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

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

```
CREATE DATABASE HMBank;

use HMBank;
```

```
DESC Customers;
```

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

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	auto_increment
first_name	varchar(50)	NO		NULL	
last_name	varchar(50)	NO		NULL	
DOB	date	YES		NULL	
email	varchar(100)	NO	UNI	NULL	
phone_number	varchar(20)	YES		NULL	
address	varchar(200)	YES		NULL	

```
33 •  DESC Accounts;
```

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

Field	Type	Null	Key	Default	Extra
account_id	int	NO	PRI	NULL	auto_increment
customer_id	int	NO	MUL	NULL	
account_type	enum('savings','current','zero_balance')	NO		NULL	
balance	decimal(10,2)	NO		0.00	

```
34 •  DESC Transactions;
```

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

Field	Type	Null	Key	Default	Extra
transaction_id	int	NO	PRI	NULL	auto_increment
account_id	int	NO	MUL	NULL	
transaction_type	enum('deposit','withdrawal','transfer')	NO		NULL	
amount	decimal(10,2)	NO		NULL	
transaction_date	date	NO		NULL	

```
5  • CREATE TABLE Customers (
6        customer_id INT PRIMARY KEY AUTO_INCREMENT,
7        first_name VARCHAR(50) NOT NULL,
8        last_name VARCHAR(50) NOT NULL,
9        DOB DATE,
10       email VARCHAR(100) NOT NULL UNIQUE,
11       phone_number VARCHAR(20),
12       address VARCHAR(200)
13   );
14
15 • CREATE TABLE Accounts (
16       account_id INT PRIMARY KEY AUTO_INCREMENT,
17       customer_id INT NOT NULL,
18       account_type ENUM('savings', 'current', 'zero_balance') NOT NULL,
19       balance DECIMAL(10,2) NOT NULL DEFAULT 0,
20       FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
21   );
22
23 • CREATE TABLE Transactions (
24       transaction_id INT PRIMARY KEY AUTO_INCREMENT,
25       account_id INT NOT NULL,
26       transaction_type ENUM('deposit', 'withdrawal', 'transfer') NOT NULL,
27       amount DECIMAL(10,2) NOT NULL,
28       transaction_date DATE NOT NULL,
29       FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
30   );
```

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', '1980-01-01', 'john.doe@example.com', '+1234567890', '1 Main Street'),
       ('Jane', 'Doe', '1985-02-02', 'jane.doe@example.com', '+9876543210', '2 Elm Street'),
       ('Alice', 'Smith', '1990-03-03', 'alice.smith@example.com', '+0987654321', '3 Oak Street'),
       ('Bob', 'Johnson', '1995-04-04', 'bob.johnson@example.com', '+1230987654', '4 Maple Street'),
       ('Mary', 'Williams', '2000-05-05', 'mary.williams@example.com', '+5432109876', '5 Pine Street'),
       ('David', 'Brown', '2005-06-06', 'david.brown@example.com', '+9876540987', '6 Cedar Street'),
       ('Jessica', 'Miller', '2010-07-07', 'jessica.miller@example.com', '+0987098765', '7 Ash Street'),
       ('Thomas', 'Garcia', '2015-08-08', 'thomas.garcia@example.com', '+1239876540', '8 Elm Street'),
       ('Sarah', 'Jones', '2020-09-09', 'sarah.jones@example.com', '+9870987098', '9 Oak Street'),
       ('Michael', 'Anderson', '2023-10-10', 'michael.anderson@example.com', '+0987654321', '10 Maple Street');

