Bank management system is a system that includes the user's transaction, account processes and all the information of bank's consultants and the bank.

Entities :

Branch :
branch_id
address
- street
  - street_number
  - street_name
  - apt_number

Customer :
customer_id
name
- first_name
- middle_name
- last_name

address
- street
  - street_number
  - street_name
  - apt_number

gender
email
date_of_birth
age() (now - date_of_birth)

Phone: (Weak Entity) # Decomposed from Customer
phone_number

Zip: # Decomposed from Customer and Bank Branch
zip
city
state

Banker :
banker_id
name
- first name
- last name

banker_email
banker_phone

Card :
card_id
card_type (defines bank card or credit card)
expire_date
password
cvc

Account :
account_number
iban
balance
open_date

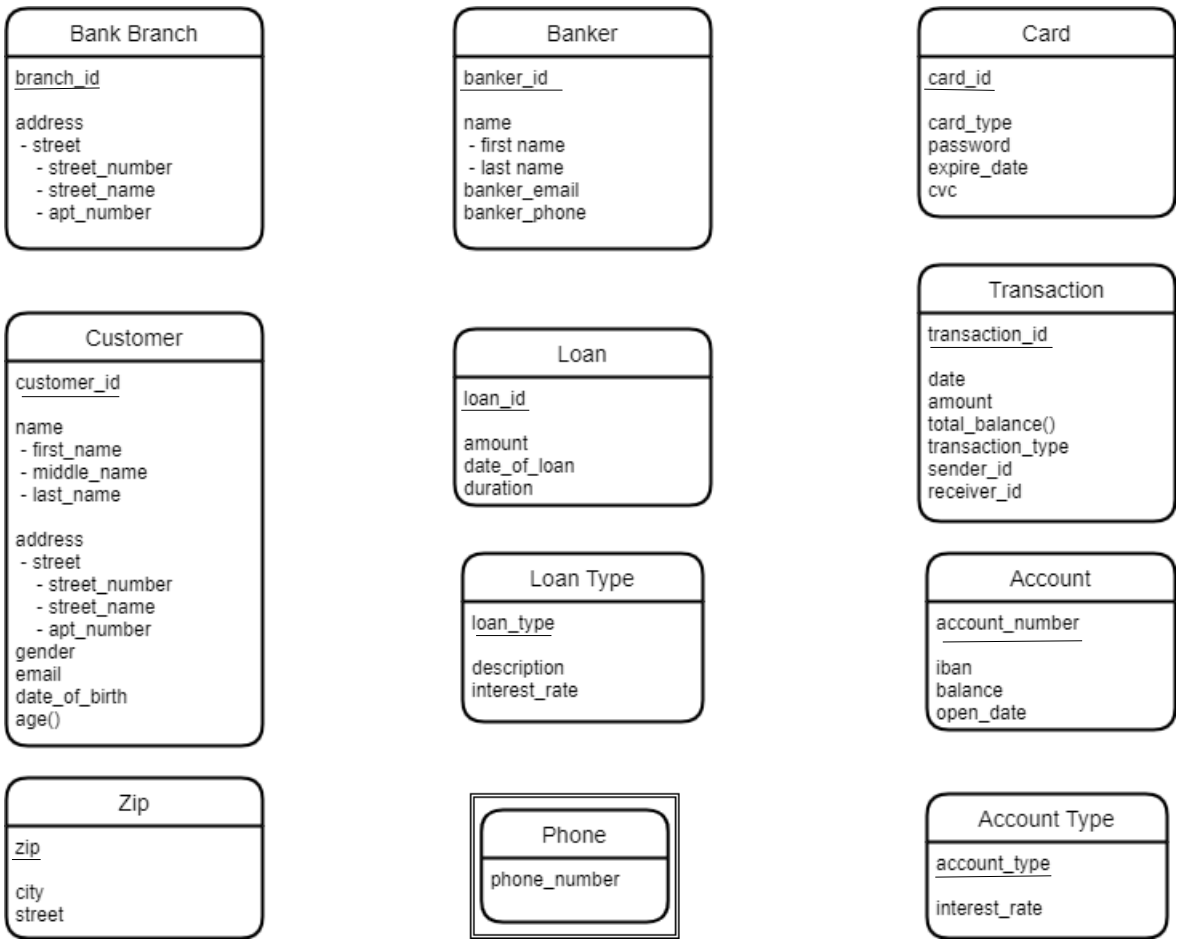
Account Type: # Decomposed from Account
account_type
interest_rate

Loan :
loan_id
amount
date_of_loan
duration

Loan Type: # Decomposed from Loan
loan_type (defines Personal Loans, Credit Cards, Home-Equity Loans, Home-Equity Lines of Credit, Credit Card Cash Advances, Small Business Loans)
description
interest_rate

Transaction :
transaction_id
date
amount
total_balance() (account.balance - transaction.amount)
transaction_type (defines payment or refund)
sender_id
receiver_id

ENTITY SET



Users :

- Customer
- Banker
- Branch

Assumption:

- Interest rate values can be modified according to economic situations.
- Details of a loan can be changed according to branches
- Working hours can be altered according to situations.

