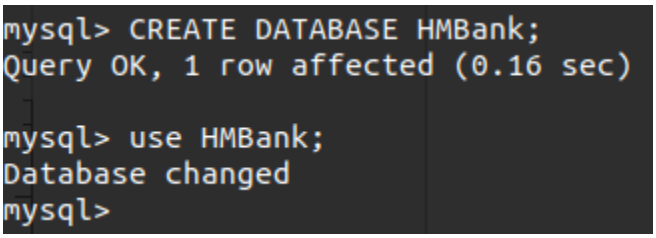
Tasks 1: Database Design:

1. *Create the database named "HMBank"*

```
CREATE DATABASE HMBank;
USE HMBank;
```

OUTPUT:



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

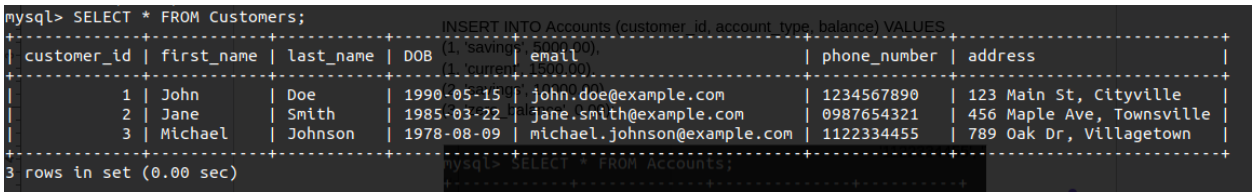
```
CREATE TABLE Customers (
    customer_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    DOB DATE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    phone_number VARCHAR(15) UNIQUE NOT NULL,
    address VARCHAR(255)
);

CREATE TABLE Accounts (
    account_id INT AUTO_INCREMENT PRIMARY KEY,
    customer_id INT,
    account_type VARCHAR(20) NOT NULL,
    balance DECIMAL(15, 2) DEFAULT 0.00
);

CREATE TABLE Transactions (
    transaction_id INT AUTO_INCREMENT PRIMARY KEY,
    account_id INT,
    transaction_type VARCHAR(20) NOT NULL,
   amount DECIMAL(15, 2) NOT NULL,
    transaction_date DATETIME NOT NULL,
);
```

Feeling the dummy data in the tables

```
INSERT INTO Customers (first_name, last_name, DOB, email, phone_number, address)
VALUES
('John', 'Doe', '1990-05-15', 'john.doe@example.com', '1234567890', '123 Main St, Cityville'),
```

('Jane', 'Smith', '1985-03-22', 'jane.smith@example.com', '0987654321', '456 Maple Ave, Townsville'),
('Michael', 'Johnson', '1978-08-09', 'michael.johnson@example.com', '1122334455', '789 Oak Dr, Villagetown');

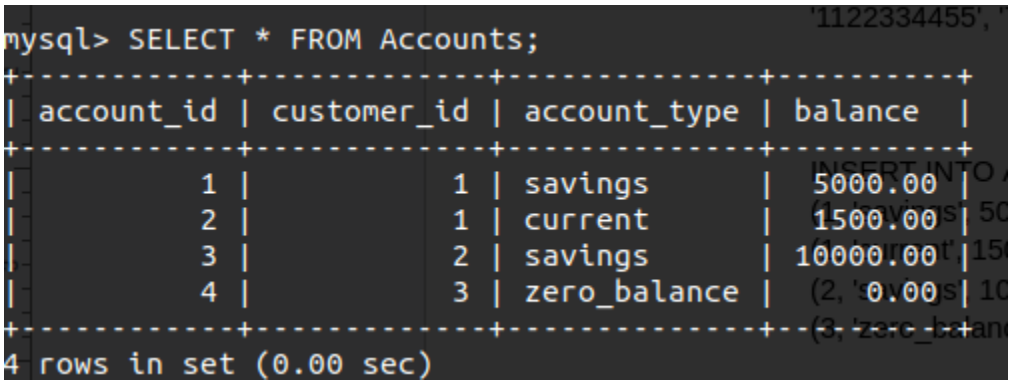
OUTPUT:-



```
mysql> SELECT * FROM Customers;
+-------------+------------+-----------+------------+-------------------------+--------------+-------------------------+
| customer_id | first_name | last_name | DOB        | email                   | phone_number | address                 |
+-------------+------------+-----------+------------+-------------------------+--------------+-------------------------+
|           1 | John       | Doe       | 1990-05-15 | john.doe@example.com    | 1234567890   | 123 Main St, Cityville  |
|           2 | Jane       | Smith     | 1985-03-22 | jane.smith@example.com  | 0987654321   | 456 Maple Ave, Townsville |
|           3 | Michael    | Johnson   | 1978-08-09 | michael.johnson@example.com | 1122334455 | 789 Oak Dr, Villagetown |
+-------------+------------+-----------+------------+-------------------------+--------------+-------------------------+
3 rows in set (0.00 sec)
```

INSERT INTO Accounts (customer_id, account_type, balance) VALUES
(1, 'savings', 5000.00),
(1, 'current', 1500.00),
(2, 'savings', 10000.00),
(3, 'zero_balance', 0.00);

OUTPUT:-



```
mysql> SELECT * FROM Accounts;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|          1 |           1 | savings      |  5000.00 |
|          2 |           1 | current      |  1500.00 |
|          3 |           2 | savings      | 10000.00 |
|          4 |           3 | zero_balance |     0.00 |
+------------+-------------+--------------+----------+
4 rows in set (0.00 sec)
```

INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date) VALUES
(1, 'deposit', 2000.00, '2024-09-01 10:30:00'),
(1, 'withdrawal', 500.00, '2024-09-05 14:00:00'),
(2, 'deposit', 1500.00, '2024-09-07 09:00:00'),
(3, 'withdrawal', 250.00, '2024-09-10 16:00:00'),
(4, 'deposit', 1000.00, '2024-09-15 11:15:00');

OUTPUT:-

```
mysql> SELECT * FROM Transactions;
+----------------+------------+------------------+---------+---------------------+
| transaction_id | account_id | transaction_type | amount  | transaction_date    |
+----------------+------------+------------------+---------+---------------------+
|              1 |          1 | deposit          | 2000.00 | 2024-09-01 10:30:00 |
|              2 |          1 | withdrawal       |  500.00 | 2024-09-05 14:00:00 |
|              3 |          2 | deposit          | 1500.00 | 2024-09-07 09:00:00 |
|              4 |          3 | withdrawal       |  250.00 | 2024-09-10 16:00:00 |
|              5 |          4 | deposit          | 1000.00 | 2024-09-15 11:15:00 |
+----------------+------------+------------------+---------+---------------------+
5 rows in set (0.00 sec)
```

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

Primary Key Constraints:

- `customer_id` in `Customers`
- `account_id` in `Accounts`
- `transaction_id` in `Transactions`

Foreign Key Constraints:

- `customer_id` in `Accounts` referencing `Customers`
- `account_id` in `Transactions` referencing `Accounts`

6. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
• Customers
• Accounts
• Transactions

```
CREATE TABLE Customers (
    customer_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    DOB DATE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    phone_number VARCHAR(15) UNIQUE NOT NULL,
    address VARCHAR(255)
);

CREATE TABLE Accounts (
    account_id INT AUTO_INCREMENT PRIMARY KEY,
    customer_id INT,
    balance DECIMAL(15, 2) DEFAULT 0.00,

    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id) -- Foreign Key
constraint
        ON DELETE CASCADE
        ON UPDATE CASCADE
```

```
);

CREATE TABLE Transactions (
    transaction_id INT AUTO_INCREMENT PRIMARY KEY,
    account_id INT,
    transaction_type VARCHAR(20) NOT NULL,
    amount DECIMAL(15, 2) NOT NULL,
    transaction_date DATETIME NOT NULL,

    FOREIGN KEY (account_id) REFERENCES Accounts(account_id) -- Foreign Key
constraint
        ON DELETE CASCADE
        ON UPDATE CASCADE;
);
```

Tasks 2: Select, Where, Between, AND, LIKE:

1. *Insert at least 10 sample records into each of the following tables.*
   • Customers
   • Accounts
   • Transactions

INSERT INTO Customers (first_name, last_name, DOB, email, phone_number, address) VALUES

('John', 'Doe', '1990-05-15', 'john.doe@example.com', '1234567890', '123 Main St, Cityville'),

('Jane', 'Smith', '1985-03-22', 'jane.smith@example.com', '0987654321', '456 Maple Ave, Townsville'),

('Michael', 'Johnson', '1978-08-09', 'michael.johnson@example.com', '1122334455', '789 Oak Dr, Villagetown'),

('Emily', 'Davis', '1992-11-30', 'emily.davis@example.com', '2233445566', '101 Pine St, Forestville'),

('David', 'Williams', '1980-12-10', 'david.williams@example.com', '3344556677', '202 Birch St, Hilltown'),

('Sophia', 'Garcia', '1995-02-25', 'sophia.garcia@example.com', '4455667788', '303 Cedar St, Lakeview'),

