**mysql> create database HMBank; \\ Created a Database named HMBank**

Query OK, 1 row affected (0.01 sec)

mysql> show databses;

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 'databses' at line 1

mysql> show databases;

+--------------------+

**| Database |**

**+--------------------+**

**| ems |**

**| hmbank |**

**| information\_schema |**

**| mydata |**

**| mysql |**

**| performance\_schema |**

**| sakila |**

**| scoe |**

**| scoe1 |**

**| scoe2 |**

**| sys |**

**| world |**

**+--------------------+**

12 rows in set (0.01 sec)

mysql> use hmbank;

Database changed

**// Create Table Customers**

**--------------------------------------------------------------------------**

CREATE TABLE Customers (customer\_id INTEGER,

-> first\_name VARCHAR(10),

-> last\_name VARCHAR(20),

-> DOB DATE,

-> email VARCHAR(20),

-> phone\_no VARCHAR(10),

-> CONSTRAINT customer\_cid\_pk Primary key (customer\_id)

-> );

Query OK, 0 rows affected (0.04 sec)

ALTER TABLE customers MODIFY email VARCHAR(20) UNIQUE;

Query OK, 0 rows affected (0.03 sec)

Records: 0 Duplicates: 0 Warnings: 0

ALTER TABLE customers MODIFY phone\_no VARCHAR(20) UNIQUE;

Query OK, 0 rows affected (0.01 sec)

Records: 0 Duplicates: 0 Warnings: 0

mysql> desc customers;

**+-------------+-------------+------+-----+---------+-------+**

**| Field | Type | Null | Key | Default | Extra |**

**+-------------+-------------+------+-----+---------+-------+**

**| customer\_id | int | NO | PRI | NULL | |**

**| first\_name | varchar(10) | YES | | NULL | |**

**| last\_name | varchar(20) | YES | | NULL | |**

**| DOB | date | YES | | NULL | |**

**| email | varchar(20) | YES | UNI | NULL | |**

**| phone\_no | varchar(20) | YES | UNI | NULL | |**

**+-------------+-------------+------+-----+---------+-------+**

**6 rows in set (0.00 sec)**

**Created table Accounts**

**-----------------------------------------------------------**

mysql> CREATE TABLE Accounts(

-> account\_id INTEGER,

-> customer\_id INTEGER,

-> account\_type VARCHAR(20),

-> balance INTEGER,

-> CONSTRAINT Accounts\_accid\_pk PRIMARY KEY (account\_id),

-> CONSTRAINT Accounts\_cid\_fk FOREIGN KEY(customer\_id) REFERENCES customers(customer\_id)

-> );

Query OK, 0 rows affected (0.07 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(20) | YES | | NULL | |**

**| balance | int | YES | | NULL | |**

**+--------------+-------------+------+-----+---------+-------+**

4 rows in set (0.00 sec)

**Created Table Transactions**

**--------------------------------------------------**

mysql> CREATE TABLE Transactions(

-> Transaction\_id INTEGER,

-> account\_id INTEGER,

-> transaction\_type VARCHAR(20),

-> amount INTEGER,

-> transcation\_date DATE,

-> CONSTRAINT Transaction\_Transid\_pk PRIMARY KEY(Transaction\_id),

-> CONSTRAINT Transaction\_accid\_fk FOREIGN KEY(account\_id) REFERENCES accounts(account\_id)

-> );

Query OK, 0 rows affected (0.04 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(20) | YES | | NULL | |**

**| amount | int | YES | | NULL | |**

**| transcation\_date | date | YES | | NULL | |**

**+------------------+-------------+------+-----+---------+-------+**

**Entering Values in Customers Table**

**--------------------------------------------------------------------------**

INSERT INTO Customers VALUES (1, 'John', 'Doe', '1990-05-12', 'john.d@gmail.com', '9876543210');

INSERT INTO Customers VALUES (2, 'Jane', 'Smith', '1985-08-23', 'jane.s@gmail.com', '8765432109');

INSERT INTO Customers VALUES (3, 'Robert', 'Johnson', '1992-11-11', 'robert.j@gmail.com', '7654321098');

INSERT INTO Customers VALUES (4, 'Emily', 'Davis', '1988-01-25', 'emily.d@gmail.com', '6543210987');

INSERT INTO Customers VALUES (5, 'Michael', 'Brown', '1995-09-15', 'michael.b@gmail.com', '5432109876');

INSERT INTO Customers VALUES (6, 'Sarah', 'Wilson', '1993-07-07', 'sarah.w@gmail.com', '4321098765');

INSERT INTO Customers VALUES (7, 'David', 'Miller', '1991-12-30', 'david.m@gmail.com', '3210987654');

INSERT INTO Customers VALUES (8, 'Laura', 'Garcia', '1987-03-19', 'laura.g@gmail.com', '2109876543');

INSERT INTO Customers VALUES (9, 'Daniel', 'Martinez', '1989-06-09', 'daniel.m@gmail.com', '1098765432');

INSERT INTO Customers VALUES (10, 'Sophia', 'Taylor', '1994-04-17', 'sophia.t@gmail.com', '0987654321');

mysql> select \* from customers;

**+-------------+------------+-----------+------------+---------------------+------------+**

**| customer\_id | first\_name | last\_name | DOB | email | phone\_no |**

**+-------------+------------+-----------+------------+---------------------+------------+**

**| 1 | John | Doe | 1990-05-12 | john.d@gmail.com | 9876543210 |**

**| 2 | Jane | Smith | 1985-08-23 | jane.s@gmail.com | 8765432109 |**

**| 3 | Robert | Johnson | 1992-11-11 | robert.j@gmail.com | 7654321098 |**

**| 4 | Emily | Davis | 1988-01-25 | emily.d@gmail.com | 6543210987 |**

**| 5 | Michael | Brown | 1995-09-15 | michael.b@gmail.com | 5432109876 |**

**| 6 | Sarah | Wilson | 1993-07-07 | sarah.w@gmail.com | 4321098765 |**

**| 7 | David | Miller | 1991-12-30 | david.m@gmail.com | 3210987654 |**

**| 8 | Laura | Garcia | 1987-03-19 | laura.g@gmail.com | 2109876543 |**

**| 9 | Daniel | Martinez | 1989-06-09 | daniel.m@gmail.com | 1098765432 |**

**| 10 | Sophia | Taylor | 1994-04-17 | sophia.t@gmail.com | 0987654321 |**

**+-------------+------------+-----------+------------+---------------------+------------+**

**10 rows in set (0.00 sec)**

**Entering Data in Accounts table :**

---------------------------------------------------------------------------

INSERT INTO Accounts VALUES (1001, 1, 'Savings', 5000);

INSERT INTO Accounts VALUES (1002, 2, 'Current', 30000);

INSERT INTO Accounts VALUES (1003, 3, 'Savings', 12000);

INSERT INTO Accounts VALUES (1004, 4, 'Savings', 15000);

INSERT INTO Accounts VALUES (1005, 5, 'Current', 40000);

INSERT INTO Accounts VALUES (1006, 6, 'Savings', 8000);