Assumptions about Cardinalities :

A customer must have one or many accounts.
Every customer must have one banker.
A customer must have zero or many loans.
Every customer must have one zip.
A customer must have one or many phones.

Every loan must have one branch.
Every loan must have one customer.
A loan must have one loan type.
A loan type must have one loan.
Every banker must have one branch.
A banker must have one or many customer.
A branch must have one or many banker.
A branch must have one or many loans.
A branch must have one or many accounts.
Every branch must have one zip.
A zip must have one or many branch.
A zip must have one or many customer.
Every account must have one branch.
Every account must have one customer.
An account must have one account type.
An account must have one or many cards.
An account must have one or many transactions.
An account type must have one account.
Every transaction must have one account.
Every card must have one account.
Every phone must have one customer.

Business Rule:
- Withdraw is limited to 1000 TL in a day.
- Money transfers between different banks can be occur in working hours.
- An account of a customer can not be connected to multiple branches.
- There cannot be more than one branch with the same zip code.

DECOMPOSING

FOR ACCOUNT :

ACCOUNT (ACCOUNT_NO, ACCOUNT_TYPE, IBAN, BALANCE, OPENDATE, INTEREST_RATE)

FD: ACCOUNT_TYPE -> INTEREST_RATE
FD violates BCNF, so we decompose.

R: (ACCOUNT_NO, ACCOUNT_TYPE, IBAN, BALANCE, OPENDATE, INTEREST_RATE)
FD: ACCOUNT_TYPE -> INTEREST_RATE

ACCOUNT_TYPE+ = ACCOUNT_TYPE,INTEREST_RATE

R1: (ACCOUNT_TYPE, INTEREST_RATE)
R2: (ACCOUNT_NO, ACCOUNT_TYPE, IBAN, BALANCE, OPENDATE)
R1 and R2 doesn't violate BCNF, so no need more decompose.

FOR LOAN :

LOAN(LOAN_ID,LOAN_TYPE,AMOUNT,DATE,DESCRIPTION,DURATION,INTEREST_RATE)

FD : LOAN_TYPE->DESCRIPTION,INTEREST_RATE
FD violates BCNF, so we decompose.

R : (LOAN_ID,LOAN_TYPE,AMOUNT,DATE,DESCRIPTION,DURATION,INTEREST_RATE)
FD : LOAN_TYPE->DESCRIPTION,INTEREST _RATE

LOAN_TYPE+ = LOAN_TYPE,DESCRIPTION,INTEREST_RATE

R1: (LOAN_TYPE,DESCRIPTION,INTRATE)
R2: (LOAN_ID,LOAN_TYPE,AMOUNT,DATE,DURATION)
R1 and R2 doesn't violate BCNF, so no need more decompose.

FOR BRANCH:

BRANCH(BRANCH_ID,STREET_NO,STREET_NAME,APT_NO,CITY,STATE,ZIP)

FD : ZIP->CITY,STATE
FD violates BCNF, so we decompose.

R : (BRANCH_ID,STREET_NO,STREET_NAME,APT_NO,CITY,STATE,ZIP)
FD : ZIP->CITY,STATE

ZIP+ = ZIP,CITY,STATE

R1 : (ZIP,CITY,STATE)
R2 :(BRANCH_ID,STREET_NO,STREET_NAME,APT_NO,ZIP)
R1 and R2 doesn't violate BCNF, so no need more decompose.

FOR CUSTOMER :

CUSTOMER(CUSTOMER_ID,FIRST_NAME,MIDDLE_NAME,LAST_NAME,STREET_NUMBER,STREET_NAME,APT_NUMBER,CITY,STATE,ZIP,GENDER,EMAIL,PHONE_NUMBER,DATE_OF_BIRTH)

FD : ZIP -> CITY,STATE
MVD : CUSTOMER_ID - -> PHONE_NUMBER
FD violates BCNF, so we decompose.
MVD violates 4NF, so we decompose as BCNF.