SELECT * FROM Customers;
```

Result Grid

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1980-01-01	john.doe@example.com	+1234567890	1 Main Street
2	Jane	Doe	1985-02-02	jane.doe@example.com	+9876543210	2 Elm Street
3	Alice	Smith	1990-03-03	alice.smith@example.com	+0987654321	3 Oak Street
4	Bob	Johnson	1995-04-04	bob.johnson@example.com	+1230987654	4 Maple Street
5	Mary	Williams	2000-05-05	mary.williams@example.com	+5432109876	5 Pine Street
6	David	Brown	2005-06-06	david.brown@example.com	+9876540987	6 Cedar Street
7	Jessica	Miller	2010-07-07	jessica.miller@example.com	+0987098765	7 Ash Street
8	Thomas	Garcia	2015-08-08	thomas.garcia@example.com	+1239876540	8 Elm Street
9	Sarah	Jones	2020-09-09	sarah.jones@example.com	+9870987098	9 Oak Street
10	Michael	Anderson	2023-10-10	michael.anderson@example.com	+0987654321	10 Maple Street
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Krushnakumar Homraj Patle

```sql
INSERT INTO Accounts (customer_id, account_type, balance)
VALUES
(1, 'savings', 1000.00),
(2, 'current', 500.00),
(3, 'savings', 2000.00),
(4, 'current', 750.00),
(5, 'zero_balance', 0.00),
(6, 'savings', 3500.00),
(7, 'current', 1200.00),
(8, 'savings', 4800.00),
(9, 'current', 2350.00),
(10, 'zero_balance', 0.00);
SELECT * FROM Accounts;
```

Result Grid | Filter Rows: | Edit: | Expor

account_id	customer_id	account_type	balance
1	1	savings	1000.00
2	2	current	500.00
3	3	savings	2000.00
4	4	current	750.00
5	5	zero_balance	0.00
6	6	savings	3600.00
7	7	current	1200.00
8	8	savings	4800.00
9	9	current	2350.00
10	10	zero_balance	0.00
NULL	NULL	NULL	NULL

```sql
INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date)
VALUES
(1, 'deposit', 100.00, '2023-10-26'),
(2, 'withdrawal', 50.00, '2023-10-27'),
(3, 'transfer', 25.00, '2023-10-28'),
(4, 'deposit', 75.00, '2023-10-29'),
(5, 'withdrawal', 20.00, '2023-10-30'),
(6, 'transfer', 15.00, '2023-10-31'),
(7, 'deposit', 125.00, '2023-11-01'),
(8, 'withdrawal', 35.00, '2023-11-02'),
(9, 'transfer', 40.00, '2023-11-03'),
(10, 'deposit', 150.00, '2023-11-04');

SELECT * FROM Transactions;
```

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

transaction_id	account_id	transaction_type	amount	transaction_date
1	1	deposit	100.00	2023-10-26
2	2	withdrawal	50.00	2023-10-27
3	3	transfer	25.00	2023-10-28
4	4	deposit	75.00	2023-10-29
5	5	withdrawal	20.00	2023-10-30
6	6	transfer	15.00	2023-10-31
7	7	deposit	125.00	2023-11-01
8	8	withdrawal	35.00	2023-11-02
9	9	transfer	40.00	2023-11-03
10	10	deposit	150.00	2023-11-04
NULL	NULL	NULL	NULL	NULL

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

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

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

first_name	last_name	email	account_type
John	Doe	john.doe@example.com	savings
Jane	Doe	jane.doe@example.com	current
Alice	Smith	alice.smith@example.com	savings
Bob	Johnson	bob.johnson@example.com	current
Mary	Williams	mary.williams@example.com	zero_balance
David	Brown	david.brown@example.com	savings
Jessica	Miller	jessica.miller@example.com	current
Thomas	Garcia	thomas.garcia@example.com	savings
Sarah	Jones	sarah.jones@example.com	current
Michael	Anderson	michael.anderson@example.com	zero_balance

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

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

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

transaction_type	amount	transaction_date	first_name	last_name
deposit	100.00	2023-10-26	John	Doe
withdrawal	50.00	2023-10-27	Jane	Doe
transfer	25.00	2023-10-28	Alice	Smith
deposit	75.00	2023-10-29	Bob	Johnson
withdrawal	20.00	2023-10-30	Mary	Williams
transfer	15.00	2023-10-31	David	Brown
deposit	125.00	2023-11-01	Jessica	Miller
withdrawal	35.00	2023-11-02	Thomas	Garcia
transfer	40.00	2023-11-03	Sarah	Jones
deposit	150.00	2023-11-04	Michael	Anderson

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