INSERT INTO Accounts VALUES (1007, 7, 'Current', 25000);

INSERT INTO Accounts VALUES (1008, 8, 'Savings', 9000);

INSERT INTO Accounts VALUES (1009, 9, 'Current', 20000);

INSERT INTO Accounts VALUES (1010, 10, 'Savings', 10000);

mysql> select \* from accounts;

**+------------+-------------+--------------+---------+**

**| account\_id | customer\_id | account\_type | balance |**

**+------------+-------------+--------------+---------+**

**| 1001 | 1 | Savings | 5000 |**

**| 1002 | 2 | Current | 30000 |**

**| 1003 | 3 | Savings | 12000 |**

**| 1004 | 4 | Savings | 15000 |**

**| 1005 | 5 | Current | 25000 |**

**| 1006 | 6 | Savings | 8000 |**

**| 1007 | 7 | Current | 40000 |**

**| 1008 | 8 | Savings | 9000 |**

**| 1009 | 9 | Current | 20000 |**

**| 1010 | 10 | Savings | 10000 |**

**+------------+-------------+--------------+---------+**

10 rows in set (0.00 sec)

**Entering Data in Transactions table**

**-----------------------------------------------------------------------------**

INSERT INTO Transactions VALUES (1, 1001, 'Deposit', 2000, '2024-09-01');

INSERT INTO Transactions VALUES (2, 1001, 'Withdrawal', 1000, '2024-09-05');

INSERT INTO Transactions VALUES (3, 1002, 'Deposit', 5000, '2024-09-10');

INSERT INTO Transactions VALUES (4, 1003, 'Withdrawal', 3000, '2024-09-15');

INSERT INTO Transactions VALUES (5, 1004, 'Deposit', 7000, '2024-09-20');

INSERT INTO Transactions VALUES (6, 1005, 'Withdrawal', 10000, '2024-09-22');

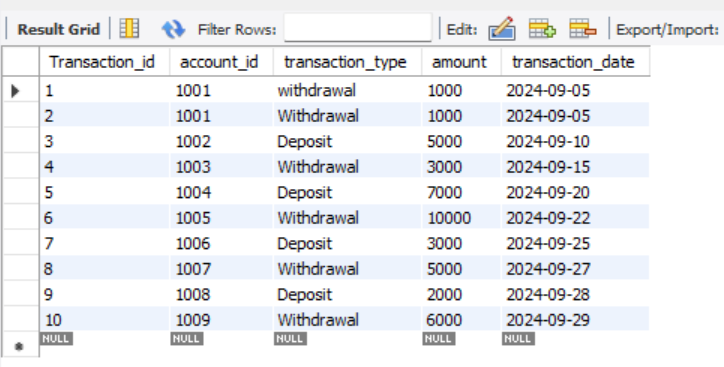
INSERT INTO Transactions VALUES (7, 1006, 'Deposit', 3000, '2024-09-25');

INSERT INTO Transactions VALUES (8, 1007, 'Withdrawal', 5000, '2024-09-27');

INSERT INTO Transactions VALUES (9, 1008, 'Deposit', 2000, '2024-09-28');

INSERT INTO Transactions VALUES (10, 1009, 'Withdrawal', 6000, '2024-09-29');

SELECT \* FROM Transactions; --- (The data was update later on )

****

------------------------------------------------------------------------

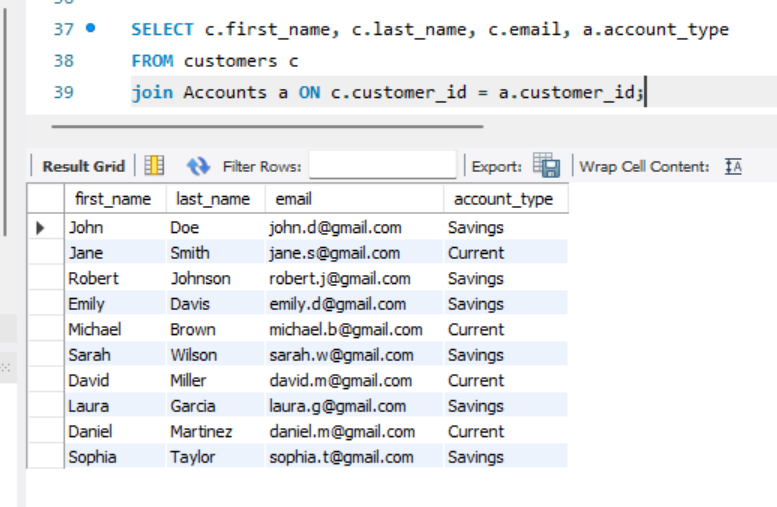
**Write SQL queries for the following tasks:**

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

SELECT c.first\_name, c.last\_name, c.email, a.account\_type

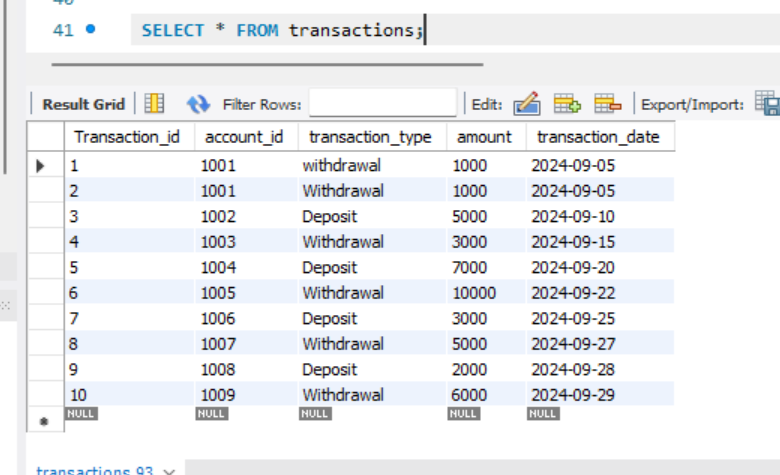
FROM customers c

join Accounts a ON c.customer\_id = a.customer\_id;



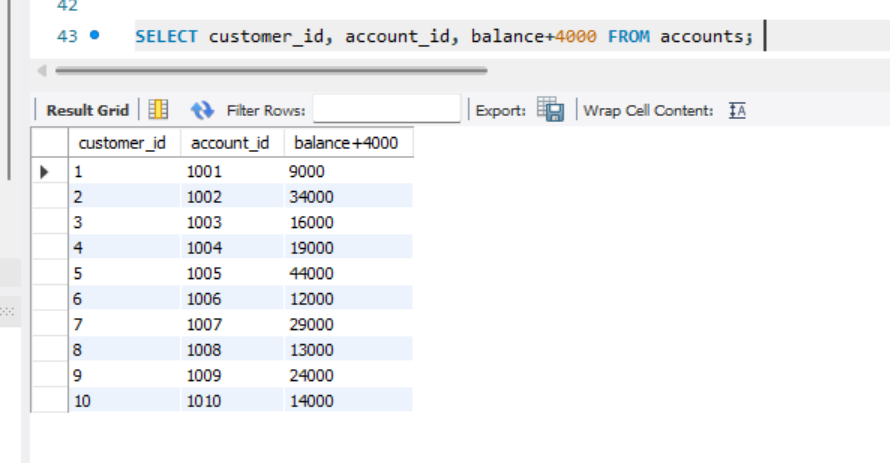
**Write a SQL query to list all transaction corresponding customer.**

