SQL Tasks:

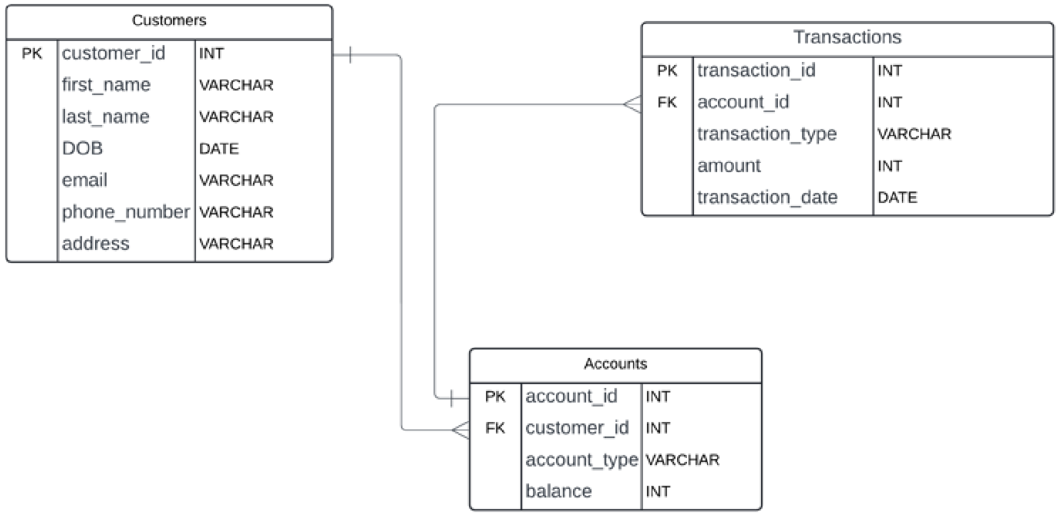
Task 1

Create the database named "HMBank"

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

```sql
USE HMBank;


-- Create Customers Table

CREATE TABLE Customers (

    customer_id INT IDENTITY(1,1) PRIMARY KEY,

    first_name NVARCHAR(50) NOT NULL,

    last_name NVARCHAR(50) NOT NULL,

    DOB DATE NOT NULL,

    email NVARCHAR(100) NOT NULL UNIQUE,

    phone_number NVARCHAR(15) NOT NULL,

    address NVARCHAR(255)

);


-- Create Accounts Table

CREATE TABLE Accounts (

    account_id INT IDENTITY(1,1) PRIMARY KEY,

    customer_id INT NOT NULL,

    account_type NVARCHAR(20) CHECK (account_type IN ('savings', 'current', 'zero_balance')),

    balance DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id) ON DELETE CASCADE

);



-- Create Transactions Table

CREATE TABLE Transactions (

    transaction_id INT IDENTITY(1,1) PRIMARY KEY,

    account_id INT NOT NULL,

    transaction_type NVARCHAR(20) CHECK (transaction_type IN ('deposit', 'withdrawal', 'transfer')),

    amount DECIMAL(10, 2) NOT NULL,

    transaction_date DATETIME DEFAULT GETDATE(),

    FOREIGN KEY (account_id) REFERENCES Accounts(account_id) ON DELETE CASCADE

);
```

ER Diagram:



Customers

PK	customer_id	INT
	first_name	VARCHAR
	last_name	VARCHAR
	DOB	DATE
	email	VARCHAR
	phone_number	VARCHAR
	address	VARCHAR

Transactions

PK	transaction_id	INT
FK	account_id	INT
	transaction_type	VARCHAR
	amount	INT
	transaction_date	DATE