```
UPDATE Accounts
SET balance = balance + 100.00
WHERE account_id = 6;
```

5. Write a SQL query to Combine first and last names of customers as a full name.

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

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

full_name
John Doe
Jane Doe
Alice Smith
Bob Johnson
Mary Williams
David Brown
Jessica Miller
Thomas Garcia
Sarah Jones
Michael Anderson

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

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

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

```
SELECT *
FROM Customers
WHERE address LIKE '%Maple%';
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address
4	Bob	Johnson	1995-04-04	bob.johnson@example.com	+1230987654	4 Maple Street
10	Michael	Anderson	2023-10-10	michael.anderson@example.com	+0987654321	10 Maple Street
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```
SELECT balance
FROM Accounts
WHERE account_id = 4;
```

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

balance
750.00

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

Limit to 1000 rows

```
SELECT *
FROM Accounts
WHERE account_type = 'current' AND balance > 1000.00;
```

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

account_id	customer_id	account_type	balance
7	7	current	1200.00
9	9	current	2350.00
NULL	NULL	NULL	NULL

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

```sql
SELECT *
FROM Transactions
WHERE account_id = 7;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

transaction_id	account_id	transaction_type	amount	transaction_date
7	7	deposit	125.00	2023-11-01
NULL	NULL	NULL	NULL	NULL

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

```sql
SELECT account_id, balance * interest_rate AS interest
FROM Accounts
WHERE account_type = 'savings';
```

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

transaction_id	account_id	transaction_type	amount	transaction_date

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

```sql
SELECT *
FROM Accounts
WHERE balance < 0;
```

Result Grid | Filter Rows: | Edit:

account_id	customer_id	account_type	balance
NULL	NULL	NULL	NULL

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

```sql
SELECT *
FROM Customers
WHERE address NOT LIKE '%Maple%';
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1980-01-01	john.doe@example.com	+1234567890	1 Main Street
2	Jane	Doe	1985-02-02	jane.doe@example.com	+9876543210	2 Elm Street
3	Alice	Smith	1990-03-03	alice.smith@example.com	+0987654321	3 Oak Street
5	Mary	Williams	2000-05-05	mary.williams@example.com	+5432109876	5 Pine Street
6	David	Brown	2005-06-06	david.brown@example.com	+9876540987	6 Cedar Street
7	Jessica	Miller	2010-07-07	jessica.miller@example.com	+0987098765	7 Ash Street
8	Thomas	Garcia	2015-08-08	thomas.garcia@example.com	+1239876540	8 Elm Street
9	Sarah	Jones	2020-09-09	sarah.jones@example.com	+9870987098	9 Oak Street
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 | Filter Rows: | Export: | Wrap Cell Content:

average_balance
1620.000000

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 | Filter Rows: | Edit:

account_id	customer_id	account_type	balance
8	8	savings	4800.00
6	6	savings	3600.00
9	9	current	2350.00
3	3	savings	2000.00
7	7	current	1200.00
1	1	savings	1000.00
4	4	current	750.00
2	2	current	500.00
5	5	zero_balance	0.00
10	10	zero_balance	0.00
NULL	NULL	NULL	NULL

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

```sql
SELECT DATE(transaction_date) AS transaction_date, SUM(amount) AS total_deposits
FROM Transactions
WHERE transaction_type = 'deposit'
GROUP BY DATE(transaction_date);
```

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

transaction_date	total_deposits
2023-10-26	100.00
2023-10-29	75.00
2023-11-01	125.00
2023-11-04	150.00

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

