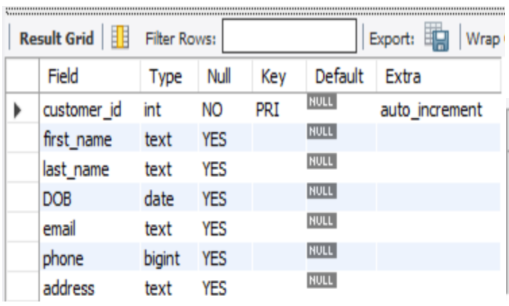
ASSIGNMENT-3(HEXA)

Task-1: By:Satyendra Singh Rathore

Tasks 1: Database Design:

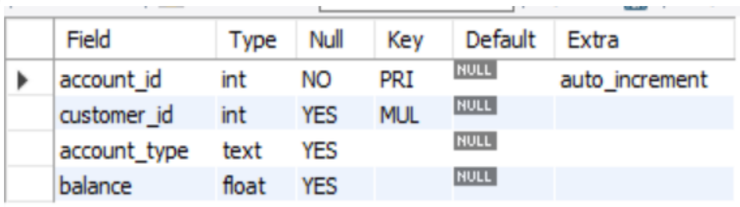
1. Create the database named "HMBank"
2. Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema.
4. Create an ERD (Entity Relationship Diagram) for the database.
5. Create appropriate Primary Key and Foreign Key constraints for referential integrity.
6. Write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

```
-- Task -1

create database HMBank;
use HMBank;

create table Customers(customer_id int primary key auto_increment,first_name text,last_name text,
DOB date,email text,phone bigint,address text);

create table Accounts(account_id int primary key auto_increment,customer_id int,account_type text,
balance float,foreign key(customer_id) references Customers(customer_id));

create table Transactions(transaction_id int primary key auto_increment,account_id int,
transaction_type text,amount bigint,transaction_date date,foreign key (account_id)
references Accounts(account_id));

desc Customers;
desc Accounts;
desc Transactions;
```

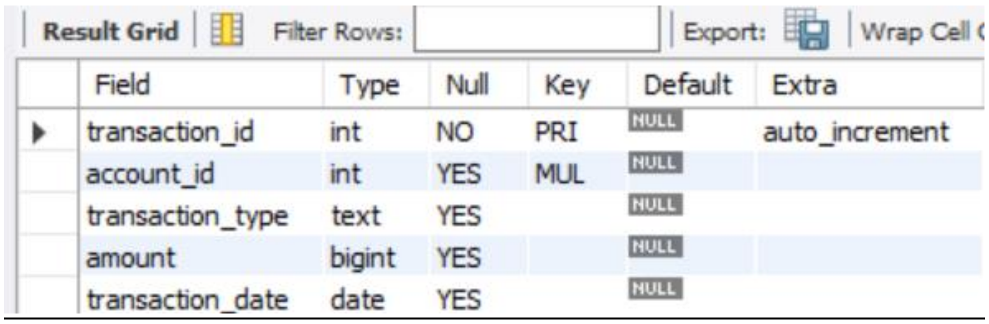


Result Grid | Filter Rows: | Export: | Wrap

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	auto_increment
first_name	text	YES		NULL	
last_name	text	YES		NULL	
DOB	date	YES		NULL	
email	text	YES		NULL	
phone	bigint	YES		NULL	
address	text	YES		NULL	

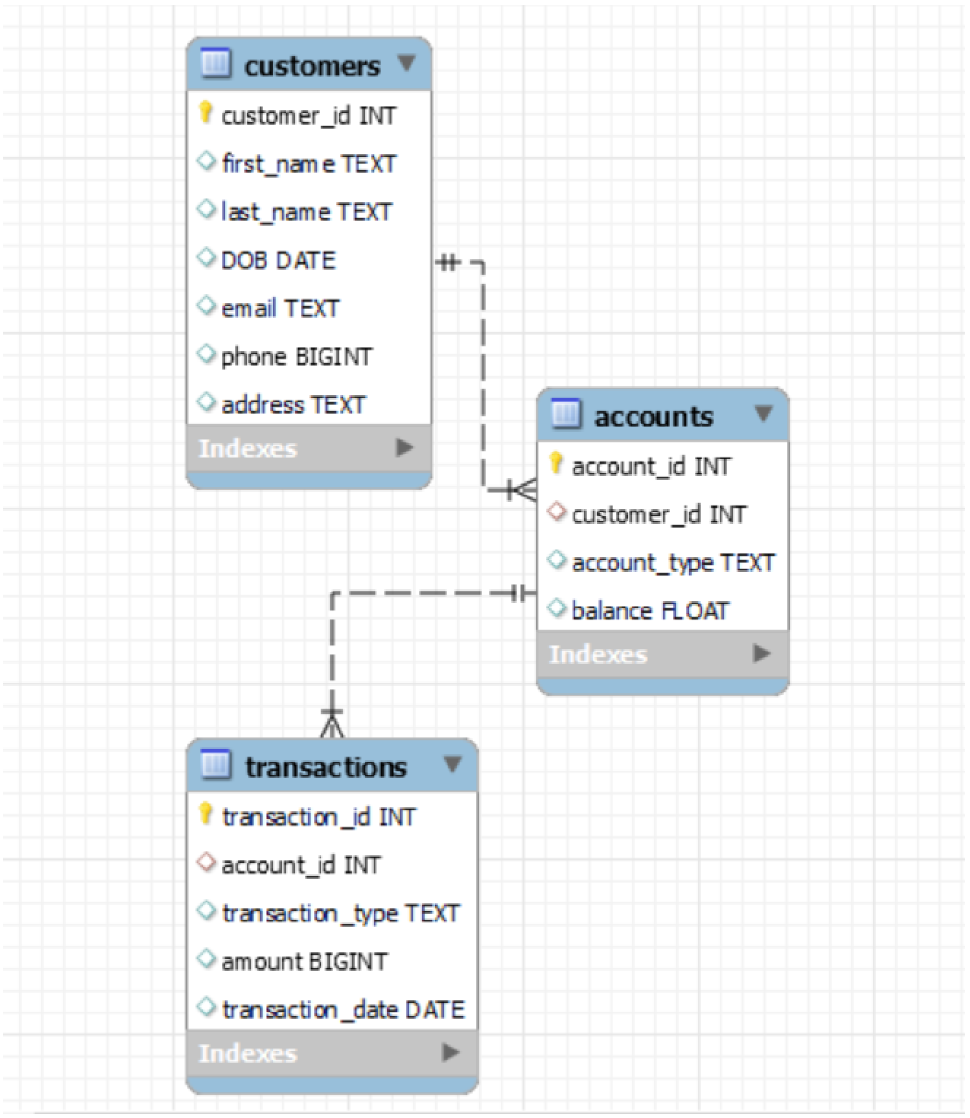


Field	Type	Null	Key	Default	Extra
account_id	int	NO	PRI	NULL	auto_increment
customer_id	int	YES	MUL	NULL	
account_type	text	YES		NULL	
balance	float	YES		NULL	



Field	Type	Null	Key	Default	Extra
transaction_id	int	NO	PRI	NULL	auto_increment
account_id	int	YES	MUL	NULL	
transaction_type	text	YES		NULL	
amount	bigint	YES		NULL	
transaction_date	date	YES		NULL	

Relationship Model:



ER Diagram:

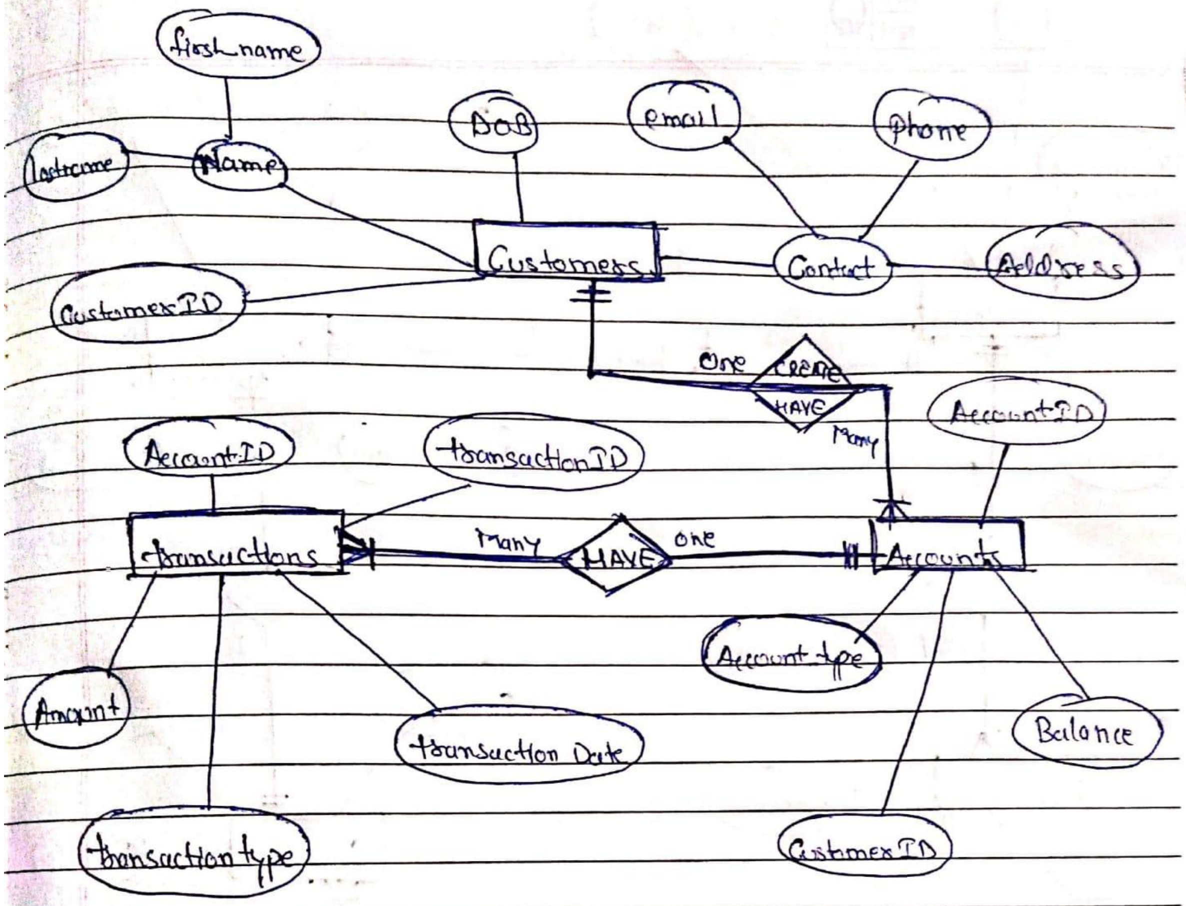
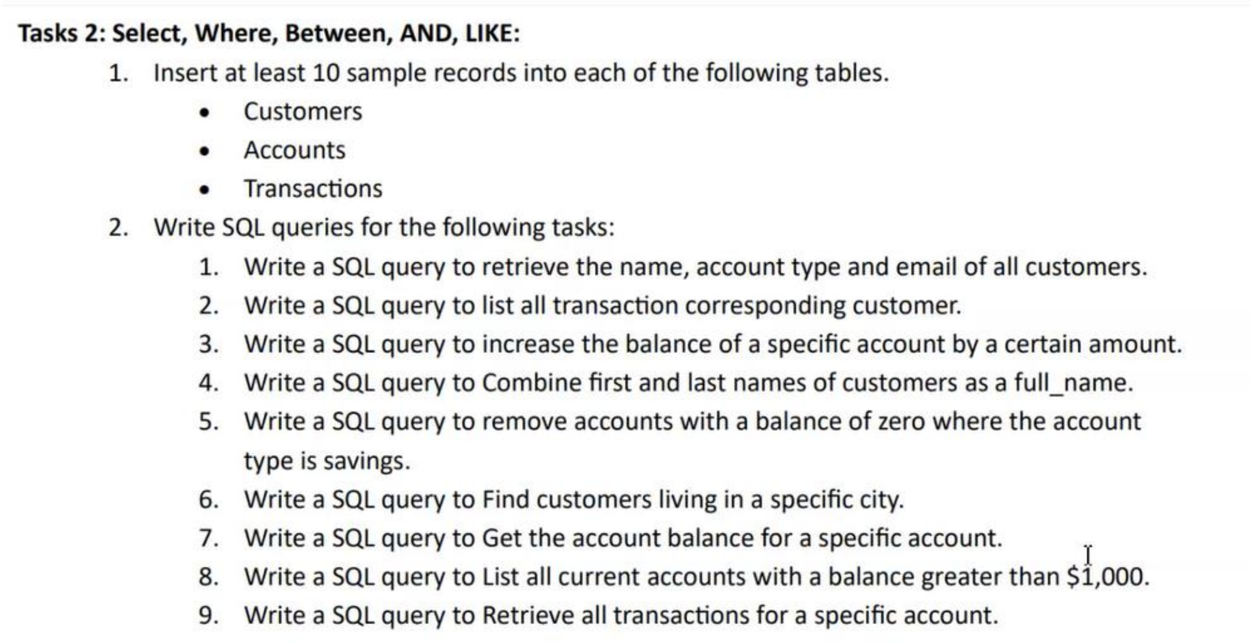


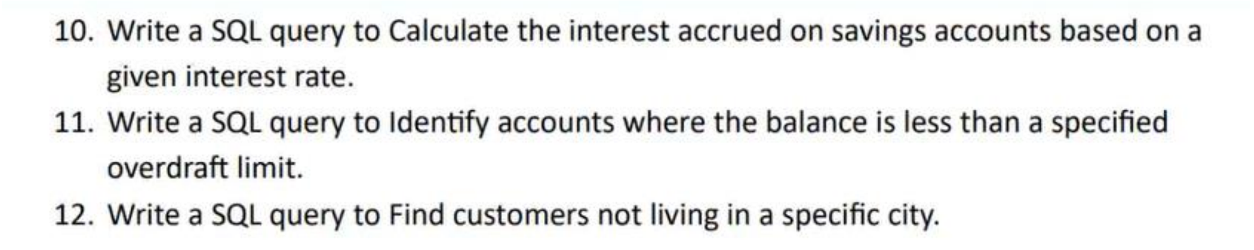
fig1 (ERD of HMBank)