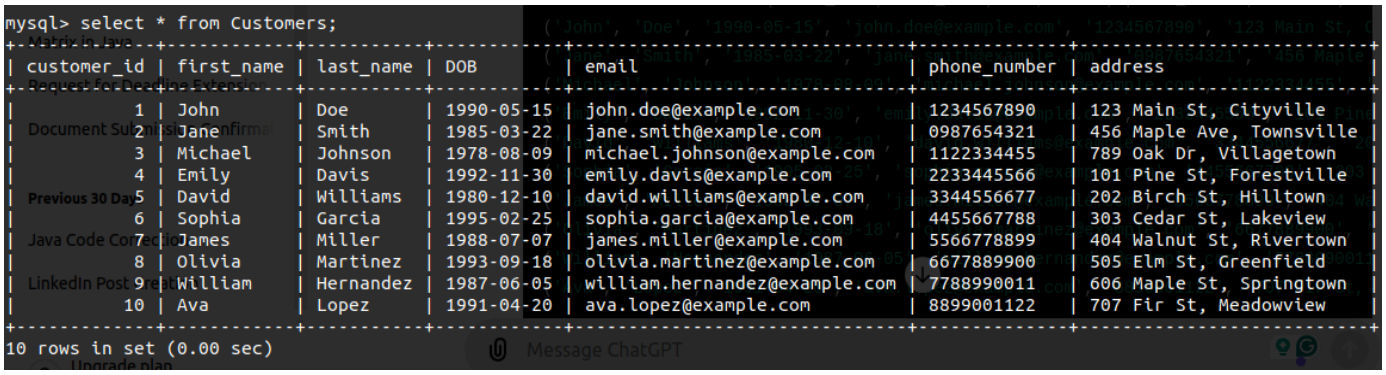
('James', 'Miller', '1988-07-07', 'james.miller@example.com', '5566778899', '404 Walnut St, Rivertown'),

('Olivia', 'Martinez', '1993-09-18', 'olivia.martinez@example.com', '6677889900', '505 Elm St, Greenfield'),

('William', 'Hernandez', '1987-06-05', 'william.hernandez@example.com', '7788990011', '606 Maple St, Springtown'),

('Ava', 'Lopez', '1991-04-20', 'ava.lopez@example.com', '8899001122', '707 Fir St, Meadowview');

OUTPUT:-

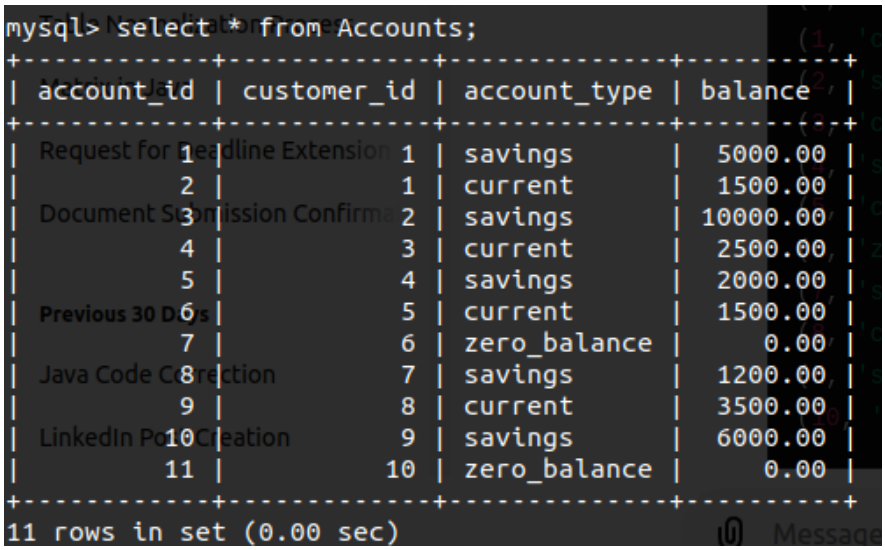


```
mysql> select * from Customers;
+-------------+------------+-----------+------------+-----------------------------+--------------+---------------------------+
| customer_id | first_name | last_name | DOB        | email                       | phone_number | address                   |
+-------------+------------+-----------+------------+-----------------------------+--------------+---------------------------+
|           1 | John       | Doe       | 1990-05-15 | john.doe@example.com        | 1234567890   | 123 Main St, Cityville    |
|           2 | Jane       | Smith     | 1985-03-22 | jane.smith@example.com      | 0987654321   | 456 Maple Ave, Townsville |
|           3 | Michael    | Johnson   | 1978-08-09 | michael.johnson@example.com | 1122334455   | 789 Oak Dr, Villagetown   |
|           4 | Emily      | Davis     | 1992-11-30 | emily.davis@example.com     | 2233445566   | 101 Pine St, Forestville  |
|           5 | David      | Williams  | 1980-12-10 | david.williams@example.com  | 3344556677   | 202 Birch St, Hilltown    |
|           6 | Sophia     | Garcia    | 1995-02-25 | sophia.garcia@example.com   | 4455667788   | 303 Cedar St, Lakeview    |
|           7 | James      | Miller    | 1988-07-07 | james.miller@example.com    | 5566778899   | 404 Walnut St, Rivertown  |
|           8 | Olivia     | Martinez  | 1993-09-18 | olivia.martinez@example.com | 6677889900   | 505 Elm St, Greenfield    |
|           9 | William    | Hernandez | 1987-06-05 | william.hernandez@example.com | 7788990011 | 606 Maple St, Springtown  |
|          10 | Ava        | Lopez     | 1991-04-20 | ava.lopez@example.com       | 8899001122   | 707 Fir St, Meadowview    |
+-------------+------------+-----------+------------+-----------------------------+--------------+---------------------------+
10 rows in set (0.00 sec)
```

INSERT INTO Accounts (customer_id, account_type, balance) VALUES
(1, 'savings', 5000.00),
(1, 'current', 1500.00),
(2, 'savings', 10000.00),
(3, 'current', 2500.00),
(4, 'savings', 2000.00),
(5, 'current', 1500.00),
(6, 'zero_balance', 0.00),
(7, 'savings', 1200.00),
(8, 'current', 3500.00),
(9, 'savings', 6000.00),

(10, 'zero_balance', 0.00);

OUTPUT:-



```
mysql> select * from Accounts;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|          1 |           1 | savings      |  5000.00 |
|          2 |           1 | current      |  1500.00 |
|          3 |           2 | savings      | 10000.00 |
|          4 |           3 | current      |  2500.00 |
|          5 |           4 | savings      |  2000.00 |
|          6 |           5 | current      |  1500.00 |
|          7 |           6 | zero_balance |     0.00 |
|          8 |           7 | savings      |  1200.00 |
|          9 |           8 | current      |  3500.00 |
|         10 |           9 | savings      |  6000.00 |
|         11 |          10 | zero_balance |     0.00 |
+------------+-------------+--------------+----------+
11 rows in set (0.00 sec)
```

INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date) VALUES
(1, 'deposit', 2000.00, '2024-09-01 10:30:00'),
(1, 'withdrawal', 500.00, '2024-09-05 14:00:00'),
(2, 'deposit', 1500.00, '2024-09-07 09:00:00'),
(3, 'withdrawal', 250.00, '2024-09-10 16:00:00'),
(4, 'deposit', 1000.00, '2024-09-15 11:15:00'),
(5, 'deposit', 800.00, '2024-09-20 12:00:00'),
(6, 'withdrawal', 100.00, '2024-09-25 09:45:00'),
(7, 'transfer', 300.00, '2024-09-28 10:00:00'),
(8, 'deposit', 500.00, '2024-09-29 08:30:00'),
(9, 'withdrawal', 1200.00, '2024-09-30 14:15:00');

OUTPUT:-

```
mysql> select * from Transactions;
+----------------+------------+------------------+---------+---------------------+
| transaction_id | account_id | transaction_type | amount  | transaction_date    |
+----------------+------------+------------------+---------+---------------------+
|              1 |          1 | deposit          | 2000.00 | 2024-09-01 10:30:00 |
|              2 |          1 | withdrawal       |  500.00 | 2024-09-05 14:00:00 |
|              3 |          2 | deposit          | 1500.00 | 2024-09-07 09:00:00 |
|              4 |          3 | withdrawal       |  250.00 | 2024-09-10 16:00:00 |
|              5 |          4 | deposit          | 1000.00 | 2024-09-15 11:15:00 |
|              6 |          5 | deposit          |  800.00 | 2024-09-20 12:00:00 |
|              7 |          6 | withdrawal       |  100.00 | 2024-09-25 09:45:00 |
|              8 |          7 | transfer         |  300.00 | 2024-09-28 10:00:00 |
|              9 |          8 | deposit          |  500.00 | 2024-09-29 08:30:00 |
|             10 |          9 | withdrawal       | 1200.00 | 2024-09-30 14:15:00 |
+----------------+------------+------------------+---------+---------------------+
10 rows in set (0.00 sec)
```

Question 2.

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

```
mysql> SELECT
    ->     c.first_name,
    ->     c.last_name,
    ->     a.account_type,
    ->     c.email
    -> FROM
    ->     Customers c
    -> JOIN
    ->     Accounts a ON c.customer_id = a.customer_id;
+------------+-----------+--------------+-----------------------------+
| first_name | last_name | account_type | email                       |
+------------+-----------+--------------+-----------------------------+
| John       | Doe       | savings      | john.doe@example.com        |
| John       | Doe       | current      | john.doe@example.com        |
| Jane       | Smith     | savings      | jane.smith@example.com      |
| Michael    | Johnson   | current      | michael.johnson@example.com |
| Emily      | Davis     | savings      | emily.davis@example.com     |
| David      | Williams  | current      | david.williams@example.com  |
| Sophia     | Garcia    | zero_balance | sophia.garcia@example.com   |
| James      | Miller    | savings      | james.miller@example.com    |
| Olivia     | Martinez  | current      | olivia.martinez@example.com |
| William    | Hernandez | savings      | william.hernandez@example.com |
| Ava        | Lopez     | zero_balance | ava.lopez@example.com       |
+------------+-----------+--------------+-----------------------------+
11 rows in set (0.00 sec)
```

