Problem 6: Bank Transactions

ER Diagram Description:

- Entities: Customers, Accounts, Transactions

- Relationships: One customer has many accounts; one account has many transactions.

Table Creation:

```
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(100),
    Address VARCHAR(100)
);

CREATE TABLE Accounts (
    AccountID INT PRIMARY KEY,
    CustomerID INT,
    Balance DECIMAL(10,2),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);

CREATE TABLE Transactions (
    TransID INT PRIMARY KEY,
    AccountID INT,
    Amount DECIMAL(10,2),
    TransDate DATE,
    Type VARCHAR(10),
```

```
    FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)
);
```

Sample Data:

```
INSERT INTO Customers VALUES
(1, 'Amit Joshi', 'Pune'),
(2, 'Reena Gupta', 'Mumbai'),
(3, 'Karan Shah', 'Pune');

INSERT INTO Accounts VALUES
(101, 1, 50000.00),
(102, 2, 30000.00),
(103, 3, 45000.00);

INSERT INTO Transactions VALUES
(1001, 101, 5000.00, '2025-04-01', 'Debit'),
(1002, 102, 8000.00, '2025-03-15', 'Credit'),
(1003, 101, 2000.00, '2025-04-05', 'Credit'),
(1004, 103, 1000.00, '2025-03-20', 'Debit');
```

Queries:

```
-- 1
SELECT * FROM Customers WHERE Address = 'Pune';

-- 2
SELECT * FROM Transactions WHERE AccountID = 101;
```

```
-- 3
SELECT AccountID, SUM(Amount) AS TotalAmount FROM Transactions GROUP BY AccountID;

-- 4
SELECT AccountID FROM Transactions WHERE Type = 'Debit'
EXCEPT
SELECT AccountID FROM Transactions WHERE Type = 'Credit';

-- 5
SELECT Customers.Name, Accounts.Balance FROM Accounts JOIN Customers ON Accounts.CustomerID =
Customers.CustomerID;
```

Problem 7: Movie Booking System

ER Diagram Description:

- Entities: Movies, Theaters, Bookings
- Relationships: A movie is booked at many theaters; a theater hosts many movies.

Table Creation:

```
CREATE TABLE Movies (
    MovieID VARCHAR(10) PRIMARY KEY,
    Title VARCHAR(100),
    Duration INT
);

CREATE TABLE Theaters (
```

```
    TheaterID VARCHAR(10) PRIMARY KEY,
    Name VARCHAR(100),
    City VARCHAR(50)
);

CREATE TABLE Bookings (
    BookingID INT PRIMARY KEY,
    MovieID VARCHAR(10),
    TheaterID VARCHAR(10),
    ShowDate DATE,
    TicketsBooked INT,
    FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
    FOREIGN KEY (TheaterID) REFERENCES Theaters(TheaterID)
);
```

Sample Data:

```
INSERT INTO Movies VALUES
('M101', 'Avengers', 180),
('M102', 'Inception', 148),
('M103', 'Coco', 105);

INSERT INTO Theaters VALUES
('T1', 'PVR', 'Delhi'),
('T2', 'INOX', 'Mumbai'),
('T3', 'Cinepolis', 'Delhi');

INSERT INTO Bookings VALUES
```

(201, 'M101', 'T1', '2025-04-10', 120),

(202, 'M102', 'T2', '2025-04-12', 75),

(203, 'M101', 'T2', '2025-04-15', 100),

(204, 'M103', 'T1', '2025-04-11', 60);

Queries:

-- 1

SELECT * FROM Movies WHERE Duration > 120;

-- 2

SELECT * FROM Bookings WHERE MovieID = 'M101';

-- 3

SELECT MovieID, SUM(TicketsBooked) AS TotalTickets FROM Bookings GROUP BY MovieID;

-- 4

SELECT MovieID FROM Bookings WHERE TheaterID = 'T1'

UNION

SELECT MovieID FROM Bookings WHERE TheaterID = 'T2';

-- 5

SELECT Movies.Title, Theaters.Name FROM Bookings JOIN Movies ON Bookings.MovieID = Movies.MovieID JOIN

Theaters ON Bookings.TheaterID = Theaters.TheaterID;

Problem 8: Online Shopping Portal

ER Diagram Description:

- Entities: Customers, Products, Orders

- Relationships: Customers place orders for products.

Table Creation:

```
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(100)
);

CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    Name VARCHAR(100),
    Price DECIMAL(10,2)
);

CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    ProductID INT,
    OrderDate DATE,
    Quantity INT,
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```

Sample Data:

```
INSERT INTO Customers VALUES
(1, 'Arjun Patel', 'arjun@example.com'),
(2, 'Sneha Rao', 'sneha@example.com'),
(3, 'Vikram Jain', 'vikram@example.com');

INSERT INTO Products VALUES
(101, 'Smartphone', 20000.00),
(102, 'Bluetooth Speaker', 3500.00),
(103, 'Backpack', 1500.00);

INSERT INTO Orders VALUES
(301, 1, 101, '2025-04-02', 1),
(302, 2, 102, '2025-04-03', 6),
(303, 1, 102, '2025-04-05', 1),
(304, 3, 103, '2025-03-30', 2);
```

Queries:

```
-- 1
SELECT DISTINCT Customers.* FROM Orders JOIN Customers ON Orders.CustomerID = Customers.CustomerID
WHERE Orders.OrderDate > '2025-04-01';

-- 2
SELECT ProductID FROM Orders GROUP BY ProductID HAVING SUM(Quantity) > 5;

-- 3
SELECT Products.Name, SUM(Orders.Quantity * Products.Price) AS Revenue FROM Orders JOIN Products ON
```

```
Orders.ProductID = Products.ProductID GROUP BY Products.Name;


-- 4
SELECT CustomerID FROM Orders WHERE ProductID = 101
UNION
SELECT CustomerID FROM Orders WHERE ProductID = 102;


-- 5
SELECT Customers.Name, Products.Name FROM Orders JOIN Customers ON Orders.CustomerID =
Customers.CustomerID JOIN Products ON Orders.ProductID = Products.ProductID;
```