Assignment – 3

Tasks 1: Database Design:

1. Create the database named "HMBank"

Ans.

CREATE DATABASE HMBank;

USE HMBank;

Command Prompt - mysql -u root -p

```
mysql> CREATE DATABASE HMBank;
Query OK, 1 row affected (0.04 sec)

mysql> USE HMBank;
Database changed
mysql> select database();
+------------+
| database() |
+------------+
| hmbank     |
+------------+
1 row in set (0.00 sec)

mysql>
```

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

Ans.

```
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(255)
);


CREATE TABLE Accounts (
   account_id INT PRIMARY KEY,
   customer_id INT,
```

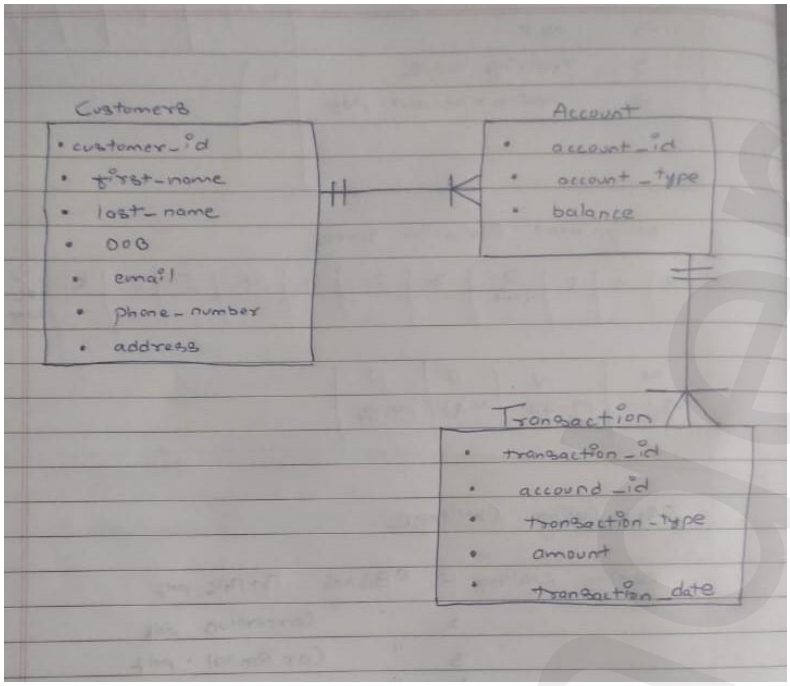
```
    account_type VARCHAR(20),
    balance DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

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)
);
```

Command Prompt - mysql -u root -p

```
1 row in set (0.00 sec)

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(255)
    -> );
Query OK, 0 rows affected (0.11 sec)

mysql>
mysql> 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)
    -> );
Query OK, 0 rows affected (0.12 sec)

mysql>
mysql> 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)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
```

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

Ans.



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

Ans. Primary Key (PK) and Foreign Key (FK) constraints are already added in the table creation scripts.

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

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

Customers

Accounts

Transactions

Ans.

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', '1985-08-20', 'jane.smith@example.com', '987-654-3210', '456 Oak Ave'),

(3, 'Bob', 'Johnson', '1978-12-03', 'bob.johnson@example.com', '555-123-4567', '789 Pine Ln'),

(4, 'Alice', 'Williams', '1995-02-28', 'alice.williams@example.com', '111-222-3333', '987 Cedar Dr'),

(5, 'Charlie', 'Brown', '1980-10-10', 'charlie.brown@example.com', '999-888-7777', '654 Birch Rd'),

(6, 'Eva', 'Martinez', '1992-07-17', 'eva.martinez@example.com', '444-555-6666', '321 Elm St'),

(7, 'David', 'Lee', '1987-04-05', 'david.lee@example.com', '777-888-9999', '234 Maple Ave'),

(8, 'Grace', 'Taylor', '1983-09-12', 'grace.taylor@example.com', '666-555-4444', '876 Pine Ln'),

(9, 'Frank', 'Brown', '1998-11-22', 'frank.brown@example.com', '222-333-4444', '543 Oak Ave'),

(10, 'Helen', 'Nguyen', '1994-01-18', 'helen.nguyen@example.com', '333-444-5555', '765 Maple Dr');

Command Prompt - mysql -u root -p

```
mysql> 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', '1985-08-20', 'jane.smith@example.com', '987-654-3210', '456 Oak Ave'),
    ->     (3, 'Bob', 'Johnson', '1978-12-03', 'bob.johnson@example.com', '555-123-4567', '789 Pine Ln'),
    ->     (4, 'Alice', 'Williams', '1995-02-28', 'alice.williams@example.com', '111-222-3333', '987 Cedar Dr'),
    ->     (5, 'Charlie', 'Brown', '1980-10-10', 'charlie.brown@example.com', '999-888-7777', '654 Birch Rd'),
    ->     (6, 'Eva', 'Martinez', '1992-07-17', 'eva.martinez@example.com', '444-555-6666', '321 Elm St'),
    ->     (7, 'David', 'Lee', '1987-04-05', 'david.lee@example.com', '777-888-9999', '234 Maple Ave'),
    ->     (8, 'Grace', 'Taylor', '1983-09-12', 'grace.taylor@example.com', '666-555-4444', '876 Pine Ln'),
    ->     (9, 'Frank', 'Brown', '1998-11-22', 'frank.brown@example.com', '222-333-4444', '543 Oak Ave'),
    ->     (10, 'Helen', 'Nguyen', '1994-01-18', 'helen.nguyen@example.com', '333-444-5555', '765 Maple Dr');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql>
```

