# Banking System

| Name | Questions Assigned |
|---|---|
| ***B Srija*** | Task 1 - Q1-Q5 \| Task 2 |
| ***Neha Rajendra Nandurkar*** | Task 3 - Q1-Q6 \| Task 2 |
| ***Lakshmi Prasanna Ramidi*** | Task 3 - Q7-Q12 \| Task 2 |
| ***Karella Rupa Naga Prasanna Raji*** | Task 4 - Q1-Q6 \| Task 2 |
| ***Sarah Ruth Oommen*** | Task 4 - Q7-Q12 \| Task 2 |
| ***Shaik Nafisa Kowsar*** | Task 4 - Q13,14 \| Task 5 - Q1-Q4 \| Task 2 |
| ***Swarna Dhevi R*** | Task 5 - Q5-Q10 \| Task 2 |

## TASK 1: Database Design (Normalisation):

**1. Create the database named "HMBank"**

**QUERY:** CREATE DATABASE HMBank;
USE HMBank;

**2. Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema.**

**QUERY:**

```
CREATE TABLE Customers (
   customer_id INT PRIMARY KEY,
   full_name VARCHAR(100),
   DOB DATE,
   contact_info VARCHAR(255),
   address VARCHAR(255)
);
CREATE TABLE Accounts (
   account_id INT PRIMARY KEY,
   customer_id INT,
   full_name VARCHAR(100),
   account_type VARCHAR(20),
   balance DECIMAL(10, 2),
   DOB DATE,
   contact_info VARCHAR(255),
   address VARCHAR(255),
   FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
CREATE TABLE Transactions (
   transaction_id INT PRIMARY KEY,
   account_id INT,
   full_name VARCHAR(100),
   transaction_type VARCHAR(20),
   amount DECIMAL(10, 2),
   transaction_date DATE,
   account_type VARCHAR(20),
   balance DECIMAL(10, 2),
   DOB DATE,
   contact_info VARCHAR(255),
   address VARCHAR(255),
   FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
```

**Output for 1 and 2:**
**(Tables created in such a way that it is not in normal form)**

```
create database HMBank;
use HMBank;
CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    full_name VARCHAR(100),
    DOB DATE,
    contact_info VARCHAR(255),
    address VARCHAR(255)
);
CREATE TABLE Accounts (
    account_id INT PRIMARY KEY,
    customer_id INT,
    full_name VARCHAR(100),
    account_type VARCHAR(20),
    balance DECIMAL(10, 2),
    DOB DATE,
    contact_info VARCHAR(255),
    address VARCHAR(255),
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
```

Output
Action Output

**Tables not in normal for**

```
select * from Customers;
select * from Accounts;
select * from Transactions;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| transaction_id | account_id | full_name | transaction_type | amount | transaction_date | account_type | balance | DOB | contact_info | address |
|---|---|---|---|---|---|---|---|---|---|---|
| 1001 | 1 | John Doe | deposit | 500.00 | 2023-01-05 | savings | 1000.00 | 1990-01-01 | 1234567890 | 123 Main St |
| 1002 | 2 | John Doe | withdrawal | 200.00 | 2023-02-10 | current | 500.00 | 1990-01-01 | 1234567890 | 123 Main St |
| 1003 | 3 | Jane Smith | transfer | 1000.00 | 2023-03-15 | savings | 2000.00 | 1985-05-15 | 9876543210 | 456 Oak Ave |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

**3. Perform the first three normal forms (1NF, 2NF, 3NF) analysis on the above tables.**

**QUERY FOR TABLES TO BE IN NORMAL FORM:**

```
CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    DOB DATE,
    email VARCHAR(100),
    phone_number VARCHAR(15),
    address VARCHAR(255)
);
CREATE TABLE accounts (
    account_id INT PRIMARY KEY,
    customer_id INT,
    account_type VARCHAR(20),
    balance DECIMAL(10, 2)
);
```

```
CREATE TABLE transactions (
    transaction_id INT PRIMARY KEY,
    account_id INT,
    transaction_type VARCHAR(20),
    amount DECIMAL(10, 2),
    transaction_date DATE
);
```

**INSERTING 20 VALUES IN EACH TABLE**

**1NF,2NF And 3NF Normalisation (Customers Table Output)**

| customer_id | first_name | last_name | DOB | email | phone_number | address |
|---|---|---|---|---|---|---|
| 1 | John | Doe | 1990-05-15 | john.doe@example.com | 123-456-7890 | 123 Main St |
| 2 | Jane | Smith | 1985-08-22 | jane.smith@example.com | 987-654-3210 | 456 Oak Ave |
| 3 | Bob | Johnson | 1978-12-10 | bob.johnson@example.com | 555-123-4567 | 789 Pine Ln |
| 4 | Alice | Williams | 1992-03-18 | alice.williams@example.com | 222-333-4444 | 567 Elm St |
| 5 | Charlie | Brown | 1980-09-05 | charlie.brown@example.com | 999-888-7777 | 890 Maple Ave |
| 6 | Eva | Davis | 1995-11-30 | eva.davis@example.com | 777-666-5555 | 901 Cedar Ln |
| 7 | Frank | Miller | 1987-07-12 | frank.miller@example.com | 444-555-6666 | 345 Birch St |
| 8 | Grace | Taylor | 1983-01-25 | grace.taylor@example.com | 666-777-8888 | 678 Pine Ave |
| 9 | Harry | Smith | 1975-06-08 | harry.smith@example.com | 111-222-3333 | 234 Oak Ln |
| 10 | Ivy | Anderson | 1998-04-20 | ivy.anderson@example.com | 333-444-5555 | 789 Birch Ave |
| 11 | Jack | Martin | 1989-02-14 | jack.martin@example.com | 777-888-9999 | 567 Maple St |
| 12 | Katherine | Jones | 1970-10-28 | katherine.jones@example.c... | 222-333-4444 | 901 Cedar Ave |
| 13 | Leo | Garcia | 1993-07-03 | leo.garcia@example.com | 555-666-7777 | 345 Pine Ln |
| 14 | Mia | Brown | 1982-04-15 | mia.brown@example.com | 999-888-7777 | 678 Elm Ave |
| 15 | Nathan | Anderson | 1973-09-22 | nathan.anderson@example... | 666-555-4444 | 890 Cedar St |
| 16 | Olivia | Taylor | 1996-12-05 | olivia.taylor@example.com | 444-333-2222 | 123 Birch Ave |
| 17 | Paul | Miller | 1984-05-18 | paul.miller@example.com | 111-222-3333 | 456 Pine Ln |
| 18 | Quinn | Smith | 1977-08-11 | quinn.smith@example.com | 888-999-0000 | 789 Maple St |
| 19 | Rachel | Davis | 1991-02-28 | rachel.davis@example.com | 222-111-0000 | 901 Elm Ave |
| 20 | Samuel | Johnson | 1986-06-14 | samuel.johnson@example.c... | 777-666-5555 | 345 Cedar Ln |

**1NF,2NF And 3NF Normalisation (Accounts Table Output)**

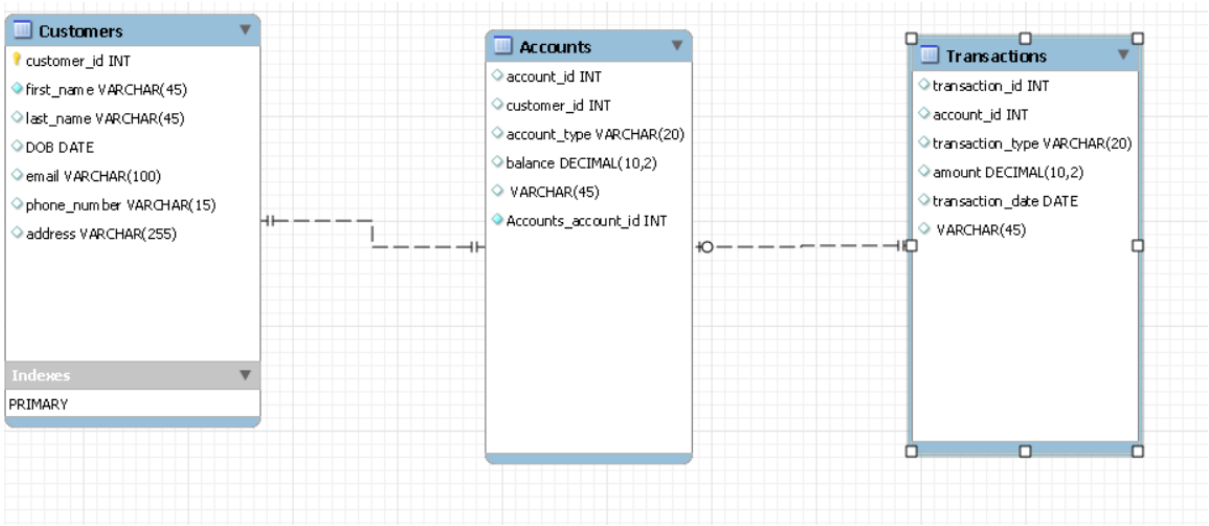
Result Grid | Filter Rows: | Edit: | Exp

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 1 | 101 | savings | 1500.00 |
| 2 | 102 | current | 500.00 |
| 3 | 103 | savings | 2000.00 |
| 4 | 104 | zero_balance | 0.00 |
| 5 | 105 | current | 1000.00 |
| 6 | 106 | savings | 2500.00 |
| 7 | 107 | savings | 1800.00 |
| 8 | 108 | current | 700.00 |
| 9 | 109 | zero_balance | 0.00 |
| 10 | 110 | savings | 3000.00 |
| 11 | 111 | current | 1200.00 |
| 12 | 112 | savings | 2200.00 |
| 13 | 113 | current | 800.00 |
| 14 | 114 | savings | 2700.00 |
| 15 | 115 | zero_balance | 0.00 |
| 16 | 116 | savings | 3200.00 |
| 17 | 117 | current | 1500.00 |
| 18 | 118 | savings | 2000.00 |
| 19 | 119 | current | 900.00 |
| 20 | 120 | savings | 3500.00 |

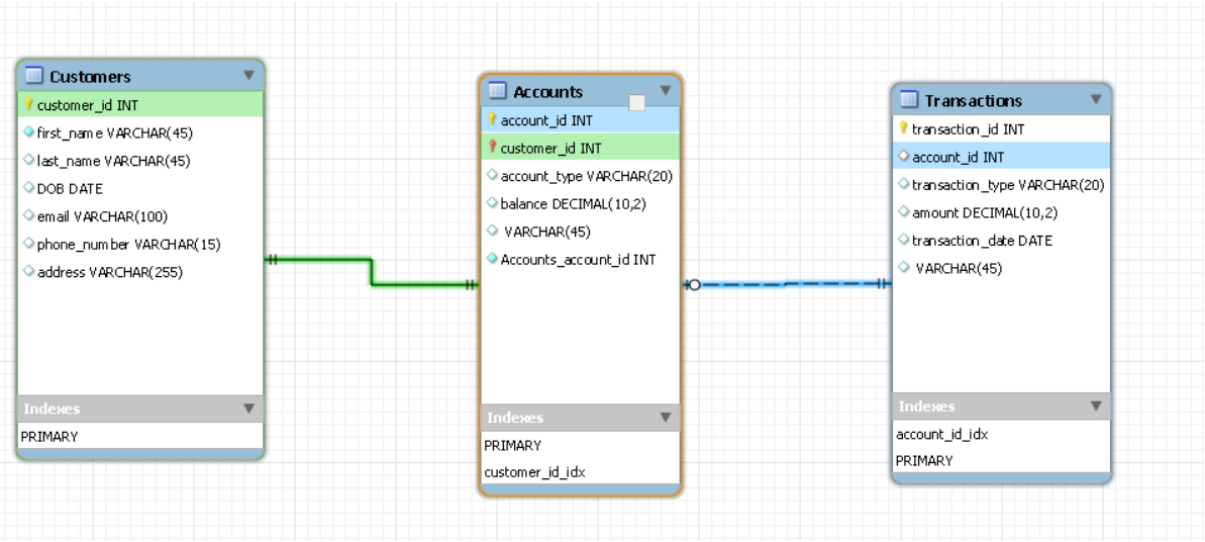
**1NF,2NF And 3NF Normalisation(Transactions Table Output)**

| transaction_id | account_id | transaction_type | amount | transaction_date |
|---|---|---|---|---|
| 1 | 1 | deposit | 500.00 | 2023-01-01 |
| 2 | 2 | withdrawal | 200.00 | 2023-01-02 |
| 3 | 3 | transfer | 1000.00 | 2023-01-03 |
| 4 | 4 | deposit | 300.00 | 2023-01-04 |
| 5 | 5 | withdrawal | 150.00 | 2023-01-05 |
| 6 | 6 | transfer | 700.00 | 2023-01-06 |
| 7 | 7 | deposit | 800.00 | 2023-01-07 |
| 8 | 8 | withdrawal | 400.00 | 2023-01-08 |
| 9 | 9 | transfer | 2000.00 | 2023-01-09 |
| 10 | 10 | deposit | 1000.00 | 2023-01-10 |
| 11 | 11 | withdrawal | 500.00 | 2023-01-11 |
| 12 | 12 | transfer | 1200.00 | 2023-01-12 |
| 13 | 13 | deposit | 600.00 | 2023-01-13 |
| 14 | 14 | withdrawal | 300.00 | 2023-01-14 |
| 15 | 15 | transfer | 1500.00 | 2023-01-15 |
| 16 | 16 | deposit | 2000.00 | 2023-01-16 |
| 17 | 17 | withdrawal | 700.00 | 2023-01-17 |
| 18 | 18 | transfer | 800.00 | 2023-01-18 |
| 19 | 19 | deposit | 1200.00 | 2023-01-19 |
| 20 | 20 | withdrawal | 500.00 | 2023-01-20 |

**4. Create an ERD (Entity Relationship Diagram) for the database.**



**5. Create appropriate Primary Key and Foreign Key constraints for referential integrity.**



## Task 2: Data Definition Language (DDL):

**1. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.**

- **Customers**
- **Accounts**
- **Transactions**

**Query-**

