-- Create Tables

CREATE TABLE Books (

    BookID INTEGER PRIMARY KEY AUTOINCREMENT,

    Title TEXT NOT NULL,

    Author TEXT NOT NULL,

    Publisher TEXT NOT NULL,

    YearPublished INTEGER CHECK(YearPublished > 0),

    ISBN TEXT UNIQUE NOT NULL,

    CopiesAvailable INTEGER DEFAULT 1 CHECK(CopiesAvailable >= 0)

);

CREATE TABLE Members (

    MemberID INTEGER PRIMARY KEY AUTOINCREMENT,

    FirstName TEXT NOT NULL,

    LastName TEXT NOT NULL,

    Email TEXT UNIQUE NOT NULL,

    Phone TEXT NOT NULL CHECK(length(Phone) BETWEEN 10 AND 15),

    MembershipDate DATE DEFAULT CURRENT\_DATE,

    Role TEXT CHECK(Role IN ('Standard', 'Premium')) DEFAULT 'Standard',

    IsActive BOOLEAN DEFAULT TRUE

);

CREATE TABLE Staff (

    StaffID INTEGER PRIMARY KEY AUTOINCREMENT,

    FirstName TEXT NOT NULL,

    LastName TEXT NOT NULL,

    Email TEXT UNIQUE NOT NULL,

    Position TEXT NOT NULL,

    Role TEXT CHECK(Role IN ('Librarian', 'Assistant', 'Admin')) DEFAULT 'Assistant',

    HireDate DATE DEFAULT CURRENT\_DATE,

    IsActive BOOLEAN DEFAULT TRUE

);

CREATE TABLE Borrow\_Transactions (

    TransactionID INTEGER PRIMARY KEY AUTOINCREMENT,

    MemberID INTEGER NOT NULL,

    BookID INTEGER NOT NULL,

    BorrowDate DATE DEFAULT CURRENT\_DATE,

    DueDate DATE NOT NULL,

    ReturnDate DATE,

    FineAmount REAL DEFAULT 0 CHECK(FineAmount >= 0),

    Status TEXT CHECK(Status IN ('Borrowed', 'Returned', 'Overdue')) DEFAULT 'Borrowed',

    FOREIGN KEY (MemberID) REFERENCES Members(MemberID),

    FOREIGN KEY (BookID) REFERENCES Books(BookID)

);

CREATE TABLE Reservations (

    ReservationID INTEGER PRIMARY KEY AUTOINCREMENT,

    MemberID INTEGER NOT NULL,

    BookID INTEGER NOT NULL,

    ReservationDate DATE DEFAULT CURRENT\_DATE,

    Status TEXT CHECK(Status IN ('Pending', 'Fulfilled', 'Cancelled')) DEFAULT 'Pending',

    FOREIGN KEY (MemberID) REFERENCES Members(MemberID),

    FOREIGN KEY (BookID) REFERENCES Books(BookID)

);

CREATE TABLE Categories (

    CategoryID INTEGER PRIMARY KEY AUTOINCREMENT,

    Name TEXT UNIQUE NOT NULL

);

CREATE TABLE Book\_Categories (

    BookID INTEGER NOT NULL,

    CategoryID INTEGER NOT NULL,

    PRIMARY KEY (BookID, CategoryID),

    FOREIGN KEY (BookID) REFERENCES Books(BookID),

    FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)

);

CREATE TABLE Audit\_Logs (

    LogID INTEGER PRIMARY KEY AUTOINCREMENT,

    Action TEXT NOT NULL,

    PerformedBy TEXT NOT NULL,

    ActionDate DATETIME DEFAULT CURRENT\_TIMESTAMP,

    Details TEXT

);

-- Insert Sample Data

INSERT INTO Books (Title, Author, Publisher, YearPublished, ISBN, CopiesAvailable)

VALUES

('The Great Gatsby', 'F. Scott Fitzgerald', 'Scribner', 1925, '9780743273565', 5),

('1984', 'George Orwell', 'Secker & Warburg', 1949, '9780451524935', 10);

INSERT INTO Members (FirstName, LastName, Email, Phone, Role)

VALUES

('Alice', 'Johnson', 'alice.johnson@example.com', '1234567890', 'Standard'),

('Bob', 'Smith', 'bob.smith@example.com', '9876543210', 'Premium');

INSERT INTO Staff (FirstName, LastName, Email, Position, Role)

VALUES

('Clara', 'Jones', 'clara.jones@example.com', 'Librarian', 'Admin'),

('Daniel', 'Brown', 'daniel.brown@example.com', 'Assistant', 'Assistant');

INSERT INTO Borrow\_Transactions (MemberID, BookID, DueDate, Status)

VALUES

(1, 1, '2024-12-01', 'Borrowed'),

(2, 2, '2024-12-05', 'Borrowed');

INSERT INTO Categories (Name)

VALUES

('Fiction'),

('Science Fiction'),

('Mystery'),

('Biography');

INSERT INTO Book\_Categories (BookID, CategoryID)

VALUES

(1, 1),

(2, 2);

INSERT INTO Reservations (MemberID, BookID, Status)

VALUES

(1, 2, 'Pending');

INSERT INTO Audit\_Logs (Action, PerformedBy, Details)

VALUES

('Borrow', 'Alice Johnson', 'Borrowed Book: 1'),

('Return', 'Bob Smith', 'Returned Book: 2');

-- Enhanced Queries

-- View Borrowed Books with Fines

SELECT bt.TransactionID, m.FirstName || ' ' || m.LastName AS MemberName,

       b.Title AS BookTitle, bt.BorrowDate, bt.DueDate, bt.Status, bt.FineAmount

FROM Borrow\_Transactions bt

JOIN Members m ON bt.MemberID = m.MemberID

JOIN Books b ON bt.BookID = b.BookID

WHERE bt.Status IN ('Borrowed', 'Overdue');

-- View Reservation Status

SELECT r.ReservationID, m.FirstName || ' ' || m.LastName AS MemberName,

       b.Title AS BookTitle, r.ReservationDate, r.Status

FROM Reservations r

JOIN Members m ON r.MemberID = m.MemberID

JOIN Books b ON r.BookID = b.BookID;

-- View Active Members

SELECT MemberID, FirstName || ' ' || LastName AS FullName, Email, Role

FROM Members

WHERE IsActive = TRUE;

-- View All Audit Logs

SELECT \* FROM Audit\_Logs;