Task-2:

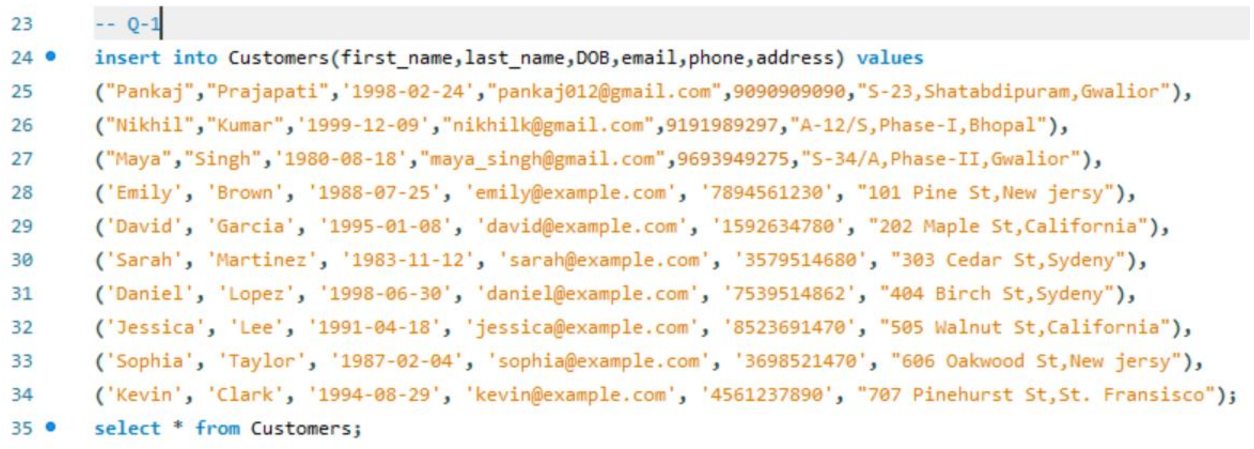


Tasks 2: Select, Where, Between, AND, LIKE:

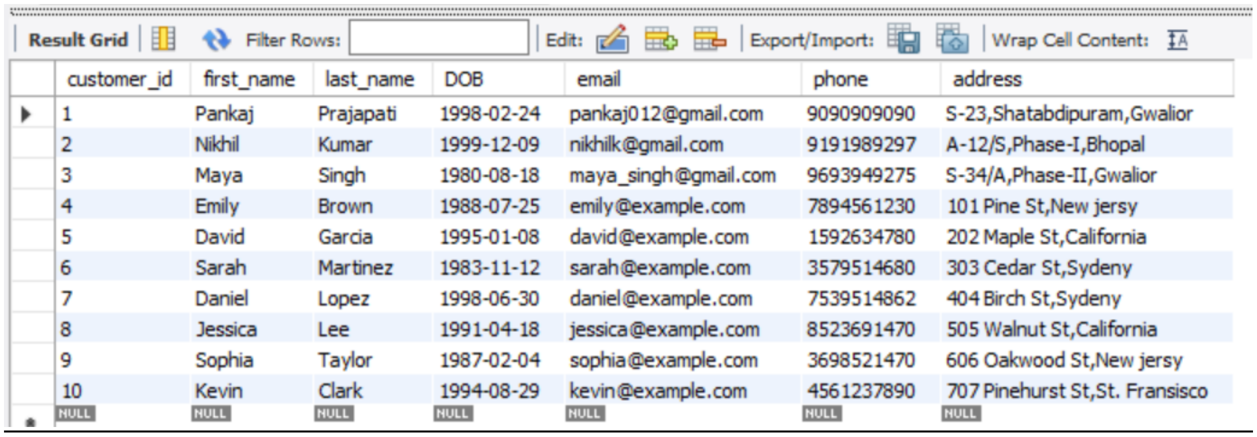
1. Insert at least 10 sample records into each of the following tables.
   - Customers
   - Accounts
   - Transactions
2. Write SQL queries for the following tasks:
   1. Write a SQL query to retrieve the name, account type and email of all customers.
   2. Write a SQL query to list all transaction corresponding customer.
   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.
   5. Write a SQL query to remove accounts with a balance of zero where the account type is savings.
   6. Write a SQL query to Find customers living in a specific city.
   7. Write a SQL query to Get the account balance for a specific account.
   8. Write a SQL query to List all current accounts with a balance greater than $1,000.
   9. Write a SQL query to Retrieve all transactions for a specific account.



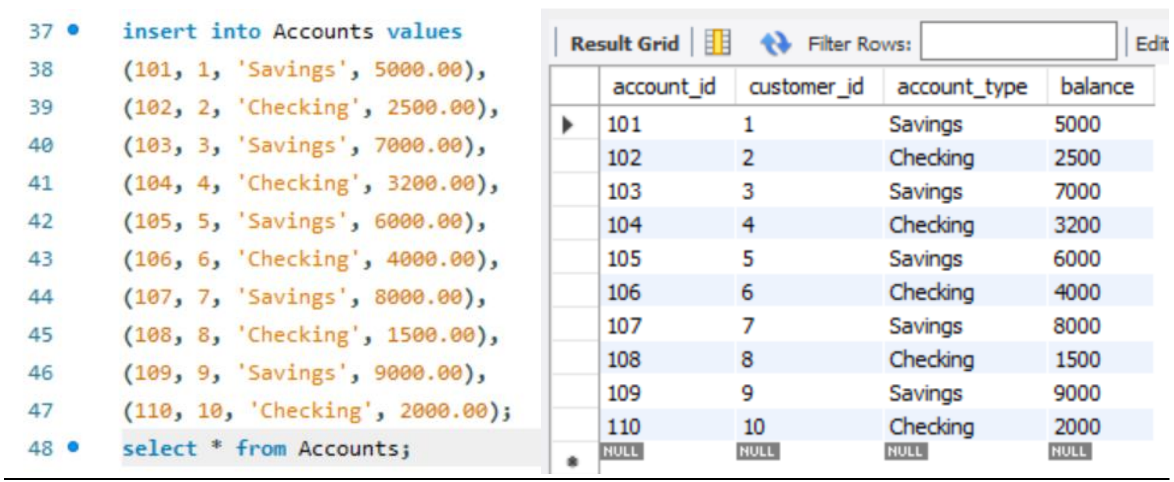
10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.
11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit.
12. Write a SQL query to Find customers not living in a specific city.



```sql
-- Q-1
insert into Customers(first_name,last_name,DOB,email,phone,address) values
("Pankaj","Prajapati",'1998-02-24',"pankaj012@gmail.com",9090909090,"S-23,Shatabdipuram,Gwalior"),
("Nikhil","Kumar",'1999-12-09',"nikhilk@gmail.com",9191989297,"A-12/S,Phase-I,Bhopal"),
("Maya","Singh",'1980-08-18',"maya_singh@gmail.com",9693949275,"S-34/A,Phase-II,Gwalior"),
('Emily', 'Brown', '1988-07-25', 'emily@example.com', '7894561230', "101 Pine St,New jersy"),
('David', 'Garcia', '1995-01-08', 'david@example.com', '1592634780', "202 Maple St,California"),
('Sarah', 'Martinez', '1983-11-12', 'sarah@example.com', '3579514680', "303 Cedar St,Sydeny"),
('Daniel', 'Lopez', '1998-06-30', 'daniel@example.com', '7539514862', "404 Birch St,Sydeny"),
('Jessica', 'Lee', '1991-04-18', 'jessica@example.com', '8523691470', "505 Walnut St,California"),
('Sophia', 'Taylor', '1987-02-04', 'sophia@example.com', '3698521470', "606 Oakwood St,New jersy"),
('Kevin', 'Clark', '1994-08-29', 'kevin@example.com', '4561237890', "707 Pinehurst St,St. Fransisco");
select * from Customers;
```

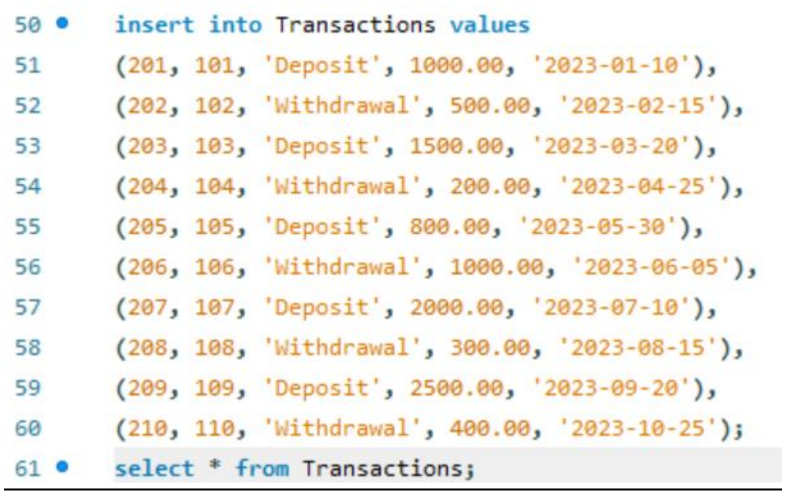


customer_id	first_name	last_name	DOB	email	phone	address
1	Pankaj	Prajapati	1998-02-24	pankaj012@gmail.com	9090909090	S-23,Shatabdipuram,Gwalior
2	Nikhil	Kumar	1999-12-09	nikhilk@gmail.com	9191989297	A-12/S,Phase-I,Bhopal
3	Maya	Singh	1980-08-18	maya_singh@gmail.com	9693949275	S-34/A,Phase-II,Gwalior
4	Emily	Brown	1988-07-25	emily@example.com	7894561230	101 Pine St,New jersy
5	David	Garcia	1995-01-08	david@example.com	1592634780	202 Maple St,California
6	Sarah	Martinez	1983-11-12	sarah@example.com	3579514680	303 Cedar St,Sydeny
7	Daniel	Lopez	1998-06-30	daniel@example.com	7539514862	404 Birch St,Sydeny
8	Jessica	Lee	1991-04-18	jessica@example.com	8523691470	505 Walnut St,California
9	Sophia	Taylor	1987-02-04	sophia@example.com	3698521470	606 Oakwood St,New jersy
10	Kevin	Clark	1994-08-29	kevin@example.com	4561237890	707 Pinehurst St,St. Fransisco
NULL	NULL	NULL	NULL	NULL	NULL	NULL

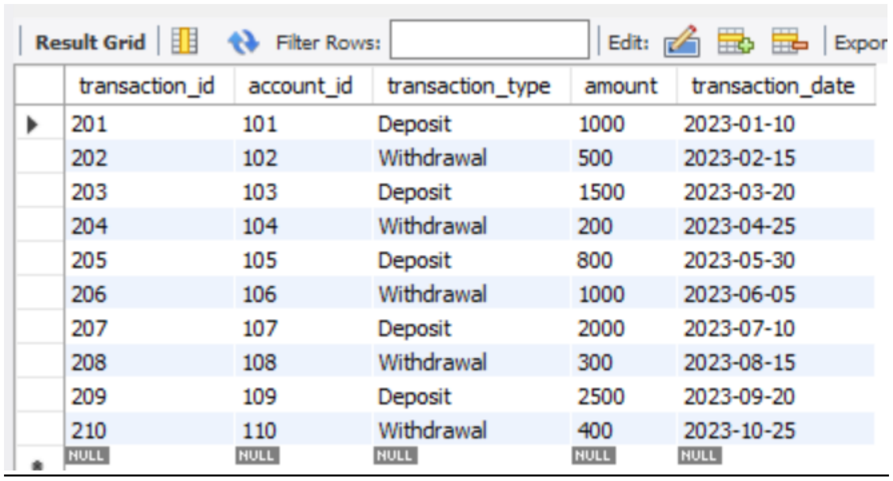


```sql
insert into Accounts values
(101, 1, 'Savings', 5000.00),
(102, 2, 'Checking', 2500.00),
(103, 3, 'Savings', 7000.00),
(104, 4, 'Checking', 3200.00),
(105, 5, 'Savings', 6000.00),
(106, 6, 'Checking', 4000.00),
(107, 7, 'Savings', 8000.00),
(108, 8, 'Checking', 1500.00),
(109, 9, 'Savings', 9000.00),
(110, 10, 'Checking', 2000.00);
select * from Accounts;
```

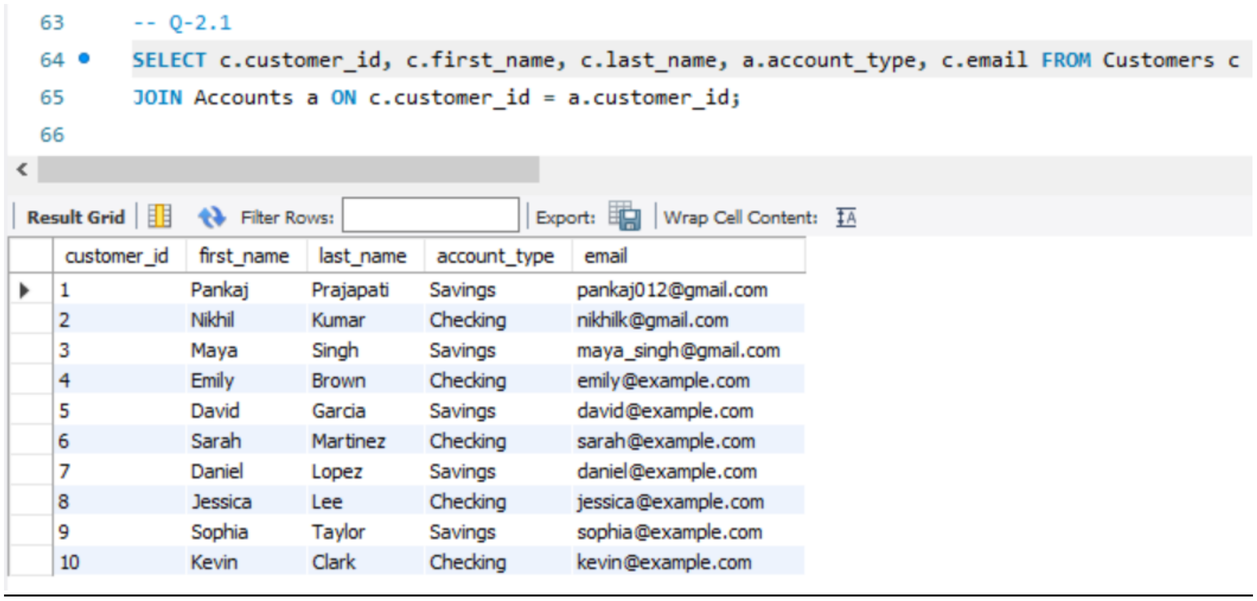
account_id	customer_id	account_type	balance
101	1	Savings	5000
102	2	Checking	2500
103	3	Savings	7000
104	4	Checking	3200
105	5	Savings	6000
106	6	Checking	4000
107	7	Savings	8000
108	8	Checking	1500
109	9	Savings	9000
110	10	Checking	2000
NULL	NULL	NULL	NULL



```sql
insert into Transactions values
(201, 101, 'Deposit', 1000.00, '2023-01-10'),
(202, 102, 'Withdrawal', 500.00, '2023-02-15'),
(203, 103, 'Deposit', 1500.00, '2023-03-20'),
(204, 104, 'Withdrawal', 200.00, '2023-04-25'),
(205, 105, 'Deposit', 800.00, '2023-05-30'),
(206, 106, 'Withdrawal', 1000.00, '2023-06-05'),
(207, 107, 'Deposit', 2000.00, '2023-07-10'),
(208, 108, 'Withdrawal', 300.00, '2023-08-15'),
(209, 109, 'Deposit', 2500.00, '2023-09-20'),
(210, 110, 'Withdrawal', 400.00, '2023-10-25');
select * from Transactions;
```

