Banking Database

Submitted By:

Vamsheeth Vadlamudi(U101115FCS274)

Submitted to
Amit Kumar Sir (DBMS CS231)

Table of Contents:

Topic *Page Numbers*

Introduction:

The Domain “Banking System " keeps the day by day tally record as a complete banking. The database Banking is constructed to minimize redundancy. It has been elaborated on further. The aim was to make the existing bank system to more easy one. In the system, the transactions are done only manually but in proposed system we have to computerize all the banking transaction using the software Banking System using this database.

We are developing an instance of the back end in order to gain greater understanding and clear our fundamental concepts of database design such as modelling and normalization. Banking Database Provides information about basic requirements Account type, account opening form, Deposit, withdrawal, and Searching the transaction, Transaction report, Individual account, opening form, Group Account etc., This database is managed in such a way that respective category will be shown respective needs in certain purpose.

Problem Statement:

Usually all persons want money for personal and commercial purposes. Banks are the oldest lending institutions in Indian scenario. They are providing all facilities to all citizens for their own purposes by their terms. To survive in this modern market every bank implements so many new innovative ideas, strategies and advanced technologies. For that they give each and every minute detail about their institution and projects to Public.

They are providing ample facilities to satisfy their customers i.e. Net Banking, Mobile Banking, Door to Door facility, Instant facility, Investment facility, Demat facility, Credit Card facility, Loans and Advances, Account facility etc. And such banks get success to create their own image in public and corporate world. These banks always accept innovative notions in Indian banking scenario like Credit Cards, ATM machines, Risk Management etc. So, as a student business economics I take keen interest in Indian economy and for that banks are the main source of development. So, this must be the first choice for me to select this topic. At this stage, every person must know about new innovation, technology of procedure new schemes and new ventures. Because of the following reasons, I prefer this project work to get the knowledge of the banking system.

Banking is an essential industry. It is where we often wind up when we are seeking a problem in financial crisis and related query. Banking is one of the most regulated business in the world. Banks remain important source for carrier opportunities for people. It is vital

system for developing economy for the nation. Banks can play a dynamic role in delivering and purchase of consumer durables.

Functioning of a Bank:

Functioning of a Bank is among the more complicated of corporate operations. Since Banking involves dealing directly with money, governments in most countries regulate this sector rather stringently. In India, the regulation traditionally has been very strict and in the opinion of certain quarters, responsible for the present condition of banks, where NPAs are of a very high order. The process of financial reforms, which started in 1991, has cleared the cobwebs somewhat but a lot remains to be done. The multiplicity of policy and regulations that a Bank has to work with makes its operations even more complicated, sometimes bordering on illogical. This section, which is also intended for banking professional, attempts to give an overview of the functions in as simple manner as possible. Banking Regulation Act of India, 1949 defines Banking as “accepting, for the purpose of lending or investment of deposits of money from the public, repayable on demand or otherwise and withdraw able by cheques, draft, and order or otherwise."

Rules Governing the Project:

- All the customers of the bank have a **unique** account number.
- The account numbers are not nullable i.e., they cannot take **null** values.
- The customers must have a **minimum** account balance fixed by the respective bank.
- Any customer is not all owed to withdraw amount from his account, if the withdrawal results in his account balance going below the minimum balance.
- A person is eligible to get a loan from the bank if he has an account in the bank.
- The percentage of interest imposed on the loan depends on the Company’s policies.
- The customers are issued cards (ATM or Debit) depending on his/her eligibilities.
- All the card holders have **unique** card and PIN numbers.
- The employees of the bank have **unique** identification numbers

Entities Taken for ER

1. Branch (branch_name, branch-id, branch_city, assets)

Branch_id	Branch_name	Branch_city	assets

2. Customer (customer-id, customer_name, customer_street, customer_city, DOB, account_no, phone_no.)

Customer_id	customer_name	customer_street	customer_city	DOB	Phone_no.	account_no

3. Account (account-no, branch_id, balance)

Account_no	branch_id	balance

4. Loan (loan-number, branch_id, loan_type, duration, amount)

Loan_number	branch_id	loan_type	duration	amount

5. Employee (employee_name, employee-id, phone_no, salary, designation, start_date, tenure)

employee_name	Employee_id	tenure	salary	designation	Start_date	Phone_no..

6. Payment (payment-no., payment _date, Payment_ amount)

Payment_no.	Payment_date	amount

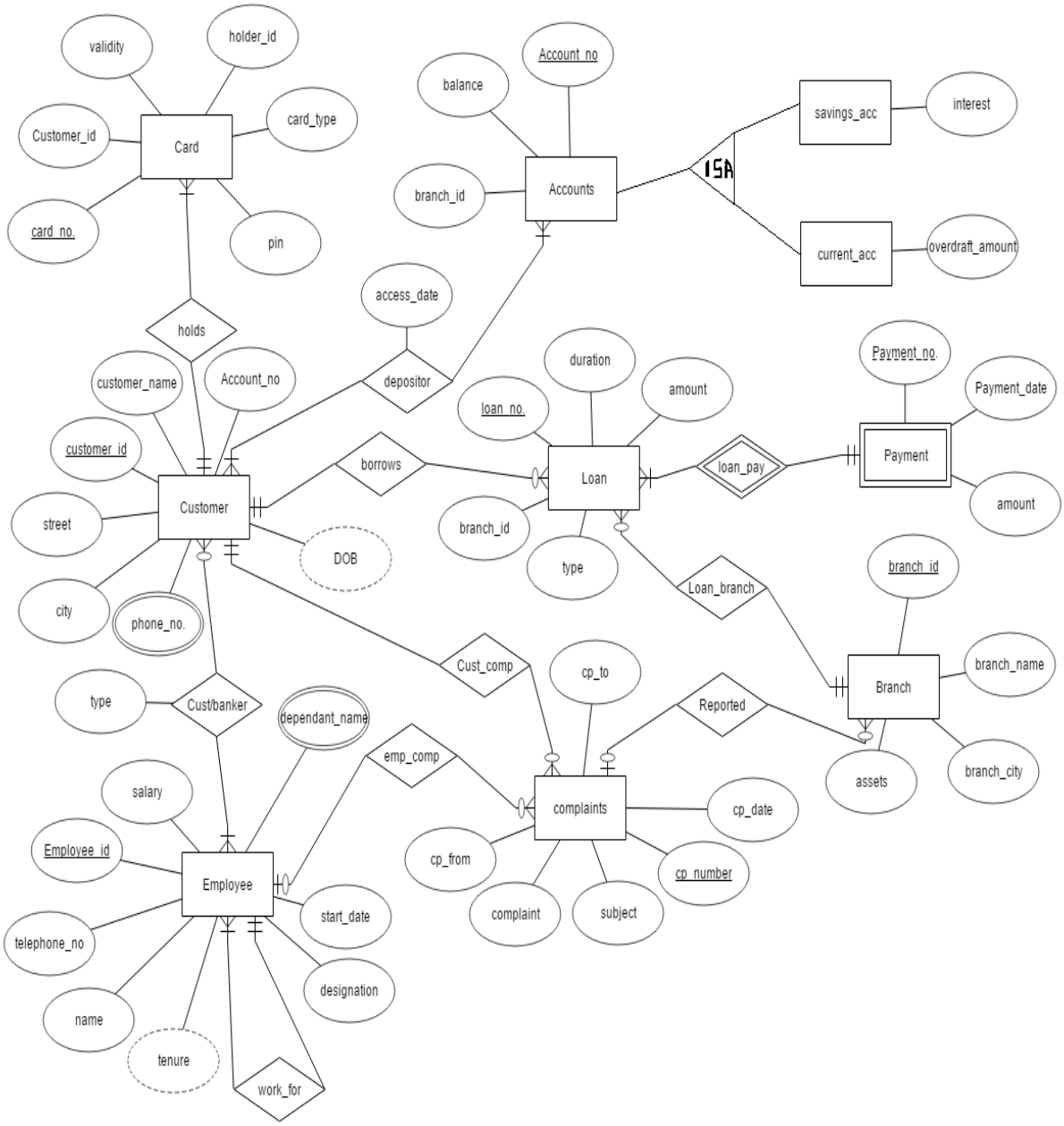
7. Complaints (cp-number, cp_to, cp_from, subject, complaint, cp_date)