```
-- Customers Table
CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    DOB DATE,
    email VARCHAR(100),
    phone_number VARCHAR(15),
    address VARCHAR(255)
);

-- Accounts Table
CREATE TABLE Accounts (
    account_id INT PRIMARY KEY,
    customer_id INT,
    account_type VARCHAR(20),
    balance DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

-- Transactions Table
CREATE TABLE Transactions (
    transaction_id INT PRIMARY KEY,
    account_id INT,
    transaction_type VARCHAR(20),
    amount DECIMAL(10, 2),
    transaction_date DATE,
    FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
```

**Output**

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 19 | 11:16:06 | CREATE TABLE Customers ( customer_id INT PRIMARY KEY, first_name VARCHAR(5... | 0 row(s) affected |
| 20 | 11:16:06 | CREATE TABLE Accounts ( account_id INT PRIMARY KEY, customer_id INT, acco... | 0 row(s) affected |
| 21 | 11:16:06 | CREATE TABLE Transactions ( transaction_id INT PRIMARY KEY, account_id INT, ... | 0 row(s) affected |

## Task 3: Data Manipulation Language (DML):

**1. Insert at least 10 sample records into each of the following tables.**

- **Customers**
- **Accounts**
- **Transactions**

**Query-**

**Inserting records into the Customers table**

INSERT INTO Customers (customer_id, first_name, last_name, DOB, email, phone_number, address) VALUES
(1, 'John', 'Doe', '1990-05-15', 'john.doe@example.com', '123-456-7890', '123 Main St'),
(2, 'Jane', 'Smith', '1985-08-22', 'jane.smith@example.com', '987-654-3210', '456 Oak Ave'),
(3, 'Bob', 'Johnson', '1978-12-10', 'bob.johnson@example.com', '555-123-4567', '789 Pine Ln'),
(4, 'Alice', 'Williams', '1992-03-18', 'alice.williams@example.com', '222-333-4444', '567 Elm St'),
(5, 'Charlie', 'Brown', '1980-09-05', 'charlie.brown@example.com', '999-888-7777', '890 Maple Ave'),
(6, 'Eva', 'Davis', '1995-11-30', 'eva.davis@example.com', '777-666-5555', '901 Cedar Ln'),
(7, 'Frank', 'Miller', '1987-07-12', 'frank.miller@example.com', '444-555-6666', '345 Birch St'),
(8, 'Grace', 'Taylor', '1983-01-25', 'grace.taylor@example.com', '666-777-8888', '678 Pine Ave'),
(9, 'Harry', 'Smith', '1975-06-08', 'harry.smith@example.com', '111-222-3333', '234 Oak Ln'),
(10, 'Ivy', 'Anderson', '1998-04-20', 'ivy.anderson@example.com', '333-444-5555', '789 Birch Ave'),
(11, 'Jack', 'Martin', '1989-02-14', 'jack.martin@example.com', '777-888-9999', '567 Maple St'),
(12, 'Katherine', 'Jones', '1970-10-28', 'katherine.jones@example.com', '222-333-4444', '901 Cedar Ave'),
(13, 'Leo', 'Garcia', '1993-07-03', 'leo.garcia@example.com', '555-666-7777', '345 Pine Ln'),
(14, 'Mia', 'Brown', '1982-04-15', 'mia.brown@example.com', '999-888-7777', '678 Elm Ave'),
(15, 'Nathan', 'Anderson', '1973-09-22', 'nathan.anderson@example.com', '666-555-4444', '890 Cedar St'),
(16, 'Olivia', 'Taylor', '1996-12-05', 'olivia.taylor@example.com', '444-333-2222', '123 Birch Ave'),
(17, 'Paul', 'Miller', '1984-05-18', 'paul.miller@example.com', '111-222-3333', '456 Pine Ln'),
(18, 'Quinn', 'Smith', '1977-08-11', 'quinn.smith@example.com', '888-999-0000', '789 Maple St'),
(19, 'Rachel', 'Davis', '1991-02-28', 'rachel.davis@example.com', '222-111-0000', '901 Elm Ave'),
(20, 'Samuel', 'Johnson', '1986-06-14', 'samuel.johnson@example.com', '777-666-5555', '345 Cedar Ln');

**Inserting records into the accounts table**

```
INSERT INTO accounts (account_id, customer_id, account_type, balance) VALUES
(1, 1, 'savings', 1500.00),
(2, 2, 'current', 500.00),
(3, 3, 'savings', 2000.00),
(4, 4, 'zero_balance', 0.00),
(5, 5, 'current', 1000.00),
(6, 6, 'savings', 2500.00),
(7, 7, 'savings', 1800.00),
(8, 8, 'current', 700.00),
(9, 9, 'zero_balance', 0.00),
(10, 10, 'savings', 3000.00),
(11, 11, 'current', 1200.00),
(12, 12, 'savings', 2200.00),
(13, 13, 'current', 800.00),
(14, 14, 'savings', 2700.00),
(15, 15, 'zero_balance', 0.00),
(16, 16, 'savings', 3200.00),
(17, 17, 'current', 1500.00),
(18, 18, 'savings', 2000.00),
(19, 19, 'current', 900.00),
(20, 20, 'savings', 3500.00);
```

**Inserting records into the transactions table**

```
INSERT INTO transactions (transaction_id, account_id, transaction_type, amount,
transaction_date) VALUES
(1, 1, 'deposit', 500.00, '2023-01-01'),
(2, 2, 'withdrawal', 200.00, '2023-01-02'),
(3, 3, 'transfer', 1000.00, '2023-01-03'),
(4, 4, 'deposit', 300.00, '2023-01-04'),
(5, 5, 'withdrawal', 150.00, '2023-01-05'),
(6, 6, 'transfer', 700.00, '2023-01-06'),
(7, 7, 'deposit', 800.00, '2023-01-07'),
(8, 8, 'withdrawal', 400.00, '2023-01-08'),
(9, 9, 'transfer', 2000.00, '2023-01-09'),
(10, 10, 'deposit', 1000.00, '2023-01-10'),
(11, 11, 'withdrawal', 500.00, '2023-01-11'),
(12, 12, 'transfer', 1200.00, '2023-01-12'),
(13, 13, 'deposit', 600.00, '2023-01-13'),
(14, 14, 'withdrawal', 300.00, '2023-01-14'),
(15, 15, 'transfer', 1500.00, '2023-01-15'),
(16, 16, 'deposit', 2000.00, '2023-01-16'),
(17, 17, 'withdrawal', 700.00, '2023-01-17'),
(18, 18, 'transfer', 800.00, '2023-01-18'),
(19, 19, 'deposit', 1200.00, '2023-01-19'),
(20, 20, 'withdrawal', 500.00,'2023-01-20');
```

## Output

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 22 | 11:25:00 | INSERT INTO Customers (customer_id, first_name, last_name, DOB, email, phone_number, ... | 20 row(s) affected Records: 20 Duplicates: 0 Warnings: 0 |
| 23 | 11:25:32 | INSERT INTO accounts (account_id, customer_id, account_type, balance) VALUES (1, 1, '... | 20 row(s) affected Records: 20 Duplicates: 0 Warnings: 0 |
| 24 | 11:26:16 | select * from transactions LIMIT 0, 1000 | 0 row(s) returned |
| 25 | 11:26:40 | INSERT INTO transactions (transaction_id, account_id, transaction_type, amount, transactio... | 20 row(s) affected Records: 20 Duplicates: 0 Warnings: 0 |

## 2. Write SQL queries for the following tasks:

## 1. Write a SQL query to retrieve the name, account type and email of all customers.

