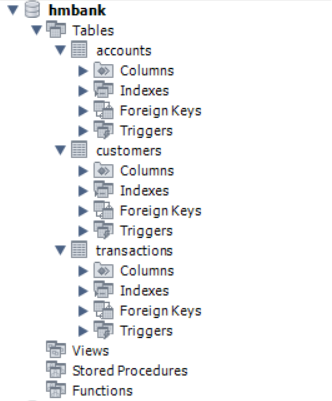
**TASK 1 – DATABASE DESIGN**

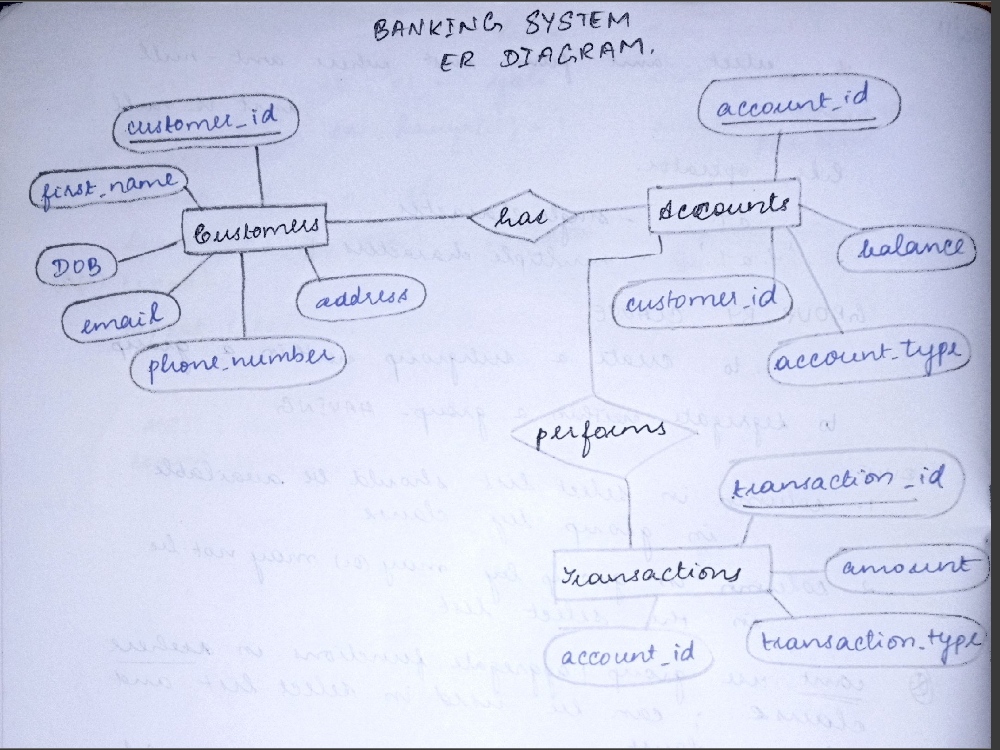
*1. Create the database named "HMBank"*

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

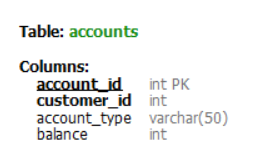
*provided schema.*

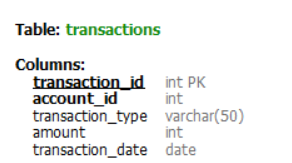


*4. Create an ERD (Entity Relationship Diagram) for the database.*



*5. Create appropriate Primary Key and Foreign Key constraints for referential integrity.*





**TASKS 2: DATA DEFINITION LANGUAGE (DDL):**

*1. Write SQL scripts to create the mentioned tables with appropriate data types, constraints,*