SELECT \* FROM transactions;



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

SELECT customer\_id, account\_id, balance+4000 FROM accounts;



**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;



**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; -- there are no zero balance accounts

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

ALTER TABLE customers ADD ( city VARCHAR(20) );

select \* from customers;

UPDATE customers SET city = 'Rome' WHERE customer\_id = 1;

UPDATE customers SET city = 'Barcelona' WHERE customer\_id = 2;

UPDATE customers SET city = 'Milan' WHERE customer\_id = 3;

UPDATE customers SET city = 'Madrid' WHERE customer\_id = 4;

UPDATE customers SET city = 'Manchester' WHERE customer\_id = 5;

UPDATE customers SET city = 'Bayern' WHERE customer\_id = 6;

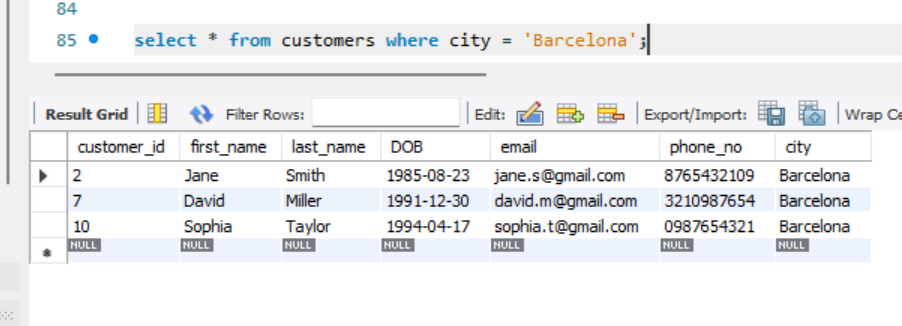
UPDATE customers SET city = 'Barcelona' WHERE customer\_id = 7;

UPDATE customers SET city = 'Madrid' WHERE customer\_id = 8;

UPDATE customers SET city = 'Paris' WHERE customer\_id = 9;

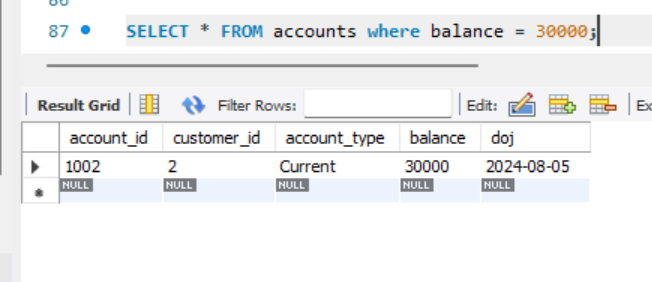
UPDATE customers SET city = 'Barcelona' WHERE customer\_id = 10;

SELECT \* FROM Customers Where city = 'Barcelona';



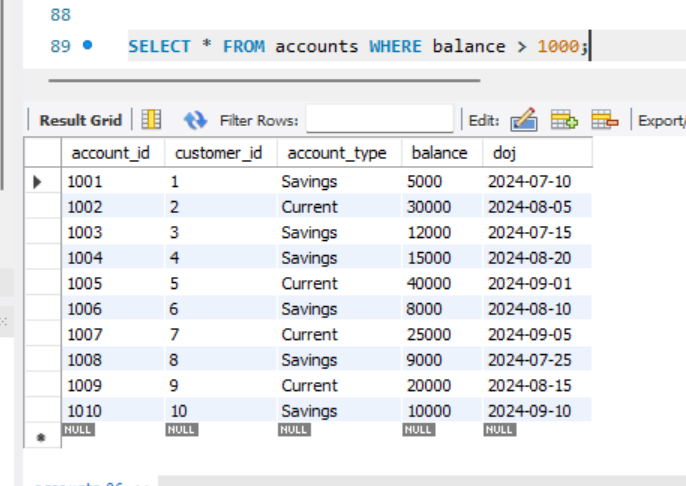
**Write a SQL query to Get the account balance for a specific account.**

SELECT \* FROM accounts where balance = 30000;



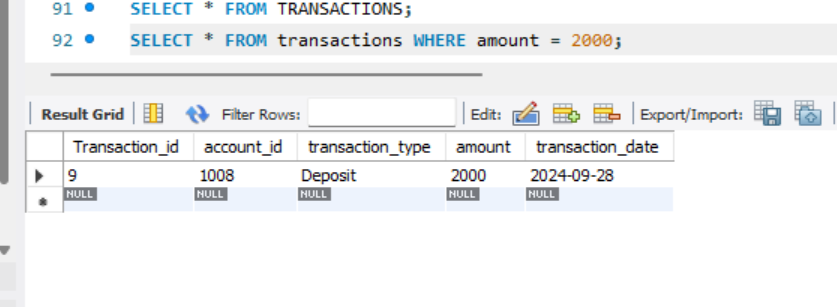
**Write a SQL query to List all current accounts with a balance greater than $1,000.**

SELECT \* FROM accounts WHERE balance < 1000;



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

SELECT \* FROM transactions WHERER amount = 2000;