2. **Write a SQL query to list all transaction corresponding customer.**

```
mysql> SELECT
    ->     c.first_name,
    ->     c.last_name,
    ->     c.email,
    ->     a.account_type,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> JOIN
    ->     Accounts a ON t.account_id = a.account_id
    -> JOIN
    ->     Customers c ON a.customer_id = c.customer_id;
+------------+-----------+---------------------------+--------------+------------------+---------+---------------------+
| first_name | last_name | email                     | account_type | transaction_type | amount  | transaction_date    |
+------------+-----------+---------------------------+--------------+------------------+---------+---------------------+
| John       | Doe       | john.doe@example.com      | savings      | deposit          | 2000.00 | 2024-09-01 10:30:00 |
| John       | Doe       | john.doe@example.com      | savings      | withdrawal       |  500.00 | 2024-09-05 14:00:00 |
| John       | Doe       | john.doe@example.com      | current      | deposit          | 1500.00 | 2024-09-07 09:00:00 |
| Jane       | Smith     | jane.smith@example.com    | savings      | withdrawal       |  250.00 | 2024-09-10 16:00:00 |
| Michael    | Johnson   | michael.johnson@example.com | current    | deposit          | 1000.00 | 2024-09-15 11:15:00 |
| Emily      | Davis     | emily.davis@example.com   | savings      | deposit          |  800.00 | 2024-09-20 12:00:00 |
| David      | Williams  | david.williams@example.com | current     | withdrawal       |  100.00 | 2024-09-25 09:45:00 |
| Sophia     | Garcia    | sophia.garcia@example.com | zero_balance | transfer         |  300.00 | 2024-09-28 10:00:00 |
| James      | Miller    | james.miller@example.com  | savings      | deposit          |  500.00 | 2024-09-29 08:30:00 |
| Olivia     | Martinez  | olivia.martinez@example.com | current    | withdrawal       | 1200.00 | 2024-09-30 14:15:00 |
+------------+-----------+---------------------------+--------------+------------------+---------+---------------------+
10 rows in set (0.01 sec)
```

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

```
mysql> UPDATE Accounts
    -> SET balance = balance + 500.00
    -> WHERE account_id = 1;
Query OK, 1 row affected (0.11 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

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

```
mysql> SELECT
    ->     CONCAT(first_name, ' ', last_name) AS full_name,
    ->     email,
    ->     phone_number
    -> FROM
    ->     Customers;
+-------------------+---------------------------------+--------------+
| full_name         | email                           | phone_number |
+-------------------+---------------------------------+--------------+
| John Doe          | john.doe@example.com            | 1234567890   |
| Jane Smith        | jane.smith@example.com          | 0987654321   |
| Michael Johnson   | michael.johnson@example.com     | 1122334455   |
| Emily Davis       | emily.davis@example.com         | 2233445566   |
| David Williams    | david.williams@example.com      | 3344556677   |
| Sophia Garcia     | sophia.garcia@example.com       | 4455667788   |
| James Miller      | james.miller@example.com        | 5566778899   |
| Olivia Martinez   | olivia.martinez@example.com     | 6677889900   |
| William Hernandez | william.hernandez@example.com   | 7788990011   |
| Ava Lopez         | ava.lopez@example.com           | 8899001122   |
+-------------------+---------------------------------+--------------+
10 rows in set (0.00 sec)
```

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

```
mysql> DELETE FROM Accounts
    -> WHERE balance = 0 AND account_type = 'savings';
Query OK, 0 rows affected (0.00 sec)
```

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

