Tasks 1: Database Design:

1. Create the database named "HMBank"
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

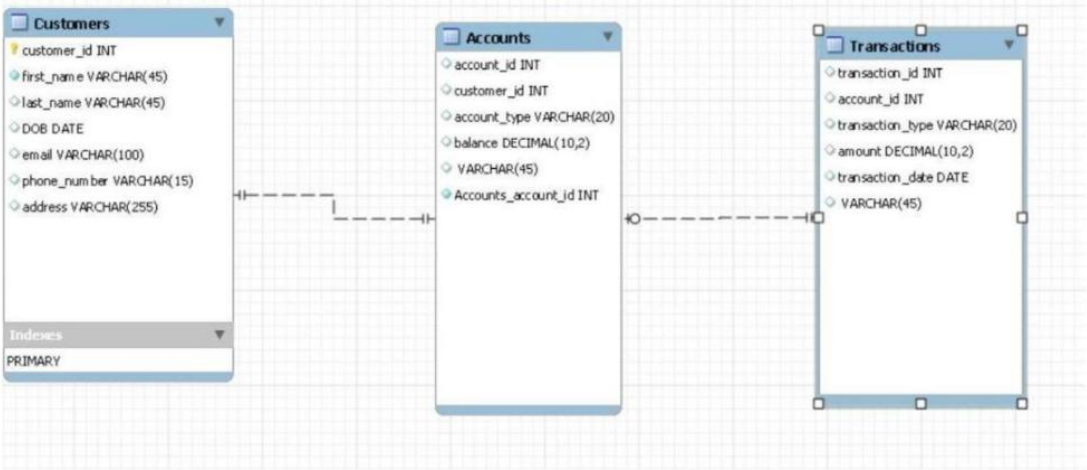
```sql
CREATE DATABASE IF NOT EXISTS HMBank;
USE HMBank;

CREATE TABLE Customers (
    customer_id INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    DOB DATE,
    email VARCHAR(255),
    phone_number VARCHAR(20),
    address VARCHAR(255)
);

CREATE TABLE Accounts (
    account_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    account_type VARCHAR(50),
    balance DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
```

Output

Action Output

#	Time	Action	Message
93	17:08:29	select * from products LIMIT 0, 1000	10 row(s) returned
94	17:08:54	INSERT INTO Products (ProductID, ProductName, Description, Price) VALUES (11, 'New Gadget', 'High-tech ...	1 row(s) affected
95	17:08:54	select * from products LIMIT 0, 1000	11 row(s) returned
96	19:50:45	CREATE DATABASE IF NOT EXISTS HMBank	1 row(s) affected
97	19:50:45	USE HMBank	0 row(s) affected
98	19:50:45	CREATE TABLE Customers (customer_id INT PRIMARY KEY AUTO_INCREMENT, first_name VARCH...	0 row(s) affected
99	19:50:45	CREATE TABLE Accounts (account_id INT PRIMARY KEY AUTO_INCREMENT, customer_id INT, a...	0 row(s) affected
100	19:50:45	CREATE TABLE Transactions (transaction_id INT PRIMARY KEY AUTO_INCREMENT, account_id IN...	0 row(s) affected

