

Banking System

Tasks 1: Database Design:

1. Create the database named "HMBank"

```
mysql> CREATE DATABASE HMBank;  
Query OK, 1 row affected (0.02 sec)  
  
mysql> use HMBank;  
Database changed
```

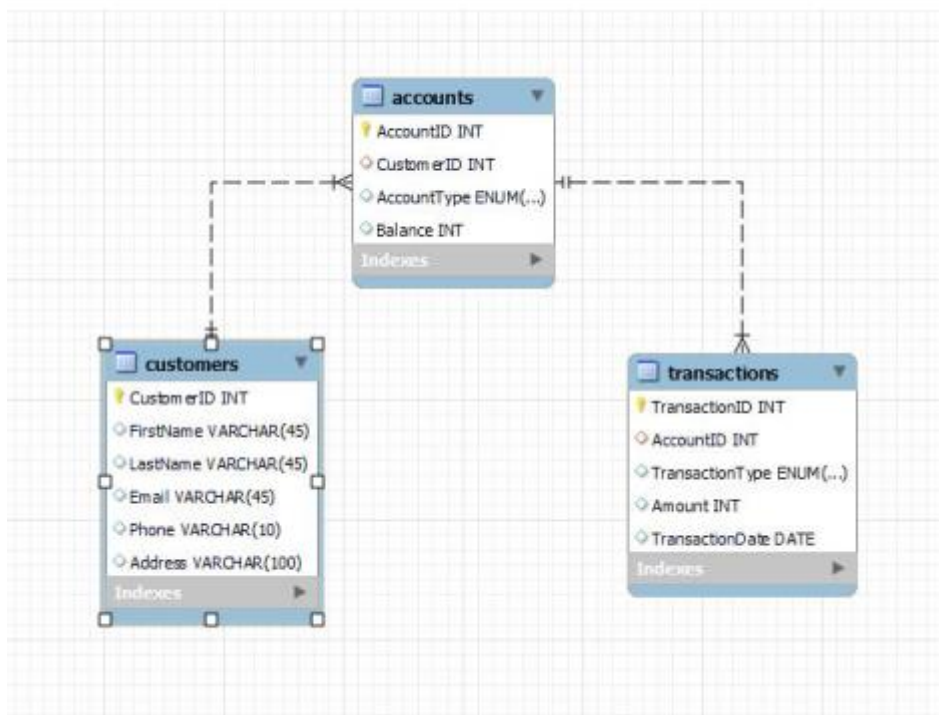
2. Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema.

```
mysql> CREATE TABLE Customers (  
->     customer_id INT PRIMARY KEY,  
->     first_name VARCHAR(255),  
->     last_name VARCHAR(255),  
->     DOB DATE,  
->     email VARCHAR(255),  
->     phone_number VARCHAR(20),  
->     address VARCHAR(255)  
-> );  
Query OK, 0 rows affected (0.05 sec)
```

```
mysql> CREATE TABLE Accounts (  
->     account_id INT PRIMARY KEY,  
->     customer_id INT,  
->     account_type VARCHAR(50),  
->     balance DECIMAL(10, 2),  
->     FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)  
-> );  
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> CREATE TABLE Transactions (  
->     transaction_id INT PRIMARY KEY,  
->     account_id INT,  
->     transaction_type VARCHAR(50),  
->     amount DECIMAL(10, 2),  
->     transaction_date DATE,  
->     FOREIGN KEY (account_id) REFERENCES Accounts(account_id)  
-> );  
Query OK, 0 rows affected (0.07 sec)
```

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



6. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.

