# BANKING APPLICATION

**CS331-003 TERM PROJECT**

**BY**

**ABEL BIJUMATHEW**
**(ab2323)**

**December 14, 2020**

# Bank Schema Documentation V1.0

# Normalize the Relations

## USER_TYPE

The table is used to store the different types of users like "Account Holder" and "Staff."

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| USER_TYPE_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| NAME | VARCHAR (255) | | Y | Y | |

**Sample Data**

| USER_TYPE_ID | NAME |
|---|---|
| 1 | Account Holder |
| 2 | Staff |

**FD: (USER_TYPE_ID) -> {NAME}**

## USER

The table is used to store the user's information.

**Key points**

- We are keeping a centralized table to keep the user login information for staff and account holders. This helps us to validate the staff and account holder login in the login flow.
- Status is used to active or inactive user login if any issue has with the security issue. In this case, the user may still need an active account holder.
- Username cannot be duplicated globally for "Staff" and "Account Holder" because we added the unique constraint.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| LOGIN_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| USERNAME | VARCHAR (255) | | Y | Y | |
| PASSCODE | VARCHAR (255) | | Y | | |
| USER_TYPE_ID | BIGINT (20) | **FOREIGN KEY** | Y | | **USER_TYPE (USER_TYPE_ID)** |
| STATUS | VARCHAR (10) | | Y | | |

**Sample Data**

| LOGIN_ID | USERNAME | PASSCODE | USER_TYPE_ID | STATUS |
|---|---|---|---|---|
| 1 | john | ~~sfsOas##92~~ | 1 | ACTIVE |
| 2 | Clark | ~~lsjd&0232~~ | 2 | ACTIVE |
| 3 | Alex | ~~ads232##$~~ | 1 | ACTIVE |
| 4 | Sara | ~~gfdffr45#$~~ | 1 | ACTIVE |
| 5 | henry | ~~dfbcv#@ff~~ | 1 | ACTIVE |
| 6 | David | ~~kfg#22323~~ | 1 | ACTIVE |

FD: {LOGIN_ID} -> {USERNAME, PASSCODE, USER_TYPE_ID, STATUS}

## STAFF

The table is used to store the staff information.

**Key points**

- We are adding the address table's foreign key in this table because the same address information is used for the account holder.
- The first name and last name column are separated instead of keep in a single column. It helps to show the first name only in the application if it is required.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| STAFF_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| BANK_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | | **BANK (BANK_ID)** |
| FIRST_NAME | VARCHAR (255) | | Y | | |
| LAST_NAME | VARCHAR (255) | | Y | | |
| ADDRESS_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | | **ADDRESS (ADDRESS_ID)** |
| STATUS | VARCHAR (255) | | Y | | |

**Sample Data**

| STAFF_ID | BANK_ID | FIRST_NAME | LAST_NAME | ADDRESS_ID | STATUS |
|---|---|---|---|---|---|
| 1 | 1 | Clark | Price | 2 | ACTIVE |

FD: {STAFF_ID, BANK_ID} -> {FIRST_NAME, LAST_NAME, ADDRESS_ID, STATUS}

## USER_STAFF_LINK

The table is used to link the user and login information.

**Key points**

- We separated the staff's login information from the staff table because login information is required only while login the user.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| USER_STAFF_LINK_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| STAFF_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | **Y** | **STAFF (STAFF_ID)** |
| LOGIN_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | **Y** | **USER (LOGIN_ID)** |

**Sample Data**

| STAFF_ID | BANK_ID | FIRST_NAME | LAST_NAME | ADDRESS_ID | STATUS |
|---|---|---|---|---|---|
| 1 | 1 | Clark | Price | 2 | ACTIVE |

## FD: {USER_STAFF_LINK_ID} -> {STAFF_ID, LOGIN_ID}

## ADDRESS

The table is used to the address information of account holders and staff.

**Key points**

- We are using the same column information of staff and account holder. We created this table, and its primary key is used as a reference key in the staff and account holder table.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| ADDRESS_ID | BIGINT (20) | PRIMARY KEY | | | |
| ADDRESS_1 | VARCHAR (255) | | Y | | |
| ADDRESS_2 | VARCHAR (255) | | Y | | |
| STATE | VARCHAR (255) | | Y | | |
| ZIP_CODE | VARCHAR (255) | | Y | | |
| PHONE | VARCHAR (255) | | Y | | |
| EMAIL | VARCHAR (255) | | Y | | |

**Sample Data**

| ADDRESS_ID | ADDRESS_1 | ADDRESS_2 | STATE | ZIP_CODE | PHONE | EMAIL |
|---|---|---|---|---|---|---|
| 1 | 100 Broadway | New York | NY | 07093 | 7188799000 | john200102@gmail.com |
| 2 | 100 Broadway | New York | NY | 07093 | 7188799001 | clark564223@gmial.com |
| 3 | 100 Broadway | New York | NY | 07093 | 7188799002 | alex344903@gmail.com |
| 4 | 100 Broadway | New York | NY | 07093 | 7188799003 | sara8333@gmail.com |
| 5 | 100 Broadway | New York | NY | 07093 | 7188799000 | john200102@gmail.com |
| 6 | 100 Broadway | New York | NY | 07093 | 7188799000 | john200102@gmail.com |

FD: {ADDRESS_ID} -> {ADDRESS_1, ADDRESS_2, STATE, ZIP_CODE, PHONE, EMAIL}

## SECURITY_QUESTION

The table is used to store all security questions required to ask “Account Holder” while registering this application.

**Key points**

- It helps us add, modify, or delete security quickly, and these changes are effected in the registration “Account Holder”.
- The status column helps to active or inactive the security question from the registration.
- We added the unique constraints in the question columns because questions should not be duplicated.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| SECURITY_QUESTION_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| QUESTION | VARCHAR (255) | | Y | Y | |
| STATUS | VARCHAR (255) | | Y | | |

**Sample Data**

| SECURITY_QUESTION_ID | QUESTION | STATUS |
|---|---|---|
| 1 | Your child name | ACTIVE |

## FD: {SECURITY_QUESTION_ID} -> {QUESTION, STATUS}

## BANK

The table is used to store the bank information.

**Key points**

- This helps to manage the bank transaction for a different bank.
- The application cannot support enter multiple bank details with the same bank name.
- "IFSC_CODE" globally is unique, so we added the individual constrain here.
- The status column helps to active or inactive.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| BANK_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| NAME | VARCHAR (255) | | Y | Y | |
| ADDRESS | VARCHAR (255) | | Y | | |
| STATE | VARCHAR (255) | | Y | | |
| ZIP | VARCHAR (255) | | Y | | |
| IFSC_CODE | VARCHAR (255) | | Y | Y | |
| STATUS | VARCHAR (255) | | Y | | |

**Sample Data**

| BANK_ID | NAME | ADDRESS | STATE | ZIP | IFSC_CODE | STATUS |
|---|---|---|---|---|---|---|
| 1 | HSBC | HSBC House, JE1 1HS | Jersey | 07097 | HSBC000393 | ACTIVE |

FD: {BANK_ID} -> {NAME, ADDRESS, STATE, ZIP, IFSC_CODE, STATUS}

IFSC_CODE is the same as the routing number.

## ACCOUNT_HOLDER

The table is used to store the account holder's information.

**Key points**

- We are adding the address table's foreign key in this table because the same address information is used for staff.
- The first name and last name column are separated instead of keep in a single column. It helps to show the first name only in the application if it's required.
- The application provides to create an account of the account holder with the help of the staff. In this case, the logged the "STAFF_ID" is storing in this column. This helps us to identify which team has created this account.

- The account holder is created their account through this application; in this case, the "STAFF_ID" column will be NULL.

- We create the “CUSTOMER_NUMBER" column to generate the unique customer account number for search an account holder, and the number cannot be duplicated, so that we added unique constraints.
- We are linking the account holder with the bank with the help of "BANK_ID" columns.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| ACCOUNT_HOLDER_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| BANK_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | | **BANK (BANK_ID)** |
| CUSTOMER_NUMBER | BIGINT (20) | | Y | Y | |
| FIRST_NAME | VARCHAR (255) | | Y | | |
| LAST_NAME | VARCHAR (255) | | Y | | |
| ADDRESS_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | | **ADDRESS (ADDRESS_ID)** |
| STATUS | VARCHAR (255) | | Y | | |
| SECURITY_QUESTION_ID | BIGINT (20) | **FOREIGN KEY** | **Y** | | **SECURITY_QUESTION (SECURITY_QUESTION_ID)** |
| ANSWER | VARCHAR (255) | | Y | | |
| STAFF_ID | BIGINT (20) | **FOREIGN KEY** | | | **STAFF (STAFF_ID)** |

FD: {ACCOUNT_HOLDER_ID} -> {BANK_ID, CUSTOMER_NUMER, FIRST_NAME, LAST_NAME, ADDRESS_ID, STATUS, SECURITY_QUESTON_ID, ANSWER, STAFF_ID}

**Sample Data**

| ACCOUNT_HOLDER_ID | BANK_ID | CUSTOMER_NUMBER | FIRST_NAME | LAST_NAME | ADDRESS_ID | STATUS | SECURITY_QUESTION_ID | ANSWER | STAFF_ID |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 10000001 | John | Smith | 1 | ACTIVE | 1 | smith | |
| 2 | 1 | 10000002 | Alex | Chris | 3 | ACTIVE | 1 | Alex | 1 |
| 3 | 1 | 10000003 | Sara | Isla | 4 | ACTIVE | 1 | Isla | 1 |
| 4 | 1 | 10000004 | Henry | Oscar | 5 | ACTIVE | 1 | henry | 1 |
| 5 | 1 | 10000005 | David | William | 6 | ACTIVE | 1 | David | 1 |

## USER_ACCOUNT_HOLDER_LINK

The table is used to link between the account holder and login information.

