**Assignment**

**Banking System**

**Database Tables**

**1. Customers:**

• customer\_id (Primary Key)

• first\_name

• last\_name

• DOB (Date of Birth)

• email

• phone\_number

• address

**2. Accounts:**

• account\_id (Primary Key)

• customer\_id (Foreign Key)

• account\_type (e.g., savings, current, zero\_balance)

• balance

**3. Transactions**:

• transaction\_id (Primary Key)

• account\_id (Foreign Key)

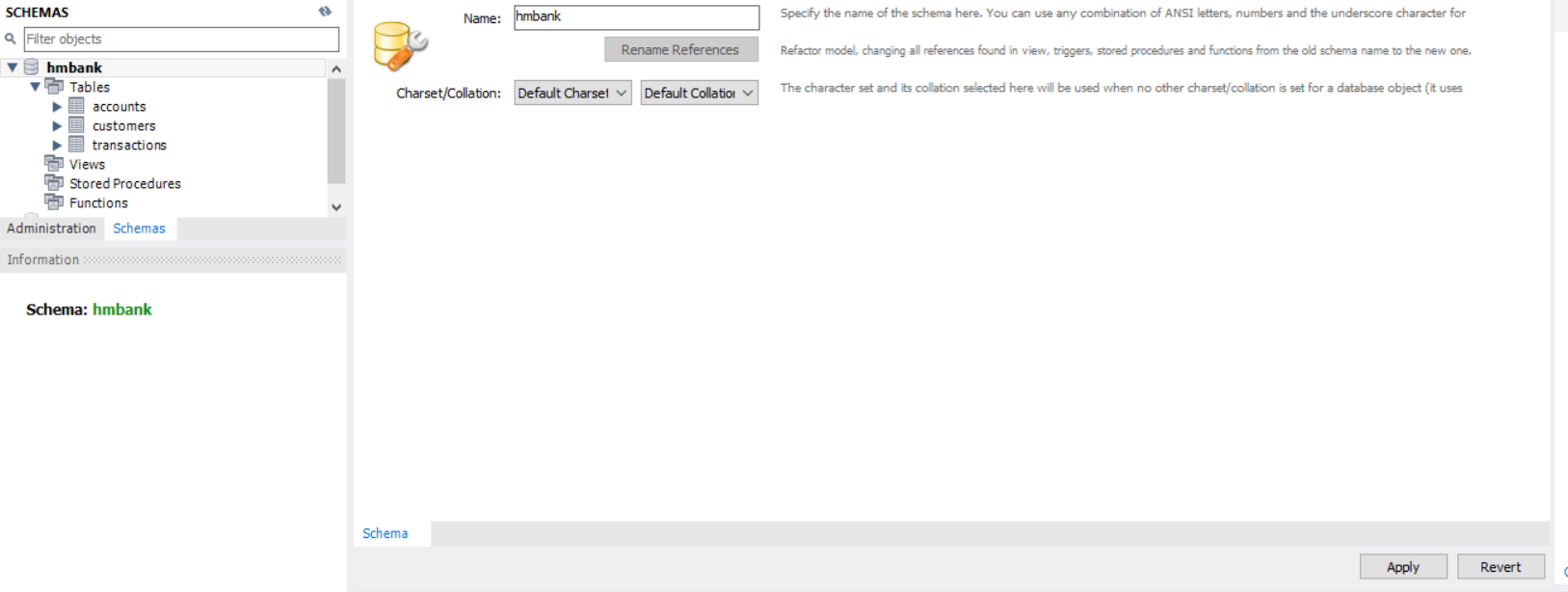
• transaction\_type (e.g., deposit, withdrawal, transfer)

• amount

• transaction\_date

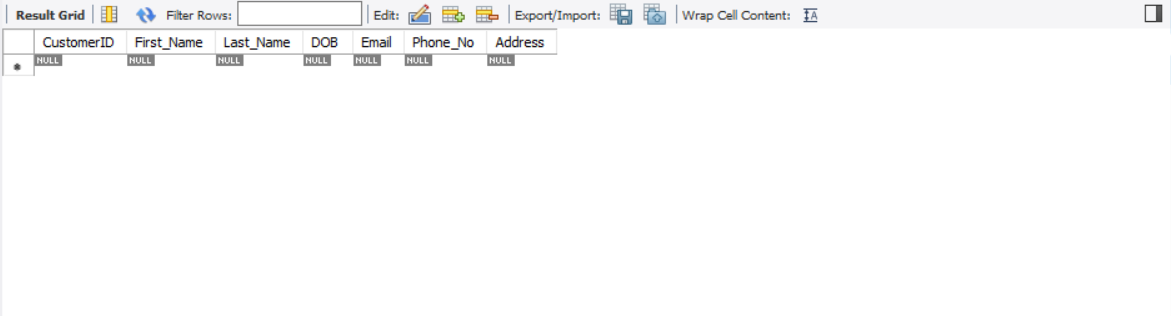
**Tasks 1: Database Design:**

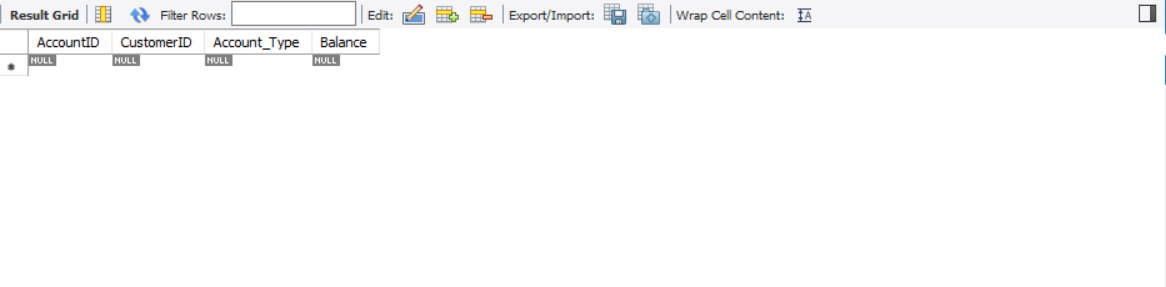
**1. Create the database named "HMBank"**