```sql
-- Question 1
select customers.first_name,customers.last_name,accounts.account_type , customers.email from customers
inner join accounts on customers.customer_id=accounts.customer_id ;

-- Question 2
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| first_name | last_name | account_type | email |
|---|---|---|---|
| John | Doe | savings | john.doe@example.com |
| Jane | Smith | current | jane.smith@example.com |
| Bob | Johnson | savings | bob.johnson@example.com |
| Alice | Williams | zero_balance | alice.williams@example.com |
| Charlie | Brown | current | charlie.brown@example.com |

## 2.Write a SQL query to list all transaction corresponding customers.

```sql
SELECT
    Transactions.transaction_id,
    Customers.first_name,
    Customers.last_name,
    Transactions.transaction_type,
    Transactions.amount,
    Transactions.transaction_date
FROM
    Transactions
JOIN
    Accounts ON Transactions.account_id = Accounts.account_id
JOIN
    Customers ON Accounts.customer_id = Customers.customer_id;


-- Question 3 Write a SQL query to increase the balance of a specific account by a certain amount
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| transaction_id | first_name | last_name | transaction_type | amount | transaction_date |
|---|---|---|---|---|---|
| 1 | John | Doe | deposit | 500.00 | 2023-01-01 |
| 2 | Jane | Smith | withdrawal | 200.00 | 2023-01-02 |
| 3 | Bob | Johnson | transfer | 1000.00 | 2023-01-03 |
| 4 | Alice | Williams | deposit | 300.00 | 2023-01-04 |
| 5 | Charlie | Brown | withdrawal | 150.00 | 2023-01-05 |

### 3.Write a SQL query to increase the balance of a specific account by a certain amount.

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 1 | 1 | savings | 1500.00 |
| 2 | 2 | current | 500.00 |
| 3 | 3 | savings | 2000.00 |
| 4 | 4 | zero_balance | 0.00 |
| 5 | 5 | current | 1000.00 |
| 6 | 6 | savings | 2500.00 |

```sql
-- Question 3 Write a SQL query to increase the balance of a specific account by a certain amount.
UPDATE Accounts
SET balance = balance + 500.00
WHERE account_id = 1;

select * from accounts;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 1 | 1 | savings | 2000.00 |
| 2 | 2 | current | 500.00 |
| 3 | 3 | savings | 2000.00 |
| 4 | 4 | zero_balance | 0.00 |
| 5 | 5 | current | 1000.00 |
| 6 | 6 | savings | 2500.00 |

### 4.Write a SQL query to Combine first and last names of customers as a full_name.

```sql
-- Question 4 Write a SQL query to Combine first and last names of customers as a full_name.
SELECT
    CONCAT(first_name, ' ', last_name) AS full_name
FROM
    Customers;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| full_name |
|---|
| John Doe |
| Jane Smith |
| Bob Johnson |
| Alice Williams |
| Charlie Brown |
| Eva Davis |

### 5.Write a SQL query to remove accounts with a balance of zero where the account type is savings.

```sql
-- Question 5 Write a SQL query to remove accounts with a balance of zero where the account type is saving
Delete  FROM accounts
WHERE balance = 0 AND account_type = 'savings';

select * from accounts;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 1 | 1 | savings | 2000.00 |
| 2 | 2 | current | 500.00 |
| 3 | 3 | savings | 2000.00 |
| 4 | 4 | current | 0.00 |
| 5 | 5 | current | 1000.00 |
| 6 | 6 | savings | 2500.00 |

Result Grid | Form Editor

**6.Write a SQL query to Find customers living in a specific city.**

```
-- Question 6 Write a SQL query to Find customers living in a specific city.
SELECT * FROM Customers1
WHERE city = 'Mumbai';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| customer_id | first_name | last_name | DOB | email | phone_number | address | city |
|---|---|---|---|---|---|---|---|
| 1 | John | Doe | 1990-05-15 | john.doe@example.com | 1234567890 | 123 Main St | Mumbai |
| 19 | Rachel | Ward | 1988-06-23 | rachel.ward@example.com | 3456789012 | 1616 Elm St | Mumbai |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

**7: Write a SQL query to Get the account balance for a specific account.**

**Query:** select balance from accounts where account_id = 9;

**Output:**

Limit to 1000 rows

```
(20, 20, 'withdrawal', 500.00, '2023-01-20');
select *from transactions;

-- 7. Write a SQL query to Get the account balance for a specific account.
select balance from accounts where account_id = 8;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| balance |
|---|
| 700.00 |

accounts 27

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 59 | 12:01:22 | select *from Customers1 where city <> 'Mumbai' LIMIT 0, 1000 | 18 row(s) returned |
| 60 | 12:06:10 | select balance from accounts where account_id = 9 LIMIT 0, 1000 | 1 row(s) returned |
| 61 | 12:06:20 | select balance from accounts where account_id = 8 LIMIT 0, 1000 | 1 row(s) returned |

**8. Write a SQL query to List all current accounts with a balance greater than $1,000.**

**Query:** select account_id, customer_id, account_type, balance from accounts where account_type = 'current' AND balance > 1000.00;

**Output:**

```
-- 8. Write a SQL query to List all current accounts with a balance greater than $1,000.
select account_id, customer_id, account_type, balance from accounts
where account_type = 'current' AND balance > 1000.00;

-- 9. Write a SQL query to Retrieve all transactions for a specific account.
select * from transactions where account_id=19;
```

Result Grid

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 11 | 11 | current | 1200.00 |
| 17 | 17 | current | 1500.00 |
| NULL | NULL | NULL | NULL |

accounts 28

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 60 | 12:06:10 | select balance from accounts where account_id = 9 LIMIT 0, 1000 | 1 row(s) returned |
| 61 | 12:06:20 | select balance from accounts where account_id = 8 LIMIT 0, 1000 | 1 row(s) returned |
| 62 | 12:10:16 | select account_id, customer_id, account_type, balance from accounts where account_type = 'current' AND ba... | 2 row(s) returned |

**9. Write a SQL query to Retrieve all transactions for a specific account.**

**Query:** select * from transactions where account_id=19;

**Output:**

```
select account_id, customer_id, account_type, balance from accounts
where account_type = 'current' AND balance > 1000.00;

-- 9. Write a SQL query to Retrieve all transactions for a specific account.
select * from transactions where account_id=19;
```

Result Grid

| transaction_id | account_id | transaction_type | amount | transaction_date |
|---|---|---|---|---|
| 19 | 19 | deposit | 1200.00 | 2023-01-19 |
| NULL | NULL | NULL | NULL | NULL |

**10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.**

**Query:** select account_id,customer_id,account_type,balance,balance*0.05 interest_accured from accounts where account_type='savings';

**Output:**

```
select * from transactions where account_id=19;

-- 10 Write a SQL query to Calculate the interest accrued on savings accounts based on a
--  Assuming the interest rate is 5%
select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured
from accounts where account_type='savings';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| account_id | customer_id | account_type | balance | interest_accured |
|---|---|---|---|---|
| 1 | 1 | savings | 1500.00 | 75.0000 |
| 3 | 3 | savings | 2000.00 | 100.0000 |
| 6 | 6 | savings | 2500.00 | 125.0000 |
| 7 | 7 | savings | 1800.00 | 90.0000 |
| 10 | 10 | savings | 3000.00 | 150.0000 |
| 12 | 12 | savings | 2200.00 | 110.0000 |
| 14 | 14 | savings | 2700.00 | 135.0000 |
| 16 | 16 | savings | 3200.00 | 160.0000 |
| 18 | 18 | savings | 2000.00 | 100.0000 |
| 20 | 20 | savings | 3500.00 | 175.0000 |

Result 33

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 65 | 12:20:42 | select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured from accounts where ... | 10 row(s) returned |
| 66 | 12:22:06 | select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured from accounts where ... | 10 row(s) returned |
| 67 | 12:22:47 | select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured from accounts where ... | 10 row(s) returned |

**11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit.**

**Query:** select * from accounts where balance< 500;

**Output:**

```
from accounts where account_type='savings';

-- 11 . Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit.
-- assuming overdraft limit is 500$
select * from accounts where balance< 500;
```

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 4 | 4 | zero_balance | 0.00 |
| 9 | 9 | zero_balance | 0.00 |
| 15 | 15 | zero_balance | 0.00 |
| NULL | NULL | NULL | NULL |

Output

| # | Time | Action | Message |
|---|---|---|---|
| 66 | 12:22:06 | select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured from accounts where ... | 10 row(s) returned |
| 67 | 12:22:47 | select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured from accounts where ... | 10 row(s) returned |
| 68 | 12:24:17 | select * from accounts where balance< 500 LIMIT 0, 1000 | 3 row(s) returned |

**12. Write a SQL query to Find customers not living in a specific city.**

**Query:** select *from Customers1 where city <> 'Mumbai';

**Output:**

```
-- 12 Write a SQL query to Find customers not living in a specific city.
select *from Customers1 where city <> 'Mumbai';
```

Result Grid

| customer_id | first_name | last_name | DOB | email | phone_number | address | city |
|---|---|---|---|---|---|---|---|
| 2 | Jane | Smith | 1985-08-22 | jane.smith@example.com | 9876543210 | 456 Oak St | Delhi |
| 3 | Bob | Johnson | 1992-03-10 | bob.johnson@example.com | 8765432109 | 789 Maple St | Bangalore |
| 4 | Alice | Williams | 1988-11-28 | alice.williams@example.com | 2345678901 | 101 Pine St | Hyderabad |
| 5 | Charlie | Brown | 1995-07-03 | charlie.brown@example.com | 7654321098 | 202 Cedar St | Chennai |
| 6 | Eva | Davis | 1993-09-18 | eva.davis@example.com | 3456789012 | 303 Birch St | Ghaziabad |
| 7 | Frank | Miller | 1987-12-09 | frank.miller@example.com | 7890123456 | 404 Elm St | Pune |
| 8 | Grace | Wilson | 1991-04-25 | grace.wilson@example.com | 2109876543 | 505 Walnut St | Jaipur |
| 9 | Henry | Taylor | 1986-06-14 | henry.taylor@example.com | 6789012345 | 606 Pine St | Lucknow |
| 10 | Ivy | Moore | 1994-02-07 | ivy.moore@example.com | 3210987654 | 707 Oak St | Nagpur |
| 11 | Jack | Allen | 1989-10-31 | jack.allen@example.com | 5678901234 | 808 Maple St | Indore |
| 12 | Katherine | Clark | 1996-01-16 | katherine.clark@example.com | 4321098765 | 909 Cedar St | Thane |
| 13 | Leo | Martin | 1984-08-03 | leo.martin@example.com | 8765432109 | 1010 Elm St | Bhopal |
| 14 | Mia | Carter | 1997-03-20 | mia.carter@example.com | 1234567890 | 1111 Walnut St | Visakhapa... |
| 15 | Nathan | Young | 1983-05-12 | nathan.young@example.com | 9876543210 | 1212 Pine St | Pimpri-Chi... |
| 16 | Olivia | Hill | 1990-09-28 | olivia.hill@example.com | 8765432109 | 1313 Oak St | Patna |
| 17 | Peter | Cooper | 1985-11-05 | peter.cooper@example.com | 2345678901 | 1414 Maple St | Vadodara |
| 18 | Quinn | Gomez | 1992-02-18 | quinn.gomez@example.com | 7654321098 | 1515 Cedar St | Ghaziabad |
| 20 | Samuel | Fisher | 1995-12-10 | samuel.fisher@example.com | 2109876543 | 1717 Walnut St | Delhi |

Customers1 35

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 67 | 12:22:47 | select account_id,customer_id,account_type,balance,balance*0.05 as interest_accured from accounts where ... | 10 row(s) returned |
| 68 | 12:24:17 | select * from accounts where balance< 500 LIMIT 0, 1000 | 3 row(s) returned |
| 69 | 12:25:47 | select *from Customers1 where city <> 'Mumbai' LIMIT 0, 1000 | 18 row(s) returned |

## Task 4: Aggregate functions, GroupBy and Joins:

**1. Write a SQL query to Find the average account balance for all customers.**

**Query-** SELECT  AVG(balance) AS average_balance FROM Accounts;

**Output**

