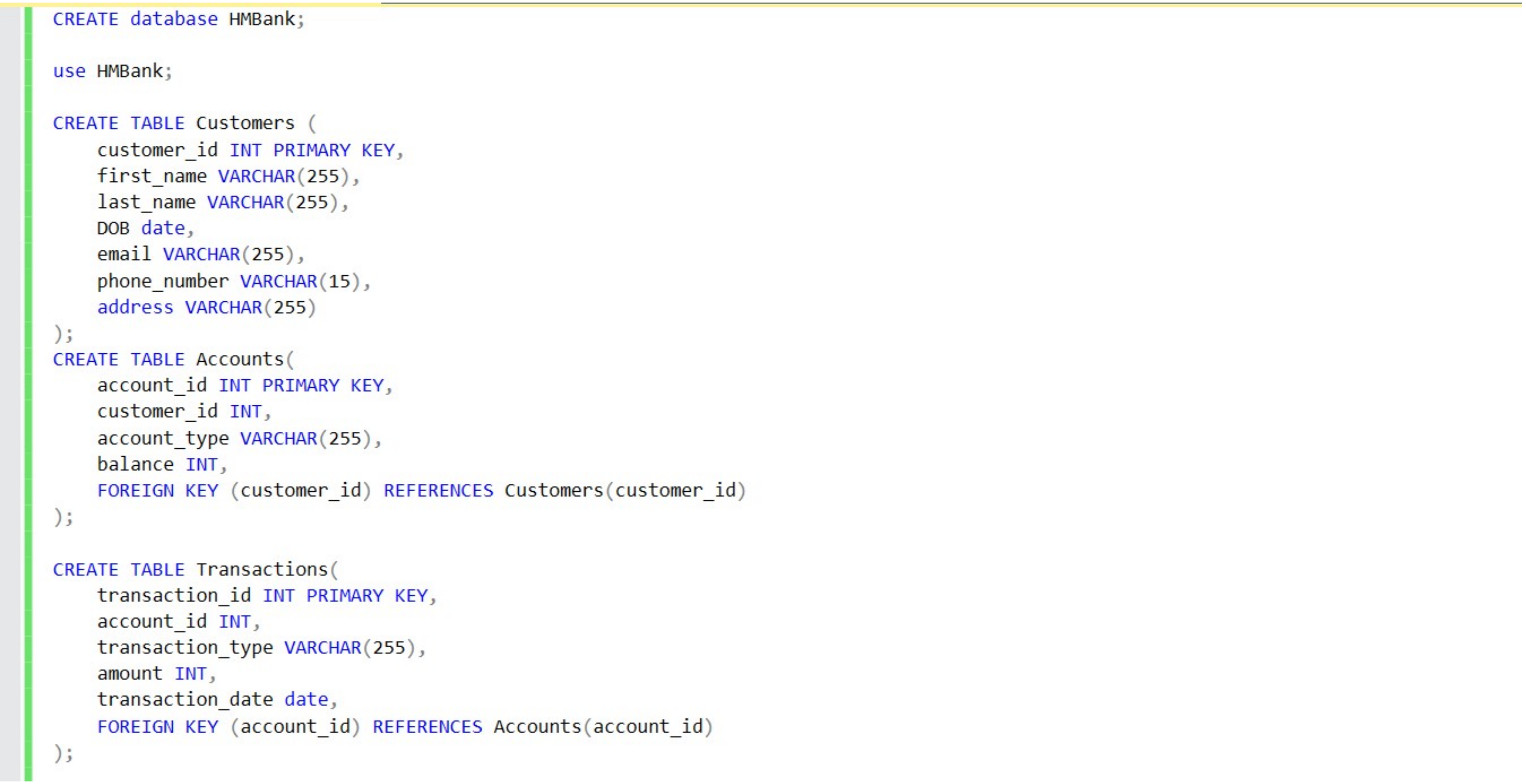
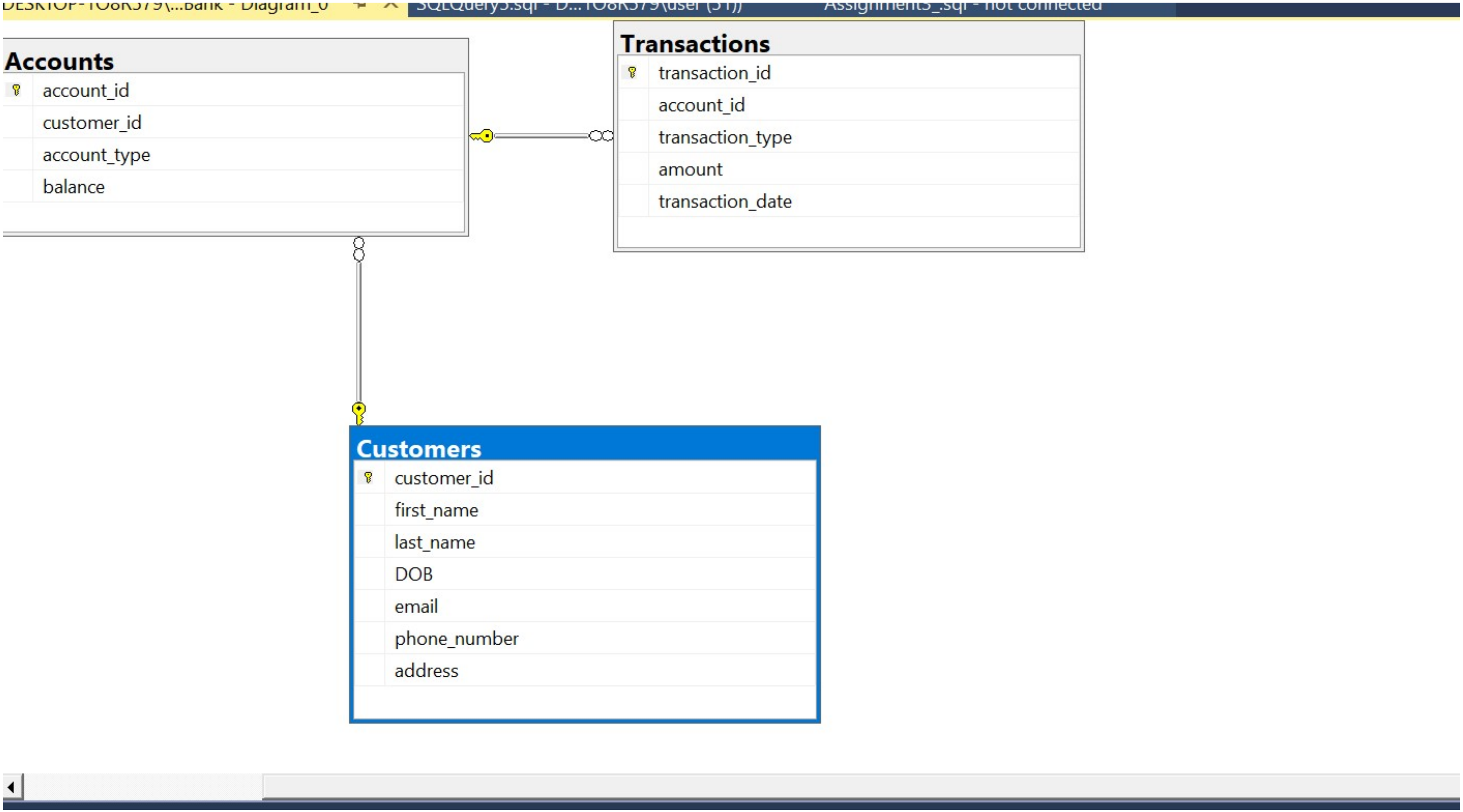
Tasks 1: Database Design:

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



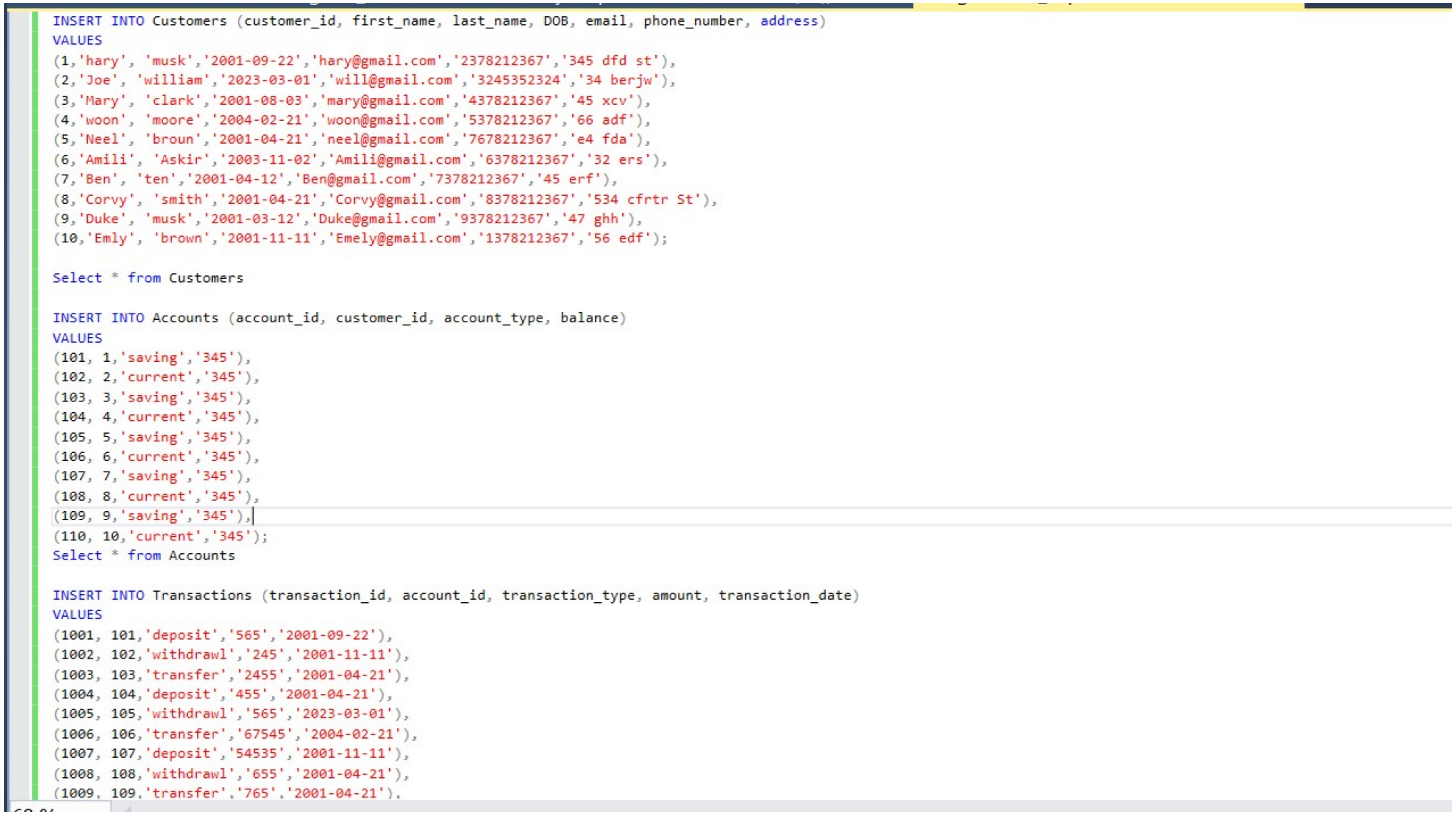
```sql
CREATE database HMBank;

use HMBank;

CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    DOB date,
    email VARCHAR(255),
    phone_number VARCHAR(15),
    address VARCHAR(255)
);
CREATE TABLE Accounts(
    account_id INT PRIMARY KEY,
    customer_id INT,
    account_type VARCHAR(255),
    balance INT,
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

CREATE TABLE Transactions(
    transaction_id INT PRIMARY KEY,
    account_id INT,
    transaction_type VARCHAR(255),
    amount INT,
    transaction_date date,
    FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
```



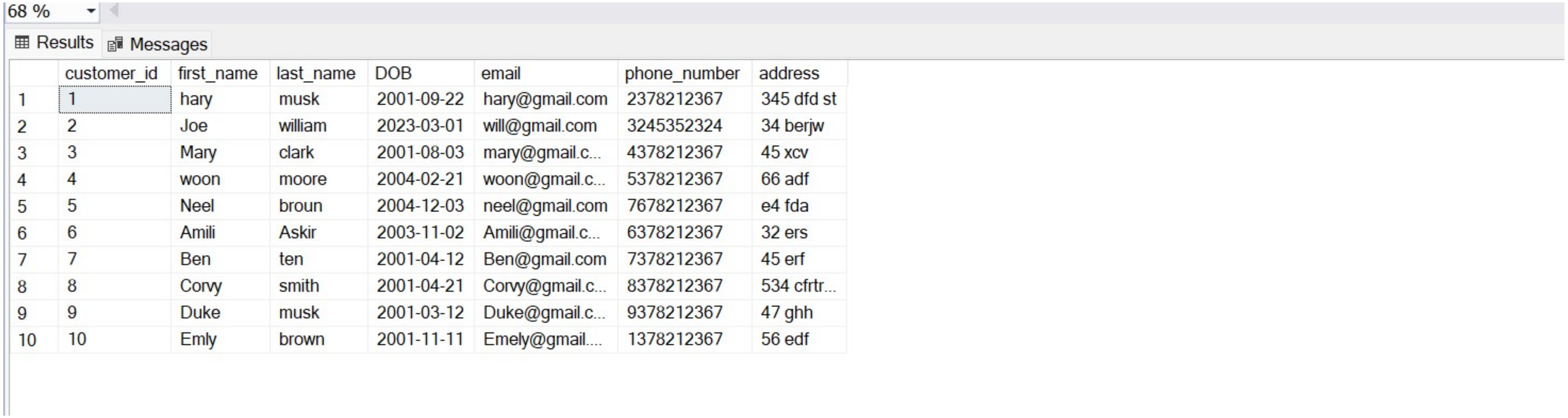
Tasks 2: Select, Where, Between, AND, LIKE:

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

- Customers
- Accounts
- Transactions



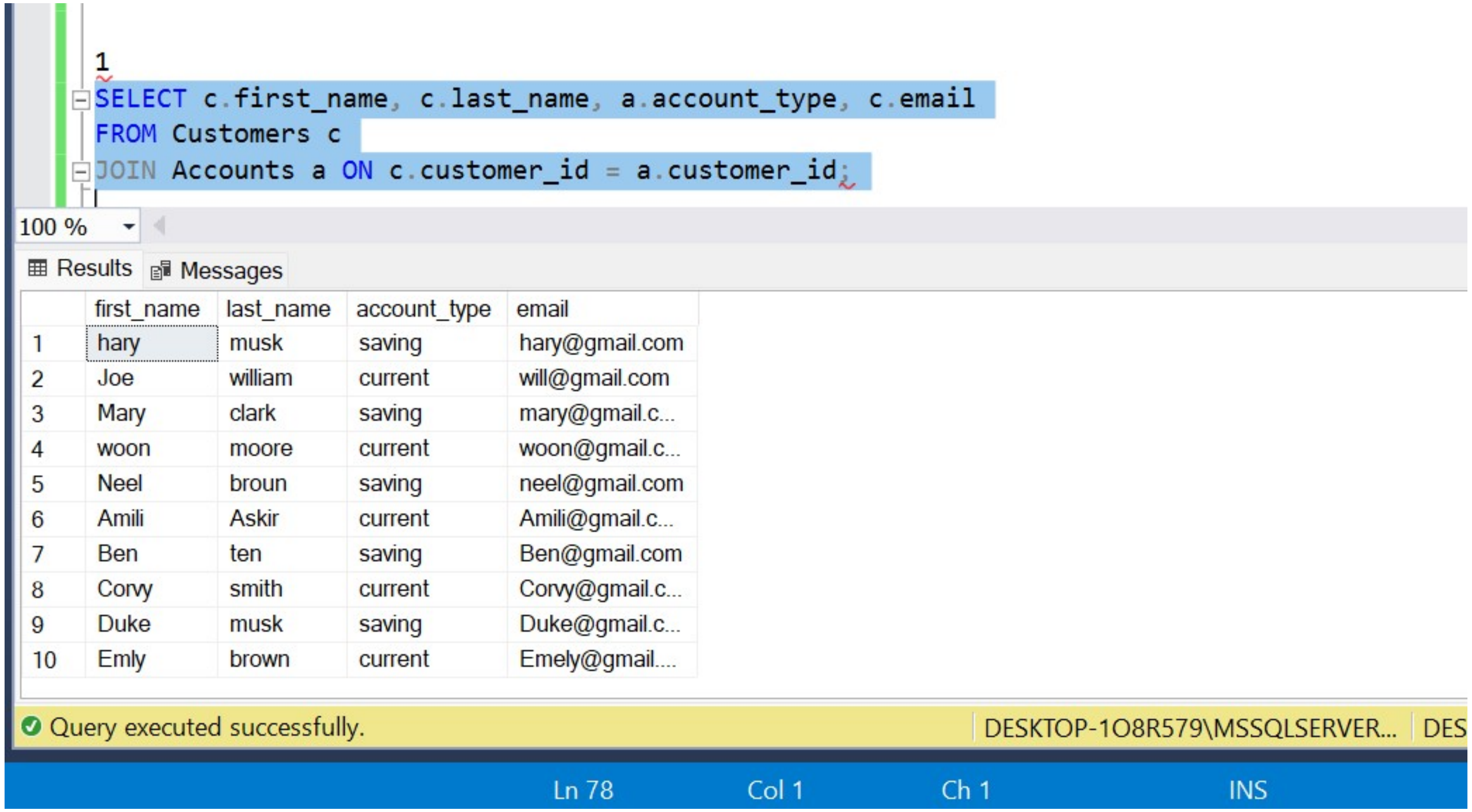
```sql
INSERT INTO Customers (customer_id, first_name, last_name, DOB, email, phone_number, address)
VALUES
(1,'hary', 'musk','2001-09-22','hary@gmail.com','2378212367','345 dfd st'),
(2,'Joe', 'william','2023-03-01','will@gmail.com','3245352324','34 berjw'),
(3,'Mary', 'clark','2001-08-03','mary@gmail.com','4378212367','45 xcv'),
(4,'woon', 'moore','2004-02-21','woon@gmail.com','5378212367','66 adf'),
(5,'Neel', 'broun','2001-04-21','neel@gmail.com','7678212367','e4 fda'),
(6,'Amili', 'Askir','2003-11-02','Amili@gmail.com','6378212367','32 ers'),
(7,'Ben', 'ten','2001-04-12','Ben@gmail.com','7378212367','45 erf'),
(8,'Corvy', 'smith','2001-04-21','Corvy@gmail.com','8378212367','534 cfrtr St'),
(9,'Duke', 'musk','2001-03-12','Duke@gmail.com','9378212367','47 ghh'),
(10,'Emly', 'brown','2001-11-11','Emely@gmail.com','1378212367','56 edf');

Select * from Customers

INSERT INTO Accounts (account_id, customer_id, account_type, balance)
VALUES
(101, 1,'saving','345'),
(102, 2,'current','345'),
(103, 3,'saving','345'),
(104, 4,'current','345'),
(105, 5,'saving','345'),
(106, 6,'current','345'),
(107, 7,'saving','345'),
(108, 8,'current','345'),
(109, 9,'saving','345'),
(110, 10,'current','345');
Select * from Accounts

INSERT INTO Transactions (transaction_id, account_id, transaction_type, amount, transaction_date)
VALUES
(1001, 101,'deposit','565','2001-09-22'),
(1002, 102,'withdrawl','245','2001-11-11'),
(1003, 103,'transfer','2455','2001-04-21'),
(1004, 104,'deposit','455','2001-04-21'),
(1005, 105,'withdrawl','565','2023-03-01'),
(1006, 106,'transfer','67545','2004-02-21'),
(1007, 107,'deposit','54535','2001-11-11'),
(1008, 108,'withdrawl','655','2001-04-21'),
(1009, 109,'transfer','765','2001-04-21'),
```



	customer_id	first_name	last_name	DOB	email	phone_number	address
1	1	hary	musk	2001-09-22	hary@gmail.com	2378212367	345 dfd st
2	2	Joe	william	2023-03-01	will@gmail.com	3245352324	34 berjw
3	3	Mary	clark	2001-08-03	mary@gmail.c...	4378212367	45 xcv
4	4	woon	moore	2004-02-21	woon@gmail.c...	5378212367	66 adf
5	5	Neel	broun	2004-12-03	neel@gmail.com	7678212367	e4 fda
6	6	Amili	Askir	2003-11-02	Amili@gmail.c...	6378212367	32 ers
7	7	Ben	ten	2001-04-12	Ben@gmail.com	7378212367	45 erf
8	8	Corvy	smith	2001-04-21	Corvy@gmail.c...	8378212367	534 cfrtr...
9	9	Duke	musk	2001-03-12	Duke@gmail.c...	9378212367	47 ghh
10	10	Emly	brown	2001-11-11	Emely@gmail....	1378212367	56 edf

Write SQL queries for the following tasks:

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



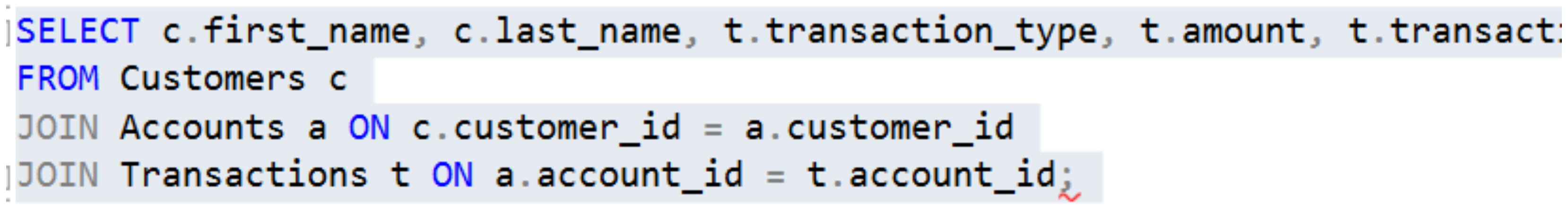
```
1
SELECT c.first_name, c.last_name, a.account_type, c.email
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id;
```

	first_name	last_name	account_type	email
1	hary	musk	saving	hary@gmail.com
2	Joe	william	current	will@gmail.com
3	Mary	clark	saving	mary@gmail.c...
4	woon	moore	current	woon@gmail.c...
5	Neel	broun	saving	neel@gmail.com
6	Amili	Askir	current	Amili@gmail.c...
7	Ben	ten	saving	Ben@gmail.com
8	Corvy	smith	current	Corvy@gmail.c...
9	Duke	musk	saving	Duke@gmail.c...
10	Emly	brown	current	Emely@gmail....