INSERT INTO Accounts (account_id, customer_id, account_type, balance)

VALUES

(101, 1, 'savings', 5000.00),

(102, 2, 'current', 10000.00),

(103, 3, 'savings', 7500.50),

(104, 4, 'current', 12000.75),

(105, 5, 'savings', 3000.25),

(106, 6, 'current', 9000.00),

(107, 7, 'savings', 6000.50),

(108, 8, 'current', 15000.00),

(109, 9, 'savings', 2000.75),

(110, 10, 'current', 18000.25);

Command Prompt - mysql -u root -p

```
mysql> INSERT INTO Accounts (account_id, customer_id, account_type, balance)
    -> VALUES
    ->     (101, 1, 'savings', 5000.00),
    ->     (102, 2, 'current', 10000.00),
    ->     (103, 3, 'savings', 7500.50),
    ->     (104, 4, 'current', 12000.75),
    ->     (105, 5, 'savings', 3000.25),
    ->     (106, 6, 'current', 9000.00),
    ->     (107, 7, 'savings', 6000.50),
    ->     (108, 8, 'current', 15000.00),
    ->     (109, 9, 'savings', 2000.75),
    ->     (110, 10, 'current', 18000.25);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql>
```

INSERT INTO Transactions (transaction_id, account_id, transaction_type, amount, transaction_date)

VALUES

(1001, 101, 'deposit', 1000.00, '2023-12-01'),

(1002, 102, 'withdrawal', 500.50, '2023-12-02'),

(1003, 103, 'deposit', 1500.75, '2023-12-03'),

(1004, 104, 'transfer', 200.25, '2023-12-04'),

(1005, 105, 'deposit', 800.00, '2023-12-05'),

(1006, 106, 'withdrawal', 1000.50, '2023-12-06'),

(1007, 107, 'transfer', 300.25, '2023-12-07'),

(1008, 108, 'deposit', 1200.75, '2023-12-08'),

(1009, 109, 'withdrawal', 50.00, '2023-12-09'),

(1010, 110, 'deposit', 600.25, '2023-12-10');

Command Prompt - mysql -u root -p

```
mysql> INSERT INTO Transactions (transaction_id, account_id, transaction_type, amount, transaction_date)
    -> VALUES
    ->     (1001, 101, 'deposit', 1000.00, '2023-12-01'),
    ->     (1002, 102, 'withdrawal', 500.50, '2023-12-02'),
    ->     (1003, 103, 'deposit', 1500.75, '2023-12-03'),
    ->     (1004, 104, 'transfer', 200.25, '2023-12-04'),
    ->     (1005, 105, 'deposit', 800.00, '2023-12-05'),
    ->     (1006, 106, 'withdrawal', 1000.50, '2023-12-06'),
    ->     (1007, 107, 'transfer', 300.25, '2023-12-07'),
    ->     (1008, 108, 'deposit', 1200.75, '2023-12-08'),
    ->     (1009, 109, 'withdrawal', 50.00, '2023-12-09'),
    ->     (1010, 110, 'deposit', 600.25, '2023-12-10');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql>
```

2. Write SQL queries for the following tasks:

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

Ans.

```
SELECT
    t.transaction_id,
    c.first_name,
    c.last_name,
    a.account_id,
    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;
```

Command Prompt - mysql -u root -p

```
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      |
| Jane       | Smith     | current      | jane.smith@example.com    |
| Bob        | Johnson   | savings      | bob.johnson@example.com   |
| Alice      | Williams  | current      | alice.williams@example.com |
| Charlie    | Brown     | savings      | charlie.brown@example.com |
| Eva        | Martinez  | current      | eva.martinez@example.com  |
| David      | Lee       | savings      | david.lee@example.com     |
| Grace      | Taylor    | current      | grace.taylor@example.com  |
| Frank      | Brown     | savings      | frank.brown@example.com   |
| Helen      | Nguyen    | current      | helen.nguyen@example.com  |
+------------+-----------+--------------+---------------------------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT
```

```
    t.transaction_id,
    c.first_name,
    c.last_name,
    a.account_id,
    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;
```

