Assignment 3 SQL

By- Kunal Sudhir Mishra

Task 1. Database Design:

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

```
mysql> CREATE DATABASE HMBank;
Query OK, 1 row affected (0.01 sec)

mysql> use HMBank;
Database changed
```

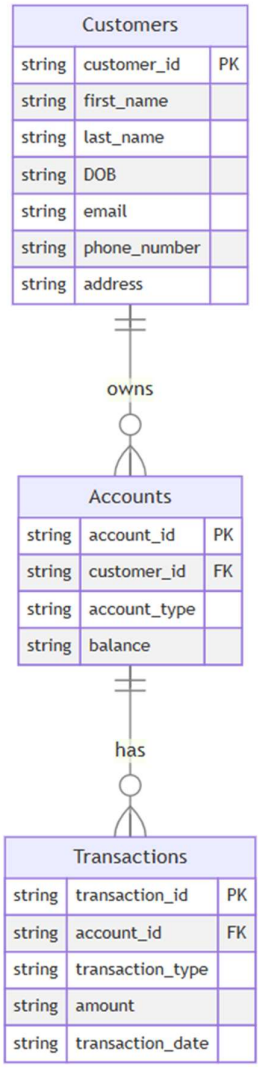
2. **Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema.**

```
mysql> CREATE TABLE Customers (
    ->   customer_id INT PRIMARY KEY,
    ->   first_name VARCHAR(50),
    ->   last_name VARCHAR(50),
    ->   DOB DATE,
    ->   email VARCHAR(100),
    ->   phone_number VARCHAR(20),
    ->   address VARCHAR(100)
    -> );CREATE TABLE Accounts (
Query OK, 0 rows affected (0.04 sec)

    ->   account_id INT PRIMARY KEY,
    ->   customer_id INT,
    ->   account_type VARCHAR(50),
    ->   balance DECIMAL(10, 2),
    ->   FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
    -> );CREATE TABLE Transactions (
Query OK, 0 rows affected (0.04 sec)

    ->   transaction_id INT PRIMARY KEY,
    ->   account_id INT,
    ->   transaction_type VARCHAR(50),
    ->   amount DECIMAL(10, 2),
    ->   transaction_date DATE,
    ->   FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

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



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

```
mysql> desc Customers;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| customer_id  | int          | NO   | PRI | NULL    |       |
| first_name   | varchar(50)  | YES  |     | NULL    |       |
| last_name    | varchar(50)  | YES  |     | NULL    |       |
| DOB          | date         | YES  |     | NULL    |       |
| email        | varchar(100) | YES  |     | NULL    |       |
| phone_number | varchar(20)  | YES  |     | NULL    |       |
| address      | varchar(100) | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
7 rows in set (0.01 sec)

mysql> desc Accounts;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| account_id   | int           | NO   | PRI | NULL    |       |
| customer_id  | int           | YES  | MUL | NULL    |       |
| account_type | varchar(50)   | YES  |     | NULL    |       |
| balance      | decimal(10,2) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> desc Transactions;
+------------------+---------------+------+-----+---------+-------+
| Field            | Type          | Null | Key | Default | Extra |
+------------------+---------------+------+-----+---------+-------+
| transaction_id   | int           | NO   | PRI | NULL    |       |
| account_id       | int           | YES  | MUL | NULL    |       |
| transaction_type | varchar(50)   | YES  |     | NULL    |       |
| amount           | decimal(10,2) | YES  |     | NULL    |       |
| transaction_date | date          | YES  |     | NULL    |       |
+------------------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

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

```
mysql> CREATE TABLE Customers (
    ->   customer_id INT PRIMARY KEY,
    ->   first_name VARCHAR(50),
    ->   last_name VARCHAR(50),
    ->   DOB DATE,
    ->   email VARCHAR(100),
    ->   phone_number VARCHAR(20),
    ->   address VARCHAR(100)
    -> );CREATE TABLE Accounts (
Query OK, 0 rows affected (0.04 sec)

    ->   account_id INT PRIMARY KEY,
    ->   customer_id INT,
    ->   account_type VARCHAR(50),
    ->   balance DECIMAL(10, 2),
    ->   FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
    -> );CREATE TABLE Transactions (
Query OK, 0 rows affected (0.04 sec)

    ->   transaction_id INT PRIMARY KEY,
    ->   account_id INT,
    ->   transaction_type VARCHAR(50),
    ->   amount DECIMAL(10, 2),
    ->   transaction_date DATE,
    ->   FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

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

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

