Banking Database Management System

BY : Sunand Agarwal - 160001057

Vishal Maurya - 160001061

**INTRODUCTION**

The Banking Management System is an application for maintaining a person's in a Bank.

This project is a model internet banking site. This site enables the customers to perform the basic banking transactions by sitting at their office or at homes through their PCs.

The main aim of this project is to provide efficient and secure net banking facilities to the banking customer over the internet.

**PROBLEM STATEMENT**

\* A Customer can have multiple accounts. An account can hold multiple customers, this situation is for a joint account.

\* Every customer is given a passbook when creating an account which is updated whenever a transaction is made.

\* Some customers may issue a cheque book too.

\* The Bank also provides a unique customer\_id to each customer which will serve as the Primary key.

\* The Address And Name Attributes are composite.

\* The phone no attribute would be multi-valued.

\* The various types of account can be savings, current, joint Or Employee.

\* The Registered Time include the date and time when the account was opened.

\* The description of the transaction made is composite, which consists of the type, which can be deposit withdrawal or money transfer; the account number of the person who sends the money and that of who receives the money (beneficiary).

\* For a particular transaction, money is either debited or credited in the account, so the other value remains null.

\* Customers must make transactions which do not result in their account balance to be less than the minimum value(here, 0).

\* For every transaction, balance is updated in the account entity.

\* The Aadhar Card and mobile no. are linked to the account of the customers, and the Aadhar No acts as the verification of the customers.

\* A customer is provided a loan only if he has an account in the bank. There must be a collateral or a Guarantor provided by the customer in order to avail the loan.

\* The amount pending keeps track of the amount still to be paid to repay the full loan and the loan interest would be applied on this amount.

\* An employee also has an account in the bank of type 'Employee'.

\* All customers have a unique card number.

**ENTITY SETS**

1. The CUSTOMER provides the following details while creating an account:

# Name -> First Name, Middle Name, Last Name

# Father's Name

# Address -> Street, City, State, PIN Code

# Gender

# DOB

# Ph. No.

# Email\_Id

# Aadhar No.

# A Customer Id will also be provided by the Bank which will be able to distinguish among customers.

2. ACCOUNT

# Account Number - Primary key

# Type

# Balance

# Status

# Cheque\_Book\_No

# Registered Time

3. TRANSACTION

# Transaction Id - Primary key

# Date

# Time

# Debit

# Credit

# Description -> Type, sender's account\_number, beneficiary's account\_number.

4. LOAN

# Loan No. - Primary key

# Type

# Period of Loan

# Interest Rate

5. BRANCH

# IFSC Code - Primary key

# Assets

# Address -> Street, City, State, PIN code

6. EMPLOYEE

# Employee\_id - Primary key

# Name -> First Name, Middle Name, Last Name

# Address -> Street, City, State, PIN code

# Gender

# DOB

# Ph. No.

# Email\_Id

# Aadhar No.

# Designation

# Salary

7. CARD\_INFO

# Card No. - Primary key

# CVV

# Valid Thru

# Card\_Holder

# Type

# PIN

# Status

**RELATIONSHIP SETS**

1. Has\_Acc Relates Account and Customer entity sets. It's attributes are :

# Customer\_id - Primary Key

# Account number

The relationship from Account to Customer is one to many. There is total participation of customers.

2. Acc\_Trans Relates Accounts and Transaction entity sets. It's attributes are :

# Transaction\_id - Primary Key

# Account number

The relationship from Account to Transaction is one to many. There is total participation of transactions.

3. is\_Emp relates branch and employee entity sets. It's attributes are :

# Employee\_id - Primary Key

# IFSC code

The relationship from Branch to Employee is one to many. There is total participation of employees.

4. Emp\_Acc relates employee and account entity sets. It's attributes are :

# Account Number - Primary Key

# Employee\_id

The relationship from Account to Employee is one to one, since each employee can have only a single account and vice-versa. There is total participation of employees.

5. Takes\_Loan relates Loan and Account entity Sets. It's attributes are :

# Descriptive attributes -

\* Loan Amount

\* Monthly Installment

\* Amount Pending

\* Collateral/Guarantor

# Account Number, Loan\_no - Primary Key

The relationship from Account to Loan is many to many.