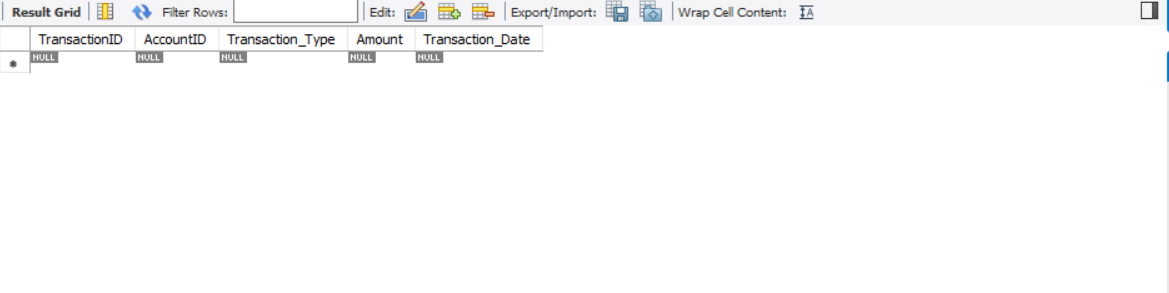
****

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

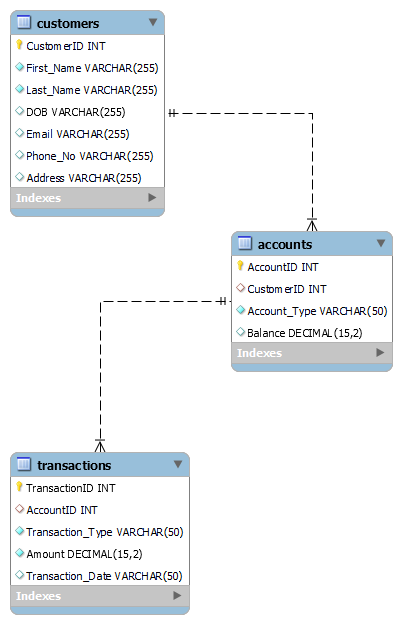
**provided schema.**

****

****

****

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

****

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

Customers (

CustomerID INT PRIMARY KEY,

)

Accounts (

AccountID INT PRIMARY KEY,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

)

Transactions (

TransactionID INT PRIMARY KEY,

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

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

and relationships.

• Customers

• Accounts

• Transactions

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

First\_Name VARCHAR(255) NOT NULL,

Last\_Name VARCHAR(255) NOT NULL,

DOB VARCHAR(255),

Email VARCHAR(255),

Phone\_No VARCHAR(255),

Address VARCHAR(255)

);

CREATE TABLE Accounts (

AccountID INT PRIMARY KEY,

CustomerID INT,

Account\_Type VARCHAR(50) NOT NULL,

Balance DECIMAL(15, 2) DEFAULT 0.0,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Transactions (

TransactionID INT PRIMARY KEY,

AccountID INT,

Transaction\_Type VARCHAR(50) NOT NULL,

Amount DECIMAL(15, 2) NOT NULL,

Transaction\_Date VARCHAR(50) ,

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

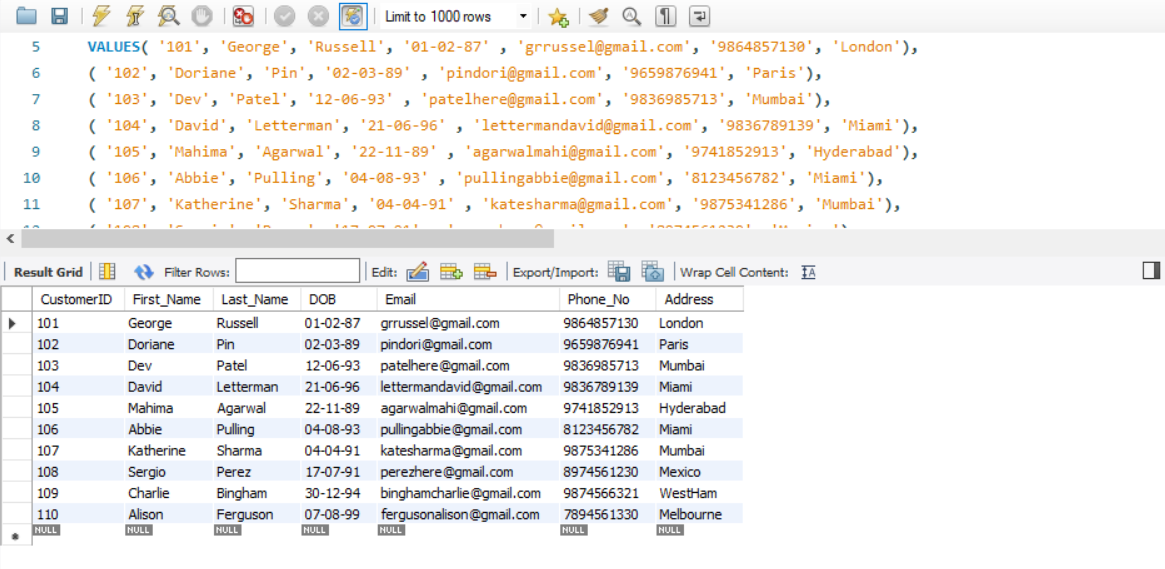
**Tasks 2: Select, Where, Between, AND, LIKE:**

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

**• Customers**

**• Accounts**

**• Transactions**

****

INSERT into Customers (CustomerID, First\_Name, Last\_Name, DOB, Email, Phone\_No, Address)

VALUES( '101', 'George', 'Russell', '01-02-87' , 'grrussel@gmail.com', '9864857130', 'London'),

( '102', 'Doriane', 'Pin', '02-03-89' , 'pindori@gmail.com', '9659876941', 'Paris'),

( '103', 'Dev', 'Patel', '12-06-93' , 'patelhere@gmail.com', '9836985713', 'Mumbai'),

( '104', 'David', 'Letterman', '21-06-96' , 'lettermandavid@gmail.com', '9836789139', 'Miami'),

( '105', 'Mahima', 'Agarwal', '22-11-89' , 'agarwalmahi@gmail.com', '9741852913', 'Hyderabad'),

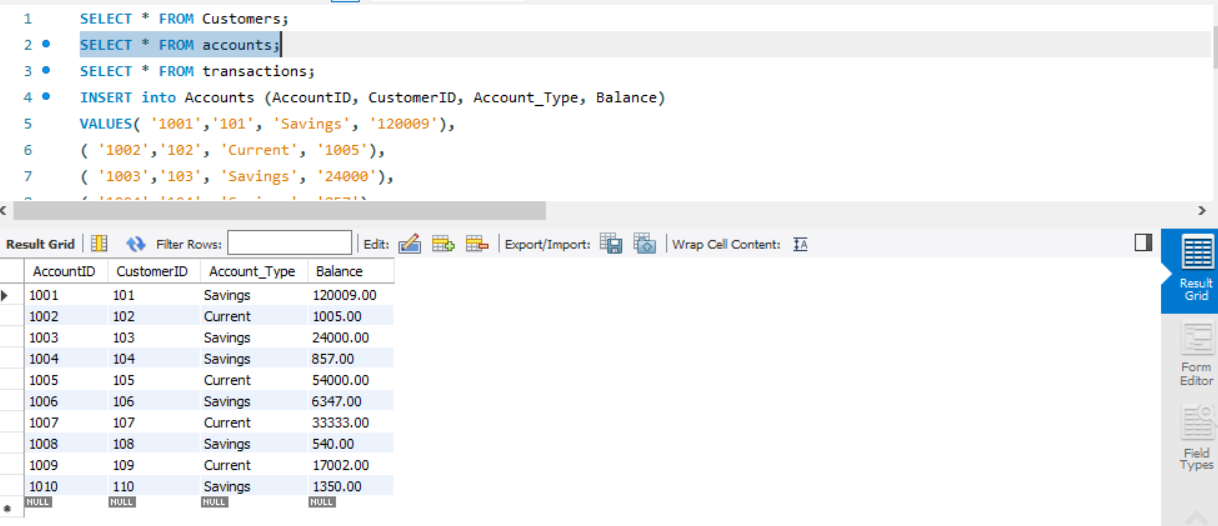
( '106', 'Abbie', 'Pulling', '04-08-93' , 'pullingabbie@gmail.com', '8123456782', 'Miami'),