Accounts

PK	account_id	INT
FK	customer_id	INT
	account_type	VARCHAR
	balance	INT

Task 2

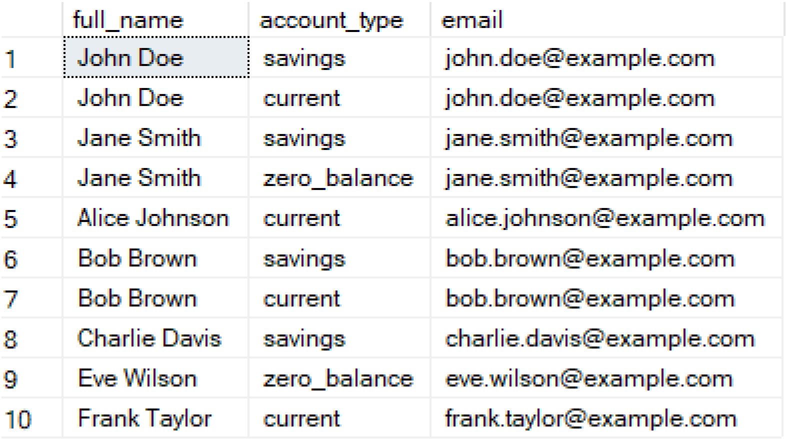
1. Insert at least 10 sample records into each of the following tables.

```
INSERT INTO Customers (first_name, last_name, DOB, email, phone_number, address)

VALUES

('John', 'Doe', '1985-06-15', 'john.doe@example.com', '1234567890', '123 Elm St, Springfield'),

('Jane', 'Smith', '1990-02-20', 'jane.smith@example.com', '0987654321', '456 Oak St, Springfield'),

('Alice', 'Johnson', '1975-12-01', 'alice.johnson@example.com', '5555555555', '789 Pine St, Springfield'),

('Bob', 'Brown', '1988-07-22', 'bob.brown@example.com', '2222222222', '321 Maple St, Springfield'),

('Charlie', 'Davis', '1995-04-30', 'charlie.davis@example.com', '3333333333', '654 Cedar St, Springfield'),

('Eve', 'Wilson', '1992-10-14', 'eve.wilson@example.com', '4444444444', '987 Birch St, Springfield'),

('Frank', 'Taylor', '1980-09-05', 'frank.taylor@example.com', '7777777777', '159 Spruce St, Springfield'),

('Grace', 'Miller', '1994-03-11', 'grace.miller@example.com', '8888888888', '753 Fir St, Springfield'),

('Hank', 'Anderson', '1982-08-25', 'hank.anderson@example.com', '6666666666', '852 Elm St, Springfield'),

('Ivy', 'Thomas', '1991-05-19', 'ivy.thomas@example.com', '9999999999', '951 Willow St, Springfield');
```

```
INSERT INTO Accounts (customer_id, account_type, balance)

VALUES

(1, 'savings', 1500.00),
(1, 'current', 2500.00),
(2, 'savings', 3000.00),
(2, 'zero_balance', 0.00),
(3, 'current', 5000.00),
(4, 'savings', 800.00),
(4, 'current', 1200.00),
(5, 'savings', 400.00),
(6, 'zero_balance', 0.00),
(7, 'current', 6000.00);

INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date)
VALUES
(1, 'deposit', 500.00, '2024-01-15 10:00:00'),
(1, 'withdrawal', 200.00, '2024-01-20 15:30:00'),
(2, 'deposit', 1000.00, '2024-01-22 11:15:00'),
(2, 'withdrawal', 500.00, '2024-01-25 09:00:00'),
(3, 'transfer', 1500.00, '2024-01-30 14:45:00'),
(4, 'deposit', 300.00, '2024-02-01 13:20:00'),
(5, 'withdrawal', 100.00, '2024-02-05 12:00:00'),
(6, 'deposit', 200.00, '2024-02-10 16:10:00'),
(7, 'transfer', 1200.00, '2024-02-15 11:30:00'),
(8, 'withdrawal', 50.00, '2024-02-20 10:50:00');
```

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

```
SELECT

    CONCAT(c.first_name, ' ', c.last_name) AS full_name,

    a.account_type,

    c.email

FROM

    Customers c

JOIN

    Accounts a ON c.customer_id = a.customer_id;
```

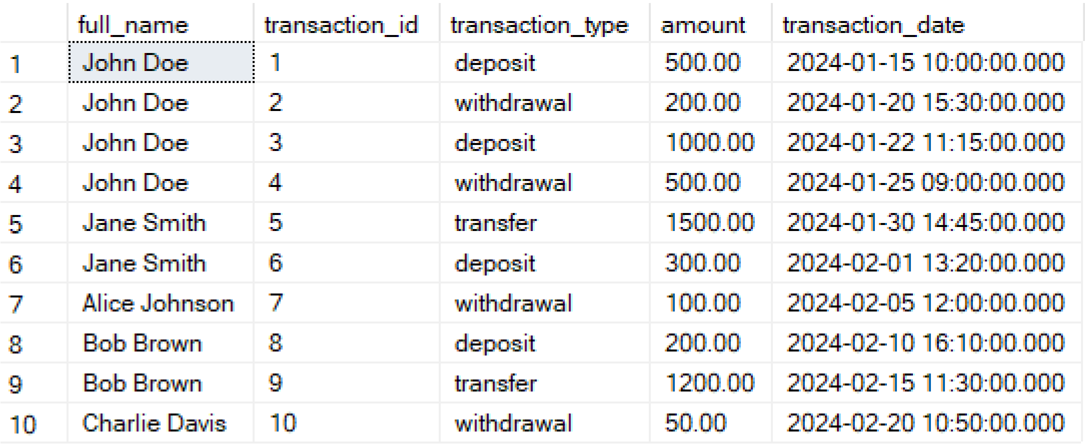


	full_name	account_type	email
1	John Doe	savings	john.doe@example.com
2	John Doe	current	john.doe@example.com
3	Jane Smith	savings	jane.smith@example.com
4	Jane Smith	zero_balance	jane.smith@example.com
5	Alice Johnson	current	alice.johnson@example.com
6	Bob Brown	savings	bob.brown@example.com
7	Bob Brown	current	bob.brown@example.com
8	Charlie Davis	savings	charlie.davis@example.com
9	Eve Wilson	zero_balance	eve.wilson@example.com
10	Frank Taylor	current	frank.taylor@example.com

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

```
SELECT

    CONCAT(c.first_name, ' ', c.last_name) AS full_name,

    t.transaction_id,

    t.transaction_type,

    t.amount,

    t.transaction_date
```