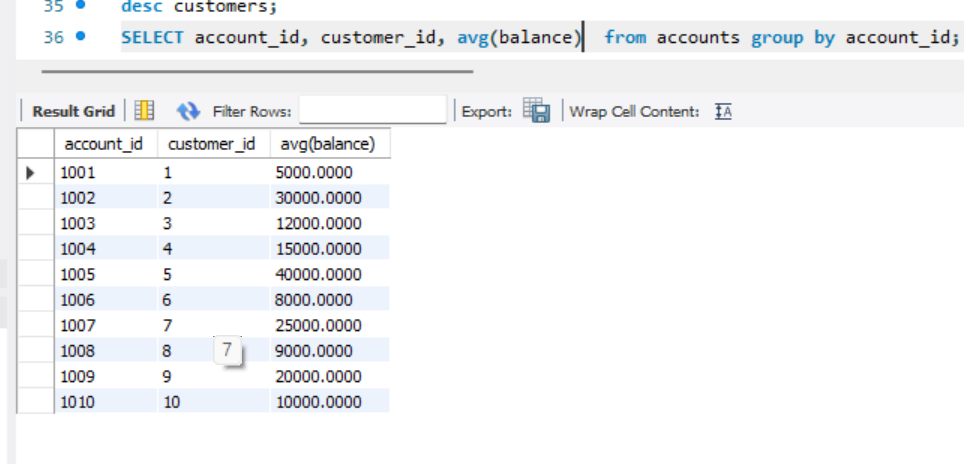


**-------------------------------------------------------------------------------------------**

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

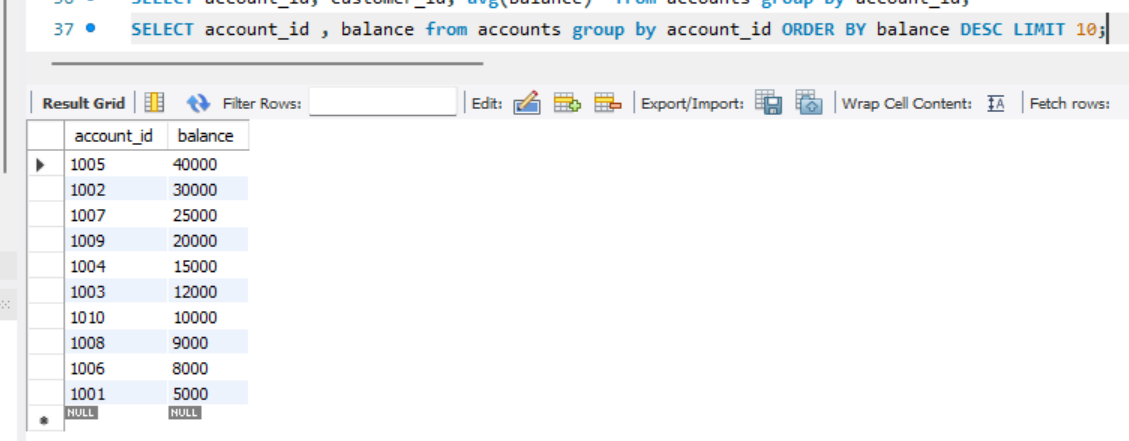
**Write a SQL query to Find the average account balance for all customers.**

SELECT account\_id, customer\_id, avg(balance) from accounts group by account\_id;’



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

SELECT account\_id , balance from accounts group by account\_id ORDER BY balance DESC LIMIT 10;



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

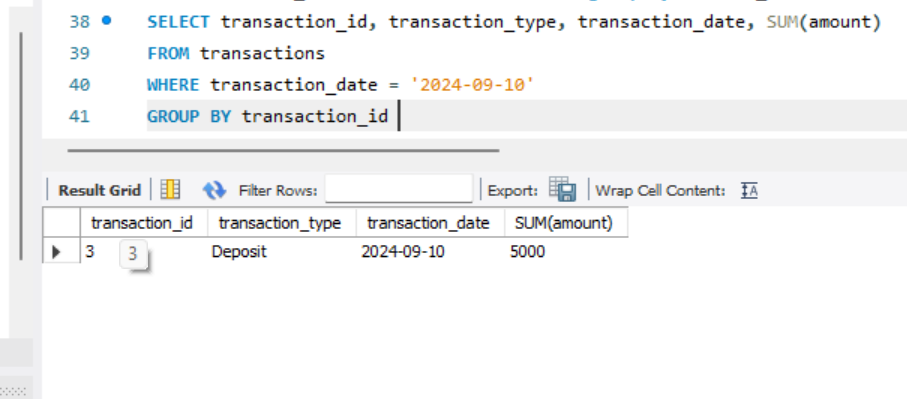
SELECT transaction\_id, transaction\_type, transaction\_date, SUM(amount)

FROM transactions

WHERE transaction\_date = '2024-09-10'

GROUP BY transaction\_id

HAVING transaction\_type = 'deposit' ;



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

ALTER TABLE accounts ADD(doj date ); -------- ---- inserted date of joining

UPDATE accounts

SET doj = CASE account\_id

WHEN 1001 THEN '2024-07-10'

WHEN 1002 THEN '2024-08-05'

WHEN 1003 THEN '2024-07-15'

WHEN 1004 THEN '2024-08-20'

WHEN 1005 THEN '2024-09-01'

WHEN 1006 THEN '2024-08-10'

WHEN 1007 THEN '2024-09-05'

WHEN 1008 THEN '2024-07-25'

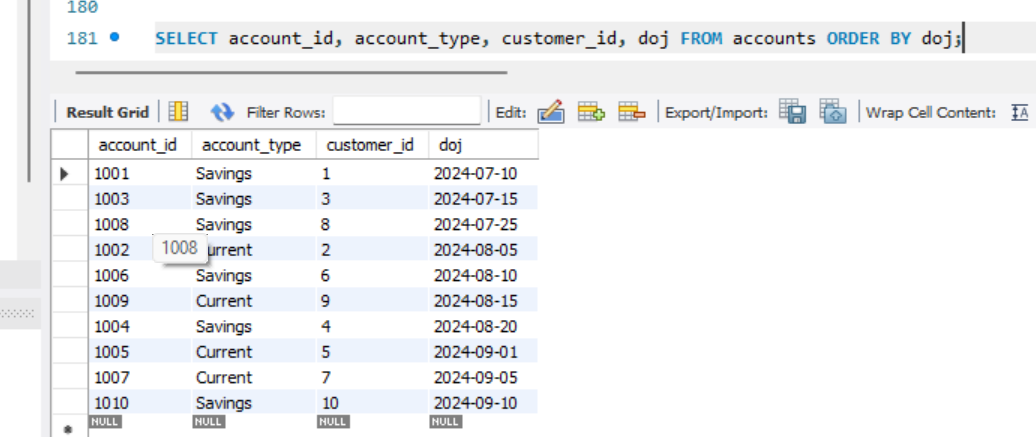
WHEN 1009 THEN '2024-08-15'

WHEN 1010 THEN '2024-09-10'

END

WHERE account\_id IN (1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008, 1009, 1010);

SELECT account\_id, account\_type, customer\_id, doj FROM accounts ORDER BY doj;



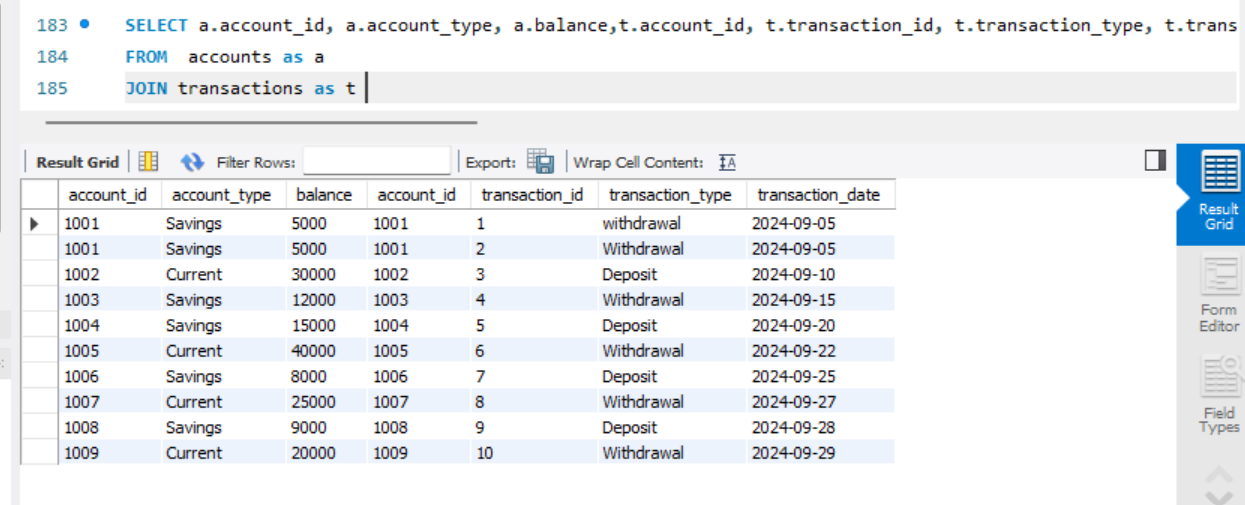
**Write a SQL query to Retrieve transaction details along with the account type.**