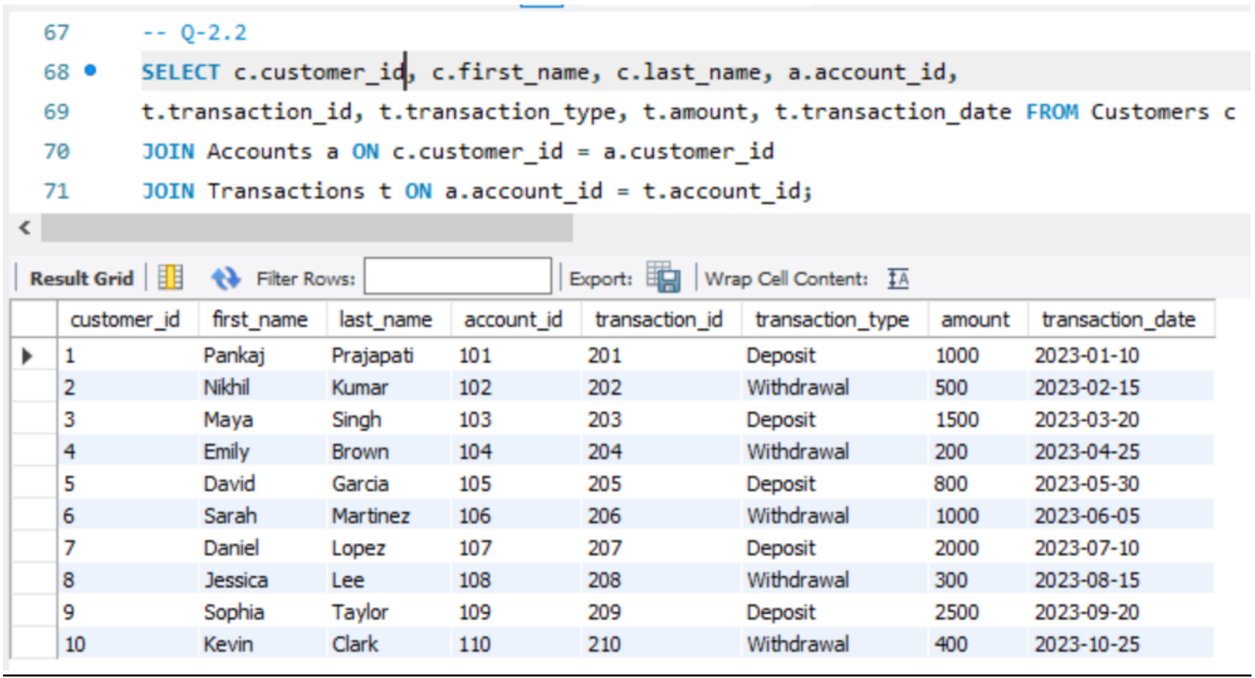


transaction_id	account_id	transaction_type	amount	transaction_date
201	101	Deposit	1000	2023-01-10
202	102	Withdrawal	500	2023-02-15
203	103	Deposit	1500	2023-03-20
204	104	Withdrawal	200	2023-04-25
205	105	Deposit	800	2023-05-30
206	106	Withdrawal	1000	2023-06-05
207	107	Deposit	2000	2023-07-10
208	108	Withdrawal	300	2023-08-15
209	109	Deposit	2500	2023-09-20
210	110	Withdrawal	400	2023-10-25
NULL	NULL	NULL	NULL	NULL



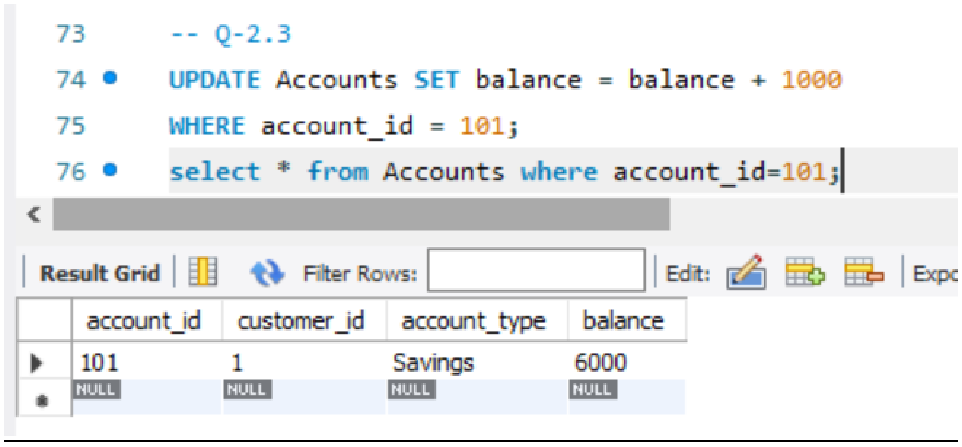
```sql
-- Q-2.1
SELECT c.customer_id, c.first_name, c.last_name, a.account_type, c.email FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id;
```

customer_id	first_name	last_name	account_type	email
1	Pankaj	Prajapati	Savings	pankaj012@gmail.com
2	Nikhil	Kumar	Checking	nikhilk@gmail.com
3	Maya	Singh	Savings	maya_singh@gmail.com
4	Emily	Brown	Checking	emily@example.com
5	David	Garcia	Savings	david@example.com
6	Sarah	Martinez	Checking	sarah@example.com
7	Daniel	Lopez	Savings	daniel@example.com
8	Jessica	Lee	Checking	jessica@example.com
9	Sophia	Taylor	Savings	sophia@example.com
10	Kevin	Clark	Checking	kevin@example.com



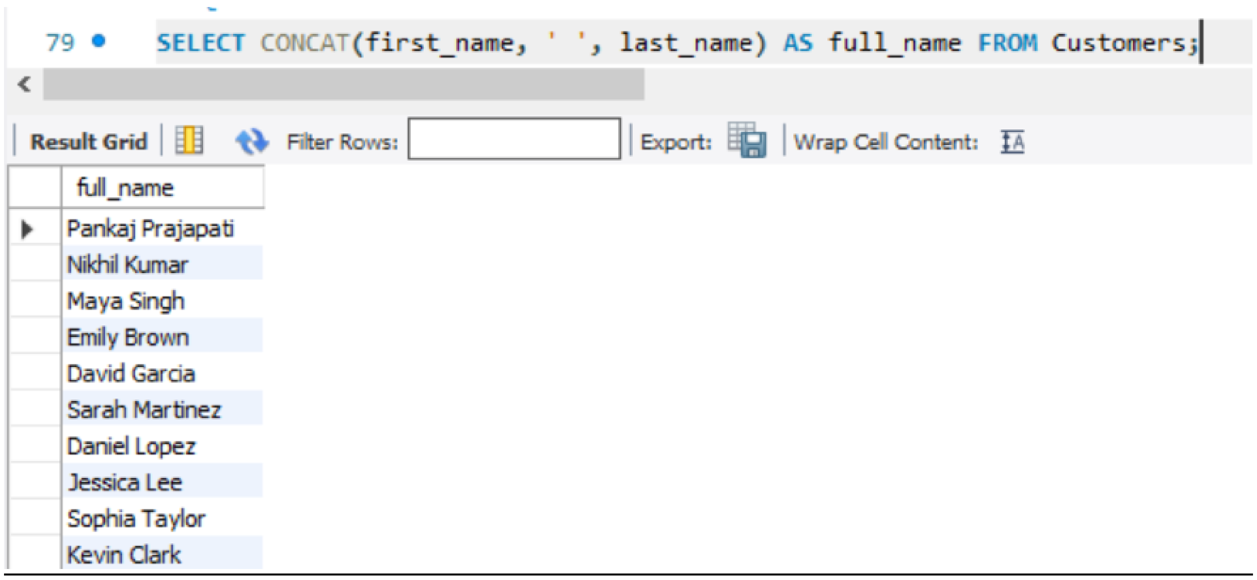
```sql
-- Q-2.2
SELECT c.customer_id, c.first_name, c.last_name, a.account_id,
t.transaction_id, t.transaction_type, t.amount, t.transaction_date FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
JOIN Transactions t ON a.account_id = t.account_id;
```

customer_id	first_name	last_name	account_id	transaction_id	transaction_type	amount	transaction_date
1	Pankaj	Prajapati	101	201	Deposit	1000	2023-01-10
2	Nikhil	Kumar	102	202	Withdrawal	500	2023-02-15
3	Maya	Singh	103	203	Deposit	1500	2023-03-20
4	Emily	Brown	104	204	Withdrawal	200	2023-04-25
5	David	Garcia	105	205	Deposit	800	2023-05-30
6	Sarah	Martinez	106	206	Withdrawal	1000	2023-06-05
7	Daniel	Lopez	107	207	Deposit	2000	2023-07-10
8	Jessica	Lee	108	208	Withdrawal	300	2023-08-15
9	Sophia	Taylor	109	209	Deposit	2500	2023-09-20
10	Kevin	Clark	110	210	Withdrawal	400	2023-10-25



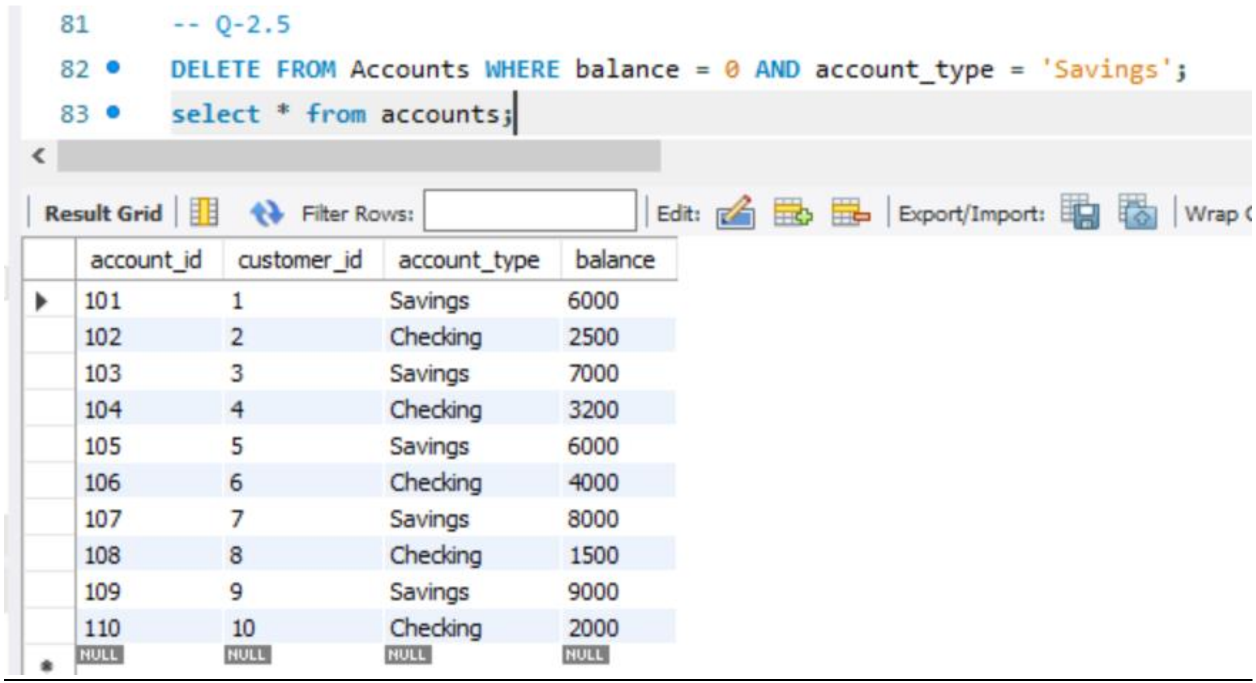
```sql
-- Q-2.3
UPDATE Accounts SET balance = balance + 1000
WHERE account_id = 101;
select * from Accounts where account_id=101;
```

account_id	customer_id	account_type	balance
101	1	Savings	6000
NULL	NULL	NULL	NULL



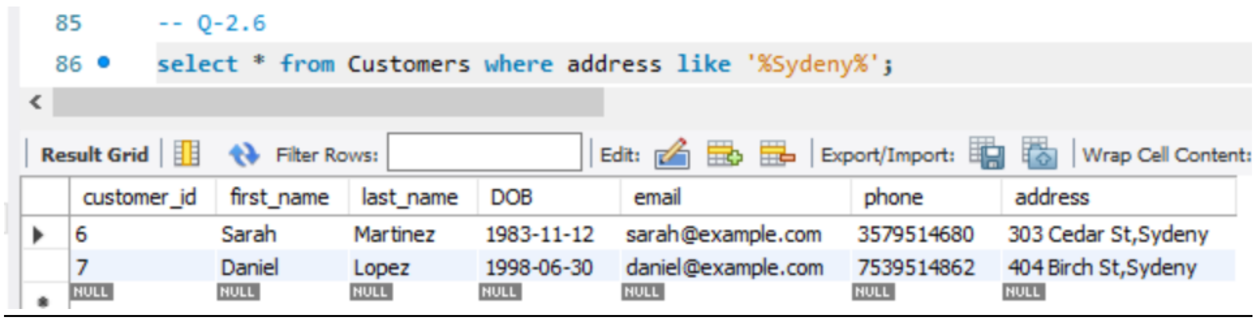
```sql
SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM Customers;
```

full_name
Pankaj Prajapati
Nikhil Kumar
Maya Singh
Emily Brown
David Garcia
Sarah Martinez
Daniel Lopez
Jessica Lee
Sophia Taylor
Kevin Clark