```
-- 1. average account balance for all customers
SELECT  AVG(balance) AS average_balance FROM Accounts;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| average_balance |
|---|
| 1550.000000 |

Result 8 | Read On

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 28 | 11:38:29 | select * from accounts LIMIT 0, 1000 | 20 row(s) returned |
| 29 | 11:38:35 | SELECT  AVG(balance) AS average_balance FROM Accounts LIMIT 0, 1000 | 1 row(s) returned |

**2. Write a SQL query to Retrieve the top 10 highest account balances.**

**Query-** SELECT * FROM Accounts ORDER BY balance DESC LIMIT 10;

**Output**

```
-- 2. retrieve the top 10 highest account balances
SELECT * FROM Accounts ORDER BY balance DESC LIMIT 10;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 20 | 20 | savings | 3500.00 |
| 16 | 16 | savings | 3200.00 |
| 10 | 10 | savings | 3000.00 |
| 14 | 14 | savings | 2700.00 |
| 6 | 6 | savings | 2500.00 |
| 12 | 12 | savings | 2200.00 |
| 3 | 3 | savings | 2000.00 |
| 18 | 18 | savings | 2000.00 |
| 7 | 7 | savings | 1800.00 |
| 1 | 1 | savings | 1500.00 |
| NULL | NULL | NULL | NULL |

Accounts 9 | Apply | Revert

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 29 | 11:38:35 | SELECT  AVG(balance) AS average_balance FROM Accounts LIMIT 0, 1000 | 1 row(s) returned |
| 30 | 11:44:32 | SELECT * FROM Accounts ORDER BY balance DESC LIMIT 10 | 10 row(s) returned |

**3. Write a SQL query to Calculate Total Deposits for All Customers on a specific date.**

**Query-**

```
SELECT
   c.customer_id,
   c.first_name,
   c.last_name,
   t.transaction_date,
   SUM(t.amount) AS total_deposits
FROM
   Customers c
JOIN
   Accounts a ON c.customer_id = a.customer_id
JOIN
   Transactions t ON a.account_id = t.account_id
WHERE
   t.transaction_type = 'deposit'
   AND t.transaction_date = '2023-01-10'
GROUP BY
   c.customer_id, c.first_name, c.last_name;
```

**Output**

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| customer_id | first_name | last_name | transaction_date | total_deposits |
|---|---|---|---|---|
| 10 | Ivy | Anderson | 2023-01-10 | 1000.00 |

Result 18 × Read Onl

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 45 | 12:06:58 | SELECT c.customer_id, c.first_name, c.last_name, SUM(t.amount) AS total_depo... | 1 row(s) returned |
| 46 | 12:07:51 | SELECT c.customer_id, c.first_name, c.last_name, t.transaction_date, SUM(t.a... | 1 row(s) returned |

**4. Write a SQL query to Find the Oldest and Newest Customers.**

**a. Find the oldest customer**

**Query-** SELECT * FROM customers LIMIT 1;

**Output**

```
-- 4. a)find oldest customer
SELECT * FROM customers LIMIT 1;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

| customer_id | first_name | last_name | DOB | email | phone_number | address |
|---|---|---|---|---|---|---|
| 1 | John | Doe | 1990-05-15 | john.doe@example.com | 123-456-7890 | 123 Main St |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

customers 21 × Ap

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 49 | 12:49:58 | SELECT * FROM customer LIMIT 1 | Error Code: 1146. Table 'hmbank.customer' doesn't exist |
| 50 | 12:50:17 | SELECT * FROM customers LIMIT 1 | 1 row(s) returned |

**b. Find the newest customer**

**Query-** SELECT * FROM customers ORDER BY customer_id DESC LIMIT 1;

**Output**

```
-- 4. b)find newest customer
SELECT * FROM customers ORDER BY customer_id DESC LIMIT 1;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

| customer_id | first_name | last_name | DOB | email | phone_number | address |
|---|---|---|---|---|---|---|
| 20 | Samuel | Johnson | 1986-06-14 | samuel.johnson@example.com | 777-666-5555 | 345 Cedar Ln |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

customers 22 × Apply

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 50 | 12:50:17 | SELECT * FROM customers LIMIT 1 | 1 row(s) returned |
| 51 | 12:54:08 | SELECT * FROM customers ORDER BY customer_id DESC LIMIT 1 | 1 row(s) returned |

**5. Write a SQL query to Retrieve transaction details along with the account type.**

**Query-**

```
SELECT
    t.transaction_id,
    t.account_id,
    t.transaction_type,
    t.amount,
    t.transaction_date,
    a.account_type
FROM
    Transactions t
JOIN
    Accounts a ON t.account_id = a.account_id;
```

**Output-**

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| transaction_id | account_id | transaction_type | amount | transaction_date | account_type |
|---|---|---|---|---|---|
| 1 | 1 | deposit | 500.00 | 2023-01-01 | savings |
| 2 | 2 | withdrawal | 200.00 | 2023-01-02 | current |
| 3 | 3 | transfer | 1000.00 | 2023-01-03 | savings |
| 4 | 4 | deposit | 300.00 | 2023-01-04 | zero_balance |
| 5 | 5 | withdrawal | 150.00 | 2023-01-05 | current |
| 6 | 6 | transfer | 700.00 | 2023-01-06 | savings |
| 7 | 7 | deposit | 800.00 | 2023-01-07 | savings |
| 8 | 8 | withdrawal | 400.00 | 2023-01-08 | current |
| 9 | 9 | transfer | 2000.00 | 2023-01-09 | zero_balance |
| 10 | 10 | deposit | 1000.00 | 2023-01-10 | savings |
| 11 | 11 | withdrawal | 500.00 | 2023-01-11 | current |
| 12 | 12 | transfer | 1200.00 | 2023-01-12 | savings |
| 13 | 13 | deposit | 600.00 | 2023-01-13 | current |
| 14 | 14 | withdrawal | 300.00 | 2023-01-14 | savings |
| 15 | 15 | transfer | 1500.00 | 2023-01-15 | zero_balance |

Result 19 | Read Only

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 46 | 12:07:51 | SELECT c.customer_id, c.first_name, c.last_name, t.transaction_date, SUM(t.a... | 1 row(s) returned |
| 47 | 12:21:38 | SELECT t.transaction_id, t.account_id, t.transaction_type, t.amount, t.transacti... | 20 row(s) returned |

**6. Write a SQL query to Get a list of customers along with their account details.**

**Query-**

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.DOB,
    c.email,
    c.phone_number,
    c.address,
    a.account_id,
    a.account_type,
    a.balance
FROM
    Customers c
JOIN Accounts a ON c.customer_id = a.customer_id;
```

**Output**

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| customer_id | first_name | last_name | DOB | email | phone_number | address | account_id | account_type | balance |
|---|---|---|---|---|---|---|---|---|---|
| 1 | John | Doe | 1990-05-15 | john.doe@example.com | 123-456-7890 | 123 Main St | 1 | savings | 1500.00 |
| 2 | Jane | Smith | 1985-08-22 | jane.smith@example.com | 987-654-3210 | 456 Oak Ave | 2 | current | 500.00 |
| 3 | Bob | Johnson | 1978-12-10 | bob.johnson@example.com | 555-123-4567 | 789 Pine Ln | 3 | savings | 2000.00 |
| 4 | Alice | Williams | 1992-03-18 | alice.williams@example.com | 222-333-4444 | 567 Elm St | 4 | zero_balance | 0.00 |
| 5 | Charlie | Brown | 1980-09-05 | charlie.brown@example.com | 999-888-7777 | 890 Maple Ave | 5 | current | 1000.00 |
| 6 | Eva | Davis | 1995-11-30 | eva.davis@example.com | 777-666-5555 | 901 Cedar Ln | 6 | savings | 2500.00 |
| 7 | Frank | Miller | 1987-07-12 | frank.miller@example.com | 444-555-6666 | 345 Birch St | 7 | savings | 1800.00 |
| 8 | Grace | Taylor | 1983-01-25 | grace.taylor@example.com | 666-777-8888 | 678 Pine Ave | 8 | current | 700.00 |
| 9 | Harry | Smith | 1975-06-08 | harry.smith@example.com | 111-222-3333 | 234 Oak Ln | 9 | zero_balance | 0.00 |
| 10 | Ivy | Anderson | 1998-04-20 | ivy.anderson@example.com | 333-444-5555 | 789 Birch Ave | 10 | savings | 3000.00 |
| 11 | Jack | Martin | 1989-02-14 | jack.martin@example.com | 777-888-9999 | 567 Maple St | 11 | current | 1200.00 |
| 12 | Katherine | Jones | 1970-10-28 | katherine.jones@example.c... | 222-333-4444 | 901 Cedar Ave | 12 | savings | 2200.00 |
| 13 | Leo | Garcia | 1993-07-03 | leo.garcia@example.com | 555-666-7777 | 345 Pine Ln | 13 | current | 800.00 |

Result 20 | Read Only

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 47 | 12:21:38 | SELECT t.transaction_id, t.account_id, t.transaction_type, t.amount, t.transacti... | 20 row(s) returned |
| 48 | 12:28:26 | SELECT c.customer_id, c.first_name, c.last_name, c.DOB, c.email, c.phone_... | 20 row(s) returned |

**7. Write a SQL query to Retrieve transaction details along with customer information for a specific account.**

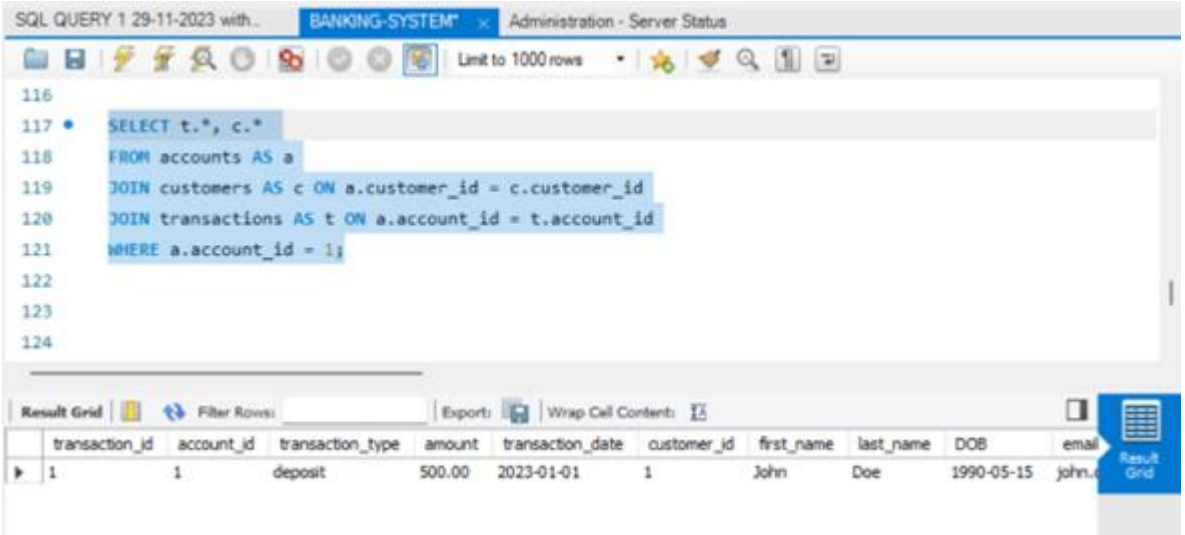
SELECT t.*, c.*

FROM accounts AS a

JOIN customers AS c ON a.customer_id = c.customer_id

JOIN transactions AS t ON a.account_id = t.account_id

WHERE a.account_id = 1;