Cp_number	cp_to	cp_from	subject	complaint	cp_date

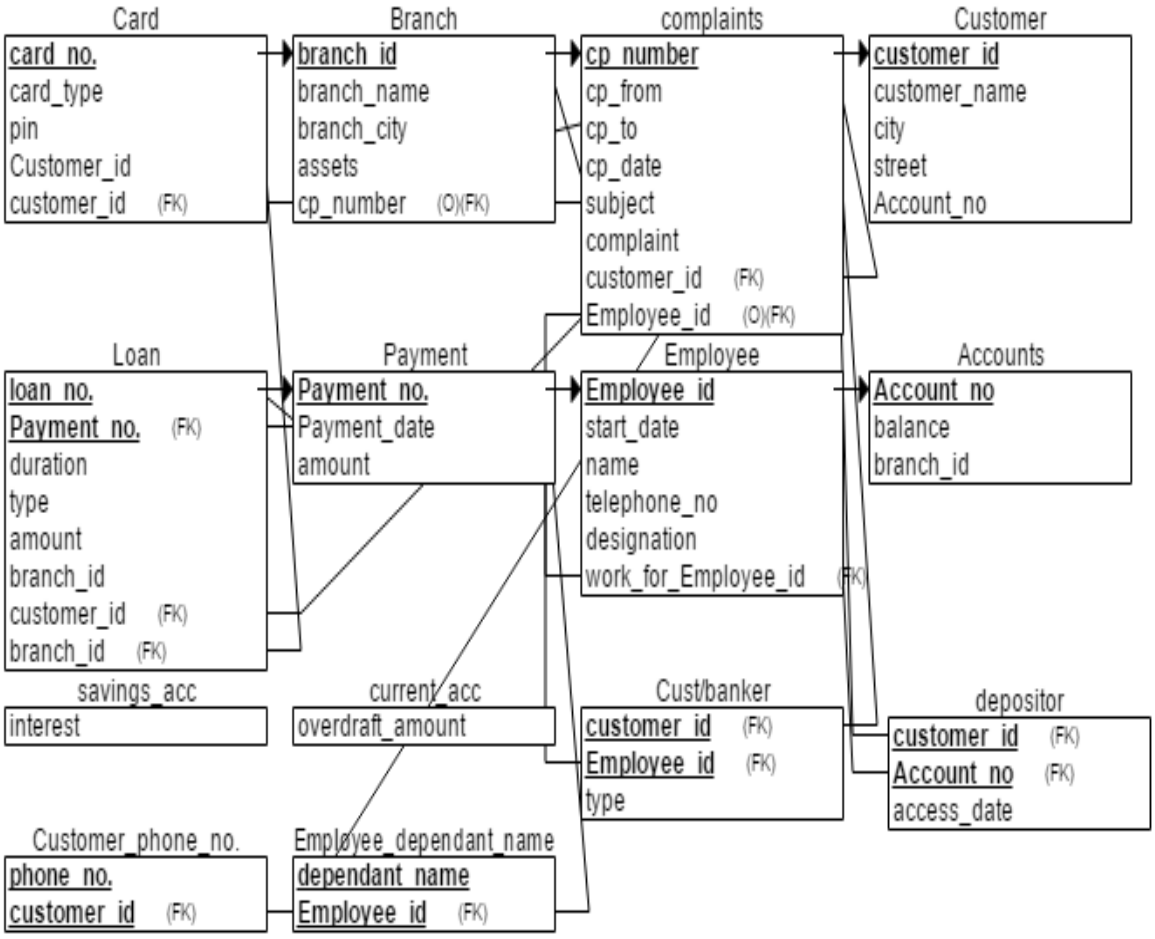
8. card-holder (holder-id, card_number, validity, encrypted_pin,card_type,Customer_id)

holder_id	card_number	validity	encrypted_pin	Card_type	Customer_id

ER Diagram


validity
holder_id
Customer_id
Card
card_type
card_no.
pin
Account_no
balance
branch_id
Accounts
ISA
savings_acc
interest
current_acc
overdraft_amount
access_date
holds
depositor
customer_name
Account_no
customer_id
street
Customer
city
phone_no.
DOB
borrows
duration
amount
loan_no.
Loan
branch_id
type
loan_pay
Payment_no.
Payment_date
Payment
amount
branch_id
Loan_branch
Branch
branch_name
branch_city
assets
Reported
Cust_comp
cp_to
type
Cust/banker
dependant_name
emp_comp
complaints
cp_date
cp_number
cp_from
complaint
subject
salary
Employee_id
Employee
start_date
telephone_no
designation
name
tenure
work_for

Relational Schema:



My SQL Executable Code for Tables:

```
CREATE TABLE Customer
(
  customer_id INT NOT NULL,
  customer_name INT NOT NULL,
  city INT NOT NULL,
```

```
  street INT NOT NULL,
  Account_no INT NOT NULL,
  PRIMARY KEY (customer_id)
);

CREATE TABLE Payment
(
  Payment_no. INT NOT NULL,
  Payment_date INT NOT NULL,
  amount INT NOT NULL,
  PRIMARY KEY (Payment_no.)
);

CREATE TABLE Accounts
(
  Account_no INT NOT NULL,
  balance INT NOT NULL,
  branch_id INT NOT NULL,
  PRIMARY KEY (Account_no)
);

CREATE TABLE savings_acc
(
  interest INT NOT NULL
);

CREATE TABLE current_acc
(
  overdraft_amount INT NOT NULL
```

```
);

CREATE TABLE depositor
(
  access_date INT NOT NULL,
  customer_id INT NOT NULL,
  Account_no INT NOT NULL,
  PRIMARY KEY (customer_id, Account_no),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (Account_no) REFERENCES Accounts(Account_no)
);

CREATE TABLE Customer_phone_no.
(
  phone_no. INT NOT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (phone_no., customer_id),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);

CREATE TABLE Card
(
  card_type INT NOT NULL,
  pin INT NOT NULL,
  card_no. INT NOT NULL,
  Customer_id INT NOT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (card_no.),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
```

```
);

CREATE TABLE Branch
(
  branch_id INT NOT NULL,
  branch_name INT NOT NULL,
  branch_city INT NOT NULL,
  assets INT NOT NULL,
  cp_number INT,
  PRIMARY KEY (branch_id),
  FOREIGN KEY (cp_number) REFERENCES complaints(cp_number)
);

CREATE TABLE complaints
(
  cp_from INT NOT NULL,
  cp_to INT NOT NULL,
  cp_date INT NOT NULL,
  cp_number INT NOT NULL,
  subject INT NOT NULL,
  complaint INT NOT NULL,
  customer_id INT NOT NULL,
  Employee_id INT,
  PRIMARY KEY (cp_number),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (Employee_id) REFERENCES Employee(Employee_id)
);

CREATE TABLE Loan
```

```
(
  loan_no. INT NOT NULL,
  duration INT NOT NULL,
  type INT NOT NULL,
  amount INT NOT NULL,
  branch_id INT NOT NULL,
  Payment_no. INT NOT NULL,
  customer_id INT NOT NULL,
  branch_id INT NOT NULL,
  PRIMARY KEY (loan_no., Payment_no.),
  FOREIGN KEY (Payment_no.) REFERENCES Payment(Payment_no.),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (branch_id) REFERENCES Branch(branch_id)
);

CREATE TABLE Employee
(
  Employee_id INT NOT NULL,
  start_date INT NOT NULL,
  name INT NOT NULL,
  telephone_no INT NOT NULL,
  designation INT NOT NULL,
  work_for_Employee_id INT NOT NULL,
  PRIMARY KEY (Employee_id),
  FOREIGN KEY (work_for_Employee_id) REFERENCES Employee(Employee_id)
);

CREATE TABLE Cust/banker
(
```

```
 type INT NOT NULL,
 customer_id INT NOT NULL,
 Employee_id INT NOT NULL,
 PRIMARY KEY (customer_id, Employee_id),
 FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
 FOREIGN KEY (Employee_id) REFERENCES Employee(Employee_id)
);

CREATE TABLE Employee_dependant_name
(
 dependant_name INT NOT NULL,
 Employee_id INT NOT NULL,
 PRIMARY KEY (dependant_name, Employee_id),
 FOREIGN KEY (Employee_id) REFERENCES Employee(Employee_id)
);
```

Conversion table:

Entities	Relations and Description	MySQL Executable Code
Branch	Here the Name,Assets,City where it is located and the unique Branch Id is shown here **Relations:** Branch to Complaints: *One to Many* Branch to Loan: *Many to one*	CREATE TABLE Branch (Branch_City VARCHAR NOT NULL, Assets INT NOT NULL, Branch_Name VARCHAR NOT NULL, Branch_Id VARCHAR NOT NULL, PRIMARY KEY (Branch_Id));
Employee	It shows the Details of Employee Id, employee name, telephone no., work	CREATE TABLE Employee (

	starting date, Salary details and for whom/where/what he is working for etc., **Employee ID acts as Primary Key** **Relations:** Employee to Customers: *Many to Many* Employee to Complaints: *Many to One* Employee to Employee: Many to One/One to Many	Employee_Id VARCHAR NOT NULL, Employee_Name VARCHAR NOT NULL, Telephone_No INT NOT NULL, Start_Date DATETIME NOT NULL, Tenure INT NOT NULL, Salary INT NOT NULL, Designation VARCHAR NOT NULL PRIMARY KEY (Employee_Id),);
Customer	Here we see the complete details of the customer like Unique customer id, DOB, Name of the customer, Account number of the customer, and complete address includes street, city, and country **Account number acts as Foreign Key from customer table** **Relations:** Customer to card: *Many to One* Customer to Employee: *Many to Many* Customer to Account: *Many to Many* Customer to Loan: *Many to One* Customer to Complaints: *Many to One*	CREATE TABLE Customer (Customer_Street VARCHAR NOT NULL, Customer_City VARCHAR NOT NULL, Phone Number INT NOT NULL, Customer_Id VARCHAR NOT NULL, DOB DATE NOT NULL, Customer_Name VARCHAR NOT NULL, Account_Number VARCHAR NOT NULL, PRIMARY KEY (Customer_Id), FOREIGN KEY (Account_Number) REFERENCES Account(Account_Number));
Accounts	Here we see the customers unique Account Number, Balance Amount, Id of that branch where the account exists **Relations:** Accounts to Customers: *Many to Many Relationship*	CREATE TABLE Account (Account_Number VARCHAR NOT NULL, Balance INT NOT NULL, Branch_Id VARCHAR NOT NULL, PRIMARY KEY (Account_Number),

	Account is in ISA relationship with current and savings account	Foreign Key(Branch_Id), UNIQUE ());
Loan	In this table we see the unique Loan Number, Loan Amount, Loan type, Duration of the Loan, Payment Number, Branch id where loan is issued. **Payment Number and Branch id acts as Foreign Key from payment table & branch table** **Along with Payment Number, Loan Number acts as Primary key** Relations: Loan to Customer: *One to Many* Loan to Payment: *One to Many* Loan to Branch: *One to Many*	CREATE TABLE Loan (Loan_Number INT NOT NULL, Amount INT NOT NULL, Loan_Type VARCHAR NOT NULL, Loan_Duration FLOAT NOT NULL, Payment_Number INT NOT NULL, Branch_Id VARCHAR NOT NULL, PRIMARY KEY (Loan_Number, Payment_Number), FOREIGN KEY (Payment_Number) REFERENCES Payment(Payment_Number), FOREIGN KEY (Branch_Id) REFERENCES Branch(Branch_Id));
Complaints	Here we can find the Complaints from the customers ,to where the complaints filed subject of the complaint, brief description, complaint date and time and unique complaint number. **Relations:** Complaints to branch: *Many to one* Complaints to customers: *One to many* Complaints to Employee: *One to many relationship*	CREATE TABLE Complaints (Cp_From VARCHAR NOT NULL, Subject VARCHAR NOT NULL, Cp_to VARCHAR NOT NULL, Complaint VARCHAR NOT NULL, Cp_Date DATETIME NOT NULL, Cp_Number VARCHAR NOT NULL, PRIMARY KEY (Cp_Number));
Card	In this table we can see the details of the card like the	CREATE TABLE Card (

	customer id, number on the card, Validy of the card, Pin of the card, card type, customer id. **Here Card Number acts as Primary Key** **And customer id acts as foreign key from customer table** **Relations:** Card to Customer: *One to Many*	Customer_Id VARCHAR NOT NULL, Card_Number INT NOT NULL, Validity DATE NOT NULL, Pin INT NOT NULL, Card_Type VARCHAR NOT NULL, Holder_name VARCHAR NOT NULL, PRIMARY KEY (Card number), FOREIGN KEY (Customer_Id) REFERENCES Customer(Customer_Id));
Payment	Here it represents the Payment Date &Time, Payment Amount, And Unique Payment Number. **Relations:** Payment to Loan: *Many to One*	CREATE TABLE Payment (Payment_Date DATETIME NOT NULL, Payment_Amount INT NOT NULL, Payment_Number INT NOT NULL, PRIMARY KEY (Payment_Number));

Normalization:

Normalisation is a technique of organizing the data in the database. It is a systematic approach

of decomposing tables used for mainly two purposes -

- Eliminating redundant (useless) data and undesirable characteristics like Insertion, Update and Deletion Anomalies.
- Ensuring data dependencies make sense i.e., data is logically stored.