```
mysql> INSERT INTO
    ->   Customers (
    ->     customer_id,
    ->     first_name,
    ->     last_name,
    ->     DOB,
    ->     email,
    ->     phone_number,
    ->     address
    ->   )
    -> VALUES
    ->   (
    ->     1,
    ->     'John',
    ->     'Doe',
    ->     '1990-01-01',
    ->     'johndoe@gmail.com',
    ->     '1234567890',
    ->     '123 Main St'
    ->   ),
    ->   (
    ->     2,
    ->     'Jane',
    ->     'Smith',
    ->     '1985-05-10',
    ->     'janesmith@gmail.com',
    ->     '9876543210',
    ->     '456 Elm St'
    ->   ),
    ->   (
```

```
mysql> select * from Customers;
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+
| customer_id | first_name | last_name | DOB        | email                    | phone_number | address       |
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+
|           1 | John       | Doe       | 1990-01-01 | johndoe@gmail.com        | 1234567890   | 123 Main St   |
|           2 | Jane       | Smith     | 1985-05-10 | janesmith@gmail.com      | 9876543210   | 456 Elm St    |
|           3 | Michael    | Johnson   | 1992-07-15 | michaeljohnson@gmail.com | 4567891230   | 789 Oak St    |
|           4 | Emily      | Brown     | 1988-03-20 | emilybrown@gmail.com     | 3216549870   | 987 Pine St   |
|           5 | David      | Wilson    | 1995-09-25 | davidwilson@gmail.com    | 7891234560   | 654 Maple St  |
|           6 | Sarah      | Taylor    | 1993-11-30 | sarahtaylor@gmail.com    | 6543217890   | 321 Cedar St  |
|           7 | Matthew    | Anderson  | 1991-02-05 | matthewanderson@gmail.com | 9870123456   | 456 Birch St  |
|           8 | Olivia     | Thomas    | 1987-06-15 | oliviathomas@gmail.com   | 0123456789   | 789 Walnut St |
|           9 | Daniel     | Robinson  | 1994-10-10 | danielrobinson@gmail.com | 9876543210   | 123 Oak St    |
|          10 | Sophia     | Harris    | 1996-12-25 | sophiaharris@gmail.com   | 0123456789   | 456 Pine St   |
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+
10 rows in set (0.00 sec)
```

- **Accounts**

```
mysql> INSERT INTO
    ->   Accounts (account_id, customer_id, account_type, balance)
    -> VALUES
    ->   (1, 1, 'savings', 1000),
    ->   (2, 2, 'current', 500),
    ->   (3, 3, 'savings', 2000),
    ->   (4, 4, 'current', 1500),
    ->   (5, 5, 'savings', 3000),
    ->   (6, 6, 'current', 2500),
    ->   (7, 7, 'savings', 4000),
    ->   (8, 8, 'current', 3500),
    ->   (9, 9, 'savings', 5000),
    ->   (10, 10, 'current', 4500);
Query OK, 10 rows affected (0.00 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

- **Transactions**

```
mysql> INSERT INTO
    ->   Transactions (
    ->     transaction_id,
    ->     account_id,
    ->     transaction_type,
    ->     amount,
    ->     transaction_date
    ->   )
    -> VALUES
    ->   (1, 1, 'deposit', 500, '2021-01-01'),
    ->   (2, 2, 'withdrawal', 200, '2021-01-02'),
    ->   (3, 3, 'deposit', 1000, '2021-01-03'),
    ->   (4, 4, 'withdrawal', 500, '2021-01-04'),
    ->   (5, 5, 'deposit', 1500, '2021-01-05'),
    ->   (6, 6, 'withdrawal', 1000, '2021-01-06'),
    ->   (7, 7, 'deposit', 2000, '2021-01-07'),
    ->   (8, 8, 'withdrawal', 1500, '2021-01-08'),
    ->   (9, 9, 'deposit', 2500, '2021-01-09'),
    ->   (10, 10, 'withdrawal', 2000, '2021-01-10');
Query OK, 10 rows affected (0.01 sec)
Records: 10  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.**