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



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 (first_name, last_name, DOB, email, phone_number, address)
VALUES
    ('John', 'Doe', '1990-05-15', 'john.doe@email.com', '123-456-7890', '123 Main St'),
    ('Jane', 'Smith', '1985-08-22', 'jane.smith@email.com', '987-654-3210', '456 Oak Ave'),
    ('Alice', 'Johnson', '1992-11-30', 'alice.johnson@email.com', '555-123-4567', '789 Pine St'),
    ('Bob', 'Williams', '1980-03-10', 'bob.williams@email.com', '333-555-7777', '101 Elm St'),
    ('Eva', 'Anderson', '1995-07-18', 'eva.anderson@email.com', '444-999-1111', '202 Maple Ave'),
    ('Michael', 'Brown', '1988-01-25', 'michael.brown@email.com', '777-222-3333', '303 Birch St'),
    ('Sophia', 'Miller', '1993-04-05', 'sophia.miller@email.com', '888-444-5555', '404 Cedar Ave'),
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1990-05-15	john.doe@email.com	123-456-7890	123 Main St
2	Jane	Smith	1985-08-22	jane.smith@email.com	987-654-3210	456 Oak Ave
3	Alice	Johnson	1992-11-30	alice.johnson@email.com	555-123-4567	789 Pine St
4	Bob	Williams	1980-03-10	bob.williams@email.com	333-555-7777	101 Elm St
5	Eva	Anderson	1995-07-18	eva.anderson@email.com	444-999-1111	202 Maple Ave
6	Michael	Brown	1988-01-25	michael.brown@email.com	777-222-3333	303 Birch St
7	Sophia	Miller	1993-04-05	sophia.miller@email.com	888-444-5555	404 Cedar Ave
8	Daniel	Garcia	1987-09-12	daniel.garcia@email.com	666-777-8888	505 Oak St
9	Olivia	Martinez	1991-12-08	olivia.martinez@email.com	999-111-2222	606 Pine Ave
10	William	Taylor	1983-06-20	william.taylor@email.com	111-333-4444	707 Maple St
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
INSERT INTO Accounts (customer_id, account_type, balance)
VALUES
    (1, 'savings', 5000.00),
    (1, 'current', 1000.00),
    (2, 'savings', 7500.00),
    (3, 'current', 3000.00),
    (4, 'savings', 12000.00),
    (5, 'current', 2000.00),
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

account_id	customer_id	account_type	balance
1	1	savings	5000.00
2	1	current	1000.00
3	2	savings	7500.00
4	3	current	3000.00
5	4	savings	12000.00
6	5	current	2000.00
7	6	savings	6000.00
8	7	current	8000.00
9	8	savings	9000.00
10	9	current	1500.00
NULL	NULL	NULL	NULL

```sql
INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date)
VALUES
    (1, 'deposit', 1000.00, '2024-01-16'),
    (2, 'withdrawal', 500.00, '2024-01-17'),
    (3, 'transfer', 2000.00, '2024-01-18'),
    (4, 'deposit', 1500.00, '2024-01-19'),
    (5, 'withdrawal', 800.00, '2024-01-20'),
    (6, 'transfer', 1000.00, '2024-01-21'),
    (7, 'deposit', 1200.00, '2024-01-22'),
```

transaction_id	account_id	transaction_type	amount	transaction_date
1	1	deposit	1000.00	2024-01-16
2	2	withdrawal	500.00	2024-01-17
3	3	transfer	2000.00	2024-01-18
4	4	deposit	1500.00	2024-01-19
5	5	withdrawal	800.00	2024-01-20
6	6	transfer	1000.00	2024-01-21
7	7	deposit	1200.00	2024-01-22
8	8	withdrawal	700.00	2024-01-23
9	9	transfer	2500.00	2024-01-24
10	10	deposit	300.00	2024-01-25
NULL	NULL	NULL	NULL	NULL

Write SQL queries for the following tasks:

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

```sql
SELECT
    CONCAT(first_name, ' ', last_name) AS customer_name,
    account_type,
    email
FROM
    Customers
JOIN
    Accounts ON Customers.customer_id = Accounts.customer_id;
```

customer_name	account_type	email
John Doe	savings	john.doe@email.com
John Doe	current	john.doe@email.com
Jane Smith	savings	jane.smith@email.com
Alice Johnson	current	alice.johnson@email.com
Bob Williams	savings	bob.williams@email.com
Eva Anderson	current	eva.anderson@email.com
Michael Brown	savings	michael.brown@email.com
Sophia Miller	current	sophia.miller@email.com
Daniel Garcia	savings	daniel.garcia@email.com
Olivia Martinez	current	olivia.martinez@email.com