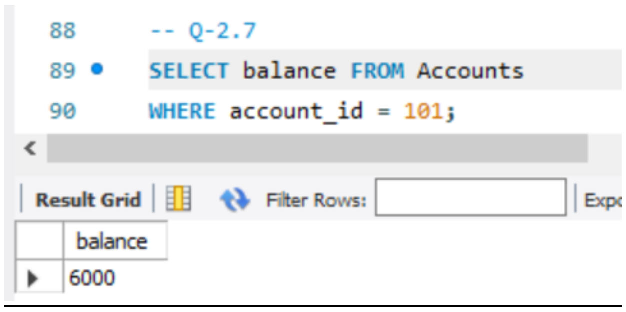
```sql
-- Q-2.5
DELETE FROM Accounts WHERE balance = 0 AND account_type = 'Savings';
select * from accounts;
```

account_id	customer_id	account_type	balance
101	1	Savings	6000
102	2	Checking	2500
103	3	Savings	7000
104	4	Checking	3200
105	5	Savings	6000
106	6	Checking	4000
107	7	Savings	8000
108	8	Checking	1500
109	9	Savings	9000
110	10	Checking	2000
NULL	NULL	NULL	NULL



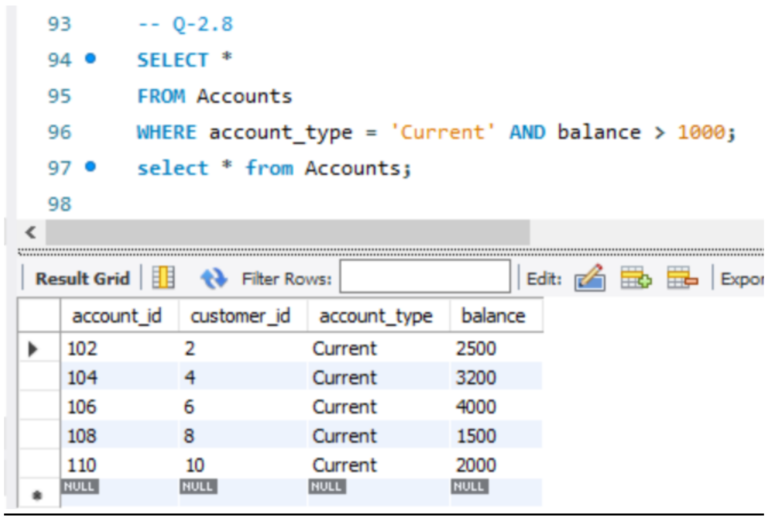
```sql
-- Q-2.6
select * from Customers where address like '%Sydeny%';
```

customer_id	first_name	last_name	DOB	email	phone	address
6	Sarah	Martinez	1983-11-12	sarah@example.com	3579514680	303 Cedar St,Sydeny
7	Daniel	Lopez	1998-06-30	daniel@example.com	7539514862	404 Birch St,Sydeny
NULL	NULL	NULL	NULL	NULL	NULL	NULL



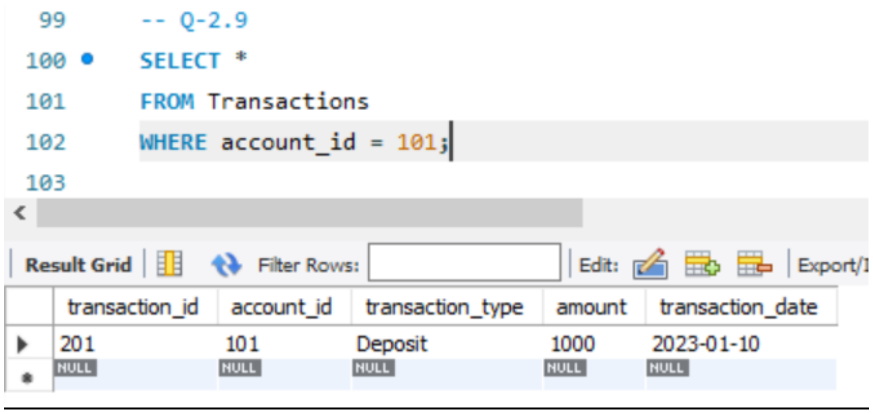
```sql
-- Q-2.7
SELECT balance FROM Accounts
WHERE account_id = 101;
```

balance
6000



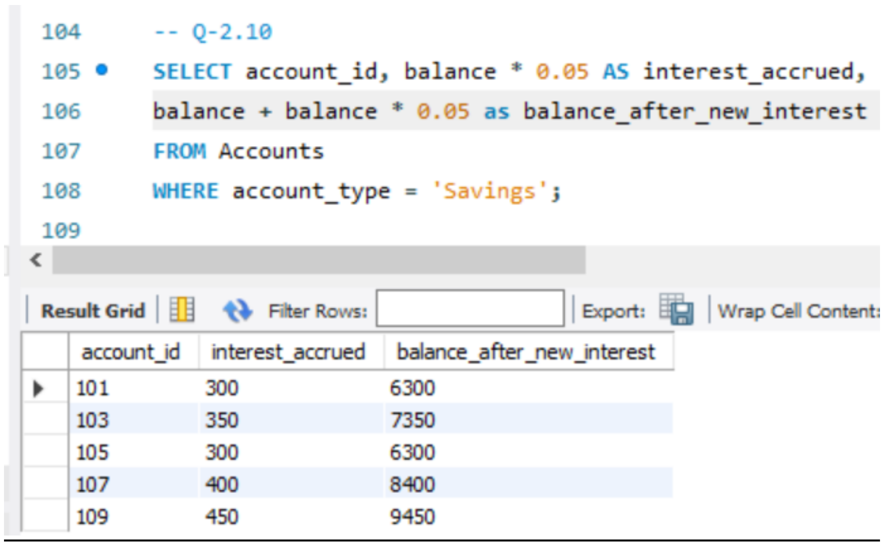
```sql
-- Q-2.8
SELECT *
FROM Accounts
WHERE account_type = 'Current' AND balance > 1000;
select * from Accounts;
```

account_id	customer_id	account_type	balance
102	2	Current	2500
104	4	Current	3200
106	6	Current	4000
108	8	Current	1500
110	10	Current	2000
NULL	NULL	NULL	NULL



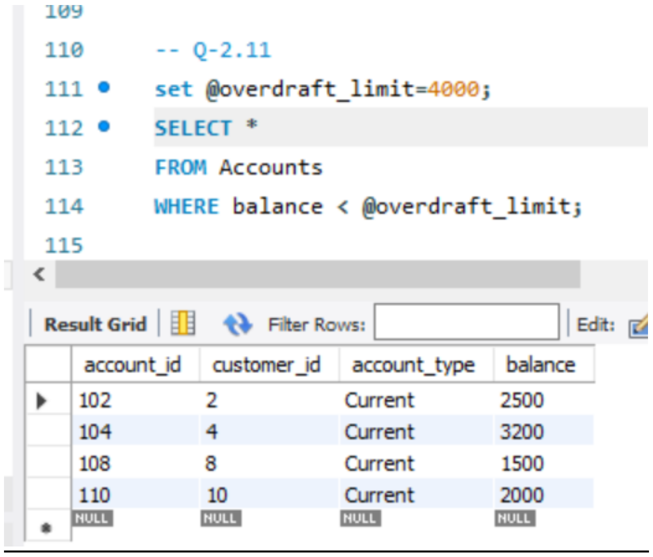
```sql
-- Q-2.9
SELECT *
FROM Transactions
WHERE account_id = 101;
```

transaction_id	account_id	transaction_type	amount	transaction_date
201	101	Deposit	1000	2023-01-10
NULL	NULL	NULL	NULL	NULL



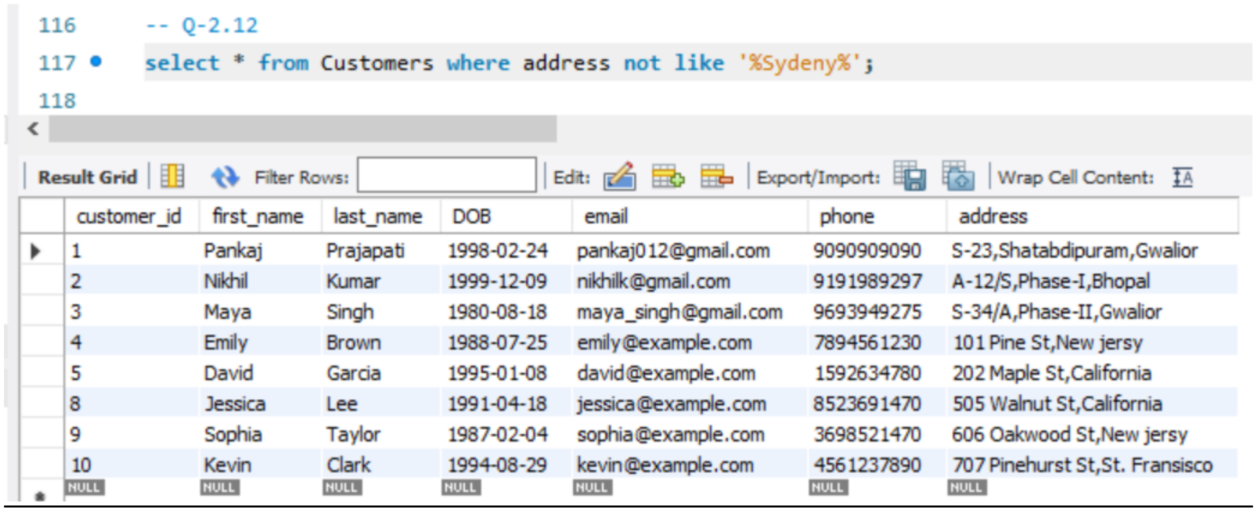
```sql
-- Q-2.10
SELECT account_id, balance * 0.05 AS interest_accrued,
balance + balance * 0.05 as balance_after_new_interest
FROM Accounts
WHERE account_type = 'Savings';
```

account_id	interest_accrued	balance_after_new_interest
101	300	6300
103	350	7350
105	300	6300
107	400	8400
109	450	9450



```sql
-- Q-2.11
set @overdraft_limit=4000;
SELECT *
FROM Accounts
WHERE balance < @overdraft_limit;
```

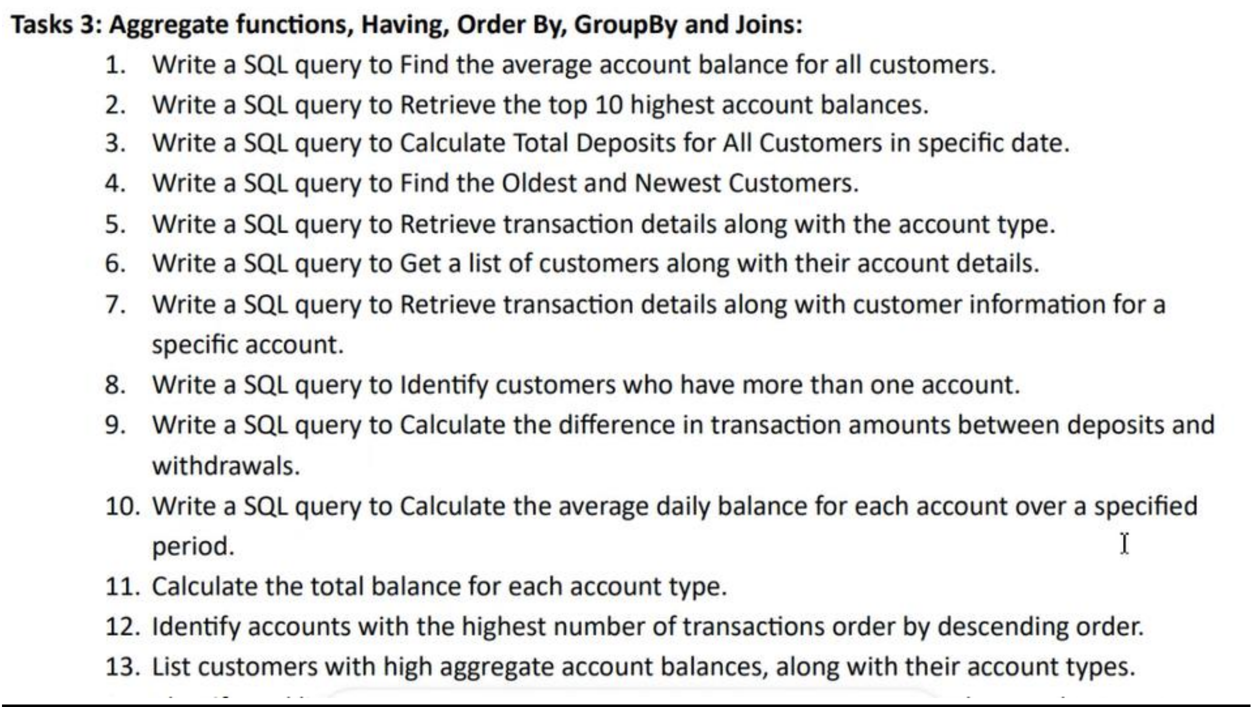
account_id	customer_id	account_type	balance
102	2	Current	2500
104	4	Current	3200
108	8	Current	1500
110	10	Current	2000
NULL	NULL	NULL	NULL



```sql
-- Q-2.12
select * from Customers where address not like '%Sydeny%';
```

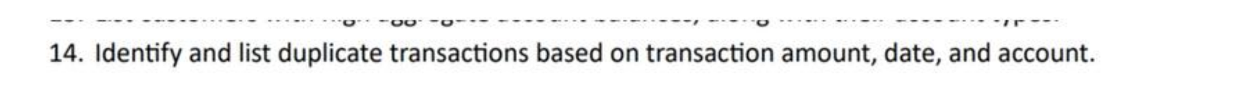
customer_id	first_name	last_name	DOB	email	phone	address
1	Pankaj	Prajapati	1998-02-24	pankaj012@gmail.com	9090909090	S-23,Shatabdipuram,Gwalior
2	Nikhil	Kumar	1999-12-09	nikhilk@gmail.com	9191989297	A-12/S,Phase-I,Bhopal
3	Maya	Singh	1980-08-18	maya_singh@gmail.com	9693949275	S-34/A,Phase-II,Gwalior
4	Emily	Brown	1988-07-25	emily@example.com	7894561230	101 Pine St,New jersy
5	David	Garcia	1995-01-08	david@example.com	1592634780	202 Maple St,California
8	Jessica	Lee	1991-04-18	jessica@example.com	8523691470	505 Walnut St,California
9	Sophia	Taylor	1987-02-04	sophia@example.com	3698521470	606 Oakwood St,New jersy
10	Kevin	Clark	1994-08-29	kevin@example.com	4561237890	707 Pinehurst St,St. Fransisco
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Task-3:

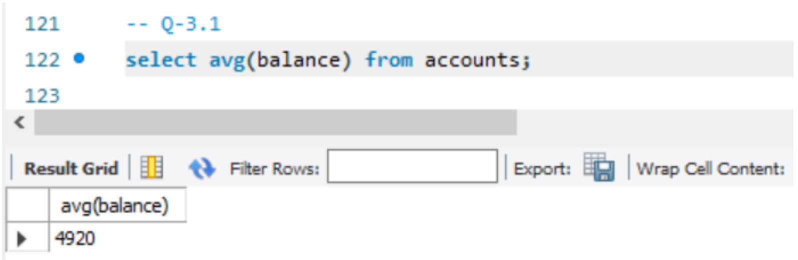


Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:

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.