SELECT a.account\_id, a.account\_type, a.balance,t.account\_id, t.transaction\_id, t.transaction\_type, t.transaction\_date

FROM accounts as a

JOIN transactions as t

on a.account\_id = t.account\_id;



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

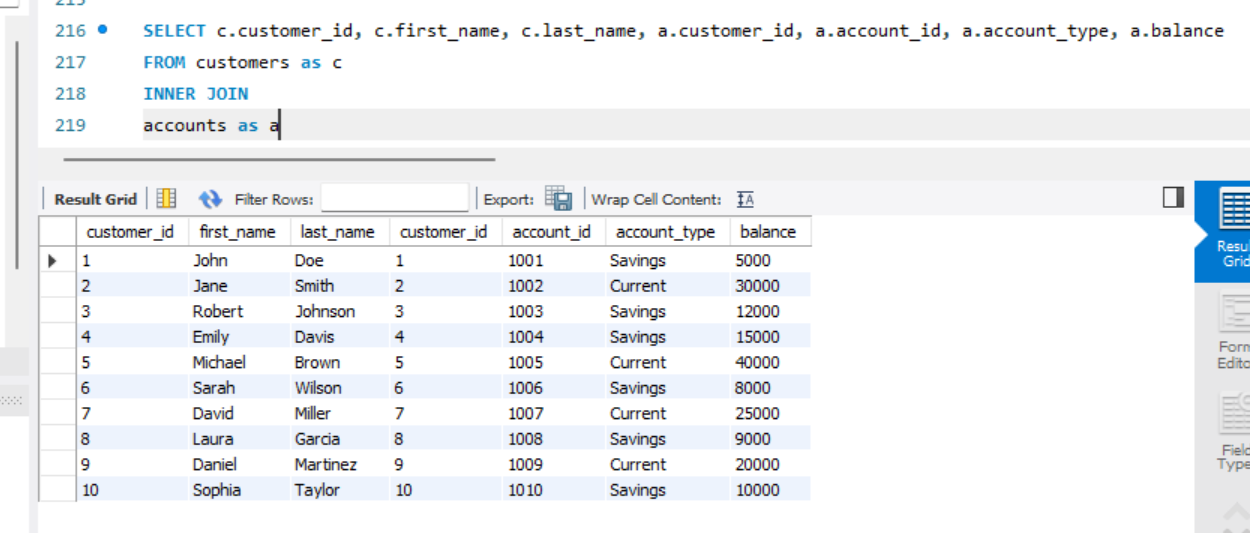
SELECT c.customer\_id, c.first\_name, c.last\_name, a.customer\_id, a.account\_id, a.account\_type, a.balance

FROM customers as c

INNER JOIN

accounts as a

on c.customer\_id = a.customer\_id



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

SELECT

CONCAT(c.first\_name, ' ', c.last\_name) AS full\_name,

t.transaction\_id,

t.transaction\_type,

t.amount,

t.transaction\_date,

a.account\_id,

a.account\_type,

c.customer\_id

FROM

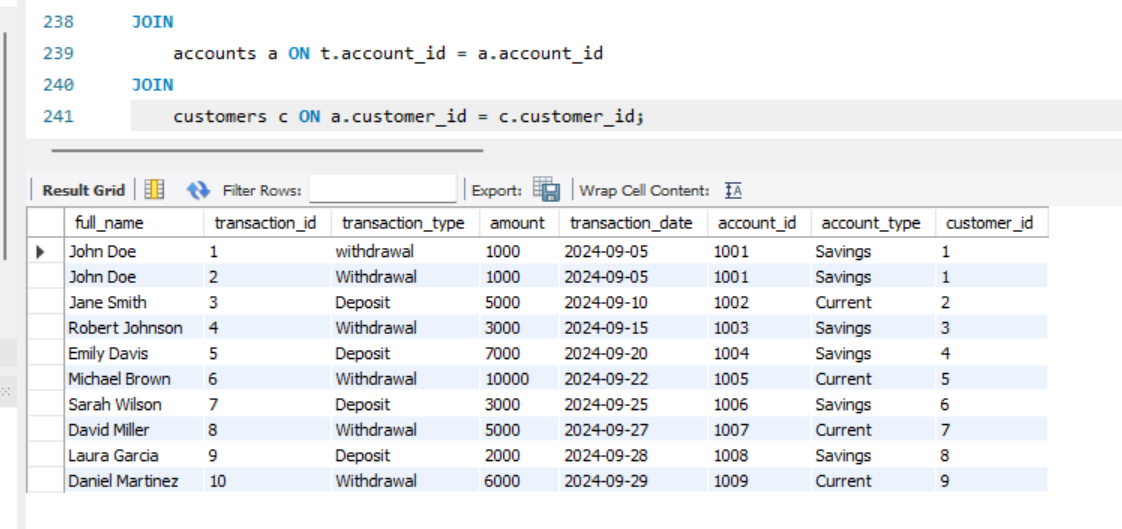
transactions t

JOIN

accounts a ON t.account\_id = a.account\_id

JOIN

customers c ON a.customer\_id = c.customer\_id;



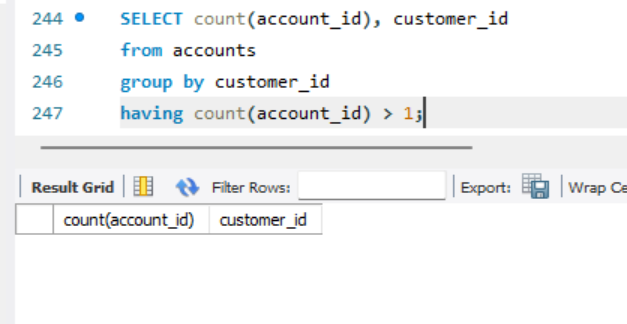
**Write a SQL query to Identify customers who have more than one account.**