6. Issue\_Card relates Account and Card\_Info entity sets. It's attributes are :

# Card No. - Primary Key

# Account Number

The relationship from Account to Card\_Info is one to many. There is total participation of cards.

**THE FOLLOWING IS THE SCHEMA REPRESENTATION OF THE ENTITY SETS:**

1. Customer (Customer ID, First Name, Middle Name, Last Name, S/O, Street, City, State, PIN Code, Gender, Date Of Birth, Email, Aadhar No., Account No.);
2. Cust\_Phone (Customer ID, Phone No.);
3. Account (Account No. , Type, Balance, Status, Cheque Book No, Registered Time);
4. Transaction (Transaction ID, Date, Time, Debit, Credit, Type, Sender’s Account No, Beneficiary’s Account No. , Account No.);
5. Loan ( Loan No. , Type, Interest Rate, Period of Loan);
6. Branch (IFSC Code , Assets, Street, City , State, PIN Code);
7. Employee (Employee ID , First Name, Middle Name, Last Name, Street, City, State, PIN Code, Gender, Date Of Birth, Email, Aadhar No., Designation, Salary, IFSC Code);
8. Emp\_Phone (Employee ID , Phone No.);
9. Card\_Info ( Card No. , CVV, Valid Thru, Card Holder, Type, PIN, Account No., Status);
10. Emp\_Acc (Account No. , Employee ID);
11. Takes\_Loan ( Account No. , Loan No. , Loan Amount, Monthly Installment, Pending Amount, Collateral/Guarantor).

FOREIGN KEY CONSTRAINTS

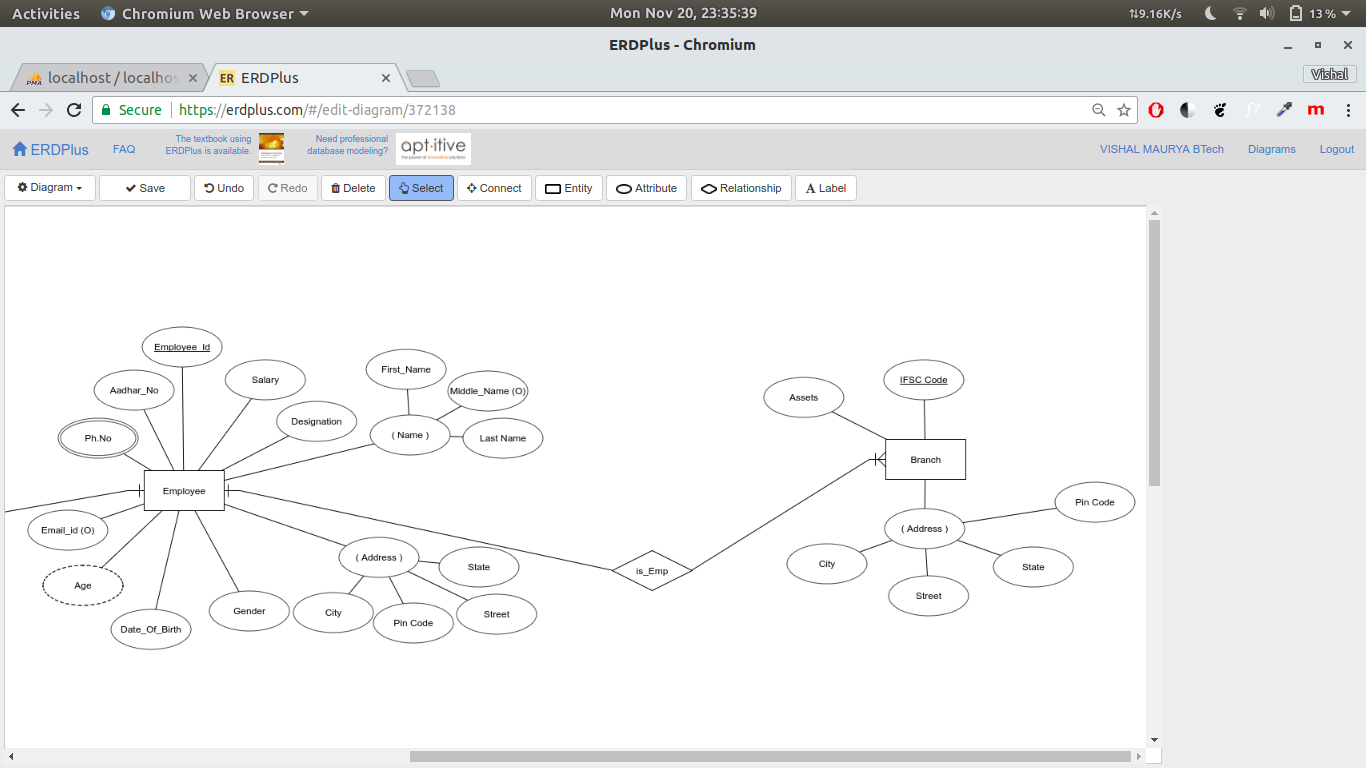
* Customer ID in Cust\_Phone references Customer.
* Account No in Transaction references Account.
* Account No in Card\_Info references Account.
* Account No in Emp\_Acc references Account.
* Account No in Takes\_Loan references Account.
* IFSC Code in Employee references Branch.
* Employee ID in Emp\_Acc references Employee.
* Employee ID in Emp\_Phone references Employee.
* Loan No in Takes\_Loan references Loan.