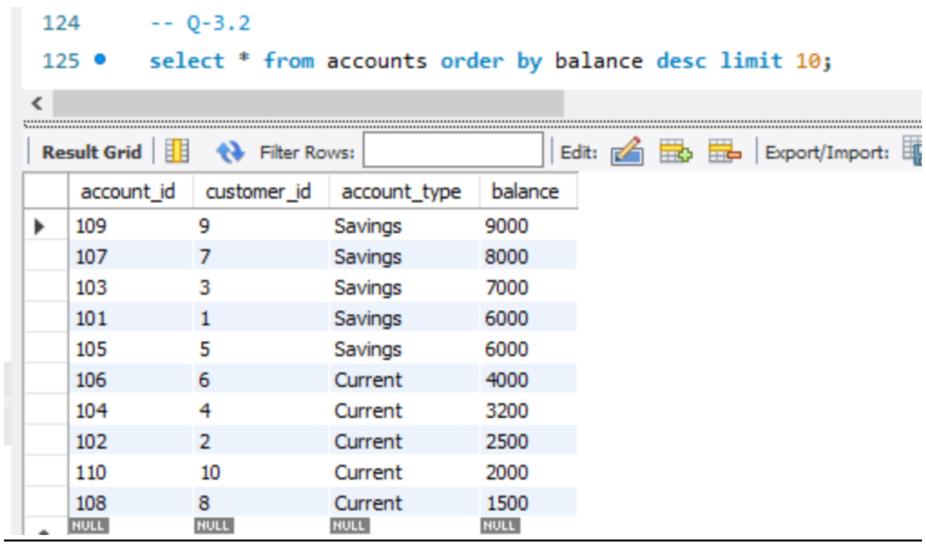


14. Identify and list duplicate transactions based on transaction amount, date, and account.



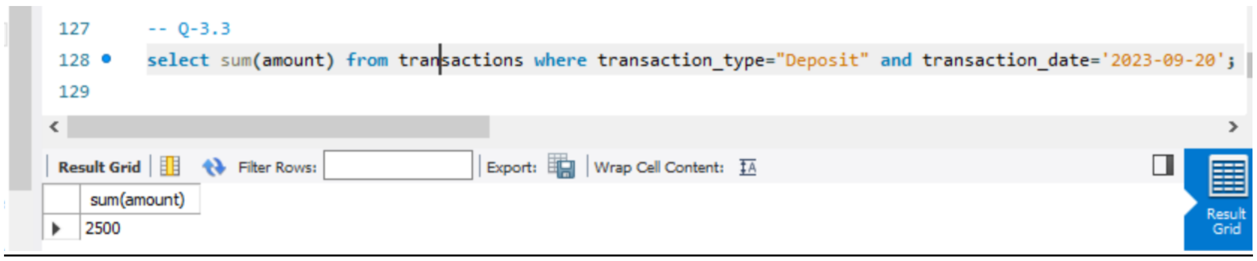
```
-- Q-3.1
select avg(balance) from accounts;
```

	avg(balance)
▶	4920



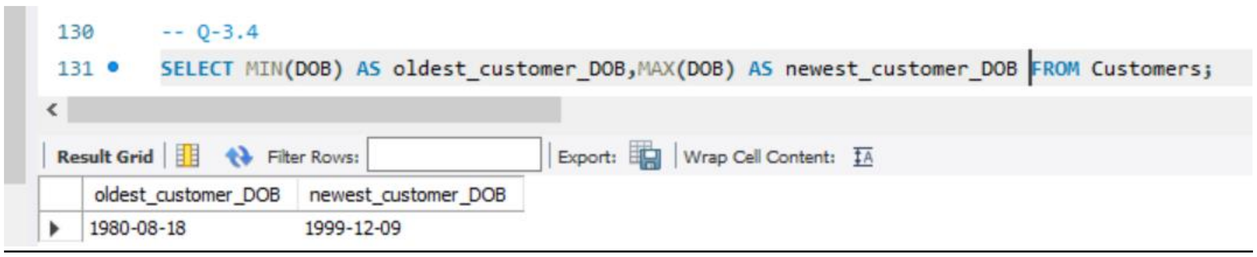
```sql
-- Q-3.2
select * from accounts order by balance desc limit 10;
```

account_id	customer_id	account_type	balance
109	9	Savings	9000
107	7	Savings	8000
103	3	Savings	7000
101	1	Savings	6000
105	5	Savings	6000
106	6	Current	4000
104	4	Current	3200
102	2	Current	2500
110	10	Current	2000
108	8	Current	1500
NULL	NULL	NULL	NULL



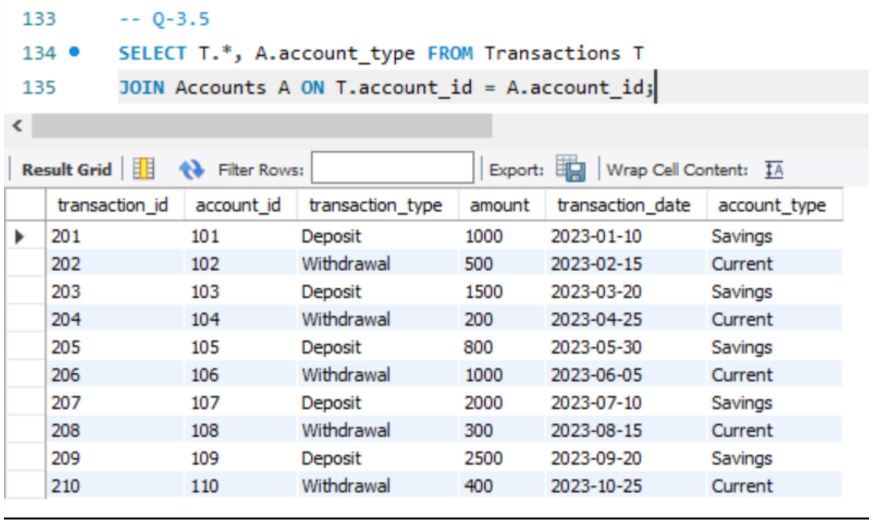
```sql
-- Q-3.3
select sum(amount) from transactions where transaction_type="Deposit" and transaction_date='2023-09-20';
```

sum(amount)
2500



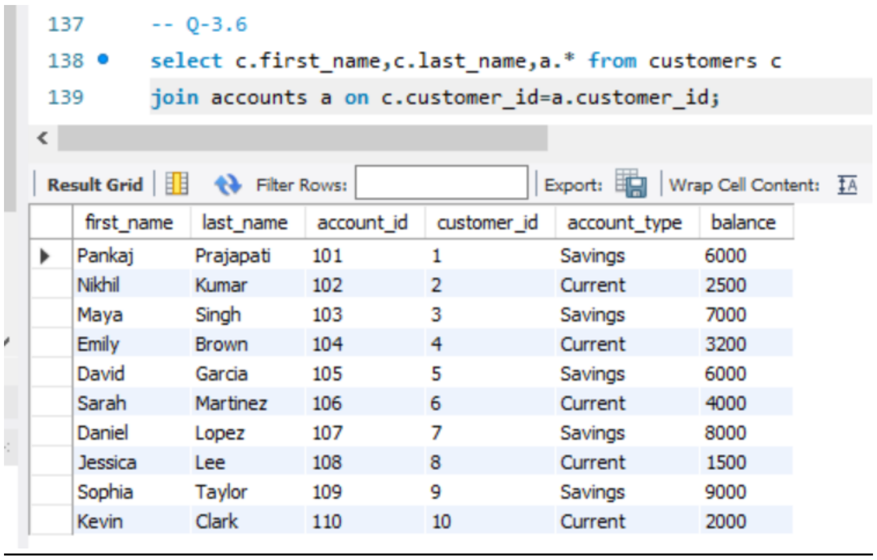
```sql
-- Q-3.4
SELECT MIN(DOB) AS oldest_customer_DOB,MAX(DOB) AS newest_customer_DOB FROM Customers;
```

oldest_customer_DOB	newest_customer_DOB
1980-08-18	1999-12-09



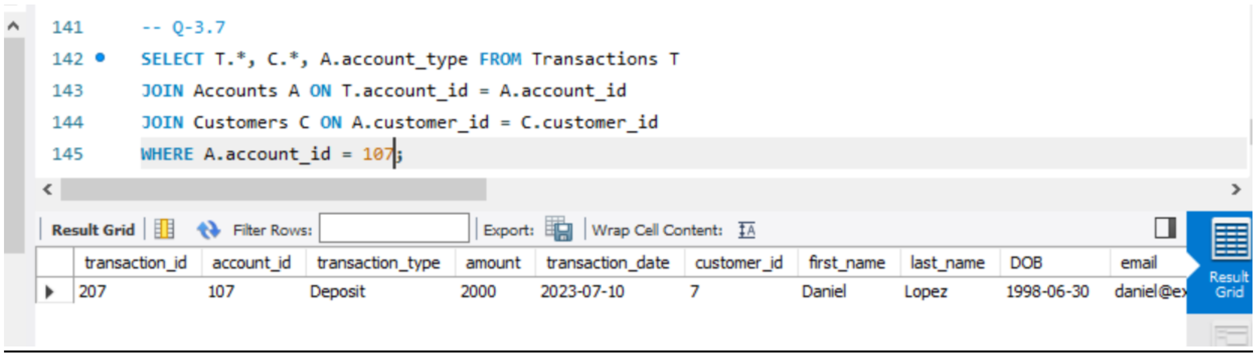
```sql
-- Q-3.5
SELECT T.*, A.account_type FROM Transactions T
JOIN Accounts A ON T.account_id = A.account_id;
```

transaction_id	account_id	transaction_type	amount	transaction_date	account_type
201	101	Deposit	1000	2023-01-10	Savings
202	102	Withdrawal	500	2023-02-15	Current
203	103	Deposit	1500	2023-03-20	Savings
204	104	Withdrawal	200	2023-04-25	Current
205	105	Deposit	800	2023-05-30	Savings
206	106	Withdrawal	1000	2023-06-05	Current
207	107	Deposit	2000	2023-07-10	Savings
208	108	Withdrawal	300	2023-08-15	Current
209	109	Deposit	2500	2023-09-20	Savings
210	110	Withdrawal	400	2023-10-25	Current



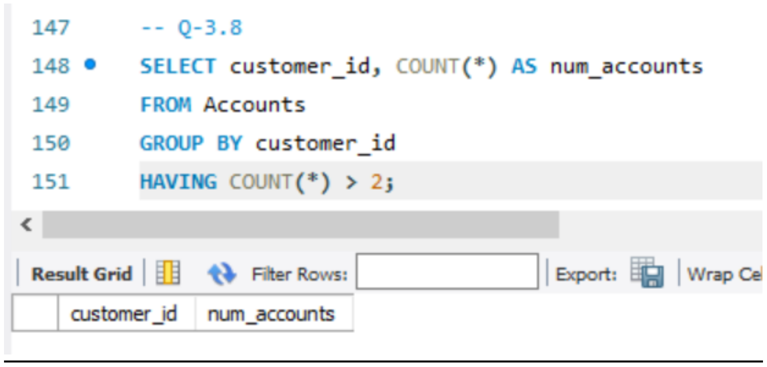
```sql
-- Q-3.6
select c.first_name,c.last_name,a.* from customers c
join accounts a on c.customer_id=a.customer_id;
```

first_name	last_name	account_id	customer_id	account_type	balance
Pankaj	Prajapati	101	1	Savings	6000
Nikhil	Kumar	102	2	Current	2500
Maya	Singh	103	3	Savings	7000
Emily	Brown	104	4	Current	3200
David	Garcia	105	5	Savings	6000
Sarah	Martinez	106	6	Current	4000
Daniel	Lopez	107	7	Savings	8000
Jessica	Lee	108	8	Current	1500
Sophia	Taylor	109	9	Savings	9000
Kevin	Clark	110	10	Current	2000