```sql
SELECT *
FROM Customers
ORDER BY DOB ASC, DOB DESC
LIMIT 1, 1;
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address
2	Jane	Doe	1985-02-02	jane.doe@example.com	+9876543210	2 Elm Street
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```sql
SELECT t.*, a.account_type
FROM Transactions t
INNER JOIN Accounts a ON t.account_id = a.account_id;
```

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

transaction_id	account_id	transaction_type	amount	transaction_date	account_type
1	1	deposit	100.00	2023-10-26	savings
2	2	withdrawal	50.00	2023-10-27	current
3	3	transfer	25.00	2023-10-28	savings
4	4	deposit	75.00	2023-10-29	current
5	5	withdrawal	20.00	2023-10-30	zero_balance
6	6	transfer	15.00	2023-10-31	savings
7	7	deposit	125.00	2023-11-01	current
8	8	withdrawal	35.00	2023-11-02	savings
9	9	transfer	40.00	2023-11-03	current
10	10	deposit	150.00	2023-11-04	zero_balance

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

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

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

customer_id	first_name	last_name	DOB	email	phone_number	address	account_id	customer_id	account_type	balance
1	John	Doe	1980-01-01	john.doe@example.com	+1234567890	1 Main Street	1	1	savings	1000.00
2	Jane	Doe	1985-02-02	jane.doe@example.com	+9876543210	2 Elm Street	2	2	current	500.00
3	Alice	Smith	1990-03-03	alice.smith@example.com	+0987654321	3 Oak Street	3	3	savings	2000.00
4	Bob	Johnson	1995-04-04	bob.johnson@example.com	+1230987654	4 Maple Street	4	4	current	750.00
5	Mary	Williams	2000-05-05	mary.williams@example.com	+5432109876	5 Pine Street	5	5	zero_balance	0.00
6	David	Brown	2005-06-06	david.brown@example.com	+9876540987	6 Cedar Street	6	6	savings	3600.00
7	Jessica	Miller	2010-07-07	jessica.miller@example.com	+0987098765	7 Ash Street	7	7	current	1200.00
8	Thomas	Garcia	2015-08-08	thomas.garcia@example.com	+1239876540	8 Elm Street	8	8	savings	4800.00
9	Sarah	Jones	2020-09-09	sarah.jones@example.com	+9870987098	9 Oak Street	9	9	current	2350.00
10	Michael	Anderson	2023-10-10	michael.anderson@example.com	+0987654321	10 Maple Street	10	10	zero_balance	0.00

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

```sql
SELECT t.*, c.first_name, c.last_name
FROM Transactions t
INNER JOIN Accounts a ON t.account_id = a.account_id
INNER JOIN Customers c ON a.customer_id = c.customer_id;
```

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

transaction_id	account_id	transaction_type	amount	transaction_date	first_name	last_name
1	1	deposit	100.00	2023-10-26	John	Doe
2	2	withdrawal	50.00	2023-10-27	Jane	Doe
3	3	transfer	25.00	2023-10-28	Alice	Smith
4	4	deposit	75.00	2023-10-29	Bob	Johnson
5	5	withdrawal	20.00	2023-10-30	Mary	Williams
6	6	transfer	15.00	2023-10-31	David	Brown
7	7	deposit	125.00	2023-11-01	Jessica	Miller
8	8	withdrawal	35.00	2023-11-02	Thomas	Garcia
9	9	transfer	40.00	2023-11-03	Sarah	Jones
10	10	deposit	150.00	2023-11-04	Michael	Anderson

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

```sql
SELECT c.*
FROM Customers c
INNER JOIN Accounts a ON c.customer_id = a.customer_id
GROUP BY c.customer_id
HAVING COUNT(a.account_id) > 1;
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address

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

```sql
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
FROM Transactions;
```

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

total_deposits	total_withdrawals
450.00	105.00

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

```sql
SELECT account_id, DATE_FORMAT(transaction_date, '%Y-%m-%d') AS transaction_date, AVG(balance) AS average_daily_balance
FROM (
    SELECT account_id, transaction_date, MAX(balance) AS balance
    FROM Transactions
    GROUP BY account_id, DATE(transaction_date)
) AS daily_balances
GROUP BY account_id, transaction_date
HAVING DATE(transaction_date) BETWEEN '2023-10-01' AND '2023-10-31';
```

11. Calculate the total balance for each account type.

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

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

account_type	total_balance
savings	11400.00
current	4800.00
zero_balance	0.00

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

```sql
SELECT account_id, COUNT(*) AS transaction_count
FROM Transactions
GROUP BY account_id
ORDER BY transaction_count DESC
LIMIT 1;
```