*and relationships.*

*• Customers*

*• Accounts*

*• Transactions*

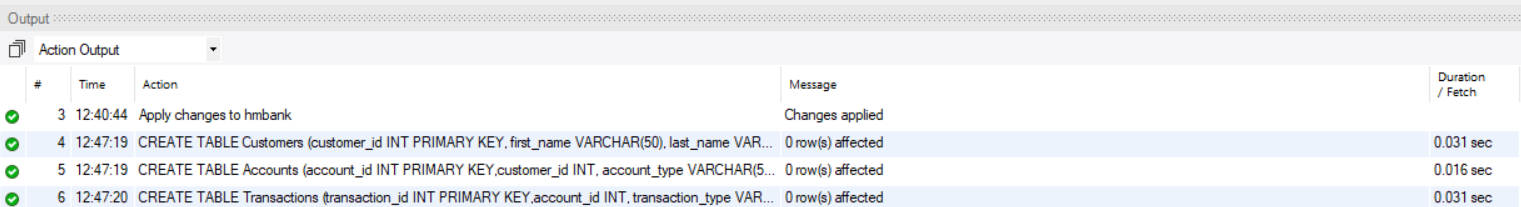
SOURCE CODE :

create table Customers (customer\_id int primary key, first\_name varchar(50), last\_name varchar(50),DOB date, email varchar(100), phone\_number varchar(20),address varchar(255));

create table Accounts (account\_id int primary key,customer\_id int, account\_type varchar(50), balance int, foreign key(customer\_id) references Customers(customer\_id));

create table Transactions (transaction\_id int primary key,account\_id int, transaction\_type varchar(50), amount int, transaction\_date date, foreign key(account\_id) references Accounts(account\_id));

OUTPUT:



**TASKS 3: DATA MANIPULATION LANGUAGE (DML):**

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

*• Customers*

*• Accounts*

*• Transactions*

SOURCE CODE :

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

(101, 'arun', 'kumar', '1992-04-15', 'arun.kumar@email.com', '9876543210', 'pollachi'),

(102, 'kavitha', 'siva', '1988-07-20', 'kavitha.siva@email.com', '7890123456', 'coimbatore'),

(103, 'suresh', 'rajan', '1985-02-10', 'suresh.rajan@email.com', '5678901234', 'madurai'),

(104, 'priya', 'mohan', '1990-11-22', 'priya.mohan@email.com', '1234567890', 'kaniyakumari'),

(105, 'krishna', 'subramaniyam', '1983-06-18', 'krishna.subramaniyam@email.com', '2345678901', 'chennai'),

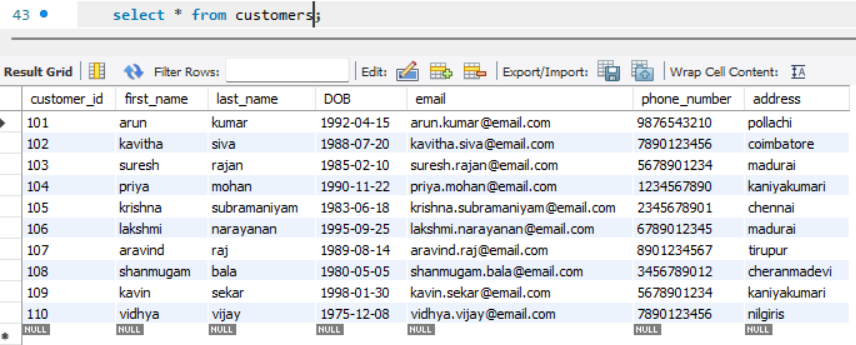
(106, 'lakshmi', 'narayanan', '1995-09-25', 'lakshmi.narayanan@email.com', '6789012345', 'madurai'),

(107, 'aravind', 'raj', '1989-08-14', 'aravind.raj@email.com', '8901234567', 'tirupur'),

(108, 'shanmugam', 'bala', '1980-05-05', 'shanmugam.bala@email.com', '3456789012', 'cheranmadevi'),

(109, 'kavin', 'sekar', '1998-01-30', 'kavin.sekar@email.com', '5678901234', 'kaniyakumari'),

(110, 'vidhya', 'vijay', '1975-12-08', 'vidhya.vijay@email.com', '7890123456', 'nilgiris');



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

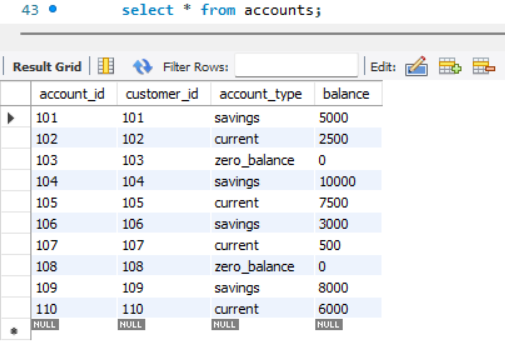
(101, 101, 'savings', 5000.00), (102, 102, 'current', 2500.00),