R:
(CUSTOMER_ID,FIRST_NAME,MIDDLE_NAME,LAST_NAME,STREET_NUMBER,STREET_NAME,APT_NUMBER,CITY,STATE,ZIP,GENDER,EMAIL,PHONE_NUMBER,DATE_OF_BIRTH)
FD : ZIP -> CITY,STATE
MVD : CUSTOMER_ID - -> PHONE_NUMBER

ZIP+ = ZIP,CITY,STATE

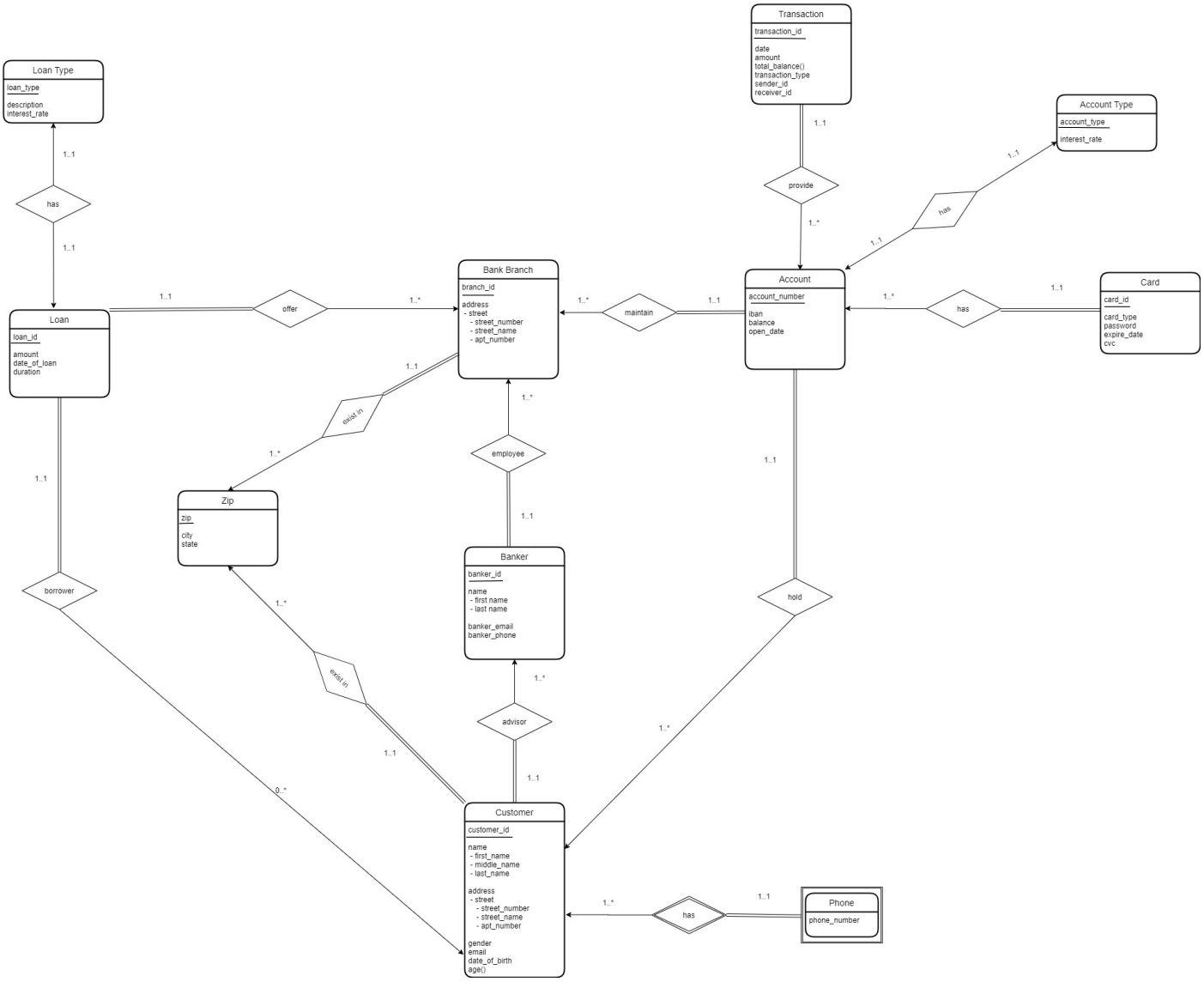
R1:(ZIP,CITY,STATE)

R2:
(CUSTOMER_ID,FIRST_NAME,MIDDLE_NAME,LAST_NAME,STREET_NUMBER,STREET_NAME,APT_NUMBER,ZIP,GENDER,EMAIL,DATE_OF_BIRTH)

R3: (CUSTOMER_ID,PHONE_NUMBER)
R1, R2 and R3 doesn't violate BCNF, so no need more decompose.

Banker, Card, Transaction have no FD. No need to decompose them.