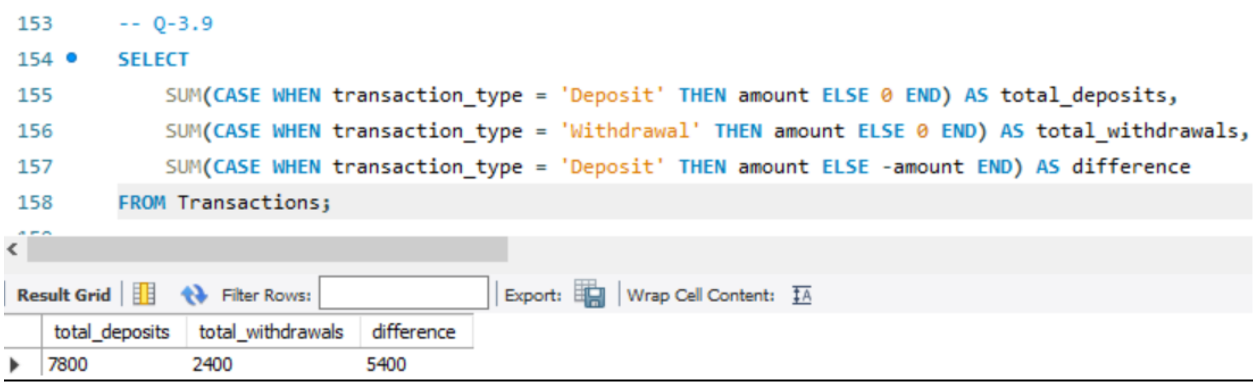
```sql
-- Q-3.7
SELECT T.*, C.*, A.account_type FROM Transactions T
JOIN Accounts A ON T.account_id = A.account_id
JOIN Customers C ON A.customer_id = C.customer_id
WHERE A.account_id = 107;
```

transaction_id	account_id	transaction_type	amount	transaction_date	customer_id	first_name	last_name	DOB	email
207	107	Deposit	2000	2023-07-10	7	Daniel	Lopez	1998-06-30	daniel@ex



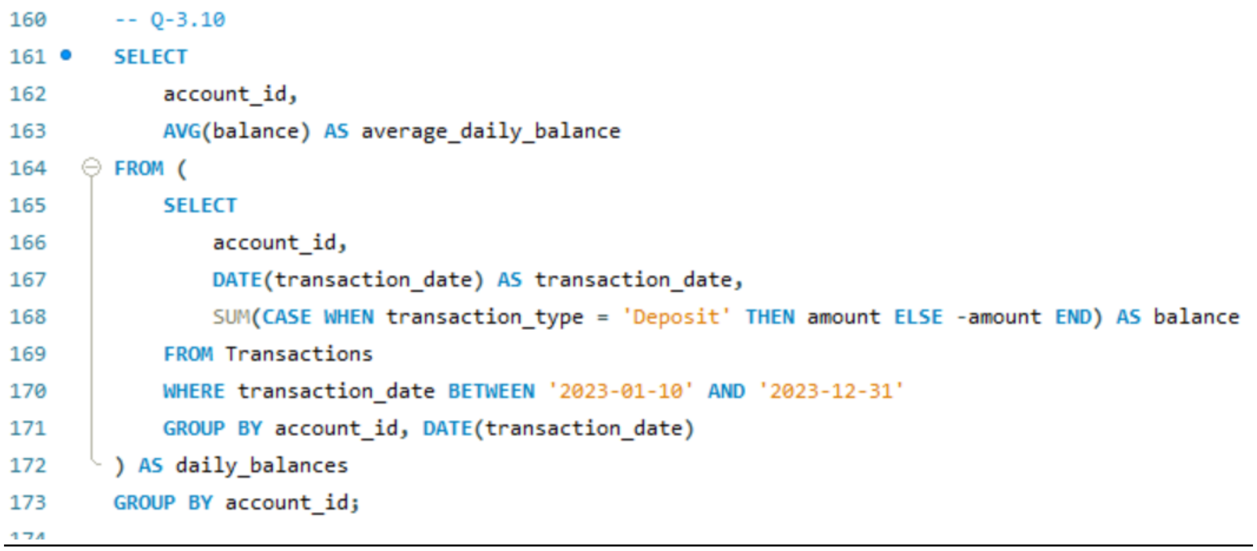
```sql
-- Q-3.8
SELECT customer_id, COUNT(*) AS num_accounts
FROM Accounts
GROUP BY customer_id
HAVING COUNT(*) > 2;
```

customer_id	num_accounts

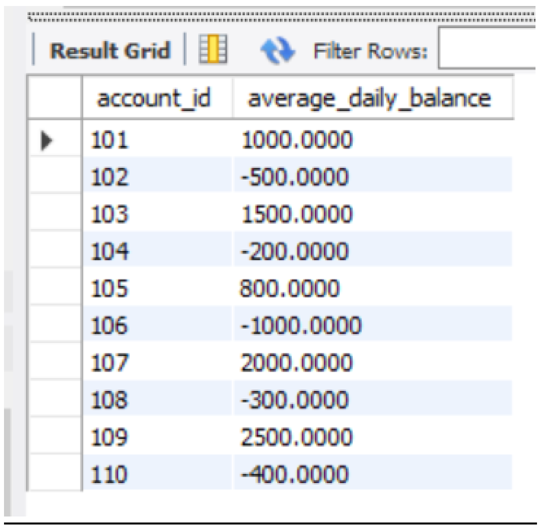


```sql
-- Q-3.9
SELECT
    SUM(CASE WHEN transaction_type = 'Deposit' THEN amount ELSE 0 END) AS total_deposits,
    SUM(CASE WHEN transaction_type = 'Withdrawal' THEN amount ELSE 0 END) AS total_withdrawals,
    SUM(CASE WHEN transaction_type = 'Deposit' THEN amount ELSE -amount END) AS difference
FROM Transactions;
```

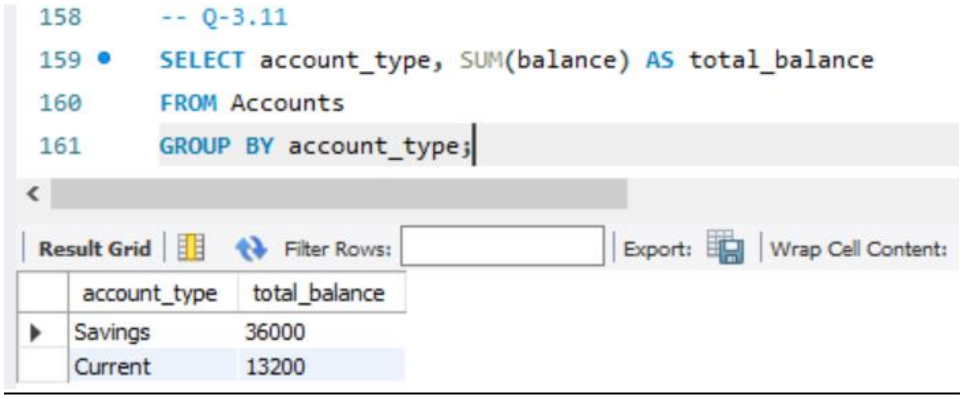
```
total_deposits  total_withdrawals  difference
7800            2400               5400
```



```sql
-- Q-3.10
SELECT
    account_id,
    AVG(balance) AS average_daily_balance
FROM (
    SELECT
        account_id,
        DATE(transaction_date) AS transaction_date,
        SUM(CASE WHEN transaction_type = 'Deposit' THEN amount ELSE -amount END) AS balance
    FROM Transactions
    WHERE transaction_date BETWEEN '2023-01-10' AND '2023-12-31'
    GROUP BY account_id, DATE(transaction_date)
) AS daily_balances
GROUP BY account_id;
```

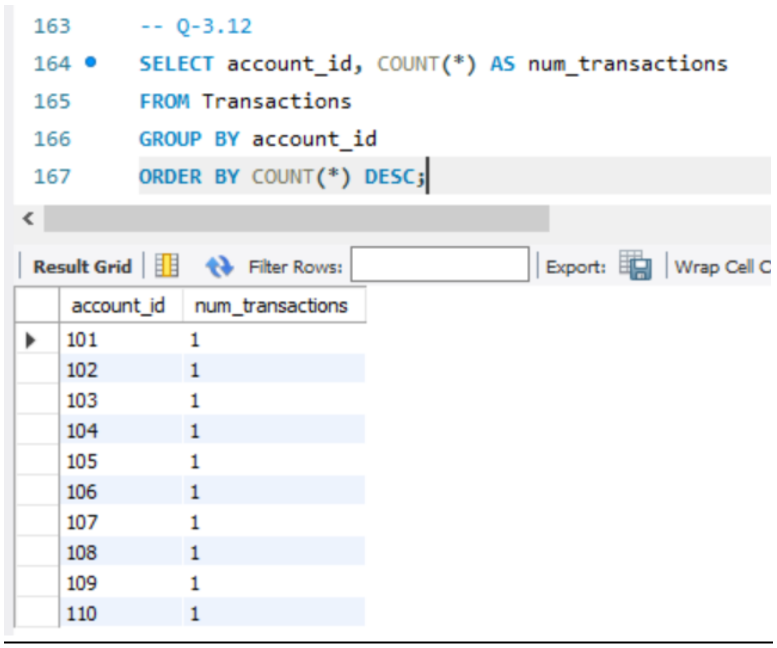


```
account_id  average_daily_balance
101         1000.0000
102         -500.0000
103         1500.0000
104         -200.0000
105         800.0000
106         -1000.0000
107         2000.0000
108         -300.0000
109         2500.0000
110         -400.0000
```



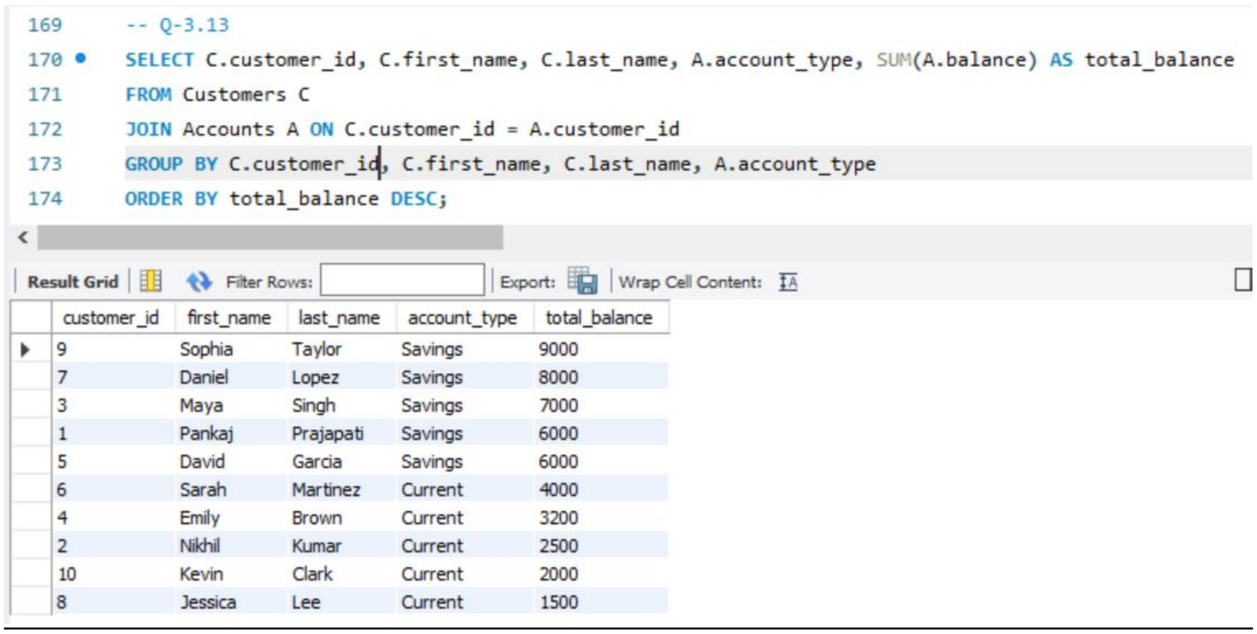
```sql
-- Q-3.11
SELECT account_type, SUM(balance) AS total_balance
FROM Accounts
GROUP BY account_type;
```

account_type	total_balance
Savings	36000
Current	13200



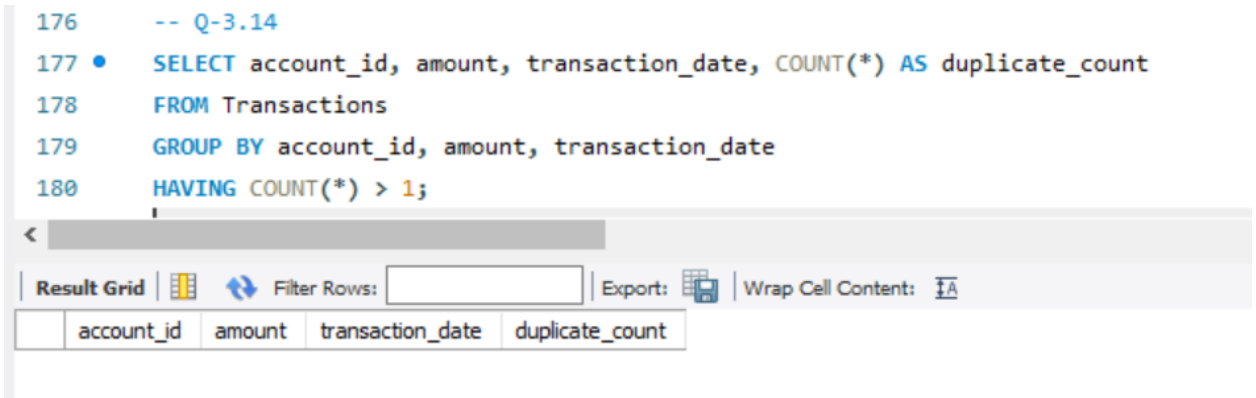
```sql
-- Q-3.12
SELECT account_id, COUNT(*) AS num_transactions
FROM Transactions
GROUP BY account_id
ORDER BY COUNT(*) DESC;
```

account_id	num_transactions
101	1
102	1
103	1
104	1
105	1
106	1
107	1
108	1
109	1
110	1



```sql
-- Q-3.13
SELECT C.customer_id, C.first_name, C.last_name, A.account_type, SUM(A.balance) AS total_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, A.account_type
ORDER BY total_balance DESC;
```

