ASSIGNMENT: BANKING SYSTEM

Tasks 1: Database Design:

1.Create the database named "HMBank" :

```
create database hmbank;

use hmbank;
```

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

```
CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50) ,
    DOB DATE ,
    email VARCHAR(100) unique ,
    phone_number VARCHAR(15) UNIQUE,
    address TEXT
);

Desc customers;
```

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
first_name	varchar(50)	YES		NULL	
last_name	varchar(50)	YES		NULL	
DOB	date	YES		NULL	
email	varchar(100)	YES	UNI	NULL	
phone_number	varchar(15)	YES	UNI	NULL	
address	text	YES		NULL	

```
CREATE TABLE Accounts (
    account_id INT PRIMARY KEY,
    customer_id INT,
    account_type ENUM('savings', 'current', 'zero_balance') NOT NULL,
    balance DECIMAL(10,2) DEFAULT 0.00,
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id) ON DELETE CASCADE
);
```

Desc accounts;

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

CREATE TABLE Transactions (

transaction_id INT PRIMARY KEY AUTO_INCREMENT,

account_id INT,

transaction_type ENUM('deposit', 'withdrawal', 'transfer') NOT NULL,

amount DECIMAL(10,2) ,

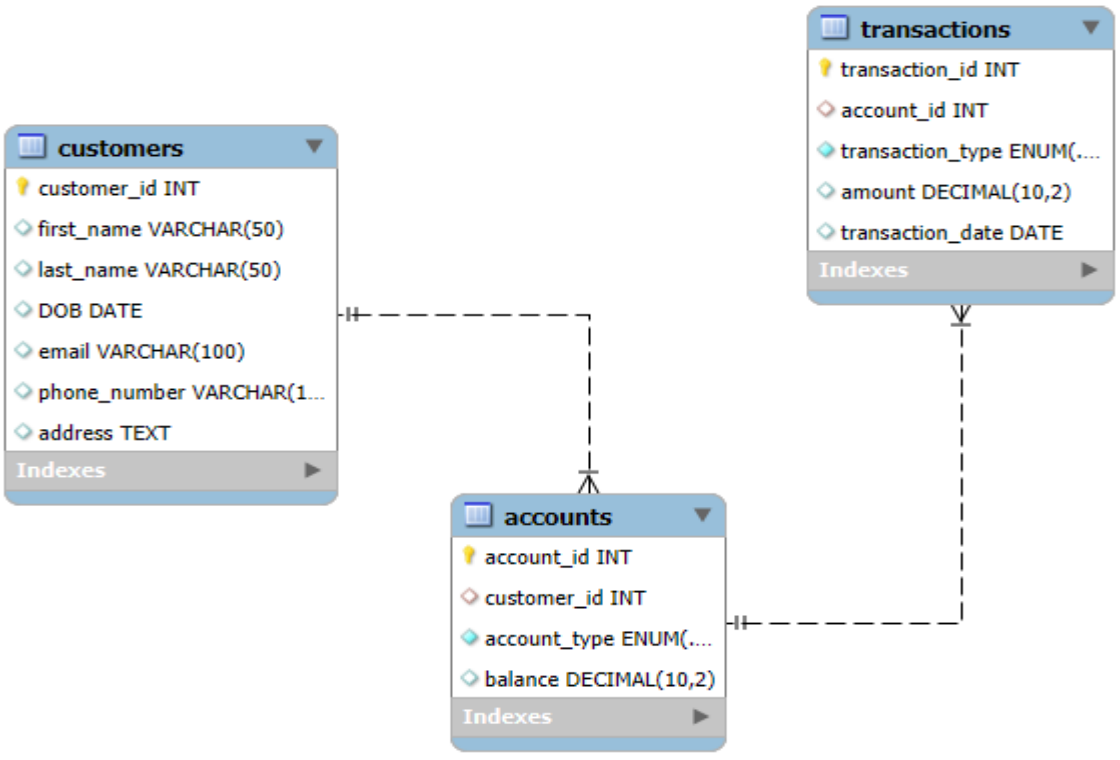
transaction_date DATE,

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

);