Command Prompt - mysql -u root -p

```
10 rows in set (0.00 sec)

mysql> SELECT
    ->     t.transaction_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     a.account_id,
    ->     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 | account_id | transaction_type | amount  | transaction_date |
+----------------+------------+-----------+------------+------------------+---------+------------------+
|           1001 | John       | Doe       |        101 | deposit          | 1000.00 | 2023-12-01       |
|           1002 | Jane       | Smith     |        102 | withdrawal       |  500.50 | 2023-12-02       |
|           1003 | Bob        | Johnson   |        103 | deposit          | 1500.75 | 2023-12-03       |
|           1004 | Alice      | Williams  |        104 | transfer         |  200.25 | 2023-12-04       |
|           1005 | Charlie    | Brown     |        105 | deposit          |  800.00 | 2023-12-05       |
|           1006 | Eva        | Martinez  |        106 | withdrawal       | 1000.50 | 2023-12-06       |
|           1007 | David      | Lee       |        107 | transfer         |  300.25 | 2023-12-07       |
|           1008 | Grace      | Taylor    |        108 | deposit          | 1200.75 | 2023-12-08       |
|           1009 | Frank      | Brown     |        109 | withdrawal       |   50.00 | 2023-12-09       |
|           1010 | Helen      | Nguyen    |        110 | deposit          |  600.25 | 2023-12-10       |
+----------------+------------+-----------+------------+------------------+---------+------------------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

```
UPDATE Accounts
SET balance = balance + 250
WHERE account_id = 101;
```

Command Prompt - mysql -u root -p

```
mysql> select * from accounts;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|        101 |           1 | savings      |  5000.00 |
|        102 |           2 | current      | 10000.00 |
|        103 |           3 | savings      |  7500.50 |
|        104 |           4 | current      | 12000.75 |
|        105 |           5 | savings      |  3000.25 |
|        106 |           6 | current      |  9000.00 |
|        107 |           7 | savings      |  6000.50 |
|        108 |           8 | current      | 15000.00 |
|        109 |           9 | savings      |  2000.75 |
|        110 |          10 | current      | 18000.25 |
+------------+-------------+--------------+----------+
10 rows in set (0.00 sec)

mysql> UPDATE Accounts
    -> SET balance = balance + 250
    -> WHERE account_id = 101;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from accounts;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|        101 |           1 | savings      |  5250.00 |
|        102 |           2 | current      | 10000.00 |
|        103 |           3 | savings      |  7500.50 |
|        104 |           4 | current      | 12000.75 |
|        105 |           5 | savings      |  3000.25 |
|        106 |           6 | current      |  9000.00 |
|        107 |           7 | savings      |  6000.50 |
|        108 |           8 | current      | 15000.00 |
|        109 |           9 | savings      |  2000.75 |
|        110 |          10 | current      | 18000.25 |
+------------+-------------+--------------+----------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT
   customer_id,
   first_name,
   last_name,
   CONCAT(first_name, ' ', last_name) AS full_name
FROM
   Customers;
```

Command Prompt - mysql -u root -p

```
10 rows in set (0.00 sec)

mysql> SELECT
    ->     customer_id,
    ->     first_name,
    ->     last_name,
    ->     CONCAT(first_name, ' ', last_name) AS full_name
    -> FROM
    ->     Customers;
+-------------+------------+-----------+----------------+
| customer_id | first_name | last_name | full_name      |
+-------------+------------+-----------+----------------+
|           1 | John       | Doe       | John Doe       |
|           2 | Jane       | Smith     | Jane Smith     |
|           3 | Bob        | Johnson   | Bob Johnson    |
|           4 | Alice      | Williams  | Alice Williams |
|           5 | Charlie    | Brown     | Charlie Brown  |
|           6 | Eva        | Martinez  | Eva Martinez   |
|           7 | David      | Lee       | David Lee      |
|           8 | Grace      | Taylor    | Grace Taylor   |
|           9 | Frank      | Brown     | Frank Brown    |
|          10 | Helen      | Nguyen    | Helen Nguyen   |
+-------------+------------+-----------+----------------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

DELETE FROM Accounts

WHERE balance = 0 AND account_type = 'savings';

Command Prompt - mysql -u root -p

```
mysql> DELETE FROM Accounts
    -> WHERE balance = 0 AND account_type = 'savings';
Query OK, 0 rows affected (0.03 sec)

mysql>
```

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

Ans.

SELECT *

FROM Customers

WHERE address LIKE '%' + [specific_city] + '%';

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

Ans.

SELECT account_id, account_type, balance

FROM Accounts

WHERE account_id = 101;

Command Prompt - mysql -u root -p

```
mysql> SELECT account_id, account_type, balance
    -> FROM Accounts
    -> WHERE account_id = 101;
+------------+--------------+---------+
| account_id | account_type | balance |
+------------+--------------+---------+
|        101 | savings      | 5250.00 |
+------------+--------------+---------+
1 row in set (0.00 sec)

mysql>
```

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

Ans.

SELECT *

FROM Accounts

WHERE account_type = 'current' AND balance > 1000.00;

Command Prompt - mysql -u root -p

```
1 row in set (0.00 sec)

mysql> SELECT *
    -> FROM Accounts
    -> WHERE account_type = 'current' AND balance > 1000.00;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|        102 |           2 | current      | 10000.00 |
|        104 |           4 | current      | 12000.75 |
|        106 |           6 | current      |  9000.00 |
|        108 |           8 | current      | 15000.00 |
|        110 |          10 | current      | 18000.25 |
+------------+-------------+--------------+----------+
5 rows in set (0.00 sec)

mysql>
```

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

Ans.