- Customers
- Accounts
- Transactions

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

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

- Customers
- Accounts
- Transactions

```

mysql> INSERT INTO Customers(customer_id, first_name, last_name, DOB, email, phone_number, address)
-> VALUES (1, 'James', 'Kelp', '2022-12-11', 'jameskelp@gmail.com', 1111111111, 'Delhi, India'),
-> (2, 'Kim', 'Kardarsian', '2022-11-01', 'kimkardasian@gmail.com', 2222222222, 'Kolkata, India'),
-> (3, 'Priya', 'Sharma', '2023-12-01', 'priyasharma@gmail.com', 3333333333, 'Kolkata, India'),
->
-> (4, 'Krisha', 'Priya', '2021-09-02', 'krishapriyaa@gmail.com', 4444444444, 'Mumbai, India'),
-> (5, 'Sneha', 'Roy', '2020-10-07', 'Sneharoy12@gmail.com', 5555555555, 'Telangana, India'),
-> (6, 'Patrik', 'Scott', '2010-11-03', 'patrikscott@gmail.com', 6666666666, 'Tamil Nadu, India'),
-> (7, 'Pane', 'Miller', '2011-09-05', 'millerpane@gmail.com', 7777777777, 'Orissa, India'),
-> (8, 'John', 'Kipler', '2012-08-02', 'kiplerjohn@gmail.com', 8888888888, 'Kolkata, India'),
-> (9, 'Mikal', 'John', '2013-08-01', 'mikaljohn@gmail.com', 9999999999, 'Jammu & Kashmir, India'),
-> (10, 'Selina', 'Pick', '2021-08-11', 'selinapickk@gmail.com', 0000000000, 'Jammu & Kashmir, India');
Query OK, 10 rows affected (0.01 sec)
Records: 10 Duplicates: 0 Warnings: 0
  
```

```
mysql> INSERT INTO Accounts(account_id, customer_id, account_type, balance)
-> VALUES (001, 1, 'savings', 30000),
-> (002, 2, 'current', 20000),
-> (003, 3, 'savings', 30000),
-> (004, 4, 'current', 40000),
-> (005, 5, 'savings', 40000),
-> (006, 6, 'savings', 70000),
-> (007, 7, 'current', 79000),
-> (008, 8, 'savings', 80000),
-> (009, 9, 'current', 70000),
-> (010, 10, 'savings', 50000);
Query OK, 10 rows affected (0.01 sec)
Records: 10 Duplicates: 0 Warnings: 0
```

```
mysql> INSERT INTO Transactions(transaction_id, account_id, transaction_type, amount, transaction_date)
-> VALUES(111, 001, 'deposit', 300, '2022-10-12'),
-> (112, 002, 'withdraw', 200, '2012-11-12'),
-> (113, 003, 'deposit', 400, '2022-10-09'),
-> (114, 004, 'withdraw', 500, '2022-10-09'),
-> (116, 006, 'deposit', 200, '2022-10-01'),
-> (117, 007, 'withdraw', 800, '2012-11-09'),
-> (119, 009, 'deposit', 900, '2022-08-09'),
-> (120, 010, 'withdraw', 1000, '2022-11-09');
Query OK, 8 rows affected (0.01 sec)
Records: 8 Duplicates: 0 Warnings: 0
```

```
mysql> INSERT INTO Transactions(transaction_id, account_id, transaction_type, amount, transaction_date)
-> VALUES(115, 005, 'deposit', 900, '2012-11-12'),
-> (118, 008, 'withdraw', 600, '2012-11-09');
Query OK, 2 rows affected (0.00 sec)
Records: 2 Duplicates: 0 Warnings: 0
```

```
mysql> SELECT * FROM Customers;
+-----+-----+-----+-----+-----+-----+-----+
| customer_id | first_name | last_name | DOB      | email      | phone_number | address      |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | James | Kelp | 2022-12-11 | jameskelp@gmail.com | 1111111111 | Delhi, India |
| 2 | Kim | Kardarsian | 2022-11-01 | kimkardasian@gmail.com | 2222222222 | Kolkata, India |
| 3 | Priya | Sharma | 2023-12-01 | priyasharma@gmail.com | 3333333333 | Kolkata, India |
| 4 | Krisha | Priya | 2021-09-02 | krishapriyaa@gmail.com | 4444444444 | Mumbai, India |
| 5 | Sneha | Roy | 2020-10-07 | Sneharoy12@gmail.com | 5555555555 | Telangana, India |
| 6 | Patrik | Scott | 2010-11-03 | patrikscott@gmail.com | 6666666666 | Tamil Nadu, India |
| 7 | Pane | Miller | 2011-09-05 | millerpane@gmail.com | 7777777777 | Orissa, India |
| 8 | John | Kipler | 2012-08-02 | kiplerjohn@gmail.com | 8888888888 | Kolkata, India |
| 9 | Mikal | John | 2013-08-01 | mikaljohn@gmail.com | 9999999999 | Jammu & Kashmir, India |
| 10 | Selina | Pick | 2021-08-11 | selinapickk@gmail.com | 0 | Jammu & Kashmir, India |
+-----+-----+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM Accounts;
+-----+-----+-----+-----+
| account_id | customer_id | account_type | balance |
+-----+-----+-----+-----+
| 1 | 1 | savings | 30000.00 |
| 2 | 2 | current | 20000.00 |
| 3 | 3 | savings | 30000.00 |
| 4 | 4 | current | 40000.00 |
| 5 | 5 | savings | 40000.00 |
| 6 | 6 | savings | 70000.00 |
| 7 | 7 | current | 79000.00 |
| 8 | 8 | savings | 80000.00 |
| 9 | 9 | current | 70000.00 |
| 10 | 10 | savings | 50000.00 |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM Transactions;
```

