Bank database Management System
1. Customer (customer_id, name, address, phone, email)
2. Account (account_id, customer_id, account_type, balance, branch_id)
3. Branch (branch_id, branch_name, location, manager_id)
4. Transaction (transaction_id, account_id, transaction_type, amount, transaction_date)
5. Loan (loan_id, customer_id, amount, loan_type, status)
6. Employee (employee_id, name, position, branch_id, salary)

Write queries for the following questions:
1. List all customers and their account details
2. Find the total balance in each branch
3. Find customers who have taken loans greater than Rs. 1,00,000
4. Retrieve transaction history for a specific account (e.g., Account ID: 101)
5. Find customers who have both a loan and an account
6. Create a view of high-value customers (balance > 1,00,000)

```
CREATE TABLE Customer (
    customer_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    address VARCHAR(255),
    phone VARCHAR(15),
    email VARCHAR(100)
);

CREATE TABLE Branch (
    branch_id INT PRIMARY KEY AUTO_INCREMENT,
    branch_name VARCHAR(100),
    location VARCHAR(100),
    manager_id INT
);

CREATE TABLE Employee (
    employee_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    position VARCHAR(100),
    branch_id INT,
    salary DECIMAL(10,2),
    FOREIGN KEY (branch_id) REFERENCES Branch(branch_id)
);

CREATE TABLE Account (
    account_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    account_type VARCHAR(50),
    balance DECIMAL(12,2),
    branch_id INT,
```

```sql
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
    FOREIGN KEY (branch_id) REFERENCES Branch(branch_id)
);

CREATE TABLE Transaction (
    transaction_id INT PRIMARY KEY AUTO_INCREMENT,
    account_id INT,
    transaction_type VARCHAR(50),
    amount DECIMAL(10,2),
    transaction_date DATE,
    FOREIGN KEY (account_id) REFERENCES Account(account_id)
);

CREATE TABLE Loan (
    loan_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT,
    amount DECIMAL(12,2),
    loan_type VARCHAR(50),
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);

-- Customers
INSERT INTO Customer (name, address, phone, email) VALUES
('Rahul', 'Delhi', '9876543210', 'rahul@gmail.com'),
('Sneha', 'Mumbai', '9123456780', 'sneha@gmail.com'),
('Amit', 'Bangalore', '9988776655', 'amit@gmail.com');

-- Branches
INSERT INTO Branch (branch_name, location, manager_id) VALUES
('Main Branch', 'Delhi', 1),
('City Branch', 'Mumbai', 2);

-- Employees
INSERT INTO Employee (name, position, branch_id, salary) VALUES
('Rakesh', 'Manager', 1, 80000),
('Priya', 'Clerk', 2, 40000);

-- Accounts
INSERT INTO Account (customer_id, account_type, balance, branch_id) VALUES
(1, 'Savings', 120000.00, 1),
(2, 'Current', 95000.00, 2),
(3, 'Savings', 180000.00, 1);

-- Transactions
INSERT INTO Transaction (account_id, transaction_type, amount, transaction_date)
VALUES
(1, 'Deposit', 5000.00, '2024-01-15'),
(1, 'Withdrawal', 2000.00, '2024-02-10'),
(2, 'Deposit', 7000.00, '2024-03-01');
```

```sql
-- Loans
INSERT INTO Loan (customer_id, amount, loan_type, status) VALUES
(1, 150000.00, 'Home', 'Approved'),
(2, 80000.00, 'Personal', 'Pending'),
(3, 200000.00, 'Car', 'Approved');

-- 1. List all customers and their account details
SELECT c.name, c.phone, a.account_id, a.account_type, a.balance
FROM Customer c
JOIN Account a ON c.customer_id = a.customer_id;

-- 2. Find the total balance in each branch
SELECT b.branch_name, SUM(a.balance) AS total_balance
FROM Branch b
JOIN Account a ON b.branch_id = a.branch_id
GROUP BY b.branch_name;

-- 3. Find customers who have taken loans greater than Rs. 1,00,000
SELECT c.name, l.amount
FROM Customer c
JOIN Loan l ON c.customer_id = l.customer_id
WHERE l.amount > 100000;

-- 4. Retrieve transaction history for a specific account (e.g., Account ID: 1)
SELECT * FROM Transaction
WHERE account_id = 1;

-- 5. Find customers who have both a loan and an account
SELECT c.name
FROM Customer c
JOIN Account a ON c.customer_id = a.customer_id
JOIN Loan l ON c.customer_id = l.customer_id;

-- 6. Create a view of high-value customers (balance > 1,00,000)
CREATE VIEW HighValueCustomers AS
SELECT c.name, a.account_id, a.balance
FROM Customer c
JOIN Account a ON c.customer_id = a.customer_id
WHERE a.balance > 100000;

SELECT * FROM HighValueCustomers;
```