```
mysql> SELECT
    ->     first_name,
    ->     last_name,
    ->     email,
    ->     phone_number
    -> FROM
    ->     Customers
    -> WHERE
    ->     address LIKE '%Maple%';
+------------+-----------+-------------------------------+--------------+
| first_name | last_name | email                         | phone_number |
+------------+-----------+-------------------------------+--------------+
| Jane       | Smith     | jane.smith@example.com        | 0987654321   |
| William    | Hernandez | william.hernandez@example.com | 7788990011   |
+------------+-----------+-------------------------------+--------------+
2 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     account_id,
    ->     balance
    -> FROM
    ->     Accounts
    -> WHERE
    ->     account_id = 1;
+------------+---------+
| account_id | balance |
+------------+---------+
|          1 | 5500.00 |
+------------+---------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     account_id,
    ->     customer_id,
    ->     account_type,
    ->     balance
    -> FROM
    ->     Accounts
    -> WHERE
    ->     account_type = 'current' AND balance > 1000;
+------------+-------------+--------------+---------+
| account_id | customer_id | account_type | balance |
+------------+-------------+--------------+---------+
|          2 |           1 | current      | 1500.00 |
|          4 |           3 | current      | 2500.00 |
|          6 |           5 | current      | 1500.00 |
|          9 |           8 | current      | 3500.00 |
+------------+-------------+--------------+---------+
4 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     transaction_id,
    ->     account_id,
    ->     transaction_type,
    ->     amount,
    ->     transaction_date
    -> FROM
    ->     Transactions
    -> WHERE
    ->     account_id = 1;
+----------------+------------+------------------+--------+---------------------+
| transaction_id | account_id | transaction_type | amount | transaction_date    |
+----------------+------------+------------------+--------+---------------------+
|              1 |          1 | deposit          | 2000.00 | 2024-09-01 10:30:00 |
|              2 |          1 | withdrawal       |  500.00 | 2024-09-05 14:00:00 |
+----------------+------------+------------------+--------+---------------------+
2 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     account_id,
    ->     customer_id,
    ->     balance,
    ->     (balance * 0.03) AS interest_accrued
    -> FROM
    ->     Accounts
    -> WHERE
    ->     account_type = 'savings';
+------------+-------------+----------+------------------+
| account_id | customer_id | balance  | interest_accrued |
+------------+-------------+----------+------------------+
|          1 |           1 |  5500.00 |         165.0000 |
|          3 |           2 | 10000.00 |         300.0000 |
|          5 |           4 |  2000.00 |          60.0000 |
|          8 |           7 |  1200.00 |          36.0000 |
|         10 |           9 |  6000.00 |         180.0000 |
+------------+-------------+----------+------------------+
5 rows in set (0.00 sec)
```

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

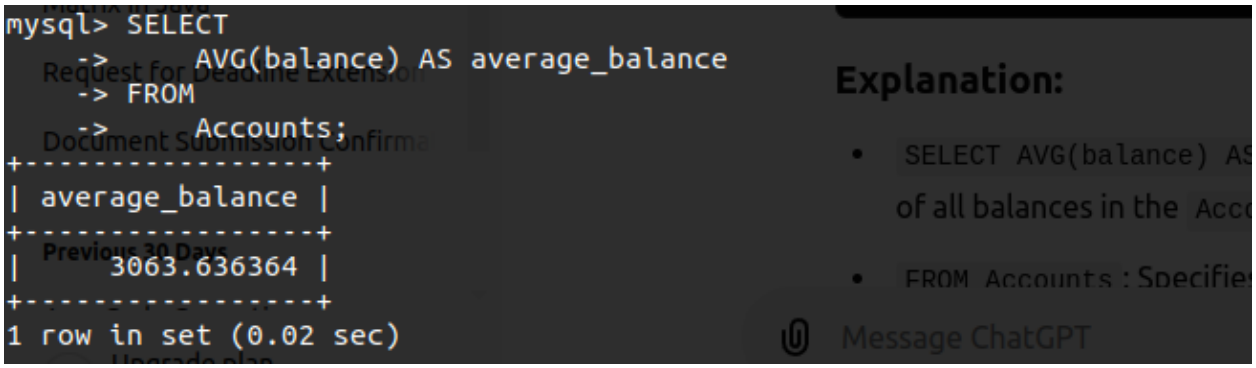
```
mysql> SELECT
    ->     account_id,
    ->     customer_id,
    ->     account_type,
    ->     balance
    -> FROM
    ->     Accounts
    -> WHERE
    ->     balance < 500;
+------------+-------------+--------------+---------+
| account_id | customer_id | account_type | balance |
+------------+-------------+--------------+---------+
|          7 |           6 | zero_balance |    0.00 |
|         11 |          10 | zero_balance |    0.00 |
+------------+-------------+--------------+---------+
2 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     first_name,
    ->     last_name,
    ->     email,
    ->     phone_number
    -> FROM
    ->     Customers
    -> WHERE
    ->     address NOT LIKE '%Maple%';
+------------+-----------+-----------------------------+--------------+
| first_name | last_name | email                       | phone_number |
+------------+-----------+-----------------------------+--------------+
| John       | Doe       | john.doe@example.com        | 1234567890   |
| Michael    | Johnson   | michael.johnson@example.com | 1122334455   |
| Emily      | Davis     | emily.davis@example.com     | 2233445566   |
| David      | Williams  | david.williams@example.com  | 3344556677   |
| Sophia     | Garcia    | sophia.garcia@example.com   | 4455667788   |
| James      | Miller    | james.miller@example.com    | 5566778899   |
| Olivia     | Martinez  | olivia.martinez@example.com | 6677889900   |
| Ava        | Lopez     | ava.lopez@example.com       | 8899001122   |
+------------+-----------+-----------------------------+--------------+
8 rows in set (0.01 sec)
```

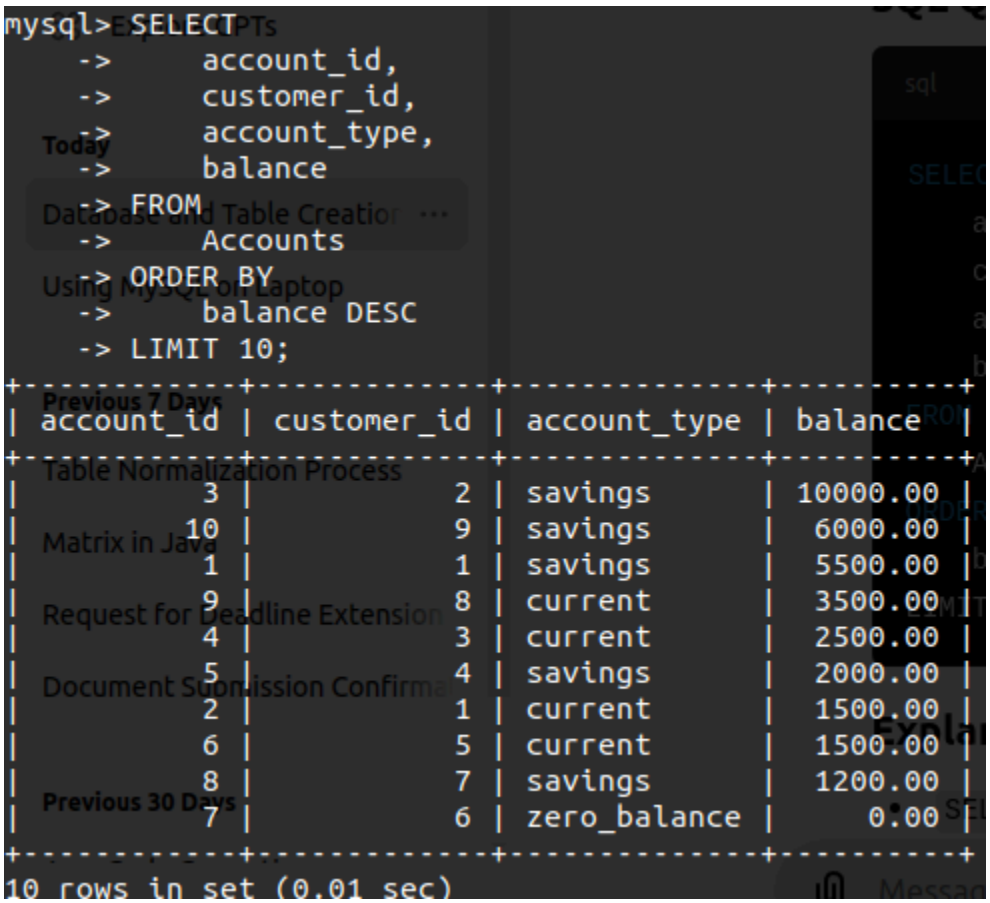
Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:

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



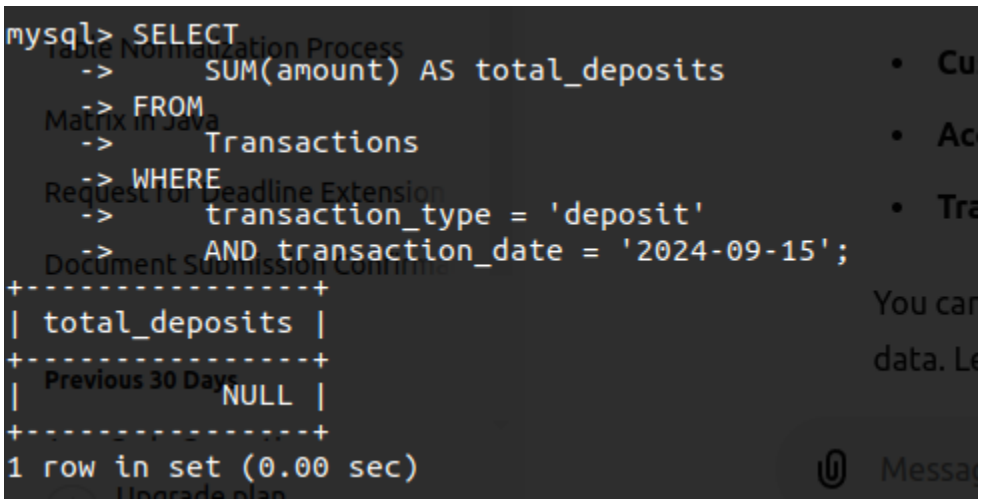
```
mysql> SELECT
    ->     AVG(balance) AS average_balance
    -> FROM
    ->     Accounts;
+-----------------+
| average_balance |
+-----------------+
|     3063.636364 |
+-----------------+
1 row in set (0.02 sec)
```

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



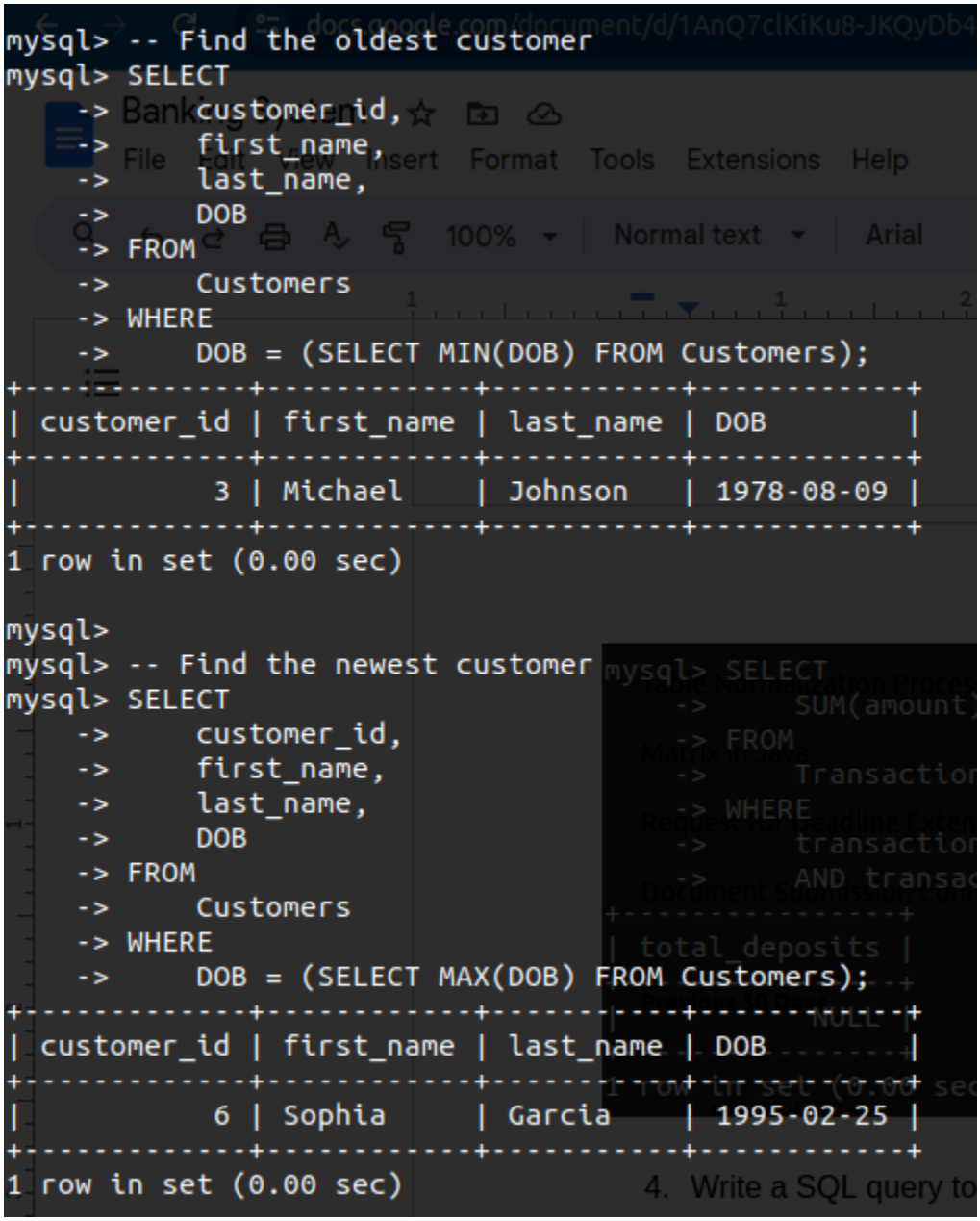
```
mysql> SELECT
    ->     account_id,
    ->     customer_id,
    ->     account_type,
    ->     balance
    -> FROM
    ->     Accounts
    -> ORDER BY
    ->     balance DESC
    -> LIMIT 10;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|          3 |           2 | savings      | 10000.00 |
|         10 |           9 | savings      |  6000.00 |
|          1 |           1 | savings      |  5500.00 |
|          9 |           8 | current      |  3500.00 |
|          4 |           3 | current      |  2500.00 |
|          5 |           4 | savings      |  2000.00 |
|          2 |           1 | current      |  1500.00 |
|          6 |           5 | current      |  1500.00 |
|          8 |           7 | savings      |  1200.00 |
|          7 |           6 | zero_balance |     0.00 |
+------------+-------------+--------------+----------+
10 rows in set (0.01 sec)
```

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



```
mysql> SELECT
    ->     SUM(amount) AS total_deposits
    -> FROM
    ->     Transactions
    -> WHERE
    ->     transaction_type = 'deposit'
    ->     AND transaction_date = '2024-09-15';
+----------------+
| total_deposits |
+----------------+
|           NULL |
+----------------+
1 row in set (0.00 sec)
```

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