**Key points**

- We separated the staff's login information from the staff table because login information is required only while login the user.
- We added the unique constraint for “ACCOUNT_HOLDER_ID” and “LOGIN_ID” because an account holder is required only login information.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| USER_ACCOUNT_HOLDER_LINK_ID | BIGINT (20) | PRIMARY KEY | | | |
| ACCOUNT_HOLDER_ID | BIGINT (20) | FOREIGN KEY | Y | Y | ACCOUNT_HOLDER (ACCOUNT_HOLDER_ID) |
| LOGIN_ID | BIGINT (20) | FOREIGN KEY | Y | Y | USER (LOGIN_ID) |

**Sample Data**

| USER_ACCOUNT_HOLDER_LINK_ID | ACCOUNT_HOLDER_ID | LOGIN_ID |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 3 |
| 3 | 3 | 4 |
| 4 | 4 | 5 |
| 5 | 5 | 6 |

## ACCOUNT_LINK

This table is used to create a unique account link for “Bank” and “Account Holder”.

**Key points**

- We are required an account number for the bank to do the transaction between "Bank" and "Account Holder".

- "BANK_ID" and "ACCOUNT_HOLDER_ID" column are allowing the "NULL" value.
    - "ACCOUNT_HOLDER_ID" will be NULL while creating an account link for a "Bank" account.
    - "BANK_ID" will be NULL while creating an account link for an "Account Holder" account.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| ACCOUNT_LINK_ID | BIGINT (20) | **PRIMARY KEY** | | | |
| BANK_ID | BIGINT (20) | **FOREIGN KEY** | | **Y** | **BANK (BANK_ID)** |
| ACCOUNT_HOLDER_ID | BIGINT (20) | **FOREIGN KEY** | | **Y** | **ACCOUNT_HOLDER (ACCOUNT_HOLDER_ID)** |

**Sample Data**

| ACCOUNT_LINK_ID | BANK_ID | ACCOUNT_HOLDER_ID |
|---|---|---|
| 1 | 1 | |
| 2 | | 1 |
| 3 | | 2 |
| 4 | | 3 |
| 5 | | 4 |
| 6 | | 5 |

## ACCOUNT

This table is used to store the account number and create an account for a "Account Holder" and "Bank".

**Key points**

- The account number cannot be duplicated to add the unique constraints in the "NUMBER" column.
- The status column helps to active or inactive an account.
- In this table account, link id can be duplicated, and the same account holder has multiple account numbers like "Saving", "FD", etc
    - .
    - E.g., account holder 2, 4, and 5 have "Saving" and "FD" account.

**Normalize**

- **1NF:-** One account holder has multiple account number
    - We moved the account number from the "ACCOUNT_HOLDER" table into "ACCOUNT."

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| ACCOUNT_ID | BIGINT (20) | PRIMARY KEY | | | |
| ACCOUNT_LINK_ID | BIGINT (20) | FOREIGN KEY | Y | | ACCOUNT_LINK (ACCOUNT_LINK_ID) |
| NUMBER | BIGINT (20) | | Y | Y | |
| DATE | DATE | | Y | | |
| STATUS | VARCHAR (255) | | Y | | |

**FD: {ACCOUNT_ID} -> {ACCOUNT_LINK_ID, NUMBER, DATE, STATUS}**

**Sample Data**

| ACCOUNT_ID | ACCOUNT_LINK_ID | NUMBER | DATE | STATUS |
|---|---|---|---|---|
| 1 | 1 | 498200292020202 | 2020-01-01 | ACTIVE |
| 2 | 2 | 393022292023422 | 2020-01-01 | ACTIVE |
| 3 | 2 | 393022292023423 | 2020-01-01 | ACTIVE |
| 4 | 3 | 393022292023322 | 2020-01-01 | ACTIVE |
| 5 | 4 | 393022292023334 | 2020-01-01 | ACTIVE |
| 6 | 4 | 393022292023666 | 2020-01-01 | ACTIVE |
| 7 | 5 | 393022292023566 | 2020-01-01 | ACTIVE |
| 8 | 5 | 393022292023578 | 2020-01-01 | ACTIVE |
| 9 | 6 | 393022292045366 | 2020-01-01 | ACTIVE |

## ACCOUNT_TYPE

This table is used to store the name of a different type of account.

**Key points**

- “NAME” cannot be duplicated to add the unique constraints in the "NAME" column.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| ACCOUNT_TYPE_ID | BIGINT (20) | PRIMARY KEY | | | |
| NAME | VARCHAR (255) | | Y | Y | |

**FD: {ACCOUNT_TYPE_ID} -> {NAME}**

**Sample Data**

| ACCOUNT_TYPE_ID | NAME |
|---|---|
| 1 | Current |
| 2 | FD |
| 3 | Loan |
| 4 | Savings |

## FIXED_DEPOSIT

This table is used to store the global interest rate of different type of “FD” account.

**Key points**

- "NAME" cannot be duplicated to add the unique constraints in the "NAME" column.
- This helps the application to change the interest rate when required.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| FIXED_DEPOSIT_ID | BIGINT (20) | PRIMARY KEY | | | |
| NAME | VARCHAR (255) | | Y | Y | |
| ACCOUNT_TYPE_ID | BIGINT (20) | FOREIGN KEY | Y | | ACCOUNT_TYPE (ACCOUNT_TYPE_ID) |
| INTEREST_RATE | FLOAT | | Y | | |

**FD: {FIXED_DEPOSIT_ID} -> {NAME, ACCOUNT_TYPE_ID, INTEREST_RATE}**

**Sample Data**

| FIXED_DEPOSIT_ID | NAME | | |
|---|---|---|---|
| 1 | Men | 3 | 7.3 |
| 2 | Woman | 3 | 8.6 |

## ACCOUNT_TYPE_LINK

This table is used to link between the account number and its account type.

**Key points**

- We introduce the table because we are required some additional information necessary to capture if the account is an FD account. The FD account more information we are keeping in the separate table, which is called "FIXED_DEPOSIT".
- The “ACCOUNT_TYPE_ID” and “FIXED_DEPOSIT_ID” are allowing the NULL value.
  - While creating the "FD" account, the "ACCOUNT_TYPE" column will be NULL.
    - Here is the example the account number 3 and 6, the interest in calculating the different interest rates.
  - While creating the "Saving" or "Current" account, the "FIXED_DEPOSIT_ID" column will be NULL.
- We added the unique constraint in the "ACCOUNT_ID" table because it always links with one account type.

**Normalize**

- **2NF:-** One account holder has multiple account type like “Savings”, “FD”, etc
  - We moved the account type from the “ACCOUNT_HOLDER” table
- **3NF:-** Avoid the data duplication of interest rate in the account type link.
  - If its store the interest rate and name against the FD account number, it will be duplicated
    - We created the "FIXED_DEPOSIT" table and kept the name and interest rate in that.
    - The "FIXED_DEPOSIT_ID" added the foreign key from the "FIXED_DEPOSIT" table into "ACCOUNT_TYPE_LINK" table.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| ACCOUNT_TYPE_LINK_ID | BIGINT (20) | PRIMARY KEY | | | |
| ACCOUNT_ID | BIGINT (20) | FOREIGN KEY | Y | Y | ACCOUNT (ACCOUNT_ID) |

| | | | | | |
|---|---|---|---|---|---|
| ACCOUNT_TYPE_ID | BIGINT (20) | FOREIGN KEY | | | ACCOUNT_TYPE (ACCOUNT_TYPE_ID) |
| FIXED_DEPOSIT_ID | BIGINT (20) | FOREIGN KEY | | | FIXED_DEPOSIT (FIXED_DEPOSIT_ID) |

**FD: {ACCOUNT_TYPE_LINK_ID} -> {ACCOUNT_ID, ACCOUNT_TYPE_ID, FIXED_DEPOSIT_ID}**

**Sample Data**

| ACCOUNT_TYPE_LINK_ID | ACCOUNT_ID | ACCOUNT_TYPE_ID | FIXED_DEPOSIT_ID |
|---|---|---|---|
| 1 | 1 | 2 | |
| 2 | 2 | 1 | |
| 3 | 3 | | 1 |
| 4 | 4 | 1 | |
| 5 | 5 | 1 | |
| 6 | 6 | | 2 |
| 7 | 7 | 1 | |
| 8 | 8 | | 1 |
| 9 | 9 | 1 | |

## FD_ACCOUNT

This table is used to store the additional information of "FD" account.

**Key points**

- We added the unique constraint in the "ACCOUNT_ID" table because it always links with one account type.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| FD_ACCOUNT_ID | BIGINT (20) | PRIMARY KEY | | | |
| ACCOUNT_ID | VARCHAR (255) | FOREIGN KEY | Y | Y | |
| DURATION | INT | | Y | | ACCOUNT (ACCOUNT_ID) |
| MATURITY_DATE | DATE | | Y | | |

**FD: {FD_ACCOUNT_ID} -> {ACCOUNT_ID, DURATION, MATURITY_DATE}**

**Sample Data**

| FD_ACCOUNT_ID | ACCOUNT_ID | DURATION | MATURITY_DATE |
|---|---|---|---|
| 1 | 3 | 12 | 2021-02-01 |
| 2 | 6 | 12 | 2021-01-01 |
| 3 | 8 | 12 | 2021-01-01 |

## TRANSACTION_TYPE

This table is used to store the different types of transaction types, and it helps the transaction be made whether "Debit" or "Credit" transaction.