( '107', 'Katherine', 'Sharma', '04-04-91' , 'katesharma@gmail.com', '9875341286', 'Mumbai'),

( '108', 'Sergio', 'Perez', '17-07-91' , 'perezhere@gmail.com', '8974561230', 'Mexico'),

( '109', 'Charlie', 'Bingham', '30-12-94' , 'binghamcharlie@gmail.com', '9874566321', 'WestHam'),

( '110', 'Alison', 'Ferguson', '07-08-99' , 'fergusonalison@gmail.com', '7894561330', 'Melbourne');

****

INSERT into Accounts (AccountID, CustomerID, Account\_Type, Balance)

VALUES( '1001','101', 'Savings', '120009'),

( '1002','102', 'Current', '1005'),

( '1003','103', 'Savings', '24000'),

( '1004','104', 'Savings', '857'),

( '1005','105', 'Current', '54000'),

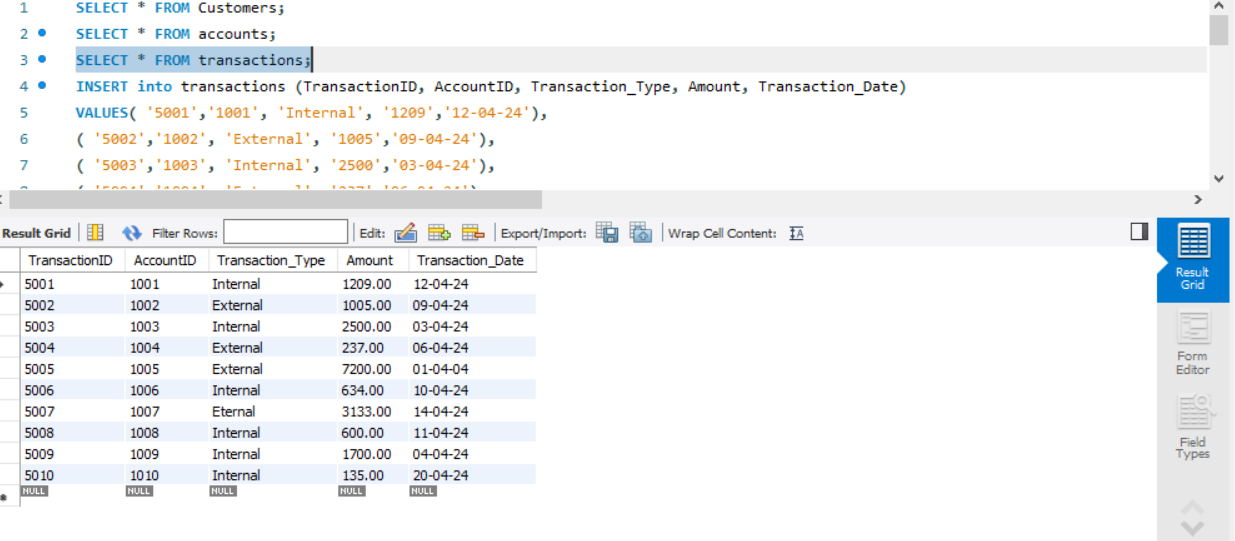
( '1006','106', 'Savings', '6347'),

( '1007','107', 'Current', '33333'),

( '1008','108', 'Savings', '540'),

( '1009','109', 'Current', '17002'),

( '1010','110', 'Savings', '1350');



INSERT into transactions (TransactionID, AccountID, Transaction\_Type, Amount, Transaction\_Date)

VALUES( '5001','1001', 'Internal', '1209','12-04-24'),

( '5002','1002', 'External', '1005','09-04-24'),

( '5003','1003', 'Internal', '2500','03-04-24'),

( '5004','1004', 'External', '237','06-04-24'),

( '5005','1005', 'External', '7200','01-04-2 4'),

( '5006','1006', 'Internal', '634','10-04-24'),

( '5007','1007', 'Eternal', '3133','14-04-24'),

( '5008','1008', 'Internal', '600','11-04-24'),

( '5009','1009', 'Internal', '1700','04-04-24'),

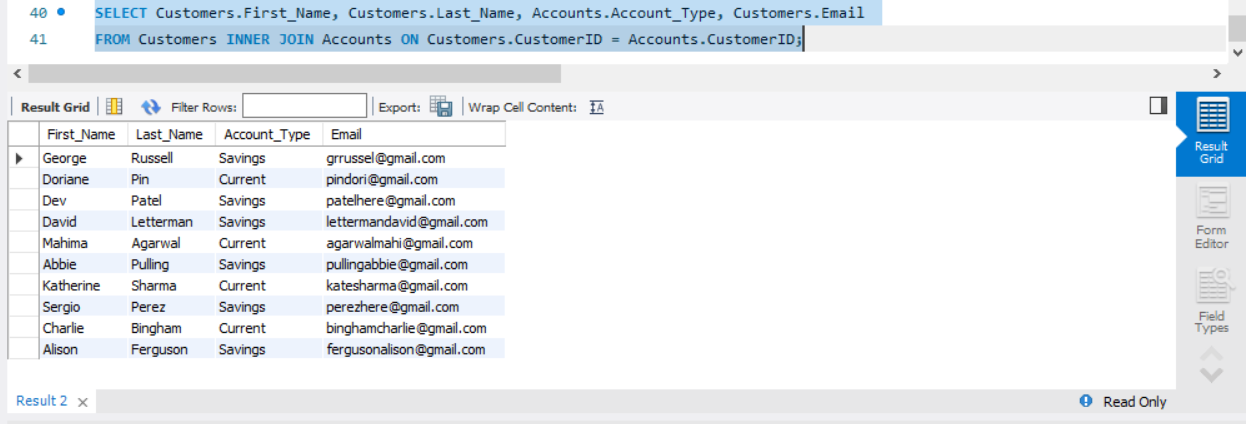
( '5010','1010', 'Internal', '135','20-04-24');