```
mysql> -- Find the oldest customer
mysql> SELECT
    ->     customer_id,
    ->     first_name,
    ->     last_name,
    ->     DOB
    -> FROM
    ->     Customers
    -> WHERE
    ->     DOB = (SELECT MIN(DOB) FROM Customers);
+-------------+------------+-----------+------------+
| customer_id | first_name | last_name | DOB        |
+-------------+------------+-----------+------------+
|           3 | Michael    | Johnson   | 1978-08-09 |
+-------------+------------+-----------+------------+
1 row in set (0.00 sec)

mysql>
mysql> -- Find the newest customer
mysql> SELECT
    ->     customer_id,
    ->     first_name,
    ->     last_name,
    ->     DOB
    -> FROM
    ->     Customers
    -> WHERE
    ->     DOB = (SELECT MAX(DOB) FROM Customers);
+-------------+------------+-----------+------------+
| customer_id | first_name | last_name | DOB        |
+-------------+------------+-----------+------------+
|           6 | Sophia     | Garcia    | 1995-02-25 |
+-------------+------------+-----------+------------+
1 row in set (0.00 sec)
```

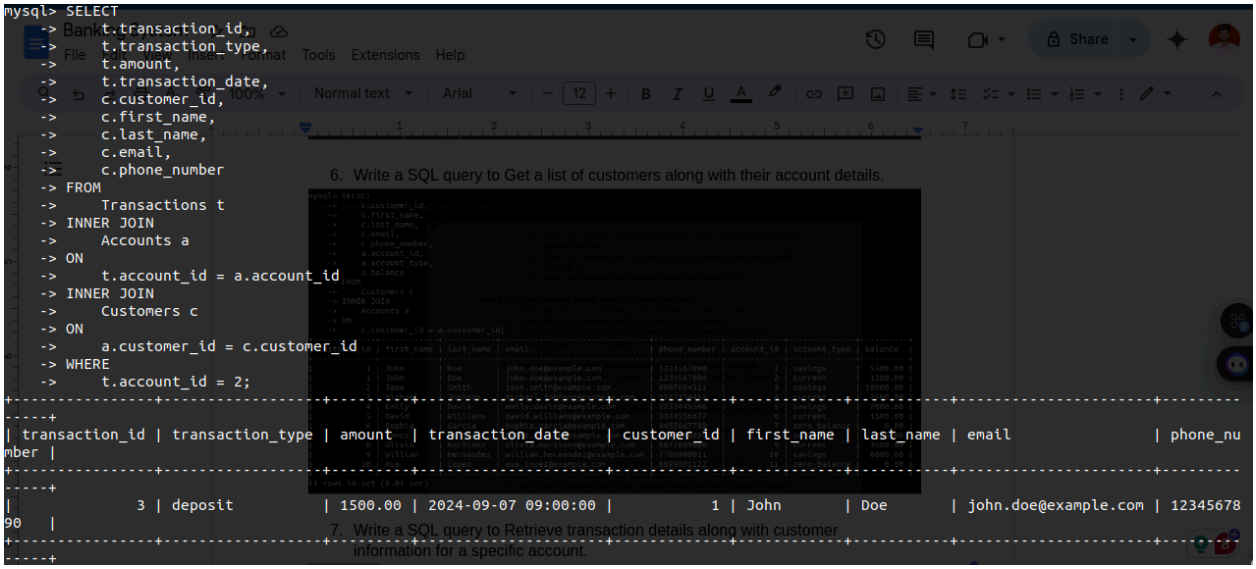
5. Write a SQL query to Retrieve transaction details along with the account type.

```
mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     a.account_type,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     t.account_id = a.account_id;
+----------------+------------+--------------+------------------+---------+---------------------+
| transaction_id | account_id | account_type | transaction_type | amount  | transaction_date    |
+----------------+------------+--------------+------------------+---------+---------------------+
|              1 |          1 | savings      | deposit          | 2000.00 | 2024-09-01 10:30:00 |
|              2 |          1 | savings      | withdrawal       |  500.00 | 2024-09-05 14:00:00 |
|              3 |          2 | current      | deposit          | 1500.00 | 2024-09-07 09:00:00 |
|              4 |          3 | savings      | withdrawal       |  250.00 | 2024-09-10 16:00:00 |
|              5 |          4 | current      | deposit          | 1000.00 | 2024-09-15 11:15:00 |
|              6 |          5 | savings      | deposit          |  800.00 | 2024-09-20 12:00:00 |
|              7 |          6 | current      | withdrawal       |  100.00 | 2024-09-25 09:45:00 |
|              8 |          7 | zero_balance | transfer         |  300.00 | 2024-09-28 10:00:00 |
|              9 |          8 | savings      | deposit          |  500.00 | 2024-09-29 08:30:00 |
|             10 |          9 | current      | withdrawal       | 1200.00 | 2024-09-30 14:15:00 |
+----------------+------------+--------------+------------------+---------+---------------------+
10 rows in set (0.00 sec)
```

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

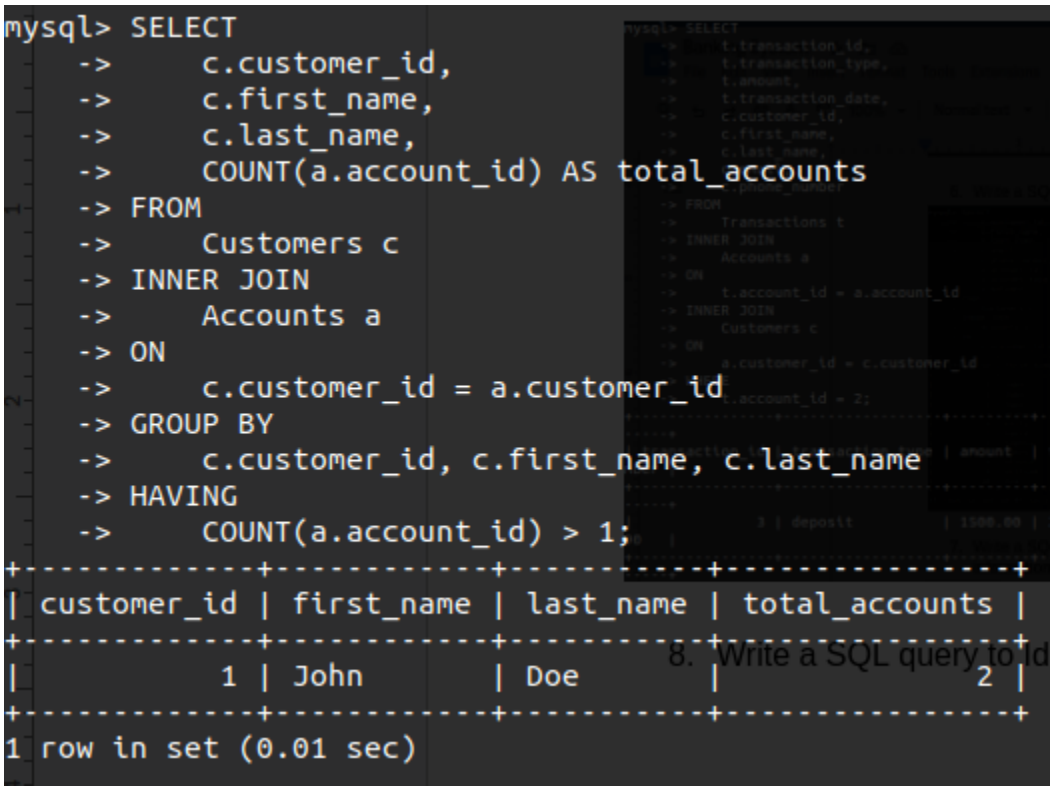
```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     c.email,
    ->     c.phone_number,
    ->     a.account_id,
    ->     a.account_type,
    ->     a.balance
    -> FROM
    ->     Customers c
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     c.customer_id = a.customer_id;
+-------------+------------+-----------+-------------------------+--------------+------------+--------------+----------+
| customer_id | first_name | last_name | email                   | phone_number | account_id | account_type | balance  |
+-------------+------------+-----------+-------------------------+--------------+------------+--------------+----------+
|           1 | John       | Doe       | john.doe@example.com    | 1234567890   |          1 | savings      |  5500.00 |
|           1 | John       | Doe       | john.doe@example.com    | 1234567890   |          2 | current      |  1500.00 |
|           2 | Jane       | Smith     | jane.smith@example.com  | 0987654321   |          3 | savings      | 10000.00 |
|           3 | Michael    | Johnson   | michael.johnson@example.com | 1122334455 |          4 | current      |  2500.00 |
|           4 | Emily      | Davis     | emily.davis@example.com | 2233445566   |          5 | savings      |  2000.00 |
|           5 | David      | Williams  | david.williams@example.com | 3344556677 |          6 | current      |  1500.00 |
|           6 | Sophia     | Garcia    | sophia.garcia@example.com | 4455667788 |          7 | zero_balance |     0.00 |
|           7 | James      | Miller    | james.miller@example.com | 5566778899   |          8 | savings      |  1200.00 |
|           8 | Olivia     | Martinez  | olivia.martinez@example.com | 6677889900 |          9 | current      |  3500.00 |
|           9 | William    | Hernandez | william.hernandez@example.com | 7788990011 |         10 | savings      |  6000.00 |
|          10 | Ava        | Lopez     | ava.lopez@example.com   | 8899001122   |         11 | zero_balance |     0.00 |
+-------------+------------+-----------+-------------------------+--------------+------------+--------------+----------+
11 rows in set (0.01 sec)
```

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



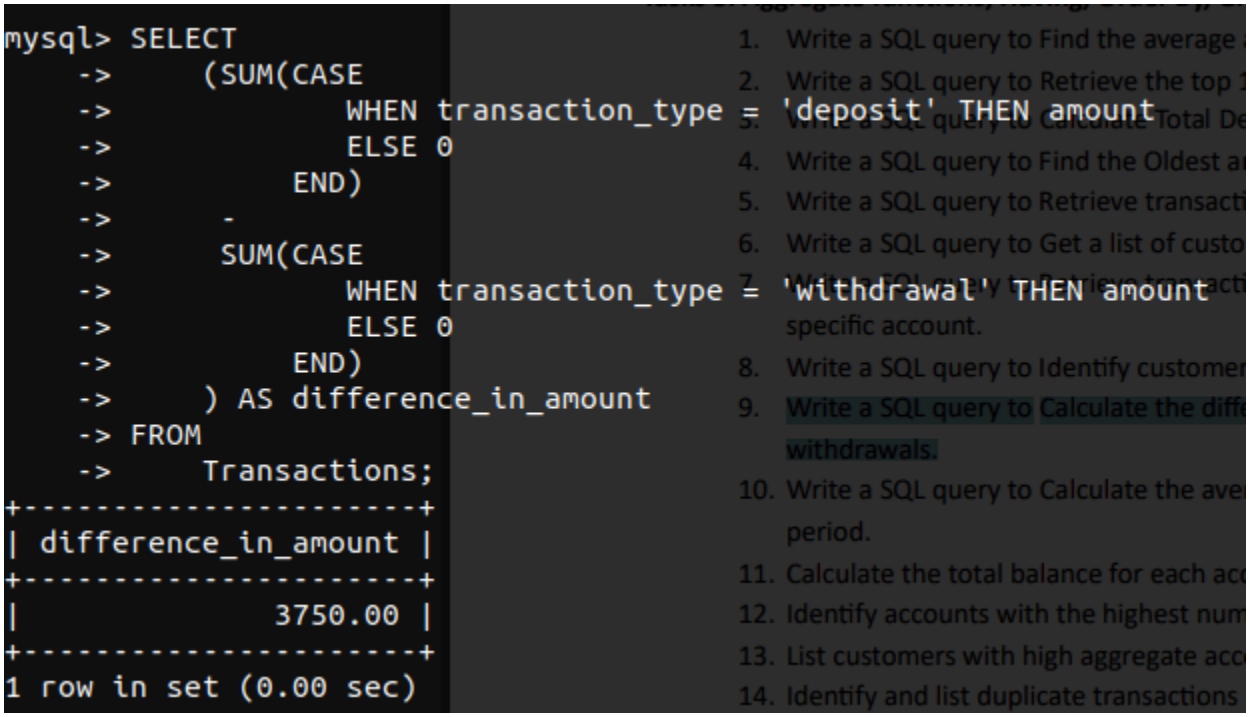
```
mysql> SELECT
    ->     t.transaction_id,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date,
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     c.email,
    ->     c.phone_number
    -> FROM
    ->     Transactions t
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     t.account_id = a.account_id
    -> INNER JOIN
    ->     Customers c
    -> ON
    ->     a.customer_id = c.customer_id
    -> WHERE
    ->     t.account_id = 2;
+----------------+------------------+---------+---------------------+-------------+------------+-----------+-----------------+-----------+
| transaction_id | transaction_type | amount  | transaction_date    | customer_id | first_name | last_name | email           | phone_number |
+----------------+------------------+---------+---------------------+-------------+------------+-----------+-----------------+-----------+
|              3 | deposit          | 1500.00 | 2024-09-07 09:00:00 |           1 | John       | Doe       | john.doe@example.com | 1234567890 |
+----------------+------------------+---------+---------------------+-------------+------------+-----------+-----------------+-----------+
```

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