FUNCTIONAL DEPENDENCIES

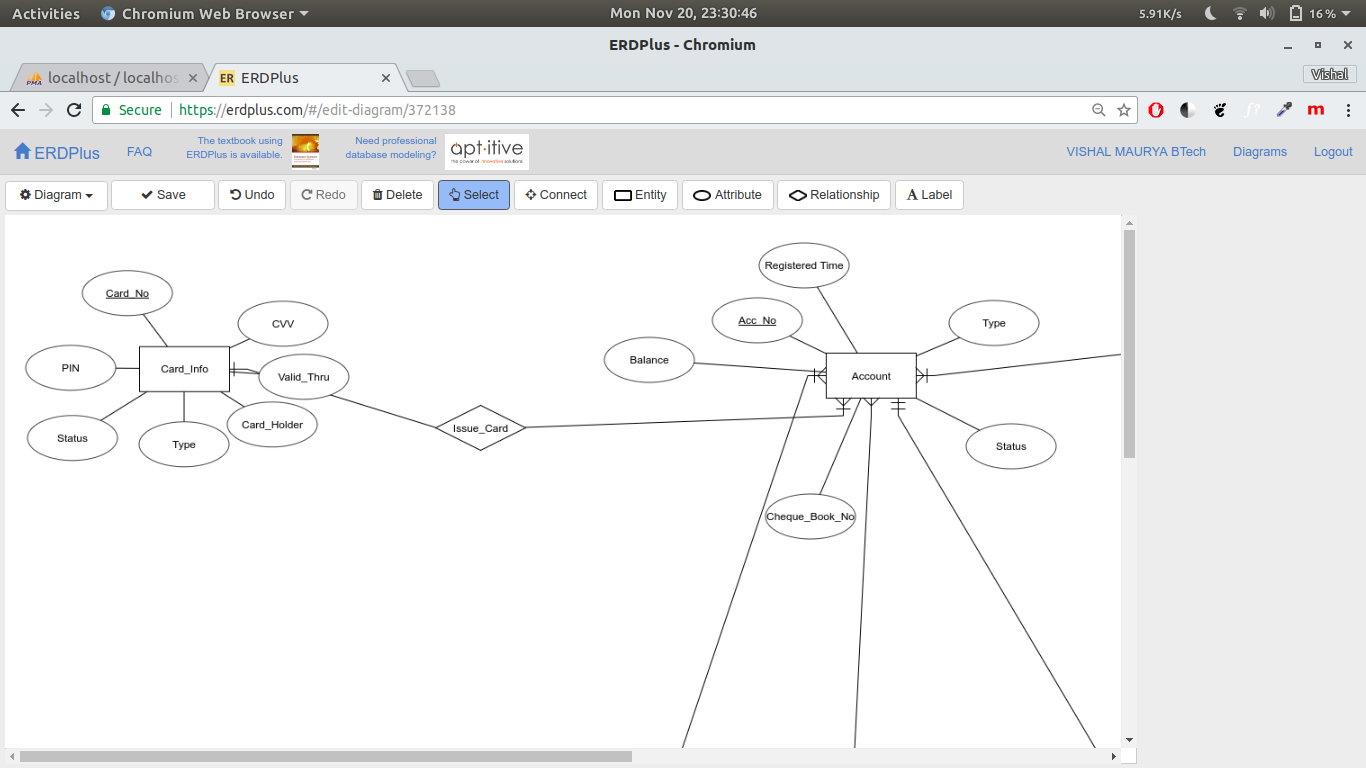
* IFSC Code determines Assets of a Branch.
* Designation of an employee determines his Salary.
* Card No determines Account No.
* Loan No determines Type of Loan and Period of loan.
* Type of Loan and period of loan determines the interest rate.