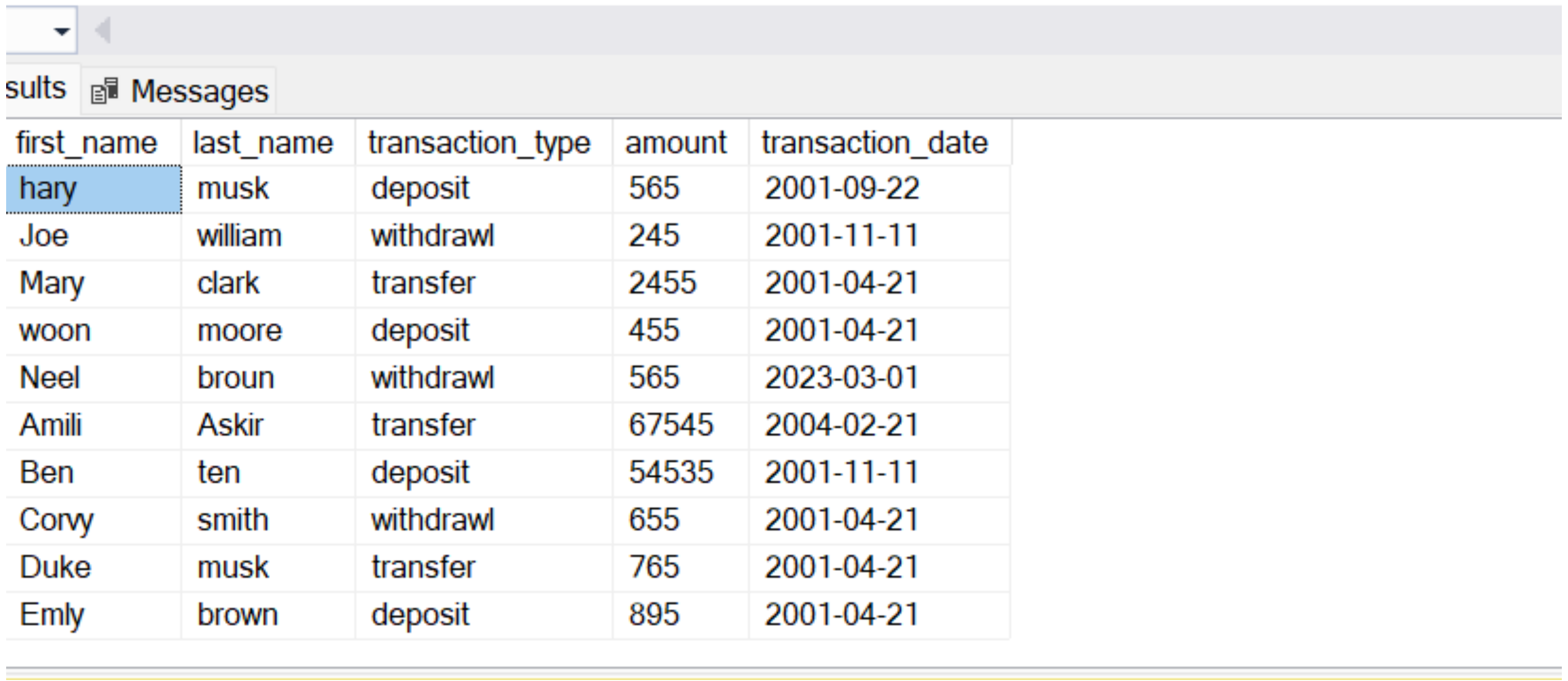
Query executed successfully.

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

2

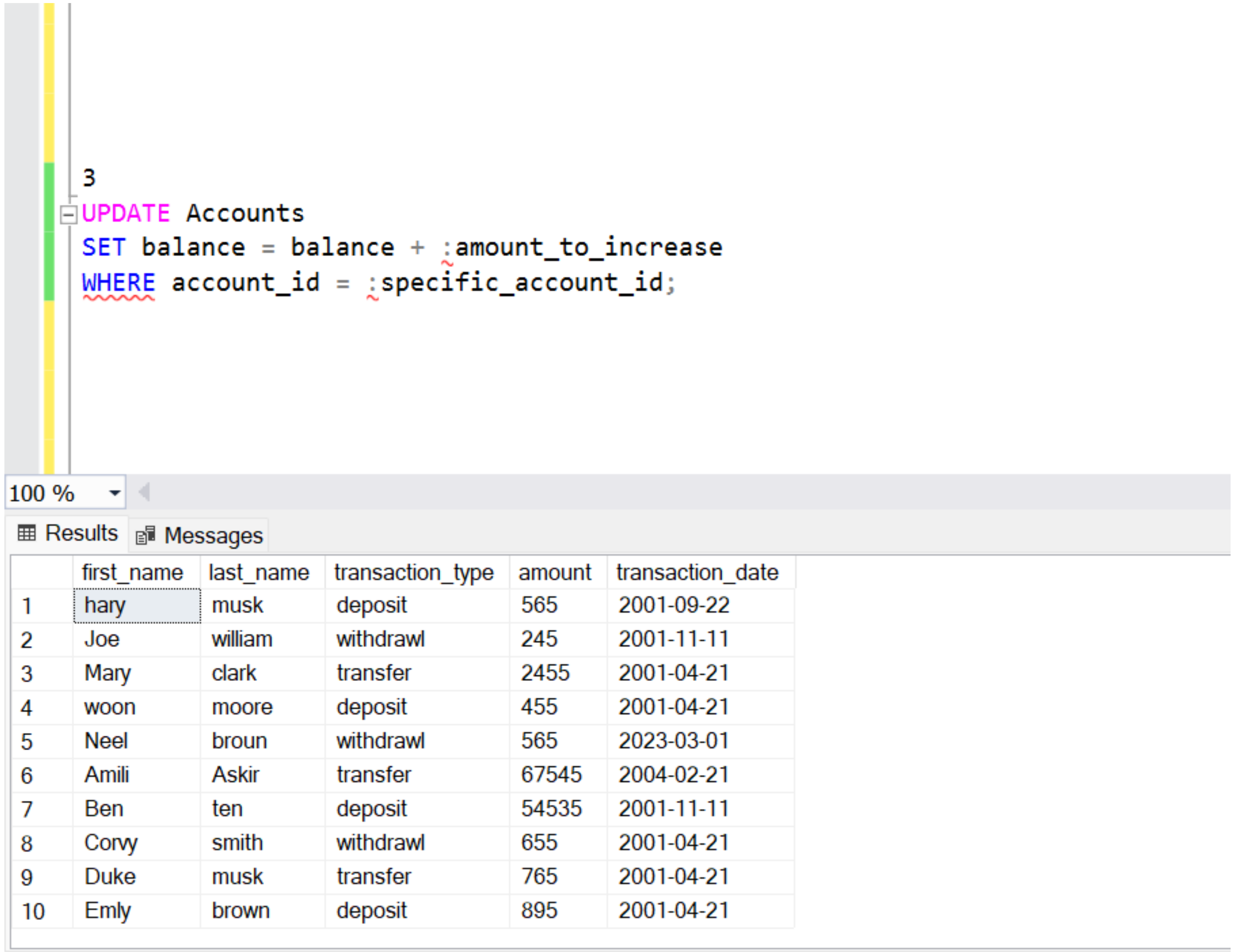


```
SELECT c.first_name, c.last_name, t.transaction_type, t.amount, t.transact:
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
JOIN Transactions t ON a.account_id = t.account_id;
```



first_name	last_name	transaction_type	amount	transaction_date
hary	musk	deposit	565	2001-09-22
Joe	william	withdrawl	245	2001-11-11
Mary	clark	transfer	2455	2001-04-21
woon	moore	deposit	455	2001-04-21
Neel	broun	withdrawl	565	2023-03-01
Amili	Askir	transfer	67545	2004-02-21
Ben	ten	deposit	54535	2001-11-11
Corvy	smith	withdrawl	655	2001-04-21
Duke	musk	transfer	765	2001-04-21
Emly	brown	deposit	895	2001-04-21

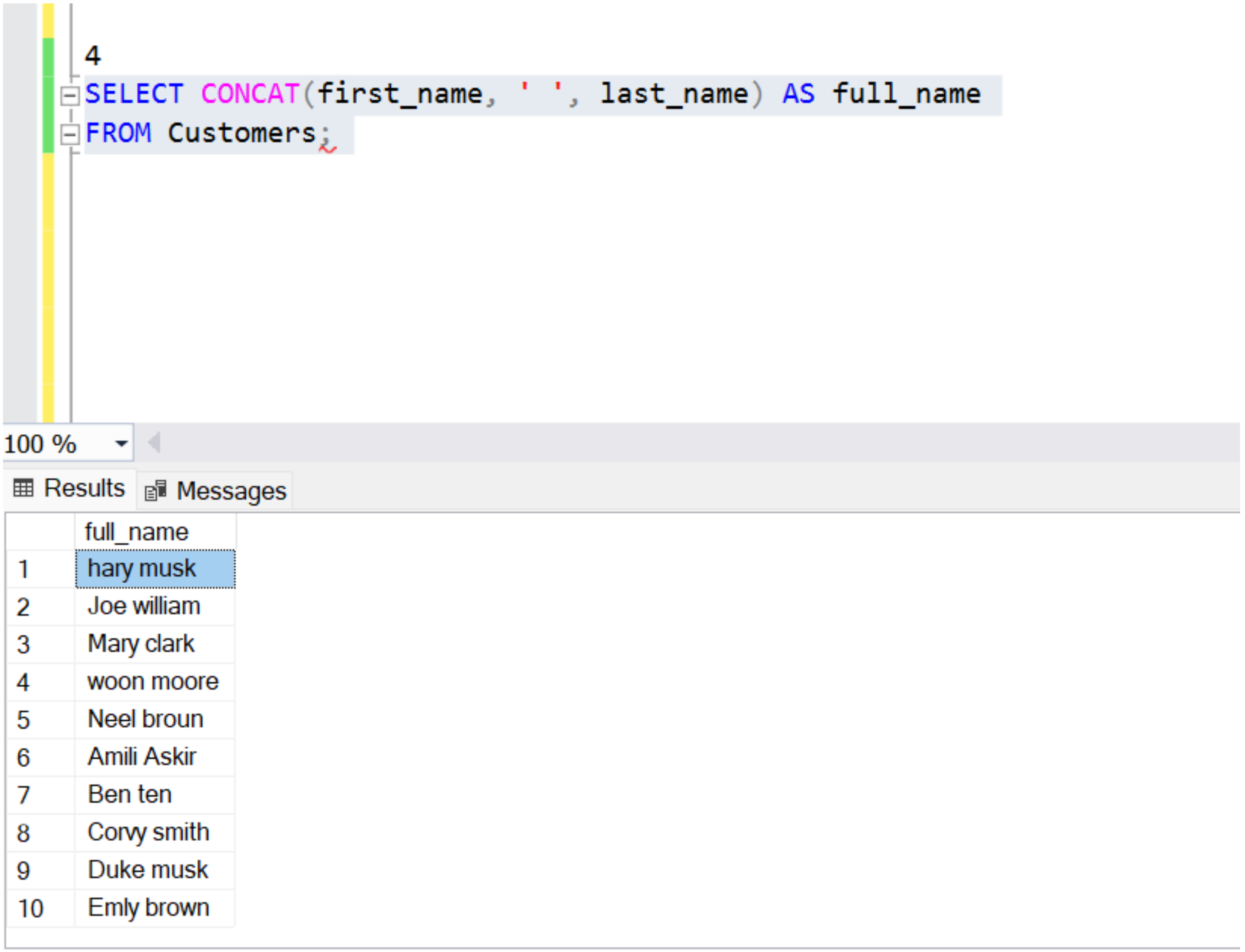
3. Write a SQL query to increase the balance of a specific account by a certain amount.



```
3
UPDATE Accounts
SET balance = balance + :amount_to_increase
WHERE account_id = :specific_account_id;
```

	first_name	last_name	transaction_type	amount	transaction_date
1	hary	musk	deposit	565	2001-09-22
2	Joe	william	withdrawl	245	2001-11-11
3	Mary	clark	transfer	2455	2001-04-21
4	woon	moore	deposit	455	2001-04-21
5	Neel	broun	withdrawl	565	2023-03-01
6	Amili	Askir	transfer	67545	2004-02-21
7	Ben	ten	deposit	54535	2001-11-11
8	Corvy	smith	withdrawl	655	2001-04-21
9	Duke	musk	transfer	765	2001-04-21
10	Emly	brown	deposit	895	2001-04-21

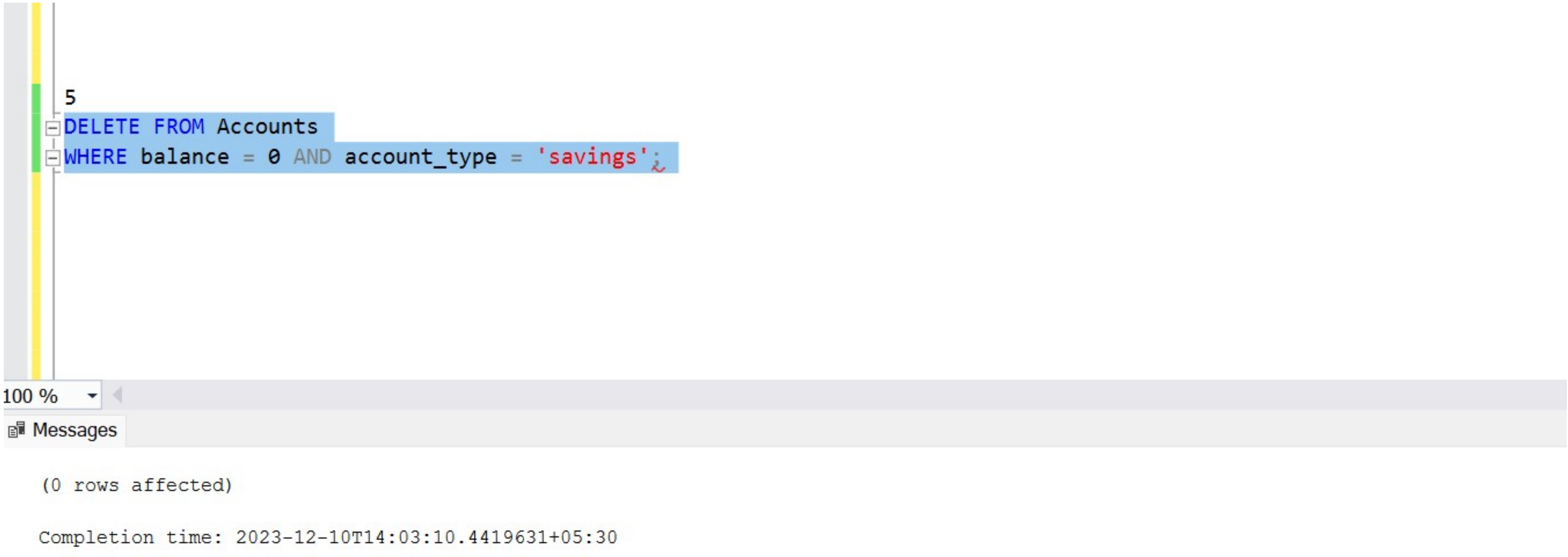
4. Write a SQL query to Combine first and last names of customers as a full_name.



```
4
SELECT CONCAT(first_name, ' ', last_name) AS full_name
FROM Customers;
```

	full_name
1	hary musk
2	Joe william
3	Mary clark
4	woon moore
5	Neel broun
6	Amili Askir
7	Ben ten
8	Corvy smith
9	Duke musk
10	Emly brown

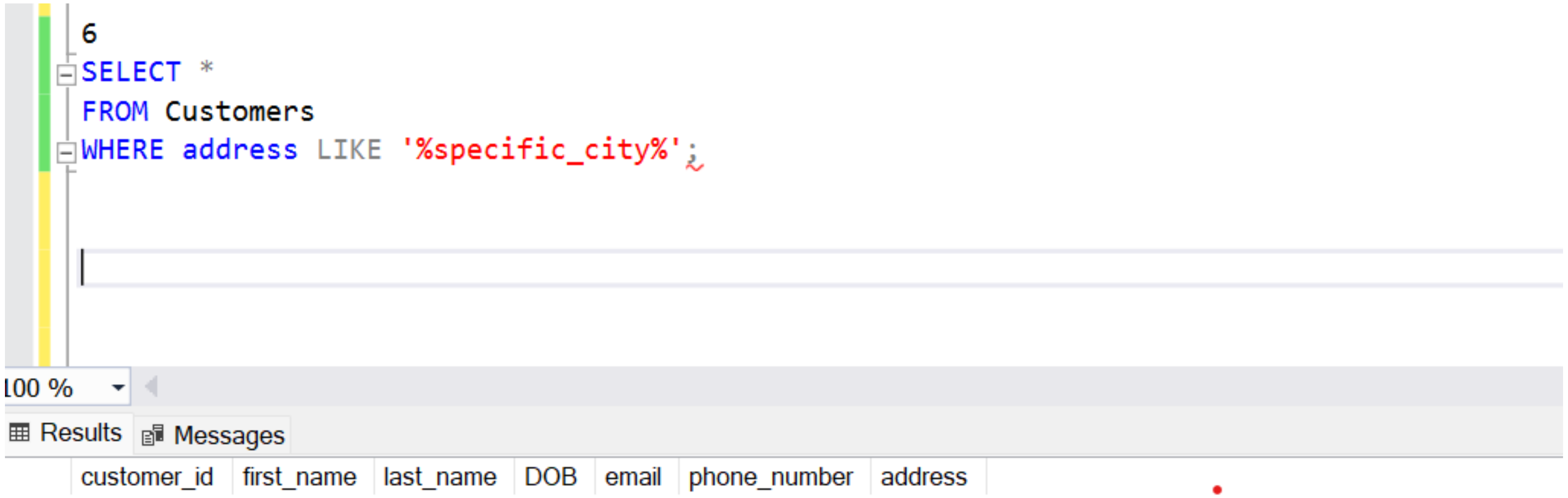
5. Write a SQL query to remove accounts with a balance of zero where the account type is savings.



```
5
DELETE FROM Accounts
WHERE balance = 0 AND account_type = 'savings';
```

```
(0 rows affected)

Completion time: 2023-12-10T14:03:10.4419631+05:30
```