It is a multi-step process that puts data into tabular form by removing duplicated data from the
relation tables. Without Normalization, it becomes difficult to handle and update the database,
without facing data loss.

Our goals of database design with functional dependencies are:

1. 4NF
2. Losslessness
3. Dependency preserving

Table 1: Branch (Branch_name, assets,Branch_city,Branch_id)

Branch_id is the primary key for the table. Hence it determines all attributes in the table

Attributes	Type	Key
branch_city	varchar (20)	
assets	int (100)	
branch_name	varchar (20)	
branch_id	varchar (10)	Primary

including itself.

Functional Dependencies:

1. Branch_name ->Branch_city

Since Branch_name is a non-key attribute and it is creating a transitvity we will split the table into two
[Branch_name,Branch_city] where Branch_name is the key.
[Branch_id,Branch_name,assets] where branch_id is the key.

Since every non-key attribute is dependent on a key the table is in BCNF.

ScreenShots:

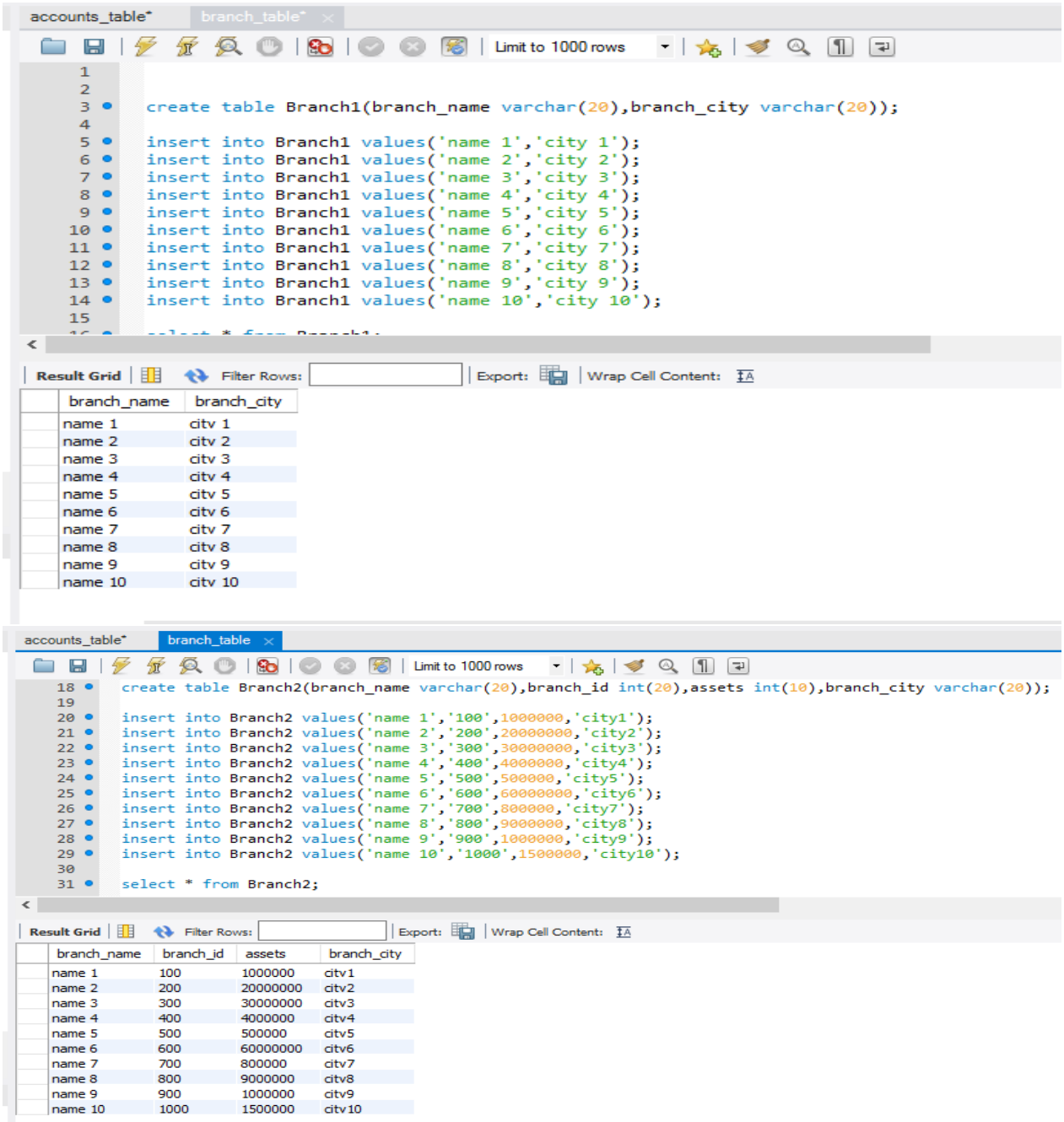


Table 2: Customer (Customer_id, Customer_name, Account_no, city, street, phone_no , age, dob)

Attributes	Type	Key
Customer_street	Varchar(20)	
Customer_city	Varchar(10)	
country	Varchar(10)	
Customer_id	Varchar(10)	Primary

DOB/Age	Date/int(5)	
Customer_name	Varchar(20)	
Account_number	int(100)	Foreign
Phone No.	Int(15)	

Customer_id is the primary key for the table. Hence it determines all attributes in the table including itself.

Account_no is the foreign key in the table

FD's

Customer_id determines all other attributes.

The table has no non_key attributes determining any other attributes hence it doesn't have any transitivity. So the table is in 3NF and is in BCNF as Customer_id is the superkey and is in the left side of all FD's

ScreenShots:

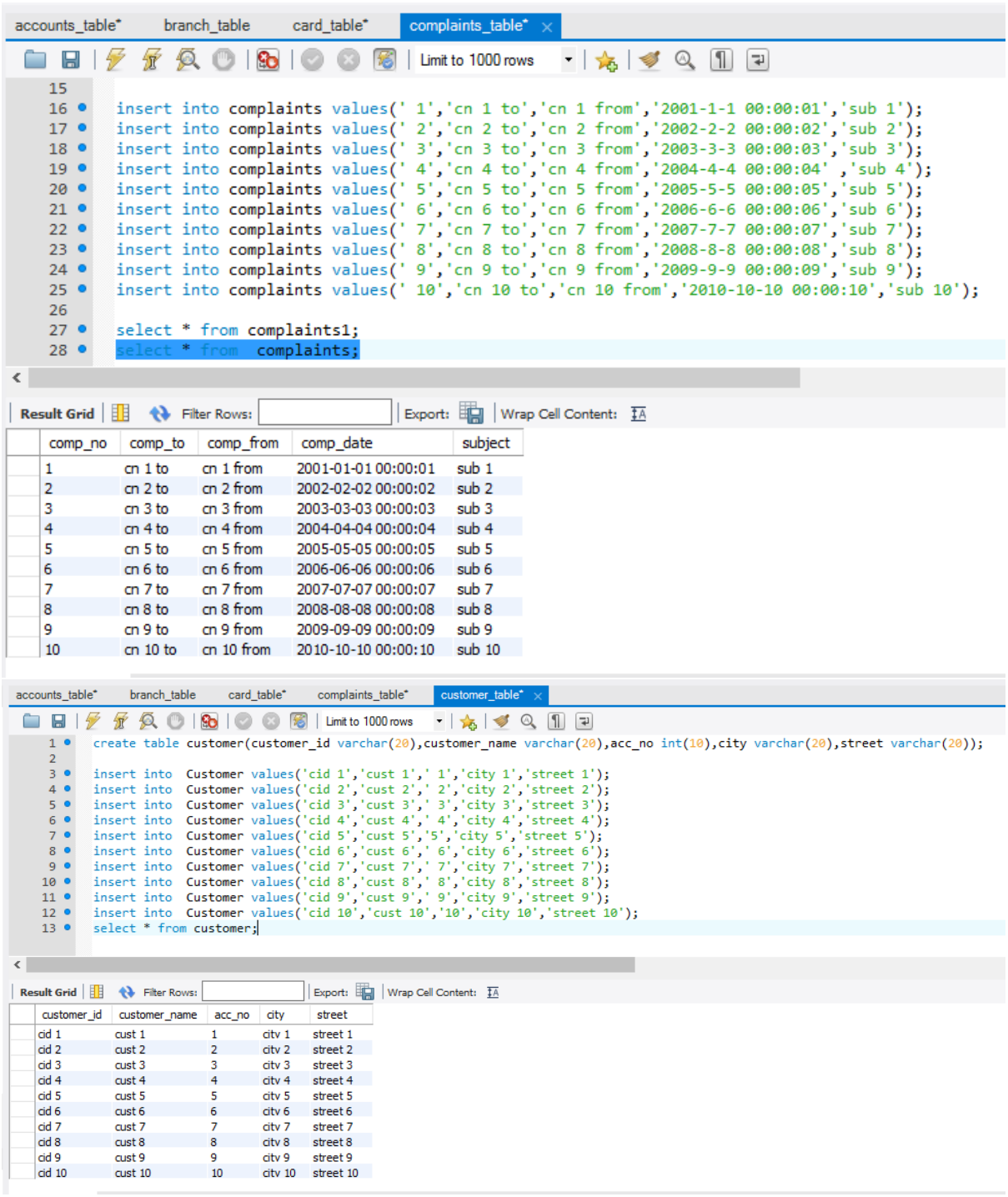


Table 3 :Account(Account_no,branch_id,balance)

Attributes	Type	Key
account_number	int(100)	Primary
balance_amount	int(100)	
branch_id	varchar(10)	

branch_name	varchar(20)	

Account_no is the primary key.

Branch_id is the foreign key.

FD's

The table has no non_key attributes determining any other attributes hence it doesn't have any transitivity. So the table is in 3NF and is in BCNF as Account_id is the superkey and is in the left side of all FD's.

ScreenShots:

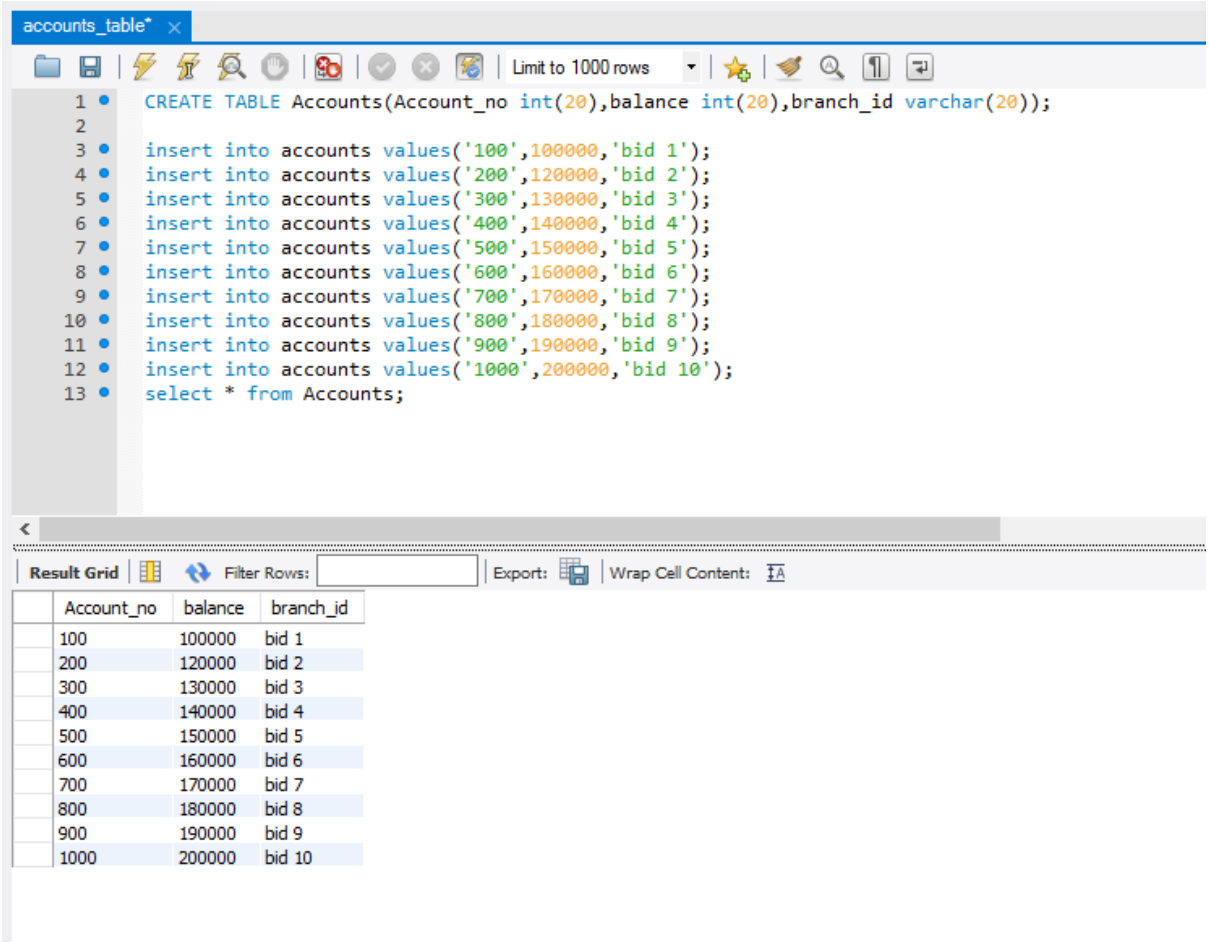


Table 4: Loan(Loan_no., Loan_type, Branch_id, duration, amount)

Attributes	Type	Key
Loan_number	Int(20)	Primary
Amount	Int(100)	

Loan_type	Varchar(20)	
Loan_duration	Int(20)	
Payment_number	Int(20)	Foreign
Branch_id	Varchar(10)	Foreign

Loan_no. is the primary key.

Branch_id is the foreign key.