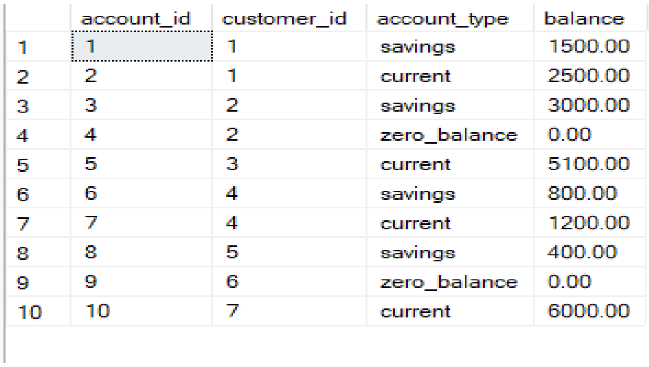
```
FROM
    Customers c
JOIN
    Accounts a ON c.customer_id = a.customer_id
JOIN
    Transactions t ON a.account_id = t.account_id;
```



	full_name	transaction_id	transaction_type	amount	transaction_date
1	John Doe	1	deposit	500.00	2024-01-15 10:00:00.000
2	John Doe	2	withdrawal	200.00	2024-01-20 15:30:00.000
3	John Doe	3	deposit	1000.00	2024-01-22 11:15:00.000
4	John Doe	4	withdrawal	500.00	2024-01-25 09:00:00.000
5	Jane Smith	5	transfer	1500.00	2024-01-30 14:45:00.000
6	Jane Smith	6	deposit	300.00	2024-02-01 13:20:00.000
7	Alice Johnson	7	withdrawal	100.00	2024-02-05 12:00:00.000
8	Bob Brown	8	deposit	200.00	2024-02-10 16:10:00.000
9	Bob Brown	9	transfer	1200.00	2024-02-15 11:30:00.000
10	Charlie Davis	10	withdrawal	50.00	2024-02-20 10:50:00.000

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

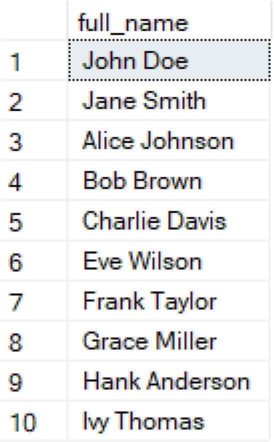
```
UPDATE Accounts
SET balance = balance + 100
WHERE account_id = 5;
```



	account_id	customer_id	account_type	balance
1	1	1	savings	1500.00
2	2	1	current	2500.00
3	3	2	savings	3000.00
4	4	2	zero_balance	0.00
5	5	3	current	5100.00
6	6	4	savings	800.00
7	7	4	current	1200.00
8	8	5	savings	400.00
9	9	6	zero_balance	0.00
10	10	7	current	6000.00

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

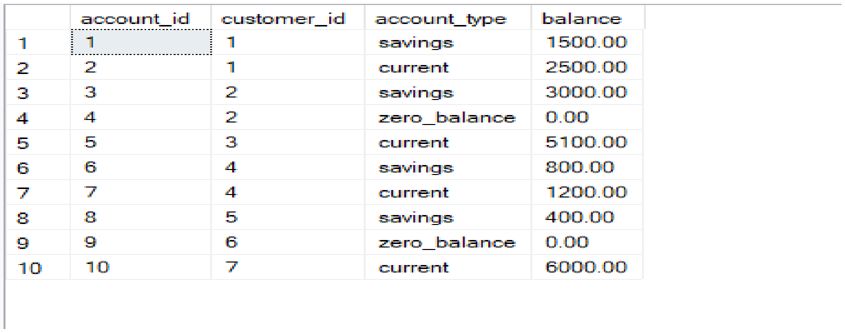
```
SELECT

    CONCAT(first_name, ' ', last_name) AS full_name

FROM

    Customers;
```



	full_name
1	John Doe
2	Jane Smith
3	Alice Johnson
4	Bob Brown
5	Charlie Davis
6	Eve Wilson
7	Frank Taylor
8	Grace Miller
9	Hank Anderson
10	Ivy Thomas

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

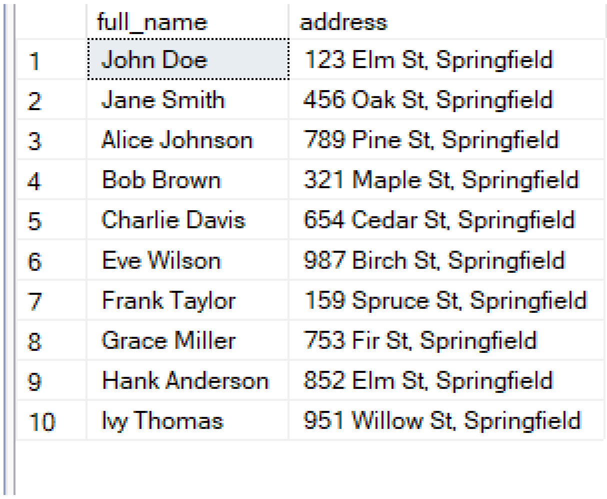
```
DELETE FROM Accounts

WHERE balance = 0 AND account_type = 'savings';
```



	account_id	customer_id	account_type	balance
1	1	1	savings	1500.00
2	2	1	current	2500.00
3	3	2	savings	3000.00
4	4	2	zero_balance	0.00
5	5	3	current	5100.00
6	6	4	savings	800.00
7	7	4	current	1200.00
8	8	5	savings	400.00
9	9	6	zero_balance	0.00
10	10	7	current	6000.00