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

```sql
SELECT
    CONCAT(C.first_name, ' ', C.last_name) AS customer_name,
    A.account_type,
    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;
```

customer_name	account_type	transaction_type	amount	transaction_date
John Doe	savings	deposit	1000.00	2024-01-16
John Doe	current	withdrawal	500.00	2024-01-17
Jane Smith	savings	transfer	2000.00	2024-01-18
Alice Johnson	current	deposit	1500.00	2024-01-19
Bob Williams	savings	withdrawal	800.00	2024-01-20
Eva Anderson	current	transfer	1000.00	2024-01-21
Michael Brown	savings	deposit	1200.00	2024-01-22
Sophia Miller	current	withdrawal	700.00	2024-01-23
Daniel Garcia	savings	transfer	2500.00	2024-01-24
Olivia Martinez	current	deposit	300.00	2024-01-25

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

```sql
UPDATE Accounts
SET balance = balance + '400'
WHERE account_id = '1';

select * from Accounts
```

account_id	customer_id	account_type	balance
1	1	savings	5400.00
2	1	current	1000.00
3	2	savings	7500.00
4	3	current	3000.00
5	4	savings	12000.00
6	5	current	2000.00
7	6	savings	6000.00
8	7	current	8000.00
9	8	savings	9000.00
10	9	current	1500.00
NULL	NULL	NULL	NULL

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

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

customer_id	full_name
1	John Doe
2	Jane Smith
3	Alice Johnson
4	Bob Williams
5	Eva Anderson
6	Michael Brown
7	Sophia Miller
8	Daniel Garcia
9	Olivia Martinez
10	William Taylor

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

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

select * from Accounts
```

account_id	customer_id	account_type	balance
1	1	savings	5400.00
2	1	current	1000.00
3	2	savings	7500.00
4	3	current	3000.00
5	4	savings	12000.00
6	5	current	2000.00
7	6	savings	6000.00
8	7	current	8000.00
9	8	savings	9000.00
10	9	current	1500.00
NULL	NULL	NULL	NULL

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

```sql
SELECT
    customer_id,
    first_name,
    last_name,
    DOB,
    email,
    phone_number,
    address
FROM
    Customers
WHERE
    address = '303 Birch St';
```

customer_id	first_name	last_name	DOB	email	phone_number	address
6	Michael	Brown	1988-01-25	michael.brown@email.com	777-222-3333	303 Birch St
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```sql
SELECT
    account_id,
    account_type,
    balance
FROM
    Accounts
WHERE
    account_id = 1;
```

account_id	account_type	balance
1	savings	5400.00
NULL	NULL	NULL

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

```sql
SELECT
    account_id,
    customer_id,
    account_type,
    balance
FROM
    Accounts
WHERE
    account_type = 'current' AND balance > 1000;
```

account_id	customer_id	account_type	balance
4	3	current	3000.00
6	5	current	2000.00
8	7	current	8000.00
10	9	current	1500.00
NULL	NULL	NULL	NULL

1500.00

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

```sql
SELECT
    transaction_id,
    account_id,
    transaction_type,
    amount,
    transaction_date
FROM
    Transactions
WHERE
    account_id = 1;
```

transaction_id	account_id	transaction_type	amount	transaction_date
1	1	deposit	1000.00	2024-01-16
NULL	NULL	NULL	NULL	NULL

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

given interest rate.

```
UPDATE Accounts
SET balance = balance + (balance * 0.10)
WHERE account_type = 'savings';

Select * from Accounts
```

account_id	customer_id	account_type	balance
1	1	savings	5940.00
2	1	current	1000.00
3	2	savings	8250.00
4	3	current	3000.00
5	4	savings	13200.00
6	5	current	2000.00
7	6	savings	6600.00
8	7	current	8000.00
9	8	savings	9900.00
10	9	current	1500.00
NULL	NULL	NULL	NULL