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

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

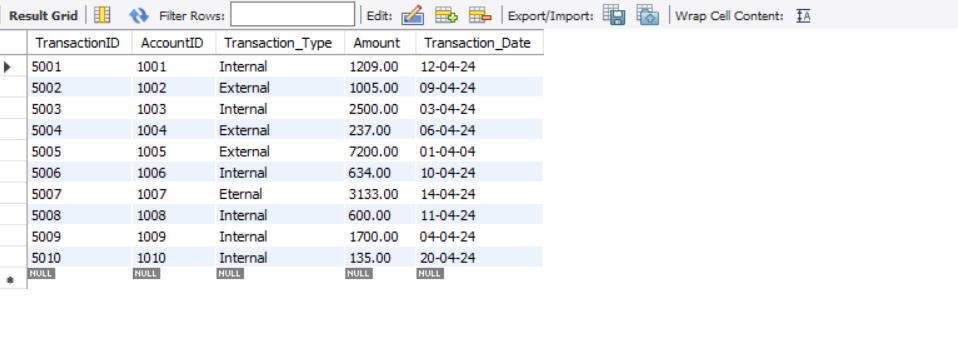
SELECT Customers.First\_Name, Customers.Last\_Name, Accounts.Account\_Type, Customers.Email

FROM Customers INNER JOIN Accounts ON Customers.CustomerID = Accounts.CustomerID;

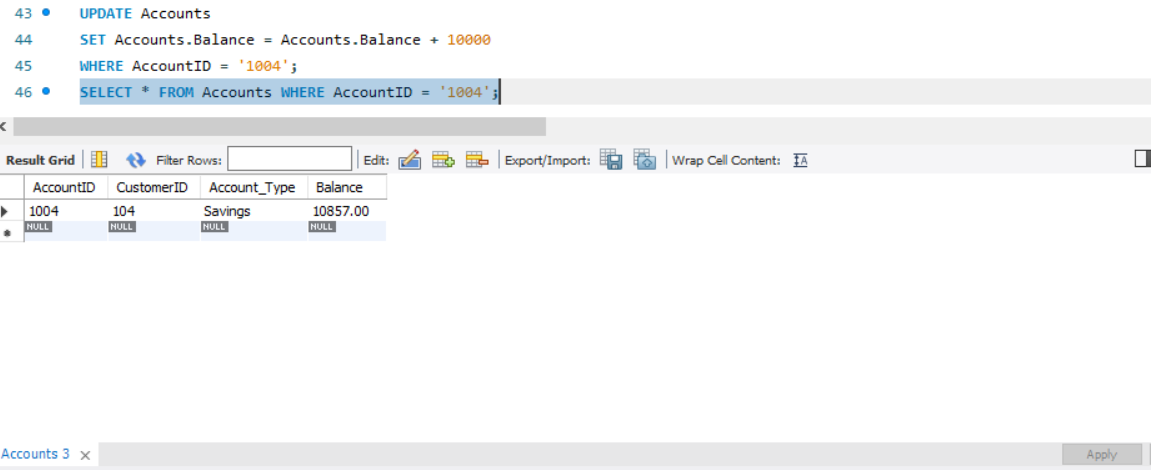
****

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

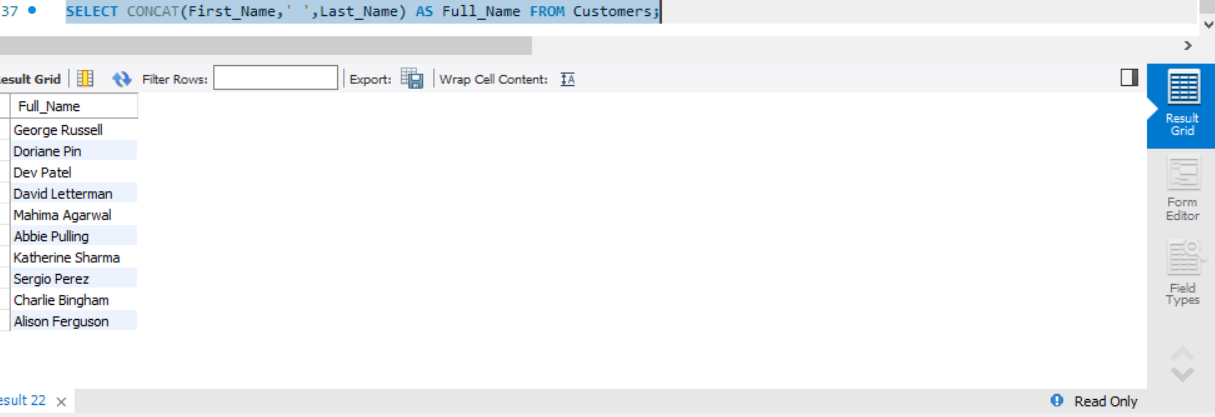
SELECT \* FROM transactions;

****

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

****

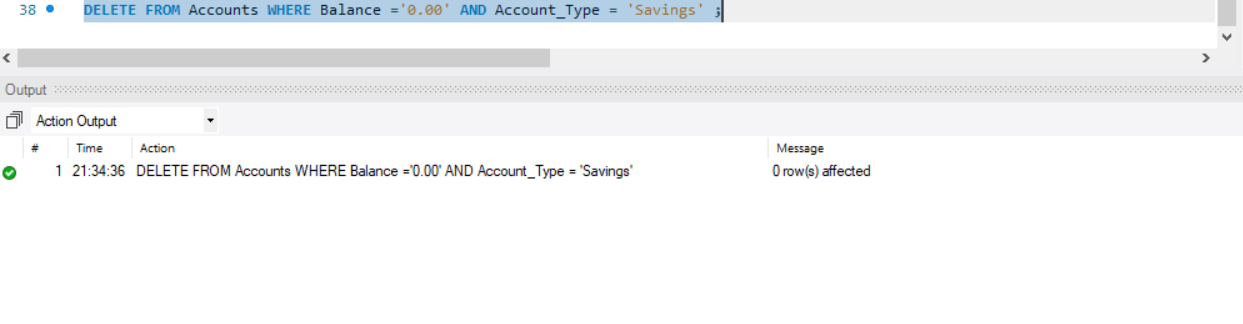
**4. Write a SQL query to Combine first and last names of customers as a full\_name.**

****

SELECT CONCAT(First\_Name,' ',Last\_Name) AS Full\_Name FROM Customers;

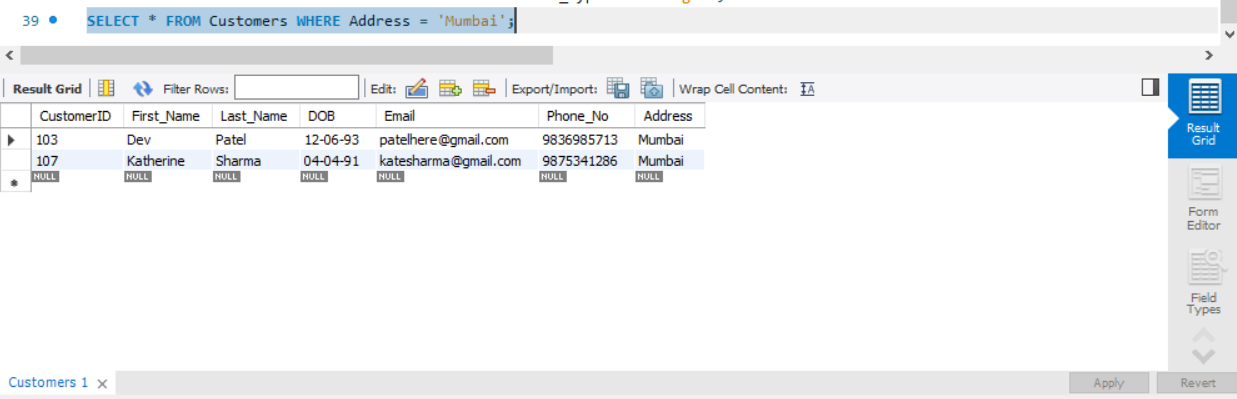
**5. Write a SQL query to remove accounts with a balance of zero where the account**

