 0 and rate <= 100) not null,  
    issued_amount int NOT NULL,  
    due_amount INT NULL,  
    issue_date DATE NOT NULL,  
    foreign key (account_number) references  
account(account_number),  
    primary key (loan_id)  
);
```

```
14. loan_repayment(  
    repayment_id SERIAL,  
    loan_id INT NOT NULL,  
    amount INT NOT NULL,  
    date DATE NOT NULL,  
    PRIMARY KEY (repayment_id),  
    FOREIGN KEY (loan_id) REFERENCES loan(loan_id)  
);
```

```
15. deposit  
(  
    deposit_id serial not null,  
    account_number int not null,
```

amount float NULL,
Primary key (deposit_id),
FOREIGN KEY (account_number) references Account
(account_number)
);

16. withdrawal

(
withdrawal_id serial not null,
account_number int not null,
withdrawal_amount float NULL,
Primary key (withdrawal_id),
FOREIGN KEY (account_number) references Account
(account_number)
);

List of relations		
Schema	Name	Type
public	account	table
public	atm_card	table
public	branch	table
public	credit_card	table
public	customer	table
public	deposit	table
public	employee	table
public	fixed_deposit	table
public	has	table
public	holds	table
public	loan	table
public	loan_repayment	table
public	locker	table
public	supports	table
public	transaction	table
public	withdrawal	table

INTEGRITY AND GENERAL CONSTRAINTS

1. Foreign Key:

1. Account Table : Branch Id (Branch)
2. Holds : Aadhar Number (Customer)

3. Holds : Account Number (Account)
4. Supports : Account Number (Account)
5. Supports : Atm Card number (ATM Card)
6. Credit Card : Account Number (Account)
7. Fixed Deposit : Account Number (Account)
8. Locker : Fd Id (Fixed Deposit)
9. Transaction : Account Number (Account)
10. Has : Branch Id (Branch)
11. Has : Account Id (Account)
12. Loan : Account Number (Account)
13. Loan Repayment : Loan id (Loan)
14. Employee : Branch Id (Branch)
15. Withdrawal : Account Number (Account)
16. Deposit : Account Number (Account)

2. Check :

1. Aadhar number > 0 : Customer
2. Pan number > 0 : Customer
3. Income > 0 : Customer
4. Atm Card Number > 0 : ATM Card
5. Card Limit $\in [0, 500000]$: Credit Card
6. Interest > 0 : Fixed Deposit
7. Rate $\in [0, 100]$: Loan

3. Unique and Primary Key Constraints :

1. Branch : Branch Id
2. Account : Account Number
3. Customer : Aadhar Number
4. Credit Card : Card Id
5. Fixed Deposit : FD Id
6. Locker : Locker Id
7. Transaction : Transaction Id
8. Employee : Employee Id
9. Loan : Loan Id
10. Loan Repayment : Repayment Id
11. Withdraw : Withdrawal Id
12. Deposit : Deposit Id

4. Checks By Triggers :

1. Transaction Check (Insufficient Balance)
2. Withdrawal Check (Insufficient Balance)

Note : All entries in all tables are ensured not null either by putting constraints while defining tables or by setting them not null through trigger.

Example :

Set Due Amount (Set loan due to be equal to loan issued which is initially null)

FUNCTIONS AND TRIGGERS

1. Functions and Procedures:

1. Add account : Opens Account by parameters (Account Number, Branch Id)

2. Last Repay Date : Finds last Repayment Date corresponding to a loan id
3. loan_amount_func : This function returns total amount of loan given by a branch (till date)
4. transaction_of_branch : this function returns all transactions related to a branch(given)
5. Set Due Amount : Sets due amount to be the issued amount when loan is issued.
6. Update Due Amount : If a person repays a part of the loan which is checked using the repayment table's new entry it updates the due amount in the loan table using compound interest 😊.
7. Update Amount : If an entry is added in the transaction table balance is updated at senders and receivers end making it atomic operation.

