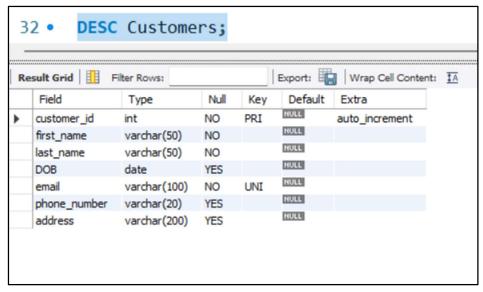
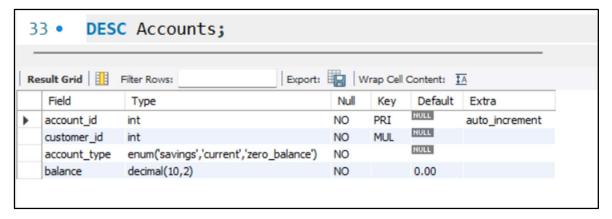
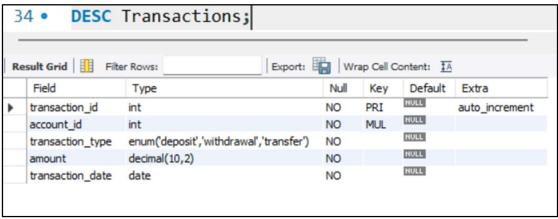
Tasks 1: Database Design:

- 1. Create the database named "HMBank".
- 2. Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema
- 4. Create an ERD
- 5. Create appropriate Primary Key and Foreign Key constraints for referential integrity.
- 6. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
- Customers
- Accounts
- Transactions









```
5 • 

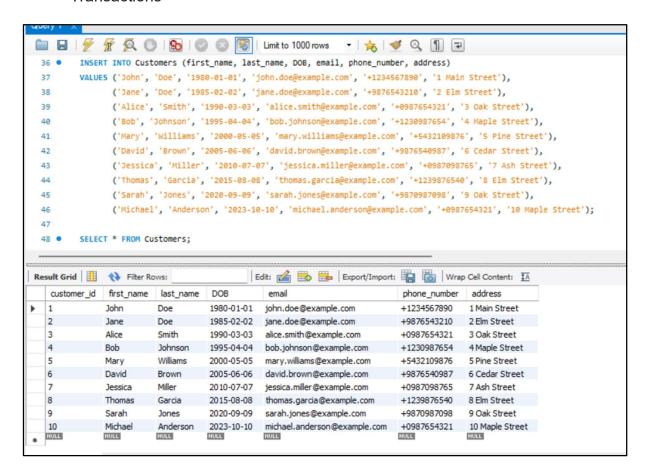
CREATE TABLE Customers (
          customer_id INT PRIMARY KEY AUTO_INCREMENT,
            first_name VARCHAR(50) NOT NULL,
            last_name VARCHAR(50) NOT NULL,
 8
 9
            DOB DATE,
            email VARCHAR(100) NOT NULL UNIQUE,
11
            phone_number VARCHAR(20),
            address VARCHAR(200)
12
        );
13
15 • 

CREATE TABLE Accounts (
          account_id INT PRIMARY KEY AUTO_INCREMENT,
            customer_id INT NOT NULL,
17
            account_type ENUM('savings', 'current', 'zero_balance') NOT NULL,
18
            balance DECIMAL(10,2) NOT NULL DEFAULT 0,
            FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
20
      - );
21
22
23 • 

CREATE TABLE Transactions (
            transaction_id INT PRIMARY KEY AUTO_INCREMENT,
24
            account_id INT NOT NULL,
25
            transaction_type ENUM('deposit', 'withdrawal', 'transfer') NOT NULL,
26
27
            amount DECIMAL(10,2) NOT NULL,
            transaction_date DATE NOT NULL,
28
            FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
29
      );
30
```

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 Accounts (customer_id, account_type, balance)
 51
         VALUES
         (1, 'savings', 1000.00),
 52
         (2, 'current', 500.00),
 53
         (3, 'savings', 2000.00),
 54
         (4, 'current', 750.00),
 55
         (5, 'zero_balance', 0.00),
 56
         (6, 'savings', 3500.00),
 57
         (7, 'current', 1200.00),
 58
         (8, 'savings', 4800.00),
 59
         (9, 'current', 2350.00),
         (10, 'zero_balance', 0.00);
 61
 62 •
         SELECT * FROM Accounts;
 63
                                             Edit: 🚣 🖶 Expor
customer_id account_type
   account_id
                                        balance
                                        1000.00
   1
              1
                          savings
              2
   2
                          current
                                        500.00
   3
              3
                                        2000.00
                          savings
   4
              4
                                       750.00
                          current
   5
              5
                                       0.00
                          zero_balance
              6
   6
                          savings
                                       3600.00
   7
              7
                          current
                                        1200.00
   8
              8
                                       4800.00
                          savings
   9
              9
                                       2350.00
                          current
   10
              10
                          zero_balance
                                       0.00
  NULL
             NULL
                          NULL
                                       NULL
```