**8. Write a SQL query to Identify customers who have more than one account.**

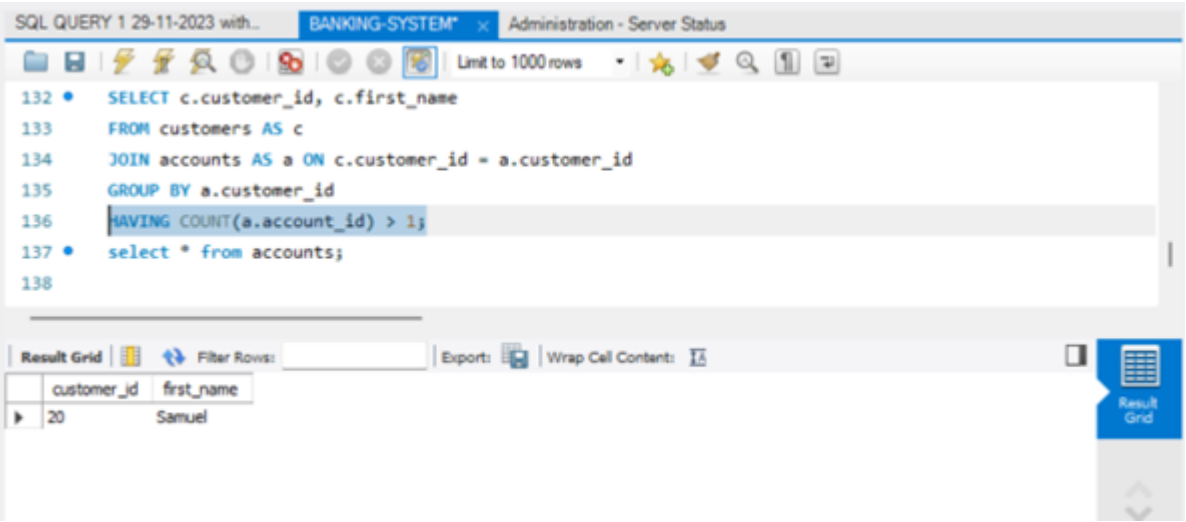
insert into accounts values(21,20,'current',1000);

SELECT c.customer_id, c.first_name

FROM customers AS c

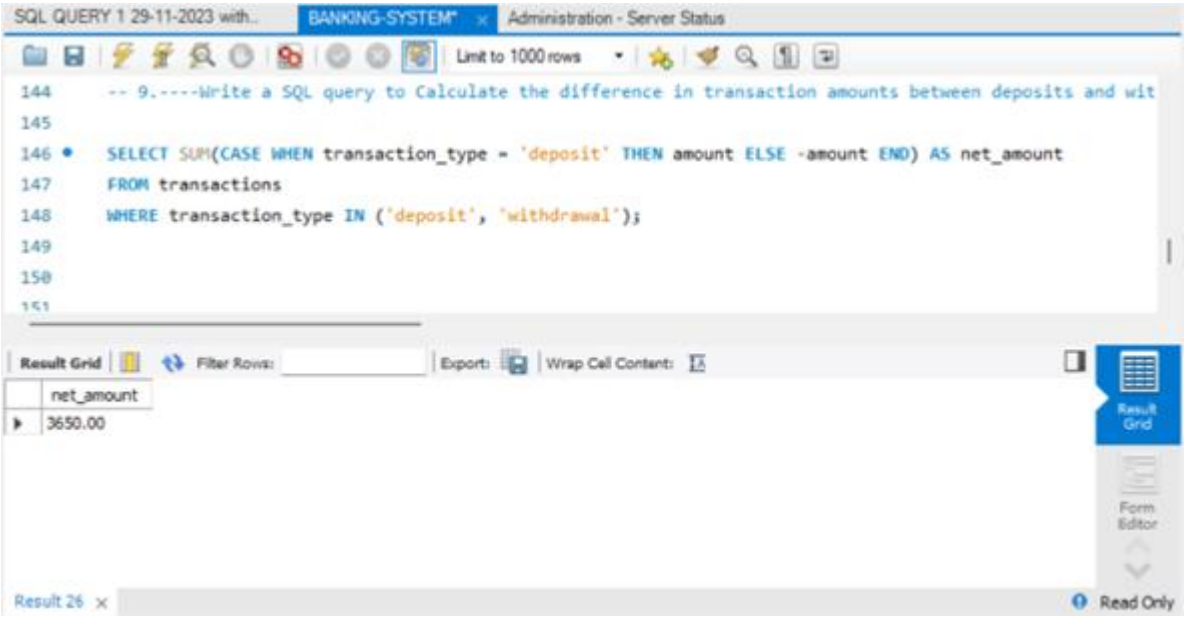
JOIN accounts AS a ON c.customer_id = a.customer_id

GROUP BY a.customer_id HAVING COUNT(a.account_id) > 1;




**9. Write a SQL query to Calculate the difference in transaction amounts between deposits and withdrawals.**

SELECT SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS net_amount FROM transactions WHERE transaction_type IN ('deposit', 'withdrawal');




**10. Write a SQL query to Calculate the average daily balance for each account over a specified period.**

SELECT

  account_id,

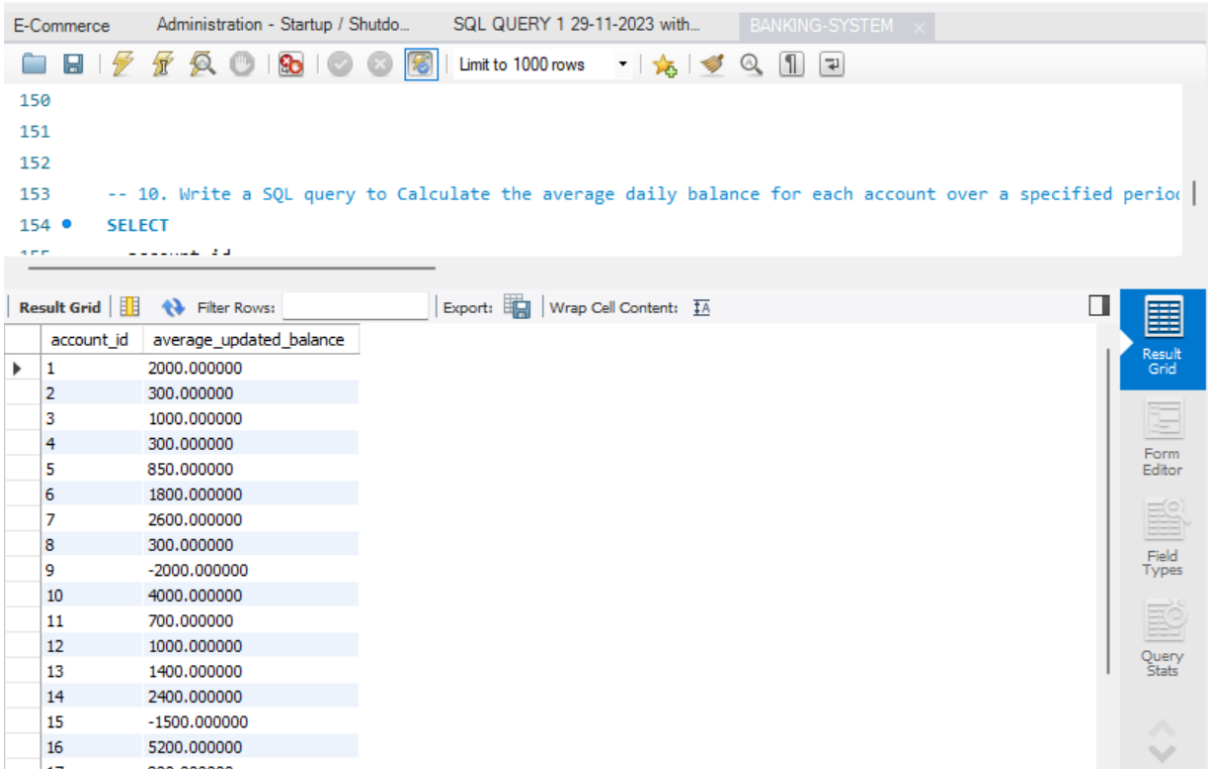
  AVG(updated_balance) AS average_updated_balance