```
mysql> SELECT
    ->   first_name,
    ->   last_name,
    ->   account_type,
    ->   email
    -> FROM
    ->   Customers
    ->   JOIN Accounts ON Customers.customer_id = Accounts.customer_id
    -> ;
+------------+-----------+--------------+--------------------------+
| first_name | last_name | account_type | email                    |
+------------+-----------+--------------+--------------------------+
| John       | Doe       | savings      | johndoe@gmail.com        |
| Jane       | Smith     | current      | janesmith@gmail.com      |
| Michael    | Johnson   | savings      | michaeljohnson@gmail.com |
| Emily      | Brown     | current      | emilybrown@gmail.com     |
| David      | Wilson    | savings      | davidwilson@gmail.com    |
| Sarah      | Taylor    | current      | sarahtaylor@gmail.com    |
| Matthew    | Anderson  | savings      | matthewanderson@gmail.com|
| Olivia     | Thomas    | current      | oliviathomas@gmail.com   |
| Daniel     | Robinson  | savings      | danielrobinson@gmail.com |
| Sophia     | Harris    | current      | sophiaharris@gmail.com   |
+------------+-----------+--------------+--------------------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   t.transaction_id,
    ->   c.first_name,
    ->   c.last_name,
    ->   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;
+----------------+------------+-----------+------------------+---------+------------------+
| transaction_id | first_name | last_name | transaction_type | amount  | transaction_date |
+----------------+------------+-----------+------------------+---------+------------------+
|              1 | John       | Doe       | deposit          |  500.00 | 2021-01-01       |
|              2 | Jane       | Smith     | withdrawal       |  200.00 | 2021-01-02       |
|              3 | Michael    | Johnson   | deposit          | 1000.00 | 2021-01-03       |
|              4 | Emily      | Brown     | withdrawal       |  500.00 | 2021-01-04       |
|              5 | David      | Wilson    | deposit          | 1500.00 | 2021-01-05       |
|              6 | Sarah      | Taylor    | withdrawal       | 1000.00 | 2021-01-06       |
|              7 | Matthew    | Anderson  | deposit          | 2000.00 | 2021-01-07       |
|              8 | Olivia     | Thomas    | withdrawal       | 1500.00 | 2021-01-08       |
|              9 | Daniel     | Robinson  | deposit          | 2500.00 | 2021-01-09       |
|             10 | Sophia     | Harris    | withdrawal       | 2000.00 | 2021-01-10       |
+----------------+------------+-----------+------------------+---------+------------------+
10 rows in set (0.00 sec)
```

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

