**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.

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

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

5. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships. • Customers • Accounts • Transactions

Q: create database HMBank;

use HMBank;

create table Customers(

customer\_id int primary key,

first\_name varchar(50) not null,

last\_name varchar(50) not null ,

DOB date not null,

email varchar(100) unique not null,

phone\_number varchar(15) unique not null,

address varchar(200) not null );

create table Accounts(

account\_id int primary key,

customer\_id int not null,

account\_type varchar(50) not null,

balance decimal(15,2) not null default 0.00,

foreign key (customer\_id) references Customers(customer\_id));

create table Transactions(

transaction\_id int primary key,

account\_id int not null,

transaction\_type varchar(50) not null,

amount decimal(15,2) not null default 0.00,

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

Q:insert into Customers (customer\_id, first\_name, last\_name, DOB, email, phone\_number, address) values

(1, 'Divya', 'Guna', '2004-08-03', 'divya@gmail.com', '9941208771', '123 School St, TNR'),

(2, 'Jeevitha', 'Dina', '2003-10-31', 'jee@gmail.com', '9941208971', '12 Tolget St, Chennai'),

(3, 'Krithi', 'Ashwin', '2004-02-04', 'krithi@gmail.com', '9941208791', '13 Ganesh St, Chennai'),

(4, 'Farah', 'Rusha', '2003-10-22', 'farah@gmail.com', '9941208781', '133 Mullai St, Chennai'),

(5, 'Amar', 'Karthik', '2003-08-03', 'amar@gmail.com', '9941208741', '15 Kovil St, Theni'),

(6, 'Abi', 'Sanjesh', '2003-08-19', 'abi@gmail.com', '9941209771', '18 Periyar St, Chennai'),

(7, 'Bhavya', 'Ramesh', '2003-06-07', 'bhavya@gmail.com', '9943208771', '43 Oil St, Avadi'),

(8, 'Josh', 'Paul', '2000-08-03', 'josh@gmail.com', '9942208771', '19 Thomas St, Kerala'),

(9, 'Seungmin', 'Kim', '2000-09-22', 'dandy@gmail.com', '9441208771', '03 Stay St, SK'),

(10, 'Harsh', 'Kumar', '2001-09-03', 'harsh@gmail.com', '9949208771', '11 100 Feet Road, Chennai');

insert into Accounts (account\_id, customer\_id, account\_type, balance) values

(101, 1, 'savings', 5000.00),

(102, 2, 'current', 15000.00),

(103, 3, 'zero\_balance', 0.00),

(104, 4, 'savings', 7500.50),

(105, 5, 'current', 20000.75),

(106, 6, 'savings', 1000.25),

(107, 7, 'zero\_balance', 0.00),

(108, 8, 'current', 12000.00),

(109, 9, 'savings', 8000.00),

(110, 10, 'current', 22000.00);

insert into Transactions (transaction\_id, account\_id, transaction\_type, amount, transaction\_date) values

(1001, 101, 'deposit', 2000.00, '2025-03-10'),

(1002, 102, 'withdrawal', 500.00, '2025-03-11'),

(1003, 103, 'deposit', 1000.00, '2025-03-12'),

(1004, 104, 'withdrawal', 250.75, '2025-03-13'),

(1005, 105, 'deposit', 3000.00, '2025-03-14'),

(1006, 106, 'withdrawal', 100.25, '2025-03-15'),

(1007, 107, 'deposit', 500.00, '2025-03-16'),

(1008, 108, 'withdrawal', 2000.00, '2025-03-17'),

(1009, 109, 'deposit', 1500.00, '2025-03-18'),

(1010, 110, 'withdrawal', 750.00, '2025-03-19');

**2. Write SQL queries for the following tasks:**

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

Q: select c.first\_name,c.last\_name,c.email,a.account\_type

from customers c

join accounts a on c.customer\_id=a.customer\_id;

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

Q: Select c.first\_name,c.last\_name,c.email,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;

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

Q: update accounts

set balance=balance+5000

where account\_id=109;

4. Write a SQL query to Combine first and last names of customers as a full\_name.

Q: select concat(first\_name,' ',last\_name) full\_name

from customers;

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

Q: delete from transactions

where account\_id in(

select account\_id from accounts

where balance=0 and account\_type='savings');

delete from accounts

where balance=0 and account\_type='savings';

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

Q: select \* from customers where address like '%chennai%';

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

Q: select c.first\_name,c.last\_name,a.balance

from customers c

join accounts a on c.customer\_id=a.customer\_id

where a.account\_id=109; (from diff table)

select \* from accounts where account\_id=107;(from same table)

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

Q: select \* from accounts where balance > 1000 and

account\_type='current';

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

Q: select \* from Transactions where account\_id= 109;

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

Q: select \* , balance\*0.03 interset\_accured from accounts where

account\_type='savings';