SELECT *

FROM Transactions

WHERE account_id = 101;

Command Prompt - mysql -u root -p

```
mysql> SELECT *
    -> FROM Transactions
    -> WHERE account_id = 101;
+----------------+------------+------------------+---------+------------------+
| transaction_id | account_id | transaction_type | amount  | transaction_date |
+----------------+------------+------------------+---------+------------------+
|           1001 |        101 | deposit          | 1000.00 | 2023-12-01       |
+----------------+------------+------------------+---------+------------------+
1 row in set (0.00 sec)

mysql>
```

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

Ans.

UPDATE Accounts

SET balance = balance + (balance * 0.10)

WHERE account_type = 'savings';

Command Prompt - mysql -u root -p

```
mysql> UPDATE Accounts
    -> SET balance = balance + (balance * 0.10)
    -> WHERE account_type = 'savings';
Query OK, 5 rows affected, 2 warnings (0.03 sec)
Rows matched: 5  Changed: 5  Warnings: 2

mysql> select * from accounts;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|        101 |           1 | savings      |  5775.00 |
|        102 |           2 | current      | 10000.00 |
|        103 |           3 | savings      |  8250.55 |
|        104 |           4 | current      | 12000.75 |
|        105 |           5 | savings      |  3300.28 |
|        106 |           6 | current      |  9000.00 |
|        107 |           7 | savings      |  6600.55 |
|        108 |           8 | current      | 15000.00 |
|        109 |           9 | savings      |  2200.83 |
|        110 |          10 | current      | 18000.25 |
+------------+-------------+--------------+----------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

SELECT *

FROM Accounts

WHERE balance < [overdraft_limit];

Command Prompt - mysql -u root -p

```
10 rows in set (0.00 sec)

mysql> SELECT *
    -> FROM Accounts
    -> WHERE balance < 9000;
+------------+-------------+--------------+---------+
| account_id | customer_id | account_type | balance |
+------------+-------------+--------------+---------+
|        101 |           1 | savings      | 5775.00 |
|        103 |           3 | savings      | 8250.55 |
|        105 |           5 | savings      | 3300.28 |
|        107 |           7 | savings      | 6600.55 |
|        109 |           9 | savings      | 2200.83 |
+------------+-------------+--------------+---------+
5 rows in set (0.00 sec)

mysql>
```

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

Ans.

SELECT *

FROM Customers

WHERE address NOT LIKE '%' + [specific_city] + '%';

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

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

Ans.

SELECT AVG(balance) AS average_balance

FROM Accounts;

Command Prompt - mysql -u root -p

```
mysql> SELECT AVG(balance) AS average_balance
    -> FROM Accounts;
+-----------------+
| average_balance |
+-----------------+
|     9012.821000 |
+-----------------+
1 row in set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT *

FROM Accounts

ORDER BY balance DESC

LIMIT 10;
```

Command Prompt - mysql -u root -p

```
mysql> SELECT *
    -> FROM Accounts
    -> ORDER BY balance DESC
    -> LIMIT 10;
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|        110 |          10 | current      | 18000.25 |
|        108 |           8 | current      | 15000.00 |
|        104 |           4 | current      | 12000.75 |
|        102 |           2 | current      | 10000.00 |
|        106 |           6 | current      |  9000.00 |
|        103 |           3 | savings      |  8250.55 |
|        107 |           7 | savings      |  6600.55 |
|        101 |           1 | savings      |  5775.00 |
|        105 |           5 | savings      |  3300.28 |
|        109 |           9 | savings      |  2200.83 |
+------------+-------------+--------------+----------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT

   t.transaction_date,

   SUM(t.amount) AS total_deposits

FROM

   Transactions t

WHERE

   t.transaction_type = 'deposit'

   AND t.transaction_date = '2023-12-03';
```

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     t.transaction_date,
    ->     SUM(t.amount) AS total_deposits
    -> FROM
    ->     Transactions t
    -> WHERE
    ->     t.transaction_type = 'deposit'
    ->     AND t.transaction_date = '2023-12-03';
+------------------+----------------+
| transaction_date | total_deposits |
+------------------+----------------+
| 2023-12-03       |        1500.75 |
+------------------+----------------+
1 row in set (0.03 sec)

mysql>
```

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

Ans.