6. Write a SQL query to Find customers living in a specific city.(ex Pune)

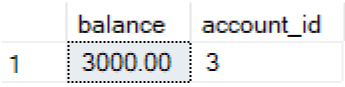
```
SELECT

    CONCAT(first_name, ' ', last_name) AS full_name,

    address

FROM

    Customers

WHERE

    address LIKE '%Springfield%';
```



	full_name	address
1	John Doe	123 Elm St, Springfield
2	Jane Smith	456 Oak St, Springfield
3	Alice Johnson	789 Pine St, Springfield
4	Bob Brown	321 Maple St, Springfield
5	Charlie Davis	654 Cedar St, Springfield
6	Eve Wilson	987 Birch St, Springfield
7	Frank Taylor	159 Spruce St, Springfield
8	Grace Miller	753 Fir St, Springfield
9	Hank Anderson	852 Elm St, Springfield
10	Ivy Thomas	951 Willow St, Springfield

7. Write a SQL query to Get the account balance for a specific account.(ex id=102)

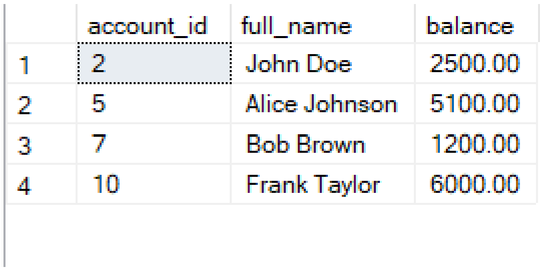
```
SELECT

    balance,account_id

FROM

    Accounts

WHERE

    account_id = 3;
```



	balance	account_id
1	3000.00	3

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

```
SELECT
    a.account_id,
    CONCAT(c.first_name, ' ', c.last_name) AS full_name,
    a.balance
FROM
    Accounts a
JOIN
    Customers c ON a.customer_id = c.customer_id
WHERE
    a.account_type = 'current' AND a.balance > 1000;
```

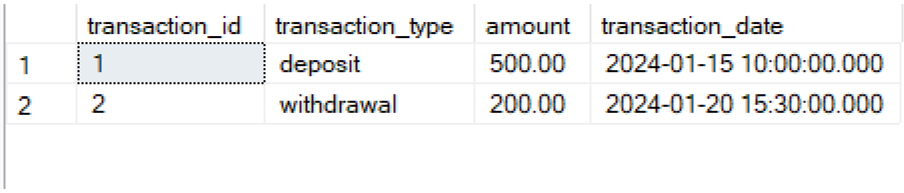


	account_id	full_name	balance
1	2	John Doe	2500.00
2	5	Alice Johnson	5100.00
3	7	Bob Brown	1200.00
4	10	Frank Taylor	6000.00

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

```
SELECT
    transaction_id,
    transaction_type,
    amount,
    transaction_date
```

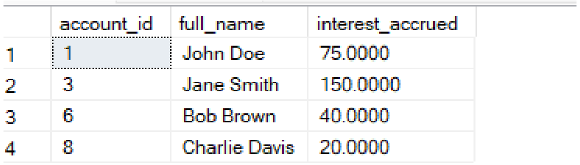
```
FROM
    Transactions
WHERE
    account_id = 1;
```



	transaction_id	transaction_type	amount	transaction_date
1	1	deposit	500.00	2024-01-15 10:00:00.000
2	2	withdrawal	200.00	2024-01-20 15:30:00.000

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

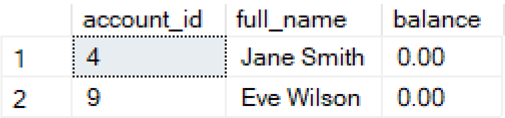
```
SELECT
    a.account_id,
    CONCAT(c.first_name, ' ', c.last_name) AS full_name,
    (a.balance * 0.05) AS interest_accrued
FROM
    Accounts a
JOIN
    Customers c ON a.customer_id = c.customer_id
WHERE
    a.account_type = 'savings';
```



	account_id	full_name	interest_accrued
1	1	John Doe	75.0000
2	3	Jane Smith	150.0000
3	6	Bob Brown	40.0000
4	8	Charlie Davis	20.0000

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

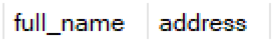
```
SELECT
    a.account_id,
    CONCAT(c.first_name, ' ', c.last_name) AS full_name,
    a.balance
FROM
    Accounts a
JOIN
    Customers c ON a.customer_id = c.customer_id
WHERE
    a.balance < 100;
```



	account_id	full_name	balance
1	4	Jane Smith	0.00
2	9	Eve Wilson	0.00

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

```
SELECT
    CONCAT(first_name, ' ', last_name) AS full_name,
    address
FROM
    Customers
WHERE
    address NOT LIKE '%Springfield%';
```