FROM (

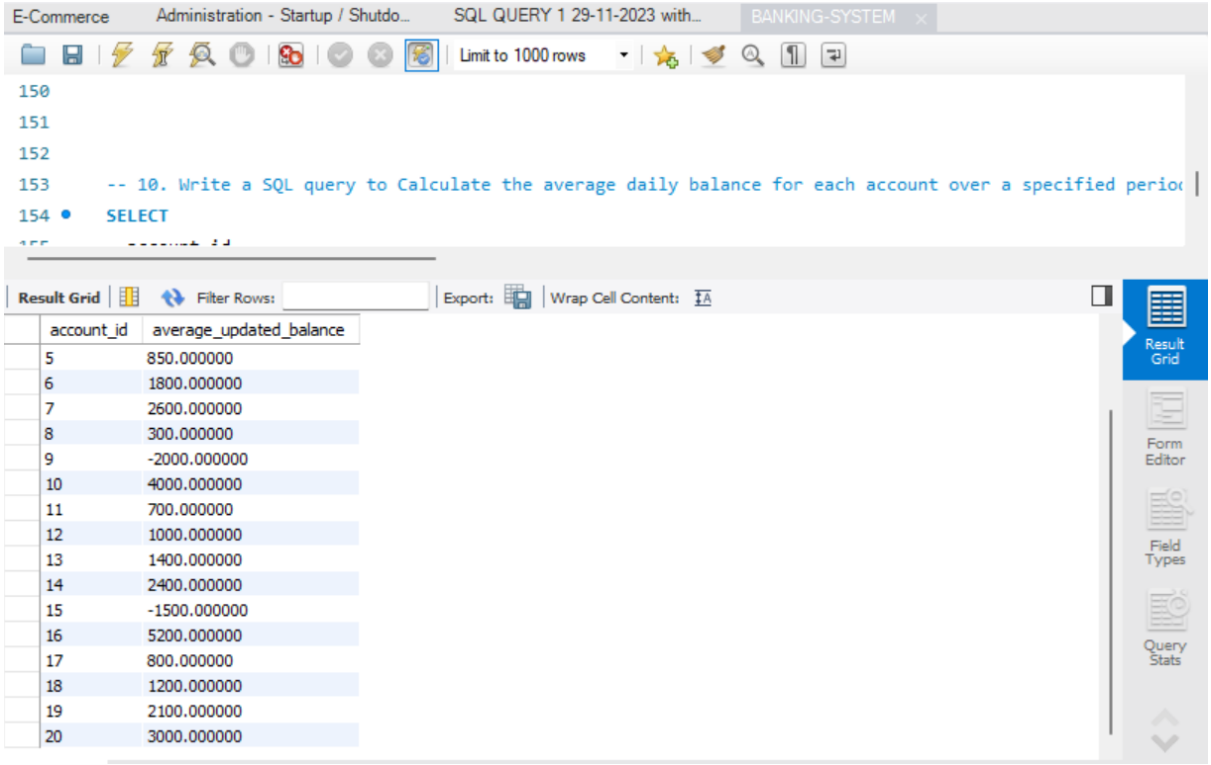
  SELECT

    Transactions.account_id,

```
    Transactions.transaction_type,

    Accounts.balance,

    Transactions.amount,

    Transactions.transaction_date,

    CASE

      WHEN Transactions.transaction_type = 'Deposit' THEN Accounts.balance +
Transactions.amount

      ELSE Accounts.balance - Transactions.amount

    END AS updated_balance

  FROM

    Transactions

  JOIN

    Accounts ON Transactions.account_id = Accounts.account_id

  WHERE

    Transactions.transaction_date BETWEEN '2023-01-01' AND '2023-01-20'

) AS updated_balances

GROUP BY account_id;
```

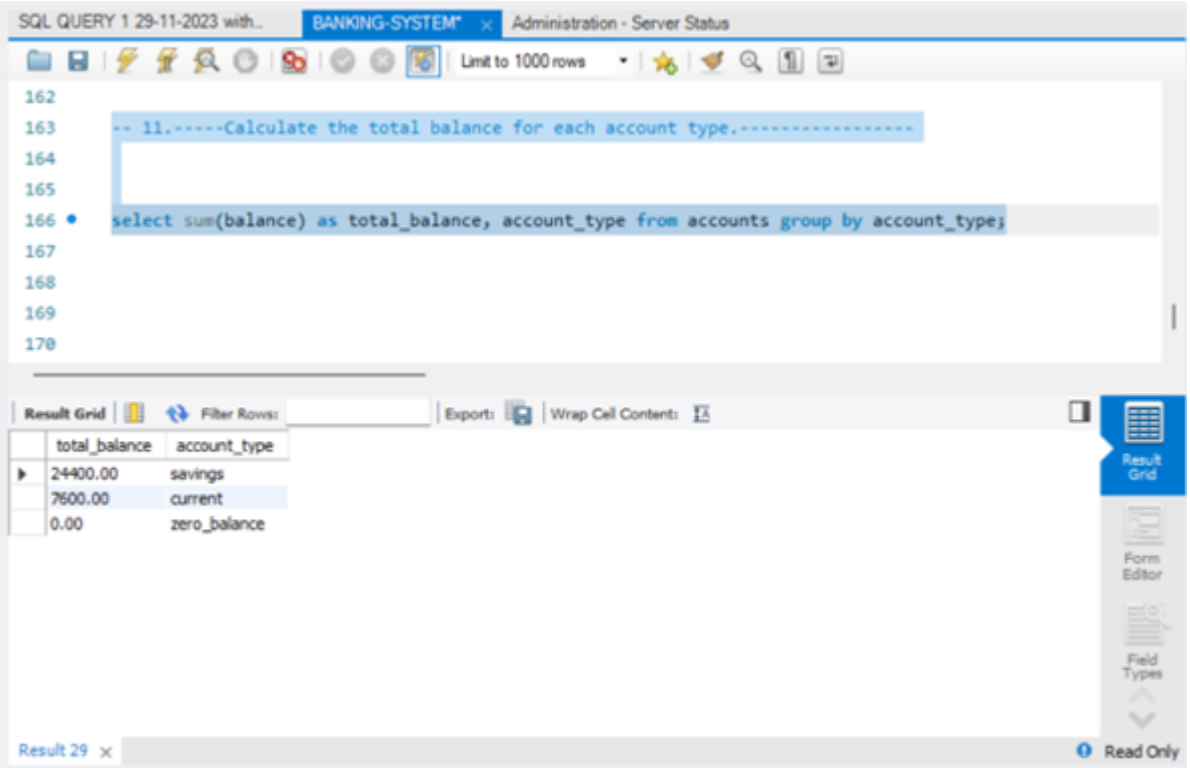



| account_id | average_updated_balance |
|---|---|
| 1 | 2000.000000 |
| 2 | 300.000000 |
| 3 | 1000.000000 |
| 4 | 300.000000 |
| 5 | 850.000000 |
| 6 | 1800.000000 |
| 7 | 2600.000000 |
| 8 | 300.000000 |
| 9 | -2000.000000 |
| 10 | 4000.000000 |
| 11 | 700.000000 |
| 12 | 1000.000000 |
| 13 | 1400.000000 |
| 14 | 2400.000000 |
| 15 | -1500.000000 |
| 16 | 5200.000000 |




**11. Calculate the total balance for each account type.**

select sum(balance) as total_balance, account_type from accounts group by account_type;



**12. Identify accounts with the highest number of transactions ordered by descending order.**

insert into transactions values (21,1,'withdrawal',200.00,'2023-05-01');

insert into transactions values (22,1,'deposit',2000.00,'2023-06-03');

insert into transactions values (23,1,'deposit',7000.00,'2023-07-04');

insert into transactions values (24,3,'transfer',5000.00,'2023-09-03');

insert into transactions values (25,7,'transfer',3000.00,'2023-10-06');

insert into transactions values (26,7,'withdrawal',1500.00,'2023-10-07');

delete from transactions value where transaction_id='21';

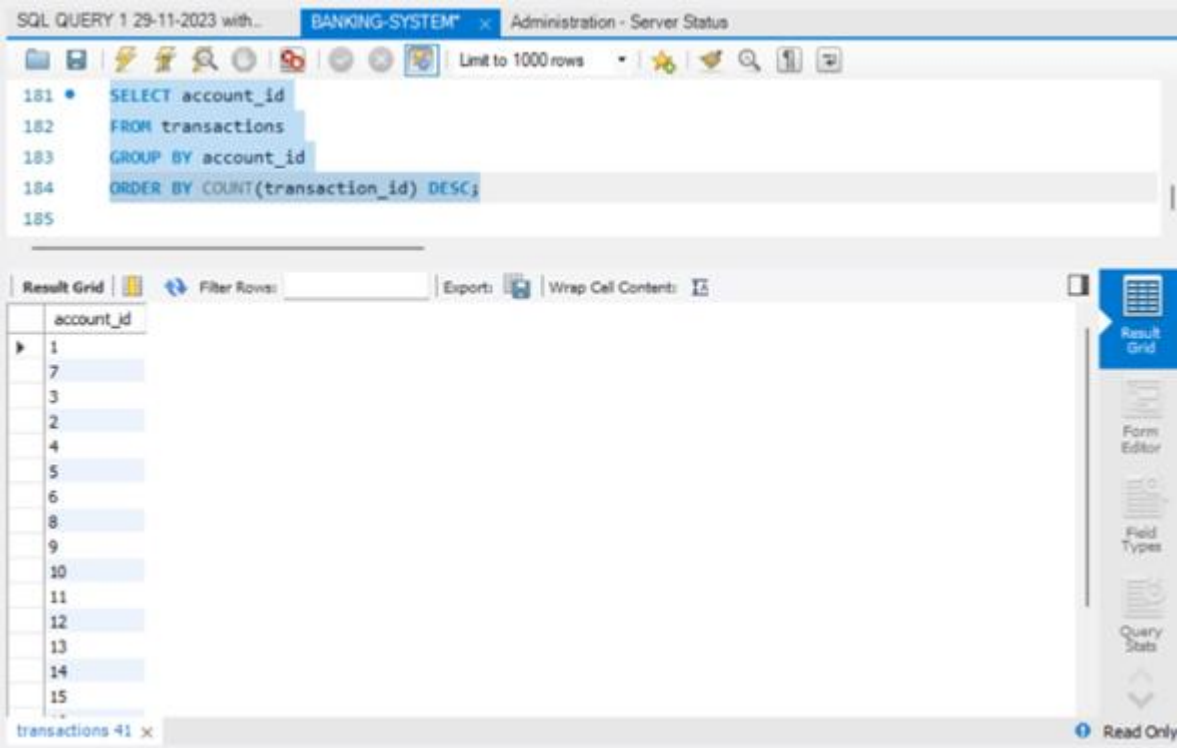
select * from transactions;

SELECT account_id

FROM transactions

GROUP BY account_id

ORDER BY COUNT(transaction_id) DESC;



**13. List customers with high aggregate account balances, along with their account types.**

**Query:**

SELECT c.customer_id, c.first_name, c.last_name, a.account_type, a.balance
FROM Customers c
JOIN accounts a ON c.customer_id = a.customer_id
WHERE a.balance >= 2000;

**Output:**

```
SELECT c.customer_id, c.first_name, c.last_name, a.account_type, a.balance
FROM Customers c
JOIN accounts a ON c.customer_id = a.customer_id
WHERE a.balance >= 2000;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| customer_id | first_name | last_name | account_type | balance |
|---|---|---|---|---|
| 3 | Bob | Johnson | savings | 2000.00 |
| 6 | Eva | Davis | savings | 2500.00 |
| 10 | Ivy | Anderson | savings | 3000.00 |
| 12 | Katherine | Jones | savings | 2200.00 |
| 14 | Mia | Brown | savings | 2700.00 |
| 16 | Olivia | Taylor | savings | 3200.00 |
| 18 | Quinn | Smith | savings | 2000.00 |
| 20 | Samuel | Johnson | savings | 3500.00 |

**14. Identify and list duplicate transactions based on transaction amount, date, and account.**

**Table Used:**