**type is savings.**

****

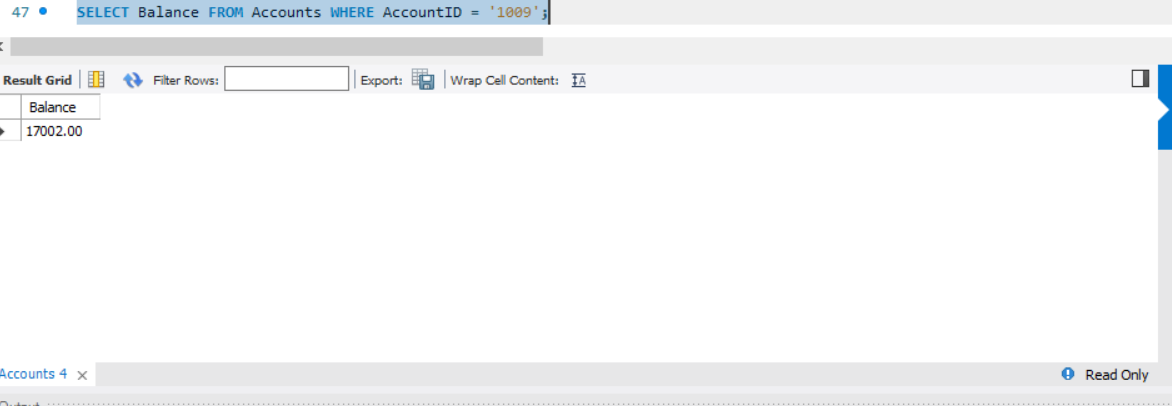
DELETE FROM Accounts WHERE Balance ='0.00' AND Account\_Type = 'Savings' ;

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

****

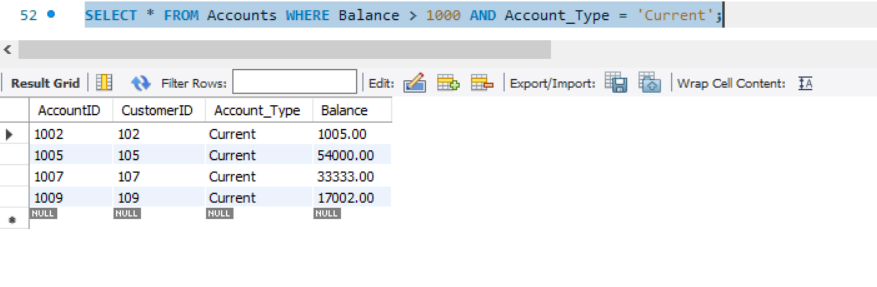
SELECT \* FROM Customers WHERE Address = 'Mumbai';

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

****

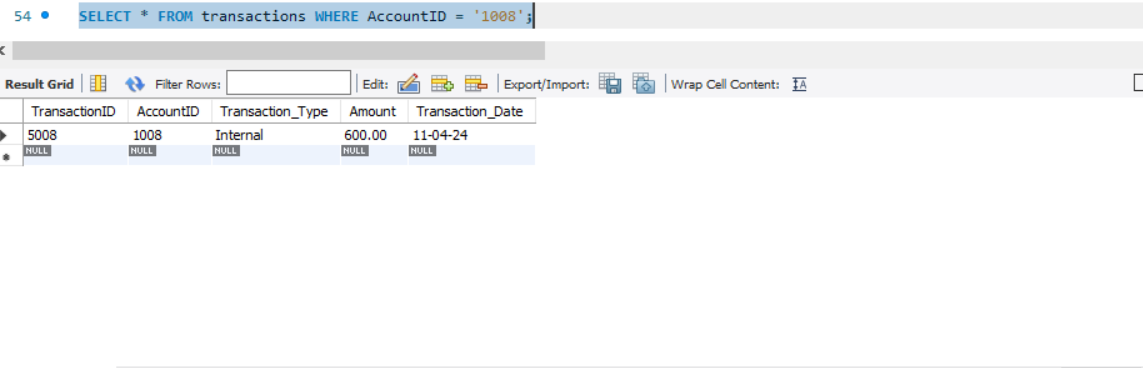
SELECT Balance FROM Accounts WHERE AccountID = '1009';

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



SELECT \* FROM Accounts WHERE Balance > 1000 AND Account\_Type = 'Current';

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



SELECT \* FROM transactions WHERE AccountID = '1008';

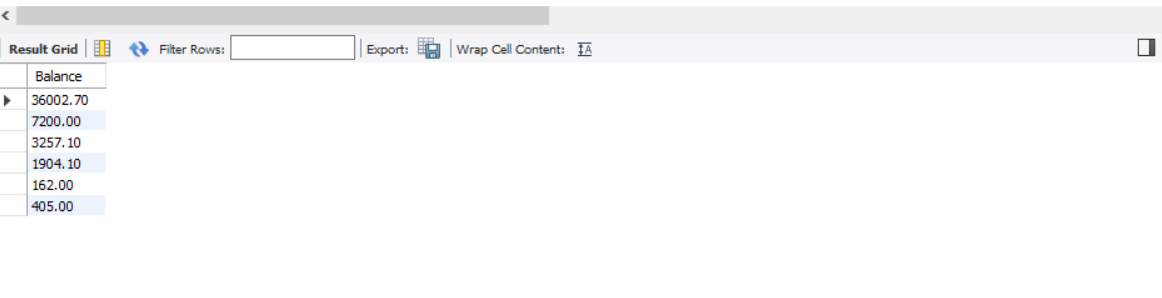
**10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.**

UPDATE Accounts

SET Accounts.Balance = (Accounts.Balance \* 10 \*3)/100

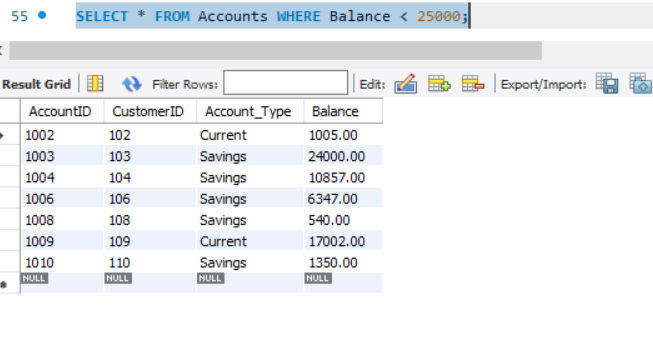
WHERE Account\_Type = 'Savings';