SELECT count(account\_id), customer\_id

from accounts

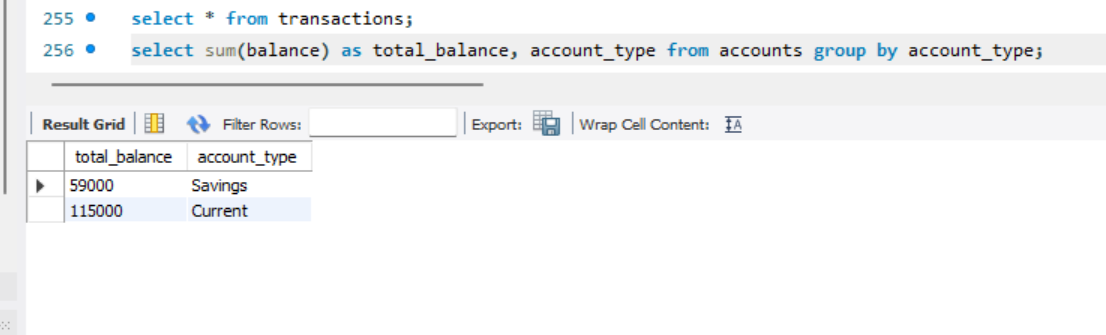
group by customer\_id

having count(account\_id) > 1; -- empty set



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

select sum(balance) as total\_balance, account\_type from accounts group by account\_type;



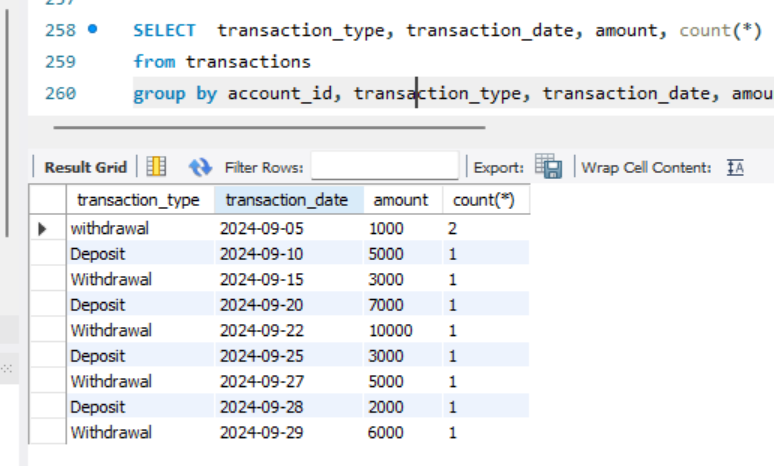
**Identify accounts with the highest number of transactions order by descending order.**

SELECT transaction\_type, transaction\_date, amount, count(\*)

from transactions

group by account\_id, transaction\_type, transaction\_date, amount

ORDER BY count(\*) desc;



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

SELECT c.customer\_id, c.first\_name, c.last\_name, a.customer\_id, a.account\_type, SUM(a.balance)

FROM customers as c

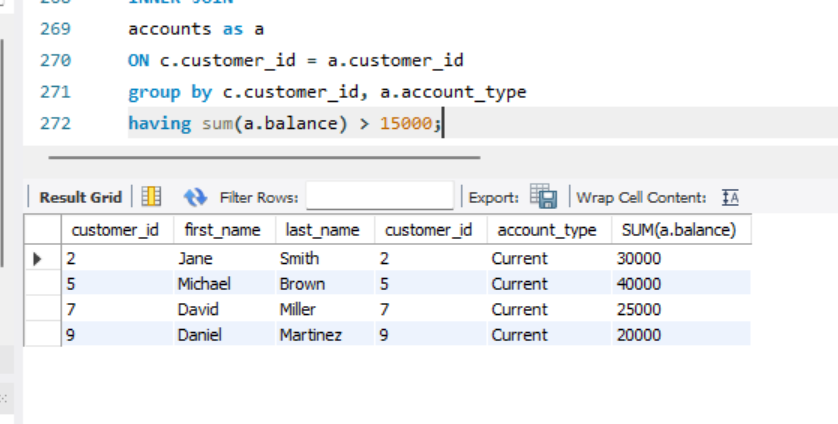
INNER JOIN

accounts as a

ON c.customer\_id = a.customer\_id

group by c.customer\_id, a.account\_type

having sum(a.balance) > 15000;



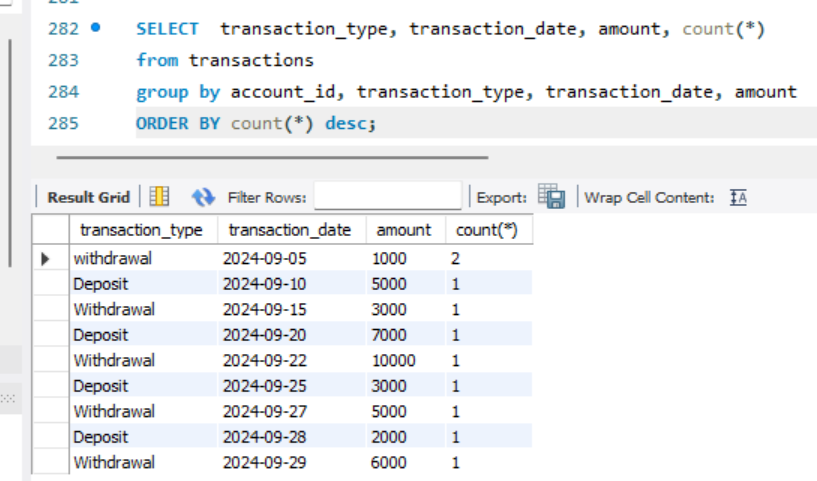
**Identify and list duplicate transactions based on transaction amount, date, and account.**

SELECT transaction\_type, transaction\_date, amount, count(\*)

from transactions

group by account\_id, transaction\_type, transaction\_date, amount

having count(\*)>1;



-------------------------------------------------------------------------------------------------

**TASK 4 : Subquery and its type:**

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

SELECT

c.customer\_id,

CONCAT(c.first\_name, ' ', c.last\_name) AS full\_name,

a.account\_type,

a.balance

FROM

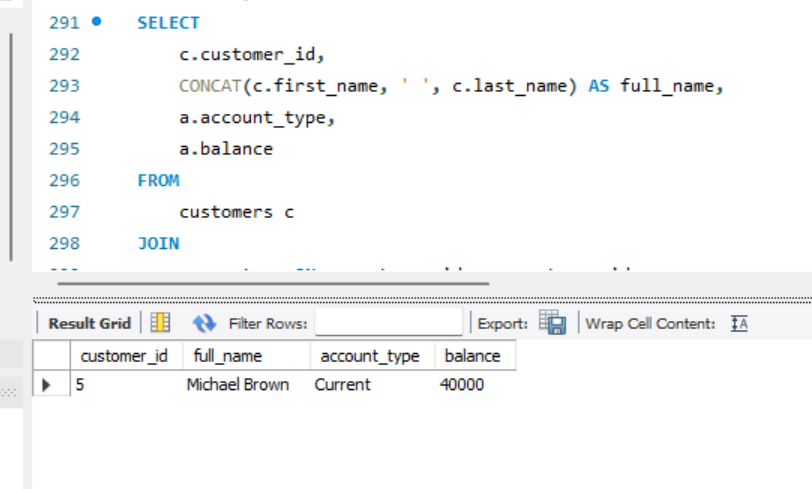
customers c

JOIN

accounts a ON c.customer\_id = a.customer\_id

WHERE

a.balance = (SELECT MAX(balance) FROM accounts);