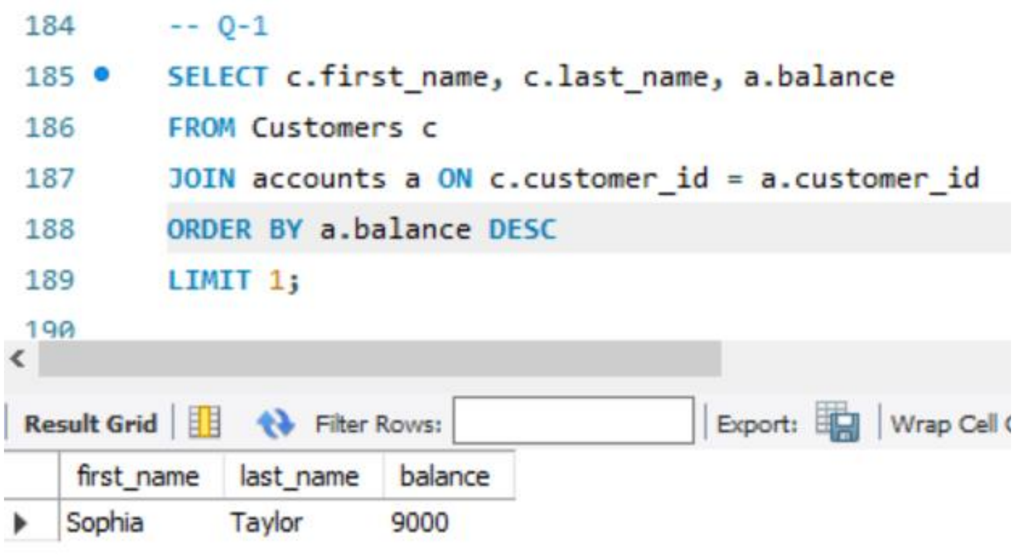
customer_id	first_name	last_name	account_type	total_balance
9	Sophia	Taylor	Savings	9000
7	Daniel	Lopez	Savings	8000
3	Maya	Singh	Savings	7000
1	Pankaj	Prajapati	Savings	6000
5	David	Garcia	Savings	6000
6	Sarah	Martinez	Current	4000
4	Emily	Brown	Current	3200
2	Nikhil	Kumar	Current	2500
10	Kevin	Clark	Current	2000
8	Jessica	Lee	Current	1500


```
-- Q-3.14
SELECT account_id, amount, transaction_date, COUNT(*) AS duplicate_count
FROM Transactions
GROUP BY account_id, amount, transaction_date
HAVING COUNT(*) > 1;
```

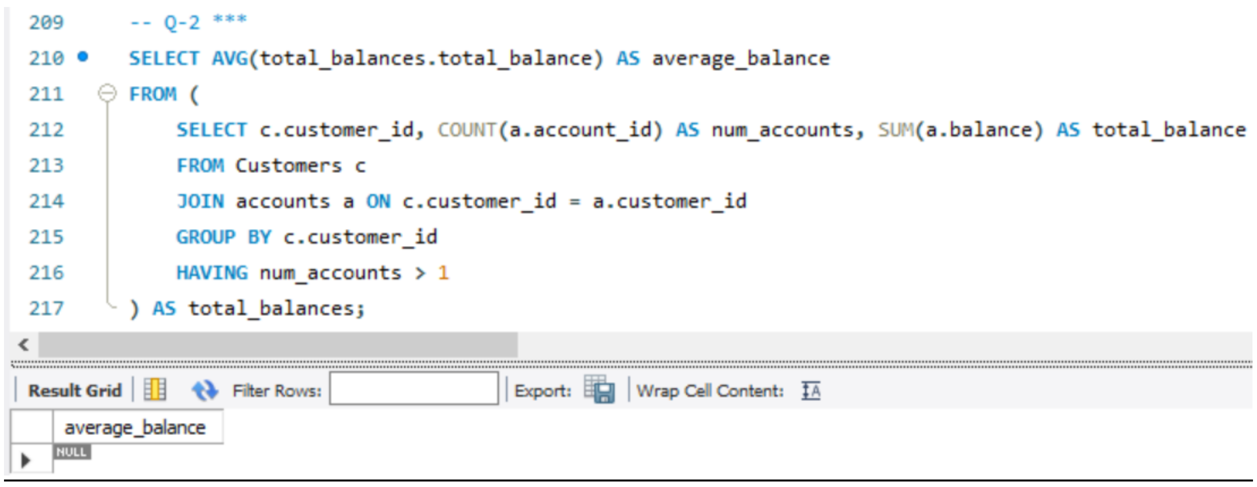
account_id	amount	transaction_date	duplicate_count

Task -4:

1. Retrieve the customer(s) with the highest account balance.
2. Calculate the average account balance for customers who have more than one account.
3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.
4. Identify customers who have no recorded transactions.
5. Calculate the total balance of accounts with no recorded transactions.
6. Retrieve transactions for accounts with the lowest balance.
7. Identify customers who have accounts of multiple types.
8. Calculate the percentage of each account type out of the total number of accounts.
9. Retrieve all transactions for a customer with a given customer_id.
10. Calculate the total balance for each account type, including a subquery within the SELECT clause.

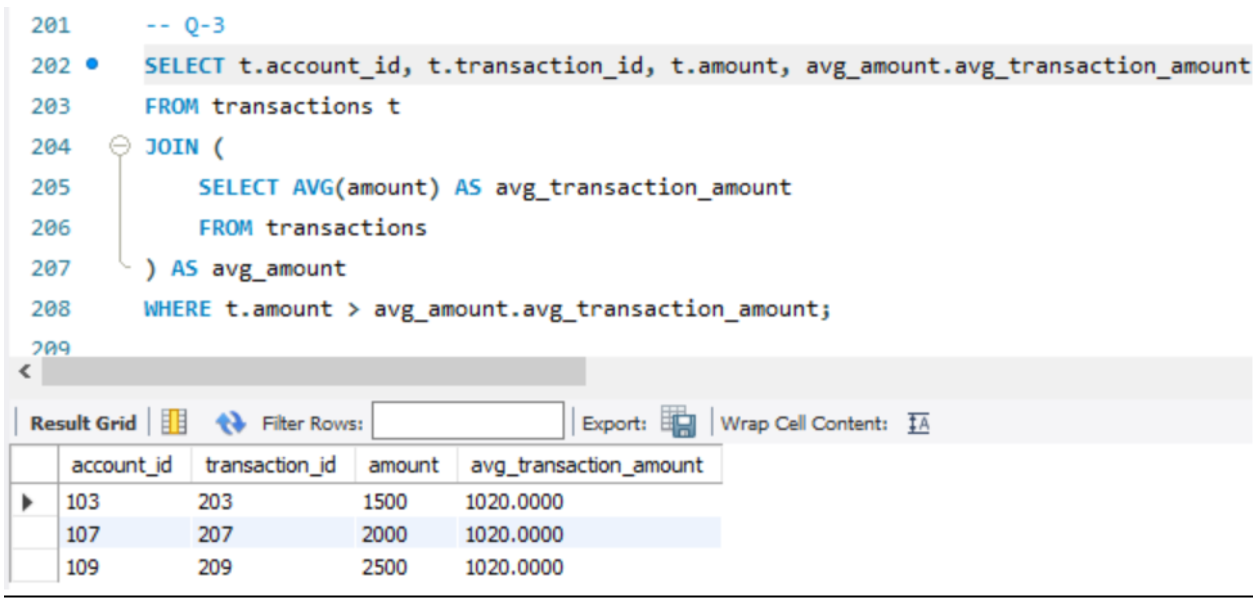

```
-- Q-1
SELECT c.first_name, c.last_name, a.balance
FROM Customers c
JOIN accounts a ON c.customer_id = a.customer_id
ORDER BY a.balance DESC
LIMIT 1;
```

first_name	last_name	balance
Sophia	Taylor	9000



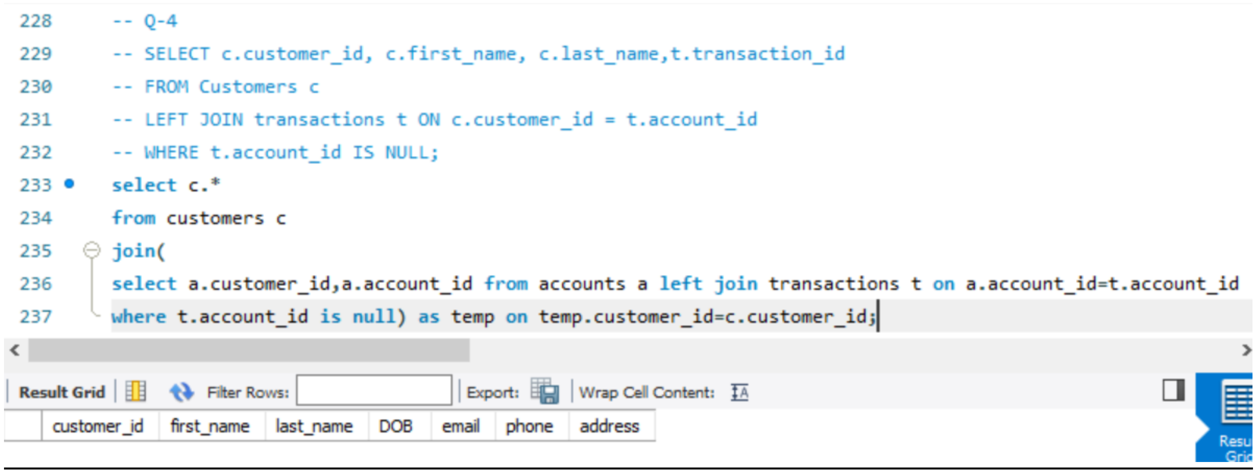
```sql
-- Q-2 ***
SELECT AVG(total_balances.total_balance) AS average_balance
FROM (
    SELECT c.customer_id, COUNT(a.account_id) AS num_accounts, SUM(a.balance) AS total_balance
    FROM Customers c
    JOIN accounts a ON c.customer_id = a.customer_id
    GROUP BY c.customer_id
    HAVING num_accounts > 1
) AS total_balances;
```

average_balance
NULL



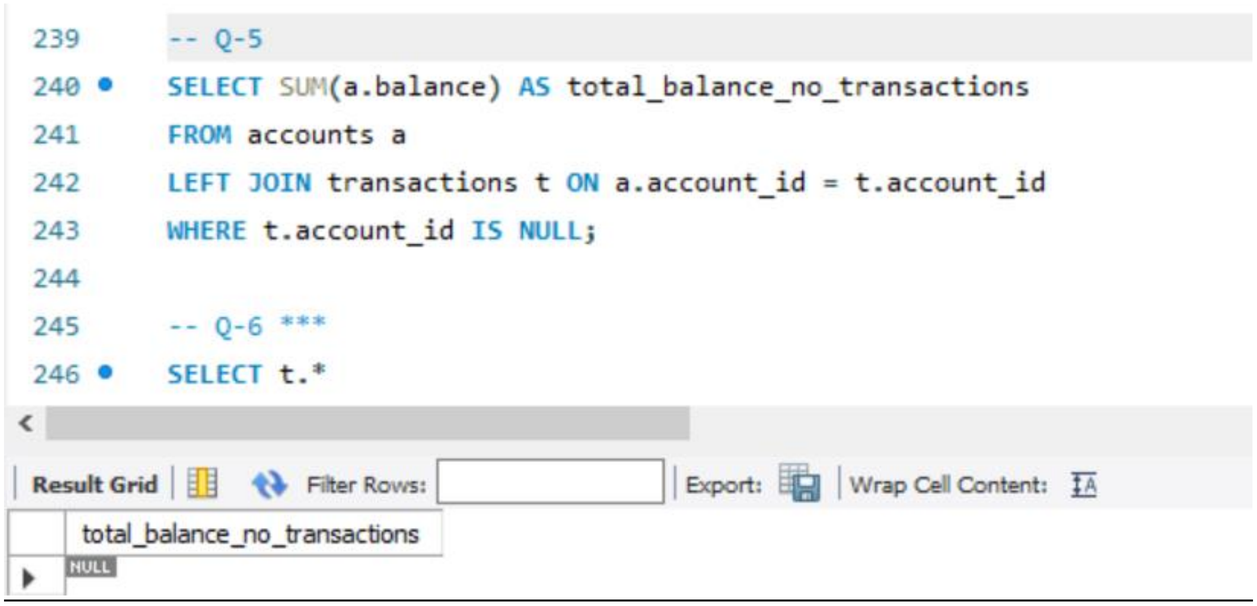
```sql
-- Q-3
SELECT t.account_id, t.transaction_id, t.amount, avg_amount.avg_transaction_amount
FROM transactions t
JOIN (
    SELECT AVG(amount) AS avg_transaction_amount
    FROM transactions
) AS avg_amount
WHERE t.amount > avg_amount.avg_transaction_amount;
```

account_id	transaction_id	amount	avg_transaction_amount
103	203	1500	1020.0000
107	207	2000	1020.0000
109	209	2500	1020.0000



