SQL ASSIGNMENT

NAME: HEMASHRI S

TITLE: BANKING SYSTEM

```
drop database if exists HMBank;
CREATE DATABASE HMBank;
USE HMBank;

-- Customers Table
CREATE TABLE Customers (
    customer_id INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    DOB DATE,
    email VARCHAR(100) UNIQUE,
    phone_number VARCHAR(15),
    address VARCHAR(255)
);
INSERT INTO Customers (first_name, last_name, DOB, email, phone_number, address) VALUES
('Aarav', 'Sharma', '1990-05-14', 'aarav.sharma@email.com', '9876543210', 'Mumbai'),
('Priya', 'Mehta', '1992-07-20', 'priya.mehta@email.com', '9123456780', 'Chennai'),
('Ravi', 'Kumar', '1985-11-01', 'ravi.kumar@email.com', '9988776655', 'Delhi'),
('Sneha', 'Patel', '1993-02-18', 'sneha.patel@email.com', '9012345678', 'Ahmedabad'),
('Kiran', 'Joshi', '1991-12-05', 'kiran.joshi@email.com', '9876012345', 'Bangalore'),
('Deepa', 'Rao', '1988-03-22', 'deepa.rao@email.com', '9900112233', 'Hyderabad'),
('Manoj', 'Nair', '1995-10-10', 'manoj.nair@email.com', '9786543210', 'Kochi'),
('Lakshmi', 'Iyer', '1990-08-25', 'lakshmi.iyer@email.com', '9123456700', 'Pune'),
('Anil', 'Kapoor', '1982-06-12', 'anil.kapoor@email.com', '9811122233', 'Delhi'),
('Meena', 'Das', '1994-09-17', 'meena.das@email.com', '9909988776', 'Kolkata');

select * from customers;

-- Accounts Table
CREATE TABLE Accounts (
    account_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    account_type ENUM('savings', 'current', 'zero_balance'),
    balance DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

INSERT INTO Accounts (customer_id, account_type, balance) VALUES
(1, 'savings', 2000.00),
(2, 'current', 15000.00),
(3, 'savings', 0.00),
(4, 'zero_balance', 500.00),
(5, 'current', 7000.00),
(6, 'savings', 100.00),
(7, 'current', 2500.00),
(8, 'zero_balance', 0.00),
(9, 'savings', 3000.00),
(10, 'current', 1200.00);

select * from accounts;

-- Transactions Table
CREATE TABLE Transactions (
    transaction_id INT PRIMARY KEY AUTO_INCREMENT,
    account_id INT,
    transaction_type ENUM('deposit', 'withdrawal', 'transfer'),
    amount DECIMAL(10, 2),
    transaction_date DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
INSERT INTO Transactions (account_id, transaction_type, amount) VALUES
(1, 'deposit', 2000.00),
(2, 'deposit', 5000.00),
(3, 'withdrawal', 200.00),
(4, 'deposit', 500.00),
(5, 'withdrawal', 3000.00),
(6, 'deposit', 100.00),
(7, 'transfer', 1000.00),
(8, 'deposit', 0.00),
(9, 'withdrawal', 500.00),
(10, 'deposit', 1200.00);

select * from transactions;

-- task 2
-- Retrieve 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;

-- List all transactions with corresponding customer
SELECT t.transaction_id, t.transaction_type, t.amount, c.first_name, c.last_name
FROM Transactions t
JOIN Accounts a ON t.account_id = a.account_id
JOIN Customers c ON a.customer_id = c.customer_id;

-- Increase balance of a specific account by ₹500
UPDATE Accounts
SET balance = balance + 500
WHERE account_id = 1;

-- Combine first and last names as full name
SELECT CONCAT(first_name, ' ', last_name) AS full_name
```

```
FROM Customers;
```

-- Remove accounts with zero balance and type 'savings'

```
ALTER TABLE Transactions
DROP FOREIGN KEY transactions_ibfk_1;

ALTER TABLE Transactions
ADD CONSTRAINT transactions_ibfk_1
FOREIGN KEY (account_id)
REFERENCES Accounts(account_id)
ON DELETE CASCADE;

-- DELETE FROM Accounts
-- WHERE balance = 0 AND account_type = 'savings';
```

-- Find customers living in a specific city

```
SELECT * FROM Customers
WHERE address LIKE '%Delhi%';
```