```
INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date)
          VALUES
 65
          (1, 'deposit', 100.00, '2023-10-26'),
 66
          (2, 'withdrawal', 50.00, '2023-10-27'),
 67
          (3, 'transfer', 25.00, '2023-10-28'),
 68
          (4, 'deposit', 75.00, '2023-10-29'),
 69
          (5, 'withdrawal', 20.00, '2023-10-30'),
 70
          (6, 'transfer', 15.00, '2023-10-31'),
 71
          (7, 'deposit', 125.00, '2023-11-01'),
 72
          (8, 'withdrawal', 35.00, '2023-11-02'),
 73
          (9, 'transfer', 40.00, '2023-11-03'),
 74
          (10, 'deposit', 150.00, '2023-11-04');
 75
 76
          SELECT * FROM Transactions;
 77 •
 78
               ♦ Filter Rows:
                                                 Edit: 🚄 📆 📙 Export/Import: 📳 👸 Wrap
Result Grid
   transaction_id
                                                         transaction_date
                  account_id
                              transaction_type
                                               amount
                                                        2023-10-26
   1
                  1
                              deposit
                                               100.00
   2
                  2
                              withdrawal
                                               50.00
                                                        2023-10-27
   3
                  3
                              transfer
                                               25.00
                                                        2023-10-28
                  4
   4
                              deposit
                                               75.00
                                                        2023-10-29
   5
                  5
                              withdrawal
                                                        2023-10-30
                                               20.00
   6
                  6
                              transfer
                                               15.00
                                                        2023-10-31
   7
                  7
                              deposit
                                               125.00
                                                        2023-11-01
   8
                  8
                              withdrawal
                                               35.00
                                                        2023-11-02
   9
                  9
                              transfer
                                               40.00
                                                        2023-11-03
   10
                 10
                             deposit
                                               150.00
                                                        2023-11-04
  NULL
                 NULL
                             NULL
                                               NULL
                                                        NULL
```

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



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

```
SELECT t.transaction type, t.amount, t.transaction date, c.first name, c.last name
84
        FROM Transactions t
85
       INNER JOIN Accounts a ON t.account_id = a.account_id
       INNER JOIN Customers c ON a.customer_id = c.customer_id;
Export: Wrap Cell Content: IA
  transaction_type amount transaction_date first_name last_name
  deposit 100.00 2023-10-26 John withdrawal 50.00 2023-10-27 Jane
deposit
                                         Doe
  transfer
               25.00
                     2023-10-28
                                  Alice
                                           Smith
            75.00 2023-10-29 Bob
  deposit
                                         Johnson
              20.00 2023-10-30 Mary Williams
15.00 2023-10-31 David Brown
  transfer
  withdrawal
              125.00 2023-11-01
35.00 2023-11-02
  deposit
                                  Jessica
                                           Miller
  withdrawal
                                          Garcia
                                 Thomas
  transfer
               40.00
                      2023-11-03
                                  Sarah
                                           Jones
  deposit 150.00 2023-11-04 Michael Anderson
```

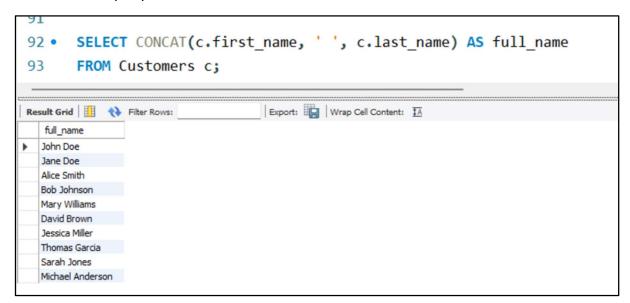
4. Write a SQL query to increase the balance of a specific account by a certain amount.