(103, 103, 'zero\_balance', 0.00), (104, 104, 'savings', 10000.00),

(105, 105, 'current', 7500.00), (106, 106, 'savings', 3000.00),

(107, 107, 'current', 500.00), (108, 108, 'zero\_balance', 0.00),

(109, 109, 'savings', 8000.00), (110, 110, 'current', 6000.00);



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

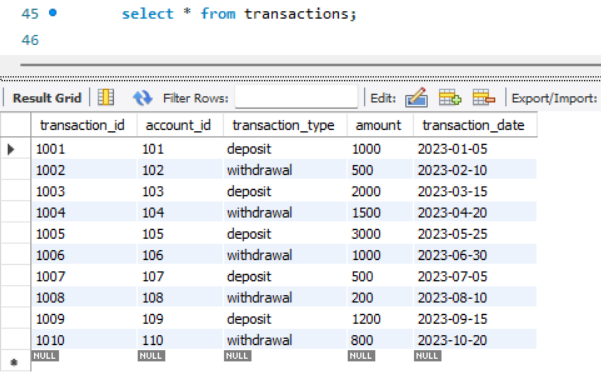
(1001, 101, 'deposit', 1000.00, '2023-01-05'), (1002, 102, 'withdrawal', 500.00, '2023-02-10'),

(1003, 103, 'deposit', 2000.00, '2023-03-15'), (1004, 104, 'withdrawal', 1500.00, '2023-04-20'),

(1005, 105, 'deposit', 3000.00, '2023-05-25'), (1006, 106, 'withdrawal', 1000.00, '2023-06-30'),

(1007, 107, 'deposit', 500.00, '2023-07-05'), (1008, 108, 'withdrawal', 200.00, '2023-08-10'),

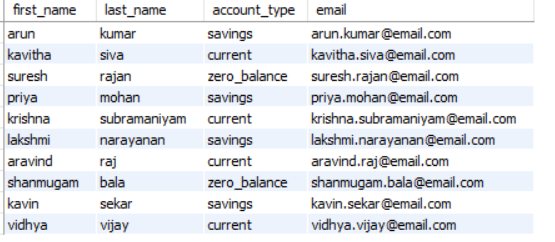
(1009, 109, 'deposit', 1200.00, '2023-09-15'), (1010, 110, 'withdrawal', 800.00, '2023-10-20');



*Write SQL queries for the following tasks:*

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

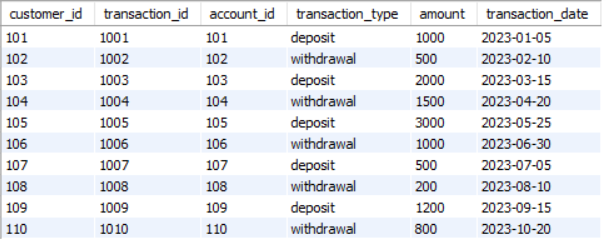
select first\_name, last\_name, account\_type, email from Customers join Accounts on Customers.customer\_id = Accounts.customer\_id;



*3.2 Write a SQL query to list all transaction corresponding to customer.*

select c.customer\_id, t.\* from Customers c join Accounts a on c.customer\_id = a.customer\_id

join Transactions t ON a.account\_id = t.account\_id;



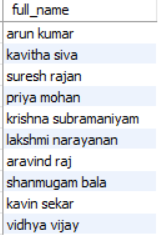
*3.3. Write a SQL query to increase the balance of a specific account by a certain amount.*

update Accounts set balance = balance + 2500 where account\_id = 106;



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

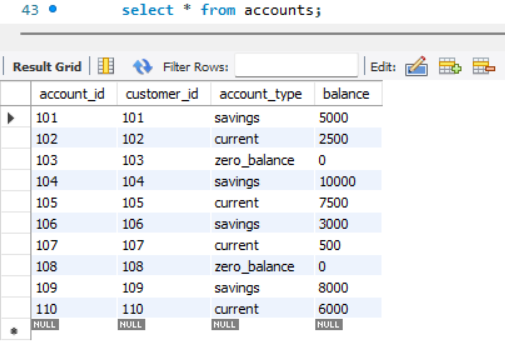
select concat(first\_name, ' ', last\_name) AS full\_name FROM Customers;



