Banking System

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.

```
Create database hmbank;
use hmbank;
create table customers(
        customer_id INT primary key,
        first_name VARCHAR(50),
        last_name VARCHAR(50),
        DOB date,
        email VARCHAR(50),
        phone_number INT,
   address varchar(100)
);
create table accounts(
        account_id VARCHAR(20)primary key,
   customer_id INT,
   account_type VARCHAR(15),
   balance decimal(7 , 2),
   foreign key(customer_id) references customers(customer_id)
);
create table transactions(
transaction_id INT primary key,
account_id VARCHAR(20),
transaction_type VARCHAR(15),
amount INT,
transaction_date date,
```

```
foreign key(account_id) references accounts(account_id)
);
```

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

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

```
INSERT INTO customers (customer_id, first_name, last_name, DOB, email, phone_number, address)
VALUES
    (1, 'John', 'Doe', '1990-05-15', 'john.doe@gmail.com', 234567890, '123 Main St, New York, USA'),
    (2, 'Jane', 'Smith', '1985-09-20', 'jane.smith@gmail.com', 876543210, '456 Elm St, Los Angeles, USA'),
    (3, 'Alice', 'Johnson', '1978-03-10', 'alice.johnson@gmail.com', 567890123, '789 Oak St, Chicago, USA'),
    (4, 'Bob', 'Brown', '1995-11-25', 'bob.brown@gmail.com', 890123456, '101 Maple Ave, Huston, USA'),
    (5, 'Emily', 'Davis', '1982-07-08', 'emily.davis@gmail.com', 345678901, '222 Pine St, Texas, USA'),
    (6, 'Michael', 'Wilson', '1973-01-30', 'michael.wilson@gmail.com', 901234567, '87 Cedar Ave, Philadelphia, USA'),
    (7, 'Sophia', 'Martinez', '1998-12-18', 'sophia.martinez@gmail.com', 678901234, '44 Birch St, San Antonio, USA'),
    (8, 'William', 'Anderson', '1989-06-05', 'william.anderson@gmail.com', 123456789, '545 Spruce Ave, Texas, USA'),
    (9, 'Olivia', 'Garcia', '1975-08-12', 'olivia.garcia@gmail.com', 789012345, '6 Walnut St, Dallas, USA'),
    (10, 'David', 'Rodriguez', '1992-04-22', 'david.rodriguez@gmail.com', 345678900, '7 Ash St, New York, USA');
```

```
INSERT INTO accounts (account_id, customer_id, account_type, balance)
VALUES
    ('ACC001', 1, 'Savings', 1500.00),
    ('ACC002', 2, 'Current', 2500.00),
    ('ACC003', 3, 'Savings', 3500.00),
    ('ACC004', 4, 'Current', 4500.00),
    ('ACC005', 5, 'Savings', 5500.00),
    ('ACC006', 6, 'Current', 6500.00),
    ('ACC007', 7, 'Savings', 7500.00),
    ('ACC008', 8, 'Current', 8500.00),
    ('ACC009', 9, 'Savings', 9500.00),
    ('ACC010', 10, 'Zero_Balance', 0.00);

INSERT INTO transactions (transaction_id, account_id, transaction_type, amount, transaction_date)
VALUES
    (1, 'ACC001', 'Deposit', 500, '2024-04-01'),
    (2, 'ACC002', 'Withdrawal', 200, '2024-04-02'),
    (3, 'ACC003', 'Transfer', 1000, '2024-04-03'),
    (4, 'ACC004', 'Deposit', 800, '2024-04-04'),
    (5, 'ACC005', 'Withdrawal', 300, '2024-04-05'),
    (6, 'ACC006', 'Transfer', 1200, '2024-04-06'),
    (7, 'ACC007', 'Deposit', 600, '2024-04-07'),
    (8, 'ACC008', 'Withdrawal', 400, '2024-04-08'),
    (9, 'ACC009', 'Transfer', 1500, '2024-04-09'),
    (10, 'ACC010', 'Deposit', 1000, '2024-04-10');
```

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1990-05-15	john.doe@gmail.com	234567890	123 Main St, New York, USA
2	Jane	Smith	1985-09-20	jane.smith@gmail.com	876543210	456 Elm St, Los Angeles, USA
3	Alice	Johnson	1978-03-10	alice.johnson@gmail.com	567890123	789 Oak St, Chicago, USA
4	Bob	Brown	1995-11-25	bob.brown@gmail.com	890123456	101 Maple Ave, Huston, USA
5	Emily	Davis	1982-07-08	emily.davis@gmail.com	345678901	222 Pine St, Texas, USA
6	Michael	Emily	1973-01-30	michael.wilson@gmail.com	901234567	87 Cedar Ave, Philadelphia, USA
7	Sophia	Martinez	1998-12-18	sophia.martinez@gmail.com	678901234	44 Birch St, San Antonio, USA
8	William	Anderson	1989-06-05	william.anderson@gmail.com	123456789	545 Spruce Ave, Texas, USA
9	Olivia	Garcia	1975-08-12	olivia.garcia@gmail.com	789012345	6 Walnut St, Dallas, USA
10	David	Rodriguez	1992-04-22	david.rodriguez@gmail.com	345678900	7 Ash St, New York, USA