2. Triggers :

1. Update amount : Updates balance at both ends if checks are satisfied using the function.

2. Deposit trigger : Add the required amount in the account number whenever any row is inserted in the deposit table.
3. Withdrawal trigger : If the account from where withdrawal happening has money then we will Cut the amount from that account else we weill raise notice of insufficient funds.
4. Loan Amount : Sets loan due amount to be the issued amount using the function.
5. Update Loan Amount : Update Loan due amount to after a part of loan repayment using the function.

Queries using functions:

1. While calculating the new due amount, we fetch the last date of repayment for a particular loan. And then calculate the interest.
2. Query will call procedure add_account with argument as account number and account will be opened with that account number.
3. Query will call loan_amount_func function with argument as branch id and sum of loan that was issued from that branch will be shown.
4. Query will call transaction_of_branch function with argument as branch id and table will be shown which consists of all transactions where either sender or receiver is from that branch.

ROLES AND PRIVILEGES

1. Admin : All privileges
2. Employee : Select, update, insert, delete on account, locker, supports, holds, customer.

3. Customer : In general all customers which have locker availability are granted to it.

4. Customer 1 : Select on locker detail, transaction history and all privileges of the customer.

views used by roles

1. Locker Availability : table of count of lockers available on basis of their size

2. Detail : table of account number, atm card number, credit card id of the customer 1.