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

```
SELECT *
FROM Accounts
WHERE balance < 9000;
```

account_id	customer_id	account_type	balance
1	1	savings	5940.00
2	1	current	1000.00
3	2	savings	8250.00
4	3	current	3000.00
6	5	current	2000.00
7	6	savings	6600.00
8	7	current	8000.00
10	9	current	1500.00
NULL	NULL	NULL	NULL

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

```
SELECT
    customer_id,
    first_name,
    last_name,
    DOB,
    email,
    phone_number,
    address
FROM
    Customers
WHERE
    address <> '456 Oak Ave';
```

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1990-05-15	john.doe@email.com	123-456-7890	123 Main St
3	Alice	Johnson	1992-11-30	alice.johnson@email.com	555-123-4567	789 Pine St
4	Bob	Williams	1980-03-10	bob.williams@email.com	333-555-7777	101 Elm St
5	Eva	Anderson	1995-07-18	eva.anderson@email.com	444-999-1111	202 Maple Ave
6	Michael	Brown	1988-01-25	michael.brown@email.com	777-222-3333	303 Birch St
7	Sophia	Miller	1993-04-05	sophia.miller@email.com	888-444-5555	404 Cedar Ave
8	Daniel	Garcia	1987-09-12	daniel.garcia@email.com	666-777-8888	505 Oak St
9	Olivia	Martinez	1991-12-08	olivia.martinez@email.com	999-111-2222	606 Pine Ave
10	William	Taylor	1983-06-20	william.taylor@email.com	111-333-4444	707 Maple St
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

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

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

Result Grid

average_balance
5939.000000

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

```sql
SELECT *
FROM Accounts
ORDER BY balance DESC
LIMIT 10;
```

Result Grid

account_id	customer_id	account_type	balance
5	4	savings	13200.00
9	8	savings	9900.00
3	2	savings	8250.00
8	7	current	8000.00
7	6	savings	6600.00
1	1	savings	5940.00
4	3	current	3000.00
6	5	current	2000.00
10	9	current	1500.00
2	1	current	1000.00
NULL	NULL	NULL	NULL

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

```sql
SELECT
 t.transaction_date,
 SUM(t.amount) AS total_deposits
FROM
 Transactions t
WHERE
 t.transaction_type = 'deposit'
 AND t.transaction_date = '2024-01-16';
```

Result Grid

transaction_date	total_deposits
2024-01-16	1000.00

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

```sql
SELECT
 MIN(DOB) AS oldest_customer_dob,
 MAX(DOB) AS newest_customer_dob
FROM
 Customers;
```

oldest_customer_dob	newest_customer_dob
1980-03-10	1995-07-18

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

```sql
SELECT
 t.transaction_id,
 t.account_id,
 a.account_type,
 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;
```

transaction_id	account_id	account_type	transaction_type	amount	transaction_date
1	1	savings	deposit	1000.00	2024-01-16
2	2	current	withdrawal	500.00	2024-01-17
3	3	savings	transfer	2000.00	2024-01-18
4	4	current	deposit	1500.00	2024-01-19
5	5	savings	withdrawal	800.00	2024-01-20
6	6	current	transfer	1000.00	2024-01-21
7	7	savings	deposit	1200.00	2024-01-22
8	8	current	withdrawal	700.00	2024-01-23
9	9	savings	transfer	2500.00	2024-01-24
10	10	current	deposit	300.00	2024-01-25