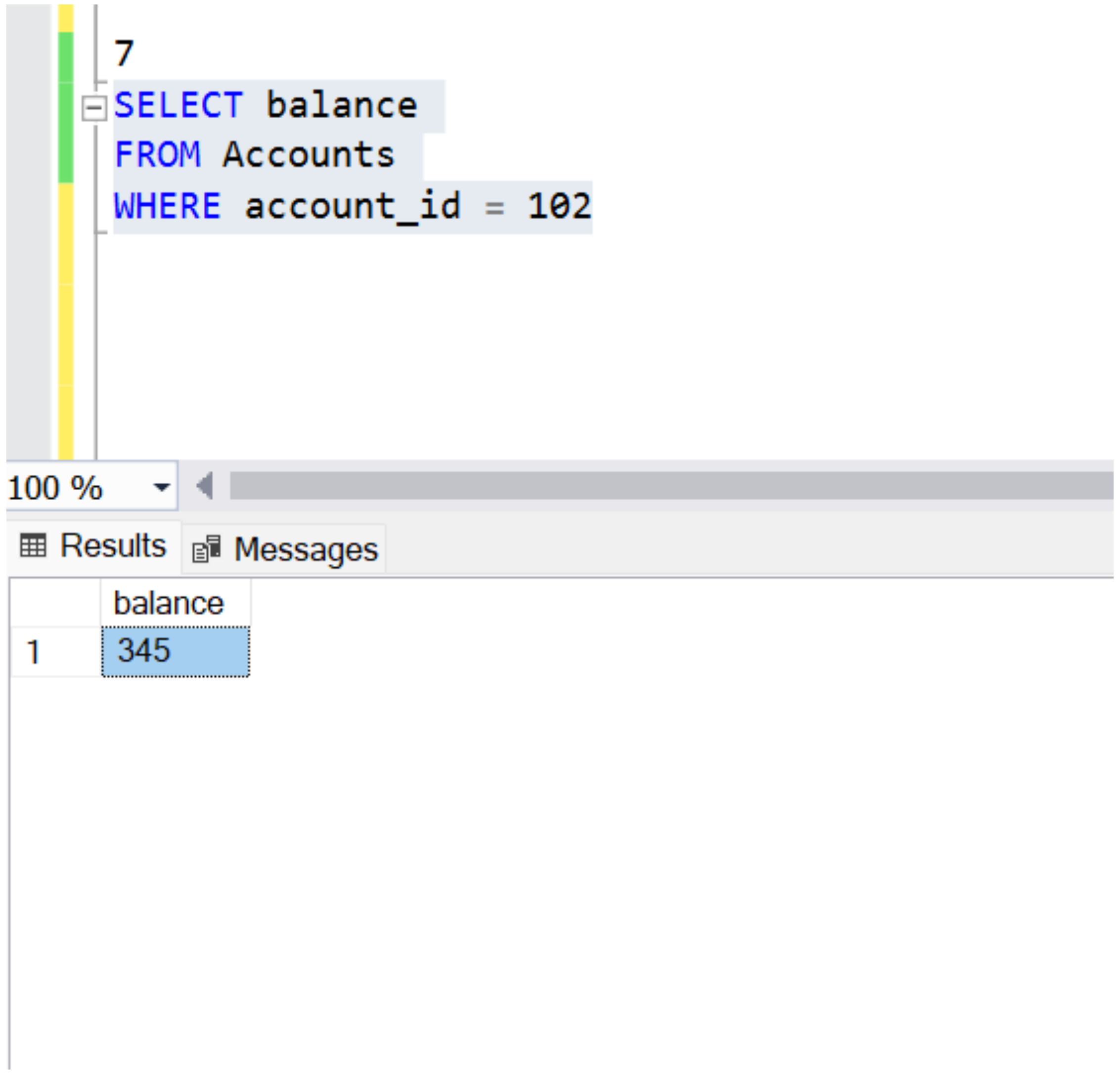
6. Write a SQL query to Find customers living in a specific city.



```
6
SELECT *
FROM Customers
WHERE address LIKE '%specific_city%';
```

customer_id	first_name	last_name	DOB	email	phone_number	address

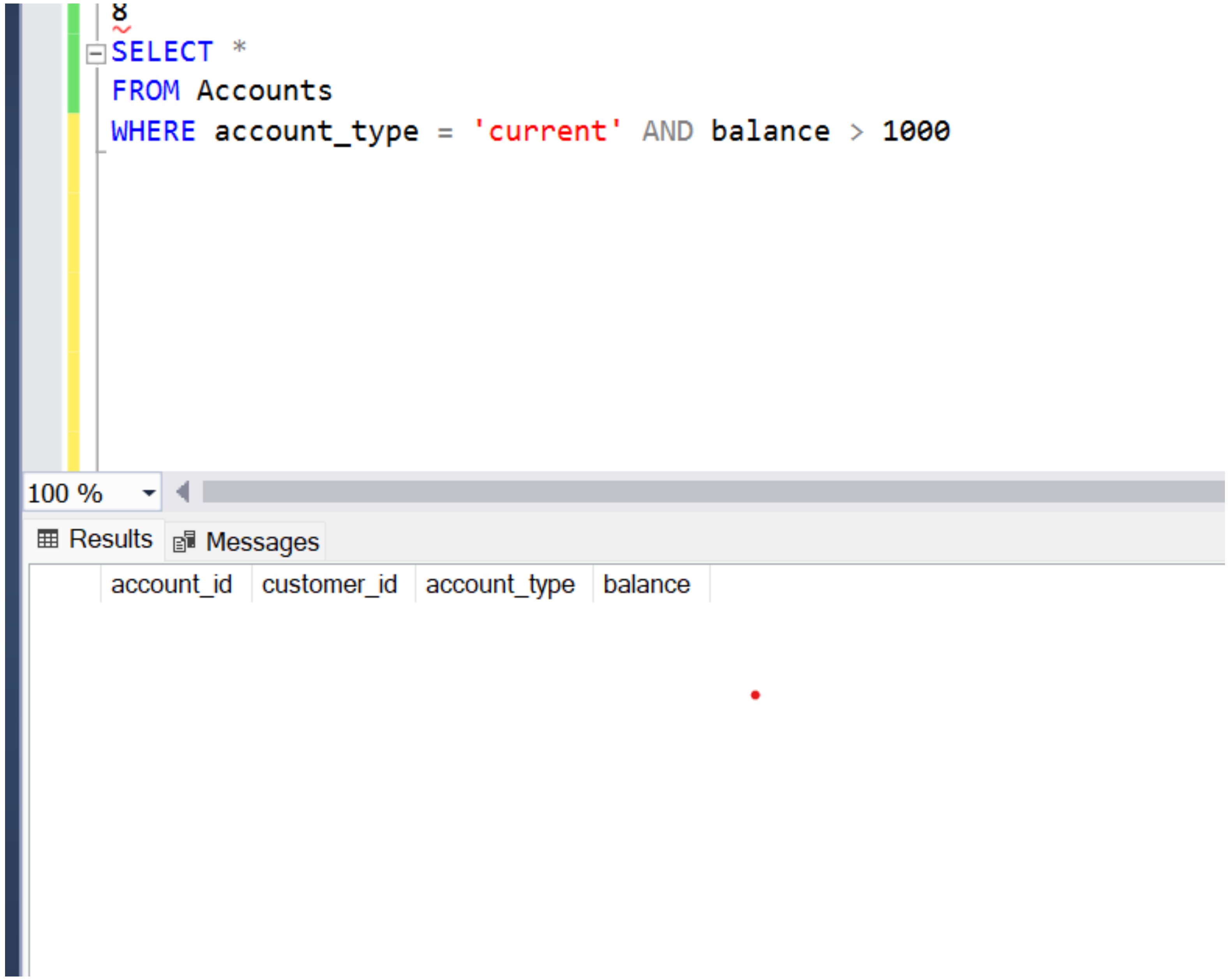
7. Write a SQL query to Get the account balance for a specific account.



```
7
SELECT balance
FROM Accounts
WHERE account_id = 102
```

```
   balance
1  345
```

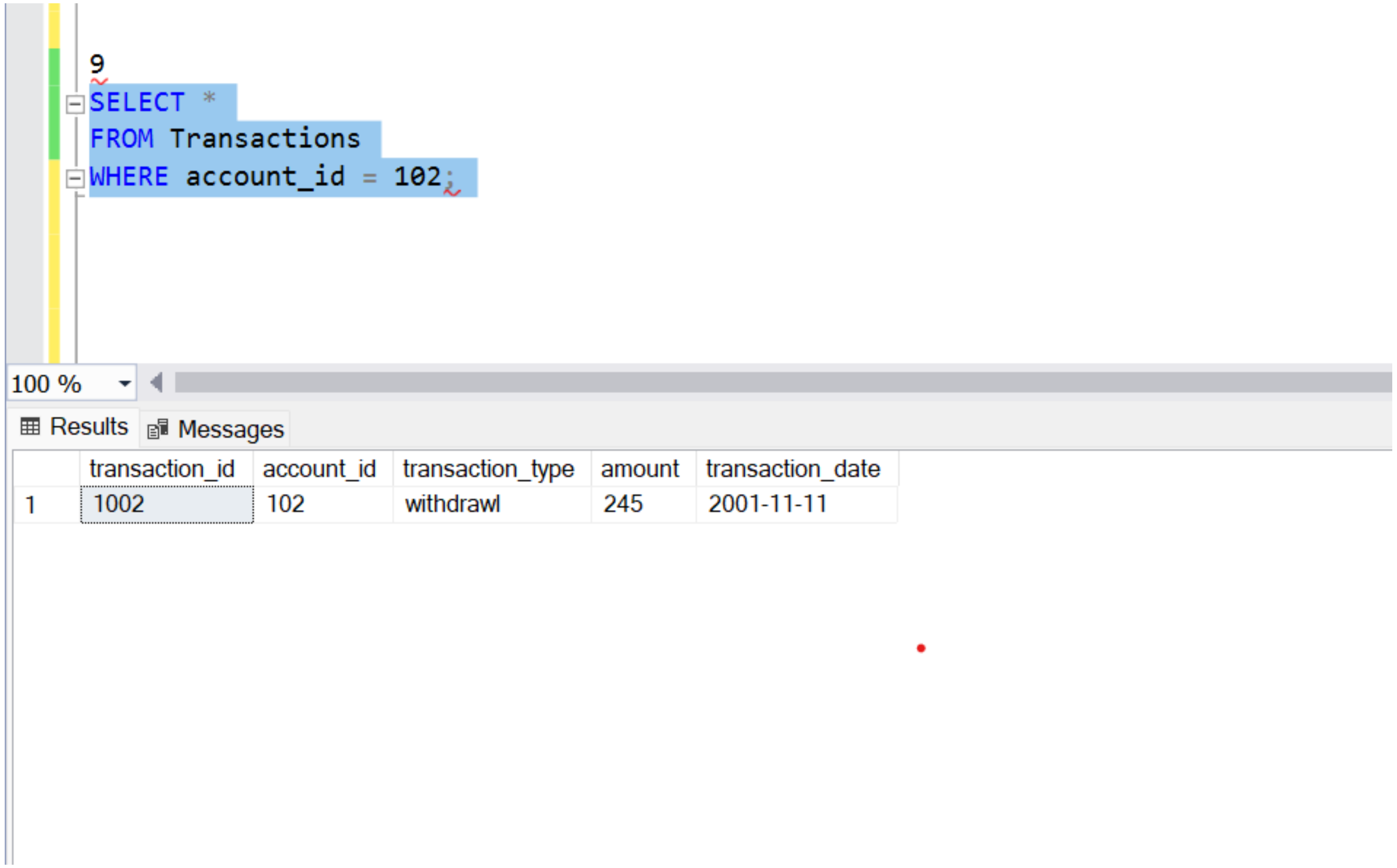
8. Write a SQL query to List all current accounts with a balance greater than $1,000.



```
8
SELECT *
FROM Accounts
WHERE account_type = 'current' AND balance > 1000
```

```
account_id   customer_id   account_type   balance
```

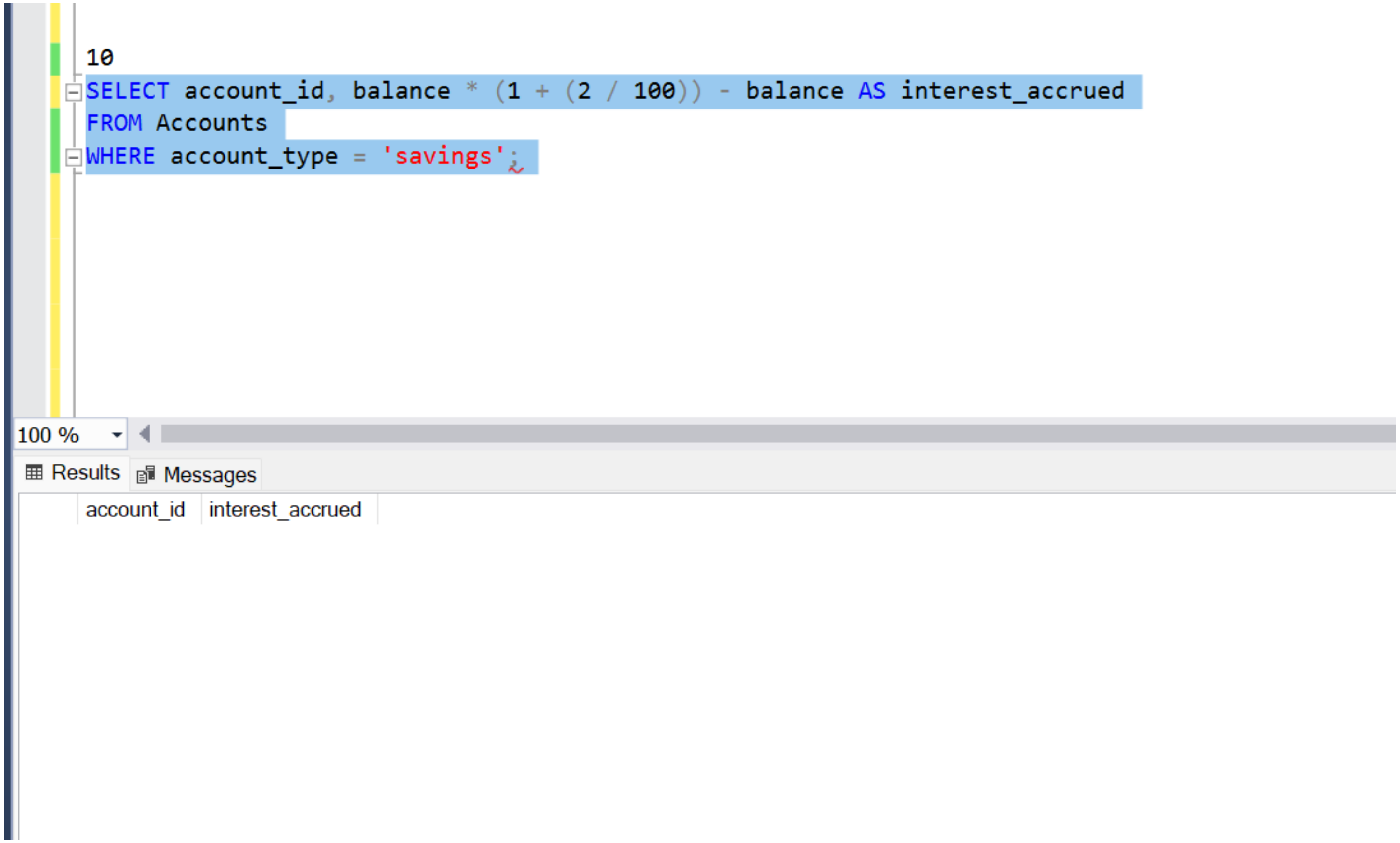
9. Write a SQL query to Retrieve all transactions for a specific account.



```
9
SELECT *
FROM Transactions
WHERE account_id = 102;
```

	transaction_id	account_id	transaction_type	amount	transaction_date
1	1002	102	withdrawl	245	2001-11-11

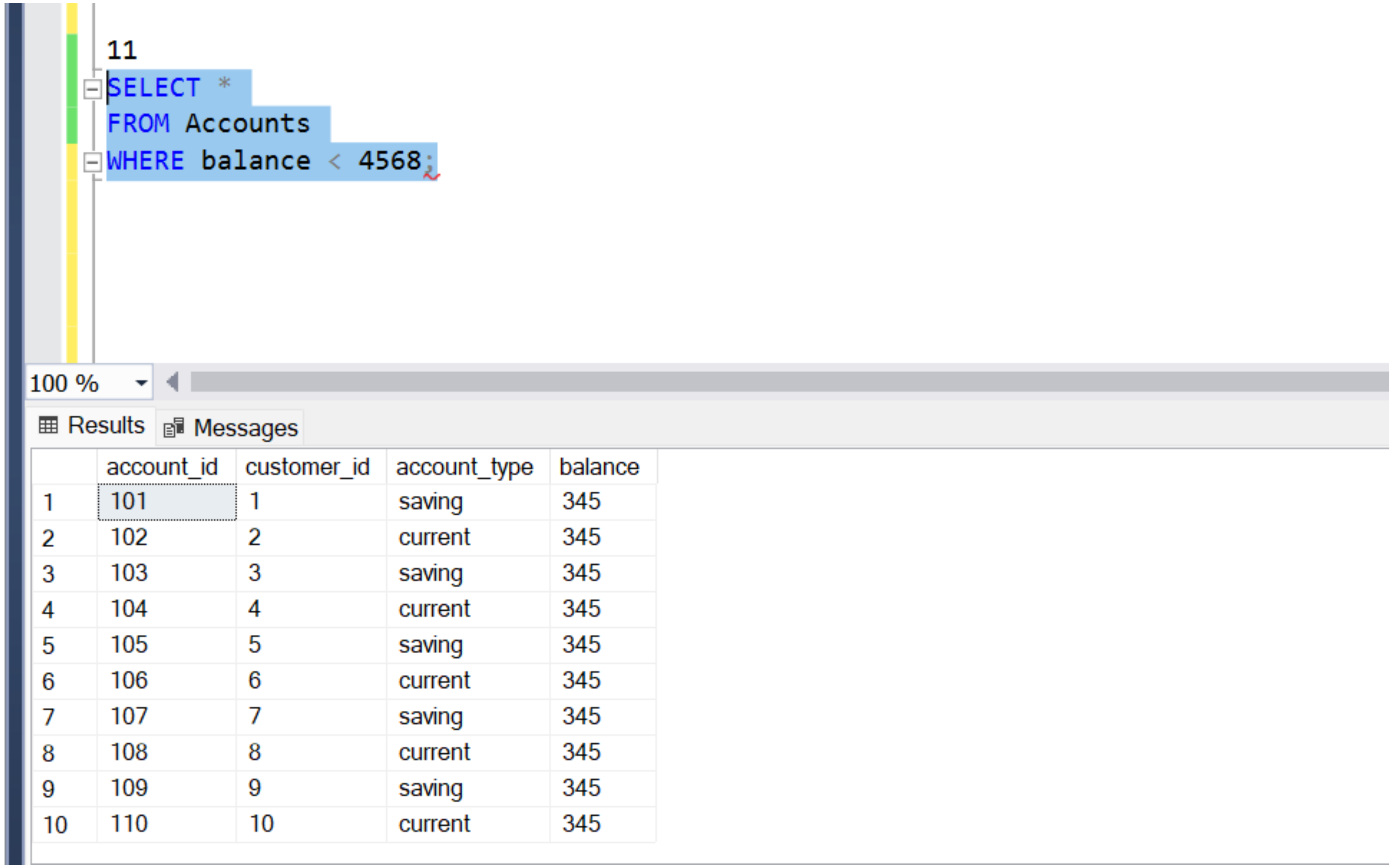
10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.



```sql
10
SELECT account_id, balance * (1 + (2 / 100)) - balance AS interest_accrued
FROM Accounts
WHERE account_type = 'savings';
```