SELECT

MIN(DOB) AS oldest_customer_dob,

MAX(DOB) AS newest_customer_dob

FROM

Customers;

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     MIN(DOB) AS oldest_customer_dob,
    ->     MAX(DOB) AS newest_customer_dob
    -> FROM
    ->     Customers;
+---------------------+---------------------+
| oldest_customer_dob | newest_customer_dob |
+---------------------+---------------------+
| 1978-12-03          | 1998-11-22          |
+---------------------+---------------------+
1 row in set (0.00 sec)

mysql>
```

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

Ans.

SELECT

t.transaction_id,

t.account_id,

```
    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;
```

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     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;
+----------------+------------+--------------+------------------+---------+------------------+
| transaction_id | account_id | account_type | transaction_type | amount  | transaction_date |
+----------------+------------+--------------+------------------+---------+------------------+
|           1001 |        101 | savings      | deposit          | 1000.00 | 2023-12-01       |
|           1002 |        102 | current      | withdrawal       |  500.50 | 2023-12-02       |
|           1003 |        103 | savings      | deposit          | 1500.75 | 2023-12-03       |
|           1004 |        104 | current      | transfer         |  200.25 | 2023-12-04       |
|           1005 |        105 | savings      | deposit          |  800.00 | 2023-12-05       |
|           1006 |        106 | current      | withdrawal       | 1000.50 | 2023-12-06       |
|           1007 |        107 | savings      | transfer         |  300.25 | 2023-12-07       |
|           1008 |        108 | current      | deposit          | 1200.75 | 2023-12-08       |
|           1009 |        109 | savings      | withdrawal       |   50.00 | 2023-12-09       |
|           1010 |        110 | current      | deposit          |  600.25 | 2023-12-10       |
+----------------+------------+--------------+------------------+---------+------------------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

```
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;

Command Prompt - mysql -u root -p

```
mysql> 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;
+-------------+------------+-----------+------------+---------------------------+--------------+---------------+------------+--------------+----------+
| 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   |        101 | savings      |  5775.00 |
|           2 | Jane       | Smith     | 1985-08-20 | jane.smith@example.com    | 987-654-3210 | 456 Oak Ave   |        102 | current      | 10000.00 |
|           3 | Bob        | Johnson   | 1978-12-03 | bob.johnson@example.com   | 555-123-4567 | 789 Pine Ln   |        103 | savings      |  8250.55 |
|           4 | Alice      | Williams  | 1995-02-28 | alice.williams@example.com | 111-222-3333 | 987 Cedar Dr  |        104 | current      | 12000.75 |
|           5 | Charlie    | Brown     | 1980-10-10 | charlie.brown@example.com | 999-888-7777 | 654 Birch Rd  |        105 | savings      |  3300.28 |
|           6 | Eva        | Martinez  | 1992-07-17 | eva.martinez@example.com  | 444-555-6666 | 321 Elm St    |        106 | current      |  9000.00 |
|           7 | David      | Lee       | 1987-04-05 | david.lee@example.com     | 777-888-9999 | 234 Maple Ave |        107 | savings      |  6600.55 |
|           8 | Grace      | Taylor    | 1983-09-12 | grace.taylor@example.com  | 666-555-4444 | 876 Pine Ln   |        108 | current      | 15000.00 |
|           9 | Frank      | Brown     | 1998-11-22 | frank.brown@example.com   | 222-333-4444 | 543 Oak Ave   |        109 | savings      |  2200.83 |
|          10 | Helen      | Nguyen    | 1994-01-18 | helen.nguyen@example.com  | 333-444-5555 | 765 Maple Dr  |        110 | current      | 18000.25 |
+-------------+------------+-----------+------------+---------------------------+--------------+---------------+------------+--------------+----------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

SELECT

t.transaction_id,

t.account_id,

a.account_type,

t.transaction_type,

t.amount,

t.transaction_date,

c.customer_id,

```
    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
    t.account_id = 101;
```

```
Command Prompt - mysql  -u root -p
    ->     t.account_id = [specific_account_id];
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '[specific_account_id]' at line 22
mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     a.account_type,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date,
    ->     c.customer_id,
    ->     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
    ->     t.account_id = 101;
+----------------+------------+--------------+------------------+---------+------------------+-------------+------------+-----------+------------+-----------------------+--------------+-------------+
| transaction_id | account_id | account_type | transaction_type | amount  | transaction_date | customer_id | first_name | last_name | DOB        | email                 | phone_number | address     |
+----------------+------------+--------------+------------------+---------+------------------+-------------+------------+-----------+------------+-----------------------+--------------+-------------+
|           1001 |        101 | savings      | deposit          | 1000.00 | 2023-12-01       |           1 | John       | Doe       | 1990-05-15 | john.doe@example.com | 123-456-7890 | 123 Main St |
+----------------+------------+--------------+------------------+---------+------------------+-------------+------------+-----------+------------+-----------------------+--------------+-------------+
1 row in set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    COUNT(a.account_id) AS num_accounts
```

FROM

Customers c

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;

Command Prompt - mysql -u root -p

```
1 row in set (0.00 sec)

mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     COUNT(a.account_id) AS num_accounts
    -> FROM
    ->     Customers c
    -> 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 (0.06 sec)

mysql>
```

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

Ans.

SELECT

account_id,

SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE 0 END) AS total_deposits,

SUM(CASE WHEN transaction_type = 'withdrawal' THEN amount ELSE 0 END) AS total_withdrawals,

SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS net_difference

FROM

Transactions

GROUP BY

account_id;