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

```sql
SELECT
 c.customer_id,
 c.first_name,
 c.last_name,
 c.DOB,
 c.email,
 c.phone_number,
 c.address,
 a.account_id,
 a.account_type,
 a.balance
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	account_type	balance
1	John	Doe	1990-05-15	john.doe@email.com	123-456-7890	123 Main St	1	savings	5940.00
1	John	Doe	1990-05-15	john.doe@email.com	123-456-7890	123 Main St	2	current	1000.00
2	Jane	Smith	1985-08-22	jane.smith@email.com	987-654-3210	456 Oak Ave	3	savings	8250.00
3	Alice	Johnson	1992-11-30	alice.johnson@email.com	555-123-4567	789 Pine St	4	current	3000.00
4	Bob	Williams	1980-03-10	bob.williams@email.com	333-555-7777	101 Elm St	5	savings	13200.00
5	Eva	Anderson	1995-07-18	eva.anderson@email.com	444-999-1111	202 Maple Ave	6	current	2000.00
6	Michael	Brown	1988-01-25	michael.brown@email.com	777-222-3333	303 Birch St	7	savings	6600.00
7	Sophia	Miller	1993-04-05	sophia.miller@email.com	888-444-5555	404 Cedar Ave	8	current	8000.00
8	Daniel	Garcia	1987-09-12	daniel.garcia@email.com	666-777-8888	505 Oak St	9	savings	9900.00
9	Olivia	Martinez	1991-12-08	olivia.martinez@email.com	999-111-2222	606 Pine Ave	10	current	1500.00

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

specific account.

```
    t.transaction_id,
    t.account_id,
    a.account_type,
    t.transaction_type,
    t.amount,
    t.transaction_date,
    c.customer_id,
    c.first_name,
    c.last_name,
    c.DOB,
    c.email,
    c.phone_number,
    c.address
FROM
    Transactions t
JOIN
    Accounts a ON t.account_id = a.account_id
JOIN
    Customers c ON a.customer_id = c.customer_id
WHERE
    t.account_id = 1;
```

transaction_id	account_id	account_type	transaction_type	amount	transaction_date	customer_id	first_name	last_name	DOB	email	phone_numbe
1	1	savings	deposit	1000.00	2024-01-16	1	John	Doe	1990-05-15	john.doe@email.com	123-456-7890

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

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    COUNT(a.account_id) AS num_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	first_name	last_name	num_accounts
1	John	Doe	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 -amount END) AS
net_difference
FROM
    Transactions
GROUP BY
    account_id;
```

account_id	total_deposits	total_withdrawals	net_difference
1	1000.00	0.00	1000.00
2	0.00	500.00	-500.00
3	0.00	0.00	-2000.00
4	1500.00	0.00	1500.00
5	0.00	800.00	-800.00
6	0.00	0.00	-1000.00
7	1200.00	0.00	1200.00
8	0.00	700.00	-700.00
9	0.00	0.00	-2500.00
10	300.00	0.00	300.00

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

period.

```sql
SELECT
 t.account_id,
 AVG(t.balance) AS average_daily_balance
FROM (
 SELECT
 account_id,
 transaction_date,
 SUM(amount) OVER (PARTITION BY account_id ORDER BY transaction_date) AS balance
 FROM
 Transactions
) t
GROUP BY
 t.account id;
```

account_id	average_daily_balance
1	1000.000000
2	500.000000
3	2000.000000
4	1500.000000
5	800.000000
6	1000.000000
7	1200.000000
8	700.000000
9	2500.000000
10	300.000000

11. Calculate the total balance for each account type.

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

account_type	total_balance
savings	43890.00
current	15500.00

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

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

account_id	num_transactions
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

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

```sql
SELECT
 c.customer_id,
 c.first_name,
 c.last_name,
 SUM(a.balance) AS aggregate_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
ORDER BY
 aggregate_balance DESC;
```

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

customer_id	first_name	last_name	aggregate_balance
4	Bob	Williams	13200.00
8	Daniel	Garcia	9900.00
2	Jane	Smith	8250.00
7	Sophia	Miller	8000.00
1	John	Doe	6940.00
6	Michael	Brown	6600.00
3	Alice	Johnson	3000.00
5	Eva	Anderson	2000.00
9	Olivia	Martinez	1500.00

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

```sql
 transaction_id,
 account_id,
 transaction_type,
 amount,
 transaction_date
FROM
 Transactions
WHERE
 (amount, transaction_date, account_id) IN (
 SELECT
 amount,
 transaction_date,
 account_id
 FROM
 Transactions
 GROUP BY
 amount,
 transaction_date,
 account_id
 HAVING
 COUNT(transaction_id) > 1
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

transaction_id	account_id	transaction_type	amount	transaction_date
NULL	NULL	NULL	NULL	NULL

Tasks 4: Subquery and its type:

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