account_id	customer_id	account_type	balance
ACC001	1	Savings	3500.00
ACC002	2	Current	2500.00
ACC003	3	Savings	3500.00
ACC004	4	Current	4500.00
ACC005	5	Savings	5500.00
ACC006	6	Current	6500.00
ACC007	7	Savings	7500.00
ACC008	8	Current	8500.00
ACC009	9	Savings	9500.00
ACC010	10	Zero_Balance	0.00

transaction_id	account_id	transaction_type	amount	transaction_date
1	ACC001	Deposit	500	2024-04-01
2	ACC002	Withdrawal	200	2024-04-02
3	ACC003	Transfer	1000	2024-04-03
4	ACC004	Deposit	800	2024-04-
5	ACC005	Withdrawal	300	2024-04-05
6	ACC006	Transfer	1200	2024-04-06
7	ACC007	Deposit	600	2024-04-07
8	ACC008	Withdrawal	400	2024-04-08
9	ACC009	Transfer	1500	2024-04-09
10	ACC010	Deposit	1000	2024-04-10

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 customers.first_name, customers.last_name, accounts.account_type,customers.email
from customers
join accounts on customers.customer_id=accounts.customer_id;
```

first_name	last_name	account_type	email
John	Doe	Savings	john.doe@gmail.com
Jane	Smith	Current	jane.smith@gmail.com
Alice	Johnson	Savings	alice.johnson@gmail.com
Bob	Brown	Current	bob.brown@gmail.com
Emily	Davis	Savings	emily.davis@gmail.com
Michael	Wilson	Current	michael.wilson@gmail.com
Sophia	Martinez	Savings	sophia.martinez@gmail.com
William	Anderson	Current	william.anderson@gmail.com
Olivia	Garcia	Savings	olivia.garcia@gmail.com
David	Rodriguez	Zero_Balance	david.rodriguez@gmail.com

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

```
select customers.customer_id, accounts.account_type, transactions.amount
from customers
join accounts on customers.customer_id=accounts.customer_id
join transactions on transactions.account_id=accounts.account_id;
```

customer_id	account_type	amount
1	Savings	500
2	Current	200
3	Savings	1000
4	Current	800
5	Savings	300
6	Current	1200
7	Savings	600
8	Current	400
9	Savings	1500
10	Zero_Balance	1000

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

update accounts

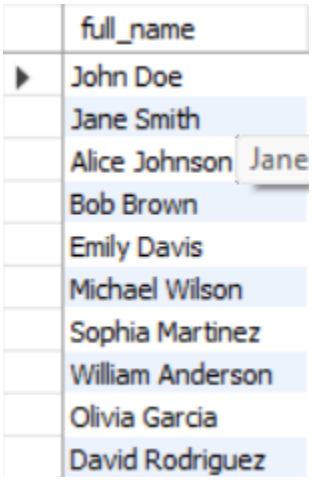
set balance = balance + 1000

where account_id="ACC001";

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

Select concat(first_name,' ',last_name) as full_name

from customers;

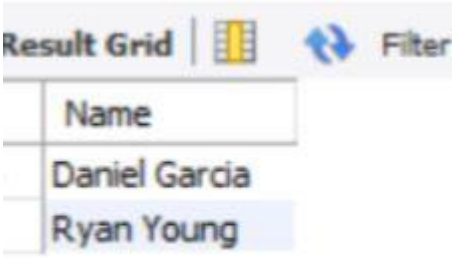


full_name
John Doe
Jane Smith
Alice Johnson
Bob Brown
Emily Davis
Michael Wilson
Sophia Martinez
William Anderson
Olivia Garcia
David Rodriguez

5.
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';



Result Grid | Filter

Name
Daniel Garcia
Ryan Young

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

```
select *
from customers
where address like '%New York%';
```

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1990-05-15	john.doe@gmail.com	234567890	123 Main St, New York, USA
10	David	Rodriguez	1992-04-22	david.rodriguez@gmail.com	345678900	7 Ash St, New York, USA

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

```
select balance
from accounts
where account_id='ACC006';
```

balance
6500.00

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' and balance > '1000';
```

account_type	balance
Current	2500.00
Current	4500.00
Current	6500.00
Current	8500.00

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

```
select *
from transactions
where account_id='ACC003';
```

	transaction_id	account_id	transaction_type	amount	transaction_date
▶	3	ACC003	Transfer	1000	2024-04-03