**Key points**

- "NAME" cannot be duplicated to add the unique constraints in the "NAME" column.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| TRANSCTION_TYPE_ID | BIGINT (20) | PRIMARY KEY | | | |
| NAME | VARCHAR (255) | | Y | Y | |

**FD: {TRANSACTION_TYPE_ID}-> {NAME}**

**Sample Data**

| TRANSCTION_TYPE_ID | NAME |
|---|---|
| 1 | Dr |
| 2 | Cr |

## TRANSACTION_MODE

This table is used to store the different types of transaction mode; its bits of help to a transaction is made whether "Deposit", "Withdraw" or "Transfer" transaction.

**Key points**

- "NAME" cannot be duplicated to add the unique constraints in the "NAME" column.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| TRANSCTION_MODE_ID | BIGINT (20) | PRIMARY KEY | | | |
| NAME | VARCHAR (255) | | Y | Y | |

**FD: {TRANSACTION_MODE_ID} -> {NAME}**

**Sample Data**

| TRANSCTION_TYPE_ID | NAME |
|---|---|
| 1 | Deposit |
| 2 | Withdraw |
| 3 | Transfer |

## TRANSACTION

This table is used to store all transactions for an account holder and bank account.

**Key points**

- We added the unique constraint in the "NUMBER" table because the number cannot be duplicated.
- The application provides to do the transaction by bank staff for an account holder. In this case, the logged the "STAFF_ID" is storing in this column. This helps us to identify which team has done the transaction.
    - The account holder does their transaction through this application; in this case, the "STAFF_ID" column will be NULL.
- We are adding the reference key of "ACCOUNT" table into two columns, which are "ACCOUNT_ID" and "RERENCE_ACCOUNT_ID".

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| TRANSACTION_ID | BIGINT (20) | PRIMARY KEY | | | |
| NUMBER | BIGINT (20) | | Y | Y | |
| TRANSACTION_TYPE_ID | BIGINT (20) | FOREIGN KEY | Y | | TRANSACTION_TYPE_ID (TRANSACTION_TYPE_ID) |
| AMOUNT | FLOAT | | Y | | |
| ACCOUNT_ID | BIGINT (20) | FOREIGN KEY | Y | | ACCOUNT (ACCOUNT_ID) |
| REFERENCE_ACCOUNT_ID | BIGINT (20) | FOREIGN KEY | Y | | ACCOUNT (ACCOUNT_ID) |
| DATE | DATE | | Y | | |
| STAFF_ID | BIGINT (20) | FOREIGN KEY | Y | | STAFF (STAFF_ID) |
| TRANSACTION_MODE_ID | BIGINT (20) | FOREIGN KEY | Y | | TRANSACTION_MODE (TRANSACTION_MODE_ID) |

FD: {TRANSACTION_ID} -> {NUMBER, TRANSACTION_TYPE_ID, AMOUNT, ACCOUNT_ID, REFERENCE_ACCOUNT_ID, DATE, STAFF_ID, TRANSACTION_MODE_ID}

**Sample Data**

| TRANSACTION_ID | NUMBER | TRANSACTION_TYPE_ID | AMOUNT | ACCOUNT_ID | REFERENCE_ACCOUNT_ID | DATE | STAFF_ID | TRANSACTION_MODE_ID |
|---|---|---|---|---|---|---|---|---|
| 1 | 100001 | 2 | 250000 | 2 | | 2020-01-01 | 1 | 1 |
| 2 | 100002 | 1 | 10000 | 2 | | 2020-01-02 | | 2 |
| 3 | 100003 | 1 | 100000 | 2 | 3 | 2020-02-01 | | 3 |
| 4 | 100004 | 2 | 100000 | 3 | 2 | 2020-02-01 | | 3 |
| 5 | 100005 | 2 | 87600 | 1 | 3 | 2021-02-01 | | 3 |
| 6 | 100006 | 2 | 87600 | 3 | 1 | 2021-02-01 | | 3 |
| 7 | 100009 | 1 | 100000 | 2 | 1 | 2021-02-03 | | 3 |
| 8 | 100010 | 2 | 100000 | 1 | 2 | 2021-02-03 | | 3 |
| 9 | 100011 | 1 | 100000 | 1 | 4 | 2021-02-04 | | 3 |
| 10 | 100012 | 2 | 100000 | 4 | 1 | 2021-02-04 | | 3 |
| 11 | 100013 | 1 | 187600 | 3 | 2 | 2021-02-01 | | 3 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 12 | 100014 | 2 | 187600 | 2 | 3 | 2021-02-01 | | 3 |

## FD_TRANSACTION_PROCESS

This table is used to store the request to transfer the interest rate from the bank account to the account holder account.

**Key points**

- We added the unique constraint in the “NUMBER" table because the number cannot be duplicated.
- We are keeping the "TRANSACTION_ID" in this table to identify which transaction is created by this process.

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| FD_TRANSACTION_PROCESS_ID | BIGINT (20) | PRIMARY KEY | | | |
| NUMBER | BIGINT (20) | | Y | Y | |
| TRANSACTION_ID | BIGINT (20) | FOREIGN KEY | Y | | TRANSACTION_ID (TRANSACTION_ID) |

**FD: {FD_TRANSACTIION_PROCESS_ID} -> {NUMBER, TRANSACTION_ID}**

**Sample Data**

| FD_TRANSACTION_PROCESS_ID | NUMBER | TRANSACTION_ID |
|---|---|---|
| 1 | 10000001 | 5 |

## DD_TRASACTION

This table is used to store the transaction between the account holder to transfer the DD amount.

**Key points**

- We added the unique constraint in the “NUMBER" table because the number cannot be duplicated.
- We are keeping the “FROM_TRANSACTION_ID” and “TO_TRANSACTION_ID” in this table to identify which transaction transfers the amount to the transaction.
- “TRANSFER_ACCOUNT_ID" is used to identify who to transfer the amount to.
- **“**REFERENCE_DD_TRASACTION_ID” is a self join implemented because when it required the transfer amount from bank to "account holder" (this account holder is "TRANSFER_ACCOUNT_ID”)
    - In this case, we set the NULL value in “NUMBER”, “TRANSFER_ACCOUNT_ID," and "VALID_UP_TO."

**Definition**

| FIELD NAME | DATA TYPE | KEY | NN | U | REFERENCE |
|---|---|---|---|---|---|
| DD_TRASACTION_ID | BIGINT (20) | PRIMARY KEY | | | |
| FROM_TRANSACTION_ID | BIGINT (20) | | Y | | TRANSACTION_ID (TRANSACTION_ID) |
| TO_TRANSACTION_ID | BIGINT (20) | FOREIGN KEY | Y | | TRANSACTION_ID (TRANSACTION_ID) |
| NUMBER | BIGINT (20) | | | Y | |
| TRANSFER_ACCOUNT_ID | BIGINT (20) | | | | ACCOUNT (ACCOUNT_ID) |
| REFERENCE_DD_TRASACTION_ID | BIGINT (20) | | | | DD_TRASACTION (DD_TRASACTION_ID) |
| VALID_UP_TO | INT(11) | | | | |

**FD: {DD_TRANSACTION_ID} -> {FROM_TRANSACTION_ID, TO_TRANSACTION_ID, NUMBER, TRANSFER_ACCOUNT_ID, REFERENCE_DD_TRANSACTION_ID, VALID_UP_TO}**

**Sample Data**

| DD_TRASACTION_ID | FROM_TRANSACTION_ID | TO_TRANSACTION_ID | NUMBER | TRANSFER_ACCOUNT_ID | VALID_UP_TO | REFERENCE_DD_TRASACTION_ID |
|---|---|---|---|---|---|---|
| 2 | 7 | 8 | 10001 | 4 | 3 | |
| 3 | 9 | 10 | | | | 2 |

# SQL Queries

## contains GROUP BY

Finding the name, total amount of debit and credit transaction of the amount and its transaction type of all active account holder

**SQL Query**

```
SELECT AH.FIRST_NAME AS ACCOUNT_HOLDER_FIRST_NAME,
AH.LAST_NAME AS ACCOUNT_HOLDER_LAST_NAME, AC.ACCOUNT_ID,
SUM(TN.AMOUNT) AS TRANSACTION_AMOUNT,
TNT.NAME AS TRANSACTION_TYPE
FROM `BANK.`USER U
INNER JOIN `BANK`.USER_ACCOUNT_HOLDER_LINK AHLNK
ON(AHLNK.LOGIN_ID=U.LOGIN_ID)
INNER JOIN `BANK`.ACCOUNT_HOLDER AH
ON(AH.ACCOUNT_HOLDER_ID=AHLNK.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT_LINK  ALNK
ON(ALNK.ACCOUNT_HOLDER_ID=AH.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT AC
ON(AC.ACCOUNT_LINK_ID=ALNK.ACCOUNT_LINK_ID)
INNER JOIN `BANK`.TRANSACTION TN
ON(TN.ACCOUNT_ID=AC.ACCOUNT_ID)
INNER JOIN `BANK`.TRANSACTION_MODE TNM
ON(TNM.TRANSACTION_MODE_ID=TN.TRANSACTION_MODE_ID)
INNER JOIN `BANK`.TRANSACTION_TYPE TNT
ON (TNT.TRANSACTION_TYPE_ID=TN.TRANSACTION_TYPE_ID)

WHERE AC.STATUS = 'ACTIVE'

GROUP BY AC.ACCOUNT_ID, TNT.NAME, AH.FIRST_NAME, AH.LAST_NAME
```