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

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



4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

All primary keys and foreign keys were properly defined in the SQL provided

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

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

INSERT INTO Customers (customer_id, first_name, last_name, DOB, email, phone_number, address) VALUES
(1, 'Nila', 'Kumar', '1995-04-10', 'nila@example.com', '9876543210', 'Chennai, India'),
(2, 'Nisha', 'Rao', '1996-07-15', 'nisha@example.com', '9876543211', 'Mumbai, India'),
(3, 'Praveen', 'Sharma', '1992-12-05', 'praveen@example.com', '9876543212', 'Delhi, India'),
(4, 'Ladha', 'Verma', '1998-03-25', 'ladha@example.com', '9876543213', 'Bangalore, India'),
(5, 'Hema', 'Devi', '1993-06-30', 'hexa@example.com', '9876543214', 'Hyderabad, India'),
(6, 'Koki', 'Singh', '1997-09-18', 'koki@example.com', '9876543215', 'Pune, India'),
(7, 'Madhu', 'Iyer', '1994-07-07', 'madhu@example.com', '9876543216', 'Kolkata, India'),
(8, 'Raaji', 'Reddy', '1991-11-11', 'raaji@example.com', '9876543217', 'Jaipur, India'),
(9, 'Revathi', 'Bose', '1990-04-05', 'revathi@example.com', '9876543218', 'Lucknow, India'),
(10, 'Dhivya', 'Menon', '1999-01-20', 'dhivya@example.com', '9876543219', 'Chandigarh, India');

Output:

customer_id	first_name	last_name	DOB	email	phone_number	address
1	Nila	Kumar	1995-04-10	nila@example.com	9876543210	Chennai, India
2	Nisha	Rao	1996-07-15	nisha@example.com	9876543211	Mumbai, India
3	Praveen	Sharma	1992-12-05	praveen@example.com	9876543212	Delhi, India
4	Ladha	Verma	1998-03-25	ladha@example.com	9876543213	Bangalore, India
5	Hema	Devi	1993-06-30	hexa@example.com	9876543214	Hyderabad, India
6	Koki	Singh	1997-09-18	koki@example.com	9876543215	Pune, India
7	Madhu	Iyer	1994-07-07	madhu@example.com	9876543216	Kolkata, India
8	Raaji	Reddy	1991-11-11	raaji@example.com	9876543217	Jaipur, India
9	Revathi	Bose	1990-04-05	revathi@example.com	9876543218	Lucknow, India
10	Dhivya	Menon	1999-01-20	dhivya@example.com	9876543219	Chandigarh, India
NULL	NULL	NULL	NULL	NULL	NULL	NULL

   - ***Accounts***

INSERT INTO Accounts (account_id, customer_id, account_type, balance) VALUES
(101, 1, 'savings', 5000.00),
(102, 2, 'current', 12000.00),
(103, 3, 'savings', 8000.50),
(104, 4, 'zero_balance', 0.00),
(105, 5, 'savings', 9500.00),
(106, 6, 'current', 20000.75),
(107, 7, 'zero_balance', 0.00),
(108, 8, 'savings', 7200.30),
(109, 9, 'current', 17500.90),
(110, 10, 'savings', 6000.40);

Output:

account_id	customer_id	account_type	balance
101	1	savings	6000.00
102	2	current	12000.00
103	3	savings	8000.50
104	4	zero_balance	0.00
105	5	savings	9500.00
106	6	current	20000.75
107	7	zero_balance	0.00
108	8	savings	7200.30
109	9	current	17500.90
110	10	savings	6000.40
NULL	NULL	NULL	NULL

• ***Transactions***

INSERT INTO Transactions (account_id, transaction_type, amount, transaction_date) VALUES
(101, 'deposit', 3000.00, '2024-03-01'),
(102, 'withdrawal', 2500.50, '2024-03-02'),
(103, 'deposit', 5000.00, '2024-03-03'),
(104, 'deposit', 2000.00, '2024-03-04'),
(105, 'withdrawal', 4000.00, '2024-03-05'),
(106, 'deposit', 3500.75, '2024-03-06'),
(107, 'deposit', 1200.00, '2024-03-07'),
(108, 'withdrawal', 6000.00, '2024-03-08'),
(109, 'deposit', 7500.90, '2024-03-09'),
(110, 'withdrawal', 3200.40, '2024-03-10');