```
SELECT * FROM transactions;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| transaction_id | account_id | transaction_type | amount | transaction_date |
|---|---|---|---|---|
| 1 | 1 | deposit | 500.00 | 2023-01-01 |
| 2 | 2 | withdrawal | 200.00 | 2023-01-02 |
| 3 | 3 | transfer | 1000.00 | 2023-01-03 |
| 4 | 4 | deposit | 300.00 | 2023-01-04 |
| 5 | 5 | withdrawal | 150.00 | 2023-01-05 |
| 6 | 6 | transfer | 700.00 | 2023-01-06 |
| 7 | 7 | deposit | 800.00 | 2023-01-07 |
| 8 | 8 | withdrawal | 400.00 | 2023-01-08 |
| 9 | 9 | transfer | 2000.00 | 2023-01-09 |
| 10 | 10 | deposit | 1000.00 | 2023-01-10 |
| 11 | 11 | withdrawal | 500.00 | 2023-01-11 |
| 12 | 12 | transfer | 1200.00 | 2023-01-12 |
| 13 | 13 | deposit | 600.00 | 2023-01-13 |
| 14 | 14 | withdrawal | 300.00 | 2023-01-14 |
| 15 | 15 | transfer | 1500.00 | 2023-01-15 |
| 16 | 16 | deposit | 2000.00 | 2023-01-16 |
| 17 | 17 | withdrawal | 700.00 | 2023-01-17 |

| | | | | |
|---|---|---|---|---|
| 18 | 18 | transfer | 800.00 | 2023-01-18 |
| 19 | 19 | deposit | 1200.00 | 2023-01-19 |
| 20 | 20 | withdrawal | 500.00 | 2023-01-20 |
| 21 | 1 | deposit | 500.00 | 2023-01-01 |
| 22 | 2 | withdrawal | 200.00 | 2023-01-02 |
| 23 | 3 | transfer | 1000.00 | 2023-01-03 |
| NULL | NULL | NULL | NULL | NULL |

transactions 11

**Query:**

```
SELECT *
FROM transactions t1
JOIN transactions t2 ON t1.amount = t2.amount
                AND t1.transaction_date = t2.transaction_date
                AND t1.account_id = t2.account_id
                AND t1.transaction_id <> t2.transaction_id
ORDER BY t1.account_id, t1.transaction_id;
```

**Output:**

```
SELECT *
FROM transactions t1
JOIN transactions t2 ON t1.amount = t2.amount
                    AND t1.transaction_date = t2.transaction_date
                    AND t1.account_id = t2.account_id
                    AND t1.transaction_id <> t2.transaction_id
ORDER BY t1.account_id, t1.transaction_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| transaction_id | account_id | transaction_type | amount | transaction_date | transaction_id | account_id | transaction_type | amount | transaction_date |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | deposit | 500.00 | 2023-01-01 | 21 | 1 | deposit | 500.00 | 2023-01-01 |
| 21 | 1 | deposit | 500.00 | 2023-01-01 | 1 | 1 | deposit | 500.00 | 2023-01-01 |
| 2 | 2 | withdrawal | 200.00 | 2023-01-02 | 22 | 2 | withdrawal | 200.00 | 2023-01-02 |
| 22 | 2 | withdrawal | 200.00 | 2023-01-02 | 2 | 2 | withdrawal | 200.00 | 2023-01-02 |
| 3 | 3 | transfer | 1000.00 | 2023-01-03 | 23 | 3 | transfer | 1000.00 | 2023-01-03 |
| 23 | 3 | transfer | 1000.00 | 2023-01-03 | 3 | 3 | transfer | 1000.00 | 2023-01-03 |

## Task 5: Subquery

**1. Retrieve the customer(s) with the highest account balance.**

**Query:**

```
SELECT customer_id, first_name, last_name, balance
FROM (
   SELECT c.customer_id, c.first_name, c.last_name, a.balance
   FROM Customers c
   JOIN accounts a ON c.customer_id = a.customer_id
   ORDER BY a.balance DESC
   LIMIT 1
) AS subquery;
```

**Output:**

```
SELECT customer_id, first_name, last_name, balance
FROM (
    SELECT c.customer_id, c.first_name, c.last_name, a.balance
    FROM Customers c
    JOIN accounts a ON c.customer_id = a.customer_id
    ORDER BY a.balance DESC
    LIMIT 1
) AS subquery;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| customer_id | first_name | last_name | balance |
|---|---|---|---|
| 20 | Samuel | Johnson | 3500.00 |

**2. Calculate the average account balance for customers who have more than one account.**

**Table used: Modified Accounts Table**

```
354 •  SELECT * FROM accounts;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 1 | 1 | savings | 1500.00 |
| 2 | 2 | current | 500.00 |
| 3 | 3 | savings | 2000.00 |
| 4 | 4 | zero_balance | 0.00 |
| 5 | 5 | current | 1000.00 |
| 6 | 6 | savings | 2500.00 |
| 7 | 7 | savings | 1800.00 |
| 8 | 8 | current | 700.00 |
| 9 | 9 | zero_balance | 0.00 |
| 10 | 10 | savings | 3000.00 |
| 11 | 11 | current | 1200.00 |
| 12 | 12 | savings | 2200.00 |
| 13 | 13 | current | 800.00 |
| 14 | 14 | savings | 2700.00 |
| 15 | 15 | zero_balance | 0.00 |
| 16 | 16 | savings | 3200.00 |
| 17 | 17 | current | 1500.00 |
| 18 | 18 | savings | 2000.00 |
| 19 | 19 | current | 900.00 |
| 20 | 20 | savings | 3500.00 |
| 21 | 1 | current | 500.00 |
| 22 | 2 | savings | 1500.00 |
| 23 | 3 | current | 2500.00 |
| NULL | NULL | NULL | NULL |

**Query:**

```
SELECT c.customer_id, c.first_name, c.last_name,
       AVG(a.balance) as average_balance
FROM Customers c
JOIN accounts a ON c.customer_id = a.customer_id
WHERE c.customer_id IN
    (SELECT customer_id FROM accounts GROUP BY customer_id HAVING COUNT(*) > 1)
GROUP BY c.customer_id, c.first_name, c.last_name;
```

**Output:**

```
356 •  SELECT c.customer_id, c.first_name, c.last_name,
357            AVG(a.balance) as average_balance
358    FROM Customers c
359    JOIN accounts a ON c.customer_id = a.customer_id
360    WHERE c.customer_id IN
361        (SELECT customer_id FROM accounts GROUP BY customer_id HAVING COUNT(*) > 1)
362    GROUP BY c.customer_id, c.first_name, c.last_name;
363
364
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| customer_id | first_name | last_name | average_balance |
|---|---|---|---|
| 1 | John | Doe | 1000.000000 |
| 2 | Jane | Smith | 1000.000000 |
| 3 | Bob | Johnson | 2250.000000 |

**3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.**

**Query:**

```
SELECT a.account_id, a.customer_id, t.transaction_id, t.amount
FROM accounts a
JOIN transactions t ON a.account_id = t.account_id
WHERE t.amount > (SELECT AVG(amount) FROM transactions);
```

**Output:**

```
SELECT a.account_id, a.customer_id, t.transaction_id, t.amount
FROM accounts a
JOIN transactions t ON a.account_id = t.account_id
WHERE t.amount > (SELECT AVG(amount) FROM transactions);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| account_id | customer_id | transaction_id | amount |
|---|---|---|---|
| 3 | 3 | 3 | 1000.00 |
| 7 | 7 | 7 | 800.00 |
| 9 | 9 | 9 | 2000.00 |
| 10 | 10 | 10 | 1000.00 |
| 12 | 12 | 12 | 1200.00 |
| 15 | 15 | 15 | 1500.00 |
| 16 | 16 | 16 | 2000.00 |
| 18 | 18 | 18 | 800.00 |
| 19 | 19 | 19 | 1200.00 |

**4. Identify customers who have no recorded transactions.**
**Table Used:**
**Customers Table**

```
SELECT * FROM customers;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| customer_id | first_name | last_name | DOB | email | phone_number | address |
|---|---|---|---|---|---|---|
| 13 | Leo | Garcia | 1993-07-03 | leo.garcia@example.com | 555-666-7777 | 345 Pine Ln |
| 14 | Mia | Brown | 1982-04-15 | mia.brown@example.com | 999-888-7777 | 678 Elm Ave |
| 15 | Nathan | Anderson | 1973-09-22 | nathan.anderson@example... | 666-555-4444 | 890 Cedar St |
| 16 | Olivia | Taylor | 1996-12-05 | olivia.taylor@example.com | 444-333-2222 | 123 Birch Ave |
| 17 | Paul | Miller | 1984-05-18 | paul.miller@example.com | 111-222-3333 | 456 Pine Ln |
| 18 | Quinn | Smith | 1977-08-11 | quinn.smith@example.com | 888-999-0000 | 789 Maple St |
| 19 | Rachel | Davis | 1991-02-28 | rachel.davis@example.com | 222-111-0000 | 901 Elm Ave |
| 20 | Samuel | Johnson | 1986-06-14 | samuel.johnson@example.c... | 777-666-5555 | 345 Cedar Ln |
| 30 | Neeru | Rao | 1992-05-16 | neeru.rao@example.com | 127-756-0890 | 123 Beside St |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

**Accounts Table:**

```
SELECT * FROM accounts;
SELECT * FROM customers;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap

| account_id | customer_id | account_type | balance |
|---|---|---|---|
| 13 | 13 | current | 800.00 |
| 14 | 14 | savings | 2700.00 |
| 15 | 15 | zero_balance | 0.00 |
| 16 | 16 | savings | 3200.00 |
| 17 | 17 | current | 1500.00 |
| 18 | 18 | savings | 2000.00 |
| 19 | 19 | current | 900.00 |
| 20 | 20 | savings | 3500.00 |
| 21 | 30 | current | 2500.00 |
| NULL | NULL | NULL | NULL |

**Query:**

```
SELECT customer_id, first_name, last_name, DOB, email, phone_number, address
FROM Customers
WHERE customer_id NOT IN (
    SELECT c.customer_id
    FROM Customers c
    LEFT JOIN accounts a ON c.customer_id = a.customer_id
    LEFT JOIN transactions t ON a.account_id = t.account_id
    WHERE t.transaction_id IS NOT NULL
);
```

**Output:**

```
SELECT customer_id, first_name, last_name, DOB, email, phone_number, address
FROM Customers
WHERE customer_id NOT IN (
    SELECT c.customer_id
    FROM Customers c
    LEFT JOIN accounts a ON c.customer_id = a.customer_id
    LEFT JOIN transactions t ON a.account_id = t.account_id
    WHERE t.transaction_id IS NOT NULL
);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