account_id	interest_accrued

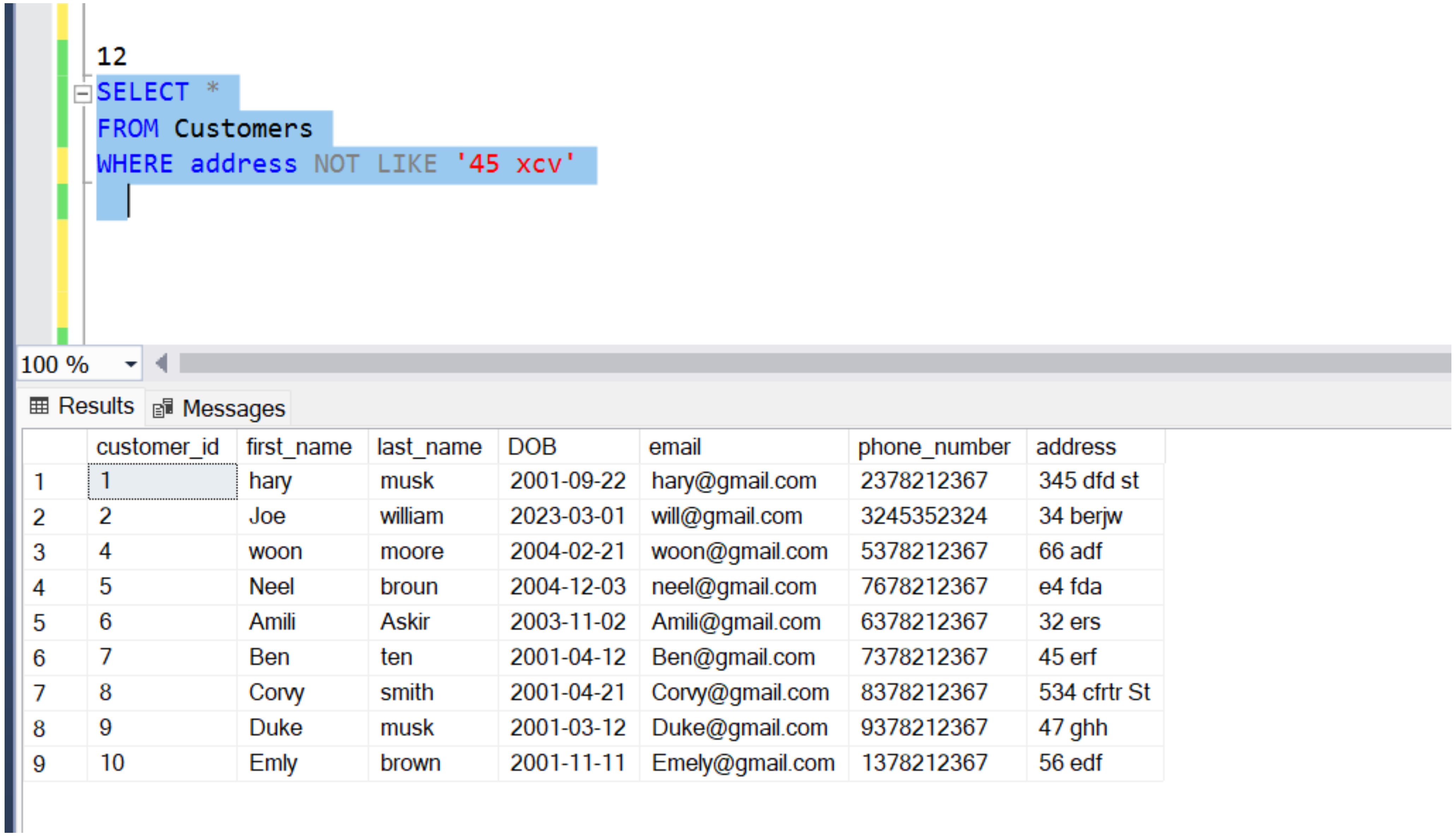
11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit.



```
11
SELECT *
FROM Accounts
WHERE balance < 4568;
```

	account_id	customer_id	account_type	balance
1	101	1	saving	345
2	102	2	current	345
3	103	3	saving	345
4	104	4	current	345
5	105	5	saving	345
6	106	6	current	345
7	107	7	saving	345
8	108	8	current	345
9	109	9	saving	345
10	110	10	current	345

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

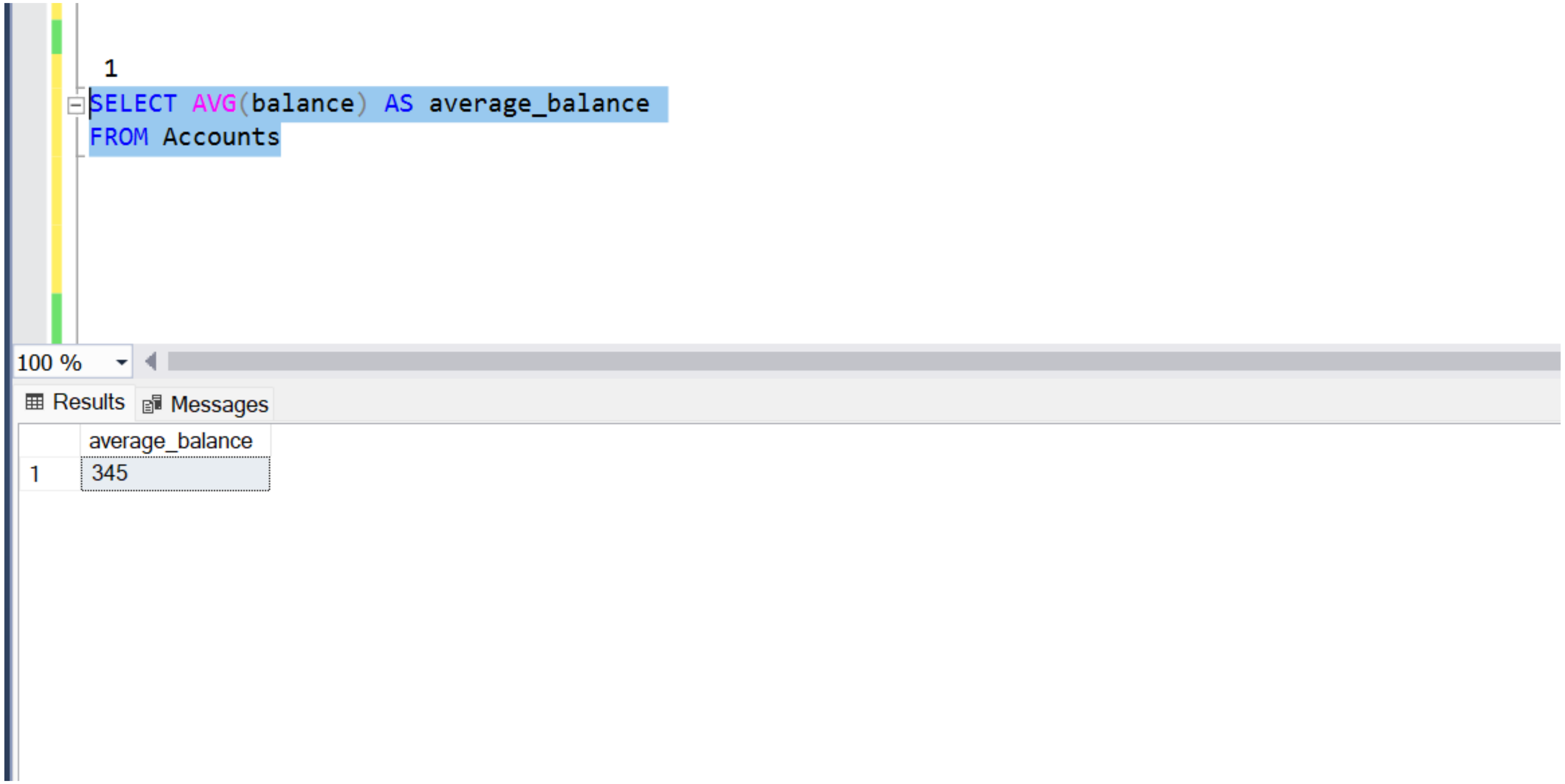


```
12
SELECT *
FROM Customers
WHERE address NOT LIKE '45 xcv'
```

	customer_id	first_name	last_name	DOB	email	phone_number	address
1	1	hary	musk	2001-09-22	hary@gmail.com	2378212367	345 dfd st
2	2	Joe	william	2023-03-01	will@gmail.com	3245352324	34 berjw
3	4	woon	moore	2004-02-21	woon@gmail.com	5378212367	66 adf
4	5	Neel	broun	2004-12-03	neel@gmail.com	7678212367	e4 fda
5	6	Amili	Askir	2003-11-02	Amili@gmail.com	6378212367	32 ers
6	7	Ben	ten	2001-04-12	Ben@gmail.com	7378212367	45 erf
7	8	Corvy	smith	2001-04-21	Corvy@gmail.com	8378212367	534 cfrtr St
8	9	Duke	musk	2001-03-12	Duke@gmail.com	9378212367	47 ghh
9	10	Emly	brown	2001-11-11	Emely@gmail.com	1378212367	56 edf

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

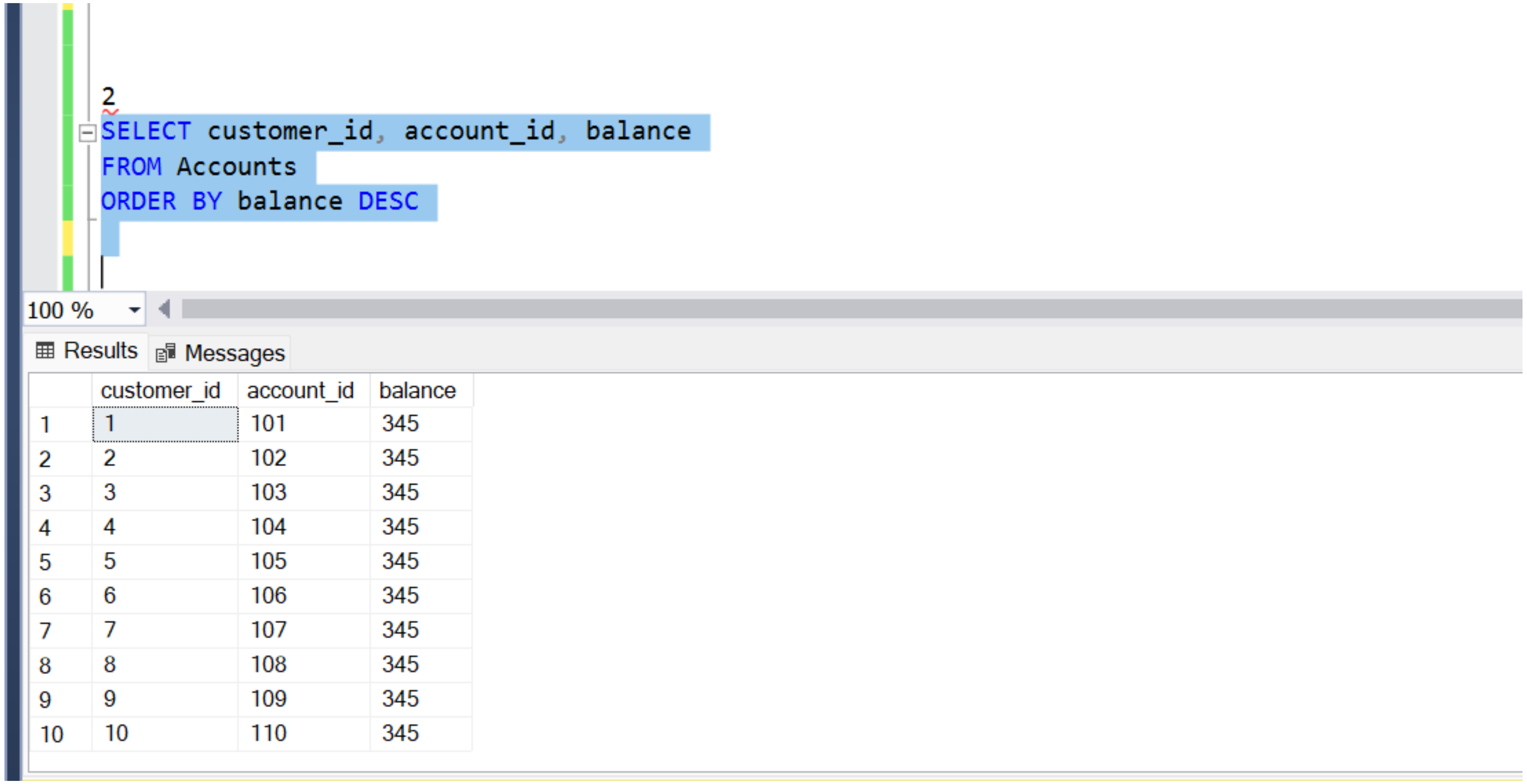
1. Write a SQL query to Find the average account balance for all customers.



```
SELECT AVG(balance) AS average_balance
FROM Accounts
```

```
   average_balance
1  345
```

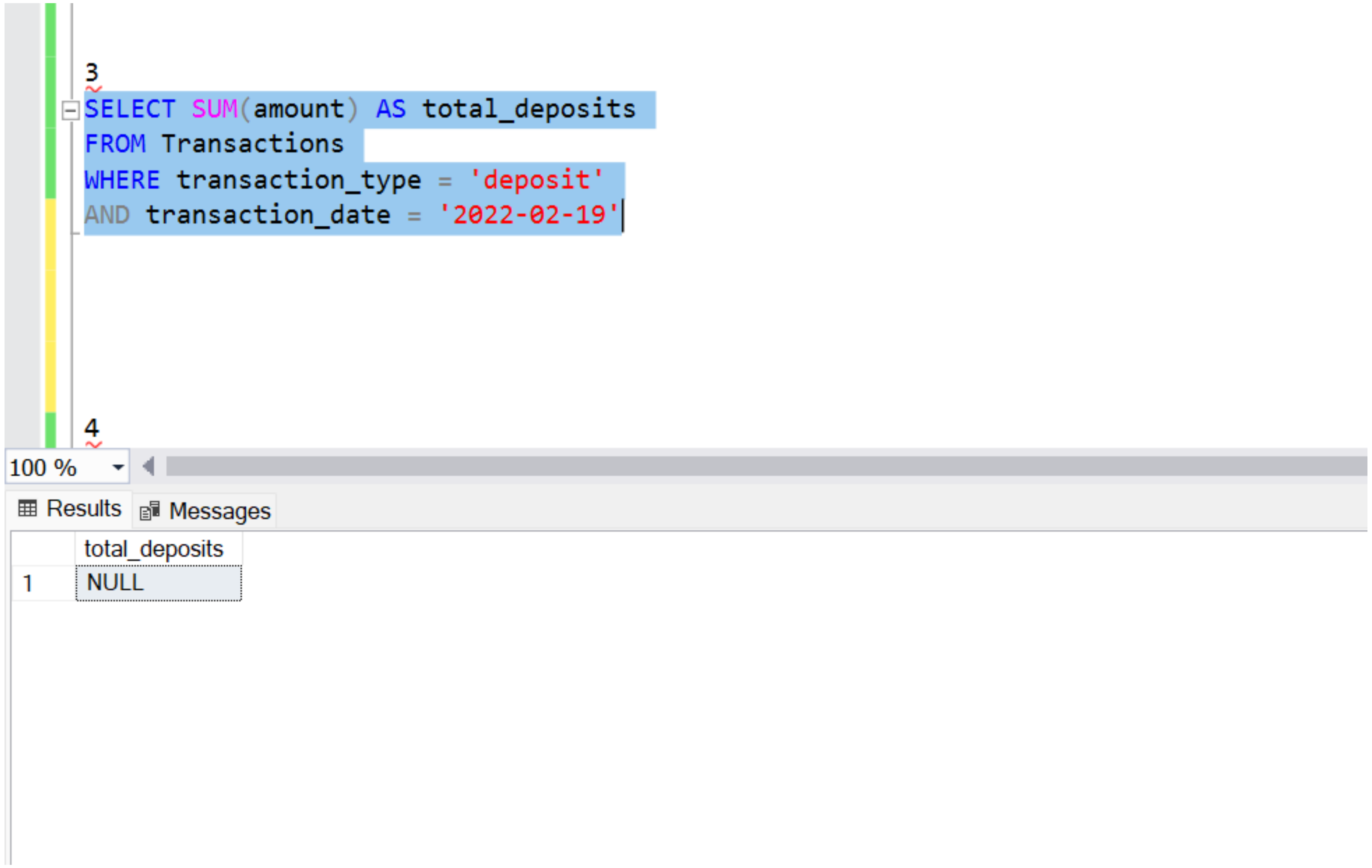
2. Write a SQL query to Retrieve the top 10 highest account balances.



```
2
SELECT customer_id, account_id, balance
FROM Accounts
ORDER BY balance DESC
```

	customer_id	account_id	balance
1	1	101	345
2	2	102	345
3	3	103	345
4	4	104	345
5	5	105	345
6	6	106	345
7	7	107	345
8	8	108	345
9	9	109	345
10	10	110	345

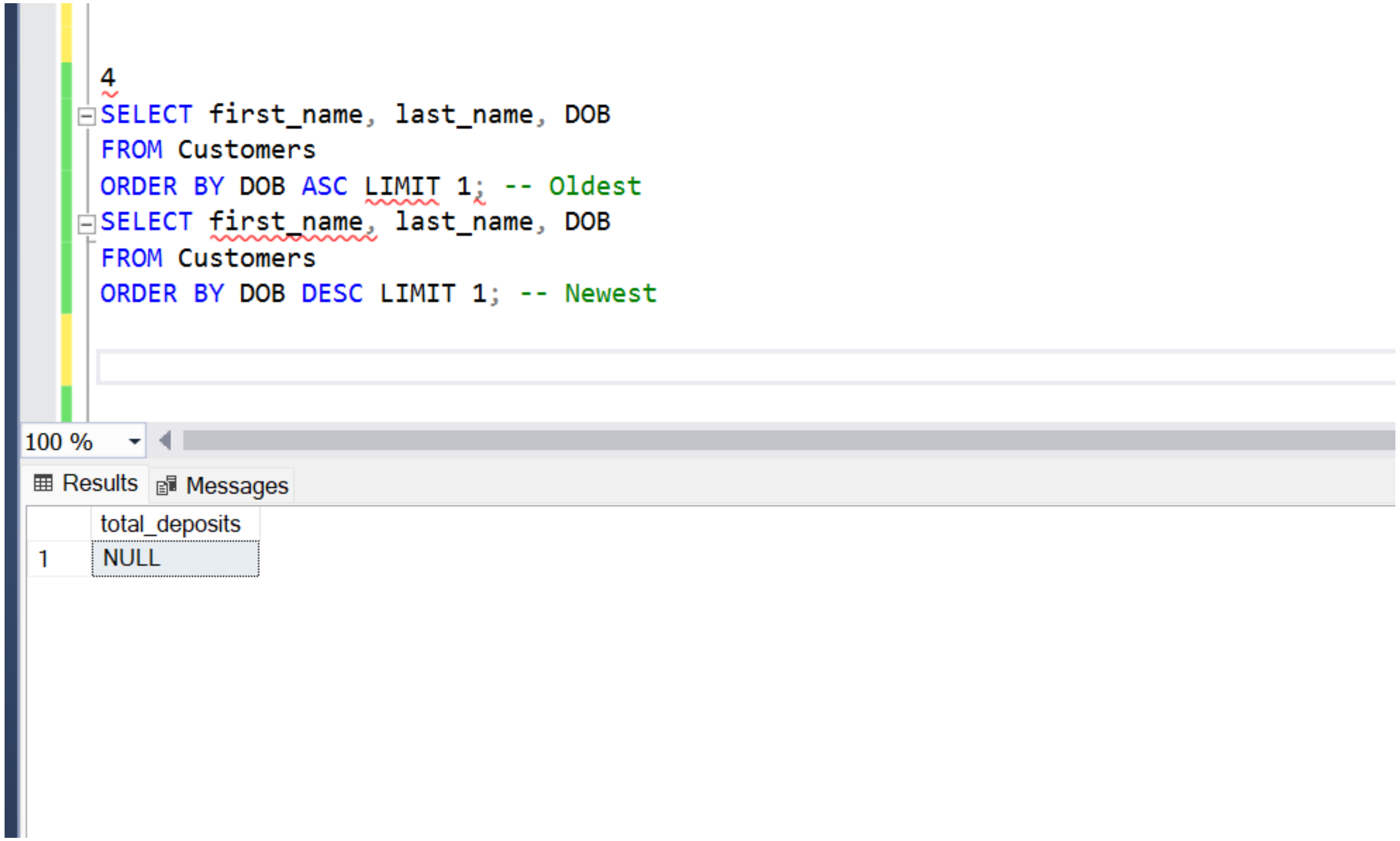
3. Write a SQL query to Calculate Total Deposits for All Customers in specific date.



```sql
3
SELECT SUM(amount) AS total_deposits
FROM Transactions
WHERE transaction_type = 'deposit'
AND transaction_date = '2022-02-19'




4
```

```
   total_deposits
1  NULL
```