transaction_id	account_id	transaction_type	amount	transaction_date
111	1	deposit	300.00	2022-10-12
112	2	withdraw	200.00	2012-11-12
113	3	deposit	400.00	2022-10-09
114	4	withdraw	500.00	2022-10-09
115	5	deposit	900.00	2012-11-12
116	6	deposit	200.00	2022-10-01
117	7	withdraw	800.00	2012-11-09
118	8	withdraw	600.00	2012-11-09
119	9	deposit	900.00	2022-08-09
120	10	withdraw	1000.00	2022-11-09

```
10 rows in set (0.00 sec)
```

2. Write SQL queries for the following tasks:

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

```
mysql> SELECT
-> first_name,
-> last_name,
-> email,
-> account_type
-> FROM
-> Customers
-> JOIN
-> Accounts ON Customers.customer_id = Accounts.customer_id;
```

first_name	last_name	email	account_type
James	Kelp	jameskelp@gmail.com	savings
Kim	Kardarsian	kimkardasian@gmail.com	current
Priya	Sharma	priyasharma@gmail.com	savings
Krishna	Priya	krishapriyaa@gmail.com	current
Sneha	Roy	Sneharoy12@gmail.com	savings
Patrik	Scott	patrikscott@gmail.com	savings
Pane	Miller	millerpane@gmail.com	current
John	Kipler	kiplerjohn@gmail.com	savings
Mikal	John	mikaljohn@gmail.com	current
Selina	Pick	selinapickk@gmail.com	savings

10 rows in set (0.00 sec)

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

```
mysql> SELECT
-> c.first_name,
-> c.last_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;
```

first_name	last_name	transaction_id	transaction_type	amount	transaction_date
James	Kelp	111	deposit	300.00	2022-10-12
Kim	Kardarsian	112	withdraw	200.00	2012-11-12
Priya	Sharma	113	deposit	400.00	2022-10-09
Krishna	Priya	114	withdraw	500.00	2022-10-09
Sneha	Roy	115	deposit	900.00	2012-11-12
Patrik	Scott	116	deposit	200.00	2022-10-01
Pane	Miller	117	withdraw	800.00	2012-11-09
John	Kipler	118	withdraw	600.00	2012-11-09
Mikal	John	119	deposit	900.00	2022-08-09
Selina	Pick	120	withdraw	1000.00	2022-11-09

10 rows in set (0.00 sec)

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

```
mysql> UPDATE Accounts
      -> SET balance = balance + 100.00
      -> WHERE account_id = 4;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

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

```
mysql> SELECT
      -> customer_id,
      -> CONCAT(first_name, last_name) AS full_name,
      -> email
      -> FROM
      -> Customers;
```

customer_id	full_name	email
1	JamesKelp	jameskelp@gmail.com
2	KimKardarsian	kimkardasian@gmail.com
3	PriyaSharma	priyasharma@gmail.com
4	KrishPriya	krishapriyaa@gmail.com
5	SnehaRoy	Sneharoy12@gmail.com
6	PatrikScott	patrikscott@gmail.com
7	PaneMiller	millerpane@gmail.com
8	JohnKipler	kiplerjohn@gmail.com
9	MikalJohn	mikaljohn@gmail.com
10	SelinaPick	selinapickk@gmail.com

```
10 rows in set (0.00 sec)
```

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

