Third normal form proof

1.

Branch(BranchID, OpeningDate, Phone, Fax, City, Street, managerID,

IsHeadOffice, province)

(Assumption : one branch has only “main service phone number” and has only one location.)

FD BranchId  openingDate, BranchId  OpeningDate, BranchId  phone, BranchId  Fax,

BranchId  City, BranchId  managerID, BranchId  isHeadOffice, BranchId  Provence.

Since for all FD's over branch, the left hand side is BranchID, which is the key of this schema,

Branch schema is in BCNF, thus also in 3NF.

2.

Employee(EmployeeID, Phone, address, StartDate, Wage, BranchID, FirstName, LastName, Email)

(Assumption: one Employee can only has one “working phone number” and has one “current address” also, he may only works for/reside in one Branch, has only one “working email”)

FD EmployeeID phone, EmployeeID address, EmployeeID startDate,

EmployeeID startDate, EmployeeID wage, EmployeeID BranchID, EmployeeID FirstName, EmployeeID lastName, EmployeeID email.

Since for all FD's over Employee, the left hand side is BranchID, which is the key,

Schema Employee is in BCNF, thus also in 3NF.

3.

Schedule(EmployeeID, Date, HourBegin, HourLeft, isHolyday)

FD EmployeeID, Date  HourBegin, EmployeeID, Date HourLeft, EmployeeID, Date  isHolyday

Employee and date makes up the composite key and is the LHS of all the related FD's, therefore

it is in BCNF thus in 3NF.

4

Services(ServicesID, servicesName, ManagerID)

FD's ServicesID servicesName, ServicesID  ManagerID

Since all the LHS are the set key, it is in 3NF

5

Client(client\_id, firstName, lastName, city, province, dob, join\_date, standing, email, phone, category, branch\_id)

(Assumption: one client has only one “bank contacting email” and one “bank contacting phone”)

FD client\_id  firstName, client\_id  lastName, client\_id  city, client\_id  province, client\_id  dob, client\_id  join\_date, client\_id  standing, client\_id  email, client\_id  phone, client\_id  category, client\_id  branch\_id;

Since the LHS are all key, this schema in BCNF, thus also in 3NF.

6

Account(account\_id, client\_id, account\_type, chargePlan\_id, balance, credit\_limit, interest\_rate, lvl, transactionLeft)

account\_id  client\_id, account\_id  account\_type, account\_id  charegPlan, account\_id  balance, account\_id  credit\_limit, account\_id  interest\_rate, account\_id  lvl, account\_id  transactionLeft)

Since the LHS are all key, this schema in BCNF, thus also in 3NF.

7

ChargePlan( chargePlan\_id, option\_name, draw\_limit, charge\_value);

FDs chargePlan\_id  option\_name, chargePlan\_id draw\_limit, chargePlan\_id charge\_value)

Since the LHS are all key, this schema in BCNF, thus also in 3NF.

8

Transaction(tid, account1\_id, account2\_id, amount, dt)

FDs tid  account1\_id, tid  account2\_id, tid \_amount, tid  dt)

Since the LHS are all key, this schema in BCNF, thus also in 3NF.

9

Bills(bill\_id, amount, account1\_id, account2\_id, recurring)

FDs bill\_id  amount, bill\_id  account1\_id, bill\_id  account2\_id, bill\_id  recurring

Since the LHS are all key, this schema in BCNF, thus also in 3NF.

10

EmployeeLogin(employee\_id, psw)

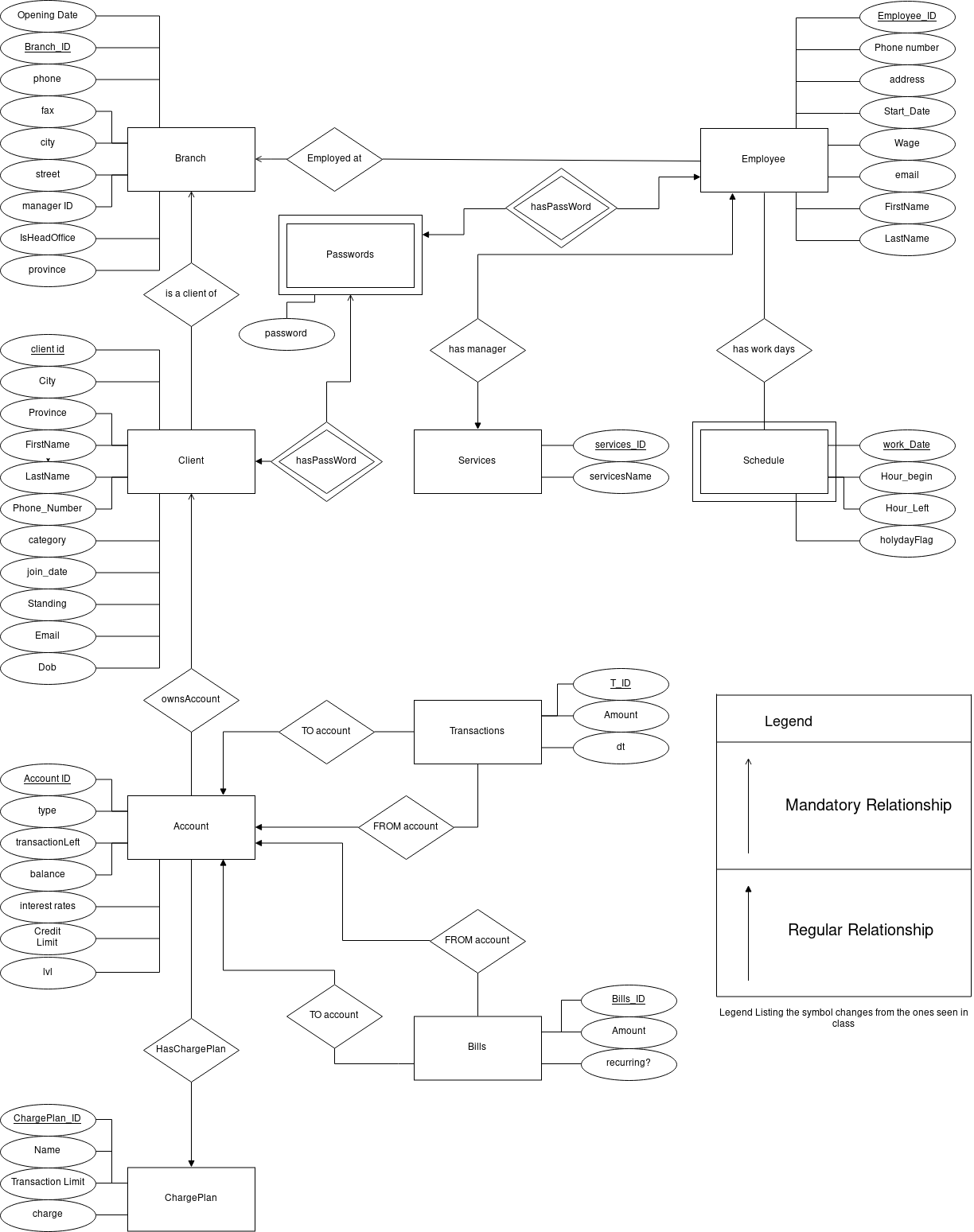
Since there are only two attributes in this schema, it must be in BCNF.