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

SELECT

customer\_id,

AVG(balance) AS average\_balance

FROM

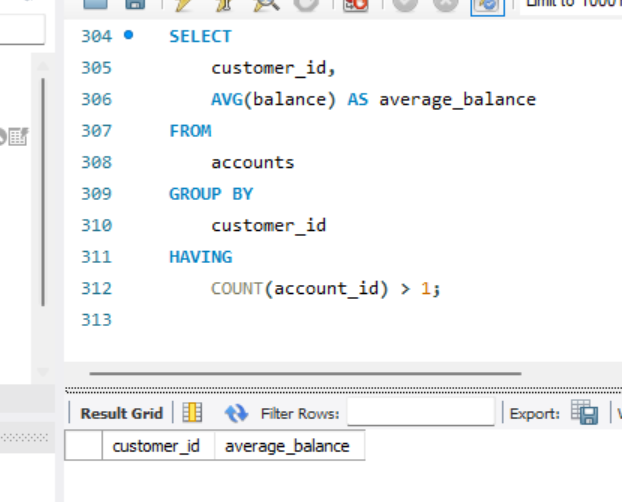
accounts

GROUP BY

customer\_id

HAVING

COUNT(account\_id) > 1;



EMPTY SET

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

SELECT

t.transaction\_id,

t.account\_id,

t.amount,

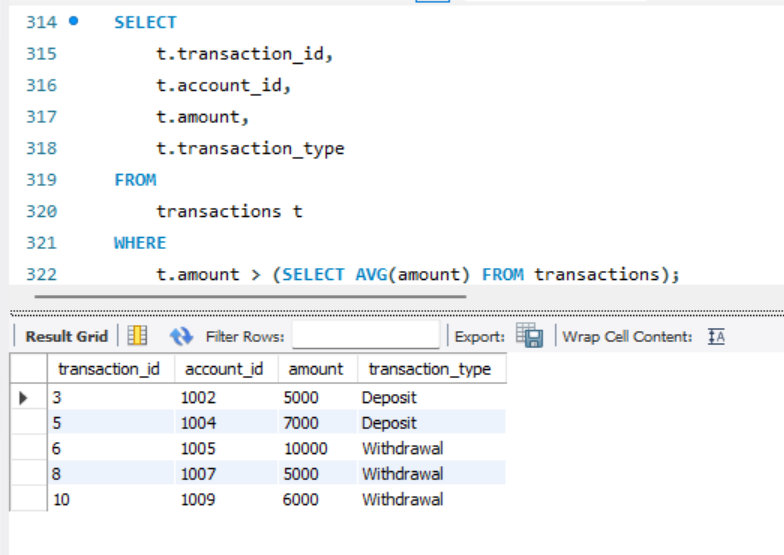
t.transaction\_type

FROM

transactions t

WHERE

t.amount > (SELECT AVG(amount) FROM transactions);



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

SELECT

c.customer\_id,

CONCAT(c.first\_name, ' ', c.last\_name) AS full\_name

FROM

customers c

LEFT JOIN

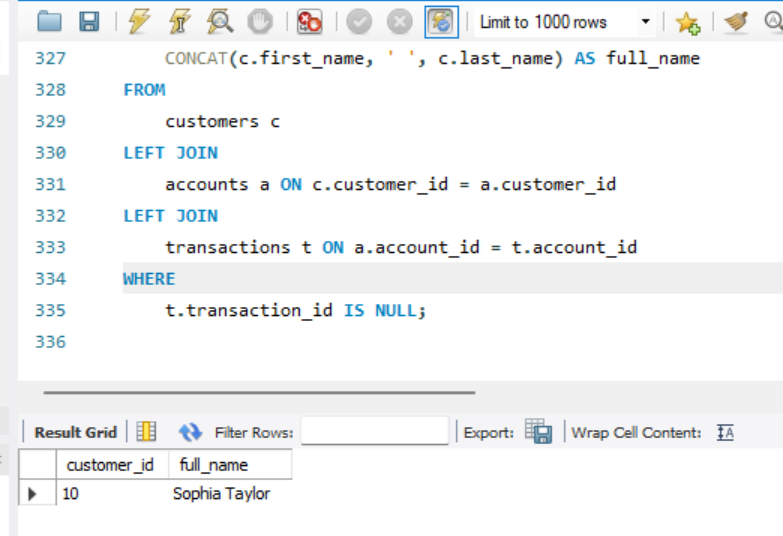
accounts a ON c.customer\_id = a.customer\_id

LEFT JOIN

transactions t ON a.account\_id = t.account\_id

WHERE

t.transaction\_id IS NULL;



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

SELECT

SUM(a.balance) AS total\_balance\_no\_transactions

FROM

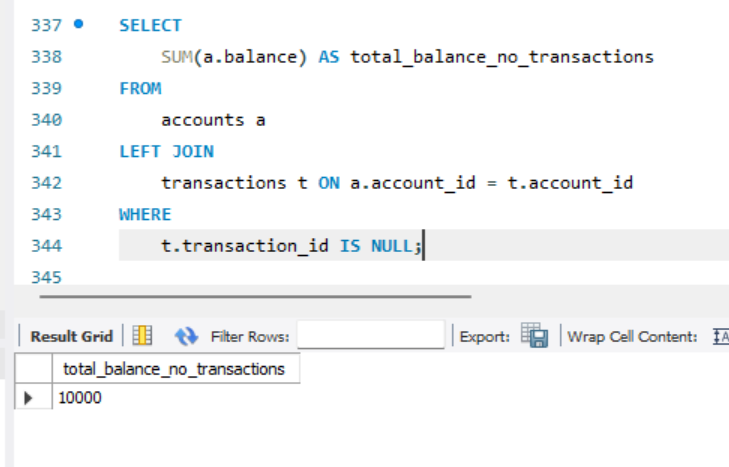
accounts a

LEFT JOIN

transactions t ON a.account\_id = t.account\_id

WHERE

t.transaction\_id IS NULL;