*3.5. Write a SQL query to remove accounts with a balance of zero where the account*

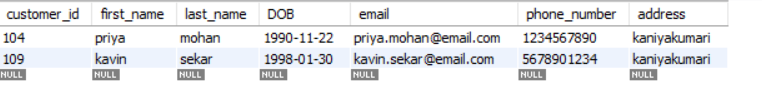
*type is savings.*

delete from Accounts where balance = 0 AND account\_type = 'savings';



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

select \* from Customers where address like 'kaniyakumari';



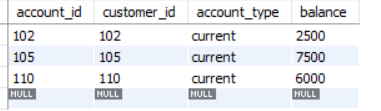
*3.7. Write a SQL query to Get the account balance for a specific account.*

select account\_id, balance from Accounts where account\_id = 102;



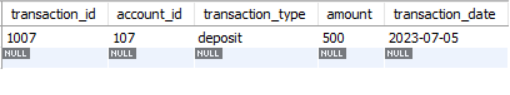
*3.8. Write a SQL query to List all current accounts with a balance greater than $1,000.*

select \* from Accounts where account\_type = 'current' AND balance > 1000;



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

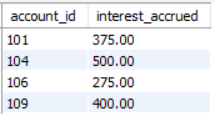
select \* from Transactions where account\_id = 107;



*3.10. Write a SQL query to Calculate the interest accrued on savings accounts based on a*

*given interest rate.*

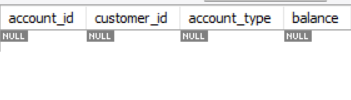
select account\_id, balance \* 0.05 as interest\_accrued from Accounts where account\_type = 'savings';



*3.11. Write a SQL query to Identify accounts where the balance is less than a specified*

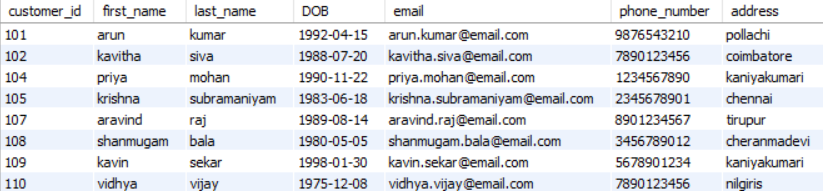
*overdraft limit.*

select \* from Accounts where balance < -500;



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

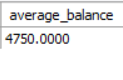
select \* from Customers where address NOT LIKE 'madurai';



**TASKS 4: AGGREGATE FUNCTIONS, GROUPBY AND JOINS:**

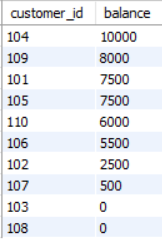
* 1. *Write a SQL query to Find the average account balance for all customers.*

select avg(balance) as average\_balance from Accounts;



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

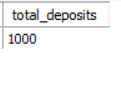
select customer\_id, balance from Accounts order by balance desc limit 10;



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

select sum(amount) as total\_deposits from Transactions where transaction\_type = 'deposit'

and transaction\_date = '2023-01-05';



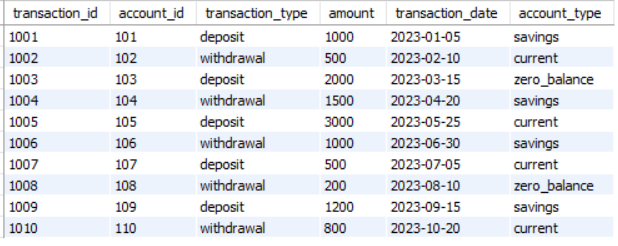
*4.4 Write a SQL query to Find the Oldest and Newest Customers.*

select min(c.customer\_id) as oldest\_customer\_id, max(c.customer\_id) as newest\_customer\_id from Customers c;