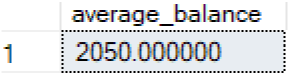


full_name	address

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

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

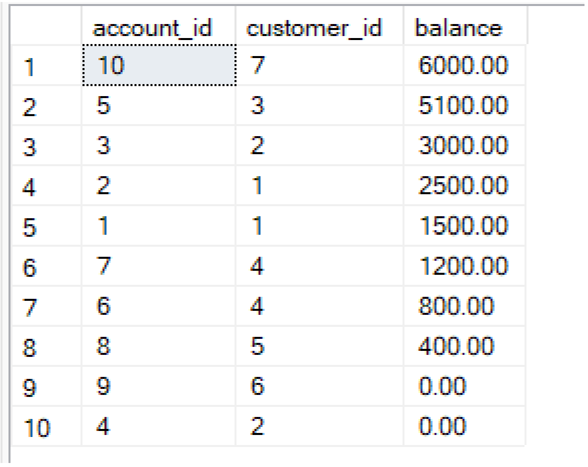
```
SELECT
    AVG(balance) AS average_balance
FROM
    Accounts;
```



	average_balance
1	2050.000000

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

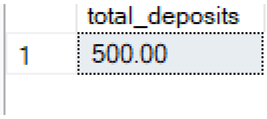
```
SELECT

    account_id,

    customer_id,

    balance

FROM

    Accounts

ORDER BY

    balance DESC

OFFSET 0 ROWS

FETCH NEXT 10 ROWS ONLY;
```



	account_id	customer_id	balance
1	10	7	6000.00
2	5	3	5100.00
3	3	2	3000.00
4	2	1	2500.00
5	1	1	1500.00
6	7	4	1200.00
7	6	4	800.00
8	8	5	400.00
9	9	6	0.00
10	4	2	0.00

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

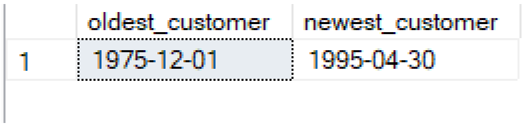
```
SELECT

   SUM(amount) AS total_deposits

FROM

   Transactions

WHERE

   transaction_type = 'deposit' AND

   CAST(transaction_date AS DATE) = '2024-01-15';
```



```
     total_deposits
1    500.00
```

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

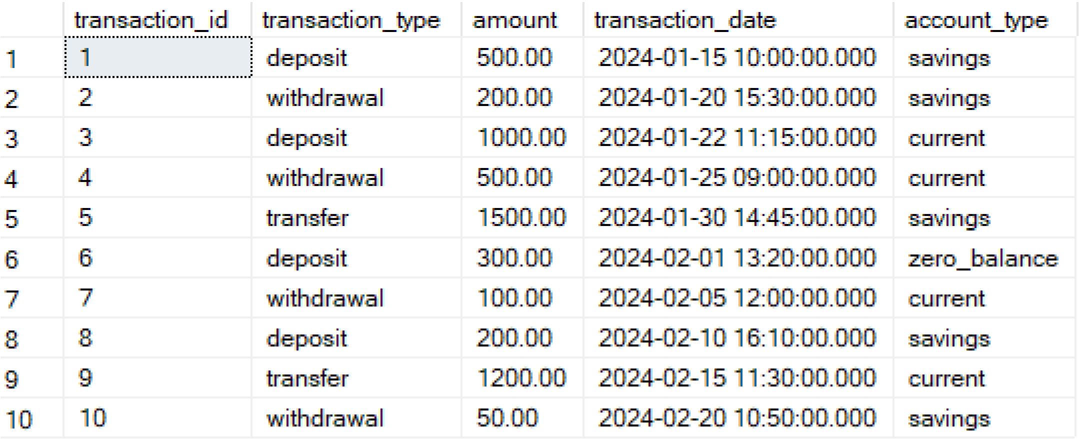
```
SELECT

   MIN(DOB) AS oldest_customer,

   MAX(DOB) AS newest_customer

FROM

   Customers;
```



```
     oldest_customer   newest_customer
1    1975-12-01        1995-04-30
```

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

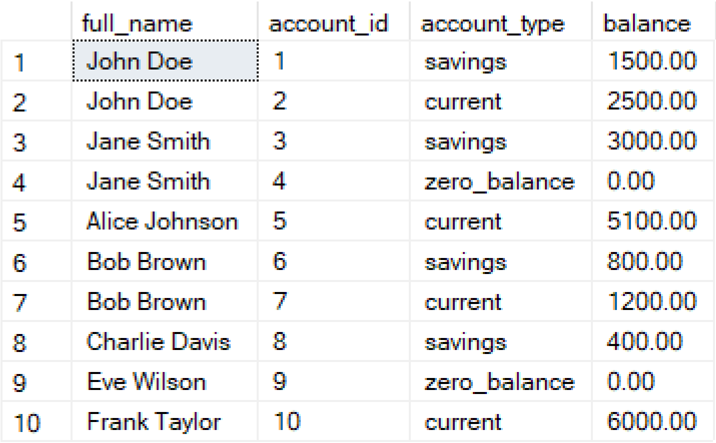
```
SELECT

   t.transaction_id,

   t.transaction_type,

   t.amount,

   t.transaction_date,
   a.account_type

FROM

   Transactions t

JOIN

   Accounts a ON t.account_id = a.account_id;
```



	transaction_id	transaction_type	amount	transaction_date	account_type
1	1	deposit	500.00	2024-01-15 10:00:00.000	savings
2	2	withdrawal	200.00	2024-01-20 15:30:00.000	savings
3	3	deposit	1000.00	2024-01-22 11:15:00.000	current
4	4	withdrawal	500.00	2024-01-25 09:00:00.000	current
5	5	transfer	1500.00	2024-01-30 14:45:00.000	savings
6	6	deposit	300.00	2024-02-01 13:20:00.000	zero_balance
7	7	withdrawal	100.00	2024-02-05 12:00:00.000	current
8	8	deposit	200.00	2024-02-10 16:10:00.000	savings
9	9	transfer	1200.00	2024-02-15 11:30:00.000	current
10	10	withdrawal	50.00	2024-02-20 10:50:00.000	savings