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

SELECT

t.transaction\_id,

t.account\_id,

t.transaction\_type,

t.amount,

t.transaction\_date

FROM

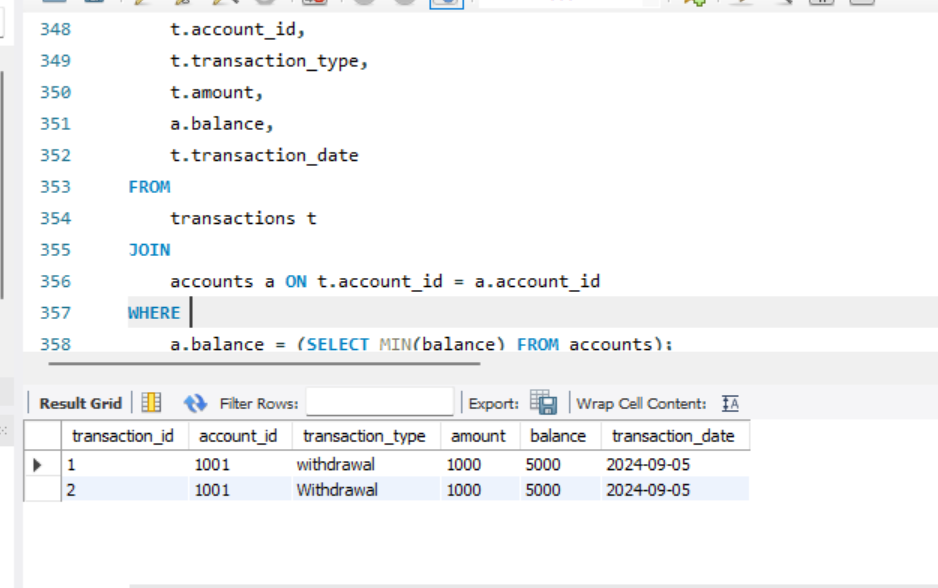
transactions t

JOIN

accounts a ON t.account\_id = a.account\_id

WHERE

a.balance = (SELECT MIN(balance) FROM accounts);



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

SELECT

c.customer\_id,

CONCAT(c.first\_name, ' ', c.last\_name) AS full\_name

FROM

accounts a

JOIN

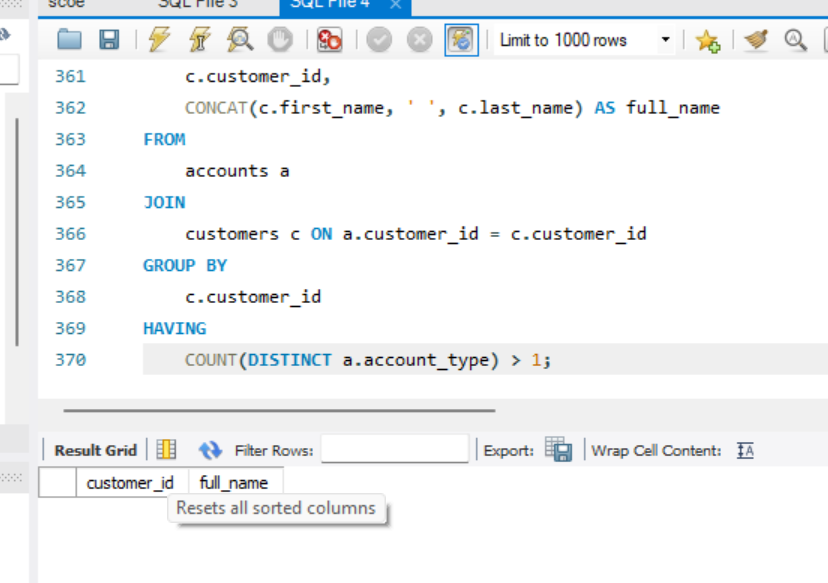
customers c ON a.customer\_id = c.customer\_id

GROUP BY

c.customer\_id

HAVING

COUNT(DISTINCT a.account\_type) > 1;



EMPTY SET

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

SELECT

account\_type,

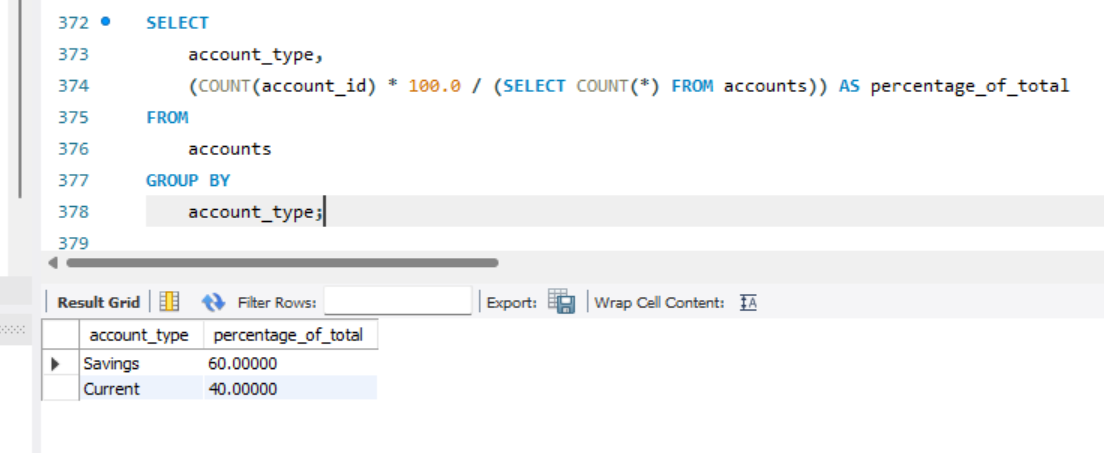
(COUNT(account\_id) \* 100.0 / (SELECT COUNT(\*) FROM accounts)) AS percentage\_of\_total

FROM

accounts

GROUP BY

account\_type;



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

SELECT

t.transaction\_id,

t.transaction\_type,

t.amount,

t.transaction\_date,

a.account\_id,

a.account\_type

FROM

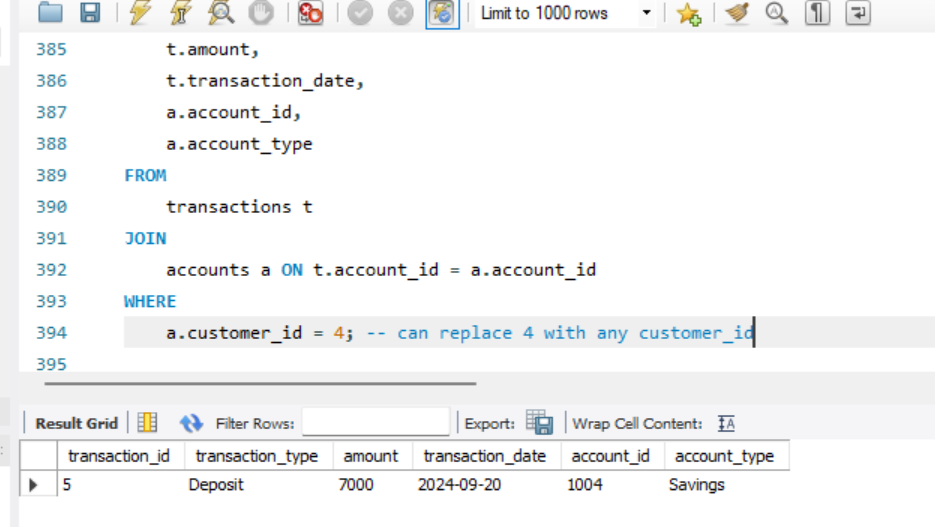
transactions t

JOIN

accounts a ON t.account\_id = a.account\_id

WHERE

a.customer\_id = 4; -- can replace 4 with any customer\_id



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

SELECT

account\_type,

(SELECT SUM(balance) FROM accounts AS a2 WHERE a2.account\_type = a1.account\_type) AS total\_balance

FROM

accounts a1

GROUP BY

account\_type;