SELECT Balance FROM Accounts WHERE Account\_Type = 'Savings';



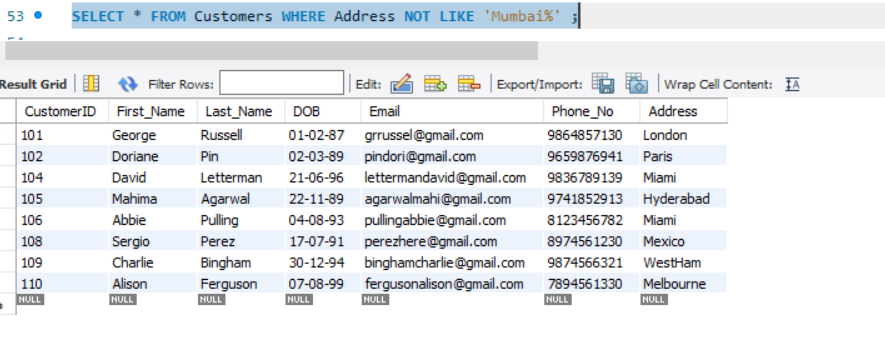
**11. Write a SQL query to Identify accounts where the balance is less than a specified**

**overdraft limit.**

****

SELECT \* FROM Accounts WHERE Balance < 25000;

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

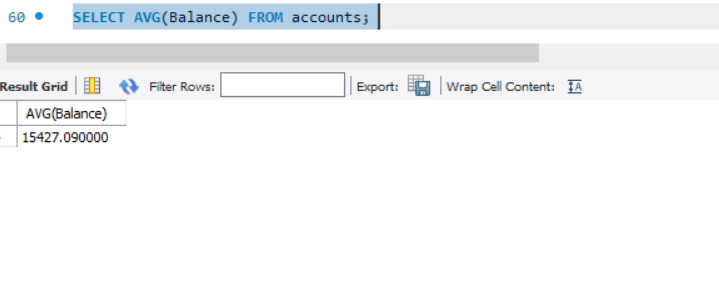
****

SELECT \* FROM Customers WHERE Address NOT LIKE 'Mumbai%' ;

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

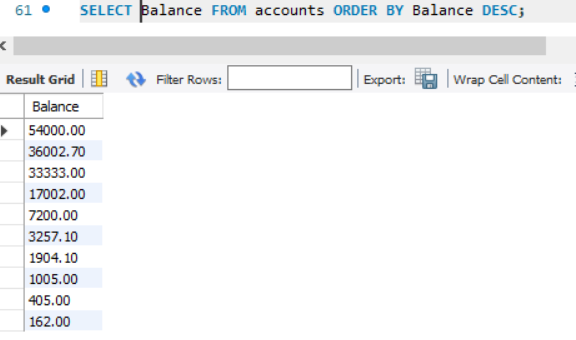
**1. Write a SQL query to Find the average account balance for all customers.**

SELECT AVG(Balance) FROM accounts;

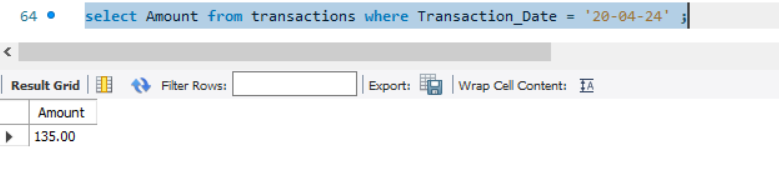
****

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

SELECT Balance FROM accounts ORDER BY Balance DESC limit 10;

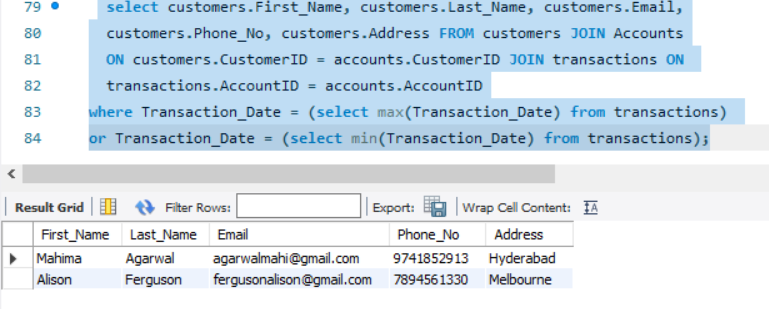
****

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

****

select Amount from transactions where Transaction\_Date = '20-04-24' ;

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

****

select customers.First\_Name, customers.Last\_Name, customers.Email,

customers.Phone\_No, customers.Address FROM customers JOIN Accounts

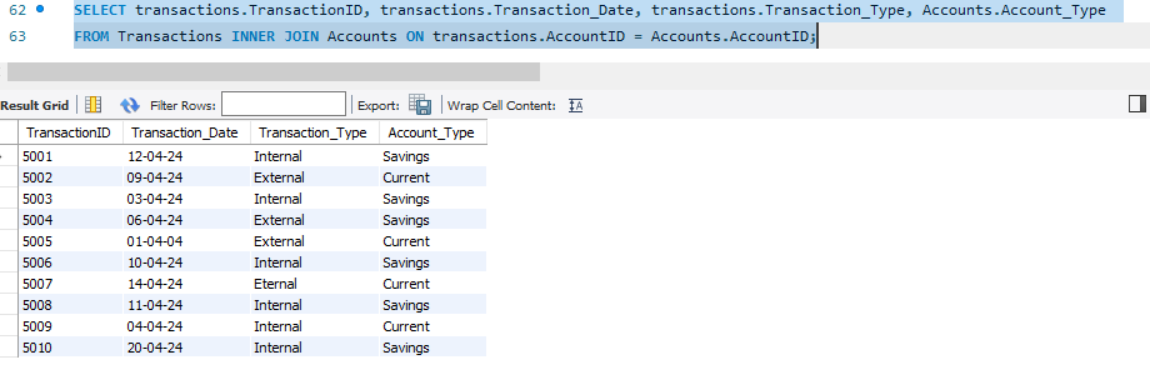
ON customers.CustomerID = accounts.CustomerID JOIN transactions ON

transactions.AccountID = accounts.AccountID

where Transaction\_Date = (select max(Transaction\_Date) from transactions)

or Transaction\_Date = (select min(Transaction\_Date) from transactions);