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

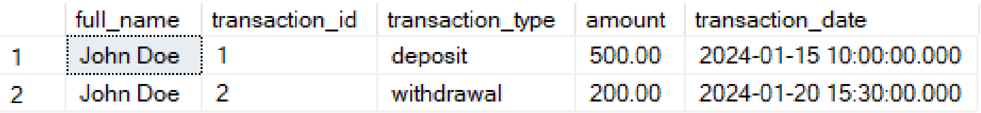
```
SELECT

    CONCAT(c.first_name, ' ', c.last_name) AS full_name,

    a.account_id,

    a.account_type,

    a.balance

FROM

    Customers c

JOIN

    Accounts a ON c.customer_id = a.customer_id;
```



	full_name	account_id	account_type	balance
1	John Doe	1	savings	1500.00
2	John Doe	2	current	2500.00
3	Jane Smith	3	savings	3000.00
4	Jane Smith	4	zero_balance	0.00
5	Alice Johnson	5	current	5100.00
6	Bob Brown	6	savings	800.00
7	Bob Brown	7	current	1200.00
8	Charlie Davis	8	savings	400.00
9	Eve Wilson	9	zero_balance	0.00
10	Frank Taylor	10	current	6000.00

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

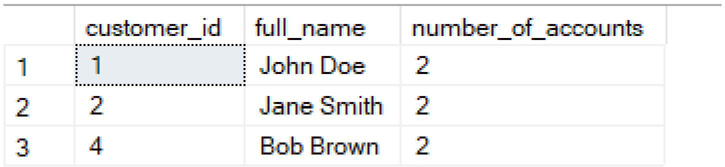
```
SELECT

   CONCAT(c.first_name, ' ', c.last_name) AS full_name,

   t.transaction_id,

   t.transaction_type,

   t.amount,

   t.transaction_date

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 = 1;
```



	full_name	transaction_id	transaction_type	amount	transaction_date
1	John Doe	1	deposit	500.00	2024-01-15 10:00:00.000
2	John Doe	2	withdrawal	200.00	2024-01-20 15:30:00.000

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

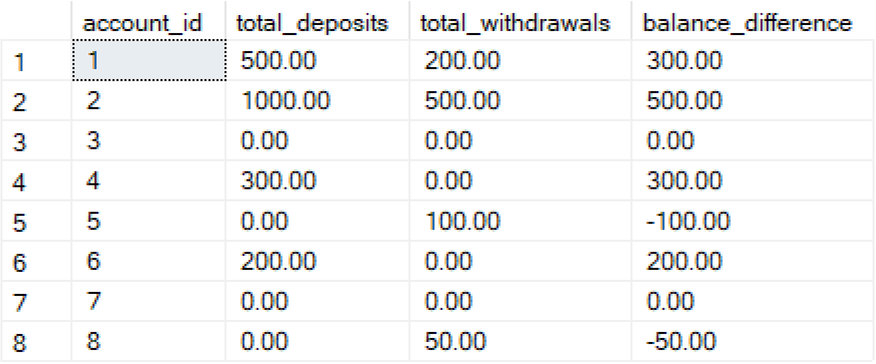
```
SELECT

   c.customer_id,

   CONCAT(c.first_name, ' ', c.last_name) AS full_name,

   COUNT(a.account_id) AS number_of_accounts

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(a.account_id) > 1;
```



	customer_id	full_name	number_of_accounts
1	1	John Doe	2
2	2	Jane Smith	2
3	4	Bob Brown	2

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

```
SELECT

   account_id,

   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 0 END) -

    SUM(CASE WHEN transaction_type = 'withdrawal' THEN amount ELSE 0 END)) AS balance_difference

FROM

   Transactions

GROUP BY

   Account_id;
```

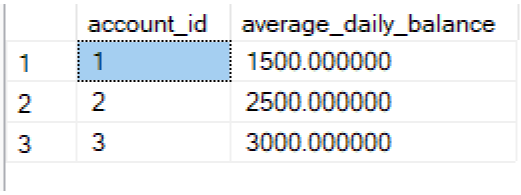


	account_id	total_deposits	total_withdrawals	balance_difference
1	1	500.00	200.00	300.00
2	2	1000.00	500.00	500.00
3	3	0.00	0.00	0.00
4	4	300.00	0.00	300.00
5	5	0.00	100.00	-100.00
6	6	200.00	0.00	200.00
7	7	0.00	0.00	0.00
8	8	0.00	50.00	-50.00

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

```
SELECT

    a.account_id,

    AVG(a.balance) AS average_daily_balance

FROM

    Accounts a

JOIN

    Transactions t ON a.account_id = t.account_id

WHERE

    t.transaction_date BETWEEN '2024-01-01' AND '2024-01-31'  -- Specify your date range.
```

