ASSIGNMENT 3 BANKING SYSTEM

J518 Sivasuriya M

TASK 1:

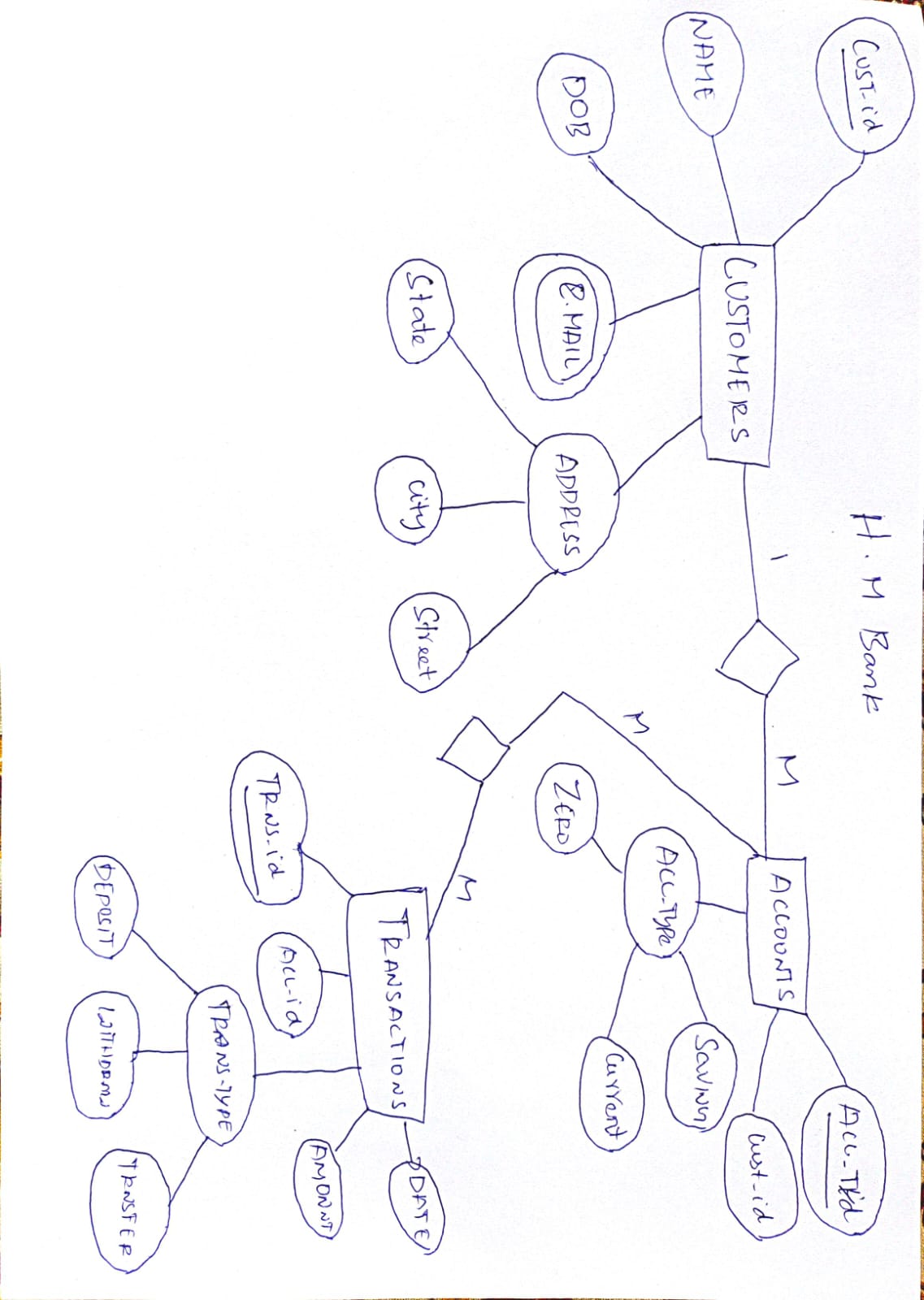
1. Create the database named "HMBank"

CREATE DATABASE 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, -- Unique ID for each customer first\_name VARCHAR(50) NOT NULL, last\_name VARCHAR(50) NOT NULL, DOB DATE NOT NULL, email VARCHAR(100) UNIQUE,phone\_number VARCHAR(15) UNIQUE, address TEXT);

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

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

CREATE TABLE Accounts ( account\_id INT PRIMARY KEY, customer\_id INT,account\_type ENUM('savings', 'current', 'zero\_balance') NOT NULL,balance DECIMAL(12,2) DEFAULT 0.00,FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id) ON DELETE CASCADE);

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

and relationships.