FD's

The table has no non_key attributes determining any other attributes hence it doesn't have any transitivity. So the table is in 3NF and is in BCNF as Loan_number is the superkey and is in the left side of all FD's.

ScreenShots:

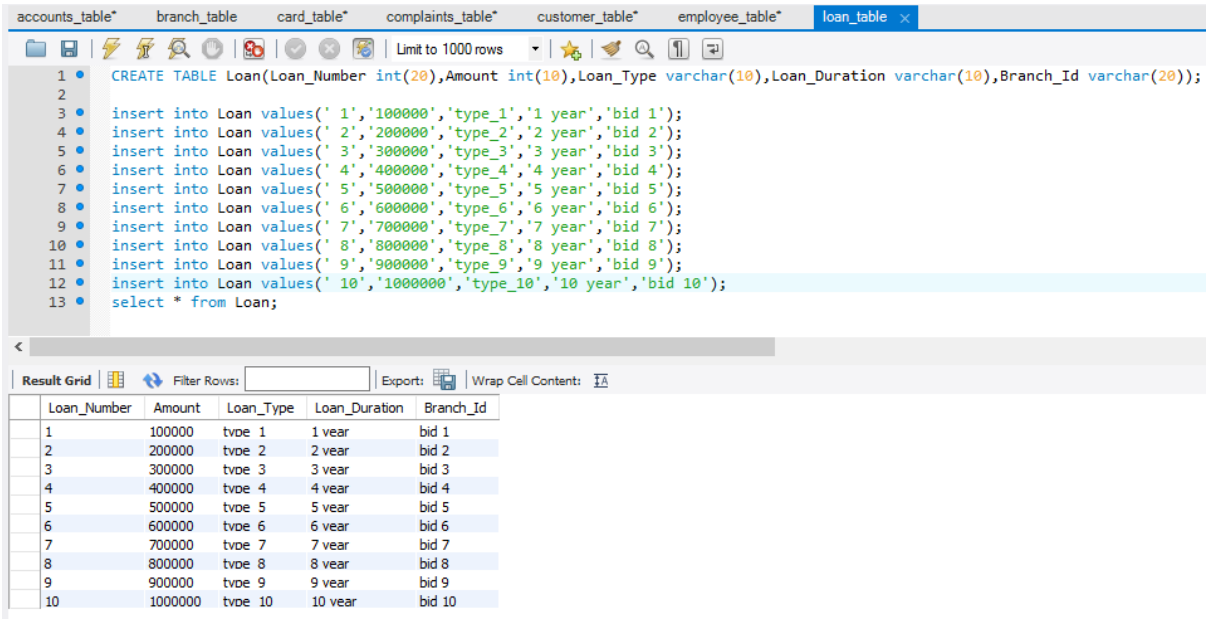


Table 5 :Complaints (cp_from, cp_to, cp_no.,cp_date, complaint, subject)

Attributes	Type	Key
Cp_from	Varchar(50)	
Cp_to	Varchar(50)	
Complaint	Varchar(100)	
Subject	Varchar(10)	
Cp_date	Date	
Cp_number	Int(10)	Primary

Cp_no. is the primary key.

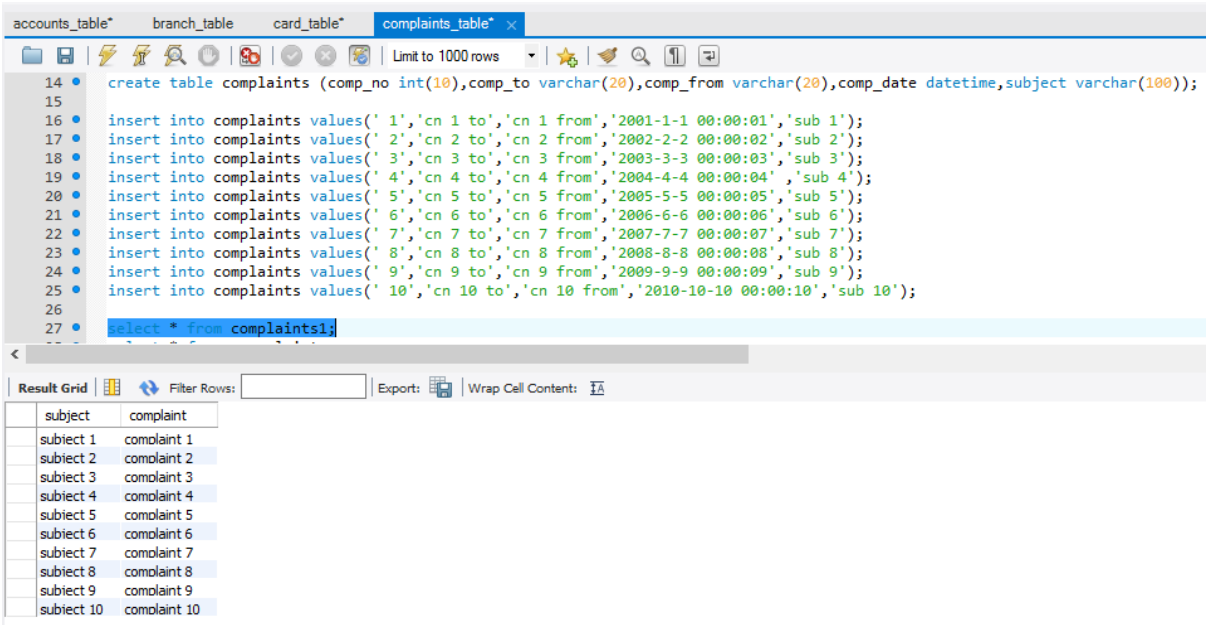
FD's

1. Subject->complaint
   Since Subject is a non-key attribute and it is creating a transitivity we will split the table into two

   [Subject, Complaint] where Subject is the key.
   [cp_no. , cp_to, cp_from, cp_date, subject] where cp_no is the key.

   Since every non-key attribute is dependent on a key the table is in BCNF.

ScreenShots:



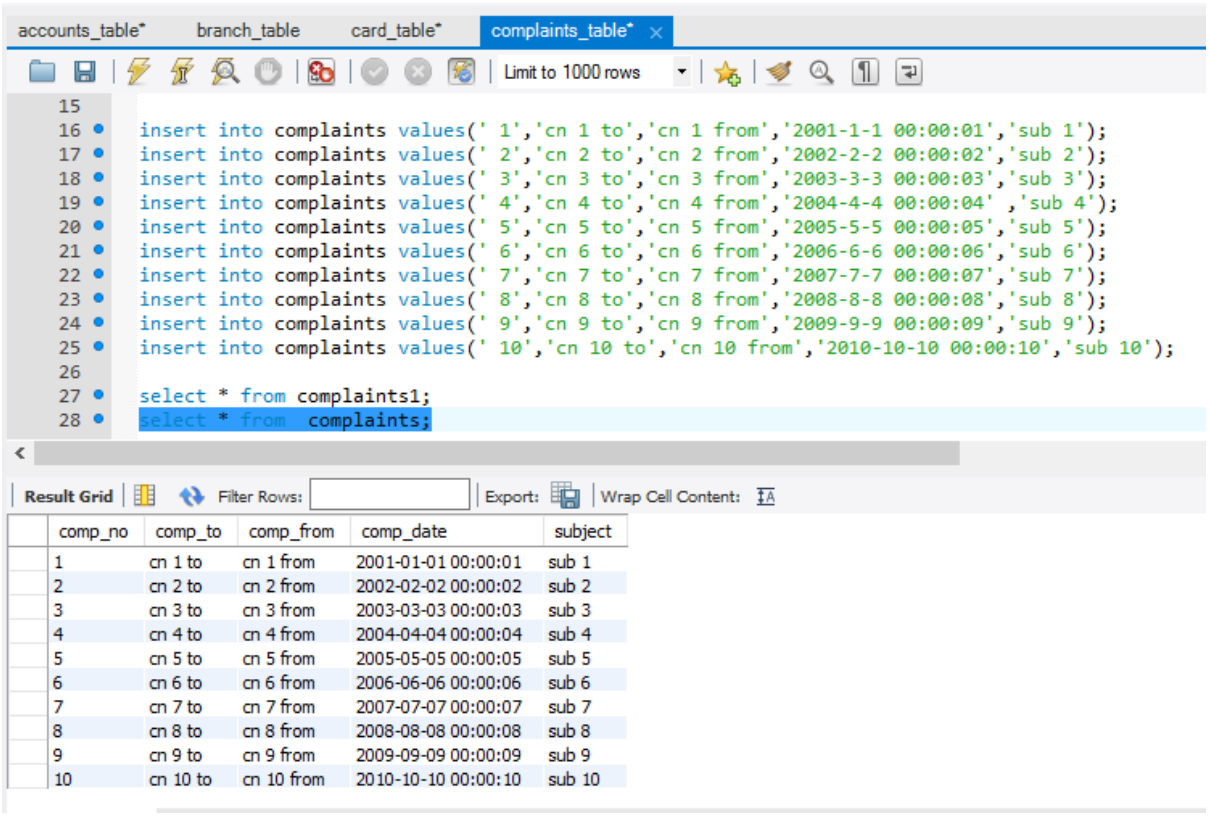


Table 6: Employee (Employee_id, Employee_name, tenure, start_date, telephone_no, designation,salary)

Attributes	Type	Key
Employee_id	varchar(20)	Primary
Employee_name	Varchar(20)	
Telephone_no.	Int(15)	
Start_date	Date	
Work_for Employee_id	Varchar(20)	

Employee_id is the primary key.

FD's

1.designation->salary

Since designation is a non-key attribute and it is creating a transitivity we will split the table into two

[designation,salary] where designation is the key

[Employee_id, Employee_name, tenure, start_date, telephone_no, designation] where Employee_id is the key.

Since every non-key attribute is dependent on a key the table is in BCNF.

ScreenShots:

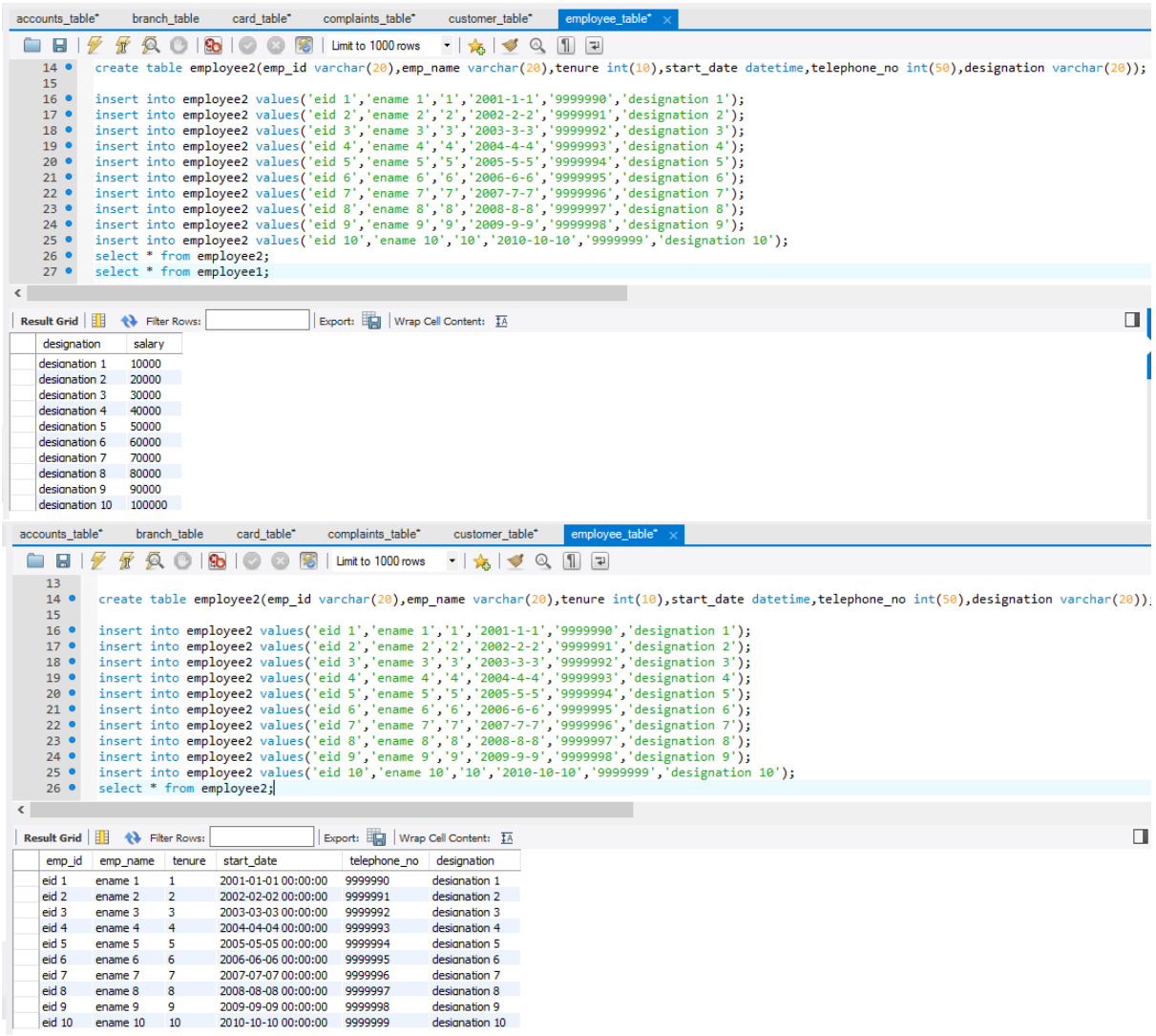


Table 7: Payment (Payment_no., amount, date)

Attributes	Type	Key
payment_date	Date	
Payment_amount	int (100)	
payment_number	int (100)	Primary

Payment_no is the primary key.

FD's

The table has no non_key attributes determining any other attributes hence it doesn't have any transitivity. So the table is in 3NF and is in BCNF as Payment_no is the superkey and is in the left side of all FD's.