**Output**

| ACCOUNT_HOLDER_FIRST_NAME | ACCOUNT_HOLDER_LAST_NAME | ACCOUNT_ID | TRANSACTION_AMOUNT | TRANSACTION_TYPE |
|---|---|---|---|---|
| John | Smith | 2 | 437600 | Cr |
| John | Smith | 2 | 210000 | Dr |
| John | Smith | 3 | 187600 | Cr |
| John | Smith | 3 | 187600 | Dr |
| Alex | Chris | 4 | 100000 | Cr |
| Sara | Isla | 5 | 250000 | Cr |
| Sara | Isla | 5 | 210000 | Dr |
| Sara | Isla | 6 | 210000 | Cr |

## contains GROUP BY and HAVING

Finding the name, the total amount of debit and credit transaction of the amount and its transaction type of all active account holder for the account id is "2"

**SQL Query**

```
SELECT AH.FIRST_NAME AS ACCOUNT_HOLDER_FIRST_NAME,
AH.LAST_NAME AS ACCOUNT_HOLDER_LAST_NAME, AC.ACCOUNT_ID,
SUM(TN.AMOUNT) AS TRANSACTION_AMOUNT,
TNT.NAME AS TRANSACTION_TYPE
FROM `BANK.`USER U
INNER JOIN `BANK`.USER_ACCOUNT_HOLDER_LINK AHLNK
ON(AHLNK.LOGIN_ID=U.LOGIN_ID)
INNER JOIN `BANK`.ACCOUNT_HOLDER AH
ON(AH.ACCOUNT_HOLDER_ID=AHLNK.ACCOUNT_HOLDER_ID)
```

```
INNER JOIN `BANK`.ACCOUNT_LINK  ALNK
ON(ALNK.ACCOUNT_HOLDER_ID=AH.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT AC
ON(AC.ACCOUNT_LINK_ID=ALNK.ACCOUNT_LINK_ID)
INNER JOIN `BANK`.TRANSACTION TN
ON(TN.ACCOUNT_ID=AC.ACCOUNT_ID)
INNER JOIN `BANK`.TRANSACTION_MODE TNM
ON(TNM.TRANSACTION_MODE_ID=TN.TRANSACTION_MODE_ID)
INNER JOIN `BANK`.TRANSACTION_TYPE TNT
ON (TNT.TRANSACTION_TYPE_ID=TN.TRANSACTION_TYPE_ID)

WHERE AC.STATUS = 'ACTIVE'

GROUP BY AC.ACCOUNT_ID, TNT.NAME, AH.FIRST_NAME, AH.LAST_NAME

HAVING AC.ACCOUNT_ID = 2
```

**Output**

| ACCOUNT_HOLDER_FIRST_NAME | ACCOUNT_HOLDER_LAST_NAME | ACCOUNT_ID | TRANSACTION_AMOUNT | TRANSACTION_TYPE |
|---|---|---|---|---|
| John | Smith | 2 | 437600 | Cr |
| John | Smith | 2 | 210000 | Dr |

## contains nested query with ALL

Find the List of all transactions of "USER_ID" is "1" and required the first name and last name of the transfer the account holder's name.

**SQL Query**

```
SELECT AC.NUMBER AS ACCOUNT_NO,
TN.NUMBER AS TRANSACTION_NO,
TN.DATE AS TRANSACTION_DATE,
TN.AMOUNT AS TRANSACTION_AMOUNT,
TNT. NAME AS TRANSACTION_TYPE,
TNM.NAME AS TRANSACTION_MODE,
REF_AH.ACCOUNT_NO AS REF_ACCOUNT_NO,
REF_AH.ACCOUNT_NAME AS REF_ACCOUNT_NAME
FROM `BANK.`USER U
INNER JOIN `BANK`.USER_ACCOUNT_HOLDER_LINK AHLNK
ON(AHLNK.LOGIN_ID=U.LOGIN_ID)
INNER JOIN `BANK`.ACCOUNT_HOLDER AH
ON(AH.ACCOUNT_HOLDER_ID=AHLNK.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT_LINK  ALNK
ON(ALNK.ACCOUNT_HOLDER_ID=AH.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT AC
ON(AC.ACCOUNT_LINK_ID=ALNK.ACCOUNT_LINK_ID)
INNER JOIN `BANK`.TRANSACTION TN
ON(TN.ACCOUNT_ID=AC.ACCOUNT_ID)
INNER JOIN `BANK`.TRANSACTION_MODE TNM
ON(TNM.TRANSACTION_MODE_ID=TN.TRANSACTION_MODE_ID)
INNER JOIN `BANK`.TRANSACTION_TYPE TNT
ON (TNT.TRANSACTION_TYPE_ID=TN.TRANSACTION_TYPE_ID)

LEFT JOIN (

SELECT AC.ACCOUNT_ID, AC.NUMBER AS ACCOUNT_NO,
IFNULL(ACT.NAME, FDACT.NAME) AS ACCOUNT_TYPE_NAME,
```

```
IFNULL(AH.FIRST_NAME, 'BANK') AS ACCOUNT_NAME
FROM'BANK'.ACCOUNT AC
INNER JOIN `BANK`.ACCOUNT_TYPE_LINK ACTLNK
ON(ACTLNK.ACCOUNT_ID=AC.ACCOUNT_ID)
LEFT JOIN `BANK`.ACCOUNT_TYPE ACT
ON(ACT.ACCOUNT_TYPE_ID=ACTLNK.ACCOUNT_TYPE_ID)
LEFT JOIN `BANK`.FIXED_DEPOSIT FD
ON(FD.FIXED_DEPOSIT_ID=ACTLNK.FIXED_DEPOSIT_ID)
LEFT JOIN `BANK`.ACCOUNT_TYPE FDACT
ON(FDACT.ACCOUNT_TYPE_ID=FD.ACCOUNT_TYPE_ID)
LEFT JOIN `BANK`.ACCOUNT_LINK ALNK
ON(ALNK.ACCOUNT_LINK_ID=AC.ACCOUNT_LINK_ID)
LEFT JOIN `BANK`.ACCOUNT_HOLDER AH
ON(AH.ACCOUNT_HOLDER_ID=ALNK.ACCOUNT_HOLDER_ID)

) REF_AH
ON(REF_AH.ACCOUNT_ID=TN.REFERENCE_ACCOUNT_ID)

WHERE U.LOGIN_ID = 1
```

**Output**

| ACCOUNT_NO | TRANSACTION_NO | TRANSACTION_DATE | TRANSACTION_AMOUNT | TRANSACTION_TYPE | TRANSACTION_MODE | REF_ACCOUNT_NO | REF_ACCOUNT_NAME |
|---|---|---|---|---|---|---|---|
| 393022292023422 | 100001 | 2020-01-01 | 250000 | Cr | Deposit | | |
| 393022292023422 | 100014 | 2021-02-01 | 187600 | Cr | Transfer | 393022292023423 | John |
| 393022292023423 | 100004 | 2020-02-01 | 100000 | Cr | Transfer | 393022292023422 | John |
| 393022292023423 | 100006 | 2021-02-01 | 87600 | Cr | Transfer | 498200292020202 | Bank |
| 393022292023422 | 100002 | 2020-01-02 | 10000 | Dr | Withdraw | | |
| 393022292023422 | 100003 | 2020-02-01 | 100000 | Dr | Transfer | 393022292023423 | John |
| 393022292023422 | 100009 | 2021-02-03 | 100000 | Dr | Transfer | 498200292020202 | Bank |
| 393022292023423 | 100013 | 2021-02-01 | 187600 | Dr | Transfer | 393022292023422 | John |

## contains nested query with IN

Find the account id, account number, and date of all saving accounts.

**SQL Query**

```
SELECT AC.ACCOUNT_ID, AC.NUMBER, AC.DATE FROM`BANK`.ACCOUNT AC
INNER JOIN `BANK`.ACCOUNT_TYPE_LINK ACTLNK
ON(ACTLNK.ACCOUNT_ID=AC.ACCOUNT_ID)
WHERE ACTLNK.ACCOUNT_TYPE_ID
IN (SELECT ACCOUNT_TYPE_ID
FROM `BANK`.ACCOUNT_TYPE ACT WHERE ACT.NAME='SAVINGS');
```

**Output**

| ACCOUNT_ID | NUMBER | DATE |
|---|---|---|
| 2 | 393022292023422 | 2020-01-01 |
| 4 | 393022292023322 | 2020-01-01 |
| 5 | 393022292023334 | 2020-01-01 |
| 7 | 393022292023566 | 2020-01-01 |
| 9 | 393022292045366 | 2020-01-01 |

# Create View

Create a view for getting the total amount of debit and credit transactions of all account information.