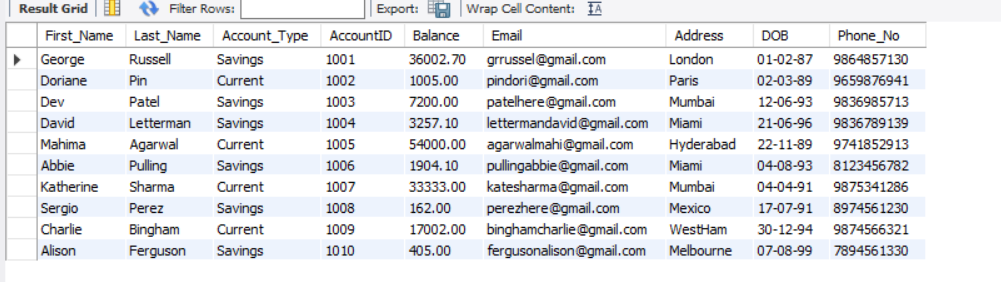
**5. Write a SQL query to Retrieve transaction details along with the account type.**

****

SELECT transactions.TransactionID, transactions.Transaction\_Date, transactions.Transaction\_Type, Accounts.Account\_Type

FROM Transactions INNER JOIN Accounts ON transactions.AccountID = Accounts.AccountID;

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

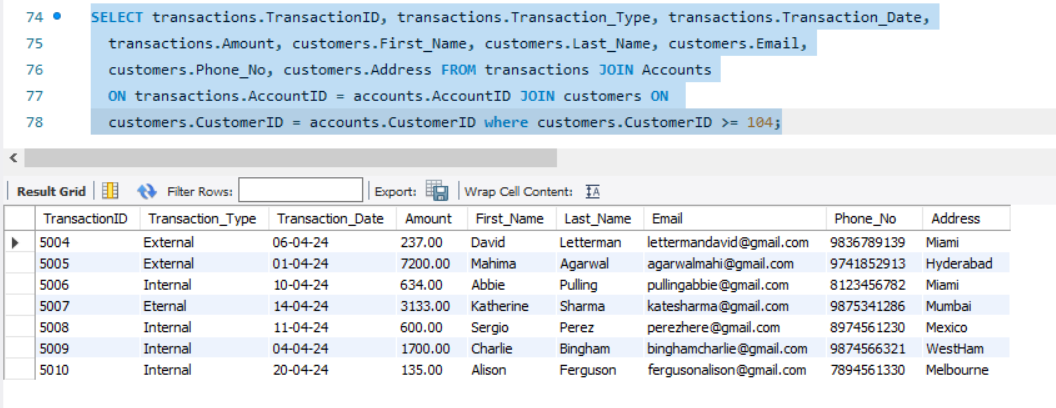
****

SELECT Customers.First\_Name, Customers.Last\_Name, Accounts.Account\_Type, accounts.AccountID, Accounts.Balance, Customers.Email,customers.Address, customers.DOB, customers.Phone\_No

FROM Customers INNER JOIN Accounts ON Customers.CustomerID = Accounts.CustomerID;

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

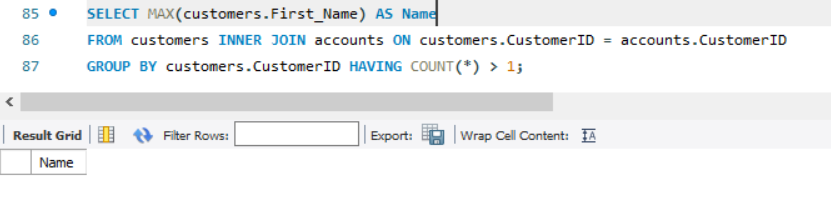
**specific account.**

****

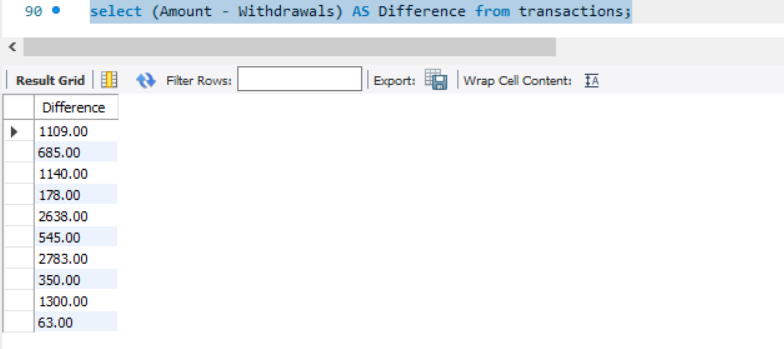
SELECT transactions.TransactionID, transactions.Transaction\_Type, transactions.Transaction\_Date,transactions.Amount,customers.First\_Name,customers.Last\_Name, customers.Email, customers.Phone\_No, customers.Address FROM transactions JOIN Accounts ON transactions.AccountID = accounts.AccountID JOIN customers ON

customers.CustomerID = accounts.CustomerID where customers.CustomerID >= 104;

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

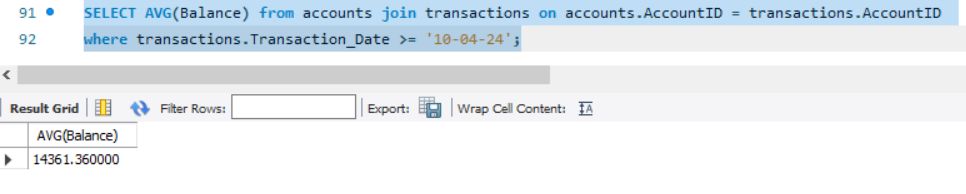
****

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

****

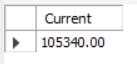
select (Amount - Withdrawals) AS Difference from transactions;

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

****

SELECT AVG(Balance) from accounts join transactions on accounts.AccountID = transactions.AccountID where transactions.Transaction\_Date >= '10-04-24';

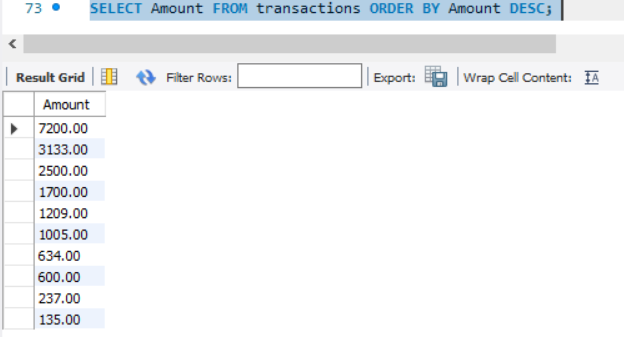
**11. Calculate the total balance for each account type.**

****

SELECT sum(Balance) AS Savings FROM accounts WHERE Account\_Type = 'Savings';