```
mysql> DELETE FROM Accounts
      -> WHERE balance = 0 AND account_type = 'savings';
Query OK, 1 row affected (0.01 sec)
```

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

```
mysql> SELECT
-> customer_id, first_name,
-> last_name, email
-> FROM Customers
-> WHERE address LIKE '%Kolkata%';
```

customer_id	first_name	last_name	email
2	Kim	Kardarsian	kimkardasian@gmail.com
3	Priya	Sharma	priyasharma@gmail.com
8	John	Kipler	kiplerjohn@gmail.com

3 rows in set (0.00 sec)

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

```
mysql> SELECT
-> balance
-> FROM
-> Accounts
-> WHERE
-> account_id = 001;
```

balance
30000.00

1 row in set (0.00 sec)

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

```
mysql> SELECT
-> account_id,
-> balance
-> FROM
-> Accounts
-> WHERE
-> account_type = 'current'
-> AND balance > 1000.00;
```

account_id	balance
2	20000.00
4	40100.00
7	79000.00
9	70000.00

4 rows in set (0.00 sec)

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

```
mysql> SELECT
-> transaction_id, transaction_type,
-> amount, transaction_date
-> FROM Transactions
-> WHERE account_id = 005;
```

transaction_id	transaction_type	amount	transaction_date
115	deposit	900.00	2012-11-12

1 row in set (0.00 sec)

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

```
mysql> SELECT
-> accounts.account_id,
-> account_type,
-> customer_id,
-> balance * 0.1 * (DATEDIFF(NOW(), transactions.transaction_date) / 365) AS interest_accrued
-> FROM
-> Accounts, Transactions
-> WHERE
-> accounts.account_id = transactions.account_id
-> AND accounts.account_type = 'savings';
```

account_id	account_type	customer_id	interest_accrued
1	savings	1	3838.3561620
3	savings	3	3875.8904083
5	savings	5	44800.0000000
6	savings	6	9167.1232870
8	savings	8	89665.7534240
10	savings	10	6013.6986300

6 rows in set (0.00 sec)

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

```
mysql> SET @overdraftlimit = 50000;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT *
-> FROM Accounts
-> WHERE balance < @overdraftlimit;
```

account_id	customer_id	account_type	balance
1	1	savings	30000.00
2	2	current	20000.00
3	3	savings	30100.00
4	4	current	40100.00
5	5	savings	40000.00

5 rows in set (0.00 sec)

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

```
mysql> SELECT * FROM Customers WHERE address NOT LIKE '%Mumbai%';
```

customer_id	first_name	last_name	DOB	email	phone_number	address
1	James	Kelp	2022-12-11	jameskelp@gmail.com	1111111111	Delhi, India
2	Kim	Kardarsian	2022-11-01	kimkardasian@gmail.com	2222222222	Kolkata, India
3	Priya	Sharma	2023-12-01	priyasharma@gmail.com	3333333333	Kolkata, India
5	Sneha	Roy	2020-10-07	Sneharoy12@gmail.com	5555555555	Telangana, India
6	Patrik	Scott	2010-11-03	patrikscott@gmail.com	6666666666	Tamil Nadu, India
7	Pane	Miller	2011-09-05	millerpane@gmail.com	7777777777	Orissa, India
8	John	Kipler	2012-08-02	kiplerjohn@gmail.com	8888888888	Kolkata, India
9	Mikal	John	2013-08-01	mikaljohn@gmail.com	9999999999	Jammu & Kashmir, India
10	Selina	Pick	2021-08-11	selinapickk@gmail.com	0	Jammu & Kashmir, India
11	Jamey	Kelp	2012-11-11	jamewykelp@gmail.com	1178653461	Delhi, India

```
10 rows in set (0.00 sec)
```

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

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

```
mysql> SELECT AVG(balance) FROM Accounts;
```

AVG(balance)
50920.000000

```
1 row in set (0.00 sec)
```

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

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

account_id	customer_id	account_type	balance
8	8	savings	80000.00
7	7	current	79000.00
6	6	savings	70000.00
9	9	current	70000.00
10	10	savings	50000.00
4	4	current	40100.00
5	5	savings	40000.00
3	3	savings	30100.00
1	1	savings	30000.00
2	2	current	20000.00