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

Q: select \* from accounts where balance<10000;

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

Q: select \* from customers where address not like '%chennai%';

**Tasks 3:**

**Aggregate functions, Having, Order By, GroupBy and Joins:**

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

Q: select customer\_id,avg(balance) average\_balance

from accounts

group by customer\_id;

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

Q: select \*

from accounts

order by balance desc limit 10;

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

Q: select transaction\_date , sum(amount) total\_ammount

from transactions

where transaction\_type='deposit' and transaction\_date='2025-03-12'

order by transaction\_date;

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

Q: (select \* ,'Oldest' category

from customers

order by dob asc limit 1)

union all

(select \* ,'Newest' category

from customers

order by dob desc limit 1);

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

Q: select t.transaction\_id, t.account\_id, t.transaction\_type,

t.amount, t.transaction\_date , a.account\_type

from transactions t

join accounts a on t.account\_id=a.account\_id;

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

Q: select c.customer\_id,c.first\_name,c.last\_name,a.account\_id,

a.account\_type, a.balance

from accounts a

join customers c on a.customer\_id=c.customer\_id;

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

Q: select c.customer\_id,c.first\_name,c.last\_name,

t.transaction\_id, a.account\_id, 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 a.account\_id=109;

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

Q: select c.customer\_id,c.first\_name,c.last\_name,

count(account\_id)acc\_count

from accounts a

join customers c on a.customer\_id=c.customer\_id

group by customer\_id

having count(account\_id)>1;

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

Q: select trans\_deposit.account\_id,

sum(trans\_deposit.amount) total\_deposit,sum(trans\_withdrawal.amount) total\_withdrawal,(sum(trans\_deposit.amount) -sum(trans\_withdrawal.amount)) transaction\_difference

from transactions trans\_deposit

join transactions trans\_withdrawal on trans\_deposit.account\_id=trans\_withdrawal.account\_id

where trans\_deposit.transaction\_type='deposit'and trans\_withdrawal.transaction\_type='withdrawal'

group by trans\_deposit.account\_id;

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

Q: select a.account\_id, avg(balance) daily\_average\_balance

from accounts a

join transactions t on a.account\_id=t.account\_id

where t.transaction\_date between '2025-03-11' and '2025-03-19'

group by t.transaction\_date,a.account\_id

order by t.transaction\_date;

11. Calculate the total balance for each account type.

Q: select account\_type , sum(balance) total\_balance\_basedonacctype

from accounts

group by account\_type;

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

Q: select account\_id,count(transaction\_id) no\_of\_transactions

from transactions

group by account\_id

order by no\_of\_transactions desc;

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

Q: select c.customer\_id, c.first\_name, c.last\_name,

a.account\_type, sum(a.balance) high\_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, a.account\_type

having sum(a.balance) > 10000

order by high\_aggregate\_balance desc;

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

Q: select account\_id, transaction\_date, amount,

count(\*) duplicate\_count

from transactions

group by account\_id, transaction\_date, amount

having count(\*) > 1;

**Tasks 4:**

**Subquery and its type:**

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

Q: select c.customer\_id,c.first\_name,c.last\_name,balance max\_balance

from customers c

join accounts a on c.customer\_id=a.customer\_id

where a.balance=(select max(balance) from accounts);

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

Q: select avg(balance) avg\_balance

from accounts

where customer\_id in(select customer\_id

from accounts

group by account\_id

having count(account\_id)>1);

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

Q: select account\_id,transaction\_id,amount

from transactions

where amount>(select avg(amount) from transactions);

4. Identify customers who have no recorded transactions.

Q: select customer\_id,first\_name,last\_name

from customers

where customer\_id not in ( select distinct customer\_id

from accounts

where account\_id in

( select distinct account\_id

from transactions));

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

Q: select sum(balance) total\_balance

from accounts

where account\_id not in (select distinct account\_id

from transactions);

6. Retrieve transactions for accounts with the lowest balance.

Q: select \*

from transactions

where account\_id in (select account\_id

from accounts

where balance=(select min(balance)

from accounts));

7. Identify customers who have accounts of multiple types.

Q: select customer\_id

from accounts

group by customer\_id

having count(distinct(account\_type))>1;

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

Q: select account\_type,

count(\*) account\_type,

(count(\*)\*100/(select count(\*) from accounts))

from accounts

group by account\_type;

9. Retrieve all transactions for a customer with a given customer\_id.

Q: select \*

from transactions

where account\_id in (select account\_id

from accounts

where customer\_id=9);

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

Q: select account\_type,

(select sum(balance)

from accounts a

where a.account\_type=b.account\_type) total\_balance

from accounts b

group by account\_type;

**ER Diagram**

does

has

**Transactions**

**Customers**

**Accounts**

One Many

Many

Many