```
Command Prompt - mysql  -u root -p
Empty set (0.06 sec)

mysql> SELECT
    ->     account_id,
    ->     SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE 0 END) AS total_deposits,
    ->     SUM(CASE WHEN transaction_type = 'withdrawal' THEN amount ELSE 0 END) AS total_withdrawals,
    ->     SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS net_difference
    -> FROM
    ->     Transactions
    -> GROUP BY
    ->     account_id;
+------------+----------------+-------------------+----------------+
| account_id | total_deposits | total_withdrawals | net_difference |
+------------+----------------+-------------------+----------------+
|        101 |        1000.00 |              0.00 |        1000.00 |
|        102 |           0.00 |            500.50 |        -500.50 |
|        103 |        1500.75 |              0.00 |        1500.75 |
|        104 |           0.00 |              0.00 |        -200.25 |
|        105 |         800.00 |              0.00 |         800.00 |
|        106 |           0.00 |           1000.50 |       -1000.50 |
|        107 |           0.00 |              0.00 |        -300.25 |
|        108 |        1200.75 |              0.00 |        1200.75 |
|        109 |           0.00 |             50.00 |         -50.00 |
|        110 |         600.25 |              0.00 |         600.25 |
+------------+----------------+-------------------+----------------+
10 rows in set (0.03 sec)

mysql>
```

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

Ans.

```
SELECT
    t.account_id,
    AVG(t.balance) AS average_daily_balance
FROM (
    SELECT
        account_id,
        transaction_date,
        SUM(amount) OVER (PARTITION BY account_id ORDER BY transaction_date) AS balance
    FROM
        Transactions
) t
GROUP BY
    t.account_id;
```

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     t.account_id,
    ->     AVG(t.balance) AS average_daily_balance
    -> FROM (
    ->     SELECT
    ->         account_id,
    ->         transaction_date,
    ->         SUM(amount) OVER (PARTITION BY account_id ORDER BY transaction_date) AS balance
    ->     FROM
    ->         Transactions
    -> ) t
    -> GROUP BY
    ->     t.account_id;
+------------+----------------------+
| account_id | average_daily_balance |
+------------+----------------------+
|        101 |          1000.000000 |
|        102 |           500.500000 |
|        103 |          1500.750000 |
|        104 |           200.250000 |
|        105 |           800.000000 |
|        106 |          1000.500000 |
|        107 |           300.250000 |
|        108 |          1200.750000 |
|        109 |            50.000000 |
|        110 |           600.250000 |
+------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

11. Calculate the total balance for each account type.

Ans.

```
SELECT
    account_type,
    SUM(balance) AS total_balance
FROM
    Accounts
GROUP BY
    account_type;
```

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     account_type,
    ->     SUM(balance) AS total_balance
    -> FROM
    ->     Accounts
    -> GROUP BY
    ->     account_type;
+--------------+---------------+
| account_type | total_balance |
+--------------+---------------+
| savings      |      26127.21 |
| current      |      64001.00 |
+--------------+---------------+
2 rows in set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT
    account_id,
    COUNT(transaction_id) AS num_transactions
FROM
    Transactions
GROUP BY
    account_id
ORDER BY
    num_transactions DESC;
```

Command Prompt - mysql -u root -p

```
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 |
+------------+------------------+
|        101 |                1 |
|        102 |                1 |
|        103 |                1 |
|        104 |                1 |
|        105 |                1 |
|        106 |                1 |
|        107 |                1 |
|        108 |                1 |
|        109 |                1 |
|        110 |                1 |
+------------+------------------+
10 rows in set (0.01 sec)

mysql>
```

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

Ans.

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    SUM(a.balance) AS aggregate_balance
FROM
    Customers c
```

JOIN

Accounts a ON c.customer_id = a.customer_id

GROUP BY

c.customer_id, c.first_name, c.last_name

ORDER BY

aggregate_balance DESC;

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     SUM(a.balance) AS aggregate_balance
    -> FROM
    ->     Customers c
    -> JOIN
    ->     Accounts a ON c.customer_id = a.customer_id
    -> GROUP BY
    ->     c.customer_id, c.first_name, c.last_name
    -> ORDER BY
    ->     aggregate_balance DESC;
+-------------+------------+-----------+------------------+
| customer_id | first_name | last_name | aggregate_balance |
+-------------+------------+-----------+------------------+
|          10 | Helen      | Nguyen    |          18000.25 |
|           8 | Grace      | Taylor    |          15000.00 |
|           4 | Alice      | Williams  |          12000.75 |
|           2 | Jane       | Smith     |          10000.00 |
|           6 | Eva        | Martinez  |           9000.00 |
|           3 | Bob        | Johnson   |           8250.55 |
|           7 | David      | Lee       |           6600.55 |
|           1 | John       | Doe       |           5775.00 |
|           5 | Charlie    | Brown     |           3300.28 |
|           9 | Frank      | Brown     |           2200.83 |
+-------------+------------+-----------+------------------+
10 rows in set (0.00 sec)

mysql>
```

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

Ans.