• Customers

• Accounts

• Transactions

CREATE TABLE Transactions (transaction\_id INT PRIMARY KEY AUTO\_INCREMENT,account\_id INT, transaction\_type ENUM('deposit', 'withdrawal', 'transfer') NOT NULL, amount DECIMAL(12,2) NOT NULL CHECK (amount > 0), transaction\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP, FOREIGN KEY (account\_id) REFERENCES Accounts(account\_id) ON DELETE CASCADE

);

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 (first\_name, last\_name, DOB, email, phone\_number, address) VALUES

('John', 'Doe', '1985-07-15', 'john.doe@email.com', '9876543210', '123 Main St, NY'),

('Alice', 'Smith', '1990-02-20', 'alice.smith@email.com', '9123456789', '456 Elm St, LA'),

('Bob', 'Johnson', '1982-05-10', 'bob.johnson@email.com', '9988776655', '789 Pine St, TX'),

('Emma', 'Brown', '1995-09-25', 'emma.brown@email.com', '9112233445', '321 Oak St, CA'),

('David', 'Miller', '1988-12-30', 'david.miller@email.com', '9001122334', '654 Cedar St, FL'),

('Sophia', 'Wilson', '1993-06-18', 'sophia.wilson@email.com', '9556677889', '987 Maple St, IL'),

('Liam', 'Moore', '1979-11-07', 'liam.moore@email.com', '9445566778', '741 Birch St, NV'),

('Olivia', 'Taylor', '1986-04-03', 'olivia.taylor@email.com', '9334455667', '852 Fir St, WA'),

('James', 'Anderson', '1991-08-22', 'james.anderson@email.com', '9223344556', '963 Willow St, OR'),

('Emily', 'Thomas', '1983-01-11', 'emily.thomas@email.com', '9112233445', '159 Redwood St, MI');

INSERT INTO Accounts (customer\_id, account\_type, balance) VALUES

(1, 'savings', 5000.00),

(2, 'current', 10000.50),

(3, 'savings', 7500.75),

(4, 'savings', 12000.20),

(5, 'current', 8000.00),

(6, 'savings', 15000.90),

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

(8, 'savings', 20000.00),

(9, 'current', 17500.50),

(10, 'savings', 3000.25);

INSERT INTO Transactions (account\_id, transaction\_type, amount) VALUES

(1, 'deposit', 1000.00),

(2, 'withdrawal', 500.00),

(3, 'deposit', 2000.50),

(4, 'withdrawal', 800.75),

(5, 'deposit', 3000.00),

(6, 'transfer', 1500.25),

(7, 'deposit', 5000.00),

(8, 'withdrawal', 700.00),

(9, 'transfer', 1200.90),

(10, 'deposit', 2500.50);

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;

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

select \* from transactions where accout\_id=101;

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

update accounts set balance=balance+500 where account\_type='savings';

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;

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;

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

select \* from customers where address like '%NY';

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

select sum(balance) from accounts where account\_type='savings';

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

select \* from acconuts where account\_type='current' and balance>1000;

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

select \* from transactions where account\_id=101;

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

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

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

select \* from accounts where balance<1000;

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

select \* from customers where address not like '%NY';

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 average\_balance from accounts;

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

select balance from accounts order by balance desc limit 10;

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

select sum(amount) from transactions where day(transaction\_date)=18;

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

select c.customer\_id,concat(first\_name,last\_name) as oldest\_customer from customers c join accounts a on c.customer\_id=a.customer\_id join transactions t on a.account\_id=t.account\_id order by transaction\_date desc limit 1;

SELECT c.customer\_id,concat(first\_name,last\_name) as newest\_customer from customers c join accounts a on c.customer\_id=a.customer\_id join transactions t on a.account\_id=t.account\_id order by transaction\_date limit 1;

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

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

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

select c.customer\_id,c.first\_name,c.last\_name,c.DOB,c.email,c.phone\_number,c.address,a.account\_id,a.account\_type,a.balance from customers c join accounts a on c.customer\_id=a.customer\_id;

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

select t.\*,c.customer\_id,c.first\_name,c.last\_name from customers c join accounts a on c.customer\_id=a.customer\_id join transactions t on t.account\_id=a.account\_id where t.account\_id=101;

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

select c.\*,count(a.account\_id) from customers c join accounts a on c.customer\_id=a.customer\_id group by c.customer\_id having count(a.account\_id)>1;

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

select sum(case when transaction\_type='deposits' then amount end) - sum(case when transactions\_type='withdrawal' then amount end) as difference from transactions;

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

period.

select avg(amount) from transactions where transaction\_date between '2025-03-18' and '2025-04-15' group by account\_id;

11. Calculate the total balance for each account type.

SELECT SUM(BALANCE) FROM ACCOUNTS GROUP BY account\_type;

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

select count(account\_id) from transactions group by account\_id order by account\_id desc;

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

select max(balance) as highest\_balance, account\_type from accounts;

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

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

Tasks 4: Subquery and its type:

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

select c.\*,a.balance from customers c join accounts a on a.customer\_id=c.customer\_id where balance =(select max(balance) from accounts);

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

select customer\_id,avg(balance) from accounts group by customer\_id having count(account\_id)>1;

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

select a.\* from accounts a join transactions t on t.account\_id=a.account\_id where t.amount>(select avg(amount) from transactions);

4. Identify customers who have no recorded transactions.

select customer\_id from customers where customer\_id not in (select customer\_id from accounts where account\_id in (select account\_id from transactions));

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

select sum(balance) from accounts where account\_id not in(select account\_id from transactions);

6. Retrieve transactions for accounts with the lowest balance.

select \* from transactions where account\_id in(select account\_id from accounts where balance in (select min(balance) from accounts));

7. Identify customers who have accounts of multiple types.

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.

select account\_type, (count(\*)/(select count(\*) from accounts))\*100.0 as percentage from accounts group by account\_type;

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 in(select customer\_id from customers where customer\_id=101));

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

select distinct account\_type ,(select sum(balance) from accounts a where a.account\_type=b.account\_type ) as total\_balance from accounts b;