```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     COUNT(a.account_id) AS total_accounts
    -> FROM
    ->     Customers c
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     c.customer_id = a.customer_id
    -> GROUP BY
    ->     c.customer_id, c.first_name, c.last_name
    -> HAVING
    ->     COUNT(a.account_id) > 1;
+-------------+------------+-----------+-----------------+
| customer_id | first_name | last_name | total_accounts |
+-------------+------------+-----------+-----------------+
|           1 | John       | Doe       |               2 |
+-------------+------------+-----------+-----------------+
1 row in set (0.01 sec)
```

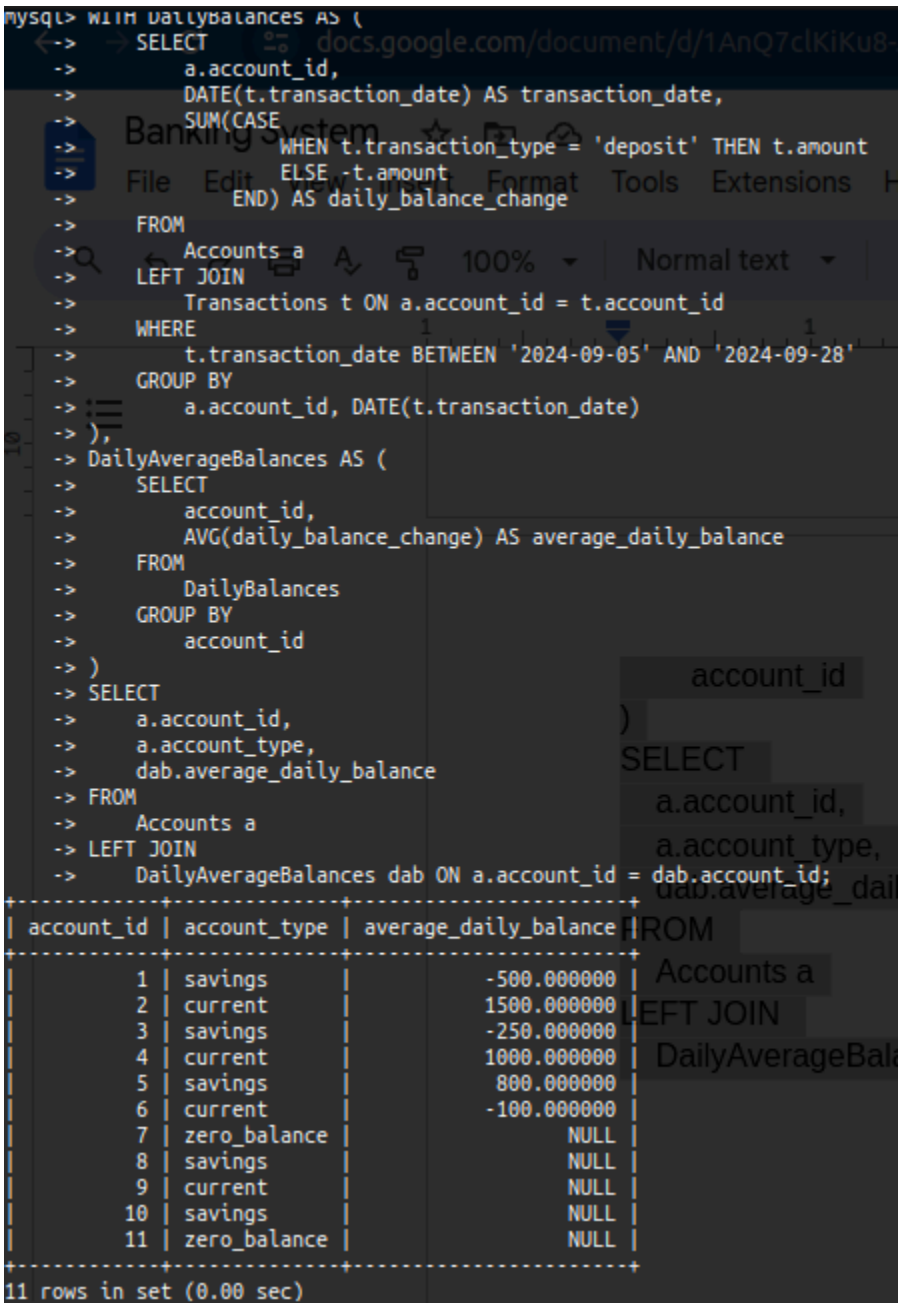
9. Write a SQL query to Calculate the difference in transaction amounts between deposits and withdrawals.



```
mysql> SELECT
    ->     (SUM(CASE
    ->             WHEN transaction_type = 'deposit' THEN amount
    ->             ELSE 0
    ->         END)
    ->      -
    ->      SUM(CASE
    ->             WHEN transaction_type = 'withdrawal' THEN amount
    ->             ELSE 0
    ->         END)
    ->     ) AS difference_in_amount
    -> FROM
    ->     Transactions;
+----------------------+
| difference_in_amount |
+----------------------+
|              3750.00 |
+----------------------+
1 row in set (0.00 sec)
```

1. Write a SQL query to Find the average
2. Write a SQL query to Retrieve the top
3. Write a SQL query to Calculate Total De
4. Write a SQL query to Find the Oldest a
5. Write a SQL query to Retrieve transacti
6. Write a SQL query to Get a list of custo
7. Write a SQL query to retrieve transacti specific account.
8. Write a SQL query to Identify customer
9. Write a SQL query to Calculate the diffe withdrawals.
10. Write a SQL query to Calculate the ave period.
11. Calculate the total balance for each acc
12. Identify accounts with the highest num
13. List customers with high aggregate acc
14. Identify and list duplicate transactions

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