**Create Statement**

```
CREATE VIEW `BANK.`ACCOUNT_HOLDER_TRANSACTIONS AS
SELECT AH.FIRST_NAME AS ACCOUNT_HOLDER_FIRST_NAME,
AH.LAST_NAME AS ACCOUNT_HOLDER_LAST_NAME, AC.ACCOUNT_ID,
SUM(TN.AMOUNT) AS TRANSACTION_AMOUNT,
TNT.NAME AS TRANSACTION_TYPE
FROM `BANK.`USER U
INNER JOIN `BANK`.USER_ACCOUNT_HOLDER_LINK AHLNK
ON(AHLNK.LOGIN_ID=U.LOGIN_ID)
INNER JOIN `BANK`.ACCOUNT_HOLDER AH
ON(AH.ACCOUNT_HOLDER_ID=AHLNK.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT_LINK  ALNK
ON(ALNK.ACCOUNT_HOLDER_ID=AH.ACCOUNT_HOLDER_ID)
INNER JOIN `BANK`.ACCOUNT AC
ON(AC.ACCOUNT_LINK_ID=ALNK.ACCOUNT_LINK_ID)
INNER JOIN `BANK`.TRANSACTION TN
ON(TN.ACCOUNT_ID=AC.ACCOUNT_ID)
INNER JOIN `BANK`.TRANSACTION_MODE TNM
ON(TNM.TRANSACTION_MODE_ID=TN.TRANSACTION_MODE_ID)
INNER JOIN `BANK`.TRANSACTION_TYPE TNT
ON (TNT.TRANSACTION_TYPE_ID=TN.TRANSACTION_TYPE_ID)

WHERE AC.STATUS = 'ACTIVE'

GROUP BY AC.ACCOUNT_ID, TNT.NAME, AH.FIRST_NAME, AH.LAST_NAME;
```

**Select Statement from view**

```
SELECT * FROM `BANK`.ACCOUNT_HOLDER_TRANSACTIONS;
```

**Output**

| ACCOUNT_HOLDER_FIRST_NAME | ACCOUNT_HOLDER_LAST_NAME | ACCOUNT_ID | TRANSACTION_AMOUNT | TRANSACTION_TYPE |
|---|---|---|---|---|
| John | Smith | 2 | 437600 | Cr |
| John | Smith | 2 | 210000 | Dr |
| John | Smith | 3 | 187600 | Cr |
| John | Smith | 3 | 187600 | Dr |
| Alex | Chris | 4 | 100000 | Cr |
| Sara | Isla | 5 | 250000 | Cr |

# Trigger

Trigger statement for inserting the auto generate number in the transaction table.

**Create Trigger Statement**

```
CREATE DEFINER=`ROOT`@`LOCALHOST` TRIGGER `TRANSACTION_BEFORE_INSERT` BEFORE INSERT ON `TRANSACTION` FOR EACH ROW
BEGIN

  DECLARE MAX_NUMBER INTEGER;

  SET MAX_NUMBER = (SELECT MAX(TR.NUMBER)
    FROM `BANK`.TRANSACTION TR);

  IF MAX_NUMBER = 0 THEN
    SET MAX_NUMBER = 1;
  ELSE
    SET MAX_NUMBER = MAX_NUMBER + 1;
  END IF;

  SET NEW.NUMBER = MAX_NUMBER;

END
```

100032 is the maximum number before executing the insert statement.

```
SELECT MAX(NUMBER) FROM `BANK`.TRANSACTION;
```

These are records present in the transaction table for account id is "6" before executing the insert the Statement.

```
SELECT * FROM `BANK`.TRANSACTION WHERE ACCOUNT_ID = 6;
```

**Output**

| TRANSACTION_ID | NUMBER | TRANSACTION_TYPE_ID | AMOUNT | ACCOUNT_ID | REFERENCE_ACCOUNT_ID | DATE | STAFF_ID | TRANSACTION_MODE_ID |
|---|---|---|---|---|---|---|---|---|
| 22 | 100022 | 2 | 200000 | 6 | 5 | 2021-02-01 | 1 | 3 |
| 26 | 100026 | 2 | 17200 | 6 | 1 | 2022-02-01 | 1 | 3 |

We exclude the number column in the insert statement and expect the incremented value to be inserted in this column.

**Insert Statement**

```
INSERT INTO `BANK.`TRANSACTION
  (TRANSACTION_ID, TRANSACTION_TYPE_ID,
  AMOUNT, ACCOUNT_ID,  DATE, TRANSACTION_MODE_ID)
VALUES (33, 1, 10000, 6, '2022-02-02', 1);
```

These are records present in the transaction table for account id is "6" after executing the insert the Statement.

```
SELECT * FROM `BANK`.TRANSACTION WHERE ACCOUNT_ID = 6;
```

**Output**

| TRANSACTION_ID | NUMBER | TRANSACTION_TYPE_ID | AMOUNT | ACCOUNT_ID | REFERENCE_ACCOUNT_ID | DATE | STAFF_ID | TRANSACTION_MODE_ID |
|---|---|---|---|---|---|---|---|---|
| 22 | 100022 | 2 | 200000 | 6 | 5 | 2021-02-01 | 1 | 3 |
| 26 | 100026 | 2 | 17200 | 6 | 1 | 2022-02-01 | 1 | 3 |
| 33 | 100033 | 1 | 10000 | 6 | | 2022-02-02 | | 1 |