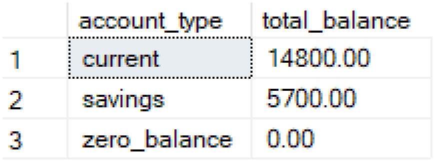
```
GROUP BY
    a.account_id;
```



	account_id	average_daily_balance
1	1	1500.000000
2	2	2500.000000
3	3	3000.000000

11. Calculate the total balance for each account type.

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

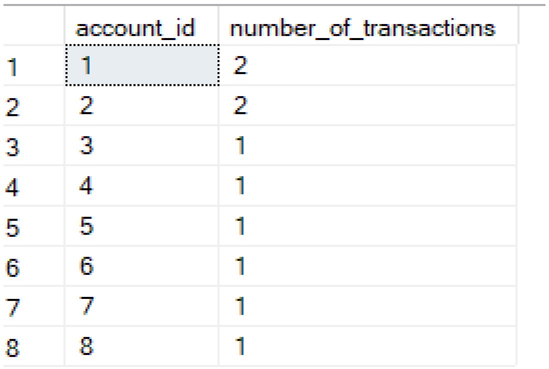


	account_type	total_balance
1	current	14800.00
2	savings	5700.00
3	zero_balance	0.00

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

```
SELECT
    a.account_id,
    COUNT(t.transaction_id) AS number_of_transactions
FROM
    Accounts a
JOIN
    Transactions t ON a.account_id = t.account_id
```

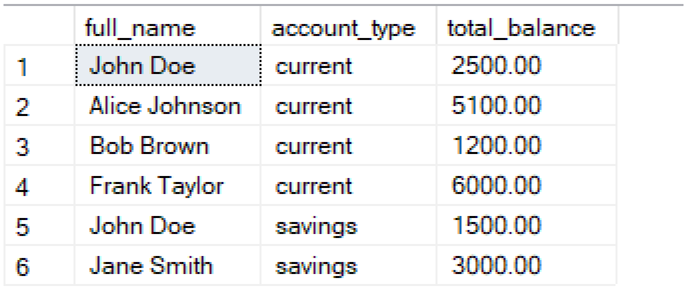
```
GROUP BY
    a.account_id
ORDER BY
    number_of_transactions DESC;
```



	account_id	number_of_transactions
1	1	2
2	2	2
3	3	1
4	4	1
5	5	1
6	6	1
7	7	1
8	8	1

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

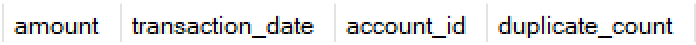
```
SELECT
    CONCAT(c.first_name, ' ', c.last_name) AS full_name,
    a.account_type,
    SUM(a.balance) AS total_balance
FROM
    Customers c
JOIN
    Accounts a ON c.customer_id = a.customer_id
GROUP BY
    c.customer_id, c.first_name, c.last_name, a.account_type
HAVING
    SUM(a.balance) > 1000;  -- Specify the desired threshol
```



	full_name	account_type	total_balance
1	John Doe	current	2500.00
2	Alice Johnson	current	5100.00
3	Bob Brown	current	1200.00
4	Frank Taylor	current	6000.00
5	John Doe	savings	1500.00
6	Jane Smith	savings	3000.00

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

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

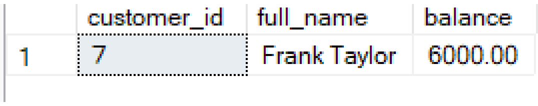


amount	transaction_date	account_id	duplicate_count

Tasks 4: Subquery and its type:

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

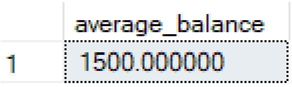
```
SELECT
    c.customer_id,
    CONCAT(c.first_name, ' ', c.last_name) AS full_name,
    a.balance
FROM
    Customers c
JOIN
    Accounts a ON c.customer_id = a.customer_id
WHERE
    a.balance = (SELECT MAX(balance) FROM Accounts);
```



	customer_id	full_name	balance
1	7	Frank Taylor	6000.00

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

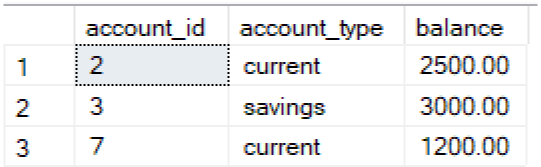
```
SELECT
    AVG(balance) AS average_balance
FROM
    Accounts
WHERE
    customer_id IN (
        SELECT
            customer_id
        FROM
            Accounts
        GROUP BY
            customer_id
        HAVING
            COUNT(account_id) > 1
    );
```



	average_balance
1	1500.000000

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

```
SELECT
    a.account_id,
    a.account_type,
    a.balance
FROM
    Accounts a
WHERE
    a.account_id IN (
        SELECT
            t.account_id
        FROM
            Transactions t
        WHERE
            t.amount > (SELECT AVG(amount) FROM Transactions)
    );
```

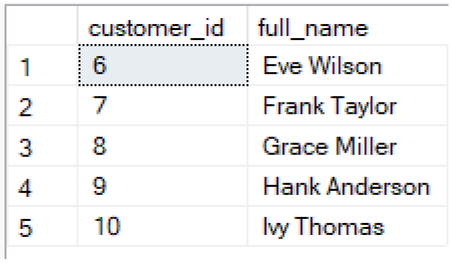


	account_id	account_type	balance
1	2	current	2500.00
2	3	savings	3000.00
3	7	current	1200.00

4. Identify customers who have no recorded transactions.

```
SELECT
```

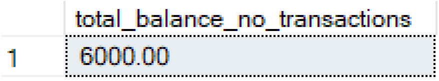
```
    c.customer_id,
    CONCAT(c.first_name, ' ', c.last_name) AS full_name
FROM
    Customers c
LEFT JOIN
    Accounts a ON c.customer_id = a.customer_id
LEFT JOIN
    Transactions t ON a.account_id = t.account_id
WHERE
    t.transaction_id IS NULL;
```



	customer_id	full_name
1	6	Eve Wilson
2	7	Frank Taylor
3	8	Grace Miller
4	9	Hank Anderson
5	10	Ivy Thomas