```
mysql> WITH DailyBalances AS (
    ->     SELECT
    ->         a.account_id,
    ->         DATE(t.transaction_date) AS transaction_date,
    ->         SUM(CASE
    ->                 WHEN t.transaction_type = 'deposit' THEN t.amount
    ->                 ELSE -t.amount
    ->             END) AS daily_balance_change
    ->     FROM
    ->         Accounts a
    ->     LEFT JOIN
    ->         Transactions t ON a.account_id = t.account_id
    ->     WHERE
    ->         t.transaction_date BETWEEN '2024-09-05' AND '2024-09-28'
    ->     GROUP BY
    ->         a.account_id, DATE(t.transaction_date)
    -> ),
    -> DailyAverageBalances AS (
    ->     SELECT
    ->         account_id,
    ->         AVG(daily_balance_change) AS average_daily_balance
    ->     FROM
    ->         DailyBalances
    ->     GROUP BY
    ->         account_id
    -> )
    -> SELECT
    ->     a.account_id,
    ->     a.account_type,
    ->     dab.average_daily_balance
    -> FROM
    ->     Accounts a
    -> LEFT JOIN
    ->     DailyAverageBalances dab ON a.account_id = dab.account_id;
+------------+--------------+-----------------------+
| account_id | account_type | average_daily_balance |
+------------+--------------+-----------------------+
|          1 | savings      |           -500.000000 |
|          2 | current      |           1500.000000 |
|          3 | savings      |           -250.000000 |
|          4 | current      |           1000.000000 |
|          5 | savings      |            800.000000 |
|          6 | current      |           -100.000000 |
|          7 | zero_balance |                  NULL |
|          8 | savings      |                  NULL |
|          9 | current      |                  NULL |
|         10 | savings      |                  NULL |
|         11 | zero_balance |                  NULL |
+------------+--------------+-----------------------+
11 rows in set (0.00 sec)
```

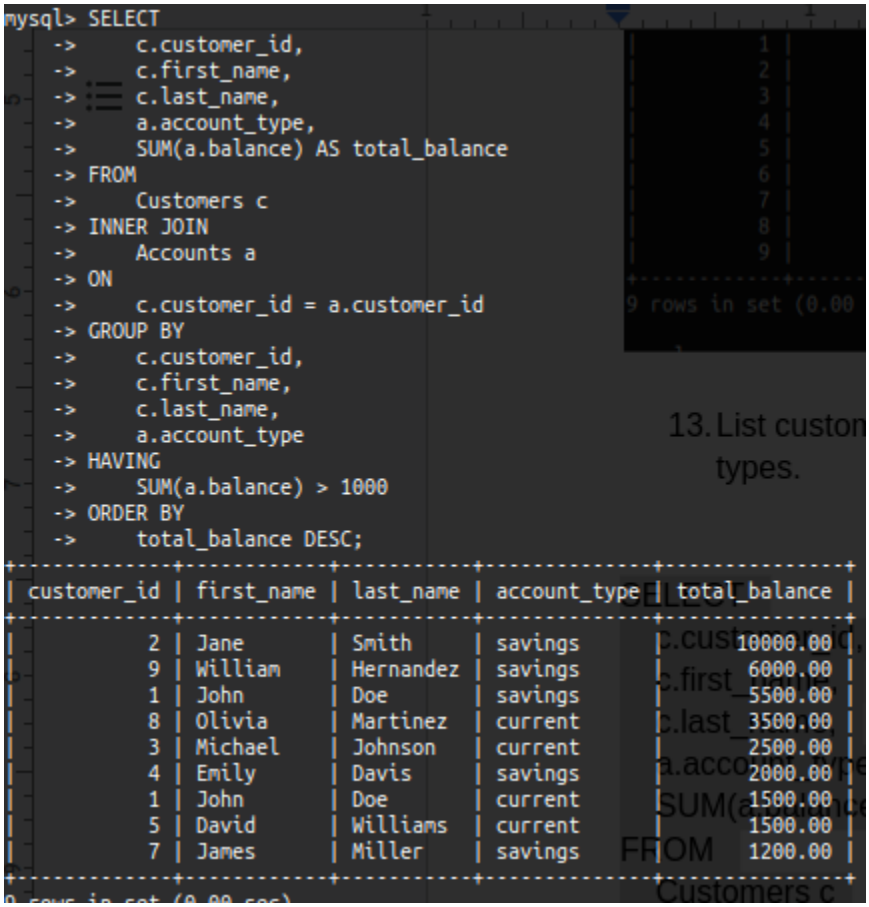
11. Calculate the total balance for each account type.

```
mysql> SELECT
    ->     account_type,
    ->     SUM(balance) AS total_balance
    -> FROM
    ->     Accounts
    -> GROUP BY
    ->     account_type;
+--------------+---------------+
| account_type | total_balance |
+--------------+---------------+
| savings      |      24700.00 |
| current      |       9000.00 |
| zero_balance |          0.00 |
+--------------+---------------+
3 rows in set (0.00 sec)
```

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

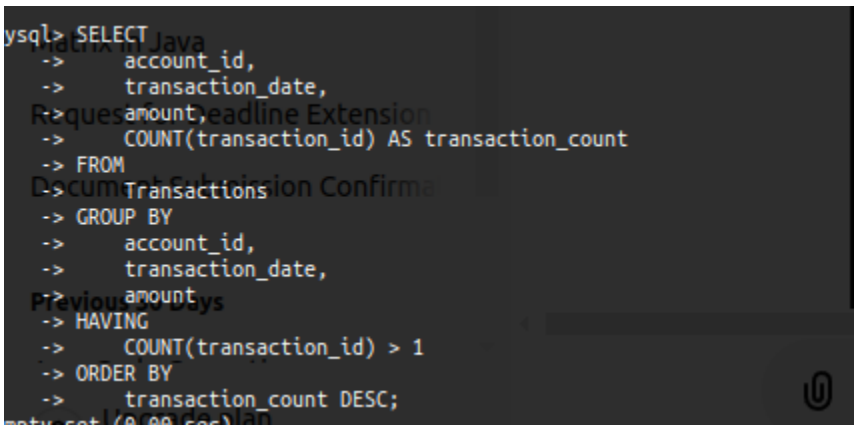
```
mysql> SELECT
    ->     account_id,
    ->     COUNT(transaction_id) AS transaction_count
    -> FROM
    ->     Transactions
    -> GROUP BY
    ->     account_id
    -> ORDER BY
    ->     transaction_count DESC;
+------------+-------------------+
| account_id | transaction_count |
+------------+-------------------+
|          1 |                 2 |
|          2 |                 1 |
|          3 |                 1 |
|          4 |                 1 |
|          5 |                 1 |
|          6 |                 1 |
|          7 |                 1 |
|          8 |                 1 |
|          9 |                 1 |
+------------+-------------------+
9 rows in set (0.00 sec)
```

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



```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     a.account_type,
    ->     SUM(a.balance) AS total_balance
    -> FROM
    ->     Customers c
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     c.customer_id = a.customer_id
    -> GROUP BY
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     a.account_type
    -> HAVING
    ->     SUM(a.balance) > 1000
    -> ORDER BY
    ->     total_balance DESC;
+-------------+------------+-----------+--------------+---------------+
| customer_id | first_name | last_name | account_type | total_balance |
+-------------+------------+-----------+--------------+---------------+
|           2 | Jane       | Smith     | savings      |      10000.00 |
|           9 | William    | Hernandez | savings      |       6000.00 |
|           1 | John       | Doe       | savings      |       5500.00 |
|           8 | Olivia     | Martinez  | current      |       3500.00 |
|           3 | Michael    | Johnson   | current      |       2500.00 |
|           4 | Emily      | Davis     | savings      |       2000.00 |
|           1 | John       | Doe       | current      |       1500.00 |
|           5 | David      | Williams  | current      |       1500.00 |
|           7 | James      | Miller    | savings      |       1200.00 |
+-------------+------------+-----------+--------------+---------------+
```

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



```
mysql> SELECT
    ->     account_id,
    ->     transaction_date,
    ->     amount,
    ->     COUNT(transaction_id) AS transaction_count
    -> FROM
    ->     Transactions
    -> GROUP BY
    ->     account_id,
    ->     transaction_date,
    ->     amount
    -> HAVING
    ->     COUNT(transaction_id) > 1
    -> ORDER BY
    ->     transaction_count DESC;
```

Tasks 4: Subquery and its type:

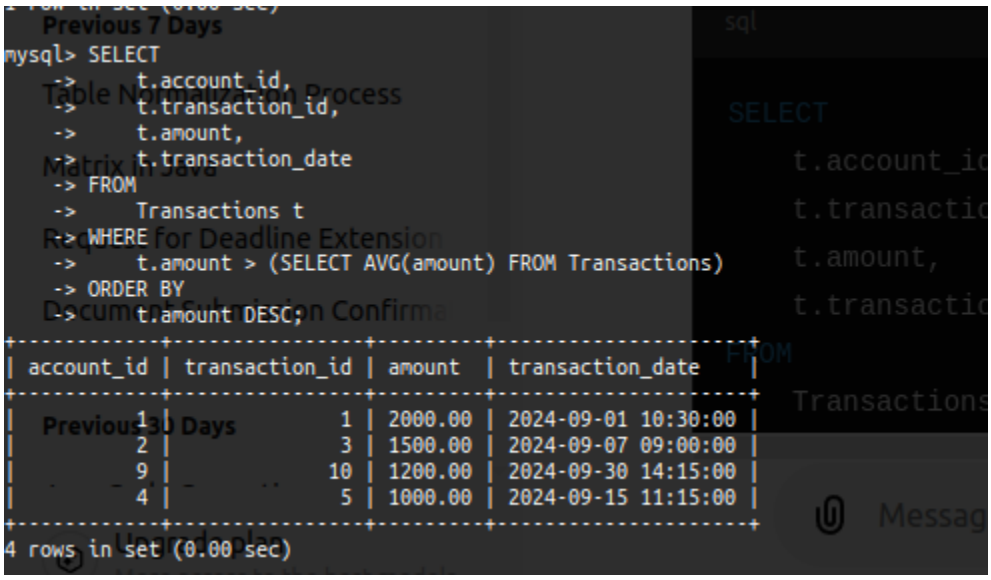
1. Retrieve the customer(s) with the highest account balance

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     a.account_id,
    ->     a.account_type,
    ->     a.balance
    -> FROM
    ->     Customers c
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     c.customer_id = a.customer_id
    -> WHERE
    ->     a.balance = (SELECT MAX(balance) FROM Accounts);
+-------------+------------+-----------+------------+--------------+----------+
| customer_id | first_name | last_name | account_id | account_type | balance  |
+-------------+------------+-----------+------------+--------------+----------+
|           2 | Jane       | Smith     |          3 | savings      | 10000.00 |
+-------------+------------+-----------+------------+--------------+----------+
1 row in set (0.00 sec)
```