```
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT SUM(amount) AS totaldeposit
-> FROM Transactions
-> WHERE transaction_date = '2022-10-12';
+-----+
| totaldeposit |
+-----+
|      300.00 |
+-----+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT * FROM Customers LIMIT 1;
+-----+-----+-----+-----+-----+-----+-----+
| customer_id | first_name | last_name | DOB      | email                | phone_number | address      |
+-----+-----+-----+-----+-----+-----+-----+
| 1           | James     | Kelp      | 2022-12-11 | jameskelp@gmail.com | 1111111111   | Delhi, India |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT * FROM Customers ORDER BY customer_id DESC LIMIT 1;
+-----+-----+-----+-----+-----+-----+-----+
| customer_id | first_name | last_name | DOB      | email                | phone_number | address      |
+-----+-----+-----+-----+-----+-----+-----+
| 11          | Jamey     | Kelp      | 2012-11-11 | jamewykelp@gmail.com | 1178653461   | Delhi, India |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT transactions.*, accounts.account_type
-> FROM Transactions
-> JOIN Accounts
-> ON transactions.account_id = accounts.account_id
-> ORDER BY transactions.transaction_id;
+-----+-----+-----+-----+-----+-----+-----+
| transaction_id | account_id | transaction_type | amount | transaction_date | account_type |
+-----+-----+-----+-----+-----+-----+-----+
| 111           | 1          | deposit         | 300.00 | 2022-10-12      | savings      |
| 112           | 2          | withdraw        | 200.00 | 2012-11-12      | current      |
| 113           | 3          | deposit         | 400.00 | 2022-10-09      | savings      |
| 114           | 4          | withdraw        | 500.00 | 2022-10-09      | current      |
| 115           | 5          | deposit         | 900.00 | 2012-11-12      | savings      |
| 116           | 6          | deposit         | 200.00 | 2022-10-01      | savings      |
| 117           | 7          | withdraw        | 800.00 | 2012-11-09      | current      |
| 118           | 8          | withdraw        | 600.00 | 2012-11-09      | savings      |
| 119           | 9          | deposit         | 900.00 | 2022-08-09      | current      |
| 120           | 10         | withdraw        | 1000.00 | 2022-11-09      | savings      |
+-----+-----+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT * FROM
-> Customers
-> JOIN Accounts
-> ON customers.customer_id = accounts.customer_id;
```

	customer_id	first_name	last_name	DOB	email	phone_number	address	account_id	customer_id	a
	ccount_type	balance								
avings	1	James	Kelp	2022-12-11	jameskelp@gmail.com	1111111111	Delhi, India	1	1	s
urrent	2	Kim	Kardarsian	2022-11-01	kimkardasian@gmail.com	2222222222	Kolkata, India	2	2	c
avings	3	Priya	Sharma	2023-12-01	priyasharma@gmail.com	3333333333	Kolkata, India	3	3	s
urrent	4	Krishna	Priya	2021-09-02	krishapriyaa@gmail.com	4444444444	Mumbai, India	4	4	c
avings	5	Sneha	Roy	2020-10-07	sneharoy12@gmail.com	5555555555	Telangana, India	5	5	s
avings	6	Patrik	Scott	2010-11-03	patrikscott@gmail.com	6666666666	Tamil Nadu, India	6	6	s
urrent	7	Pane	Miller	2011-09-05	millierpane@gmail.com	7777777777	Orissa, India	7	7	c
avings	8	John	Kipler	2012-08-02	kiplerjohn@gmail.com	8888888888	Kolkata, India	8	8	s
urrent	9	Mikal	John	2013-08-01	mikaljohn@gmail.com	9999999999	Jammu & Kashmir, India	9	9	c
avings	10	Selina	Pick	2021-08-11	selinapickk@gmail.com	0	Jammu & Kashmir, India	10	10	s

10 rows in set (0.00 sec)

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

```
mysql> SELECT customers.*, transactions.*
-> FROM Customers JOIN Accounts
-> ON customers.customer_id = accounts.customer_id
-> JOIN Transactions
-> ON accounts.account_id = transactions.account_id
-> WHERE accounts.account_id = 002;
```

	customer_id	first_name	last_name	DOB	email	phone_number	address	transaction_id	account_id	transa
	ction_type	amount	transaction_date							
aw	2	Kim	Kardarsian	2022-11-01	kimkardasian@gmail.com	2222222222	Kolkata, India	112	2	withdr

1 row in set (0.00 sec)

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