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

```
select account_id, balance*0.5 as intrest
from accounts
where account_type="Savings";
```

	account_id	intrest
▶	ACC001	1750.000
	ACC003	1750.000
	ACC005	2750.000
	ACC007	3750.000
	ACC009	4750.000

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

```
select account_id
from accounts
where balance < 200;
```

	account_id
▶	ACC010

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

```
select customer_id,first_name,last_name
from customers
where address not like '%new york%';
```

	customer_id	first_name	last_name
▶	2	Jane	Smith
	3	Alice	Johnson
	4	Bob	Brown
	5	Emily	Davis
	6	Michael	Wilson
	7	Sophia	Martinez
	8	William	Anderson
	9	Olivia	Garcia

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

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

```
select avg(balance) as avg_balance
from accounts;
```

	avg_balance
▶	5150.000000

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

```
select *from accounts
order by balance desc limit 10;
```

	account_id	customer_id	account_type	balance
▶	ACC009	9	Savings	9500.00
	ACC008	8	Current	8500.00
	ACC007	7	Savings	7500.00
	ACC006	6	Current	6500.00
	ACC005	5	Savings	5500.00
	ACC004	4	Current	4500.00
	ACC001	1	Savings	3500.00
	ACC003	3	Savings	3500.00
	ACC002	2	Current	2500.00

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

```
select sum(amount) as total_deposit
from transactions
where transaction_date= '2024-04-07' and transaction_type="deposit";
```

	total_deposit
▶	600

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

```
select first_name,last_name,DOB
from customers
order by DOB asc limit 1;

select first_name,last_name,DOB
from customers
order by DOB desc limit 1;
```

	first_name	last_name	DOB
▶	Michael	Wilson	1973-01-30

	first_name	last_name	DOB
▶	Sophia	Martinez	1998-12-18

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

```
select transactions.*,accounts.account_type
from transactions
join accounts on transactions.account_id=accounts.account_id;
```

	transaction_id	account_id	transaction_type	amount	transaction_date	account_type
▶	1	ACC001	Deposit	500	2024-04-01	Savings
	2	ACC002	Withdrawal	200	2024-04-02	Current
	3	ACC003	Transfer	1000	2024-04-03	Savings
	4	ACC004	Deposit	800	2024-04-04	Current
	5	ACC005	Withdrawal	300	2024-04-05	Savings
	6	ACC006	Transfer	1200	2024-04-06	Current
	7	ACC007	Deposit	600	2024-04-07	Savings
	8	ACC008	Withdrawal	400	2024-04-08	Current
	9	ACC009	Transfer	1500	2024-04-09	Savings
	10	ACC010	Deposit	1000	2024-04-10	Zero_Bala

Zero_Bal

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

```
select customers.*,accounts.*
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	account_type	balance
1	John	Doe	1990-05-15	john.doe@gmail.com	234567890	123 Main St, New York, USA	ACC001	1	Savings	3500.00
2	Jane	Smith	1985-09-20	jane.smith@gmail.com	876543210	456 Elm St, Los Angeles, USA	ACC002	2	Current	2500.00
3	Alice	Johnson	1978-03-10	alice.johnson@gmail.com	567890123	789 Oak St, Chicago, USA	ACC003	3	Savings	3500.00
4	Bob	Brown	1995-11-25	bob.brown@gmail.com	890123456	101 Maple Ave, Huston, USA	ACC004	4	Current	4500.00
5	Emily	Davis	1982-07-08	emily.davis@gmail.com	345678901	222 Pine St, Texas, USA	ACC005	5	Savings	5500.00
6	Michael	Wilson	1973-01-30	michael.wilson@gmail.com	901234567	87 Cedar Ave, Philadelphia, USA	ACC006	6	Current	6500.00
7	Sophia	Martinez	1998-12-18	sophia.martinez@gmail.com	678901234	44 Birch St, San Antonio, USA	ACC007	7	Savings	7500.00
8	William	Anderson	1989-06-05	william.anderson@gmail.com	123456789	545 Spruce Ave, Texas, USA	ACC008	8	Current	8500.00
9	Olivia	Garcia	1975-08-12	olivia.garcia@gmail.com	789012345	6 Walnut St, Dallas, USA	ACC009	9	Savings	9500.00
10	David	Rodriguez	1992-04-22	david.rodriguez@gmail.com	345678900	7 Ash St, New York, USA	ACC010	10	Zero_Balance	0.00

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

select transactions.*,customers.*

from transactions

join accounts on transactions.account_id=accounts.account_id

join customers on customers.customer_id=accounts.customer_id

where accounts.account_id="ACC004";

transaction_id	account_id	transaction_type	amount	transaction_date	customer_id	first_name	last_name	DOB	email	phone_number	address
4	ACC004	Deposit	800	2024-04-04	4	Bob	Brown	1995-11-25	bob.brown@gmail.com	890123456	101 Maple Ave, Huston, USA

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

select customer_id

from accounts

group by customer_id

having count(*)>1;

customer_id

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) -

sum(case when transaction_type="withdrawal" then amount else 0 end) as difference

from transactions

group by account_id;

account_id	difference
ACC001	500
ACC002	-200
ACC003	0
ACC004	800
ACC005	-300
ACC006	0
ACC007	600
ACC008	-400
ACC009	0
ACC010	1000

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