```
mysql> UPDATE
    ->   Accounts
    -> SET
    ->   balance = balance + 2000
    -> WHERE
    ->   account_id = 3;
Query OK, 1 row affected (0.01 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
    -> FROM
    ->   Customers;
+-----------------+
| full_name       |
+-----------------+
| John Doe        |
| Jane Smith      |
| Michael Johnson |
| Emily Brown     |
| David Wilson    |
| Sarah Taylor    |
| Matthew Anderson|
| Olivia Thomas   |
| Daniel Robinson |
| Sophia Harris   |
+-----------------+
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
    -> *
    -> FROM
    ->   Customers
    -> WHERE
    ->   address LIKE '654 Maple St';
+-------------+------------+-----------+------------+---------------------+--------------+--------------+
| customer_id | first_name | last_name | DOB        | email               | phone_number | address      |
+-------------+------------+-----------+------------+---------------------+--------------+--------------+
|           5 | David      | Wilson    | 1995-09-25 | davidwilson@gmail.com | 7891234560   | 654 Maple St |
+-------------+------------+-----------+------------+---------------------+--------------+--------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   balance
    -> FROM
    ->   Accounts
    -> WHERE
    ->   account_id = 4;
+---------+
| balance |
+---------+
| 1500.00 |
+---------+
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
    ->   *
    -> FROM
    ->   Accounts
    -> WHERE
    ->   account_type = 'current'
    ->   AND balance > 1000;
+------------+-------------+--------------+---------+
| account_id | customer_id | account_type | balance |
+------------+-------------+--------------+---------+
|          4 |           4 | current      | 1500.00 |
|          6 |           6 | current      | 2500.00 |
|          8 |           8 | current      | 3500.00 |
|         10 |          10 | current      | 4500.00 |
+------------+-------------+--------------+---------+
4 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   *
    -> FROM
    ->   Transactions
    -> WHERE
    ->   account_id = 4;
+----------------+------------+------------------+--------+------------------+
| transaction_id | account_id | transaction_type | amount | transaction_date |
+----------------+------------+------------------+--------+------------------+
|              4 |          4 | withdrawal       | 500.00 | 2021-01-04       |
+----------------+------------+------------------+--------+------------------+
1 row 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,
    ->   balance * .04 AS interest_accrued
    -> FROM
    ->   Accounts
    -> WHERE
    ->   account_type = 'savings';
+------------+------------------+
| account_id | interest_accrued |
+------------+------------------+
|          1 |          40.0000 |
|          3 |         160.0000 |
|          5 |         120.0000 |
|          7 |         160.0000 |
|          9 |         200.0000 |
+------------+------------------+
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
    ->   *
    -> FROM
    ->   Accounts
    -> WHERE
    ->   balance < 2000;
+------------+-------------+--------------+---------+
| account_id | customer_id | account_type | balance |
+------------+-------------+--------------+---------+
|          1 |           1 | savings      | 1000.00 |
|          2 |           2 | current      |  500.00 |
|          4 |           4 | current      | 1500.00 |
+------------+-------------+--------------+---------+
3 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   *
    -> FROM
    ->   Customers
    -> WHERE
    ->   address NOT LIKE ' 321 Cedar St'
    -> ;
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+
| customer_id | first_name | last_name | DOB        | email                    | phone_number | address       |
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+
|           1 | John       | Doe       | 1990-01-01 | johndoe@gmail.com        | 1234567890   | 123 Main St   |
|           2 | Jane       | Smith     | 1985-05-10 | janesmith@gmail.com      | 9876543210   | 456 Elm St    |
|           3 | Michael    | Johnson   | 1992-07-15 | michaeljohnson@gmail.com | 4567891230   | 789 Oak St    |
|           4 | Emily      | Brown     | 1988-03-20 | emilybrown@gmail.com     | 3216549870   | 987 Pine St   |
|           5 | David      | Wilson    | 1995-09-25 | davidwilson@gmail.com    | 7891234560   | 654 Maple St  |
|           6 | Sarah      | Taylor    | 1993-11-30 | sarahtaylor@gmail.com    | 6543217890   | 321 Cedar St  |
|           7 | Matthew    | Anderson  | 1991-02-05 | matthewanderson@gmail.com | 9870123456   | 456 Birch St  |
|           8 | Olivia     | Thomas    | 1987-06-15 | oliviathomas@gmail.com   | 0123456789   | 789 Walnut St |
|           9 | Daniel     | Robinson  | 1994-10-10 | danielrobinson@gmail.com | 9876543210   | 123 Oak St    |
|          10 | Sophia     | Harris    | 1996-12-25 | sophiaharris@gmail.com   | 0123456789   | 456 Pine St   |
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+
10 rows in set (0.00 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)
    -> FROM
    ->   Accounts;
+---------------+
| AVG(balance) |
+---------------+
|  2950.000000 |
+---------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   *
    -> FROM
    ->   Accounts
    -> ORDER BY
    ->   balance DESC
    -> LIMIT
    ->   10;
+------------+-------------+--------------+---------+
| account_id | customer_id | account_type | balance |
+------------+-------------+--------------+---------+
|          9 |           9 | savings      | 5000.00 |
|         10 |          10 | current      | 4500.00 |
|          3 |           3 | savings      | 4000.00 |
|          7 |           7 | savings      | 4000.00 |
|          8 |           8 | current      | 3500.00 |
|          5 |           5 | savings      | 3000.00 |
|          6 |           6 | current      | 2500.00 |
|          4 |           4 | current      | 1500.00 |
|          1 |           1 | savings      | 1000.00 |
|          2 |           2 | current      |  500.00 |
+------------+-------------+--------------+---------+
10 rows in set (0.00 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 =  2021-01-04;
+----------------+
| total_deposits |
+----------------+
|           NULL |
+----------------+
1 row in set, 1 warning (0.00 sec)
```

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