```
88 • UPDATE Accounts

89 SET balance = balance + 100.00

90 WHERE account_id = 6;
```

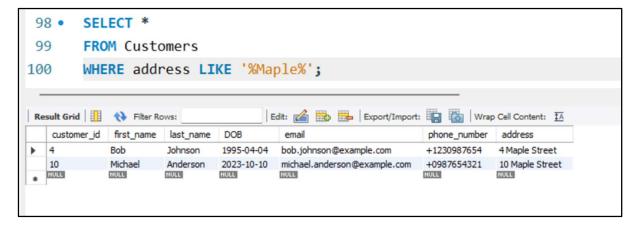
5. Write a SQL query to Combine first and last names of customers as a full name.



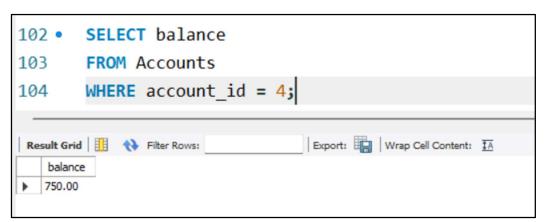
6. Write a SQL query to remove accounts with a balance of zero where the account type is savings.

```
DELETE FROM Accounts
WHERE account_type = 'savings' AND balance = 0;
```

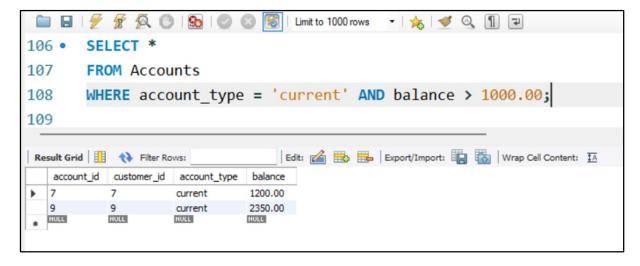
7. Write a SQL query to Find customers living in a specific city.



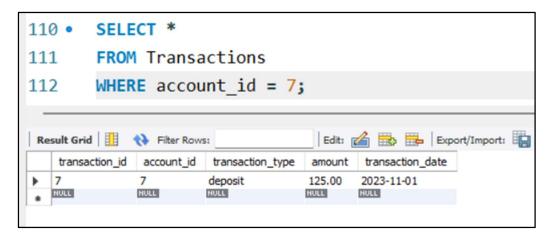
8. Write a SQL query to Get the account balance for a specific account.



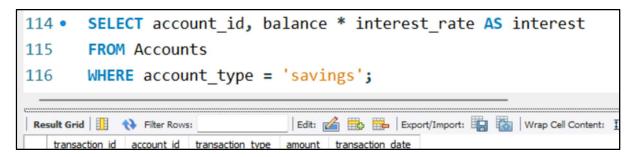
9. Write a SQL query to List all current accounts with a balance greater than \$1,000.



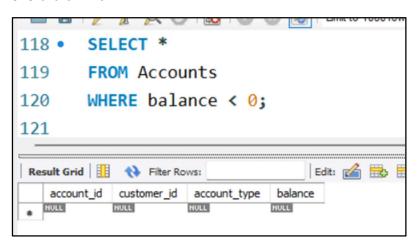
10. Write a SQL query to Retrieve all transactions for a specific account.



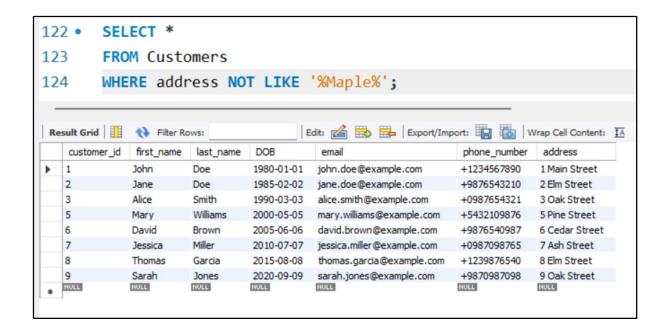
11. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.



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

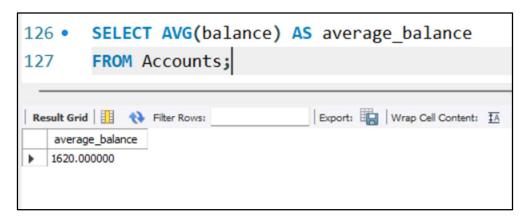


13. Write a SQL query to Find customers not living in a specific city.

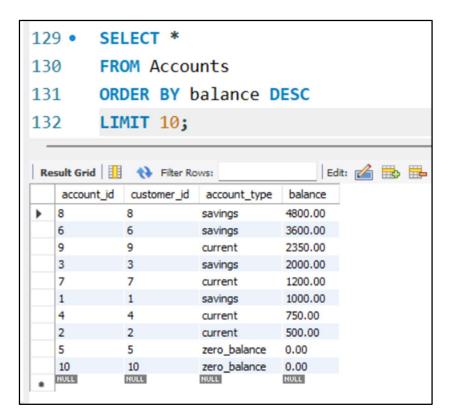


Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:

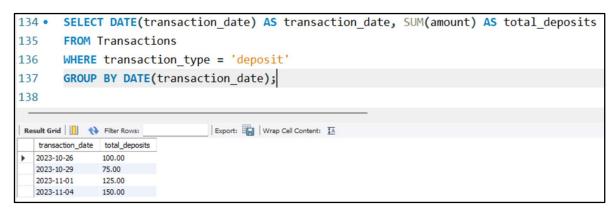
1. Write a SQL query to Find the average account balance for all customers.



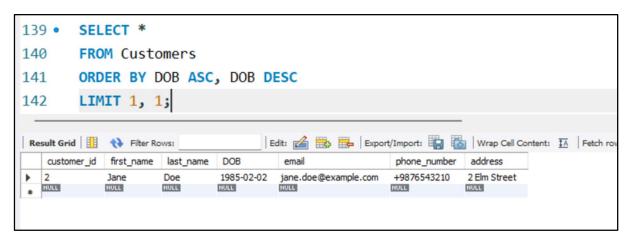
2. Write a SQL query to Retrieve the top 10 highest account balances.



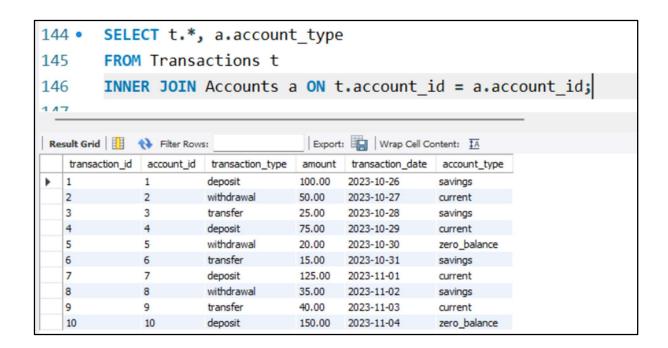
3. Write a SQL query to Calculate Total Deposits for All Customers in specific date.



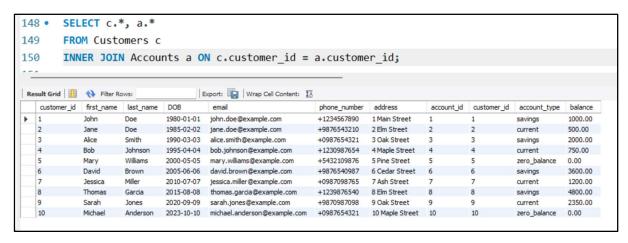
4. Write a SQL query to Find the Oldest and Newest Customers.



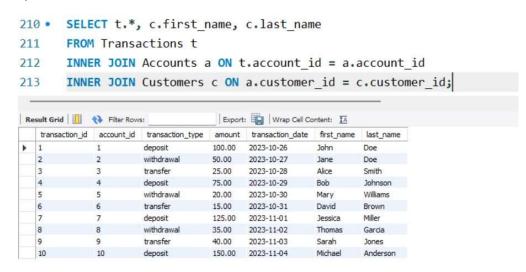
5. Write a SQL query to Retrieve transaction details along with the account type.



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



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



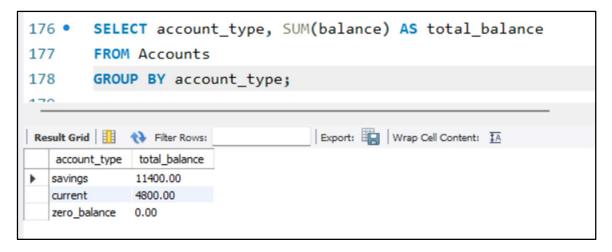
8. Write a SQL query to Identify customers who have more than one account.

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



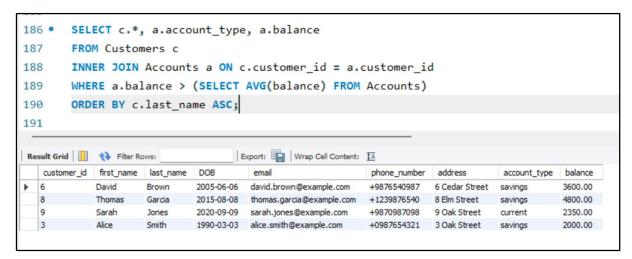
10. Write a SQL query to Calculate the average daily balance for each account over a specified period.

11. Calculate the total balance for each account type.



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

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



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