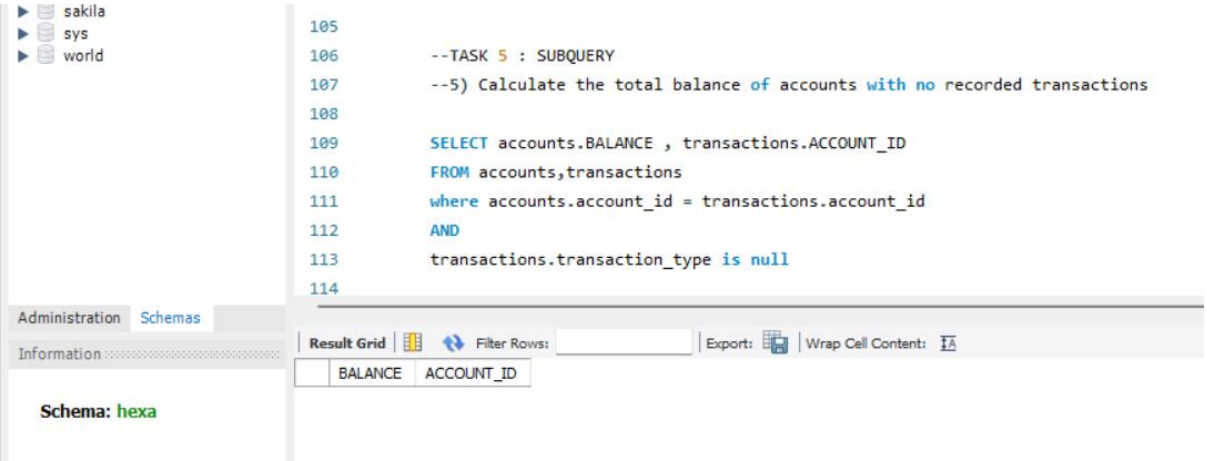
| customer_id | first_name | last_name | DOB | email | phone_number | address |
|---|---|---|---|---|---|---|
| 30 | Neeru | Rao | 1992-05-16 | neeru.rao@example.com | 127-756-0890 | 123 Beside St |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

**5.Calculate the total balance of accounts with no recorded transactions**

**QUERY:**

```
SELECT accounts.BALANCE , transactions.ACCOUNT_ID
FROM accounts,transactions
where accounts.account_id = transactions.account_id
AND
transactions.transaction_type is null
```

**OUTPUT:**

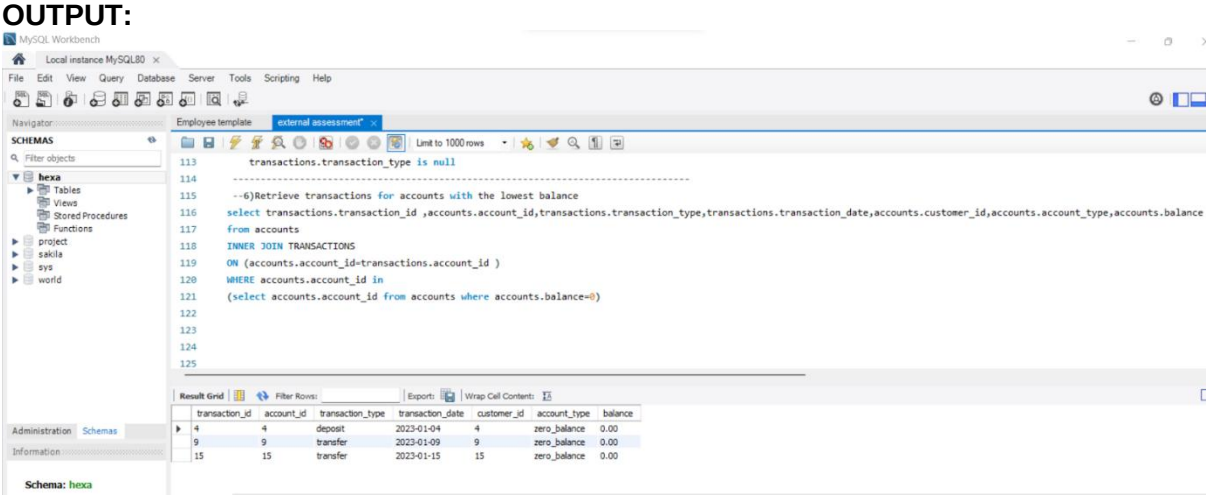



**6.Retrieve transactions for accounts with the lowest balance**

**QUERY:**

```
select transactions.transaction_id
,accounts.account_id,transactions.transaction_type,transactions.transaction_date,accounts.
customer_id,accounts.account_type,accounts.balance
from accounts
INNER JOIN TRANSACTIONS
ON (accounts.account_id=transactions.account_id )
WHERE accounts.account_id in
(select accounts.account_id from accounts where accounts.balance=0)
```

**OUTPUT:**

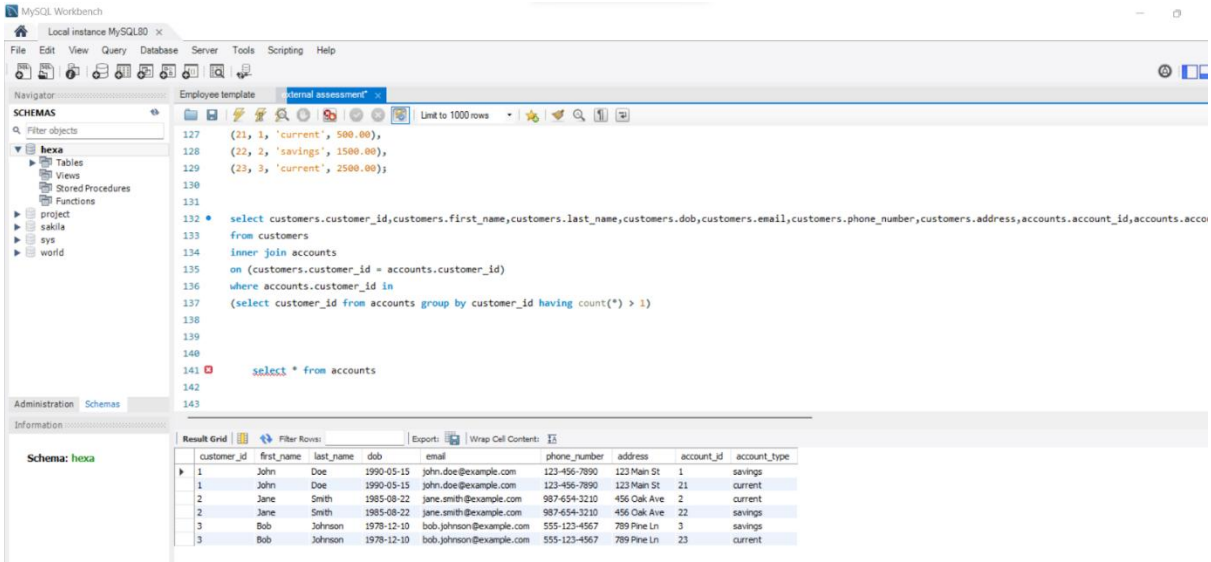


**7.Identify customers who have accounts of multiple types**

**QUERY:**

```
select
customers.customer_id,customers.first_name,customers.last_name,customers.dob,customers.email,customers.phone_number,customers.address,accounts.account_id,accounts.account_type
from customers
inner join accounts
on (customers.customer_id = accounts.customer_id)
where accounts.customer_id in
(select customer_id from accounts group by customer_id having count(*) > 1)
```

**OUTPUT:**

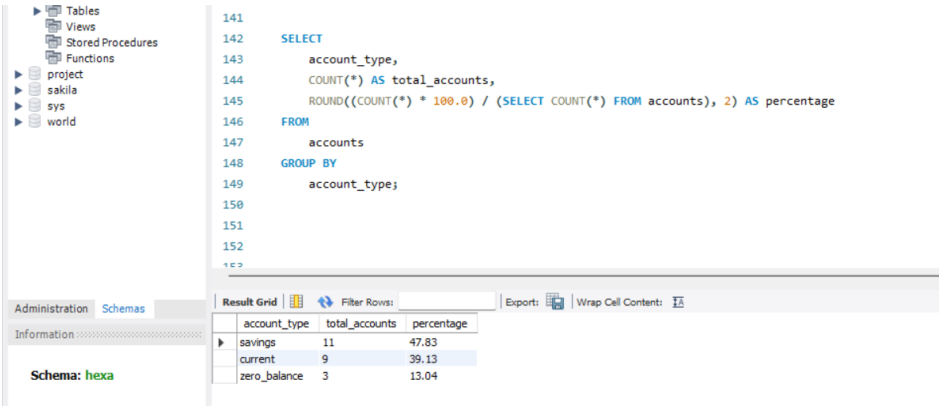



**8.Calculate the percentage of each account type out of the total number of accounts**

**QUERY:**

```
SELECT
   account_type,
   COUNT(*) AS total_accounts,
   ROUND((COUNT(*) * 100.0) / (SELECT COUNT(*) FROM accounts), 2) AS percentage
FROM
   accounts
GROUP BY
   account_type;
```

**OUTPUT:**

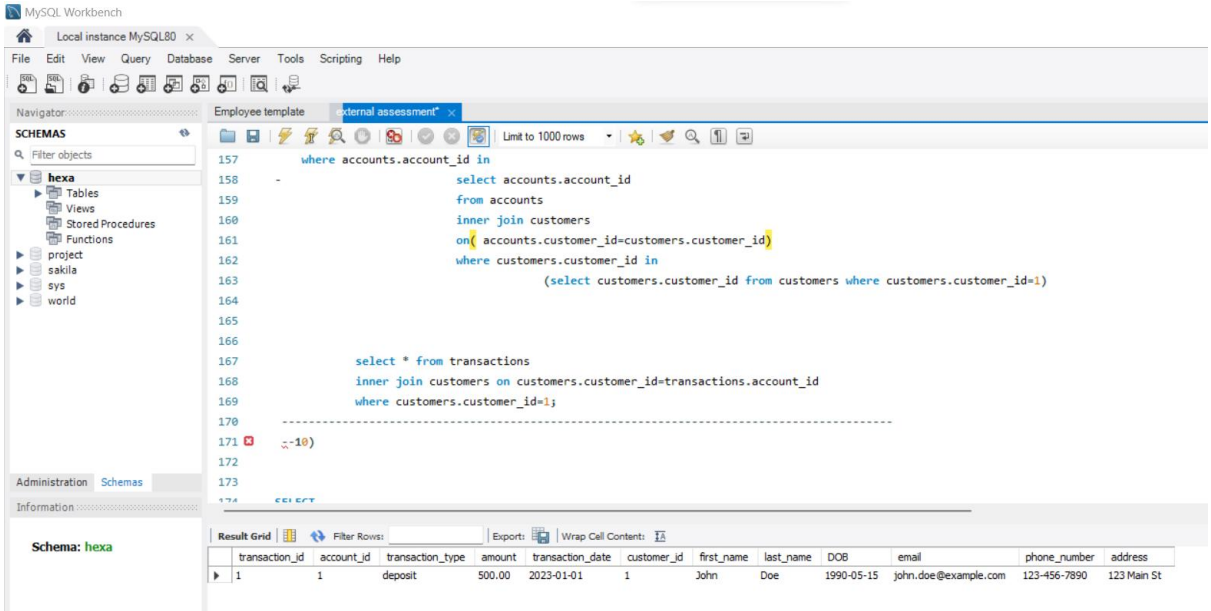


**9.Retrieve all transactions for a customer with a given customer_id.**

**QUERY:**

```
select * from transactions
        inner join customers on customers.customer_id=transactions.account_id
        where customers.customer_id=1;
```

**OUTPUT:**




**10.Calculate the total balance for each account type, including a subquery within the SELECT clause.**

**QUERY:**

```
SELECT
   account_type,
   (SELECT SUM(balance)
    FROM accounts a
    WHERE a.account_type = accounts.account_type
   ) AS total_balance
FROM accounts
GROUP BY Account_type;
```

**OUTPUT:**