SELECT

transaction_id,

account_id,

transaction_type,

amount,

transaction_date

FROM

Transactions

WHERE

(amount, transaction_date, account_id) IN (

SELECT

```
        amount,
        transaction_date,
        account_id
    FROM
        Transactions
    GROUP BY
        amount,
        transaction_date,
        account_id
    HAVING
        COUNT(transaction_id) > 1
);
```

Command Prompt - mysql -u root -p

```
; this is incompatible with sql_mode=only_full_group_by
mysql> SELECT
    ->     transaction_id,
    ->     account_id,
    ->     transaction_type,
    ->     amount,
    ->     transaction_date
    -> FROM
    ->     Transactions
    -> WHERE
    ->     (amount, transaction_date, account_id) IN (
    ->         SELECT
    ->             amount,
    ->             transaction_date,
    ->             account_id
    ->         FROM
    ->             Transactions
    ->         GROUP BY
    ->             amount,
    ->             transaction_date,
    ->             account_id
    ->         HAVING
    ->             COUNT(transaction_id) > 1
    ->     );
Empty set (0.03 sec)

mysql>
```

Tasks 4: Subquery and its type:

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

Ans.

```
SELECT
    c.customer_id,
    c.first_name,
```

```
    c.last_name,
    c.DOB,
    c.email,
    c.phone_number,
    c.address
FROM
    Customers c
JOIN
    Accounts a ON c.customer_id = a.customer_id
WHERE
    a.balance = (SELECT MAX(balance) FROM Accounts);
```

Command Prompt - mysql -u root -p

```
Empty set (0.03 sec)

mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name,
    ->     c.DOB,
    ->     c.email,
    ->     c.phone_number,
    ->     c.address
    -> FROM
    ->     Customers c
    -> JOIN
    ->     Accounts a ON c.customer_id = a.customer_id
    -> WHERE
    ->     a.balance = (SELECT MAX(balance) FROM Accounts);
+-------------+------------+-----------+------------+-------------------------+--------------+--------------+
| customer_id | first_name | last_name | DOB        | email                   | phone_number | address      |
+-------------+------------+-----------+------------+-------------------------+--------------+--------------+
|          10 | Helen      | Nguyen    | 1994-01-18 | helen.nguyen@example.com | 333-444-5555 | 765 Maple Dr |
+-------------+------------+-----------+------------+-------------------------+--------------+--------------+
1 row in set (0.01 sec)

mysql>
```

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

Ans.

```
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(account_id) > 1
    )
GROUP BY
    c.customer_id, c.first_name, c.last_name;
```

Command Prompt - mysql -u root -p

```
1 row in set (0.01 sec)

mysql> 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(account_id) > 1
    ->     )
    -> GROUP BY
    ->     c.customer_id, c.first_name, c.last_name;
Empty set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT
    a.account_id,
    a.customer_id,
    a.account_type,
```

```
    a.balance
FROM
    Accounts a
JOIN
    Transactions t ON a.account_id = t.account_id
WHERE
    t.amount > (SELECT AVG(amount) FROM Transactions);
```

Command Prompt - mysql -u root -p

```
Empty set (0.00 sec)

mysql> SELECT
    ->     a.account_id,
    ->     a.customer_id,
    ->     a.account_type,
    ->     a.balance
    -> FROM
    ->     Accounts a
    -> JOIN
    ->     Transactions t ON a.account_id = t.account_id
    -> WHERE
    ->     t.amount > (SELECT AVG(amount) FROM Transactions);
+------------+-------------+--------------+----------+
| account_id | customer_id | account_type | balance  |
+------------+-------------+--------------+----------+
|        101 |           1 | savings      |  5775.00 |
|        103 |           3 | savings      |  8250.55 |
|        105 |           5 | savings      |  3300.28 |
|        106 |           6 | current      |  9000.00 |
|        108 |           8 | current      | 15000.00 |
+------------+-------------+--------------+----------+
5 rows in set (0.00 sec)

mysql>
```

4. Identify customers who have no recorded transactions.

Ans.

```
SELECT
    customer_id,
    first_name,
    last_name,
    DOB,
    email,
    phone_number,
    address
FROM
```

Customers c

WHERE

NOT EXISTS (

SELECT 1

FROM Transactions t

WHERE t.account_id IN (SELECT account_id FROM Accounts WHERE customer_id = c.customer_id)

);

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     customer_id,
    ->     first_name,
    ->     last_name,
    ->     DOB,
    ->     email,
    ->     phone_number,
    ->     address
    -> FROM
    ->     Customers c
    -> WHERE
    ->     NOT EXISTS (
    ->         SELECT 1
    ->         FROM Transactions t
    ->         WHERE t.account_id IN (SELECT account_id FROM Accounts WHERE customer_id = c.customer_id)
    ->     );
Empty set (0.00 sec)

mysql>
```

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

Ans.

SELECT

SUM(balance) AS total_balance_no_transactions

FROM

Accounts a

WHERE

NOT EXISTS (

SELECT 1

FROM Transactions t

WHERE t.account_id = a.account_id

);

Command Prompt - mysql -u root -p

```
    ->     );
Empty set (0.00 sec)

mysql> SELECT
    ->     SUM(balance) AS total_balance_no_transactions
    -> FROM
    ->     Accounts a
    -> WHERE
    ->     NOT EXISTS (
    ->         SELECT 1
    ->         FROM Transactions t
    ->         WHERE t.account_id = a.account_id
    ->     );
+-------------------------------+
| total_balance_no_transactions |
+-------------------------------+
|                          NULL |
+-------------------------------+
1 row in set (0.00 sec)

mysql>
```

6. Retrieve transactions for accounts with the lowest balance.

Ans.