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

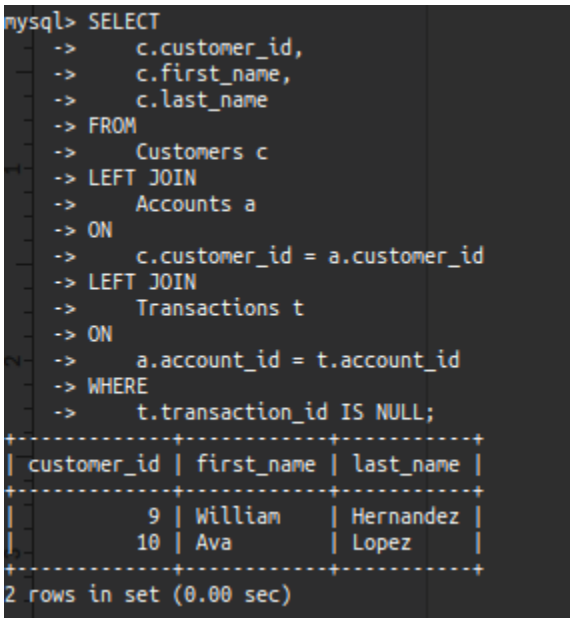
```
mysql> SELECT
    ->     AVG(a.balance) AS average_balance
    -> FROM
    ->     Accounts a
    -> WHERE
    ->     a.customer_id IN (
    ->         SELECT
    ->             customer_id
    ->         FROM
    ->             Accounts
    ->         GROUP BY
    ->             customer_id
    ->         HAVING
    ->             COUNT(account_id) > 1
    ->     );
+-----------------+
| average_balance |
+-----------------+
|    3500.000000 |
+-----------------+
1 row in set (0.00 sec)
```

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



```
mysql> SELECT
    ->     t.account_id,
    ->     t.transaction_id,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> WHERE
    ->     t.amount > (SELECT AVG(amount) FROM Transactions)
    -> ORDER BY
    ->     t.amount DESC;
+------------+----------------+---------+---------------------+
| account_id | transaction_id | amount  | transaction_date    |
+------------+----------------+---------+---------------------+
|          1 |              1 | 2000.00 | 2024-09-01 10:30:00 |
|          2 |              3 | 1500.00 | 2024-09-07 09:00:00 |
|          9 |             10 | 1200.00 | 2024-09-30 14:15:00 |
|          4 |              5 | 1000.00 | 2024-09-15 11:15:00 |
+------------+----------------+---------+---------------------+
4 rows in set (0.00 sec)
```

4. Identify customers who have no recorded transactions.



```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name
    -> 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 NULL;
+-------------+------------+-----------+
| customer_id | first_name | last_name |
+-------------+------------+-----------+
|           9 | William    | Hernandez |
|          10 | Ava        | Lopez     |
+-------------+------------+-----------+
2 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     SUM(a.balance) AS total_balance
    -> FROM
    ->     Accounts a
    -> LEFT JOIN
    ->     Transactions t
    -> ON
    ->     a.account_id = t.account_id
    -> WHERE
    ->     t.transaction_id IS NULL;
+---------------+
| total_balance |
+---------------+
|       6000.00 |
+---------------+
1 row in set (0.00 sec)
```

6. Retrieve transactions for accounts with the lowest balance.

```
mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> WHERE
    ->     t.account_id IN (
    ->         SELECT
    ->             account_id
    ->         FROM
    ->             Accounts
    ->         WHERE
    ->             balance = (SELECT MIN(balance) FROM Accounts)
    ->     )
    -> ORDER BY
    ->     t.transaction_date;
+----------------+------------+------------------+--------+---------------------+
| transaction_id | account_id | transaction_type | amount | transaction_date    |
+----------------+------------+------------------+--------+---------------------+
|              8 |          7 | transfer         | 300.00 | 2024-09-28 10:00:00 |
+----------------+------------+------------------+--------+---------------------+
1 row in set (0.00 sec)
```

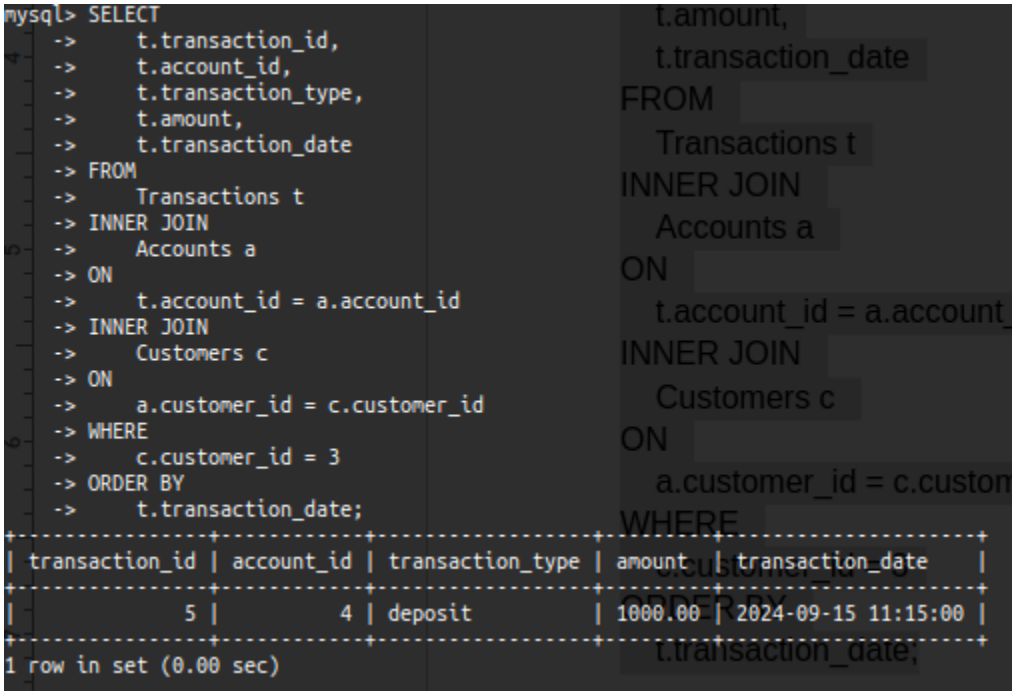
7. Identify customers who have accounts of multiple types.

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name
    -> FROM
    ->     Customers c
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     c.customer_id = a.customer_id
    -> GROUP BY
    ->     c.customer_id, c.first_name, c.last_name
    -> HAVING
    ->     COUNT(DISTINCT a.account_type) > 1;
+-------------+------------+-----------+
| customer_id | first_name | last_name |
+-------------+------------+-----------+
|           1 | John       | Doe       |
+-------------+------------+-----------+
1 row in set (0.00 sec)
```

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

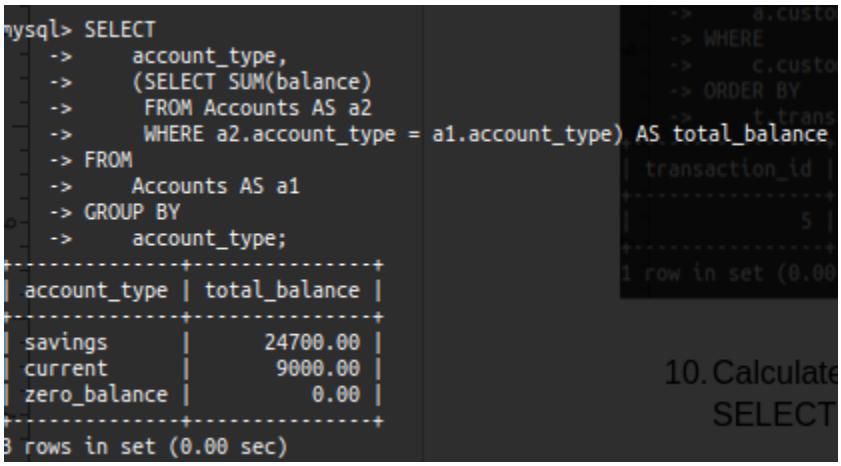
```
mysql> SELECT
    ->     account_type,
    ->     COUNT(account_id) AS total_accounts,
    ->     ROUND((COUNT(account_id) / (SELECT COUNT(*) FROM Accounts) * 100), 2) AS percentage
    -> FROM
    ->     Accounts
    -> GROUP BY
    ->     account_type;
+--------------+----------------+------------+
| account_type | total_accounts | percentage |
+--------------+----------------+------------+
| savings      |              5 |      45.45 |
| current      |              4 |      36.36 |
| zero_balance |              2 |      18.18 |
+--------------+----------------+------------+
3 rows in set (0.00 sec)
```

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



```
mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> INNER JOIN
    ->     Accounts a
    -> ON
    ->     t.account_id = a.account_id
    -> INNER JOIN
    ->     Customers c
    -> ON
    ->     a.customer_id = c.customer_id
    -> WHERE
    ->     c.customer_id = 3
    -> ORDER BY
    ->     t.transaction_date;
+----------------+------------+------------------+---------+---------------------+
| transaction_id | account_id | transaction_type | amount  | transaction_date    |
+----------------+------------+------------------+---------+---------------------+
|              5 |          4 | deposit          | 1000.00 | 2024-09-15 11:15:00 |
+----------------+------------+------------------+---------+---------------------+
1 row in set (0.00 sec)
```

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



```
mysql> SELECT
    ->     account_type,
    ->     (SELECT SUM(balance)
    ->      FROM Accounts AS a2
    ->      WHERE a2.account_type = a1.account_type) AS total_balance
    -> FROM
    ->     Accounts AS a1
    -> GROUP BY
    ->     account_type;
+--------------+---------------+
| account_type | total_balance |
+--------------+---------------+
| savings      |      24700.00 |
| current      |       9000.00 |
| zero_balance |          0.00 |
+--------------+---------------+
3 rows in set (0.00 sec)
```