ScreenShots:

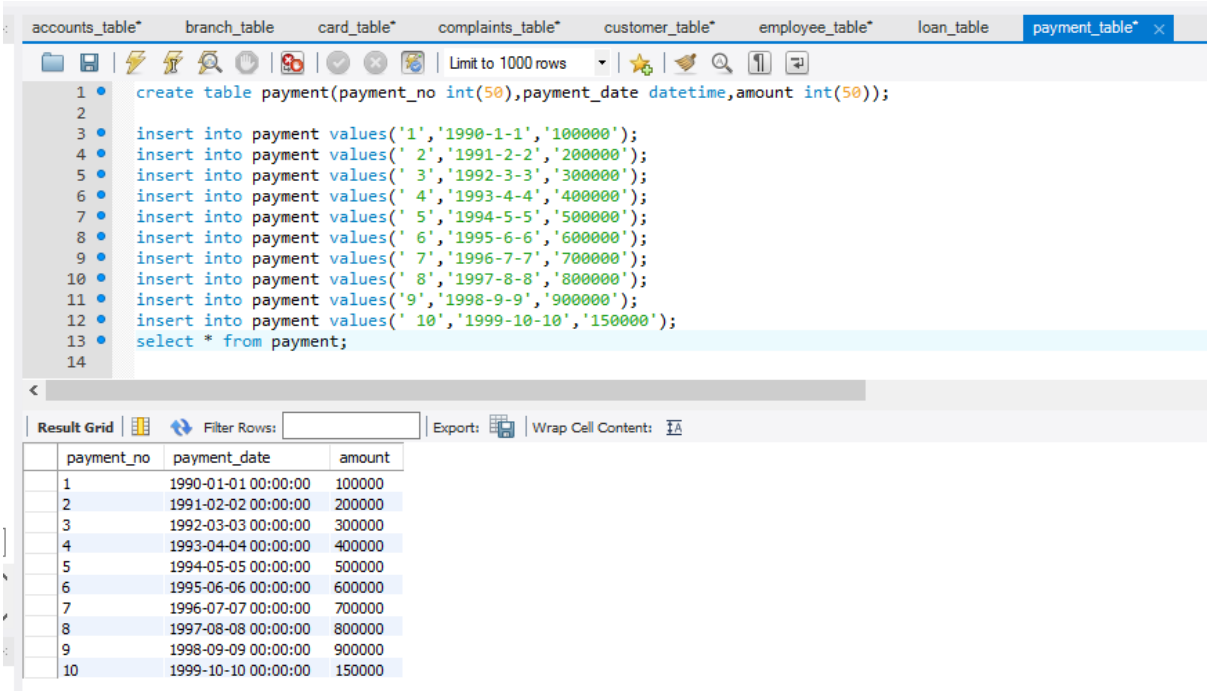


Table 8: Card (card_no, pin, holder_name, validity, card_type, customer_id)

Attributes	Type	Key
Customer_Id	Varchar(10)	Foreign
Card_number	Int(20)	Primary
Validity	Int(10)	
Pin	Int(4)	
Card_type	varchar(10)	

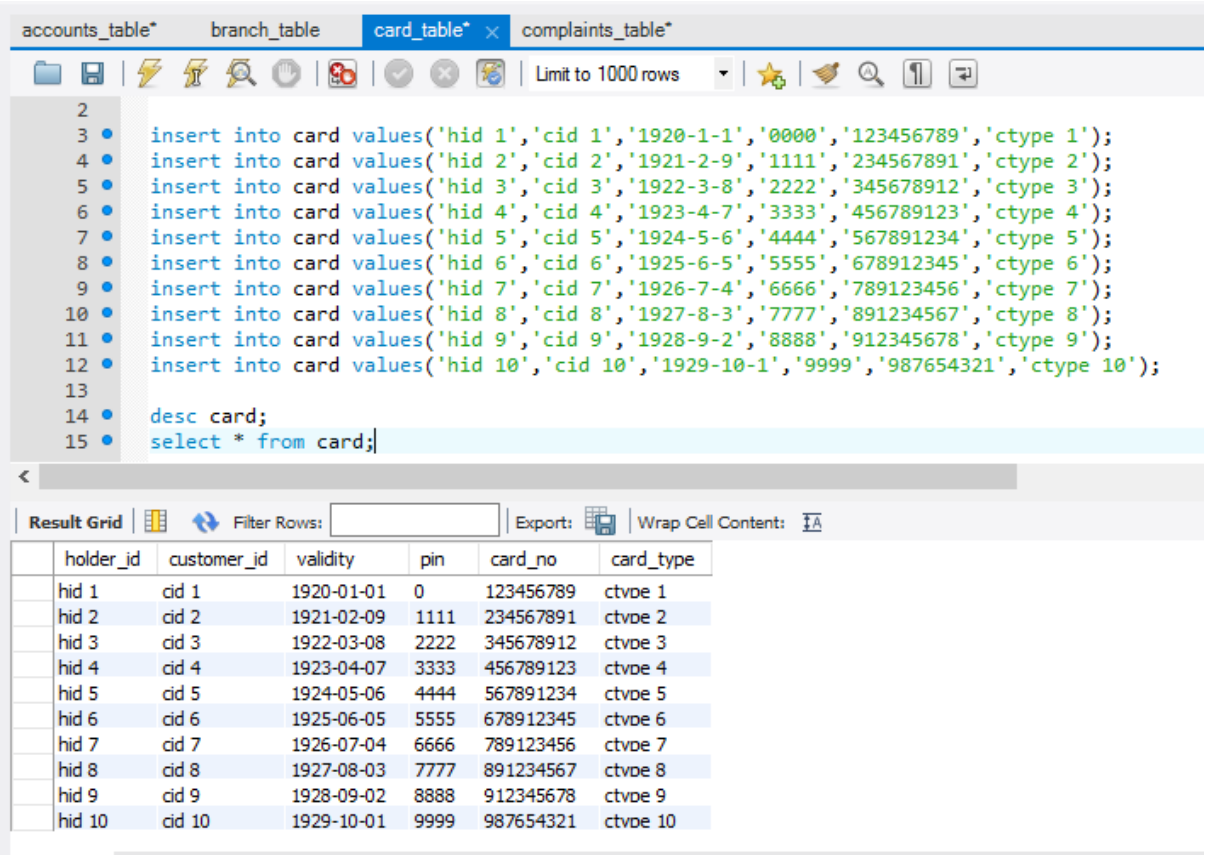
Card_no is the primary key

Customer_id is the foreign key

FD's

The table has no non_key attributes determining any other attributes hence it doesn't have any transitivity. So the table is in 3NF and is in BCNF as Card_no is the superkey and is in the left side of all FD's.

ScreenShots:



BCNF :- Boyce codd normal form

If a relational schema is in BCNF then all redundancy based on FD has been removed, although other types of redundancy may still exist. A relational schema *R* is in Boyce–Codd normal form if and only iffor every one of its dependencies $X \rightarrow Y$, at least one of the following conditions hold

- X→Y is atrivial functional dependency(Y⊆X)
- X is a super Key for schema R

As many of our relationships are many to many and one to many there is only attribute that is the primary key. So it is clear that they are in BCNF. We also checked all the possibilities for multivalued dependencies.

Conclusion:

Our main aim was to reduce redundancy as possible as we can.
The normal form chosen for the Database is in BCNF FORM. We choose this normal form because it minimizes the redundancy and also to achieve losslessness.

----------Thank You----------