SCOPE OF PROJECT

- It carries out processes easily and quickly which are not possible manually.
- It aims to automate the transactions of banks and provide a better and faster service to customers online via the internet. It focuses on better performance and paperless banking up to a point.
- Customers can perform financial transactions like transfer funds online, pay bills, apply for loans and open a savings account among various other debit card transactions.
- It will maintain a large no. of transactions with ease and safety.
- It will manage the details of all registered customers and help them to operate their accounts online via the internet.
- It will make updating, modification and deleting of records easier.

STAKEHOLDERS

1. **BANK:** Controls/owns branches and also, manages account details of all its customers across branches.
2. **Employee**: Works for the bank(specific branch) and assists customers for loans and transaction of money(cashier) or repayment of loans(manager), along with the opening of a new account, issuing of credit/debit cards.
3. **Customer**: Owns an account in the bank and can do transactions by taking loans from the bank(borrower) or depositing money to his/her account(depositor). Also, make payments for daily use using cards of bank.
4. **Insurance Company**: These provide insurance to the customers of the bank and hence, they are external stakeholder for the bank.
5. **Government**: Rules and regulations are issued by the government of the country and banks have to follow these. It can be a shareholder too(External Stakeholder).
6. **Shareholders**: These are the bodies/persons which hold shares of a bank(External Stakeholder) and they can also provide money for the management or creation of banks(Creditor).

Weak Entity

1. **Account type:** It is a weak entity as it can only exist if “Account”(entity) exists in the database.
2. **Payment:** It is a weak entity as it can only exist if “Employee” and “Customer”(entity) exists in the database.
3. **Loan:** It is a weak entity as it can only exist if “Branch” and “Customer”(entity) exists in the database.

Ternary Relationship

Loan, **Customer** and **Branch** forms a Ternary relationship with a relation called “**request**”. It is a Ternary relation as:
Loan is taken by a customer,
Loan is given by particular Branch of bank &,
Customers have an account in the same (/belongs to a particular) Branch of the bank.

RELATIONAL SCHEMA

Login(<u>email_id</u>, role_of_user, username, password**)**

Bank(<u>bank_ssn</u>, Bank_name, Head_Office{ HO_email, HO_address{ State, City, Pincode } }**)**

Customer(<u>cust_id</u>, cust_dob, cust_name, {phone_no} , {account_no}, pan_no, cust_email, address{ State, City, Pincode }**)**

Insurance(**<u>Insurance_no</u>** , type, term, premium_payment, issuing_company , cust_id, date_of_insurance**)**

Employee(<u>emp_ID</u>, emp_Salary, emp_type, emp_name, emp_DOB, {phone_no}, emp_address{ State, City, Pincode } , mgr_id**)**

Branch(<u>branch_id,</u> branch_address{ State, City, Pincode }, branch_email**)**

Loans(term, rate_of_interest, amount, type_of_loan, account_no, loan_id**)**

Account(account_no , overdraft , balance**)**

Account type(account_no **,** deposit_amt, transaction_limit, withdraw_limit, interest_rate)

Credit Card(cc_limit, **cc_number** , cc_cvv, cc_expirydate, account_no)

Debit Card(debit_number , cc_cvv, debit_expirydate, account_no)

Payment(payment_id, payment_amountdue, payment_duedate, account_no**)**

request(date_of_request, cust_id, branch_id, loan_id**)**

Branches(date_of_opening , bank_ssn , branch_id**)**

acc_branch(date_of_opening , branch_id , acc_no**)**

Assistance(cust_id , emp_id , date_of_assistance**)**

SQL QUERIES

1.
   ```
   select d.debit_number, d.debit_expirydate, c.cust_id, c.email, c.account_no
   from customer c, debit_card d
   where c.account_no = d.account_no;
   ```

2.
   ```
   SELECT *
   FROM customer
   WHERE cust_id BETWEEN 3 AND 9;
   ```

3.
   ```
   SELECT * FROM employee
   LIMIT 6;
   ```

4.
   ```
   select *
   from employee e
   where e.first_name like 'a%';
   ```

5.
   ```
   SELECT * FROM customer WHERE customer.cust_address_city IN ('Delhi',
   'Chandigarh', 'Lahore')
   union all
   SELECT * FROM customer WHERE customer.cust_address_city IN ('Imphal',
   'Daman', 'Aizwal');
   ```

6.
   ```
   SELECT emp_branch.date_of_joining, employee.first_name,
   branch.branch_email
   from ((emp_branch
   inner join employee on emp_branch.emp_id = employee.emp_id)
   inner join branch on branch.branch_id = emp_branch.branch_id);
   ```

7.
   ```
   SELECT COUNT(cust_id), customer.cust_address_city
   FROM customer
   GROUP BY customer.cust_address_city
   HAVING COUNT(cust_id) > 1;
   ```

8.
   ```
   SELECT customer.first_name, loans.loan_id
   FROM customer, loans
   where loans.account_no = customer.account_no
   ORDER BY customer.first_name;
   ```

9.
   ```
   CREATE VIEW vname as
   SELECT payment.payment_id, payment.payment_duedate,
   credit_card.cc_number, credit_card.cc_expirydate
   FROM payment, credit_card
   WHERE payment.account_no = credit_card.account_no;
   SELECT * FROM vname;
   drop view vname;
   ```

10.
    ```
    SELECT customer.first_name, customer.last_name
    FROM customer
    ```

```
WHERE EXISTS (SELECT account_branch.date_of_opening from
account_branch where account_branch.account_no = customer.account_no and
customer.account_no > 600000 );
```

11.
    ```
    SELECT credit_card.cc_number, debit_card.debit_number, customer.cust_id
    FROM credit_card, debit_card, customer
    WHERE customer.account_no = debit_card.account_no and
    debit_card.account_no = credit_card.account_no;
    ```

12.
    ```
    SELECT insurance.Insurance_no, insurance.issuing_company, insurance.type ,
    request.loan_id
    from insurance, request
    where insurance.cust_id = request.cust_id;
    ```