```
mysql> SELECT
    ->   first_name,
    ->   last_name,
    ->   DOB
    -> FROM
    ->   Customers
    -> ORDER BY
    ->   DOB DESC
    -> LIMIT
    ->   1;
+------------+-----------+------------+
| first_name | last_name | DOB        |
+------------+-----------+------------+
| Sophia     | Harris    | 1996-12-25 |
+------------+-----------+------------+
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,
    ->   t.transaction_type,
    ->   t.amount,
    ->   t.transaction_date,
    ->   a.account_type
    -> FROM
    ->   Transactions t
    ->   JOIN Accounts a ON t.account_id = a.account_id
    -> ;
+----------------+------------+------------------+---------+------------------+--------------+
| transaction_id | account_id | transaction_type | amount  | transaction_date | account_type |
+----------------+------------+------------------+---------+------------------+--------------+
|              1 |          1 | deposit          |  500.00 | 2021-01-01       | savings      |
|              2 |          2 | withdrawal       |  200.00 | 2021-01-02       | current      |
|              3 |          3 | deposit          | 1000.00 | 2021-01-03       | savings      |
|              4 |          4 | withdrawal       |  500.00 | 2021-01-04       | current      |
|              5 |          5 | deposit          | 1500.00 | 2021-01-05       | savings      |
|              6 |          6 | withdrawal       | 1000.00 | 2021-01-06       | current      |
|              7 |          7 | deposit          | 2000.00 | 2021-01-07       | savings      |
|              8 |          8 | withdrawal       | 1500.00 | 2021-01-08       | current      |
|              9 |          9 | deposit          | 2500.00 | 2021-01-09       | savings      |
|             10 |         10 | withdrawal       | 2000.00 | 2021-01-10       | current      |
+----------------+------------+------------------+---------+------------------+--------------+
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
    ->   Customers.customer_id,
    ->   Customers.first_name,
    ->   Customers.last_name,
    ->   Customers.DOB,
    ->   Customers.email,
    ->   Customers.phone_number,
    ->   Customers.address,
    ->   Accounts.account_id,
    ->   Accounts.account_type,
    ->   Accounts.balance
    -> FROM
    ->   Customers
    ->   INNER JOIN Accounts ON Customers.customer_id = Accounts.customer_id
    -> ;
```