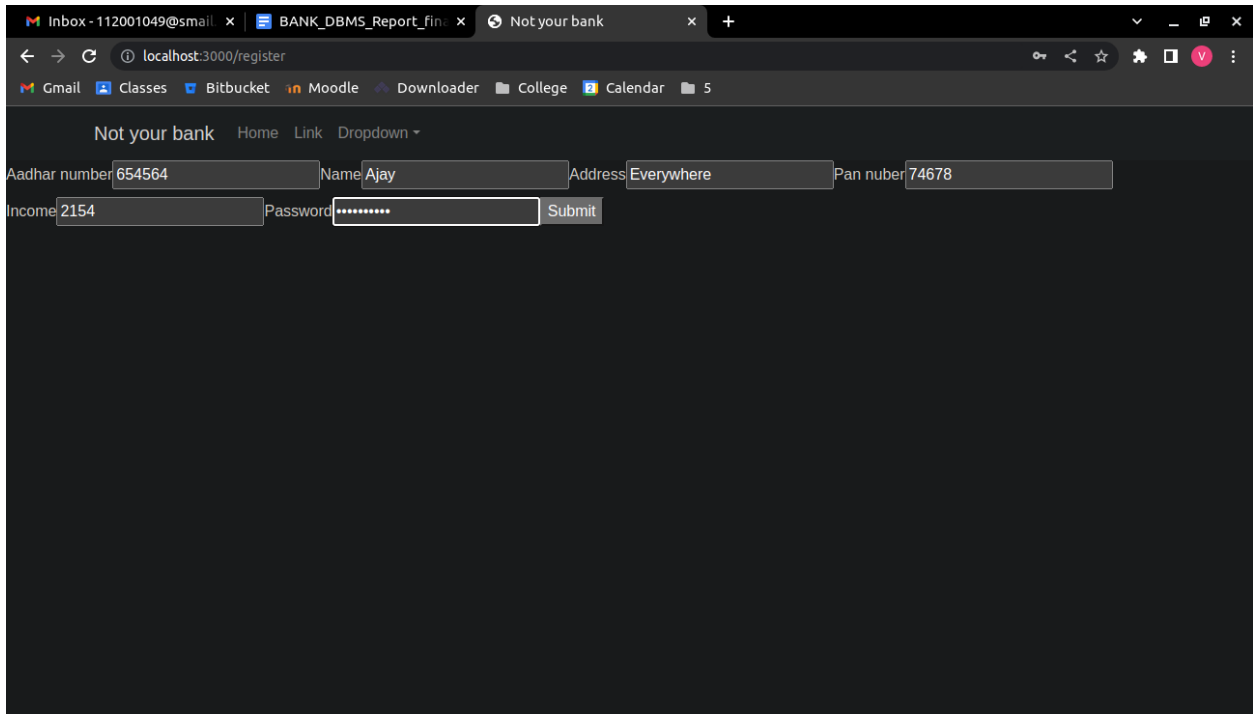
3. Transaction history : table of transactions where either sender is customer 1 or receiver is customer 1.

INDICES AND QUERIES USING THAT INDEX

1. Created b tree index for sender_id of transaction table for queries in check_balance function.
2. Created b tree index for receiver_id of transaction table for queries in check_balance function.
3. Created a b tree index for loan_id in loan table for query function last_repay_date.
4. Created a b tree index for repayment_id in loan table for query function update_due_amount
5. Created a b tree index for account_number in Account table for query function update_amount_function.

FRONTEND

A user can register (himself / herself) on banking system

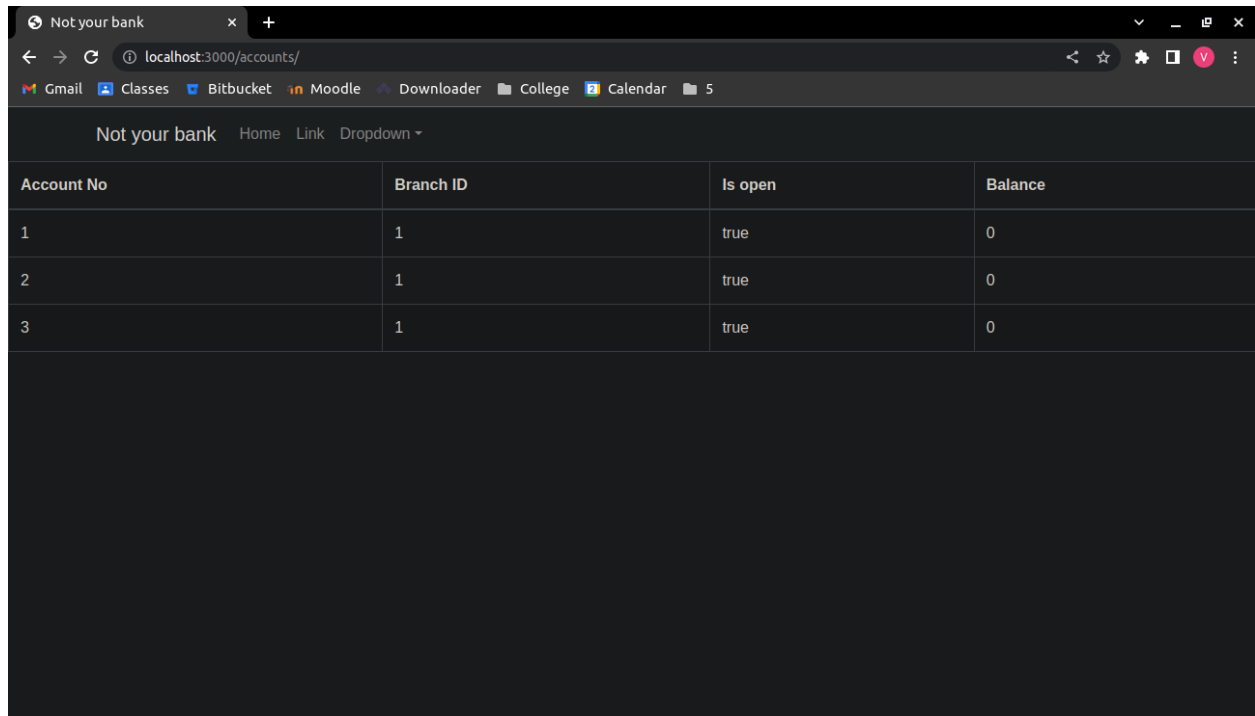


The screenshot shows a web browser window with the address bar displaying 'localhost:3000/register'. The browser's tab bar includes 'Inbox - 112001049@gmail.com', 'BANK_DBMS_Report_fin...', and 'Not your bank'. The browser's bookmark bar lists 'Gmail', 'Classes', 'Bitbucket', 'Moodle', 'Downloader', 'College', 'Calendar', and '5'. The page title is 'Not your bank' with a navigation menu containing 'Home', 'Link', and a 'Dropdown' menu. The registration form contains the following fields and values:

Field	Value
Aadhar number	654564
Name	Ajay
Address	Everywhere
Pan nuber	74678
Income	2154
Password	*****

A 'Submit' button is located at the bottom right of the form.