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

account_id	transaction_count
1	1

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

```sql
SELECT c.*, a.account_type, a.balance
FROM Customers c
INNER JOIN Accounts a ON c.customer_id = a.customer_id
WHERE a.balance > (SELECT AVG(balance) FROM Accounts)
ORDER BY c.last_name ASC;
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address	account_type	balance
6	David	Brown	2005-06-06	david.brown@example.com	+9876540987	6 Cedar Street	savings	3600.00
8	Thomas	Garcia	2015-08-08	thomas.garcia@example.com	+1239876540	8 Elm Street	savings	4800.00
9	Sarah	Jones	2020-09-09	sarah.jones@example.com	+9870987098	9 Oak Street	current	2350.00
3	Alice	Smith	1990-03-03	alice.smith@example.com	+0987654321	3 Oak Street	savings	2000.00

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

```sql
SELECT *
FROM Transactions
WHERE (account_id, transaction_type, amount, transaction_date) IN (
    SELECT account_id, transaction_type, amount, transaction_date
    FROM Transactions
    GROUP BY account_id, transaction_type, amount, transaction_date
    HAVING COUNT(*) > 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.*, a.balance
FROM Customers c
INNER JOIN Accounts a ON c.customer_id = a.customer_id
WHERE a.balance = (
    SELECT MAX(balance)
    FROM Accounts
);
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address	balance
8	Thomas	Garcia	2015-08-08	thomas.garcia@example.com	+1239876540	8 Elm Street	4800.00

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

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

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

average_balance

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

```
SELECT a.*, t.*
FROM Accounts a
INNER JOIN Transactions t ON a.account_id = t.account_id
WHERE t.amount > (
    SELECT AVG(amount)
    FROM Transactions
);
```

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

account_id	customer_id	account_type	balance	transaction_id	account_id	transaction_type	amount	transaction_date
1	1	savings	1000.00	1	1	deposit	100.00	2023-10-26
4	4	current	750.00	4	4	deposit	75.00	2023-10-29
7	7	current	1200.00	7	7	deposit	125.00	2023-11-01
10	10	zero_balance	0.00	10	10	deposit	150.00	2023-11-04

4. Identify customers who have no recorded transactions.

```
SELECT *
FROM Customers c
WHERE NOT EXISTS (
    SELECT *
    FROM Transactions t
    WHERE c.customer_id = t.account_id
);
```

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

customer_id	first_name	last_name	DOB	email	phone_number	address
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```
SELECT SUM(balance) AS total_balance
FROM Accounts a
WHERE NOT EXISTS (
    SELECT *
    FROM Transactions t
    WHERE a.account_id = t.account_id
);
```

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

total_balance
NULL

6. Retrieve transactions for accounts with the lowest balance.

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

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

transaction_id	account_id	transaction_type	amount	transaction_date
5	5	withdrawal	20.00	2023-10-30
10	10	deposit	150.00	2023-11-04

7. Identify customers who have accounts of multiple types.

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

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

customer_id	first_name	last_name	DOB	email	phone_number	address

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

```sql
SELECT account_type, COUNT(*) AS total_accounts,
       (COUNT(*) / (SELECT COUNT(*) FROM Accounts)) * 100 AS percentage
FROM Accounts
GROUP BY account_type;
```

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

account_type	total_accounts	percentage
savings	4	40.0000
current	4	40.0000
zero_balance	2	20.0000

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

```sql
SELECT *
FROM Transactions
WHERE account_id IN (
    SELECT account_id
    FROM Accounts
    WHERE customer_id = 4
);
```

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

transaction_id	account_id	transaction_type	amount	transaction_date
4	4	deposit	75.00	2023-10-29
NULL	NULL	NULL	NULL	NULL

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

```sql
SELECT account_type, SUM(balance) AS total_balance
FROM Accounts a
WHERE a.account_type IN (
    SELECT DISTINCT account_type
    FROM Accounts
)
GROUP BY account_type;
```

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

account_type	total_balance
savings	11400.00
current	4800.00
zero_balance	0.00