```
SELECT
 c.customer_id,
 c.first_name,
 c.last_name,
 c.DOB,
 c.email,
 c.phone_number,
 c.address
FROM
 Customers c
JOIN
 Accounts a ON c.customer_id = a.customer_id
WHERE
 a.balance = (SELECT MAX(balance) FROM Accounts);
```

Result Grid

customer_id	first_name	last_name	DOB	email	phone_number	address
4	Bob	Williams	1980-03-10	bob.williams@email.com	333-555-7777	101 Elm St

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

```
 c.customer_id,
 c.first_name,
 c.last_name,
 AVG(a.balance) AS average_balance
FROM
 Customers c
JOIN
 Accounts a ON c.customer_id = a.customer_id
WHERE
 c.customer_id IN (
 SELECT
 customer_id
 FROM
 Accounts
 GROUP BY
 customer_id
 HAVING
 COUNT(account_id) > 1
 )
GROUP BY
 c.customer_id, c.first_name, c.last_name;
```

Result Grid

customer_id	first_name	last_name	average_balance
1	John	Doe	3470.000000

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

```
SELECT
 a.account_id,
 a.customer_id,
 a.account_type,
 a.balance
FROM
 Accounts a
JOIN
 Transactions t ON a.account_id = t.account_id
WHERE
 t.amount > (SELECT AVG(amount) FROM Transactions)
```

Result Grid

account_id	customer_id	account_type	balance
3	2	savings	8250.00
4	3	current	3000.00
7	6	savings	6600.00
9	8	savings	9900.00

4. Identify customers who have no recorded transactions.

```sql
SELECT
 customer_id,
 first_name,
 last_name,
 DOB,
 email,
 phone_number,
 address
FROM
 Customers c
WHERE
 NOT EXISTS (
 SELECT 1
 FROM Transactions t
 WHERE t.account_id IN (SELECT account_id FROM Accounts WHERE customer_id =
c.customer_id)
```

customer_id	first_name	last_name	DOB	email	phone_number	address
10	William	Taylor	1983-06-20	william.taylor@email.com	111-333-4444	707 Maple St
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```sql
SELECT
 SUM(balance) AS total_balance_no_transactions
FROM
 Accounts a
WHERE
 NOT EXISTS (
 SELECT 1
 FROM Transactions t
 WHERE t.account_id = a.account_id
 );
```

total_balance_no_transactions
NULL

6. Retrieve transactions for accounts with the lowest balance.

```sql
SELECT
 t.transaction_id,
 t.account_id,
 t.transaction_type,
 t.amount,
 t.transaction_date
FROM
 Transactions t
JOIN (
 SELECT
 account_id
 FROM
 Accounts
 ORDER BY
 balance ASC
 LIMIT 1
) a ON t.account_id = a.account_id;
```

transaction_id	account_id	transaction_type	amount	transaction_date
2	2	withdrawal	500.00	2024-01-17

7. Identify customers who have accounts of multiple types.

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

customer_id	first_name	last_name
1	John	Doe

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

```sql
SELECT
 account_type,
 COUNT(*) AS num_accounts,
 (COUNT(*) * 100.0 / (SELECT COUNT(*) FROM Accounts)) AS percentage
FROM
 Accounts
GROUP BY
 account_type;
```

account_type	num_accounts	percentage
savings	5	50.00000
current	5	50.00000

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

```sql
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.customer id = 2;
```

transaction_id	account_id	transaction_type	amount	transaction_date
3	3	transfer	2000.00	2024-01-18

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

Clause

```
SELECT
 account_type,
 (SELECT SUM(balance) FROM Accounts a WHERE a.account_type = t.account_type) AS
total_balance
FROM
 Accounts t
GROUP BY
 account_type;
```

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

account_type	total_balance
savings	43890.00
current	15500.00