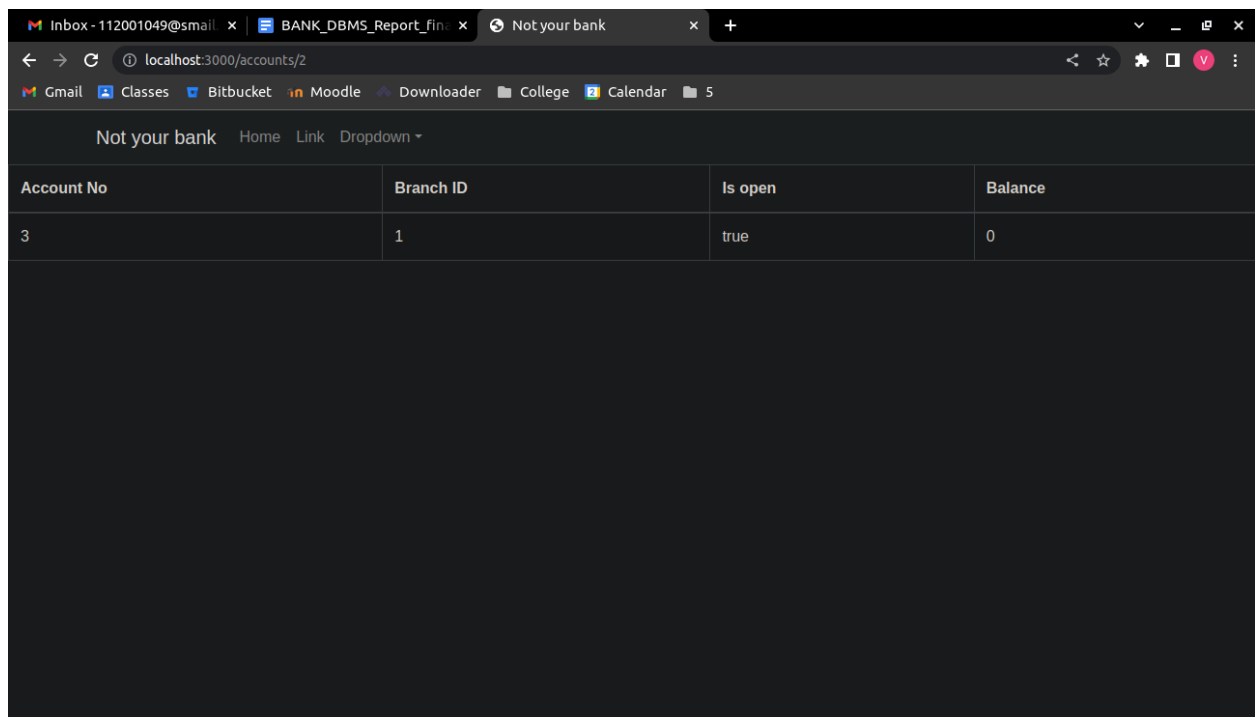
A employee can see list of accounts



The screenshot shows a web browser window with the address bar at `localhost:3000/accounts/`. The page title is "Not your bank". Below the title is a navigation bar with "Home", "Link", and a "Dropdown" menu. The main content is a table with four columns: "Account No", "Branch ID", "Is open", and "Balance". The table contains three rows of data.

Account No	Branch ID	Is open	Balance
1	1	true	0
2	1	true	0
3	1	true	0

A customer can view account details associated with him/ her.



The screenshot shows a web browser window with the address bar at `localhost:3000/accounts/2`. The page title is "Not your bank". Below the title is a navigation bar with "Home", "Link", and a "Dropdown" menu. The main content is a table with four columns: "Account No", "Branch ID", "Is open", and "Balance". The table contains one row of data.

Account No	Branch ID	Is open	Balance
3	1	true	0