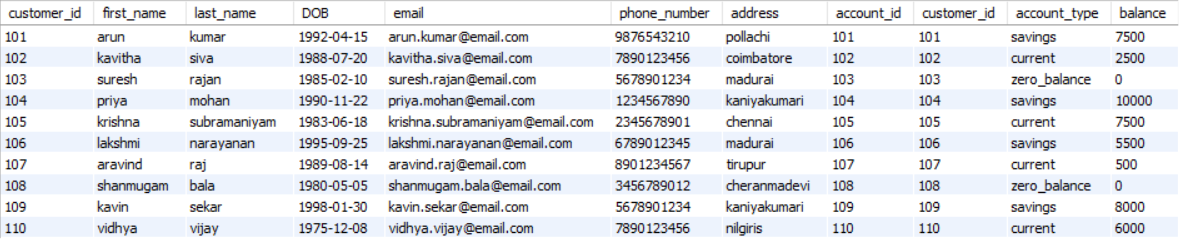
*4.5 Write a SQL query to Retrieve transaction details along with the account type.*

select t.\*, a.account\_type from Transactions t join Accounts a on t.account\_id = a.account\_id;



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

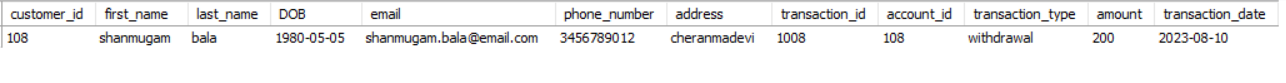
select c.\*, a.\* from Customers c join Accounts a on c.customer\_id = a.customer\_id;



*4.7 Write a SQL query to Retrieve transaction details along with customer information for a*

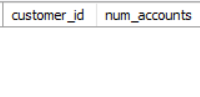
*specific account.*

select c.\*, t.\* FROM Customers c join Accounts a on c.customer\_id = a.customer\_id join Transactions t ON a.account\_id = t.account\_id where a.account\_id = 108;



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

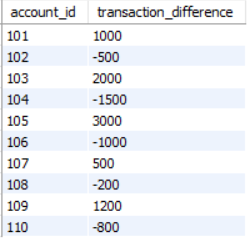
select customer\_id, count(account\_id) as num\_accounts from Accounts group by customer\_id having count(account\_id) > 1;



*4.9. Write a SQL query to Calculate the difference in transaction amounts between deposits and*

*withdrawals.*

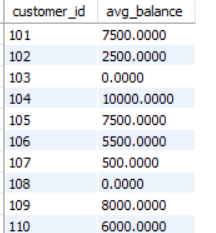
select account\_id, sum(amount \* (transaction\_type = 'deposit') - amount \* (transaction\_type = 'withdrawal')) as transaction\_difference from Transactions group by account\_id;



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

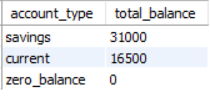
*period.*

select c.customer\_id, avg(a.balance) as avg\_balance from Customers c join Accounts a on c.customer\_id = a.customer\_id group by c.customer\_id;



*4.11. Calculate the total balance for each account type.*

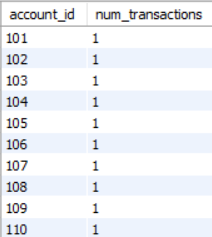
select account\_type, sum(balance) as total\_balance from Accounts group by account\_type;



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

select account\_id, count(transaction\_id) as num\_transactions from Transactions

group by account\_id order by num\_transactions desc;

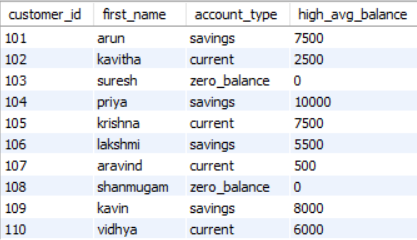


*4.13. List customers with high aggregate account balances, along with their account types.*

select c.customer\_id,c.first\_name, a.account\_type, max(a.balance) as high\_avg\_balance

from Customers c join Accounts a on c.customer\_id = a.customer\_id

group by c.customer\_id,c.first\_name, a.account\_type ;



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

select amount, transaction\_date, account\_id, count(\*) as num\_duplicate

from Transactions group by amount, transaction\_date, account\_id having count(\*) > 1;