```
select account_id,

avg(balance) as avg_balance

from accounts

group by account_id;
```

account_id	avg_balance
ACC001	3500.000000
ACC002	2500.000000
ACC003	3500.000000
ACC004	4500.000000
ACC005	5500.000000
ACC006	6500.000000
ACC007	7500.000000
ACC008	8500.000000
ACC009	9500.000000
ACC010	0.000000

11. Calculate the total balance for each account type.

```
select account_type,

sum(balance)as total_balance

from accounts

group by account_type;
```

account_type	total_balance
Savings	29500.00
Current	22000.00

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

```
select account_id ,count(account_id) as transactions_number
from transactions
group by account_id
order by transactions_number desc;
```

account_id	transactions_number
ACC001	1
ACC002	1
ACC003	1
ACC004	1
ACC005	1
ACC006	1
ACC007	1
ACC008	1
ACC009	1
ACC010	1

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

```
select account_id,concat(first_name," ",last_name) as full_name,balance,account_type
from customers
join accounts on accounts.customer_id=customers.customer_id
group by account_id
order by balance desc;
```

account_id	full_name	balance	account_type
ACC009	Olivia Garcia	9500.00	Savings
ACC008	William Anderson	8500.00	Current
ACC007	Sophia Martinez	7500.00	Savings
ACC006	Michael Wilson	6500.00	Current
ACC005	Emily Davis	5500.00	Savings
ACC004	Bob Brown	4500.00	Current
ACC001	John Doe	3500.00	Savings
ACC003	Alice Johnson	3500.00	Savings
ACC002	Jane Smith	2500.00	Current
ACC010	David Rodriguez	0.00	Zero_Balance

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

```
select account_id,amount,transaction_date
```

```
from transactions
group by account_id,amount,transaction_date
having count(*)>1;
```

	account_id	amount	transaction_date

Tasks 4: Subquery and its type:

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

```
select *
from customers
where customer_id=( select customer_id
from accounts
where balance=(select max(balance)from accounts));
```

	customer_id	first_name	last_name	DOB	email	phone_number	address
▶	9	Olivia	Garcia	1975-08-12	olivia.garcia@gmail.com	789012345	6 Walnut St, Dallas, USA

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

```
select avg(avg_balance) as average_balance
from (
   select avg(balance) as avg_balance
   from accounts
   group by customer_id
   having COUNT(*) > 1
) AS sub;
```

	average_balance
▶	NULL

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

```
select transaction_id,account_id,amount
```

from transactions

where amount>(select avg(amount)

from transactions);

transaction_id	account_id	amount
3	ACC003	1000
4	ACC004	800
6	ACC006	1200
9	ACC009	1500
10	ACC010	1000

4. Identify customers who have no recorded transactions.

select *

from customers

where customer_id not in(select customer_id

from accounts

where account_id in(select account_id

from transactions));

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

where not exists(select transaction_id

from transactions);

total_balance
NULL

6. Retrieve transactions for accounts with the lowest balance.

select *

from transactions

where account_id in(select account_id

from accounts

where balance=(select min(balance)

from accounts));

transaction_id	account_id	transaction_type	amount	transaction_date
10	ACC010	Deposit	1000	2024-04-10
NULL	NULL	NULL	NULL	NULL

7. Identify customers who have accounts of multiple types.

```
select *
from customers
where customer_id in (
select customer_id
from accounts
group by customer_id
having count(distinct account_type) > 1
);
```

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

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

```
select account_type,(COUNT(*) * 100.0) / (SELECT COUNT(*)
from accounts) as percentage
from accounts
group by account_type;
```

account_type	percentage
Savings	50.00000
Current	40.00000
Zero_Balance	10.00000

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

```
select *
from transactions
where account_id in(select account_id
from accounts
```

where customer_id = 6);

	transaction_id	account_id	transaction_type	amount	transaction_date
▶	6	ACC006	Transfer	1200	2024-04-06

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

```
select account_type,(select sum(balance)
from accounts
where account_type = accounts.account_type) as total_balance
from accounts
group by account_type;
```

	account_type	total_balance
▶	Savings	51500.00
	Current	51500.00
	Zero_Balance	51500.00