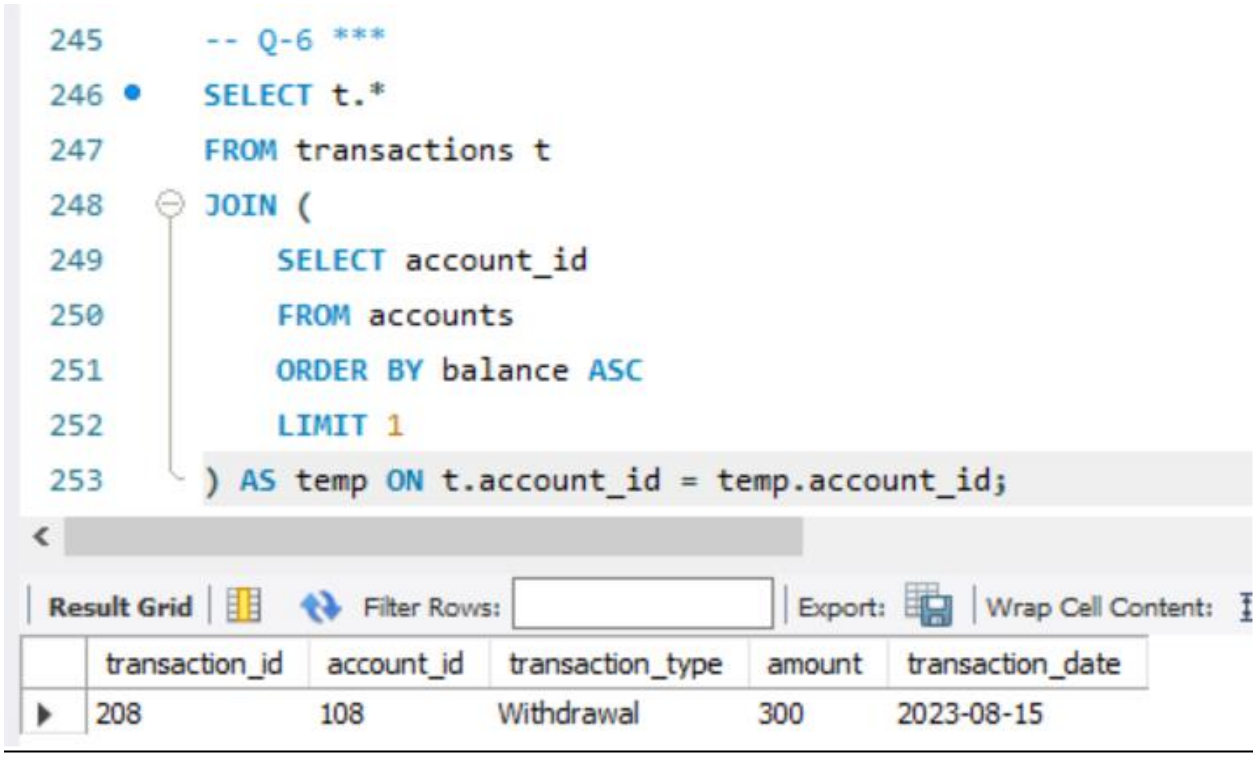
```sql
-- Q-4
-- SELECT c.customer_id, c.first_name, c.last_name,t.transaction_id
-- FROM Customers c
-- LEFT JOIN transactions t ON c.customer_id = t.account_id
-- WHERE t.account_id IS NULL;
select c.*
from customers c
join(
select a.customer_id,a.account_id from accounts a left join transactions t on a.account_id=t.account_id
where t.account_id is null) as temp on temp.customer_id=c.customer_id;
```

customer_id	first_name	last_name	DOB	email	phone	address



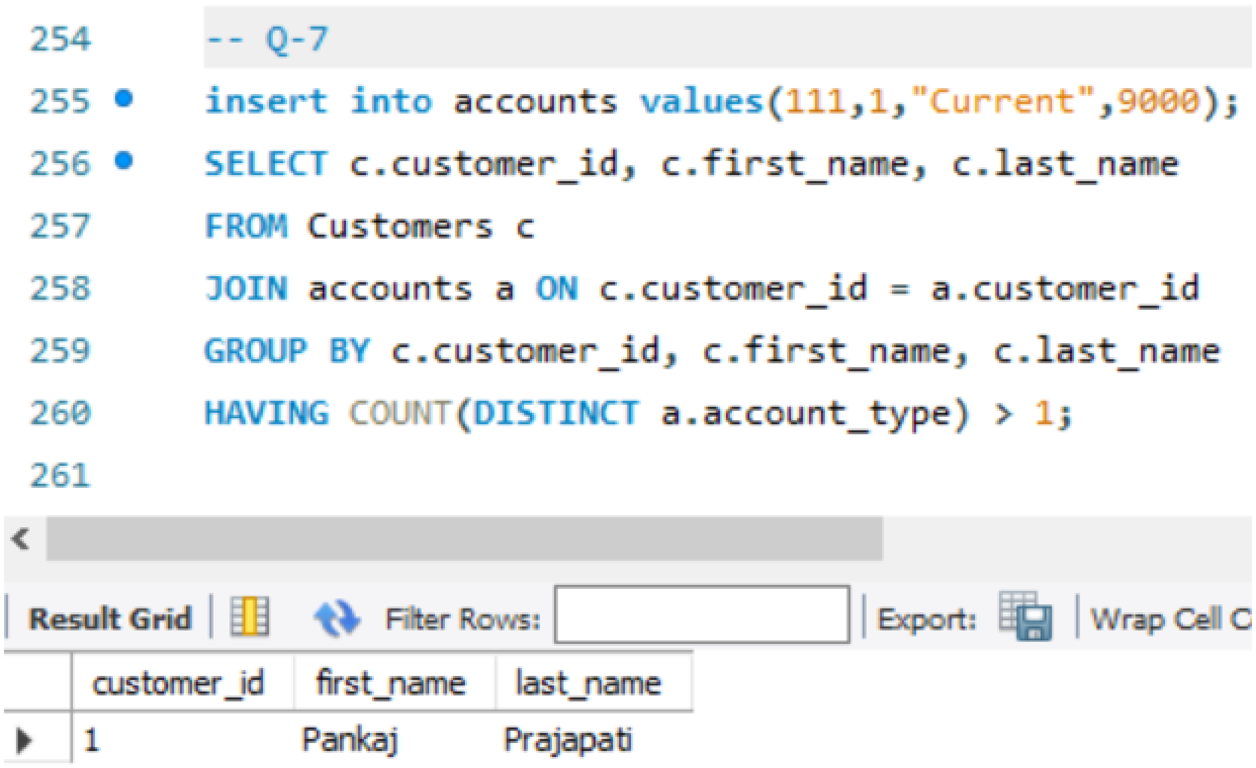
```sql
-- Q-5
SELECT SUM(a.balance) AS total_balance_no_transactions
FROM accounts a
LEFT JOIN transactions t ON a.account_id = t.account_id
WHERE t.account_id IS NULL;

-- Q-6 ***
SELECT t.*
```

total_balance_no_transactions
NULL



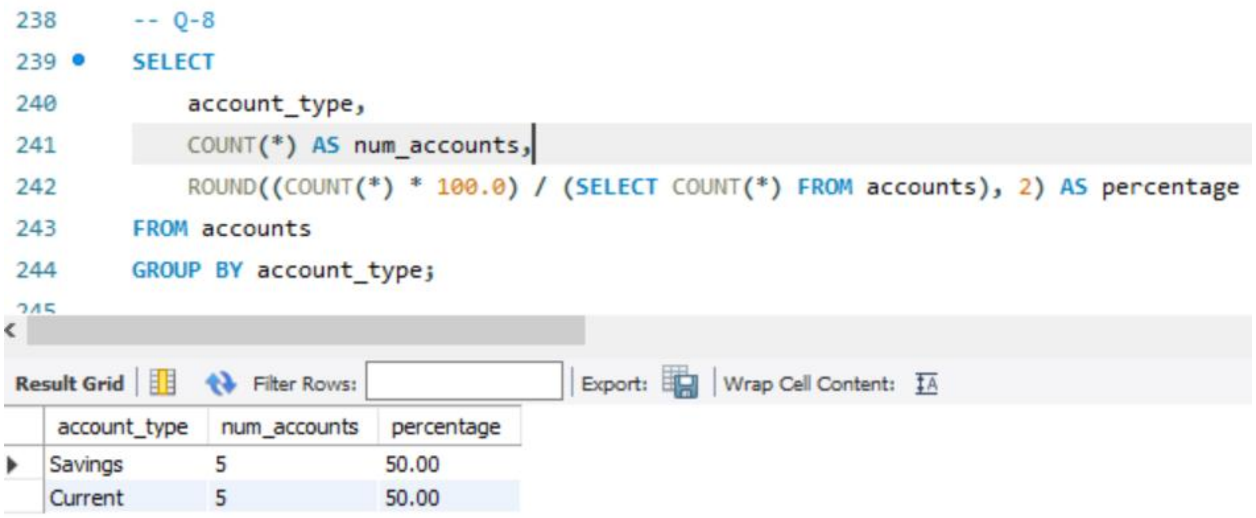
```sql
-- Q-6 ***
SELECT t.*
FROM transactions t
JOIN (
    SELECT account_id
    FROM accounts
    ORDER BY balance ASC
    LIMIT 1
) AS temp ON t.account_id = temp.account_id;
```

transaction_id	account_id	transaction_type	amount	transaction_date
208	108	Withdrawal	300	2023-08-15



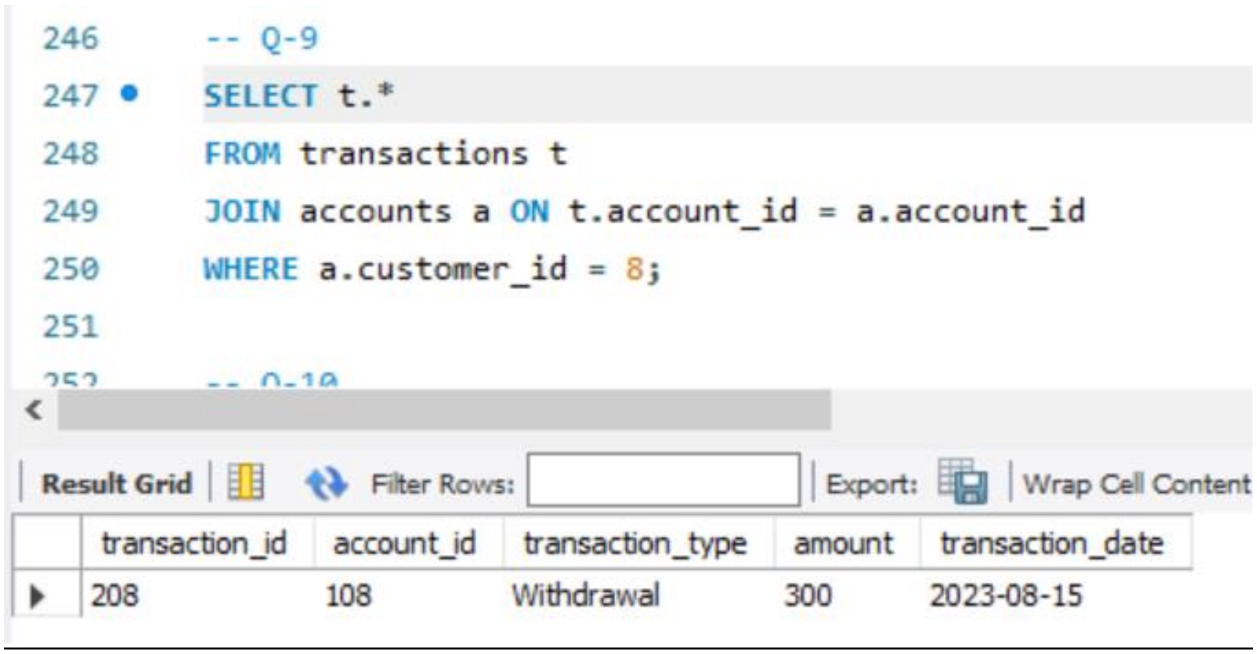
```sql
-- Q-7
insert into accounts values(111,1,"Current",9000);
SELECT c.customer_id, c.first_name, c.last_name
FROM Customers c
JOIN accounts a ON c.customer_id = a.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name
HAVING COUNT(DISTINCT a.account_type) > 1;
```

customer_id	first_name	last_name
1	Pankaj	Prajapati



```sql
-- Q-8
SELECT
    account_type,
    COUNT(*) AS num_accounts,
    ROUND((COUNT(*) * 100.0) / (SELECT COUNT(*) FROM accounts), 2) AS percentage
FROM accounts
GROUP BY account_type;
```

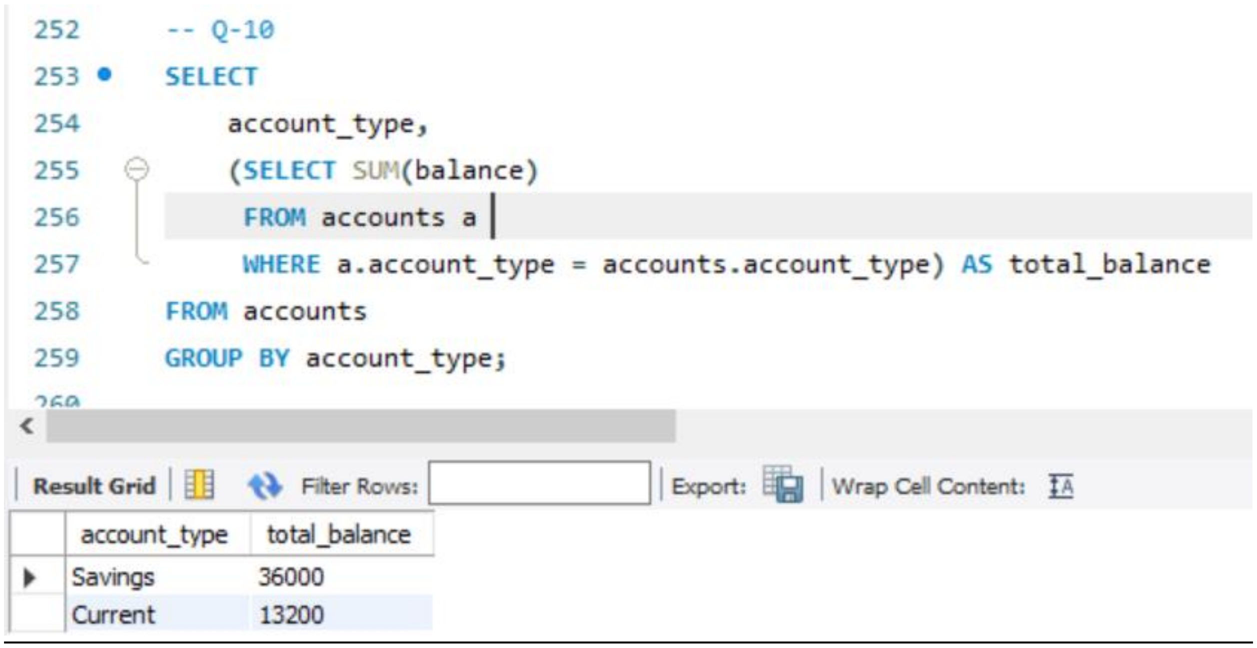
account_type	num_accounts	percentage
Savings	5	50.00
Current	5	50.00



```
-- Q-9
SELECT t.*
FROM transactions t
JOIN accounts a ON t.account_id = a.account_id
WHERE a.customer_id = 8;

-- Q-10
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content

transaction_id	account_id	transaction_type	amount	transaction_date
208	108	Withdrawal	300	2023-08-15



```
-- Q-10
SELECT
    account_type,
    (SELECT SUM(balance)
     FROM accounts a
     WHERE a.account_type = accounts.account_type) AS total_balance
FROM accounts
GROUP BY account_type;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

account_type	total_balance
Savings	36000
Current	13200