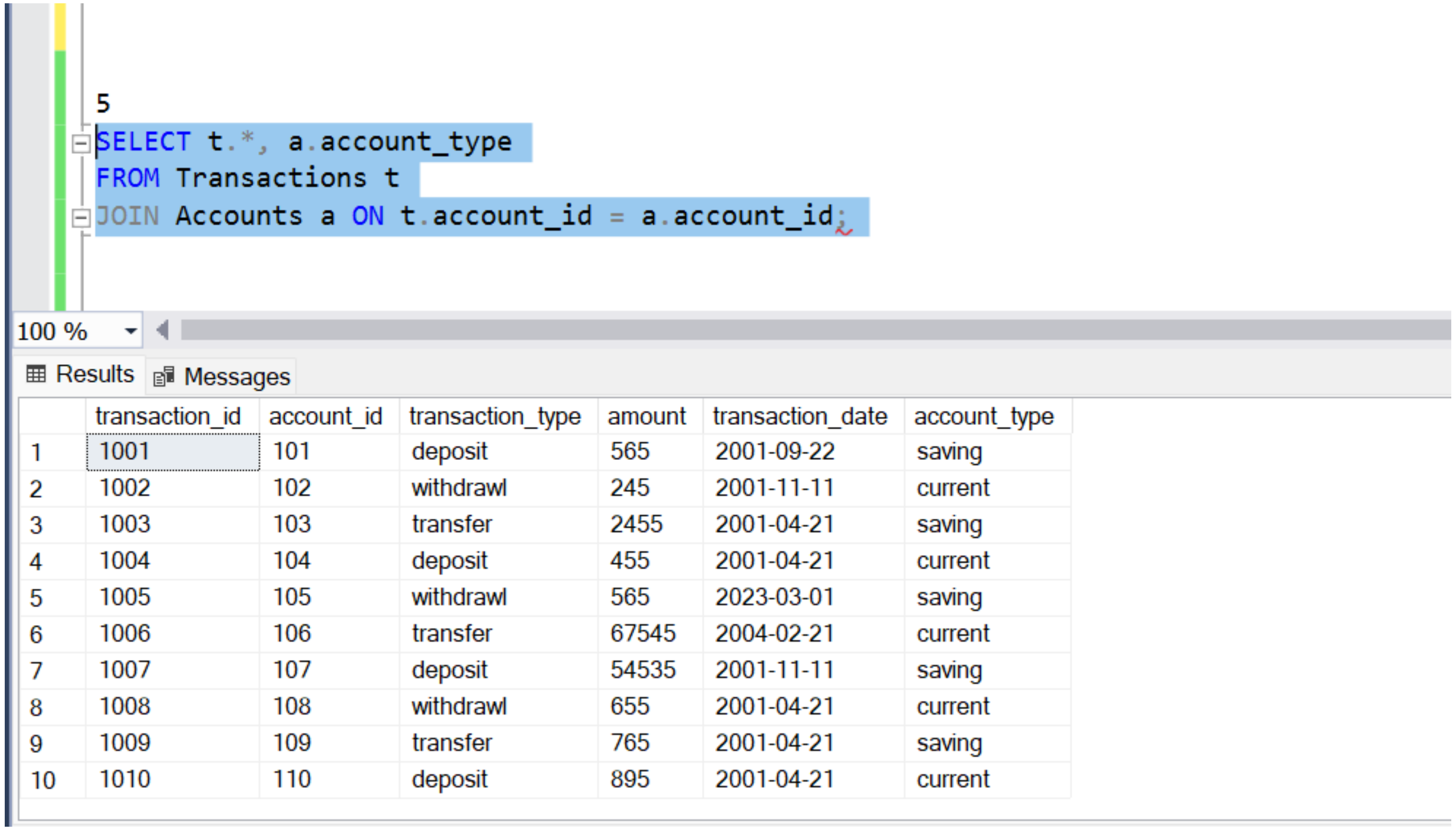
4. Write a SQL query to Find the Oldest and Newest Customers.



```
4
SELECT first_name, last_name, DOB
FROM Customers
ORDER BY DOB ASC LIMIT 1; -- Oldest
SELECT first_name, last_name, DOB
FROM Customers
ORDER BY DOB DESC LIMIT 1; -- Newest
```

```
   total_deposits
1  NULL
```

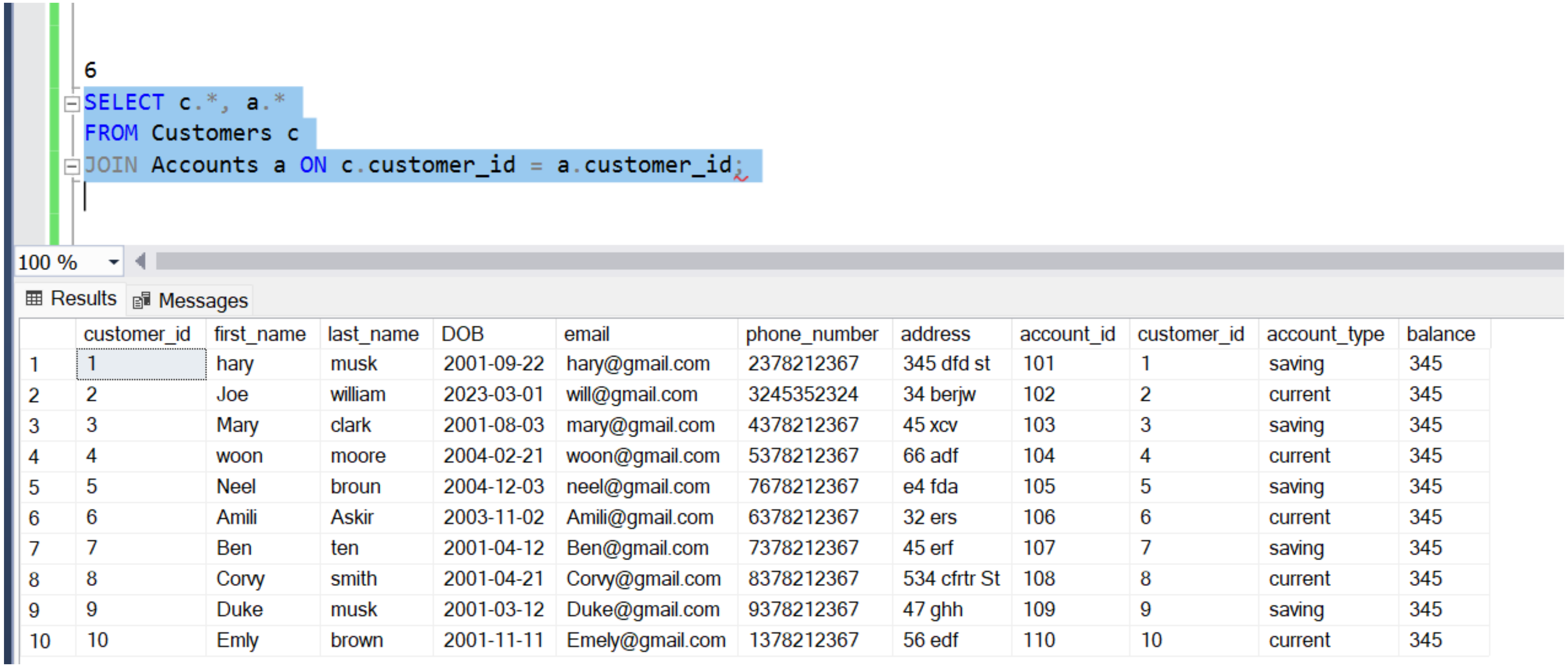
5. Write a SQL query to Retrieve transaction details along with the account type.



```
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
1	1001	101	deposit	565	2001-09-22	saving
2	1002	102	withdrawl	245	2001-11-11	current
3	1003	103	transfer	2455	2001-04-21	saving
4	1004	104	deposit	455	2001-04-21	current
5	1005	105	withdrawl	565	2023-03-01	saving
6	1006	106	transfer	67545	2004-02-21	current
7	1007	107	deposit	54535	2001-11-11	saving
8	1008	108	withdrawl	655	2001-04-21	current
9	1009	109	transfer	765	2001-04-21	saving
10	1010	110	deposit	895	2001-04-21	current

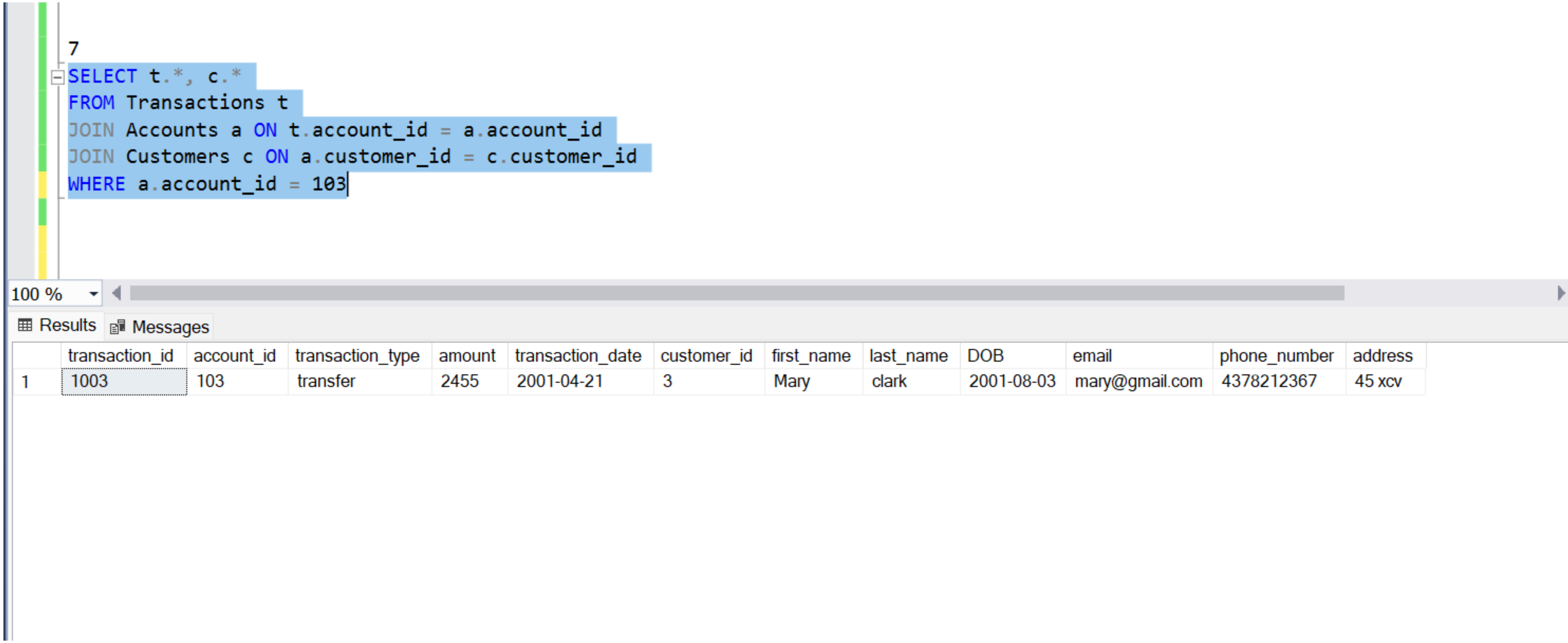
6. Write a SQL query to Get a list of customers along with their account details.



```
6
SELECT c.*, a.*
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id;
```

	customer_id	first_name	last_name	DOB	email	phone_number	address	account_id	customer_id	account_type	balance
1	1	hary	musk	2001-09-22	hary@gmail.com	2378212367	345 dfd st	101	1	saving	345
2	2	Joe	william	2023-03-01	will@gmail.com	3245352324	34 berjw	102	2	current	345
3	3	Mary	clark	2001-08-03	mary@gmail.com	4378212367	45 xcv	103	3	saving	345
4	4	woon	moore	2004-02-21	woon@gmail.com	5378212367	66 adf	104	4	current	345
5	5	Neel	broun	2004-12-03	neel@gmail.com	7678212367	e4 fda	105	5	saving	345
6	6	Amili	Askir	2003-11-02	Amili@gmail.com	6378212367	32 ers	106	6	current	345
7	7	Ben	ten	2001-04-12	Ben@gmail.com	7378212367	45 erf	107	7	saving	345
8	8	Corvy	smith	2001-04-21	Corvy@gmail.com	8378212367	534 cfrtr St	108	8	current	345
9	9	Duke	musk	2001-03-12	Duke@gmail.com	9378212367	47 ghh	109	9	saving	345
10	10	Emly	brown	2001-11-11	Emely@gmail.com	1378212367	56 edf	110	10	current	345

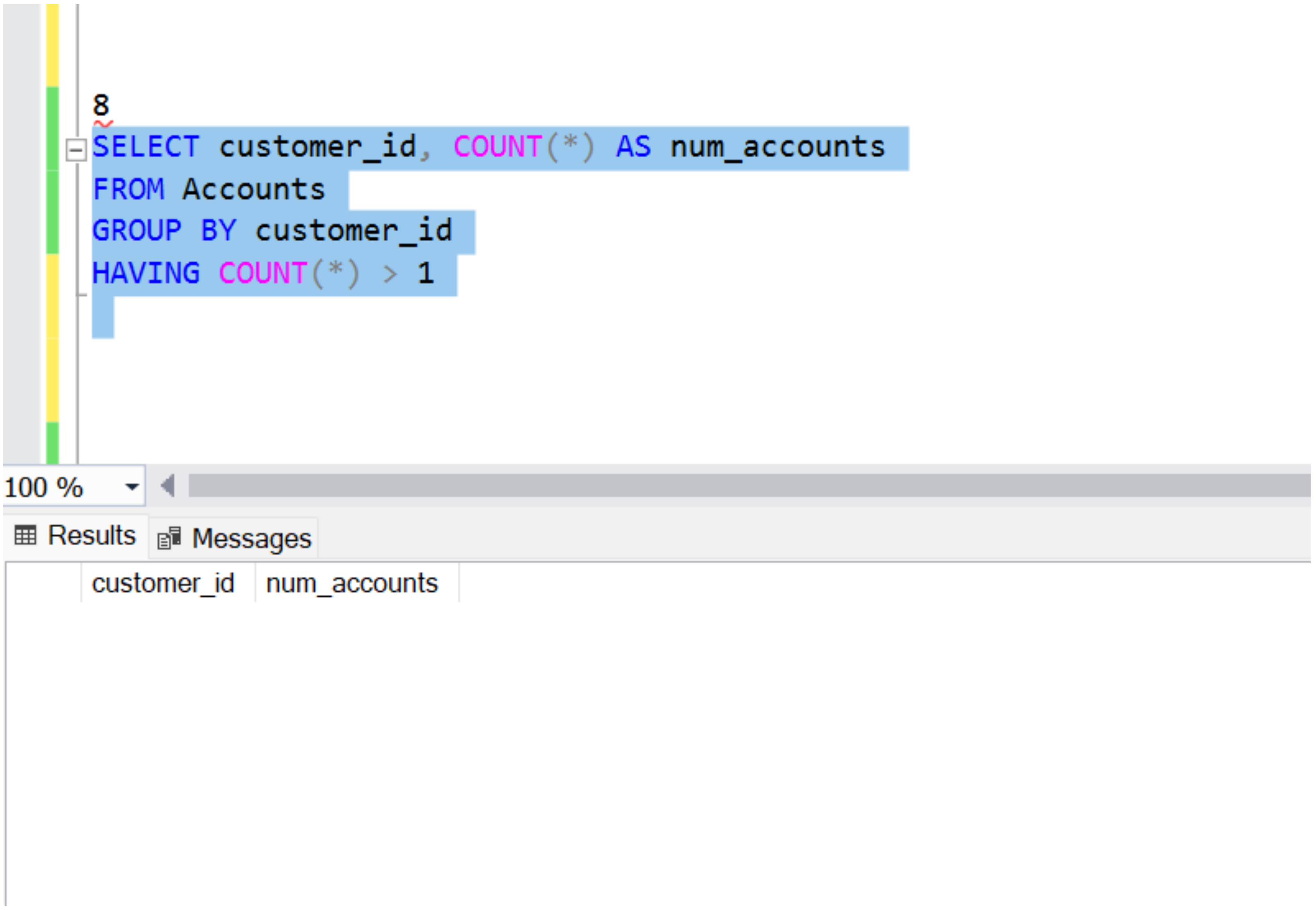
7. Write a SQL query to Retrieve transaction details along with customer information for a specific account.



```
7
SELECT t.*, c.*
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 = 103
```

	transaction_id	account_id	transaction_type	amount	transaction_date	customer_id	first_name	last_name	DOB	email	phone_number	address
1	1003	103	transfer	2455	2001-04-21	3	Mary	clark	2001-08-03	mary@gmail.com	4378212367	45 xcv

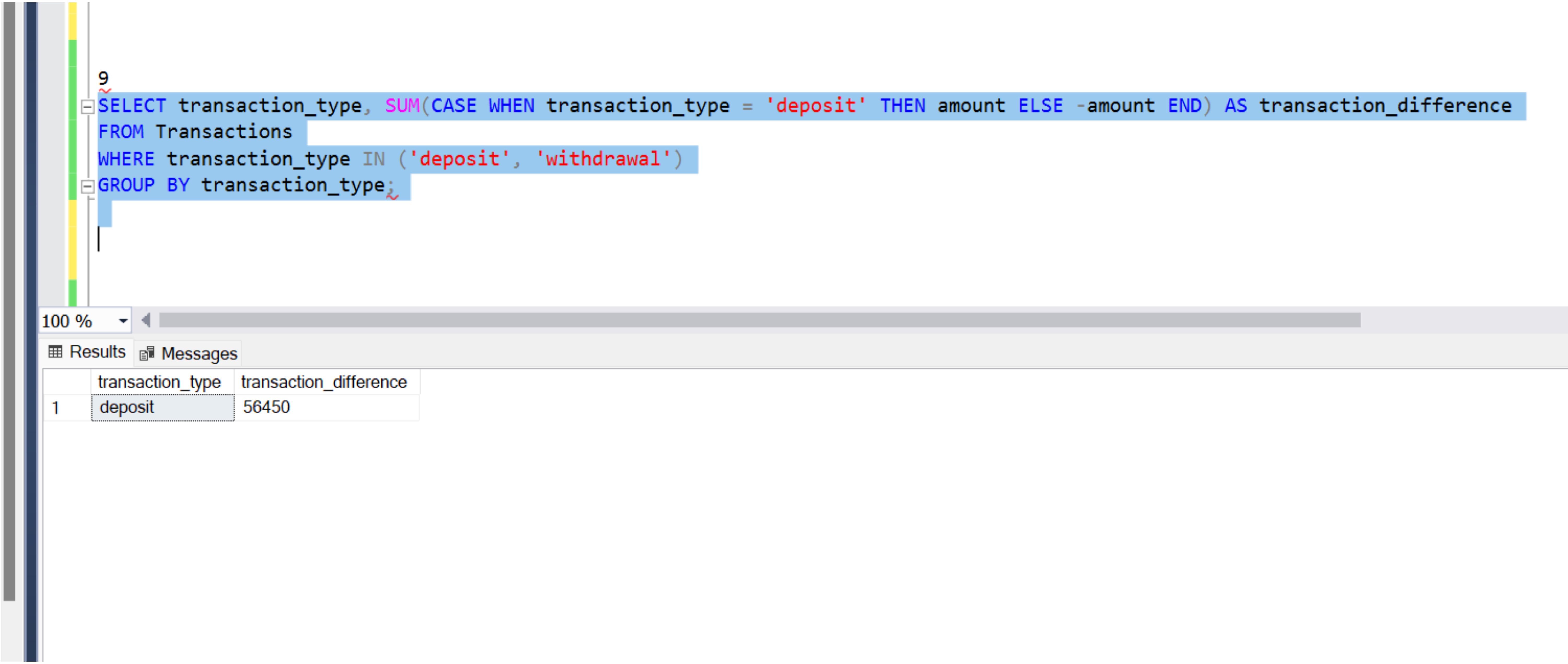
8. Write a SQL query to Identify customers who have more than one account.



