***ER DIAGRAM OF A BANKING SYSTEM***

**STEP: 01 DEFINING ENTITY SETS AND THEIR TYPES**

1. Branch
2. Customer
3. Employee
4. Account : 1)Current account 2)Savings account
5. Loan
6. Payment ( Weak entity--total participation depends on strong entity)

**STEP: 02 DEFINING ATTRIBUTES OF EACH ENTITY AND THEIR TYPE**

1. **Branch** : B-name(composite), B-id (primary-key), B-address(composite), B-Asset(single-valued), B-liability(single-valued)
2. **Customer** : C-name (Composite), C-id (primary-key), C-address(composite),

C-contact number (single-valued) ,DOB(single-valued),C- Age(derived)

1. **Employee :** E-name (Composite), E-id (primary-key), E-address(composite),

E-contact number (single-valued), DOB(single-valued),

Age (derived), Department (single-valued),E-Post(single-valued)

**Customer—Employee—Generalize (Person—related—to—bank)**

1. **Savings account** : Account-number (primary-key), Balance(single-valued),interest-rate (single-valued), Daily-withdrawal limit(single-valued)
2. **Current account** : Account-number (primary-key), Balance(single-valued),Pertransaction\_charge(Single-valued),Overdraft-amount(Single-valued) **Savings account-- Current account—Generalize**
3. **Loan :** Loan-ID (Primary-key),loan-amount(single-valued)
4. **Payment (Weak-Entity**) : Payment-Number(Primary-key), amount,date

**STEP: 03 DEFINING RELATIONSHIP, THEIR TYPE AND MAPPING CARDIANILITY**

1. Bank has branches.(1:N)
2. Employee works in branches (N:1).
3. Customer several accounts.(1:N)
4. Customer takes loan(1;N)
5. Loan has payment(1:N)
6. Employee manages employee(1:N)
7. Employee manages customer.(1:N)

**BRANCH**

**RELATED**

**MANAGES**

**IS A**

**CUSTOMER**

**EMPLOYEE**

**PERSON RELATED TO BANK**

HAS

ACCOUNT

TAKES

HAS

**IS A**

**CURRENT-ACCOUNT**

SAVINGS-

ACCOUNT

**PAYMENT**

**LOAN**

**TRANSFORMATION OF ER MODEL TO THE RELATIONAL MODEL CODE**

Step 01: Creation of all the tables using ddl (create)

create database banking\_system;

use banking\_system;

#creation of tables

create table branch (

branch\_name varchar(255) unique not null,

branch\_id int primary key,

branch\_address\_statename varchar(255) unique not null,

branch\_address\_pincode varchar(255) unique not null,

branch\_asset bigint not null,

branch\_liability bigint not null

);

##alter:add/modify/rename

alter table branch add branch\_address\_cityname varchar(255) not null;

select \* from branch;

select version();

CREATE TABLE per\_rel\_to\_bank (

fname VARCHAR(255) NOT NULL,

mname VARCHAR(255),

lname VARCHAR(255),

address VARCHAR(500) NOT NULL,

dob DATE NOT NULL,

age int not null

);

select \* from per\_rel\_to\_bank;

create table employee(

e\_id int primary key,

e\_post varchar(255) not null,

e\_department varchar(255) not null

);

select \* from employee;

create table customer(

c\_id int primary key);

select \* from customer;

create table bank\_accounts(

account\_number int primary key,

account\_balance int not null

);

create table savings\_account(

interest\_rate int default 6.25,

daily\_withdrawl\_limit int not null

);

create table current\_account(

per\_transaction\_ch int default 10,

overdraft\_amt int

);

create table loan (

loan\_id int primary key,

loan\_amount int

);

create table payment (

payment\_number int primary key auto\_increment,

pay\_amount int not null

);

#STEP 02 : CREATION OF RELATIONSHIP AMONG THEM

# A) all person\_rel\_to\_bank is connected to a specific branch

alter table per\_rel\_to\_bank add bank\_id int not null;

alter table per\_rel\_to\_bank add constraint foreign key(bank\_id) references branch(branch\_id) on delete cascade;

select \* from per\_rel\_to\_bank;

##B)person related to bank relate

alter table per\_rel\_to\_bank add person\_id int primary key auto\_increment;

select \* from per\_rel\_to\_bank;

alter table employee add column p\_id int;

alter table customer add column p\_id int;

alter table employee add constraint foreign key(p\_id) references per\_rel\_to\_bank(person\_id) on delete cascade;

alter table customer add constraint foreign key(p\_id) references per\_rel\_to\_bank(person\_id) on delete cascade;

select \* from employee;

select \* from customer;

##c)account generalization handle

alter table savings\_account add column abtacc int;

alter table current\_account add column abtacc int;

alter table savings\_account add constraint foreign key(abtacc) references bank\_accounts(account\_number) on delete cascade;

alter table current\_account add constraint foreign key(abtacc) references bank\_accounts(account\_number) on delete cascade;

select \* from savings\_account;

select \* from current\_account;

##d))loan has payment

alter table payment add column l\_id int;

alter table payment add constraint foreign key(l\_id) references loan(loan\_id) on delete cascade;

select \* from payment;

#e) customer takes loan

alter table loan add column infocust\_id int;

alter table loan add constraint foreign key(infocust\_id) references customer(c\_id) on delete cascade;

select \* from loan;

#f) customer has accounts

alter table bank\_accounts add column abtcust int;

alter table bank\_accounts add constraint foreign key(abtcust) references customer(c\_id) on delete cascade;

select \* from bank\_accounts;

#g)employee manages employee

alter table employee add manager\_id int not null;

select \* from employee;