## DATABASE SCHEMA

```
-- Table structure for table `account.`
--

CREATE TABLE `account` (
  `account_id` int(11) NOT NULL,
  `account_link_id` int(11) NOT NULL,
  `number` bigint(20) NOT NULL,
  `date` date NOT NULL,
  `staff_id` int(11) DEFAULT NULL,
  `status` enum('active','pending','deleted') NOT NULL DEFAULT 'active'
);

-- Table structure for table `account_holder.`

CREATE TABLE `account_holder` (
  `account_holder_id` int(11) NOT NULL,
  `bank_id` int(11) NOT NULL,
  `customer_number` bigint(20) NOT NULL,
  `first_name` varchar(45) NOT NULL,
  `last_name` varchar(45) NOT NULL,
  `address_id` int(11) NOT NULL,
  `status` varchar(45) NOT NULL,
  `security_question_id` varchar(45) NOT NULL,
  `answer` varchar(45) NOT NULL,
  `staff_id` varchar(45) DEFAULT NULL
);

-- Table structure for table `account_link.`
--

CREATE TABLE `account_link` (
  `account_link_id` int(11) NOT NULL,
  `account_holder_id` int(11) DEFAULT NULL,
  `bank_id` int(11) DEFAULT NULL
);
```

```
-- Table structure for table `account_type`

CREATE TABLE `account_type` (
  `account_type_id` int(11) NOT NULL,
  `name` varchar(45) NOT NULL
);

-- Table structure for table `account_type_link`

CREATE TABLE `account_type_link` (
  `account_type_link_id` int(11) NOT NULL,
  `account_id` int(11) NOT NULL,
  `account_type_id` int(11) DEFAULT NULL,
  `fixed_deposit_id` int(11) DEFAULT NULL
);

-- Table structure for table `address.`

CREATE TABLE `address` (
  `address_id` int(11) NOT NULL,
  `address_1` varchar(45) DEFAULT NULL,
  `address_2` varchar(45) DEFAULT NULL,
  `state` varchar(45) DEFAULT NULL,
  `zip_code` varchar(45) NOT NULL,
  `phone` varchar(45) NOT NULL,
  `email` varchar(45) NOT NULL
);

-- Table structure for table `bank.`

CREATE TABLE `bank` (
  `bank_id` int(11) NOT NULL,
  `name` varchar(45) NOT NULL,
  `address` varchar(45) NOT NULL,
  `state` varchar(45) NOT NULL,
  `zip` varchar(45) NOT NULL,
  `ifsc_code` varchar(45) NOT NULL,
  `status` varchar(45) NOT NULL
) ;

-- Table structure for table `dd_trasaction`
--

CREATE TABLE `dd_trasaction` (
  `dd_trasaction_id` int(11) NOT NULL,
  `from_transaction_id` int(11) NOT NULL,
  `to_transaction_id` int(11) NOT NULL,
```

```
  `number` bigint(20) DEFAULT NULL,
  `transfer_account_id` int(11) DEFAULT NULL,
  `valid_upto` int(11) DEFAULT NULL,
  `reference_dd_trasaction_id` int(11) DEFAULT NULL
);

-- Dumping data for table `dd_trasaction.`

--
-- Table structure for table `fd_account`
--

CREATE TABLE `fd_account` (
  `fd_account_id` int(11) NOT NULL,
  `account_id` int(11) NOT NULL,
  `duration` int(11) NOT NULL,
  `maturity_date` date NOT NULL
) ;
--
-- Table structure for table `fd_transaction_process`
--

CREATE TABLE `fd_transaction_process` (
  `fd_transaction_process_id` int(11) NOT NULL,
  `number` bigint(20) NOT NULL,
  `transaction_id` int(11) NOT NULL
);


-- Table structure for table `fixed_deposit.`
--

CREATE TABLE `fixed_deposit` (
  `fixed_deposit_id` int(11) NOT NULL,
  `name` varchar(45) NOT NULL,
  `account_type_id` int(11) NOT NULL,
  `interest_rate` float NOT NULL
) ;

-- Table structure for table `security_question.`
--

CREATE TABLE `security_question` (
  `security_question_id` int(11) NOT NULL,
  `question` varchar(45) NOT NULL,
  `status` varchar(45) NOT NULL
);
```

```
-- Table structure for table `staff.`
--

CREATE TABLE `staff` (
  `staff_id` int(11) NOT NULL,
  `bank_id` int(11) NOT NULL,
  `first_name` varchar(45) NOT NULL,
  `last_name` varchar(45) NOT NULL,
  `address_id` int(11) NOT NULL,
  `status` varchar(45) NOT NULL
) ;

-- Table structure for table `transaction.`

CREATE TABLE `transaction` (
  `transaction_id` int(11) NOT NULL,
  `number` bigint(20) NOT NULL,
  `transaction_type_id` int(11) NOT NULL,
  `amount` float NOT NULL,
  `account_id` int(11) NOT NULL,
  `reference_account_id` int(11) DEFAULT NULL,
  `date` date NOT NULL,
  `staff_id` int(11) DEFAULT NULL,
  `transaction_mode_id` int(11) NOT NULL
);

-- Table structure for table `transaction_mode.`

CREATE TABLE `transaction_mode` (
  `transaction_mode_id` int(11) NOT NULL,
  `name` varchar(45) NOT NULL
);

-- Table structure for table `transaction_type`

CREATE TABLE `transaction_type` (
  `transaction_type_id` int(11) NOT NULL,
  `name` varchar(45) NOT NULL
);


-- Table structure for table `user.`

CREATE TABLE `user` (
  `login_id` int(11) NOT NULL,
  `username` varchar(45) NOT NULL,
```

```
  `passcode` varchar(45) NOT NULL,
  `user_type_id` int(11) NOT NULL,
  `status` varchar(45) NOT NULL
);
-- Table structure for table `user_account_holder_link`

CREATE TABLE `user_account_holder_link` (
  `user_account_holder_link_id` int(11) NOT NULL,
  `account_holder_id` int(11) NOT NULL,
  `login_id` int(11) NOT NULL
);
-- Table structure for table `user_staff_link`

CREATE TABLE `user_staff_link` (
  `user_staff_link_id` int(11) NOT NULL,
  `staff_id` int(11) NOT NULL,
  `login_id` int(11) NOT NULL
);

-- Table structure for table `user_type.`

CREATE TABLE `user_type` (
  `user_type_id` int(11) NOT NULL,
  `name` varchar(45) NOT NULL
);

-- Indexes for table `account.`
--
ALTER TABLE `account`
  ADD PRIMARY KEY (`account_id`),
  ADD UNIQUE KEY `number_UNIQUE` (`number`),
  ADD KEY `fk_acc_acc_lnk_idx` (`account_link_id`),
  ADD KEY `fk_acc_staff_idx` (`staff_id`);

--
-- Indexes for table `account_holder.`
--
ALTER TABLE `account_holder`
  ADD PRIMARY KEY (`account_holder_id`),
  ADD KEY `fk_bank_idx` (`bank_id`),
  ADD KEY `fk_address_idx` (`address_id`);

--
-- Indexes for table `account_link.`
--
ALTER TABLE `account_link`
  ADD PRIMARY KEY (`account_link_id`),
```

```
  ADD UNIQUE KEY `account_holder_id_UNIQUE` (`account_holder_id`),
  ADD UNIQUE KEY `bank_id_UNIQUE` (`bank_id`);

--
-- Indexes for table `account_type`
--
ALTER TABLE `account_type`
  ADD PRIMARY KEY (`account_type_id`),
  ADD UNIQUE KEY `name_UNIQUE` (`name`);

--
-- Indexes for table `account_type_link`
--
ALTER TABLE `account_type_link`
  ADD PRIMARY KEY (`account_type_link_id`),
  ADD KEY `fk_typ_lnk_acc_idx` (`account_id`),
  ADD KEY `fk_typ_lnk_acc_type_idx` (`account_type_id`),
  ADD KEY `fk_typ_lnk_fd_idx` (`fixed_deposit_id`);

--
-- Indexes for table `address.`
--
ALTER TABLE `address.`
  ADD PRIMARY KEY (`address_id`);

--
-- Indexes for table `bank.`
--
ALTER TABLE `bank`
  ADD PRIMARY KEY (`bank_id`),
  ADD UNIQUE KEY `name_UNIQUE` (`name`),
  ADD UNIQUE KEY `ifsc_code_UNIQUE` (`ifsc_code`);

--
-- Indexes for table `dd_trasaction`
--
ALTER TABLE `dd_trasaction`
  ADD PRIMARY KEY (`dd_trasaction_id`),
  ADD UNIQUE KEY `number_UNIQUE` (`number`),
  ADD KEY `fk_dd_idx` (`from_transaction_id`),
  ADD KEY `fk_dd_to_trn_idx` (`to_transaction_id`),
  ADD KEY `fk_dd_trn_acc_idx` (`transfer_account_id`),
  ADD KEY `fk_dd_trn_ref_idx` (`reference_dd_trasaction_id`);

--
-- Indexes for table `fd_account`
--
```

```
ALTER TABLE `fd_account`
  ADD PRIMARY KEY (`fd_account_id`),
  ADD UNIQUE KEY `account_id_UNIQUE` (`account_id`);

--
-- Indexes for table `fd_transaction_process`
--
ALTER TABLE `fd_transaction_process`
  ADD PRIMARY KEY (`fd_transaction_process_id`),
  ADD UNIQUE KEY `number_UNIQUE` (`number`),
  ADD KEY `fk_fd_proc_tran_idx` (`transaction_id`);

--
-- Indexes for table `fixed_deposit.`
--
ALTER TABLE `fixed_deposit`
  ADD PRIMARY KEY (`fixed_deposit_id`),
  ADD UNIQUE KEY `name_UNIQUE` (`name`),
  ADD KEY `fk_fd_account_type_idx` (`account_type_id`);

--
-- Indexes for table `security_question`
--
ALTER TABLE `security_question`
  ADD PRIMARY KEY (`security_question_id`),
  ADD UNIQUE KEY `question_UNIQUE` (`question`);

--
-- Indexes for table `staff.`
--
ALTER TABLE `staff`
  ADD PRIMARY KEY (`staff_id`),
  ADD KEY `fk_addres_idx` (`address_id`),
  ADD KEY `fk_bank_idx` (`bank_id`);

--
-- Indexes for table `transaction.`
--
ALTER TABLE `transaction`
  ADD PRIMARY KEY (`transaction_id`),
  ADD UNIQUE KEY `number_UNIQUE` (`number`),
  ADD KEY `fk_tr_trn_type_idx` (`transaction_type_id`),
  ADD KEY `fk_tr_acc_ref_idx` (`reference_account_id`),
  ADD KEY `fk_tr_staff_idx` (`staff_id`),
  ADD KEY `fk_tr_trn_mode_idx` (`transaction_mode_id`),
  ADD KEY `fk_tr_acc_idx` (`account_id`);
```

```
--
-- Indexes for table `transaction_mode`
--
ALTER TABLE `transaction_mode`
  ADD PRIMARY KEY (`transaction_mode_id`),
  ADD UNIQUE KEY `name_UNIQUE` (`name`);

--
-- Indexes for table `transaction_type`
--
ALTER TABLE `transaction_type`
  ADD PRIMARY KEY (`transaction_type_id`),
  ADD UNIQUE KEY `name_UNIQUE` (`name`);

--
-- Indexes for table `user.`
--
ALTER TABLE `user`
  ADD PRIMARY KEY (`login_id`),
  ADD UNIQUE KEY `username_UNIQUE` (`username`),
  ADD KEY `fk_user_type_idx` (`user_type_id`);

--
-- Indexes for table `user_account_holder_link`
--
ALTER TABLE `user_account_holder_link`
  ADD PRIMARY KEY (`user_account_holder_link_id`),
  ADD UNIQUE KEY `account_holder_id_UNIQUE` (`account_holder_id`),
  ADD UNIQUE KEY `login_id_UNIQUE` (`login_id`);

--
-- Indexes for table `user_staff_link`
--
ALTER TABLE `user_staff_link`
  ADD PRIMARY KEY (`user_staff_link_id`),
  ADD UNIQUE KEY `staff_id_UNIQUE` (`staff_id`),
  ADD UNIQUE KEY `login_id_UNIQUE` (`login_id`);

--
-- Indexes for table `user_type`
--
ALTER TABLE `user_type`
  ADD PRIMARY KEY (`user_type_id`),
  ADD UNIQUE KEY `name_UNIQUE` (`name`);

-- AUTO_INCREMENT for table `account`
--
```

```
ALTER TABLE `account`
  MODIFY `account_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=38;

--
-- AUTO_INCREMENT for table `account_holder`
--
ALTER TABLE `account_holder`
  MODIFY `account_holder_id` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=35;

--
-- AUTO_INCREMENT for table `account_link`
--
ALTER TABLE `account_link`
  MODIFY `account_link_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=36;

--
-- AUTO_INCREMENT for table `account_type`
--
ALTER TABLE `account_type`
  MODIFY `account_type_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;

--
-- AUTO_INCREMENT for table `account_type_link`
--
ALTER TABLE `account_type_link`
  MODIFY `account_type_link_id` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=28;

--
-- AUTO_INCREMENT for table `address`
--
ALTER TABLE `address`
  MODIFY `address_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=36;

--
-- AUTO_INCREMENT for table `bank`
--
ALTER TABLE `bank`
  MODIFY `bank_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT for table `dd_trasaction`
--
ALTER TABLE `dd_trasaction`
  MODIFY `dd_trasaction_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

```
--
-- AUTO_INCREMENT for table `fd_account`
--
ALTER TABLE `fd_account`
  MODIFY `fd_account_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;

--
-- AUTO_INCREMENT for table `fd_transaction_process`
--
ALTER TABLE `fd_transaction_process`
  MODIFY `fd_transaction_process_id` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=5;