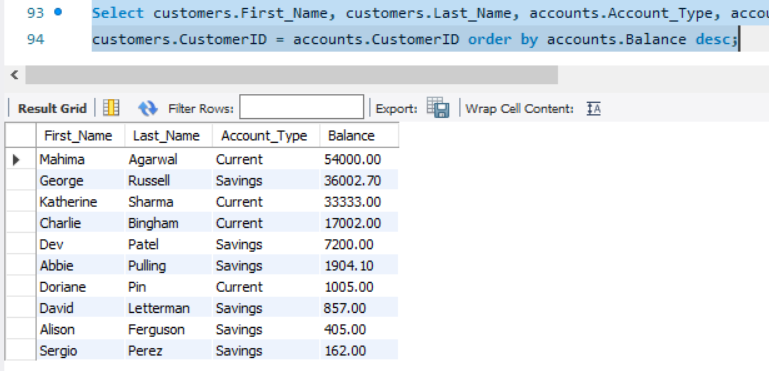
SELECT sum(Balance) AS Current FROM accounts WHERE Account\_Type = 'Current';

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



SELECT Amount FROM transactions ORDER BY Amount DESC;

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

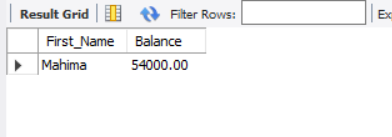
****

Select customers.First\_Name, customers.Last\_Name, accounts.Account\_Type, accounts.Balance from customers Join accounts on

customers.CustomerID = accounts.CustomerID order by accounts.Balance desc;

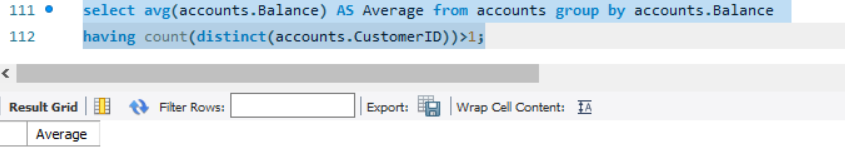
**Tasks 4: Subquery and its type:**

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

****

SELECT customers.First\_Name, accounts.Balance FROM customers Join accounts on customers.CustomerID = accounts.CustomerID where balance = (select MAX(Balance) from accounts) limit 1 ;

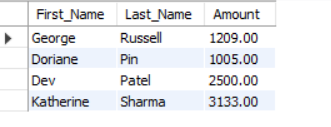
**2. Calculate the average account balance for customers who have more than one account.**

****

select avg(accounts.Balance) AS Average from accounts group by accounts.Balance

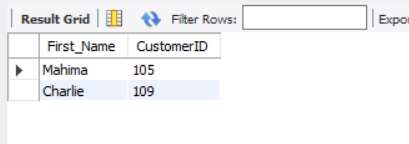
having count(distinct(accounts.CustomerID))>1;

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

****

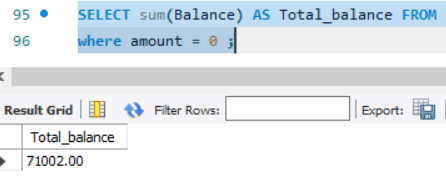
SELECT customers.First\_Name, customers.Last\_Name, transactions.Amount FROM customers Join accounts on customers.CustomerID = accounts.CustomerID join transactions on accounts.AccountID = transactions.AccountID where amount >= 1000 ;

**4. Identify customers who have no recorded transactions.**

****

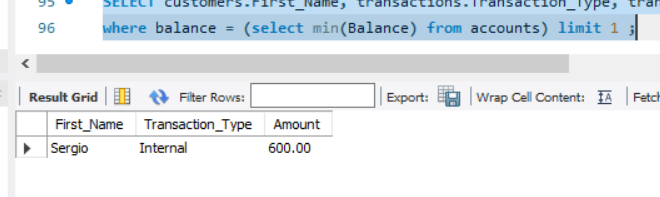
SELECT customers.First\_Name, customers.CustomerID FROM customers Join accounts on customers.CustomerID = accounts.CustomerID join transactions on accounts.AccountID = transactions.AccountID where amount = 0 **;**

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

****

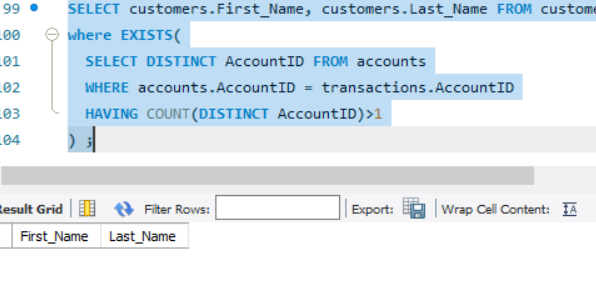
SELECT sum(Balance) AS Total\_balance FROM customers Join accounts on customers.CustomerID = accounts.CustomerID join transactions on accounts.AccountID = transactions.AccountID where amount = 0 ;

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

****

SELECT customers.First\_Name, transactions.Transaction\_Type, transactions.Amount FROM customers Join accounts on customers.CustomerID = accounts.CustomerID join transactions on accounts.AccountID = transactions.AccountID where balance = (select min(Balance) from accounts) limit 1 ;

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

****

SELECT customers.First\_Name, customers.Last\_Name FROM customers Join accounts on customers.CustomerID = accounts.CustomerID join transactions on accounts.AccountID = transactions.AccountID where EXISTS( SELECT DISTINCT AccountID FROM accounts WHERE accounts.AccountID = transactions.AccountID HAVING COUNT(DISTINCT AccountID)>1 ) **;**

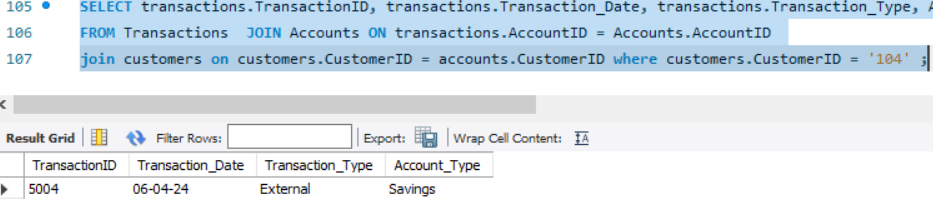
**8. Calculate the percentage of each account type out of the total number of accounts.**

****

select (((select count(Account\_Type) from accounts where Account\_Type = 'Savings') / (select count(accounts.Account\_Type) from accounts)) \* 100) AS savings,

(((select count(Account\_Type) from accounts where Account\_Type = 'Current') / (select count(accounts.Account\_Type) from accounts)) \* 100) AS current from accounts limit 1 ;

**9. Retrieve all transactions for a customer with a given customer\_id.**

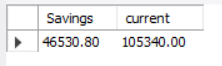
****

SELECT transactions.TransactionID, transactions.Transaction\_Date, transactions.Transaction\_Type, Accounts.Account\_Type

FROM Transactions JOIN Accounts ON transactions.AccountID = Accounts.AccountID

join customers on customers.CustomerID = accounts.CustomerID where customers.CustomerID = '104' ;

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

****

select SUM(accounts.Balance) as Savings ,(select SUM(accounts.Balance) from accounts where Account\_Type = 'Current') as current from accounts where Account\_Type = 'Savings' ;