ER DIAGRAM

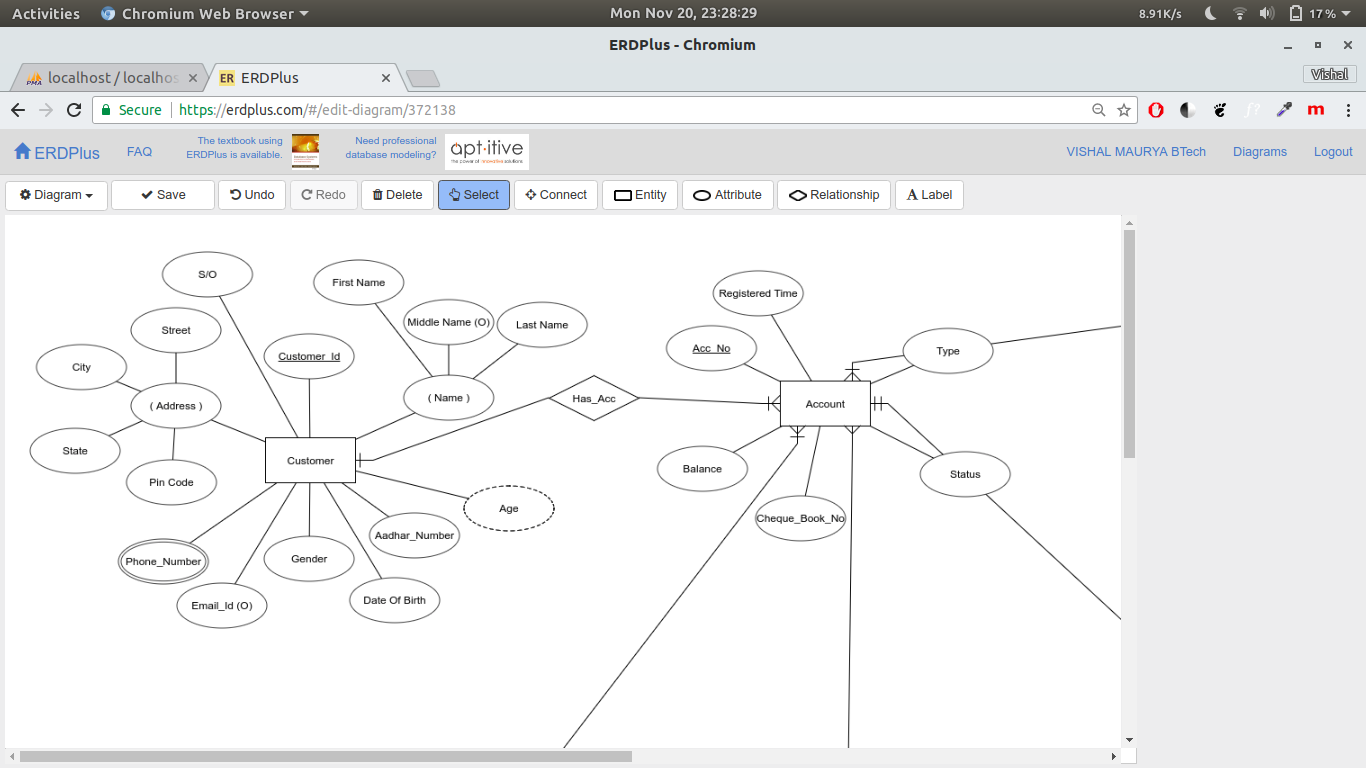
THIS IS THE RELATIONSHIP SET BETWEEN EMPLOYEE AND BRANCH



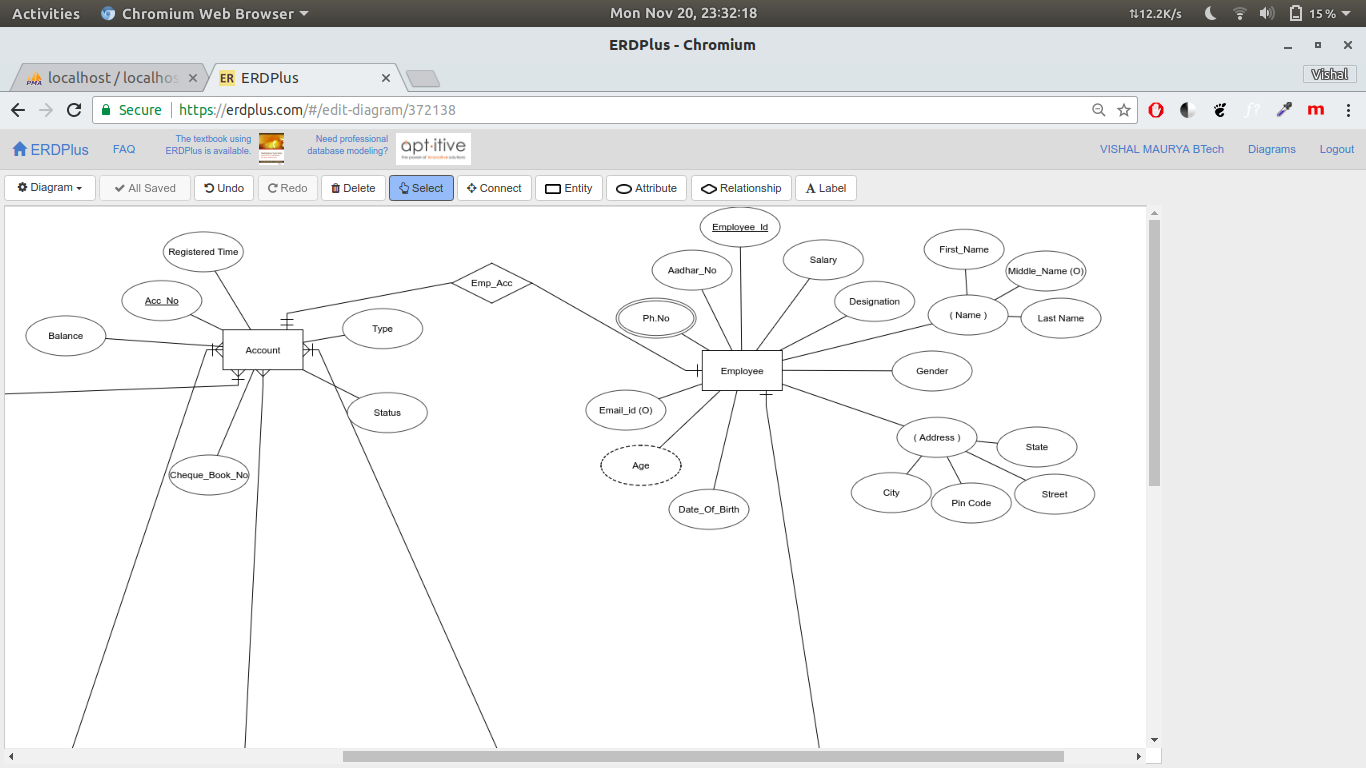
THIS IS THE RELATIONSHIP SET BETWEEN CARD\_INFO AND ACCOUNTS



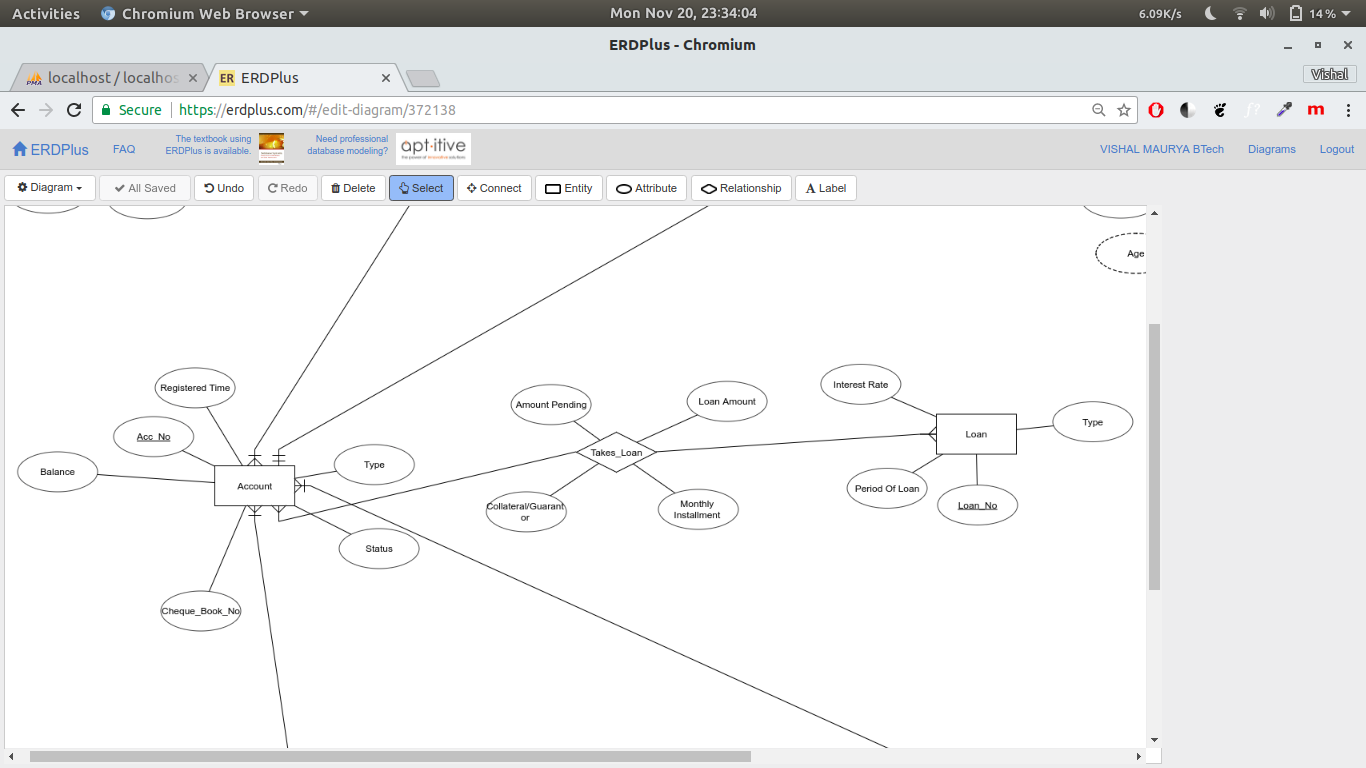
THIS IS THE RELATIONSHIP SET BETWEEN CUSTOMER AND ACCOUNT