```
8
SELECT customer_id, COUNT(*) AS num_accounts
FROM Accounts
GROUP BY customer_id
HAVING COUNT(*) > 1
```

customer_id	num_accounts

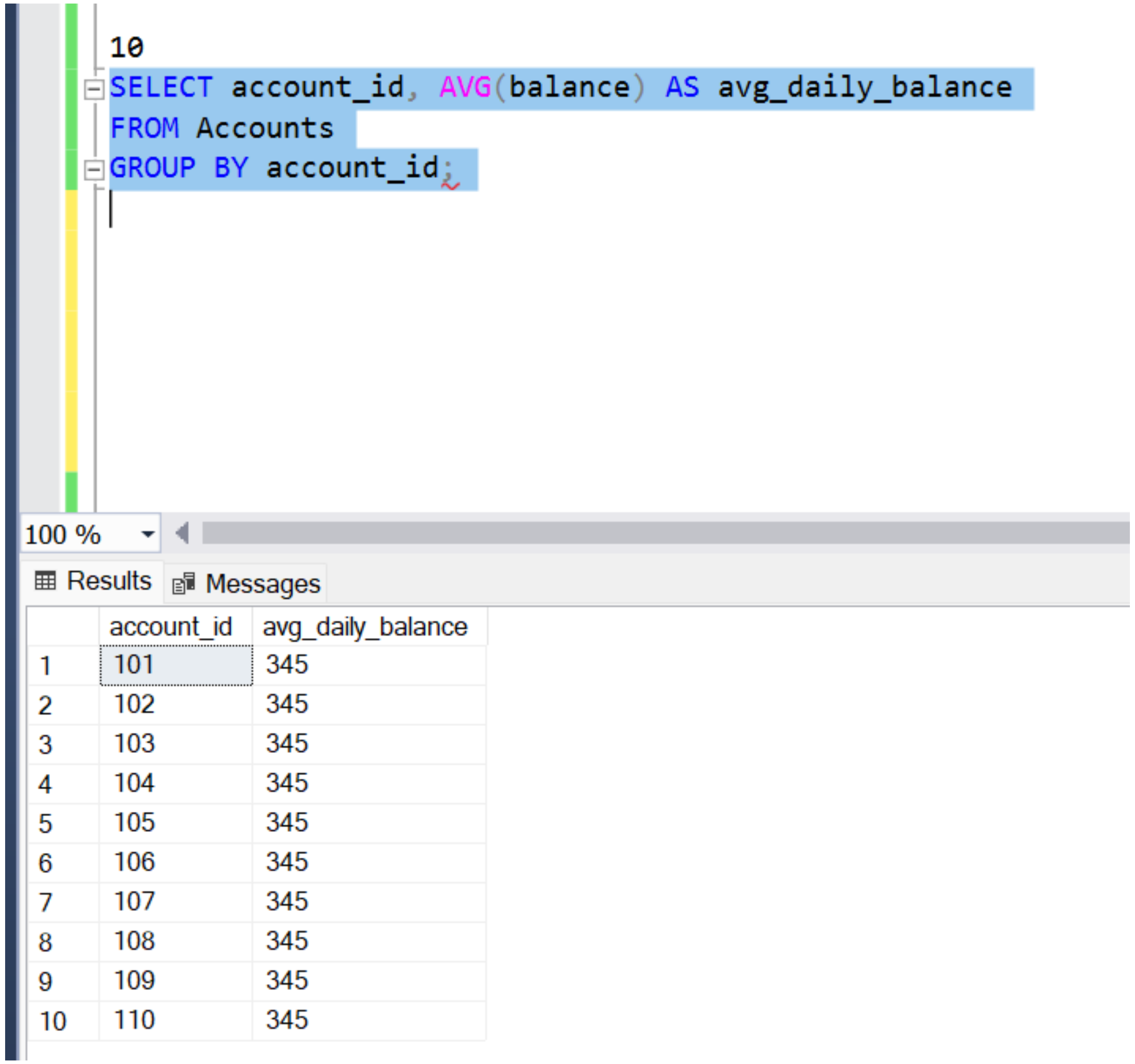
9. Write a SQL query to Calculate the difference in transaction amounts between deposits and withdrawals.



```
9
SELECT transaction_type, SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS transaction_difference
FROM Transactions
WHERE transaction_type IN ('deposit', 'withdrawal')
GROUP BY transaction_type;
```

```
   transaction_type   transaction_difference
1  deposit            56450
```

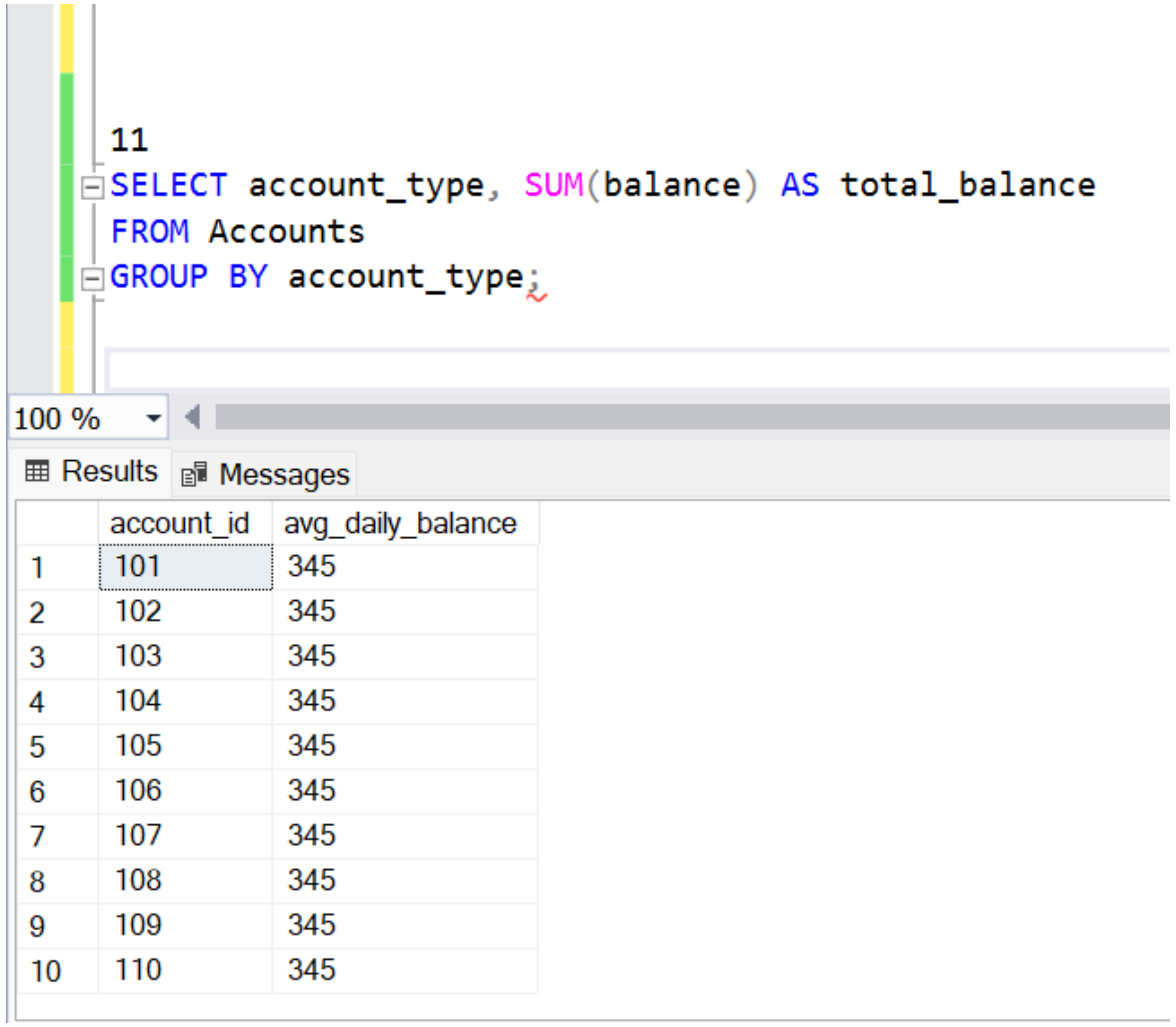
10. Write a SQL query to Calculate the average daily balance for each account over a specified
period.



```sql
10
SELECT account_id, AVG(balance) AS avg_daily_balance
FROM Accounts
GROUP BY account_id;
```

	account_id	avg_daily_balance
1	101	345
2	102	345
3	103	345
4	104	345
5	105	345
6	106	345
7	107	345
8	108	345
9	109	345
10	110	345

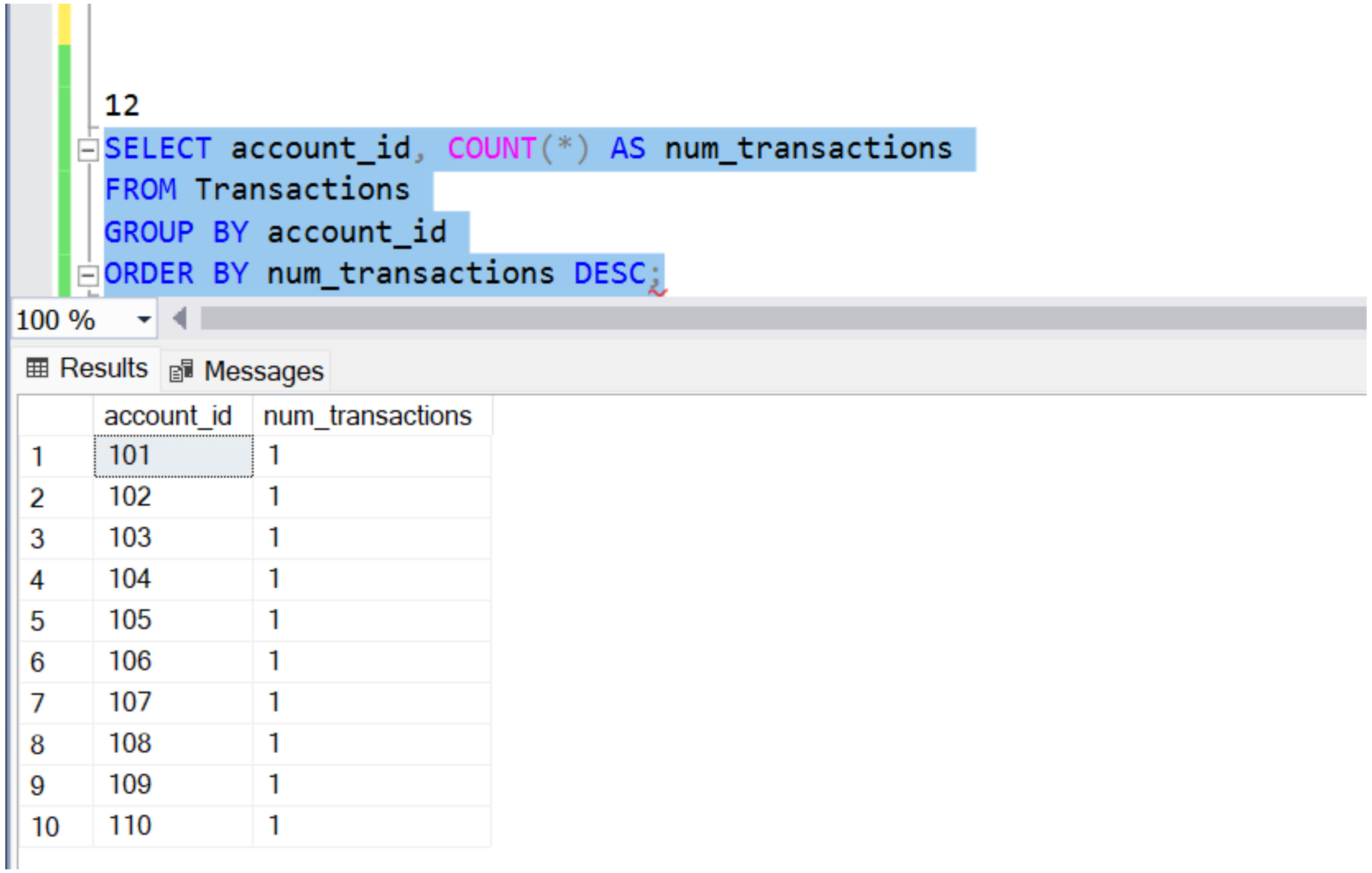
11.Calculate the total balance for each account type.



```
11
SELECT account_type, SUM(balance) AS total_balance
FROM Accounts
GROUP BY account_type;
```

	account_id	avg_daily_balance
1	101	345
2	102	345
3	103	345
4	104	345
5	105	345
6	106	345
7	107	345
8	108	345
9	109	345
10	110	345

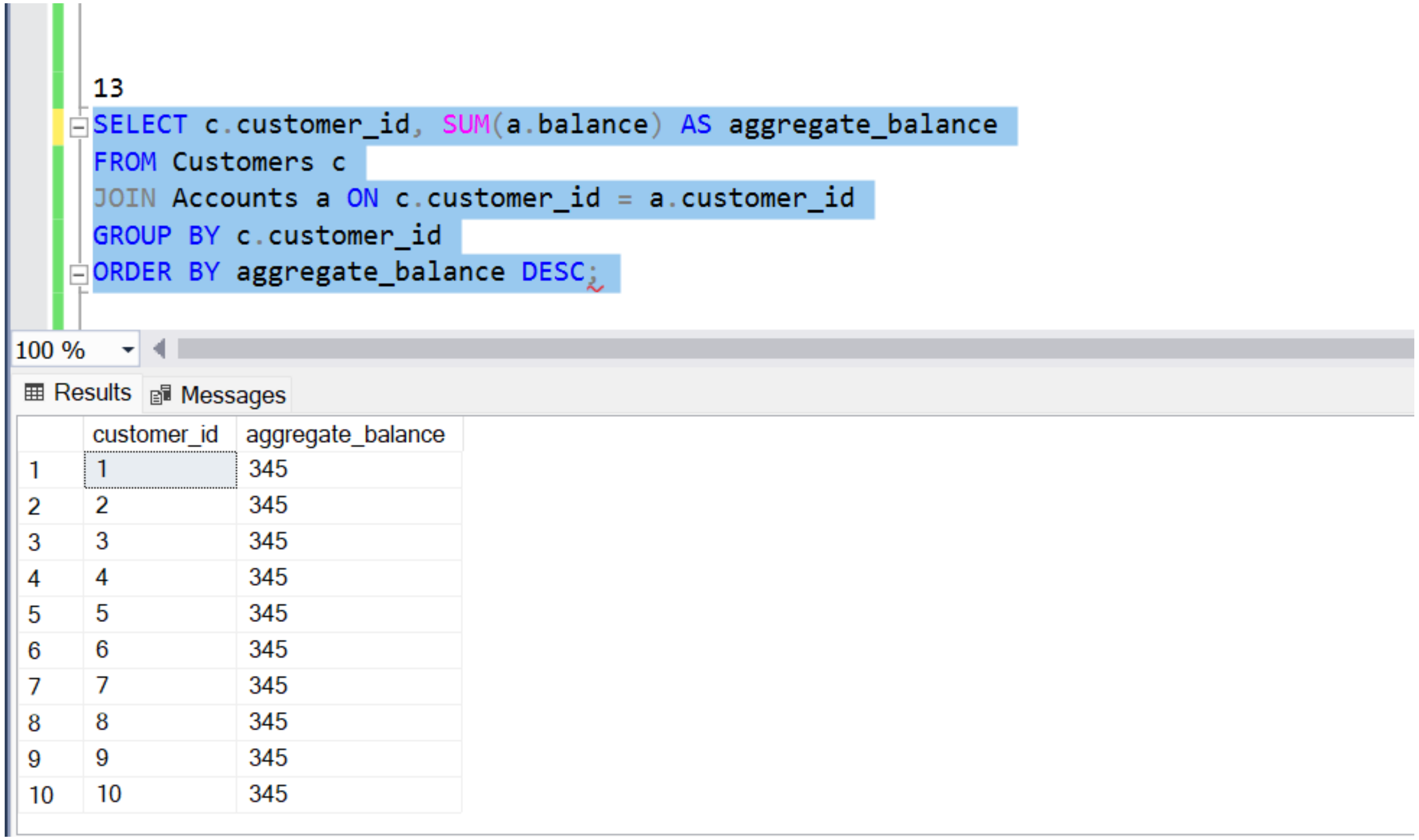
12. Identify accounts with the highest number of transactions order by descending order.



```
12
SELECT account_id, COUNT(*) AS num_transactions
FROM Transactions
GROUP BY account_id
ORDER BY num_transactions DESC;
```

	account_id	num_transactions
1	101	1
2	102	1
3	103	1
4	104	1
5	105	1
6	106	1
7	107	1
8	108	1
9	109	1
10	110	1

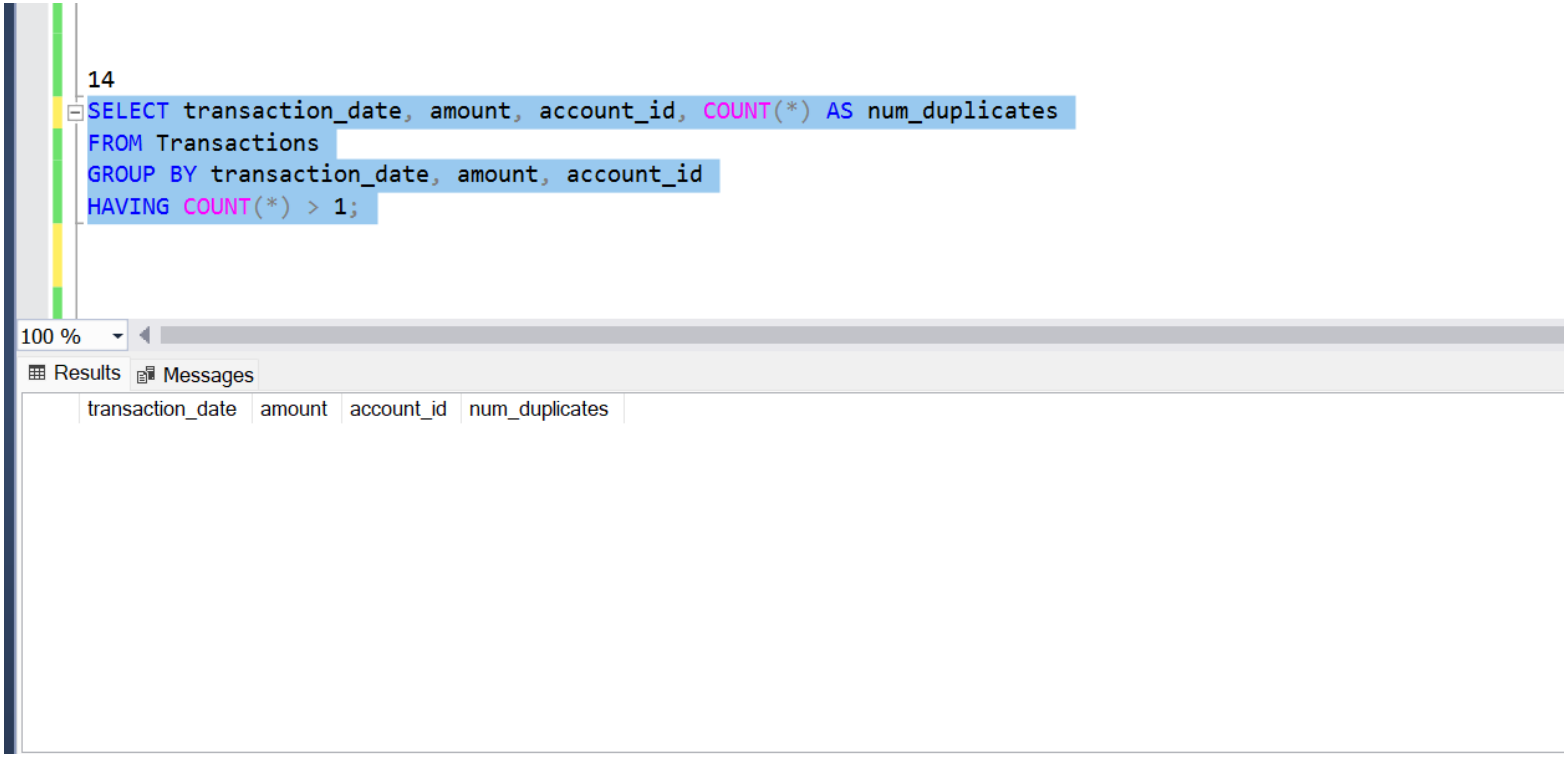
13. List customers with high aggregate account balances, along with their account types.