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

```
SELECT
    SUM(a.balance) AS total_balance_no_transactions
FROM
    Accounts a
WHERE
    a.account_id NOT IN (
        SELECT
            DISTINCT t.account_id
        FROM
            Transactions t
    );
```

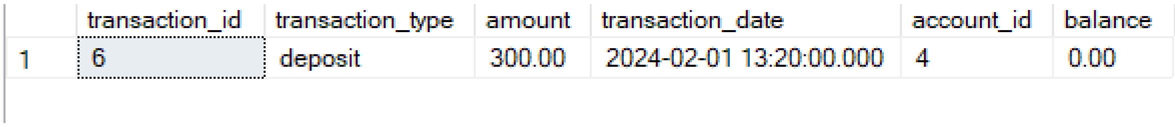


	total_balance_no_transactions
1	6000.00

6. Retrieve transactions for accounts with the lowest balance.

```
SELECT
    t.transaction_id,
    t.transaction_type,
    t.amount,
    t.transaction_date,
    a.account_id,
    a.balance
```

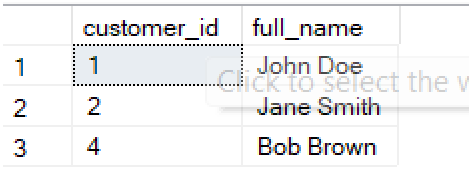
```
FROM
    Transactions t
JOIN
    Accounts a ON t.account_id = a.account_id
WHERE
    a.balance = (SELECT MIN(balance) FROM Accounts);
```



	transaction_id	transaction_type	amount	transaction_date	account_id	balance
1	6	deposit	300.00	2024-02-01 13:20:00.000	4	0.00

7. Identify customers who have accounts of multiple types.

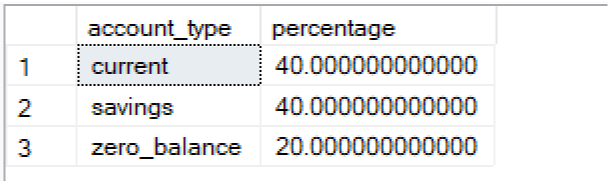
```
SELECT
    c.customer_id,
    CONCAT(c.first_name, ' ', c.last_name) AS full_name
FROM
    Customers c
JOIN
    Accounts a ON c.customer_id = a.customer_id
GROUP BY
    c.customer_id, c.first_name, c.last_name
HAVING
    COUNT(DISTINCT a.account_type) > 1;
```



	customer_id	full_name
1	1	John Doe
2	2	Jane Smith
3	4	Bob Brown

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

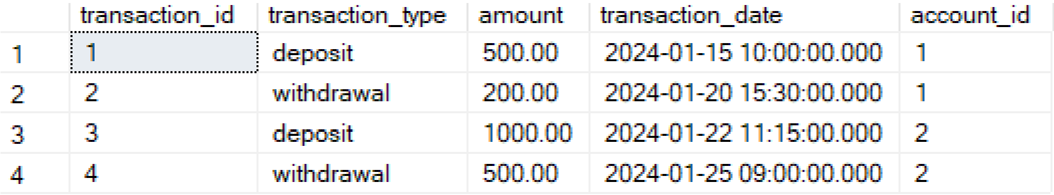
```
SELECT
    a.account_type,
    COUNT(a.account_id) * 100.0 / (SELECT COUNT(*) FROM Accounts) AS percentage
FROM
    Accounts a
GROUP BY
    a.account_type;
```



	account_type	percentage
1	current	40.000000000000
2	savings	40.000000000000
3	zero_balance	20.000000000000

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

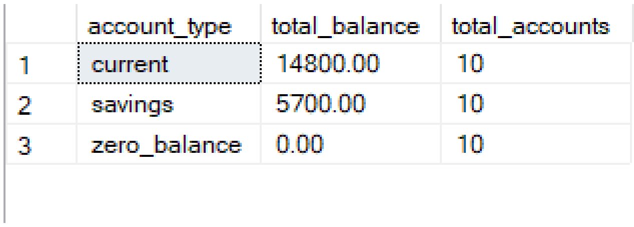
```
SELECT
    t.transaction_id,
    t.transaction_type,
    t.amount,
    t.transaction_date,
    a.account_id
FROM
    Transactions t
JOIN
    Accounts a ON t.account_id = a.account_id
WHERE
    a.customer_id = 1;
```



	transaction_id	transaction_type	amount	transaction_date	account_id
1	1	deposit	500.00	2024-01-15 10:00:00.000	1
2	2	withdrawal	200.00	2024-01-20 15:30:00.000	1
3	3	deposit	1000.00	2024-01-22 11:15:00.000	2
4	4	withdrawal	500.00	2024-01-25 09:00:00.000	2

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

```
SELECT
    account_type,
    SUM(balance) AS total_balance,
    (SELECT COUNT(*) FROM Accounts) AS total_accounts
FROM
    Accounts
GROUP BY
    account_type;
```



	account_type	total_balance	total_accounts
1	current	14800.00	10
2	savings	5700.00	10
3	zero_balance	0.00	10