```
mysql> SELECT customers.customer_id,
-> first_name, last_name, COUNT(account_id)
-> AS num_of_acc FROM Customers
-> JOIN Accounts
-> ON customers.customer_id = accounts.customer_id
-> GROUP BY customers.customer_id, first_name, last_name
-> HAVING COUNT(account_id) > 1;
```

	customer_id	first_name	last_name	num_of_acc
	1	James	Kelp	2

1 row in set (0.00 sec)

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

```
mysql> SELECT (SELECT SUM(amount) FROM Transactions
-> WHERE transaction_type = 'deposit') - (SELECT SUM(amount)
-> FROM Transactions WHERE transaction_type = 'withdraw') AS difference;
+-----+
| difference |
+-----+
|    -400.00 |
+-----+
1 row in set (0.00 sec)
```

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

```
mysql> DELIMITER @@
mysql> CREATE PROCEDURE avgbal(IN val1 DATE, val2 DATE)
-> BEGIN
-> SELECT account_id,
-> (DATEDIFF(val1, val2))*balance/DATEDIFF(val1, val2) AS avgbal
-> FROM Accounts;
-> END @@
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> DELIMITER ;
mysql> CALL avgbal('2024-01-01', '2023-01-01');
```

account_id	avgbal
1	30000.000000
2	20000.000000
3	30100.000000
4	40100.000000
5	40000.000000
6	70000.000000
7	79000.000000
8	80000.000000
9	70000.000000
10	50000.000000
12	30000.000000

11 rows in set (0.01 sec)

```
Query OK, 0 rows affected (0.03 sec)
```

11 Calculate the total balance for each account type.

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

account_type	total_balance
savings	300100.00
current	239100.00

```
2 rows in set (0.00 sec)
```

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

```
mysql> SELECT accounts.account_id, COUNT(transaction_id) AS count
-> FROM Accounts JOIN Transactions
-> ON accounts.account_id = transactions.account_id
-> GROUP BY accounts.account_id ORDER BY count DESC;
```

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

```
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT customers.first_name,  
-> GROUP_CONCAT(accounts.account_type) AS account_types,  
-> SUM(accounts.balance) AS total  
-> FROM Customers JOIN Accounts  
-> ON customers.customer_id = accounts.customer_id  
-> GROUP BY customers.first_name ORDER BY total DESC;  
+-----+-----+-----+  
| first_name | account_types | total |  
+-----+-----+-----+  
| John      | savings      | 80000.00 |  
| Pane      | current      | 79000.00 |  
| Mikal     | current      | 70000.00 |  
| Patrik    | savings      | 70000.00 |  
| James     | savings,current | 60000.00 |  
| Selina    | savings      | 50000.00 |  
| Krisha    | current      | 40100.00 |  
| Sneha     | savings      | 40000.00 |  
| Priya     | savings      | 30100.00 |  
| Kim       | current      | 20000.00 |  
+-----+-----+-----+  
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT * FROM Transactions  
-> WHERE(account_id, transaction_date, amount)  
-> IN (SELECT account_id, transaction_date, amount  
-> FROM transactions GROUP BY account_id, transaction_date, amount  
-> HAVING COUNT(*)>1);  
+-----+-----+-----+-----+-----+  
| transaction_id | account_id | transaction_type | amount | transaction_date |  
+-----+-----+-----+-----+-----+  
| 120 | 10 | withdraw | 1000.00 | 2022-11-09 |  
| 121 | 10 | withdraw | 1000.00 | 2022-11-09 |  
+-----+-----+-----+-----+-----+  
2 rows in set (0.00 sec)
```

Tasks 4: Subquery and its type:

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