```
13
SELECT c.customer_id, SUM(a.balance) AS aggregate_balance
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
GROUP BY c.customer_id
ORDER BY aggregate_balance DESC;
```

	customer_id	aggregate_balance
1	1	345
2	2	345
3	3	345
4	4	345
5	5	345
6	6	345
7	7	345
8	8	345
9	9	345
10	10	345

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

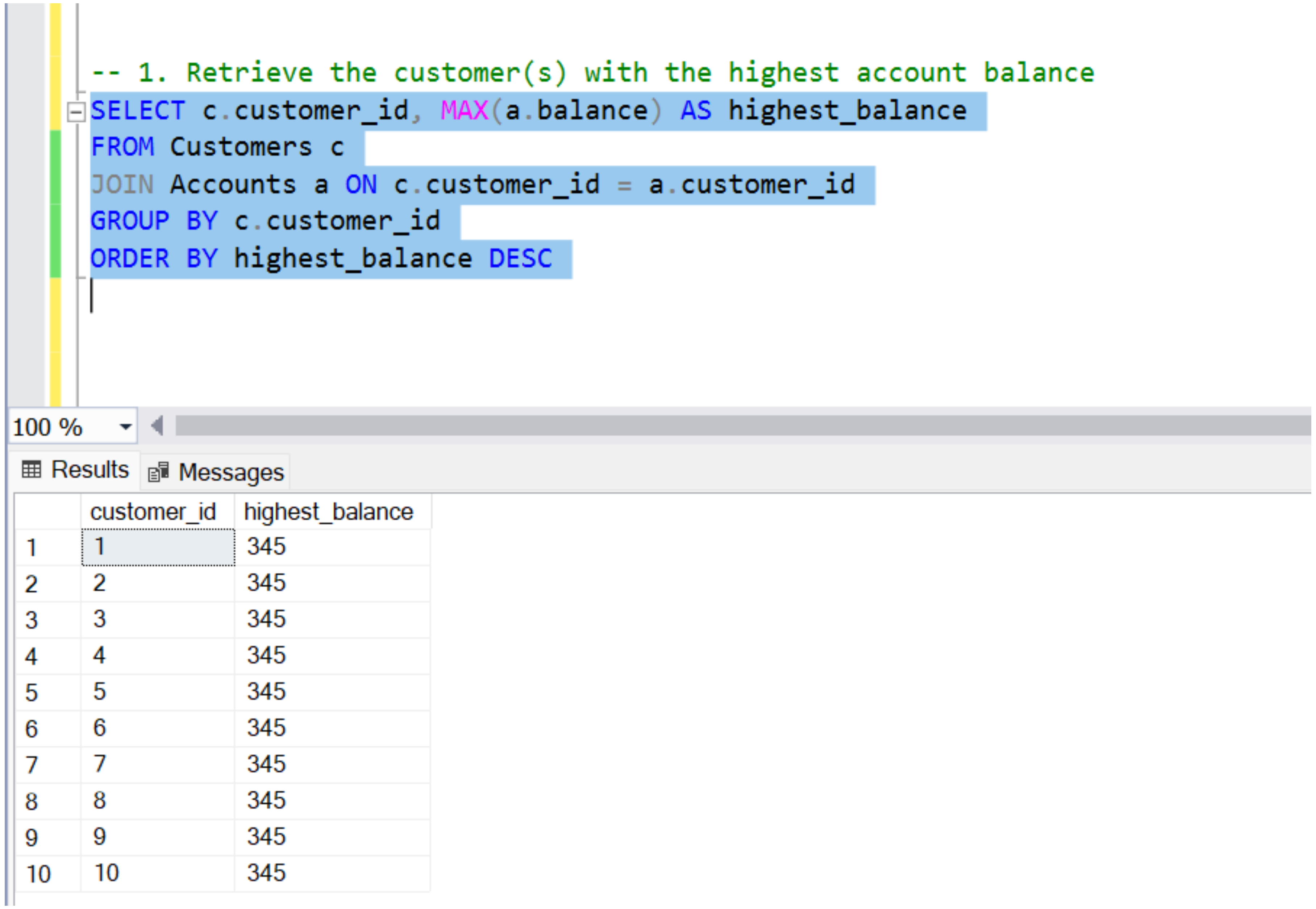


```
14
SELECT transaction_date, amount, account_id, COUNT(*) AS num_duplicates
FROM Transactions
GROUP BY transaction_date, amount, account_id
HAVING COUNT(*) > 1;
```

transaction_date	amount	account_id	num_duplicates

Tasks 4: Subquery and its type:

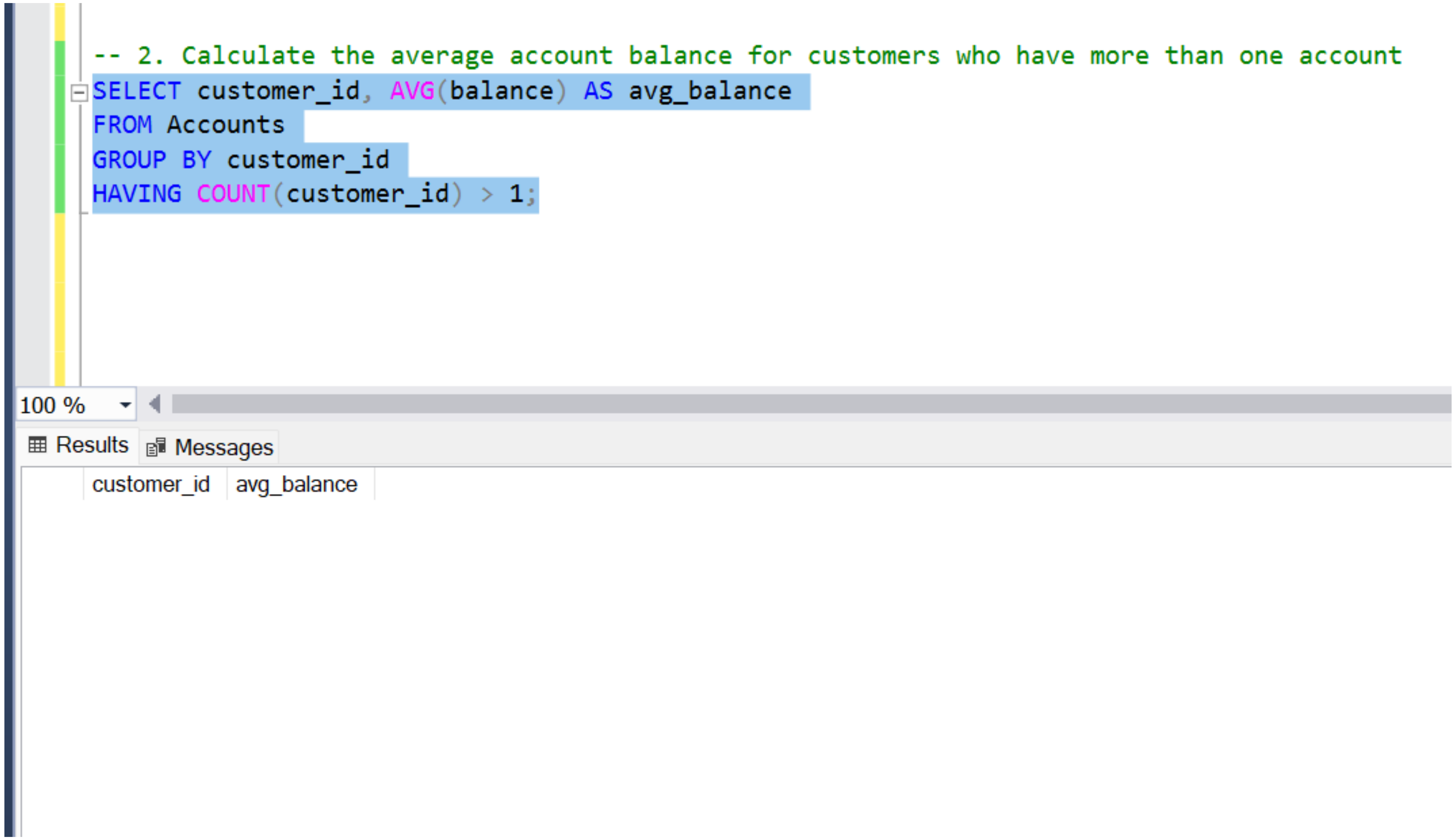
1. Retrieve the customer(s) with the highest account balance.



```sql
-- 1. Retrieve the customer(s) with the highest account balance
SELECT c.customer_id, MAX(a.balance) AS highest_balance
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
GROUP BY c.customer_id
ORDER BY highest_balance DESC
```

	customer_id	highest_balance
1	1	345
2	2	345
3	3	345
4	4	345
5	5	345
6	6	345
7	7	345
8	8	345
9	9	345
10	10	345

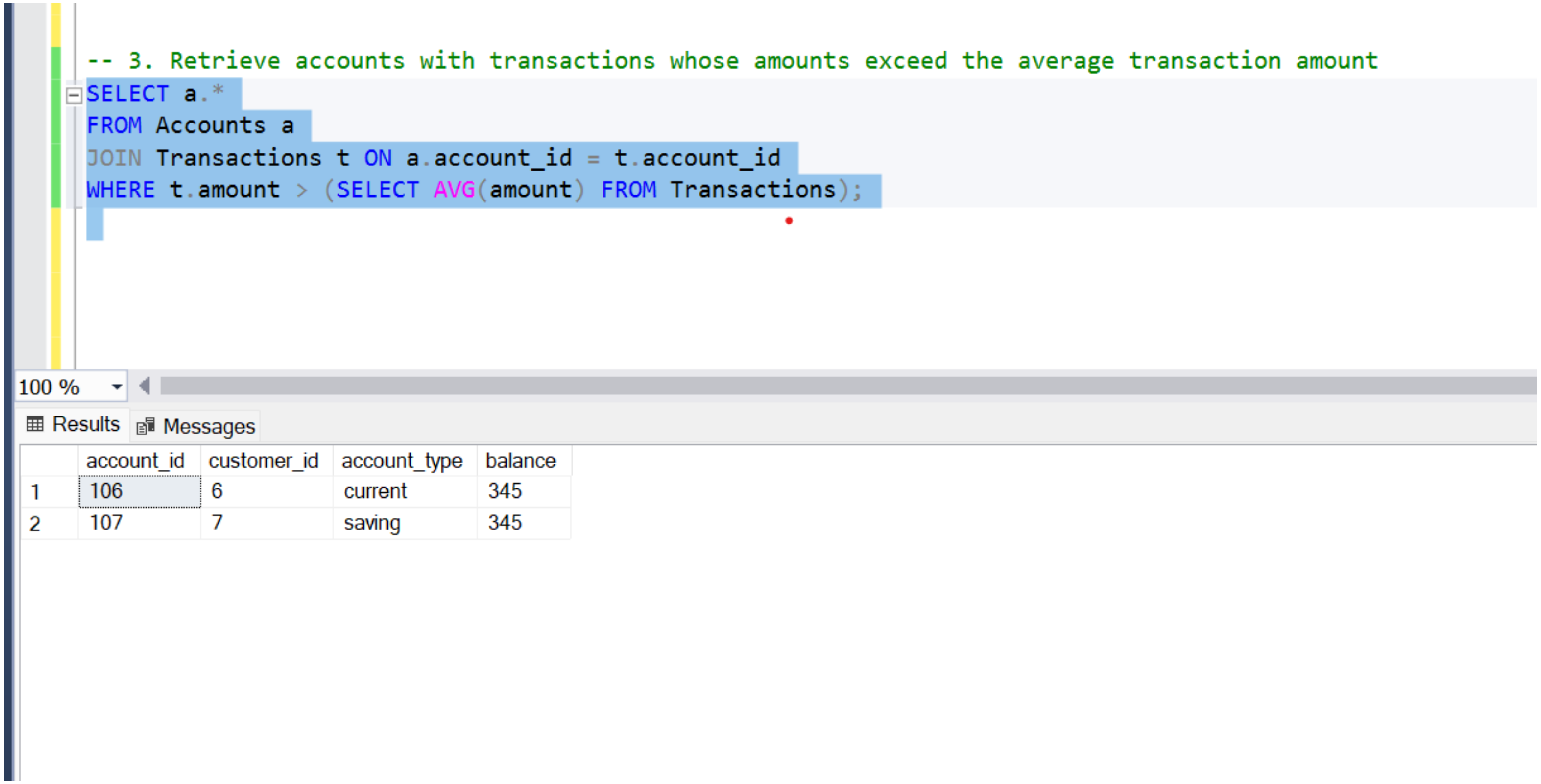
2. Calculate the average account balance for customers who have more than one account.



```sql
-- 2. Calculate the average account balance for customers who have more than one account
SELECT customer_id, AVG(balance) AS avg_balance
FROM Accounts
GROUP BY customer_id
HAVING COUNT(customer_id) > 1;
```

```
customer_id   avg_balance
```

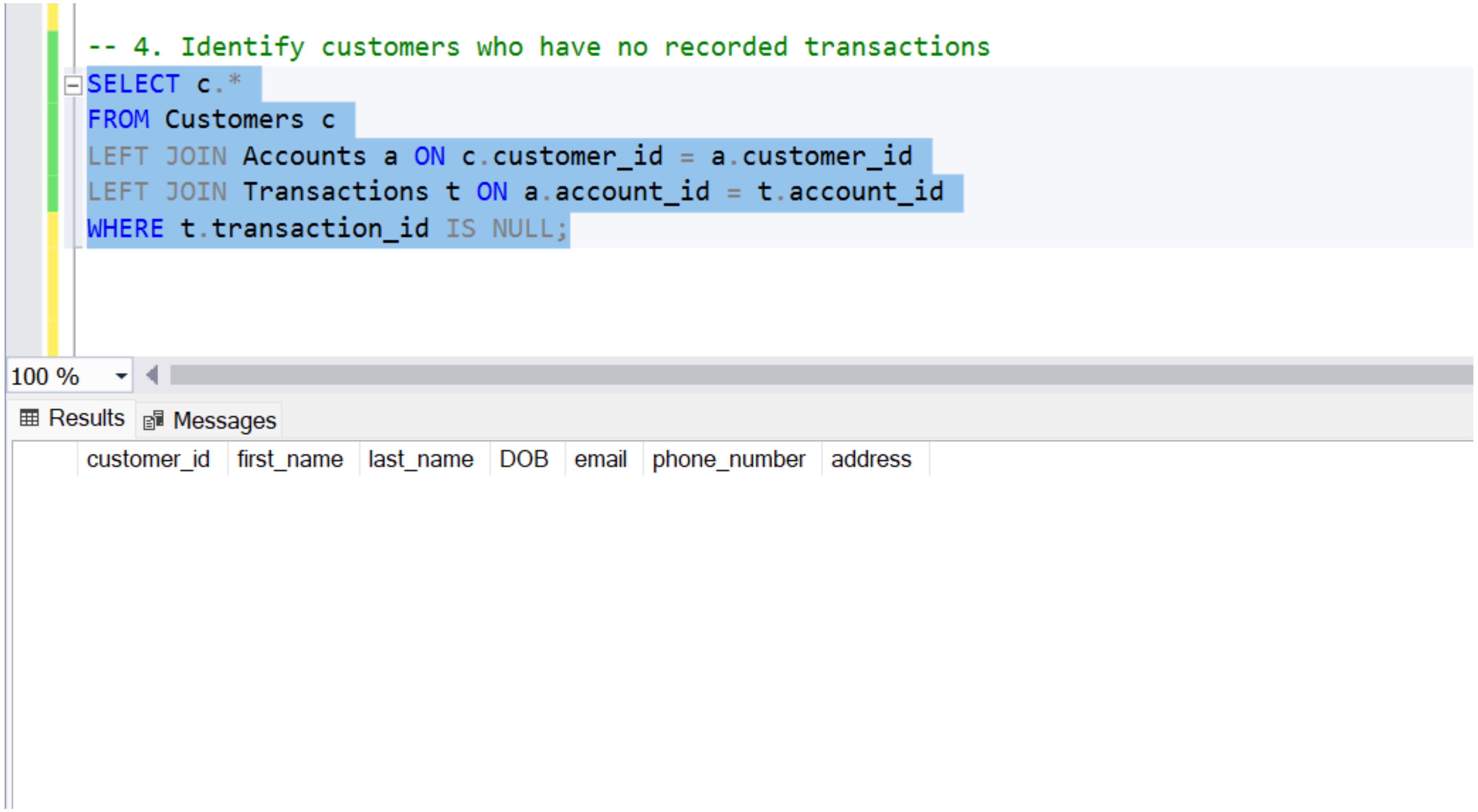
3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.



```sql
-- 3. Retrieve accounts with transactions whose amounts exceed the average transaction amount
SELECT a.*
FROM Accounts a
JOIN Transactions t ON a.account_id = t.account_id
WHERE t.amount > (SELECT AVG(amount) FROM Transactions);
```

	account_id	customer_id	account_type	balance
1	106	6	current	345
2	107	7	saving	345

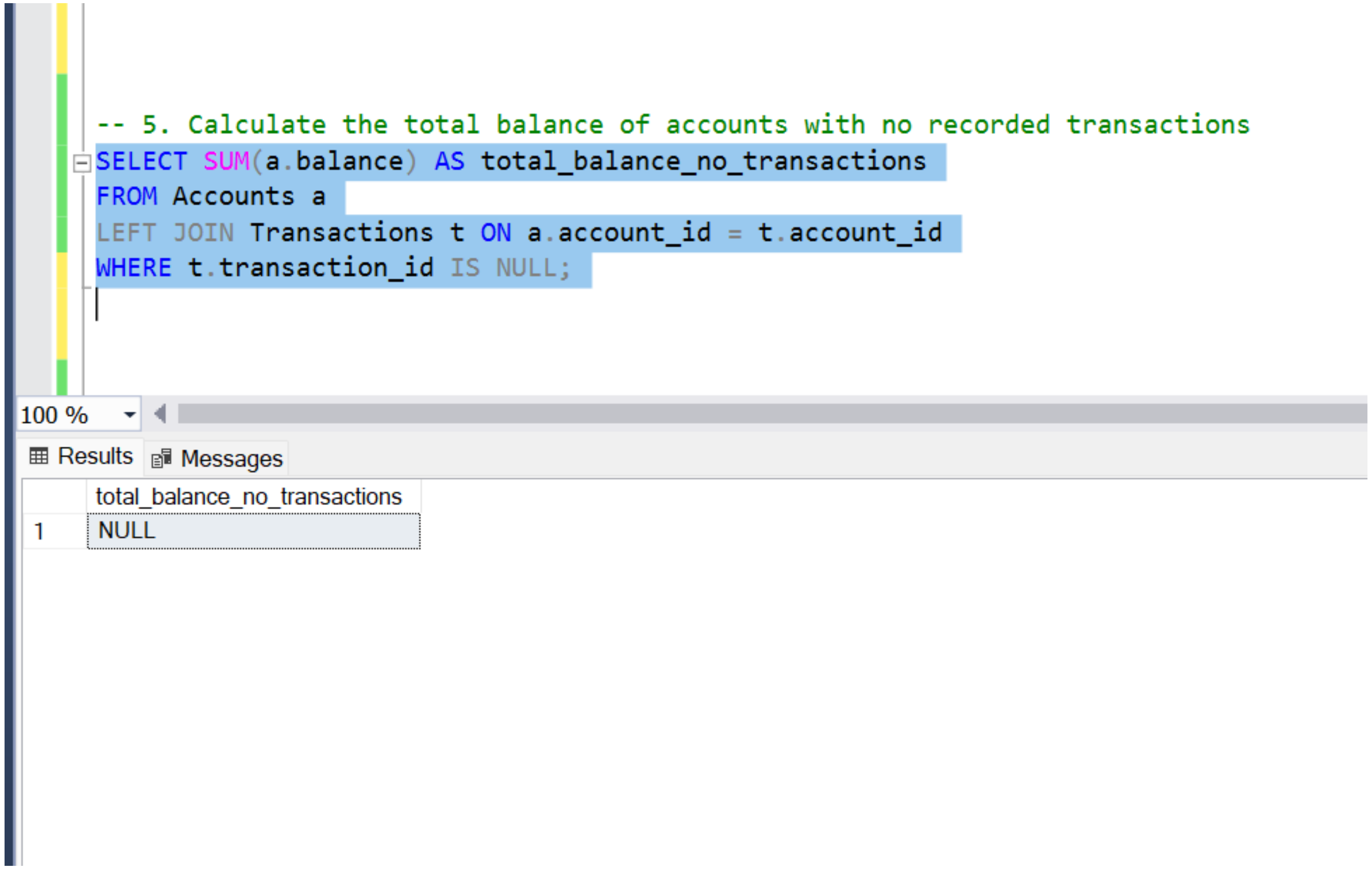
4. Identify customers who have no recorded transactions.