11

ClientLOgin(client\_id, psw)

Since there are only two attributes in this schema, it must be in BCNF. Transactions History

# ER Diagram



# Assumptions

This section will split the project up into a number of parts and will evaluate the associated assumptions that were made. The parts are as such: global logic of the bank, client pages and administrative pages.

## Global Assumptions

First up is the global logic of the bank. In order to identify the main branch or the head office, we created a flag in the Branch table that is set to true only for the head office. The president of the bank is considered to be the manager of the head office. The managers of the branches are also considered employees of the bank and have an entry in the Employee table.

### Branch

In order to keep track of the banks earnings, clients and accounts are created for each branch. The setup makes it possible to send any fees that a client incurs to the account of his branch. A transaction of the fee is also created. The transaction includes a date and time which allows the calculation of the profits of each branch for a window in time. Thus, this approach to processing fees is conducive to an easy compilation of annual profits.

### Bills

Bills are between a sender and receiver account. Importantly to note, bills are not processed immediately. The bills get processed at a consistent frequecy by the bank and the bank takes a fee on a successful payment. A crontab job is setup on the server to run monthly. This periodic PHP script looks at all the bills and performs the transfer of money only if the sender account has sufficient funds. If the bill is recurrent, the bill will be left in the database. In the case that the account had insufficient funds two things may happen. If the bill is not recurrent, then the bill will remain unpaid and in the Bills table. In the situation where it is recurrent, then the server will duplicate the information in the bill and add it to the Bills table (with the recurrent tag set to false). The logic behind this is that if you failed to pay a bill this month, you will owe the receiver two payments the next month. Of note, clients can setup bills for themselves and administrators have the power to create and modify the bills of all clients.

## Clients

For the clients, they have to log in from the home page. Upon succesfully entering their client id and their password, they are redirected to the client hub. This hub lists all their information to them. The hub is their launching point to perform all the actions they need.

At the top of the page, the clients may see their personal information which includes their name, their address, their date of birth and other fields. Under their personal information, there is a table with all their accounts. In this table, the Ids , the levels and the current balances of their accounts are listed. From this table, clients are able to dig deeper into the state of their specific accounts. They simply need to select which account they want more information about and then click the ‘See Account’ button. Once pressed, they will go the their specific account page. On their account’s page, they will see their past transactions and the bills that are linked to this particular account. The transactions and the bills are chronologically ordered in their respective tables.

The quick sending money functionnality of the bank allows user to send money to other clients, given either the client ID of the other party, or by giving their unique email or unique phone number.

## Administrators

The administrators, authorized employees, can gain super user powers into the database by entering their employee id and password on the login page and selecting the Employee radio button. This redirects them to the admin hub. Similarly to the client hub, the admin hub contains all the links for them to perform their desired actions. In short, it is from there that they can get access to pages that modify every single table within the database. It also allows them to search the content of the tables.

## Abstractions

In regards to the backend of the bank’s website, classes were created to reflect the main tables in the database. A few examples of the classes are for bills, clients, branches and others. This form of programming decreased code reuse and streamlined the work between all the team members. Some people could work on certain parts and only deal the methods without having to implement all the logic associated with their actions.

## Security

Given that login page is used both by clients and administrators, a security measure has been put in place to limit the exposition of the clients to the higher privilege pages. To access admin pages, session information is verified and only up meeting certain criterias can a user view the admin pages. The user must have an employee ID that is registered in the EmployeeLogin table. This table records the id of employees and their password to gain access to the website. Similarly, only a client may view client and account information. The information is logged in the session information and verification is performed at every page so as not to leak any personal information. The choice of not putting the passwords with the Employee and Client table was deliberate as it is not good practice to merge such sensitive information into the same table. In addition, the passwords should not be put into a clear text form. However, given the focus of the project being on the implementation of a small database and not on the security aspects of it we did not store hashed forms of the passwords.