```
192 •
      SELECT *
193
      FROM Transactions
195
         SELECT account_id, transaction_type, amount, transaction_date
         FROM Transactions
196
197
         GROUP BY account_id, transaction_type, amount, transaction_date
         HAVING COUNT(*) > 1
198
     );
199
200
                           Edit: 🚄 🖶 Export/Import: 📳 🐻 | Wrap Cell Content: 🖽
transaction_id account_id transaction_type amount transaction_date
                 NULL
                          NULL
                               NULL
```

Tasks 4: Subquery and its type:

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

```
SELECT c.*, a.balance
     FROM Customers c
 4
      INNER JOIN Accounts a ON c.customer_id = a.customer_id
 6 ♥ WHERE a.balance = (
           SELECT MAX(balance)
 7
          FROM Accounts
 8
   );
 9
10
                        | Export: | Wrap Cell Content: 1
customer_id first_name last_name DOB
                            email
                                             phone_number address
         Thomas
               Garcia
                      2015-08-08 thomas.garcia@example.com +1239876540 8 Elm Street 4800.00
```

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

```
11 • SELECT AVG(a.balance) AS average_balance

12 FROM Customers c

13 INNER JOIN Accounts a ON c.customer_id = a.customer_id

14 GROUP BY c.customer_id

15 HAVING COUNT(a.account_id) > 1;

16

Result Grid ** Filter Rows: Export: ** Wrap Cell Content: ** Average_balance**
```

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

4. Identify customers who have no recorded transactions.

```
27 • SELECT *
28
     FROM Customers c
29 ♥ WHERE NOT EXISTS (
           SELECT *
30
           FROM Transactions t
31
           WHERE c.customer id = t.account id
32
      );
33
                           | Edit: 🚄 🔜 🖶 | Export/Import: 🏣 🎳 | Wrap Cell Content: 🔣
customer_id first_name last_name DOB email phone_number address
NULL
                NULL
                      NULL
         NULL
                          NULL
                              NULL
                                       NULL
```

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

```
35 • SELECT SUM(balance) AS total_balance

36  FROM Accounts a

37  WHERE NOT EXISTS (

38  SELECT *

40  WHERE a.account_id = t.account_id

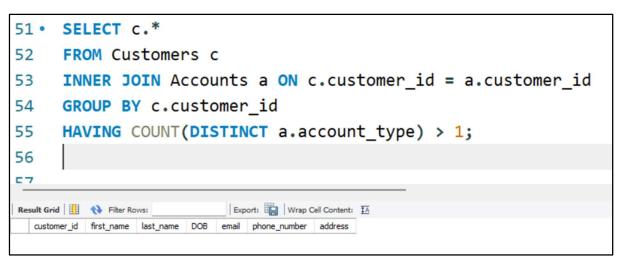
41 );

Result Grid  Fiter Rows: Export: Wrap Cell Content: IA
```

6. Retrieve transactions for accounts with the lowest balance.

```
43 • SELECT t.*
44
      FROM Transactions t
      INNER JOIN Accounts a ON t.account_id = a.account_id
45
46 ♥ WHERE a.balance = (
            SELECT MIN(balance)
47
            FROM Accounts
48
     );
49
50
E1
Result Grid H Tilter Rows:
                             Export: Wrap Cell Content: IA
  transaction_id account_id transaction_type amount transaction_date
5
         5 withdrawal
10 deposit
                            20.00 2023-10-30
  10
                          150.00 2023-11-04
```

7. Identify customers who have accounts of multiple types.



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

```
SELECT account type, COUNT(*) AS total accounts,
57 •
              (COUNT(*) / (SELECT COUNT(*) FROM Accounts)) * 100 AS percentage
58
     FROM Accounts
59
     GROUP BY account type;
60
61
Export: Wrap Cell Content: TA
 account_type total_accounts percentage
                 40.0000
savings
                 40.0000
 zero_balance 2
                 20,0000
```

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

```
62 · SELECT *
      FROM Transactions
64 ♥ WHERE account id IN (
            SELECT account id
65
            FROM Accounts
66
67
            WHERE customer id = 4
      );
68
60
                             | Edit: 🔏 🖶 | Export/Import: 📳 🐻 | Wrap Cell Content: 🏗
transaction_id account_id transaction_type amount transaction_date
• 4
• NULL
                  deposit
                            75.00
                                  2023-10-29
                            NULL
```

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