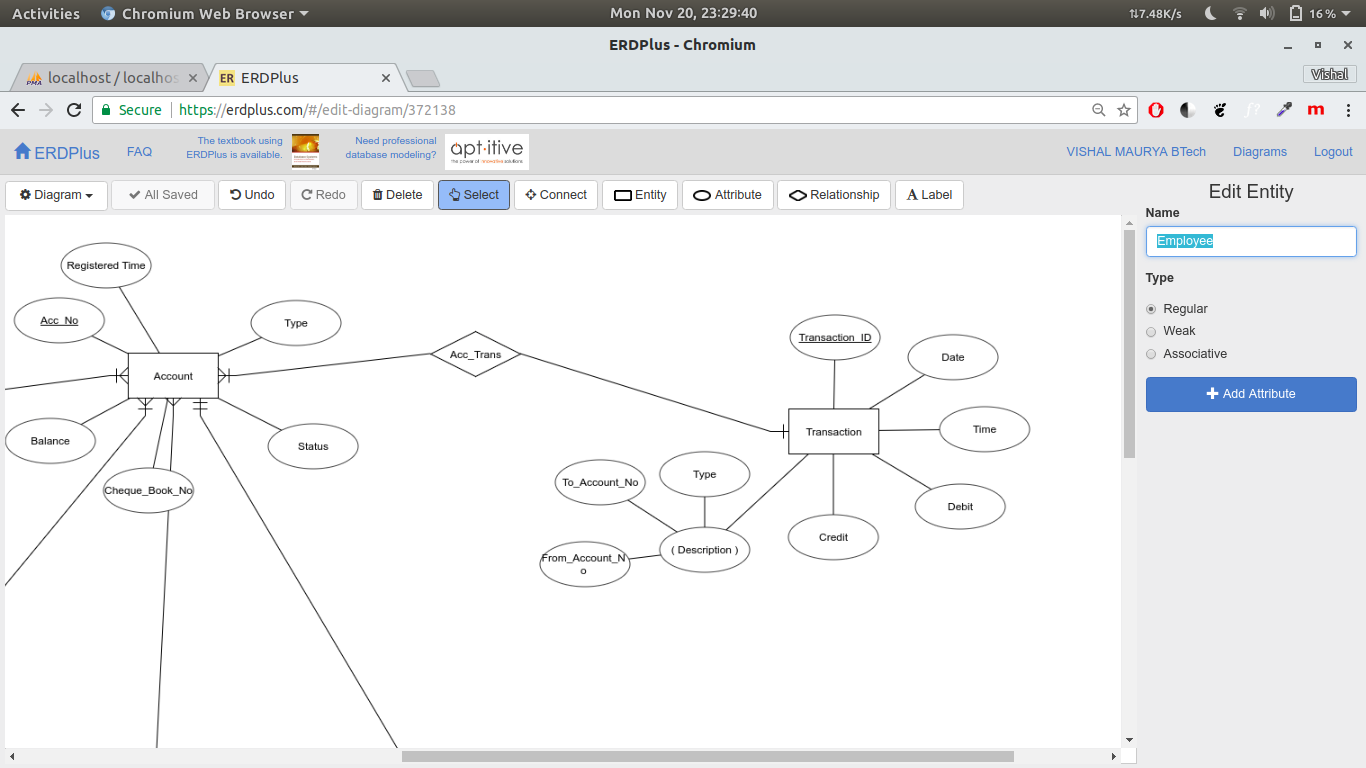
THIS IS THE RELATIONSHIP SET BETWEEN ACCOUNT AND EMPLOYEE



THIS IS THE RELATIONSHIP SET BETWEEN LOAN AND ACCOUNT



THIS IS THE RELATIONSHIP SET BETWEEN ACCOUNT AND TRANSACTION



TRANSFORMATION INTO SET OF TABLES :

CREATE TABLE accounts (

Account\_Number bigint PRIMARY KEY,

Type text NOT NULL,

Balance real NOT NULL,

Status text NOT NULL,

Cheque\_Book\_No bigint,

Registered\_time datetime

);

CREATE TABLE customers (

Customer\_ID int PRIMARY KEY AUTO\_INCREMENT,

first\_name varchar(15) NOT NULL,

middle\_name varchar(15),

last\_name varchar(15) NOT NULL,

Father\_name varchar(25) NOT NULL,

Street varchar(25),

City varchar(15),

State varchar(15),

PIN\_Code int NOT NULL,

Gender enum('M', 'F', 'O') NOT NULL,

Date\_of\_birth date NOT NULL,

Email varchar(25) NOT NULL UNIQUE,

Aadhar\_number bigint NOT NULL UNIQUE,

Account\_Number bigint NOT NULL

)