-- Get the account balance for a specific account

```
SELECT balance
FROM Accounts
WHERE account_id = 5;
```

-- List all current accounts with balance > ₹1000

```sql
SELECT * FROM Accounts
WHERE account_type = 'current' AND balance > 1000;

-- Retrieve all transactions for a specific account
SELECT * FROM Transactions
WHERE account_id = 2;

-- Calculate interest on savings accounts
SELECT account_id, balance, balance * 0.04 AS interest -- 4%
FROM Accounts
WHERE account_type = 'savings';

 -- Identify accounts below a specific overdraft limit
 SELECT *
FROM Accounts
WHERE balance < 500;

 -- Find customers not living in a specific city
 SELECT *
FROM Customers
WHERE address NOT LIKE '%Chennai%';

 -- task 3

 -- Average account balance
 SELECT AVG(balance) AS average_balance
FROM Accounts;
```

-- Top 10 highest account balances

```
SELECT *
FROM Accounts
ORDER BY balance DESC
LIMIT 10;
```

-- total deposits on a specific date

```
SELECT SUM(amount) AS total_deposit
FROM Transactions
WHERE transaction_type = 'deposit'
  AND DATE(transaction_date) = '2025-04-10';
```

-- Oldest and newest customers

```
SELECT * FROM Customers
ORDER BY DOB ASC
LIMIT 1;  -- Oldest

SELECT * FROM Customers
ORDER BY DOB DESC
LIMIT 1;  -- Newest
```

-- Transaction details with account type

```sql
SELECT t.*, a.account_type
FROM Transactions t
JOIN Accounts a ON t.account_id = a.account_id;

 -- Customers with their account details
SELECT c.*, a.account_id, a.account_type, a.balance
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id;

-- Transaction details + customer info for a specific account
SELECT t.*, c.first_name, c.last_name
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 = 1;

-- Customers with more than one account
SELECT customer_id, COUNT(account_id) AS account_count
FROM Accounts
GROUP BY customer_id
HAVING COUNT(account_id) > 1;

-- Difference between total deposits and withdrawals
SELECT
  (SELECT SUM(amount) FROM Transactions WHERE transaction_type = 'deposit') -
 (SELECT SUM(amount) FROM Transactions WHERE transaction_type = 'withdrawal') AS difference;
```

-- Average daily balance per account over a period

```
SELECT account_id, AVG(balance) AS avg_balance
FROM Accounts
GROUP BY account_id;
```

-- Total balance by account type

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

-- Accounts with the most transactions

```
SELECT account_id, COUNT(*) AS transaction_count
FROM Transactions
GROUP BY account_id
ORDER BY transaction_count DESC;
```

-- Customers with high total balances and their account types

```
SELECT c.customer_id, c.first_name, c.last_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 total_balance > 5000;

-- Duplicate transactions by amount, date, and account
SELECT account_id, amount, DATE(transaction_date), COUNT(*) as duplicate_count
FROM Transactions
GROUP BY account_id, amount, DATE(transaction_date)
HAVING COUNT(*) > 1;

-- Task 4:
-- Retrieve customer(s) with the highest account balance

SELECT c.*
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
WHERE a.balance = (
  SELECT MAX(balance)
  FROM Accounts
);

-- Average account balance for customers with more than one account

SELECT 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
);
```

-- Accounts with transactions above average transaction amount

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

-- Customers with no recorded transactions

```
SELECT DISTINCT c.*
FROM Customers c
WHERE c.customer_id NOT IN (
  SELECT a.customer_id
  FROM Accounts a
  JOIN Transactions t ON a.account_id = t.account_id
);
```

-- Total balance of accounts with no recorded transactions

```
SELECT SUM(balance) AS total_balance
FROM Accounts
WHERE account_id NOT IN (
  SELECT DISTINCT account_id
  FROM Transactions
);
```

-- 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
  )
);
```

-- Customers with multiple account types

```
SELECT customer_id
FROM Accounts
GROUP BY customer_id
HAVING COUNT(DISTINCT account_type) > 1;
```

-- Percentage of each account type from total

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

-- All transactions for a customer by their customer_id

```
SELECT t.*
FROM Transactions t
JOIN Accounts a ON t.account_id = a.account_id
WHERE a.customer_id = 101;
```

-- Total balance for each account type using subquery in SELECT

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

```
FROM Accounts a1;
```