Output:

transaction_id	account_id	transaction_type	amount	transaction_date
1	101	deposit	3000.00	2024-03-01
2	102	withdrawal	2500.50	2024-03-02
3	103	deposit	5000.00	2024-03-03
4	104	deposit	2000.00	2024-03-04
5	105	withdrawal	4000.00	2024-03-05
6	106	deposit	3500.75	2024-03-06
7	107	deposit	1200.00	2024-03-07
8	108	withdrawal	6000.00	2024-03-08
9	109	deposit	7500.90	2024-03-09
10	110	withdrawal	3200.40	2024-03-10
NULL	NULL	NULL	NULL	NULL

2.Write SQL queries for the following tasks:

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

```
SELECT  c.first_name,  c.last_name,  a.account_type, c.email
FROM Customers c
```

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

first_name	last_name	account_type	email
Nila	Kumar	savings	nila@example.com
Nisha	Rao	current	nisha@example.com
Praveen	Sharma	savings	praveen@example.com
Ladha	Verma	zero_balance	ladha@example.com
Hema	Devi	savings	hexa@example.com
Koki	Singh	current	koki@example.com
Madhu	Iyer	zero_balance	madhu@example.com
Raaji	Reddy	savings	raaji@example.com
Revathi	Bose	current	revathi@example.com
Dhivya	Menon	savings	dhivya@example.com

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

sELECT concat(c.first_name ,' ' ,c.last_name) , a.account_id ,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;

Output:

concat(c.first_name ,' ' ,c.last_name)	account_id	transaction_id	transaction_type	amount	transaction_date
Nila Kumar	101	1	deposit	3000.00	2024-03-01
Nisha Rao	102	2	withdrawal	2500.50	2024-03-02
Praveen Sharma	103	3	deposit	5000.00	2024-03-03
Ladha Verma	104	4	deposit	2000.00	2024-03-04
Hema Devi	105	5	withdrawal	4000.00	2024-03-05
Koki Singh	106	6	deposit	3500.75	2024-03-06
Madhu Iyer	107	7	deposit	1200.00	2024-03-07
Raaji Reddy	108	8	withdrawal	6000.00	2024-03-08
Revathi Bose	109	9	deposit	7500.90	2024-03-09
Dhivya Menon	110	10	withdrawal	3200.40	2024-03-10

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

update accounts set balance =balance +1000 where account_id =101;

Output:

4. 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;

Output:

full_name
Nila Kumar
Nisha Rao
Praveen Sharma
Ladha Verma
Hema Devi
Koki Singh
Madhu Iyer
Raaji Reddy
Revathi Bose
Dhivya Menon

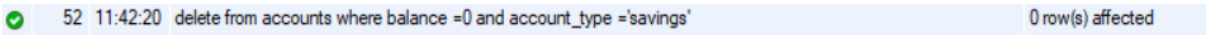
5. Write a SQL query to remove accounts with a balance of zero where the account type is savings.

```
SET SQL_SAFE_UPDATES = 0;

delete from accounts where balance =0 and account_type ='savings';

SET SQL_SAFE_UPDATES = 1;
```

Output:



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

```
select * from customers where address ='Chennai, India'; -- we should specify the city ...
```

Output:

Output:

customer_id	first_name	last_name	DOB	email	phone_number	address
1	Nila	Kumar	1995-04-10	nila@example.com	9876543210	Chennai, India
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```
select account_type , balance from accounts where account_type ='savings';
```

Output:

account_type	balance
savings	7000.00
savings	8000.50
savings	9500.00
savings	7200.30
savings	6000.40

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

select account_type ,balance from accounts where account_type ='current' having balance>1000

Output:

	account_type	balance
▶	current	12000.00
	current	20000.75
	current	17500.90

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 = 105;

Output*:*

	transaction_id	transaction_type	amount	transaction_date
▶	5	withdrawal	4000.00	2024-03-05
*	NULL	NULL	NULL	NULL

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

select account_type,balance, round(balance * 0.05 , 2) as interest_accured from accounts where account_type = 'savings';

Output:

	account_type	balance	interest_accured
▶	savings	7000.00	350.00
	savings	8000.50	400.03
	savings	9500.00	475.00
	savings	7200.30	360.02
	savings	6000.40	300.02

11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit -- assuming the overdraft as -500

select account_id from accounts where balance < -500;

Output:

	account_id
*	NULL

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

select concat(first_name , ' ', last_name) from customers where address <>'chennai, india';

Output:

	concat(first_name , '', last_name)
▶	Nisha Rao
	Praveen Sharma
	Ladha Verma
	Hema Devi
	Koki Singh
	Madhu Iyer
	Raaji Reddy
	Revathi Bose
	Dhivya Menon

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 customer_id , avg(balance) from accounts group by customer_id;

Output:

	customer_id	avg(balance)
▶	1	7000.000000
	2	12000.000000
	3	8000.500000
	4	0.000000
	5	9500.000000
	6	20000.750000
	7	0.000000
	8	7200.300000
	9	17500.900000
	10	6000.400000

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

select account_id , balance from accounts order by balance desc limit 10;

Output:

account_id	balance
106	20000.75
109	17500.90
102	12000.00
105	9500.00
103	8000.50
108	7200.30
101	7000.00
110	6000.40
104	0.00
107	0.00
NULL	NULL

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

```
SELECT c.customer_id,  c.first_name,  c.last_name, SUM(t.amount) AS total_deposits
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
JOIN Transactions t ON a.account_id = t.account_id
WHERE t.transaction_type = 'deposit'
AND t.transaction_date = '2024-03-07'
GROUP BY c.customer_id, c.first_name, c.last_name;
```

Output:

customer_id	first_name	last_name	total_deposits
7	Madhu	Iyer	1200.00

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

```
select first_name as oldest , dob  from customers order by dob asc limit 1;
```

Output:

oldest	dob
Revathi	1990-04-05

```
select first_name as eldest , dob  from customers order by dob desc limit 1;
```

Output:

eldest	dob
Dhivya	1999-01-20

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

```
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;
```

Output:

transaction_id	account_id	account_type	transaction_type	amount	transaction_date
1	101	savings	deposit	3000.00	2024-03-01
2	102	current	withdrawal	2500.50	2024-03-02
3	103	savings	deposit	5000.00	2024-03-03
4	104	zero_balance	deposit	2000.00	2024-03-04
5	105	savings	withdrawal	4000.00	2024-03-05
6	106	current	deposit	3500.75	2024-03-06
7	107	zero_balance	deposit	1200.00	2024-03-07
8	108	savings	withdrawal	6000.00	2024-03-08
9	109	current	deposit	7500.90	2024-03-09
10	110	savings	withdrawal	3200.40	2024-03-10

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 fullname ,c.customer_id ,a.customer_id,
a.account_type,a.balance
from accounts a
join customers c on c.customer_id = a.customer_id;
```

Output:

fullname	customer_id	customer_id	account_type	balance
Nila Kumar	1	1	savings	7000.00
Nisha Rao	2	2	current	12000.00
Praveen Sharma	3	3	savings	8000.50
Ladha Verma	4	4	zero_balance	0.00
Hema Devi	5	5	savings	9500.00
Koki Singh	6	6	current	20000.75
Madhu Iyer	7	7	zero_balance	0.00
Raaji Reddy	8	8	savings	7200.30
Revathi Bose	9	9	current	17500.90
Dhivya Menon	10	10	savings	6000.40

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

```
SELECT t.transaction_id, t.account_id, concat(c.first_name ,' ', c.last_name) as fullname, c.email,
    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
WHERE t.account_id = 101;
```

Output:

transaction_id	account_id	fullname	email	account_type	transaction_type	amount	transaction_date
1	101	Nila Kumar	nila@example.com	savings	deposit	3000.00	2024-03-01

-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 account_count
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;
```

Output:

customer_id	first_name	last_name	account_count

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

```
SELECT t.account_id, SUM(CASE WHEN t.transaction_type = 'deposit' THEN t.amount ELSE 0 END) -
SUM(CASE WHEN t.transaction_type = 'withdrawal' THEN t.amount ELSE 0 END) AS
transaction_difference FROM Transactions t GROUP BY t.account_id;
```

Output:

account_id	transaction_difference
101	3000.00
102	-2500.50
103	5000.00
104	2000.00
105	-4000.00
106	3500.75
107	1200.00
108	-6000.00
109	7500.90
110	-3200.40

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