--
-- AUTO_INCREMENT for table `fixed_deposit`
--
ALTER TABLE `fixed_deposit`
  MODIFY `fixed_deposit_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `security_question`
--
ALTER TABLE `security_question`
  MODIFY `security_question_id` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `staff`
--
ALTER TABLE `staff`
  MODIFY `staff_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT for table `transaction`
--
ALTER TABLE `transaction`
  MODIFY `transaction_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=49;

--
-- AUTO_INCREMENT for table `transaction_mode`
--
ALTER TABLE `transaction_mode`
  MODIFY `transaction_mode_id` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=4;

--
-- AUTO_INCREMENT for table `user`
```

```
--
ALTER TABLE `user`
  MODIFY `login_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=36;

--
-- AUTO_INCREMENT for table `user_account_holder_link`
--
ALTER TABLE `user_account_holder_link`
  MODIFY `user_account_holder_link_id` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=35;

--
-- AUTO_INCREMENT for table `user_staff_link`
--
ALTER TABLE `user_staff_link`
  MODIFY `user_staff_link_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT for table `user_type`
--
ALTER TABLE `user_type`
  MODIFY `user_type_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- Constraints for table `account.`
--
ALTER TABLE `account`
  ADD CONSTRAINT `fk_acc_acc_lnk` FOREIGN KEY (`account_link_id`) REFERENCES
`account_link` (`account_link_id`),
  ADD CONSTRAINT `fk_acc_staff` FOREIGN KEY (`staff_id`) REFERENCES `staff` (`staff_id`);

--
-- Constraints for table `account_holder`
--
ALTER TABLE `account_holder`
  ADD CONSTRAINT `fk_account_holder_address` FOREIGN KEY (`address_id`) REFERENCES
`address` (`address_id`),
  ADD CONSTRAINT `fk_account_holder_bank` FOREIGN KEY (`bank_id`) REFERENCES `bank`
(`bank_id`);

--
-- Constraints for table `account_link`
--
ALTER TABLE `account_link`
  ADD CONSTRAINT `fk_acc_lnk_account_holder` FOREIGN KEY (`account_holder_id`)
REFERENCES `account_holder` (`account_holder_id`),
```

```
  ADD CONSTRAINT `fk_acc_lnk_bank` FOREIGN KEY (`bank_id`) REFERENCES `bank`
(`bank_id`);

--
-- Constraints for table `account_type_link`
--
ALTER TABLE `account_type_link`
  ADD CONSTRAINT `fk_typ_lnk_acc` FOREIGN KEY (`account_id`) REFERENCES `account`
(`account_id`),
  ADD CONSTRAINT `fk_typ_lnk_acc_type` FOREIGN KEY (`account_type_id`) REFERENCES
`account_type` (`account_type_id`),
  ADD CONSTRAINT `fk_typ_lnk_fd` FOREIGN KEY (`fixed_deposit_id`) REFERENCES
`fixed_deposit` (`fixed_deposit_id`);

--
-- Constraints for table `dd_trasaction`
--
ALTER TABLE `dd_trasaction`
  ADD CONSTRAINT `fk_dd_from_trn` FOREIGN KEY (`from_transaction_id`) REFERENCES
`transaction` (`transaction_id`),
  ADD CONSTRAINT `fk_dd_to_trn` FOREIGN KEY (`to_transaction_id`) REFERENCES
`transaction` (`transaction_id`),
  ADD CONSTRAINT `fk_dd_trn_acc` FOREIGN KEY (`transfer_account_id`) REFERENCES
`account` (`account_id`),
  ADD CONSTRAINT `fk_dd_trn_ref` FOREIGN KEY (`reference_dd_trasaction_id`)
REFERENCES `dd_trasaction` (`dd_trasaction_id`);

--
-- Constraints for table `fd_account`
--
ALTER TABLE `fd_account`
  ADD CONSTRAINT `fd_acc_acc` FOREIGN KEY (`account_id`) REFERENCES `account`
(`account_id`);

--
-- Constraints for table `fd_transaction_process`
--
ALTER TABLE `fd_transaction_process`
  ADD CONSTRAINT `fk_fd_proc_tran` FOREIGN KEY (`transaction_id`) REFERENCES
`transaction` (`transaction_id`);

--
-- Constraints for table `fixed_deposit.`
--
ALTER TABLE `fixed_deposit`
  ADD CONSTRAINT `fk_fd_account_type` FOREIGN KEY (`account_type_id`) REFERENCES
`account_type` (`account_type_id`);
```

```
--
-- Constraints for table `staff.`
--
ALTER TABLE `staff`
  ADD CONSTRAINT `fk_addres` FOREIGN KEY (`address_id`) REFERENCES `address`
(`address_id`),
  ADD CONSTRAINT `fk_bank` FOREIGN KEY (`bank_id`) REFERENCES `bank` (`bank_id`);

--
-- Constraints for table `transaction.`
--
ALTER TABLE `transaction`
  ADD CONSTRAINT `fk_tr_acc` FOREIGN KEY (`account_id`) REFERENCES `account`
(`account_id`),
  ADD CONSTRAINT `fk_tr_acc_ref` FOREIGN KEY (`reference_account_id`) REFERENCES
`account` (`account_id`),
  ADD CONSTRAINT `fk_tr_staff` FOREIGN KEY (`staff_id`) REFERENCES `staff` (`staff_id`),
  ADD CONSTRAINT `fk_tr_trn_mode` FOREIGN KEY (`transaction_mode_id`) REFERENCES
`transaction_mode` (`transaction_mode_id`),
  ADD CONSTRAINT `fk_tr_trn_type` FOREIGN KEY (`transaction_type_id`) REFERENCES
`transaction_type` (`transaction_type_id`);

--
-- Constraints for table `user.`
--
ALTER TABLE `user`
  ADD CONSTRAINT `fk_user_type` FOREIGN KEY (`user_type_id`) REFERENCES `user_type`
(`user_type_id`);

--
-- Constraints for table `user_account_holder_link`
--
ALTER TABLE `user_account_holder_link`
  ADD CONSTRAINT `fk_link_account_holder` FOREIGN KEY (`account_holder_id`)
REFERENCES `account_holder` (`account_holder_id`),
  ADD CONSTRAINT `fk_link_user` FOREIGN KEY (`login_id`) REFERENCES `user` (`login_id`);

--
-- Constraints for table `user_staff_link`
--
ALTER TABLE `user_staff_link`
  ADD CONSTRAINT `fk_staff` FOREIGN KEY (`staff_id`) REFERENCES `staff` (`staff_id`),
  ADD CONSTRAINT `fk_user` FOREIGN KEY (`login_id`) REFERENCES `user` (`login_id`);
```

## Banking Operations

-- Create account holder

INSERT INTO `bank`.`address` (`address_1`, `address_2`, `state`, `zip_code`, `phone`, `email`) VALUES (100 Broadway', New York', 'NY', '09102', '9292829928', 'arun@gmail.com');

INSERT INTO `bank`.`account_holder` (`bank_id`, `customer_number`, `first_name`, `last_name`, `address_id`, `status`, `security_question_id`, `answer`, `staff_id`) VALUES ('1', '10000001', 'Arun', 'T D', '1', 'ACTIVE', '1', 'Appu', '1');

INSERT INTO `bank`.`user` (`username`, `passcode`, `user_type_id`, `status`) VALUES ('arun', 'sfsOas##92', '1', 'ACTIVE');

INSERT INTO `bank`.`user_account_holder_link` (`account_holder_id`, `login_id`) VALUES ('1', '1');

INSERT INTO `bank`.`account_link` (`account_holder_id`) VALUES ('1');

-- Saving / Current Account Creation
INSERT INTO `bank`.`account` (`account_link_id`, `number`, `date`, `staff_id`) VALUES ('2', '393022292023422', '2020/1/1', '1');

INSERT INTO `bank`.`account_type_link` (`account_id`, `account_type_id`) VALUES ('2', '1');

-- FD Account Creation
INSERT INTO `bank`.`account` (`account_link_id`, `number`, `date`, `staff_id`) VALUES ('2', '393022292023423', '2020/2/1', '1');

INSERT INTO `bank`.`account_type_link` (`account_id`, `fixed_deposit_id`) VALUES ('3', '1');

INSERT INTO `bank`.`fd_account` (`account_id`, `duration`, `maturity_date`) VALUES ('3', '12', '2021/02/1');

-- Deposit Transaction
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100001', '2', '250000', '2', '2020/1/1', '1', '1');

-- Withdraw Transaction

```
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100002', '1', '10000', '2', '2020/1/2', '1', '2');


-- Transfer Amount Saving - FD
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100003', '1', '100000', '2', '3', '2020/2/1', '1', '3');

INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100004', '2', '100000', '3', '2', '2020/2/1', '1', '3');


-- Interest Trasfer (Bank - FD)
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100005', '2', '87600', '1', '3', '2021/2/1', '1', '3');

INSERT INTO `bank`.`fd_transaction_process` (`number`, `transaction_id`) VALUES ('10000001', '5');

INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100006', '2', '87600', '3', '1', '2021-02-01', '1', '3');

-- Transfer Amount FD - Saving
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100013', '1', '187600', '3', '2', '2021/2/1', '1', '3');

INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100014', '2', '187600', '2', '3', '2021/2/1', '1', '3');


-- DD Transfer (Request by Arun)
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100009', '1', '100000', '2', '1', '2021/2/3', '1', '3');

INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100010', '2', '100000', '1', '2', '2021-02-03', '1', '3');
```

```
INSERT INTO `bank`.`dd_trasaction` (`from_transaction_id`, `to_transaction_id`, `number`, `transfer_account_id`, `valid_upto`) VALUES ('7', '8', '10001', '4', '3');


-- DD Transfer (Bank transfer to Mahendra)
INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100011', '1', '100000', '1', '4', '2021-02-04', '1', '3');

INSERT INTO `bank`.`transaction` (`number`, `transaction_type_id`, `amount`, `account_id`, `reference_account_id`, `date`, `staff_id`, `transaction_mode_id`) VALUES ('100012', '2', '100000', '4', '1', '2021-02-04', '1', '3');

INSERT INTO `bank`.`dd_trasaction` (`from_transaction_id`, `to_transaction_id`, `reference_dd_trasaction_id`) VALUES ('9', '10', '2');


-- List of transaction details

SELECT ac.number AS account_no,
tn.number AS transaction_no,
tn.date AS transaction_date,
tn.amount AS transaction_amount,
tnt.name AS transaction_type,
tnm.name AS transaction_mode,
ref_ah.account_no AS ref_account_no,
ref_ah.account_name AS ref_account_name
FROM `bank`.user u
INNER JOIN `bank`.user_account_holder_link ahlnk
ON(ahlnk.login_id=u.login_id)
INNER JOIN `bank`.account_holder ah
ON(ah.account_holder_id=ahlnk.account_holder_id)
INNER JOIN `bank`.account_link  alnk
ON(alnk.account_holder_id=ah.account_holder_id)
INNER JOIN `bank`.account ac
ON(ac.account_link_id=alnk.account_link_id)
INNER JOIN `bank`.transaction tn
ON(tn.account_id=ac.account_id)
INNER JOIN `bank`.transaction_mode tnm
ON(tnm.transaction_mode_id=tn.transaction_mode_id)
INNER JOIN `bank`.transaction_type tnt
ON (tnt.transaction_type_id=tn.transaction_type_id)

LEFT JOIN (

SELECT ac.account_id, ac.number AS account_no,
IFNULL(act.name, fdact.name) AS account_type_name,
```

```
IFNULL(ah.first_name, 'Bank') AS account_name
FROM'bank'.account ac
INNER JOIN `bank`.account_type_link actlnk
ON(actlnk.account_id=ac.account_id)
LEFT JOIN `bank`.account_type act
ON(act.account_type_id=actlnk.account_type_id)
LEFT JOIN `bank`.fixed_deposit fd
ON(fd.fixed_deposit_id=actlnk.fixed_deposit_id)
LEFT JOIN `bank`.account_type fdact
ON(fdact.account_type_id=fd.account_type_id)
LEFT JOIN `bank`.account_link alnk
ON(alnk.account_link_id=ac.account_link_id)
LEFT JOIN `bank`.account_holder ah
ON(ah.account_holder_id=alnk.account_holder_id)

) ref_ah
ON(ref_ah.account_id=tn.reference_account_id)


WHERE u.login_id = 1
-- AND ac.number = '393022292023422'
-- AND tnm.name = 'Transfer'
;


-- List of account details and its name

SELECT ac.account_id, ac.number AS account_no,
IFNULL(act.name, fdact.name) AS account_type_name
FROM `bank`.user u
INNER JOIN `bank`.user_account_holder_link ahlnk
ON(ahlnk.login_id=u.login_id)
INNER JOIN `bank`.account_holder ah
ON(ah.account_holder_id=ahlnk.account_holder_id)
INNER JOIN `bank`.account_link  alnk
ON(alnk.account_holder_id=ah.account_holder_id)
INNER JOIN `bank`.account ac
ON(ac.account_link_id=alnk.account_link_id)
INNER JOIN `bank`.account_type_link actlnk
ON(actlnk.account_id=ac.account_id)
LEFT JOIN `bank`.account_type act
ON(act.account_type_id=actlnk.account_type_id)
LEFT JOIN `bank`.fixed_deposit fd
ON(fd.fixed_deposit_id=actlnk.fixed_deposit_id)
LEFT JOIN `bank`.account_type fdact
ON(fdact.account_type_id=fd.account_type_id)
WHERE u.login_id = 1
```

```
;


-- Find Balance Amount

SELECT account_holder_first_name, account_holder_last_name,
gbalance.account_id, cr_amount, dr_amount, balance,
account_no, account_type_name
FROM (SELECT account_holder_first_name, account_holder_last_name,
gbalance.account_id,
SUM(dr_amount) AS dr_amount,
SUM(cr_amount) AS cr_amount,
SUM(cr_amount) - SUM(dr_amount) AS balance
FROM (SELECT account_holder_first_name, account_holder_last_name,
gbalance.account_id,
CASE WHEN transaction_type='Dr' THEN transaction_amount ELSE 0 END dr_amount,
CASE WHEN transaction_type='Cr' THEN transaction_amount ELSE 0 END cr_amount
FROM (SELECT ah.first_name AS account_holder_first_name,
ah.last_name AS account_holder_last_name, ac.account_id,
SUM(tn.amount) AS transaction_amount,
tnt.name AS transaction_type
FROM `bank`.user u
INNER JOIN `bank`.user_account_holder_link ahlnk
ON(ahlnk.login_id=u.login_id)
INNER JOIN `bank`.account_holder ah
ON(ah.account_holder_id=ahlnk.account_holder_id)
INNER JOIN `bank`.account_link  alnk
ON(alnk.account_holder_id=ah.account_holder_id)
INNER JOIN `bank`.account ac
ON(ac.account_link_id=alnk.account_link_id)
INNER JOIN `bank`.transaction tn
ON(tn.account_id=ac.account_id)
INNER JOIN `bank`.transaction_mode tnm
ON(tnm.transaction_mode_id=tn.transaction_mode_id)
INNER JOIN `bank`.transaction_type tnt
ON (tnt.transaction_type_id=tn.transaction_type_id)
WHERE u.login_id = 1 AND ac.account_id = 3
GROUP BY ac.account_id, tnt.name, ah.first_name, ah.last_name) gbalance) gbalance
GROUP BY gbalance.account_holder_first_name, gbalance.account_holder_last_name,
gbalance.account_id) gbalance

INNER JOIN(
SELECT ac.account_id, ac.number AS account_no,
IFNULL(act.name, fdact.name) AS account_type_name
FROM'bank'.account ac
INNER JOIN `bank`.account_type_link actlnk
ON(actlnk.account_id=ac.account_id)
```

```
LEFT JOIN `bank`.account_type act
ON(act.account_type_id=actlnk.account_type_id)
LEFT JOIN `bank`.fixed_deposit fd
ON(fd.fixed_deposit_id=actlnk.fixed_deposit_id)
LEFT JOIN `bank`.account_type fdact
ON(fdact.account_type_id=fd.account_type_id)
WHERE ac.account_id = 3) ac
ON(ac.account_id=gbalance.account_id)
;


-- Get FD account details by account id

SELECT ac.account_id, ac.number AS account_no,
IFNULL(ah.first_name, 'Bank') AS account_name,
fd.interest_rate, fdacc.duration, fdacc.maturity_date
FROM`bank`.account ac
INNER JOIN`bank`.account_type_link actlnk
ON(actlnk.account_id=ac.account_id)
INNER JOIN`bank`.fixed_deposit fd
ON(fd.fixed_deposit_id=actlnk.fixed_deposit_id)
INNER JOIN`bank`.account_type fdact
ON(fdact.account_type_id=fd.account_type_id)
INNER JOIN`bank`.account_link alnk
ON(alnk.account_link_id=ac.account_link_id)
INNER JOIN`bank`.account_holder ah
ON(ah.account_holder_id=alnk.account_holder_id)
INNER JOIN`bank`.fd_account fdacc
ON(fdacc.account_id=ac.account_id)
WHERE ac.account_id = 3;


-- Get account Id by Account Number

SELECT acc.account_id FROM bank.account acc WHERE acc.number = 393022292023423;

-- Get FD mature details based on the date

SELECT fd.account_id, fd.account_no, fd.account_name,
fd.interest_rate, fd.duration, fd.maturity_date,
tp.reference_account_id
FROM (SELECT ac.account_id, ac.number AS account_no,
IFNULL(ah.first_name, 'Bank') AS account_name,
fd.interest_rate, fdacc.duration, fdacc.maturity_date
FROM`bank`.account ac
INNER JOIN `bank`.account_type_link actlnk
ON(actlnk.account_id=ac.account_id)
```

```
INNER JOIN `bank`.fixed_deposit fd
ON(fd.fixed_deposit_id=actlnk.fixed_deposit_id)
INNER JOIN `bank`.account_type fdact
ON(fdact.account_type_id=fd.account_type_id)
INNER JOIN `bank`.account_link alnk
ON(alnk.account_link_id=ac.account_link_id)
INNER JOIN `bank`.account_holder ah
ON(ah.account_holder_id=alnk.account_holder_id)
INNER JOIN `bank`.fd_account fdacc
ON(fdacc.account_id=ac.account_id)) fd
```

```
LEFT JOIN (SELECT t.reference_account_id
FROM bank.fd_transaction_process tp
INNER JOIN bank.transaction t ON(t.transaction_id=tp.transaction_id)) tp
ON (tp.reference_account_id=fd.account_id)
WHERE maturity_date<=CURRENT_DATE
AND tp.reference_account_id IS NULL
```