```
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+------------+-----
------+---------+
| customer_id | first_name | last_name | DOB        | email                    | phone_number | address       | account_id | acc
_type | balance |
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+------------+-----
------+---------+
|           1 | John       | Doe       | 1990-01-01 | johndoe@gmail.com        | 1234567890   | 123 Main St   |          1 | savi
      | 1000.00 |
|           2 | Jane       | Smith     | 1985-05-10 | janesmith@gmail.com      | 9876543210   | 456 Elm St    |          2 | curr
      |  500.00 |
|           3 | Michael    | Johnson   | 1992-07-15 | michaeljohnson@gmail.com | 4567891230   | 789 Oak St    |          3 | savi
      | 4000.00 |
|           4 | Emily      | Brown     | 1988-03-20 | emilybrown@gmail.com     | 3216549870   | 987 Pine St   |          4 | curr
      | 1500.00 |
|           5 | David      | Wilson    | 1995-09-25 | davidwilson@gmail.com    | 7891234560   | 654 Maple St  |          5 | savi
      | 3000.00 |
|           6 | Sarah      | Taylor    | 1993-11-30 | sarahtaylor@gmail.com    | 6543217890   | 321 Cedar St  |          6 | curr
      | 2500.00 |
|           7 | Matthew    | Anderson  | 1991-02-05 | matthewanderson@gmail.com | 9870123456   | 456 Birch St  |          7 | savi
      | 4000.00 |
|           8 | Olivia     | Thomas    | 1987-06-15 | oliviathomas@gmail.com   | 0123456789   | 789 Walnut St |          8 | curr
      | 3500.00 |
|           9 | Daniel     | Robinson  | 1994-10-10 | danielrobinson@gmail.com | 9876543210   | 123 Oak St    |          9 | savi
      | 5000.00 |
|          10 | Sophia     | Harris    | 1996-12-25 | sophiaharris@gmail.com   | 0123456789   | 456 Pine St   |         10 | curr
      | 4500.00 |
+-------------+------------+-----------+------------+--------------------------+--------------+---------------+------------+-----
------+---------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   t.transaction_id,
    ->   t.account_id,
    ->   t.transaction_type,
    ->   t.amount,
    ->   t.transaction_date,
    ->   c.first_name,
    ->   c.last_name,
    ->   c.DOB,
    ->   c.email,
    ->   c.phone_number,
    ->   c.address
    -> FROM
    ->   Transactions t
    ->   JOIN Accounts a ON t.account_id = a.account_id
    ->   JOIN Customers c ON a.customer_id = c.customer_id
    -> WHERE
    ->   a.account_id = 3;
+----------------+------------+------------------+---------+------------------+------------+-----------+------------+-------------
----------+--------------+------------+
| transaction_id | account_id | transaction_type | amount  | transaction_date | first_name | last_name | DOB        | email
         | phone_number | address    |
+----------------+------------+------------------+---------+------------------+------------+-----------+------------+-------------
----------+--------------+------------+
|              3 |          3 | deposit          | 1000.00 | 2021-01-03       | Michael    | Johnson   | 1992-07-15 | michaeljohns
gmail.com | 4567891230   | 789 Oak St |
+----------------+------------+------------------+---------+------------------+------------+-----------+------------+-------------
----------+--------------+------------+
1 row in set (0.00 sec)
```

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
    -> 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;
Empty set (0.00 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 - amount END
    ->   ) AS difference
    -> FROM
    ->   Transactions
    -> ;
+------------+
| difference |
+------------+
|    2300.00 |
+------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   Accounts.account_id,
    ->   AVG(balance) AS average_daily_balance
    -> FROM
    ->   Accounts
    ->   JOIN Transactions ON Accounts.account_id = Transactions.account_id
    -> WHERE
    ->   transaction_date >= ' 2021-01-01'
    ->   AND transaction_date <= ' 2021-01-07'
    -> GROUP BY
    ->   Accounts.account_id;
+------------+----------------------+
| account_id | average_daily_balance |
+------------+----------------------+
|          1 |          1000.000000 |
|          2 |           500.000000 |
|          3 |          4000.000000 |
|          4 |          1500.000000 |
|          5 |          3000.000000 |
|          6 |          2500.000000 |
|          7 |          4000.000000 |
+------------+----------------------+
7 rows in set, 2 warnings (0.01 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      |      17000.00 |
| current      |      12500.00 |
+--------------+---------------+
2 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 num_transactions
    -> FROM
    ->   Transactions
    -> GROUP BY
    ->   account_id
    -> ORDER BY
    ->   num_transactions DESC
    -> ;
+------------+------------------+
| account_id | num_transactions |
+------------+------------------+
|          1 |                1 |
|          2 |                1 |
|          3 |                1 |
|          4 |                1 |
|          5 |                1 |
|          6 |                1 |
|          7 |                1 |
|          8 |                1 |
|          9 |                1 |
|         10 |                1 |
+------------+------------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   c.first_name,
    ->   c.last_name,
    ->   MIN(a.account_type) AS account_type,
    ->   SUM(a.balance) AS aggregate_balance
    -> FROM
    ->   Customers c
    ->   JOIN Accounts a ON c.customer_id = a.customer_id
    -> GROUP BY
    ->   c.customer_id
    -> HAVING
    ->   aggregate_balance > 3000;
+------------+-----------+--------------+-------------------+
| first_name | last_name | account_type | aggregate_balance |
+------------+-----------+--------------+-------------------+
| Michael    | Johnson   | savings      |           4000.00 |
| Matthew    | Anderson  | savings      |           4000.00 |
| Olivia     | Thomas    | current      |           3500.00 |
| Daniel     | Robinson  | savings      |           5000.00 |
| Sophia     | Harris    | current      |           4500.00 |
+------------+-----------+--------------+-------------------+
5 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   MIN(transaction_id) AS transaction_id,
    ->   amount,
    ->   transaction_date,
    ->   account_id
    -> FROM
    ->   Transactions
    -> GROUP BY
    ->   amount,
    ->   transaction_date,
    ->   account_id
    -> HAVING
    ->   COUNT(*) > 1;
Empty set (0.00 sec)
```