CREATE TABLE branches (

IFSC\_Code varchar(15) Primary key,

Assets real NOT NULL,

Street varchar(25),

City varchar(15) NOT NULL,

State varchar(15) NOT NULL,

PIN\_Code int NOT NULL

)

CREATE TABLE employees (

Employee\_ID int PRIMARY KEY AUTO\_INCREMENT,

first\_name varchar(15) NOT NULL,

middle\_name varchar(15),

last\_name varchar(15) NOT NULL,

Street text,

City text,

State text,

Pin\_Code int NOT NULL,

Gender enum('M','F','O') NOT NULL,

Date\_Of\_birth date NOT NULL,

Email varchar(25) NOT NULL UNIQUE,

Aadhar\_number bigint NOT NULL UNIQUE,

Designation text NOT NULL,

Salary bigint NOT NULL,

IFSC\_Code varchar(15) NOT NULL,

Foreign Key (IFSC\_Code) References branches(IFSC\_Code)

)

CREATE TABLE emp\_\_phones (

Employee\_ID int,

Phone\_No bigint,

PRIMARY KEY(Employee\_ID, Phone\_No),

Foreign Key(Employee\_ID) References employees (Employee\_ID)

)

CREATE TABLE cust\_\_phones (

Customer\_ID int,

Phone\_No bigint,

Primary Key(Customer\_ID, Phone\_No),

Foreign Key (Customer\_ID) References customers(Customer\_ID)

)

CREATE TABLE transactions

(

Transaction\_ID bigint Primary key,

Date date,

Time time,

Debit real,

Credit real,

Type text NOT NULL,

Account\_Number bigint NOT NULL,

Sender\_Acc\_No bigint,

Beneficiary\_Acc\_No bigint,

Foreign Key(Account\_Number) References accounts (Account\_Number)

)

CREATE TABLE loans

(

Loan\_No bigint Primary key,

Type text NOT NULL,

Interest\_Rate real NOT NULL,

Period text NOT NULL

)

CREATE TABLE takes\_\_loan(

Account\_Number bigint,

Loan\_No bigint,

Loan\_Amount real NOT NULL,

Monthly\_Installment real NOT NULL,

Pending\_Amount real NOT NULL,

Collateral\_Guarantor text NOT NULL,

PRIMARY KEY(Account\_Number, Loan\_No),

Foreign Key(Account\_Number) References accounts (Account\_Number),

Foreign Key(Loan\_No) References loans (Loan\_No)

)

CREATE TABLE card\_\_info (

Card\_No bigint NOT NULL PRIMARY KEY,

CVV varchar(255) NOT NULL,

Valid\_Thru varchar(8) NOT NULL,

Card\_Holder text,

Type text NOT NULL,

PIN varchar(255) NOT NULL,

Account\_Number bigint NOT NULL,

Status text NOT NULL,

Foreign Key(Account\_Number) References accounts (Account\_Number)

)

CREATE TABLE emp\_\_acc (

Account\_Number bigint PRIMARY KEY,

Employee\_ID int,

Foreign Key(Account\_Number) References accounts (Account\_Number),

Foreign Key(Employee\_ID) References employees (Employee\_ID)

)

TRIGGERS

TO UPDATE THE BALANCE AFTER REPAYING THE LOAN.

*CREATE TRIGGER updateAmtLoan*

*AFTER UPDATE ON takes\_\_loan*

*FOR EACH ROW*

*BEGIN*

*UPDATE accounts SET balance = balance - (old.Pending\_Amount - new.Pending\_Amount)*

*WHERE Account\_Number = old.Account\_Number*

*AND*

*(balance - (old.Pending\_Amount - new.Pending\_Amount)) >= 0;*

*END*

TO UPDATE THE BALANCE AFTER TAKING THE LOAN

*CREATE TRIGGER insertAmtLoan*

*AFTER INSERT ON takes\_\_loan*

*FOR EACH ROW*

*BEGIN*

*UPDATE accounts SET balance = balance + new.Loan\_Amount;*

*END*

FEATURES AND THE FUNCTIONS OF THE WEBSITES