```
SELECT
  t.transaction_id,
  t.account_id,
  t.transaction_type,
  t.amount,
  t.transaction_date
FROM
  Transactions t
JOIN (
  SELECT
    account_id
  FROM
    Accounts
  ORDER BY
    balance ASC
  LIMIT 1
) a ON t.account_id = a.account_id;
```

Command Prompt - mysql -u root -p

```
1 row in set (0.00 sec)

mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> JOIN (
    ->     SELECT
    ->         account_id
    ->     FROM
    ->         Accounts
    ->     ORDER BY
    ->         balance ASC
    ->     LIMIT 1
    -> ) a ON t.account_id = a.account_id;
+----------------+------------+------------------+--------+------------------+
| transaction_id | account_id | transaction_type | amount | transaction_date |
+----------------+------------+------------------+--------+------------------+
|           1009 |        109 | withdrawal       |  50.00 | 2023-12-09       |
+----------------+------------+------------------+--------+------------------+
1 row in set (0.00 sec)

mysql>
```

7. Identify customers who have accounts of multiple types.

Ans.

```
SELECT
   c.customer_id,
   c.first_name,
   c.last_name
FROM
   Customers c
JOIN (
   SELECT
      customer_id
   FROM
      Accounts
   GROUP BY
      customer_id
   HAVING
      COUNT(DISTINCT account_type) > 1
) a ON c.customer_id = a.customer_id;
```

Command Prompt - mysql -u root -p

```
1 row in set (0.00 sec)

mysql> SELECT
    ->     c.customer_id,
    ->     c.first_name,
    ->     c.last_name
    -> FROM
    ->     Customers c
    -> JOIN (
    ->     SELECT
    ->         customer_id
    ->     FROM
    ->         Accounts
    ->     GROUP BY
    ->         customer_id
    ->     HAVING
    ->         COUNT(DISTINCT account_type) > 1
    -> ) a ON c.customer_id = a.customer_id;
Empty set (0.00 sec)

mysql>
```

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

Ans.

SELECT

account_type,

COUNT(*) AS num_accounts,

(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM Accounts)) AS percentage

FROM

Accounts

GROUP BY

account_type;

Command Prompt - mysql -u root -p

```
Empty set (0.00 sec)

mysql> SELECT
    ->     account_type,
    ->     COUNT(*) AS num_accounts,
    ->     (COUNT(*) * 100.0 / (SELECT COUNT(*) FROM Accounts)) AS percentage
    -> FROM
    ->     Accounts
    -> GROUP BY
    ->     account_type;
+--------------+--------------+------------+
| account_type | num_accounts | percentage |
+--------------+--------------+------------+
| savings      |            5 |   50.00000 |
| current      |            5 |   50.00000 |
+--------------+--------------+------------+
2 rows in set (0.02 sec)

mysql>
```

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

Ans.

```
SELECT
    t.transaction_id,
    t.account_id,
    t.transaction_type,
    t.amount,
    t.transaction_date
FROM
    Transactions t
JOIN
    Accounts a ON t.account_id = a.account_id
WHERE
    a.customer_id = [given_customer_id];
```

Command Prompt - mysql -u root -p

```
ERROR 1064 (42000): You have an error in your SQL syntax
mysql> SELECT
    ->     t.transaction_id,
    ->     t.account_id,
    ->     t.transaction_type,
    ->     t.amount,
    ->     t.transaction_date
    -> FROM
    ->     Transactions t
    -> JOIN
    ->     Accounts a ON t.account_id = a.account_id
    -> WHERE
    ->     a.customer_id = 101;
Empty set (0.00 sec)

mysql>
```

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

Ans.

```
SELECT
    account_type,
    (SELECT SUM(balance) FROM Accounts a WHERE a.account_type = t.account_type) AS total_balance
FROM
```

Accounts t

GROUP BY

account_type;

Command Prompt - mysql -u root -p

```
mysql> SELECT
    ->     account_type,
    ->     (SELECT SUM(balance) FROM Accounts a WHERE a.account_type = t.account_type) AS total_balance
    -> FROM
    ->     Accounts t
    -> GROUP BY
    ->     account_type;
+--------------+---------------+
| account_type | total_balance |
+--------------+---------------+
| savings      |      26127.21 |
| current      |      64001.00 |
+--------------+---------------+
2 rows in set (0.06 sec)

mysql>
```