```sql
-- 4. Identify customers who have no recorded transactions
SELECT c.*
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;
```

customer_id	first_name	last_name	DOB	email	phone_number	address

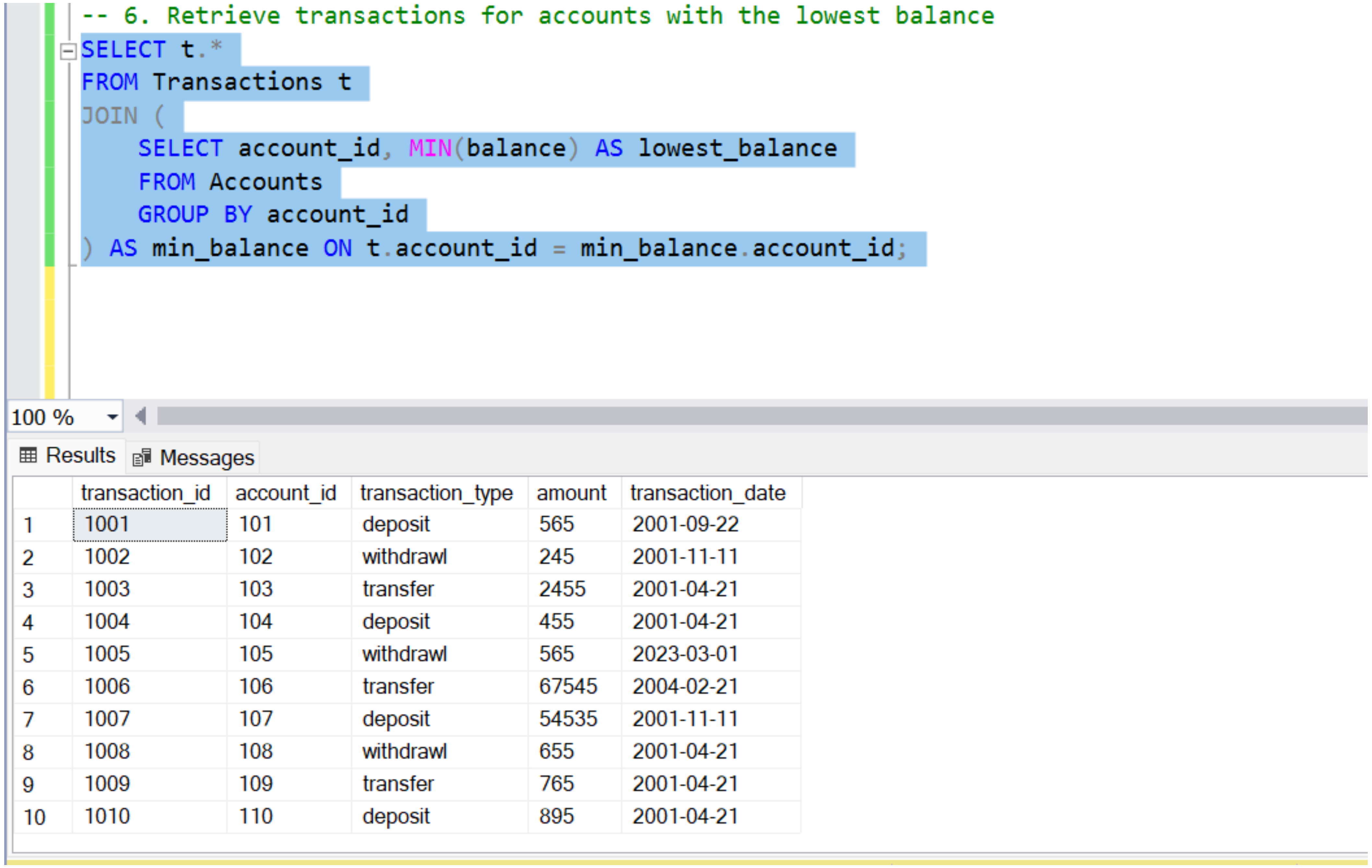
5. Calculate the total balance of accounts with no recorded transactions.



```
-- 5. 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;
```

	total_balance_no_transactions
1	NULL

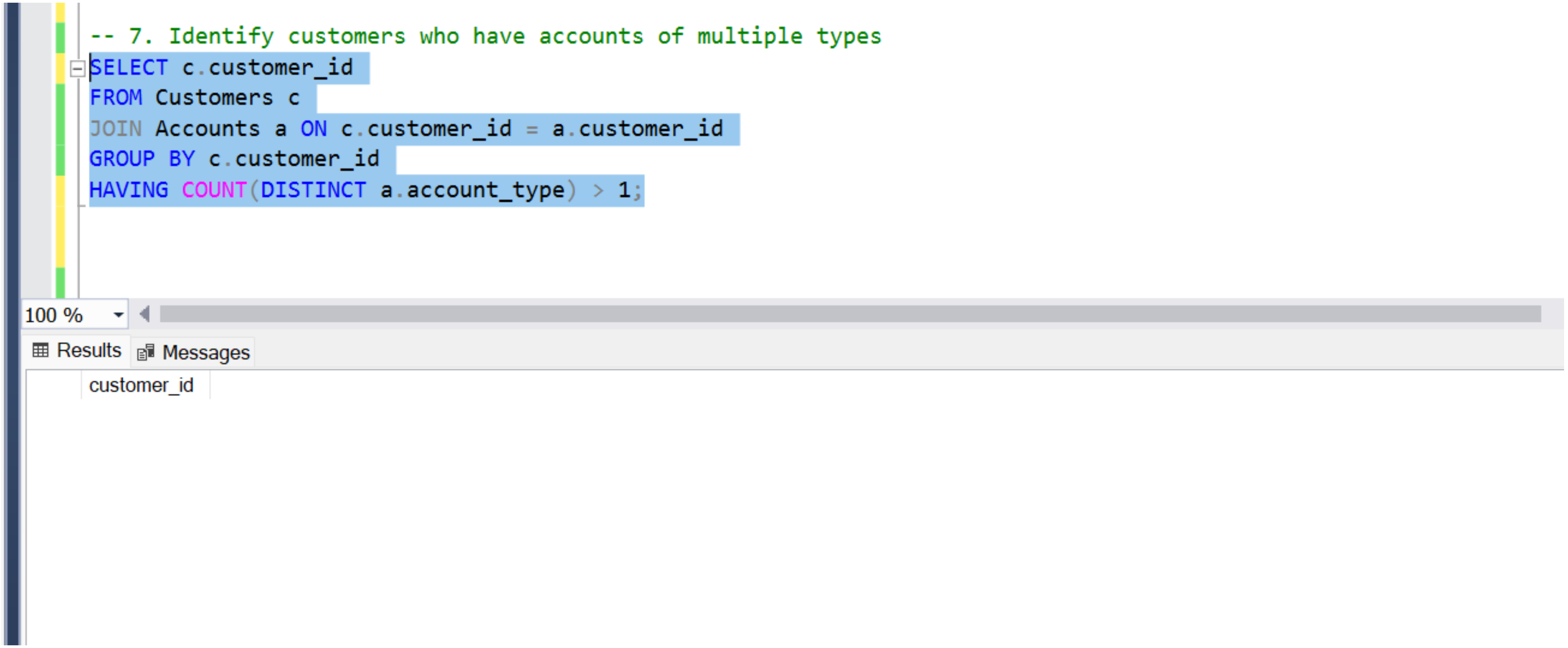
6. Retrieve transactions for accounts with the lowest balance.



```
-- 6. Retrieve transactions for accounts with the lowest balance
SELECT t.*
FROM Transactions t
JOIN (
    SELECT account_id, MIN(balance) AS lowest_balance
    FROM Accounts
    GROUP BY account_id
) AS min_balance ON t.account_id = min_balance.account_id;
```

	transaction_id	account_id	transaction_type	amount	transaction_date
1	1001	101	deposit	565	2001-09-22
2	1002	102	withdrawl	245	2001-11-11
3	1003	103	transfer	2455	2001-04-21
4	1004	104	deposit	455	2001-04-21
5	1005	105	withdrawl	565	2023-03-01
6	1006	106	transfer	67545	2004-02-21
7	1007	107	deposit	54535	2001-11-11
8	1008	108	withdrawl	655	2001-04-21
9	1009	109	transfer	765	2001-04-21
10	1010	110	deposit	895	2001-04-21

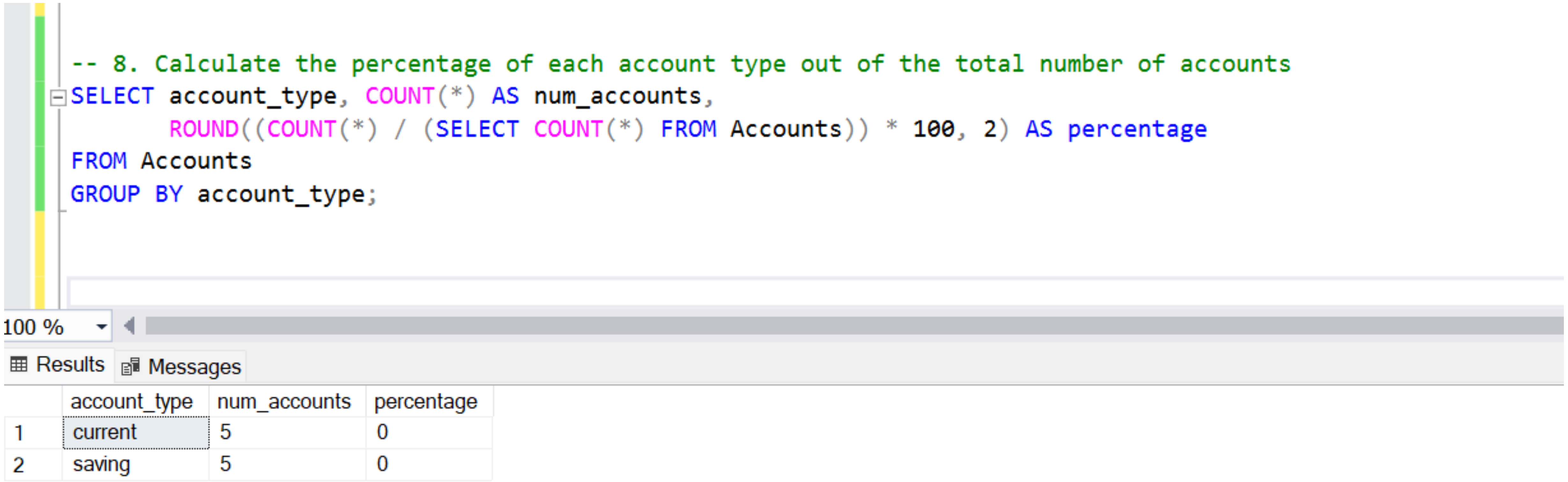
7. Identify customers who have accounts of multiple types.



```
-- 7. Identify customers who have accounts of multiple types
SELECT c.customer_id
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
GROUP BY c.customer_id
HAVING COUNT(DISTINCT a.account_type) > 1;
```

customer_id

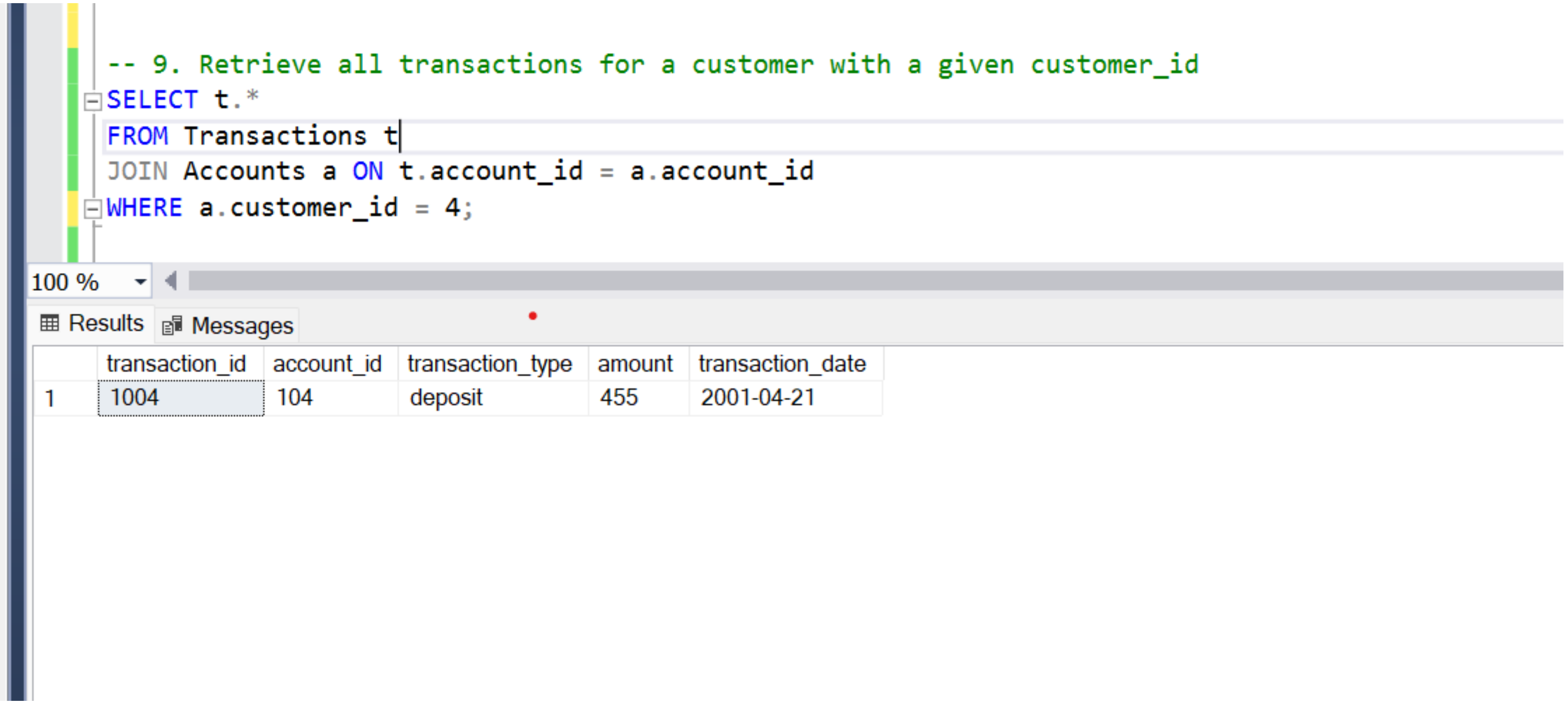
8. Calculate the percentage of each account type out of the total number of accounts.



```sql
-- 8. Calculate the percentage of each account type out of the total number of accounts
SELECT account_type, COUNT(*) AS num_accounts,
       ROUND((COUNT(*) / (SELECT COUNT(*) FROM Accounts)) * 100, 2) AS percentage
FROM Accounts
GROUP BY account_type;
```

	account_type	num_accounts	percentage
1	current	5	0
2	saving	5	0

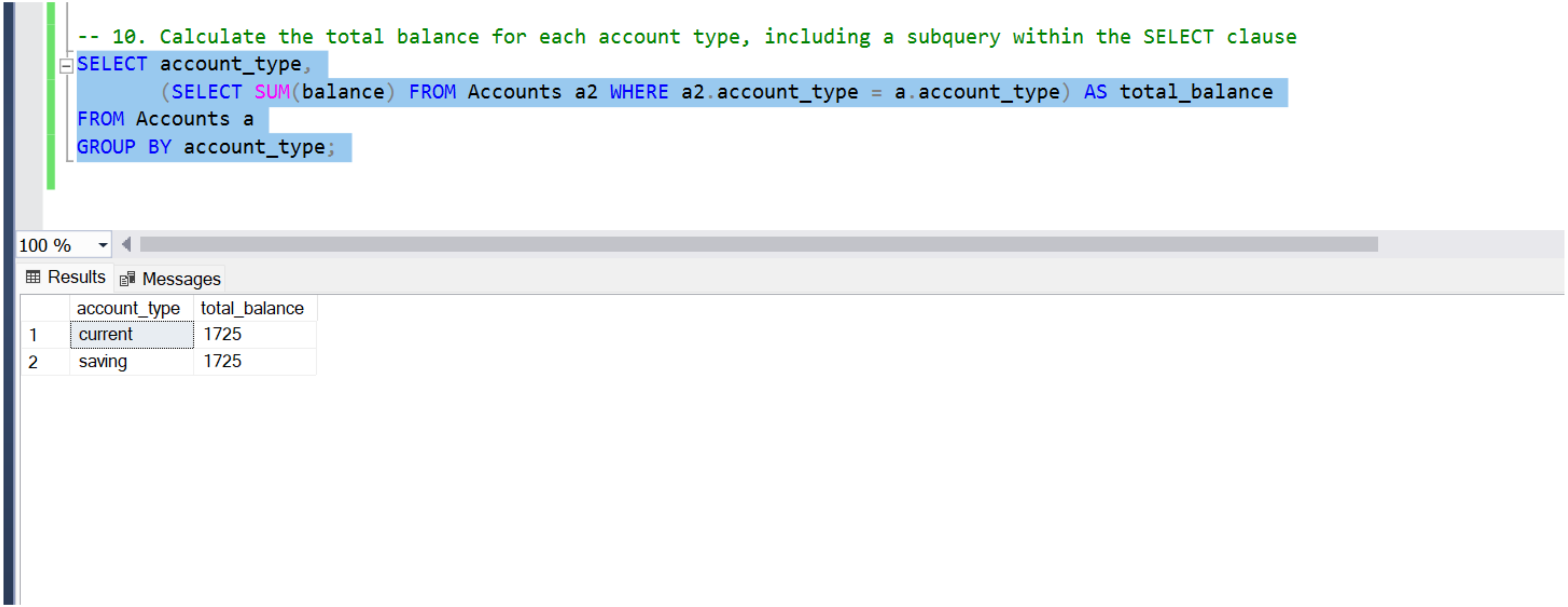
9. Retrieve all transactions for a customer with a given customer_id.



```
-- 9. Retrieve all transactions for a customer with a given customer_id
SELECT t.*
FROM Transactions t
JOIN Accounts a ON t.account_id = a.account_id
WHERE a.customer_id = 4;
```

	transaction_id	account_id	transaction_type	amount	transaction_date
1	1004	104	deposit	455	2001-04-21

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



```sql
-- 10. Calculate the total balance for each account type, including a subquery within the SELECT clause
SELECT account_type,
       (SELECT SUM(balance) FROM Accounts a2 WHERE a2.account_type = a.account_type) AS total_balance
FROM Accounts a
GROUP BY account_type;
```

	account_type	total_balance
1	current	1725
2	saving	1725