Problem 12: Library Fines System

ER Diagram Description:

Entities: Students, Books, Fines

Relationships: A student is fined for a book.

Table Creation:

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(100)
);

CREATE TABLE Books (
    BookID INT PRIMARY KEY,
    Title VARCHAR(100)
);

CREATE TABLE Fines (
    StudentID INT,
    BookID INT,
    ReturnDate DATE,
    FineAmount DECIMAL(10,2),
    PRIMARY KEY (StudentID, BookID),
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (BookID) REFERENCES Books(BookID)
);
```

Sample Data:

```
INSERT INTO Students VALUES
(1, 'Anjali Sharma'),
(2, 'Rohan Gupta'),
(3, 'Neha Yadav');

INSERT INTO Books VALUES
(101, 'Data Structures'),
(102, 'Operating Systems'),
(103, 'Database Systems');

INSERT INTO Fines VALUES
(1, 101, '2025-04-02', 120.00),
(2, 102, '2025-03-30', 90.00),
(1, 102, '2025-04-05', 50.00),
(3, 103, '2025-04-07', 200.00);
```

Queries:

```
SELECT DISTINCT Students.* FROM Fines JOIN Students ON Fines.StudentID = Students.StudentID WHERE FineAmount > 0;
SELECT * FROM Fines WHERE FineAmount > 100;
SELECT StudentID, SUM(FineAmount) AS TotalFine FROM Fines GROUP BY StudentID;
SELECT StudentID FROM Fines WHERE BookID = 101 INTERSECT SELECT StudentID FROM Fines WHERE BookID = 102;
SELECT Students.Name, Books.Title, Fines.FineAmount FROM Fines JOIN Students ON Fines.StudentID = Students.StudentID JOIN Books ON Fines.BookID = Books.BookID;
```

Problem 13: Music Streaming Service

ER Diagram Description:

Entities: Songs, Artists, Playlists

Relationships: Users add songs to playlists, songs have artists.

Table Creation:

```
CREATE TABLE Songs (
    SongID INT PRIMARY KEY,
    Title VARCHAR(100),
    Duration INT
);

CREATE TABLE Artists (
    ArtistID INT PRIMARY KEY,
    Name VARCHAR(100)
);

CREATE TABLE Playlists (
    UserID INT,
    SongID INT,
    PRIMARY KEY (UserID, SongID),
    FOREIGN KEY (SongID) REFERENCES Songs(SongID)
);
```

Sample Data:

```
INSERT INTO Songs VALUES
(201, 'Blinding Lights', 210),
(202, 'Levitating', 180),
(203, 'Bohemian Rhapsody', 360);

INSERT INTO Artists VALUES
(301, 'The Weeknd'),
(302, 'Dua Lipa'),
(303, 'Queen');

INSERT INTO Playlists VALUES
(1, 201),
(1, 202),
(2, 202),
(2, 203),
(3, 201),
(3, 203);
```

Queries:

```
SELECT * FROM Songs WHERE Duration > 300;
-- Optional: Add ArtistID to Songs table
SELECT * FROM Songs WHERE ArtistID = 301;
SELECT UserID, COUNT(*) AS SongCount FROM Playlists GROUP BY UserID;
SELECT UserID FROM Playlists WHERE SongID = 201 INTERSECT SELECT UserID FROM Playlists WHERE SongID
= 203;
SELECT Songs.Title, Artists.Name FROM Songs JOIN Artists ON Songs.ArtistID = Artists.ArtistID;
```

Problem 14: School Transport System

ER Diagram Description:

Entities: Students, Routes, Assignments

Relationships: A student is assigned to a route.

Table Creation:

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(100),
    Class VARCHAR(10)
);

CREATE TABLE Routes (
    RouteID VARCHAR(10) PRIMARY KEY,
    StartPoint VARCHAR(50),
    EndPoint VARCHAR(50)
);

CREATE TABLE Assignments (
    StudentID INT,
    RouteID VARCHAR(10),
    BusNumber VARCHAR(10),
    PRIMARY KEY (StudentID, RouteID),
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
```

```
    FOREIGN KEY (RouteID) REFERENCES Routes(RouteID)
);
```

Sample Data:

```
INSERT INTO Students VALUES
(1, 'Maya Sen', '10A'),
(2, 'Kunal Roy', '9B'),
(3, 'Farhan Ali', '8C');

INSERT INTO Routes VALUES
('R01', 'Sector 1', 'School'),
('R02', 'Sector 5', 'School');

INSERT INTO Assignments VALUES
(1, 'R01', 'B1'),
(2, 'R02', 'B2'),
(3, 'R01', 'B1'),
(1, 'R02', 'B2');
```

Queries:

```
SELECT Students.* FROM Assignments JOIN Students ON Assignments.StudentID = Students.StudentID WHERE
RouteID = 'R01';
-- Query for buses driven by 'John' would require a Driver table (not included).
SELECT RouteID, COUNT(*) AS StudentCount FROM Assignments GROUP BY RouteID;
SELECT StudentID FROM Assignments WHERE RouteID = 'R01' INTERSECT SELECT StudentID FROM Assignments
WHERE RouteID = 'R02';
SELECT Students.Name, Routes.StartPoint, Routes.EndPoint FROM Assignments JOIN Students ON
```

Assignments.StudentID = Students.StudentID JOIN Routes ON Assignments.RouteID = Routes.RouteID;