```
mysql> SELECT customers.first_name, accounts.balance
-> FROM Accounts, Customers
-> WHERE customers.customer_id = accounts.account_id
-> AND balance = (SELECT MAX(balance) FROM Accounts);
+-----+-----+
| first_name | balance |
+-----+-----+
| John      | 80000.00 |
+-----+-----+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT customers.first_name,
-> AVG(balance) FROM Customers, Accounts
-> WHERE customers.customer_id = accounts.customer_id
-> GROUP BY customers.first_name
-> HAVING COUNT(balance)>1;
+-----+-----+
| first_name | AVG(balance) |
+-----+-----+
| James      | 30000.000000 |
+-----+-----+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT accounts.account_id, accounts.customer_id, accounts.account_type
-> FROM Accounts, Transactions
-> WHERE accounts.account_id = transactions.account_id
-> AND amount > (SELECT AVG(amount) FROM Transactions)
-> GROUP BY accounts.account_id, accounts.customer_id, accounts.account_type;
+-----+-----+-----+
| account_id | customer_id | account_type |
+-----+-----+-----+
| 5          | 5          | savings      |
| 7          | 7          | current      |
| 9          | 9          | current      |
| 10         | 10         | savings      |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

4 Identify customers who have no recorded transactions.

```
mysql> SELECT customers.first_name, accounts.account_id, accounts.account_type
-> FROM Customers, Accounts
-> WHERE customers.customer_id = accounts.customer_id
-> AND accounts.account_id IN (SELECT accounts.account_id FROM Accounts
-> WHERE accounts.account_id NOT IN (SELECT account_id FROM Transactions));
```

first_name	account_id	account_type
James	12	current

```
1 row in set (0.00 sec)
```

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

```
mysql> SELECT SUM(balance) AS no_tranc
-> FROM Accounts
-> WHERE account_id IN (SELECT accounts.account_id FROM Accounts
-> WHERE accounts.account_id NOT IN (SELECT account_id FROM Transactions));
```

no_tranc
30000.00

```
1 row in set (0.00 sec)
```

6 Retrieve transactions for accounts with the lowest balance

```
mysql> SELECT accounts.account_id, transactions.*
-> FROM Accounts, Transactions
-> WHERE accounts.account_id = transactions.account_id
-> AND accounts.balance = (SELECT MIN(balance) FROM Accounts);
```

account_id	transaction_id	account_id	transaction_type	amount	transaction_date
2	112	2	withdraw	200.00	2012-11-12

```
1 row in set (0.00 sec)
```

7 Identify customers who have accounts of multiple types.

```
mysql> SELECT first_name, COUNT(account_type) AS no_of_accs
-> FROM (
-> SELECT customers.first_name, accounts.account_type
-> FROM Customers, Accounts
-> WHERE customers.customer_id = accounts.customer_id
-> GROUP BY customers.first_name, accounts.account_type
-> ) AS subquery
-> GROUP BY first_name
-> HAVING COUNT(account_type) > 1;
```

first_name	no_of_accs
James	2

```
1 row in set (0.00 sec)
```


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

```
mysql> SELECT
->     account_type,
->     COUNT(*) AS no_of_accs,
->     COUNT(*) / (SELECT COUNT(*) FROM Accounts) * 100 AS percentage
-> FROM
->     Accounts
-> GROUP BY
->     account_type;
+-----+-----+-----+
| account_type | no_of_accs | percentage |
+-----+-----+-----+
| savings      |          6 |    54.5455 |
| current      |          5 |    45.4545 |
+-----+-----+-----+
2 rows in set (0.01 sec)
```

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

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE retritransaction( IN val1 INT)
-> BEGIN
-> SELECT * FROM Transactions WHERE account_id
-> IN (SELECT account_id FROM Accounts WHERE customer_id = val1);
-> END //
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DELIMITER ;
mysql> CALL retritransaction(10);
+-----+-----+-----+-----+-----+
| transaction_id | account_id | transaction_type | amount | transaction_date |
+-----+-----+-----+-----+-----+
|          120 |          10 | withdraw        | 1000.00 | 2022-11-09      |
|          121 |          10 | withdraw        | 1000.00 | 2022-11-09      |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

Query OK, 0 rows affected (0.02 sec)
```

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

```
mysql> SELECT a1.account_type,
->     (SELECT SUM(a2.balance) FROM accounts a2
->     WHERE a1.account_type = a2.account_type) AS total_balance
-> FROM Accounts a1
-> GROUP BY a1.account_type;
+-----+-----+
| account_type | total_balance |
+-----+-----+
| savings      |    300100.00 |
| current      |    239100.00 |
+-----+-----+
2 rows in set (0.00 sec)
```