Tasks 4: Subquery and its type:

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

```
mysql> SELECT
    ->   c.first_name,
    ->   c.last_name
    -> FROM
    ->   Customers c
    ->   JOIN Accounts a ON c.customer_id = a.customer_id
    -> WHERE
    ->   a.balance = (
    ->     SELECT
    ->       MAX(balance)
    ->     FROM
    ->       Accounts
    ->   )
    -> ;
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| Daniel     | Robinson  |
+------------+-----------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   AVG(balance)
    -> FROM
    ->   Accounts
    -> WHERE
    ->   customer_id IN (
    ->     SELECT
    ->       customer_id
    ->     FROM
    ->       Accounts
    ->     GROUP BY
    ->       customer_id
    ->     HAVING
    ->       COUNT(*) > 1
    ->   )
    -> ;
+--------------+
| AVG(balance) |
+--------------+
|         NULL |
+--------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   a.account_id,
    ->   a.account_type,
    ->   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
    ->   )
    -> ;
+------------+--------------+----------------+---------+
| account_id | account_type | transaction_id | amount  |
+------------+--------------+----------------+---------+
|          5 | savings      |              5 | 1500.00 |
|          7 | savings      |              7 | 2000.00 |
|          8 | current      |              8 | 1500.00 |
|          9 | savings      |              9 | 2500.00 |
|         10 | current      |             10 | 2000.00 |
+------------+--------------+----------------+---------+
5 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;
Empty set (0.00 sec)
```

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

```
mysql> SELECT
    ->   SUM(balance)
    -> FROM
    ->   Accounts
    -> WHERE
    ->   account_id NOT IN (
    ->     SELECT
    ->       account_id
    ->     FROM
    ->       Transactions
    ->   )
    -> ;
+--------------+
| SUM(balance) |
+--------------+
|         NULL |
+--------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->   *
    -> FROM
    ->   Transactions
    -> WHERE
    ->   account_id IN (
    ->     SELECT
    ->       account_id
    ->     FROM
    ->       Accounts
    ->     WHERE
    ->       balance = (
    ->         SELECT
    ->           MIN(balance)
    ->         FROM
    ->           Accounts
    ->       )
    ->   )
    -> ;
+----------------+------------+------------------+--------+------------------+
| transaction_id | account_id | transaction_type | amount | transaction_date |
+----------------+------------+------------------+--------+------------------+
|              2 |          2 | withdrawal       | 200.00 | 2021-01-02       |
+----------------+------------+------------------+--------+------------------+
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
    -> ;
Empty set (0.00 sec)
```

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

```
mysql> SELECT
    ->   account_type,
    ->   COUNT(*) * 100.0 / (
    ->     SELECT
    ->       COUNT(*)
    ->     FROM
    ->       Accounts
    ->   ) AS percentage
    -> FROM
    ->   Accounts
    -> GROUP BY
    ->   account_type;
+--------------+------------+
| account_type | percentage |
+--------------+------------+
| savings      |   50.00000 |
| current      |   50.00000 |
+--------------+------------+
2 rows in set (0.01 sec)
```

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

```
mysql> SELECT
    ->   *
    -> FROM
    ->   Transactions
    -> WHERE
    ->   account_id IN (
    ->     SELECT
    ->       account_id
    ->     FROM
    ->       Accounts
    ->     WHERE
    ->       customer_id = 4
    ->   );
+----------------+------------+------------------+--------+------------------+
| transaction_id | account_id | transaction_type | amount | transaction_date |
+----------------+------------+------------------+--------+------------------+
|              4 |          4 | withdrawal       | 500.00 | 2021-01-04       |
+----------------+------------+------------------+--------+------------------+
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
    ->     WHERE
    ->       account_type = a.account_type
    ->   ) AS total_balance
    -> FROM
    ->   Accounts AS a
    -> GROUP BY
    ->   account_type
    -> ;
+--------------+---------------+
| account_type | total_balance |
+--------------+---------------+
| savings      |      17000.00 |
| current      |      12500.00 |
+--------------+---------------+
2 rows in set (0.00 sec)
```