ER DIAGRAM


Transaction
transaction_id
date
amount
total_balance()
transaction_type
sender_id
receiver_id
Account Type
account_type
interest_rate
Loan Type
loan_type
description
interest_rate
provide
has
has
1..1
1..*
Bank Branch
branch_id
address
- street
- street_number
- street_name
- apt_number
Account
account_number
iban
balance
open_date
Card
card_id
card_type
password
expire_date
cvc
offer
maintain
has
Loan
loan_id
amount
date_of_loan
duration
exist in
employee
Zip
zip
city
state
Banker
banker_id
name
- first name
- last name
banker_email
banker_phone
borrower
hold
exist in
advisor
Customer
customer_id
name
- first_name
- middle_name
- last_name
address
- street
- street_number
- street_name
- apt_number
gender
email
date_of_birth
age()
has
Phone
phone_number

Relational Schema

Without relation :
Bank Branch (branch_id, street_number, street_name, apt_number)

Banker (banker_id , first_name, last_name, banker_email, banker_phone)

Customer (customer_id, first_name, middle_name, last_name, street_number, street_name, apt_number, gender, email, date_of_birth)

Card (card_id, card_type, password, expire_date, cvc)

Account (account_number, iban, balance, open_date)

Account Type (account_type, interest_rate)

Transaction (transaction_id, date, amount, transaction_type,sender_id,receiver_id)

Loan (loan_id, amount, date_of_loan, duration)

Loan Type (loan_type, description,interest_rate)

Zip (zip, city,state)

Phone(customer_id, phone_number)

Relations:
Maintain (branch_id, account_number) : between Account and Branch
Provide (account_number, transaction_id) : between Account and Transaction
Offer (loan_id, branch_id) : between Loan and Branch
Has (account_number, card_id) : between Account and Card
Hold (customer_id, account_number) : between Customer and Account
Advisor (banker_id, customer_id) : between Banker and Customer
Borrower (customer_id , loan_id) : between Loan and Customer
Employee (banker_id, branch_id) : between Banker and Branch
Has(loan_type,loan_id) : between Loan Type and Loan
Has(account_type,account_id) : between Account Type and Account
Exist in (branch_id, zip): between Zip and Branch
Exist in (zip,customer_id) : between Customer and Zip
Has (customer_id,phone_number) : between Customer and Phone

With relation :

Bank Branch (branch_id, street_number, street_name, apt_number, city, state,zip)

Banker (banker_id, first_name, last_name, banker_email, banker_phone, branch_id)

Customer (customer_id, first_name, middle_name, last_name, street_number, street_name, apt_number, city, state, zip, gender, email, date_of_birth, banker_id, zip)

Card (card_id, card_type,password, expire_date, cvv, account_number)

Account (account_number, iban, balance, open_date, interest_rate, branch_id , customer_id, account_type)

Account Type (account_type,interest_rate)

Transaction (transaction_id, date, amount, transaction_type, sender_id,receiver_id, account_number)

Loan (loan_id, amount, date_of_loan, description, duration, interest_rate, branch_id, customer_id, loan_type)

Loan Type (loan_type,description,interest_rate)

Zip (zip,city,state)

Phone(customer_id,phone_numbers)

~~Maintain (branch_id, account_number) : between Account and Branch~~
~~Provide (account_number, transaction_id) : between Account and Transaction~~
~~Offer (loan_id, branch_id) : between Loan and Branch~~
~~Has (account_number, card_id) : between Account and Card~~
~~Hold (customer_id, account_number) : between Customer and Account~~
~~Advisor (banker_id, customer_id) : between Banker and Customer~~
~~Borrower (customer_id , loan_id) : between Loan and Customer~~
~~Employee (banker_id, branch_id) : between Banker and Branch~~
~~Has(loan_type,loan_id) : between Loan Type and Loan~~
~~Has(account_type,account_id) : between Account Type and Account~~
~~Exist in (branch_id, zip): between Zip and Branch~~
~~Exist in (zip,customer_id) : between Customer and Zip~~
~~Has (customer_id,phone_number) : between Customer and Phone~~

Last Relational Schema :

Bank Branch (branch_id, street_number, street_name, apt_number, city, state,zip)

Banker (banker_id, first_name, last_name, banker_email, banker_phone, branch_id)

Customer (customer_id, first_name, middle_name, last_name, street_number, street_name, apt_number, city, state, zip, gender, email, date_of_birth, banker_id, zip)

Card (card_id, card_type,password, expire_date, cvv, account_number)

Account (account_number, iban, balance, open_date, interest_rate, branch_id , customer_id, account_type)

Account Type (account_type,interest_rate)

Transaction (transaction_id, date, amount, transaction_type, sender_id, receiver_id, account_number)

Loan (loan_id, amount, date_of_loan, description, duration, interest_rate, branch_id, customer_id, loan_type)

Loan Type (loan_type,description,interest_rate)

Zip (zip,city,state)

Phone(customer_id,phone_numbers)