```
SELECT
   T.account_id,
   T.transaction_date,
   AVG(A.balance) AS avg_daily_balance
FROM Accounts A
JOIN Transactions T ON A.account_id = T.account_id
WHERE T.transaction_date BETWEEN '2024-01-01' AND '2024-03-01'
GROUP BY T.account_id, T.transaction_date
ORDER BY T.account_id, T.transaction_date;
```

Output:

account_id	transaction_date	avg_daily_balance
101	2024-03-01	7000.000000

11. Calculate the total balance for each account type.

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

Output:

account_type	total_balance
savings	37701.20
current	49501.65
zero_balance	0.00

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

```
SELECT  account_id,
   COUNT(transaction_id) AS transaction_count
FROM Transactions
```

GROUP BY account_id

ORDER BY transaction_count DESC;

Output:

account_id	transaction_count
101	1
102	1
103	1
104	1
105	1
106	1
107	1
108	1
109	1
110	1

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

SELECT c.customer_id, 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, a.account_type

HAVING SUM(a.balance) > 10000

ORDER BY total_balance DESC;

Output:

customer_id	full_name	account_type	total_balance
6	Koki Singh	current	20000.75
9	Revathi Bose	current	17500.90
2	Nisha Rao	current	12000.00

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

SELECT account_id, transaction_date, amount, COUNT(*) AS duplicate_count

FROM Transactions

GROUP BY account_id, transaction_date, amount

```
HAVING COUNT(*) > 1
ORDER BY duplicate_count DESC;
```

Output:

	account_id	transaction_date	amount	duplicate_count

Tasks 4: Subquery and its type:

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

```
SELECT *
FROM Customers
WHERE customer_id IN (
    SELECT customer_id
    FROM Accounts
    WHERE balance = (SELECT MAX(balance) FROM Accounts)
);
```

Output:

	customer_id	first_name	last_name	DOB	email	phone_number	address
▶	6	Koki	Singh	1997-09-18	koki@example.com	9876543215	Pune, India
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

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

Output:

	customer_id	avg_balance

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

SELECT account_id, transaction_id, amount

FROM Transactions

WHERE amount > (SELECT AVG(amount) FROM Transactions);

Output:

	account_id	transaction_id	amount
▶	103	3	5000.00
	105	5	4000.00
	108	8	6000.00
	109	9	7500.90
*	NULL	NULL	NULL

4. Identify customers who have no recorded transactions.

SELECT customer_id, first_name, last_name

FROM Customers

WHERE customer_id NOT IN (SELECT DISTINCT customer_id FROM Transactions);

Output:

	customer_id	first_name	last_name
*	NULL	NULL	NULL

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

SELECT SUM(balance) AS total_balance_without_transactions

FROM Accounts

WHERE account_id NOT IN (SELECT DISTINCT account_id FROM Transactions);

Output:

total_balance_without_transactions
NULL

6. Retrieve transactions for accounts with the lowest balance.

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

Output:

transaction_id	account_id	transaction_type	amount	transaction_date
4	104	deposit	2000.00	2024-03-04
7	107	deposit	1200.00	2024-03-07
NULL	NULL	NULL	NULL	NULL

7. Identify customers who have accounts of multiple types.

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

Output:

customer_id	first_name

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

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

Output:

	account_type	account_count	percentage
▶	savings	5	50.00000
	current	3	30.00000
	zero_balance	2	20.00000

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

```
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 = 1;  -- Replace 101 with the desired customer_id
```

Output:

	transaction_id	account_id	transaction_type	amount	transaction_date
▶	1	101	deposit	3000.00	2024-03-01

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

```
SELECT account_type,
    (SELECT SUM(balance)
    FROM Accounts a2
    WHERE a2.account_type = a1.account_type) AS total_balance
FROM Accounts a1
GROUP BY account_type;
```

Output:

	account_type	total_balance
▶	savings	37701.20
	current	49501.65
	zero_balance	0.00