**Library Management System - SQL Queries**

-- Create Books table

CREATE TABLE Books (

Book\_ID INT PRIMARY KEY AUTO\_INCREMENT,

Title VARCHAR(255) NOT NULL,

Author VARCHAR(255) NOT NULL,

Genre VARCHAR(100) NOT NULL,

ISBN VARCHAR(20) UNIQUE NOT NULL,

Availability BOOLEAN DEFAULT TRUE,

Created\_At TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Create Users table

CREATE TABLE Users (

User\_ID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(255) NOT NULL,

Email VARCHAR(255) UNIQUE NOT NULL,

Contact VARCHAR(15),

Registered\_At TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Create Transactions table

CREATE TABLE Transactions (

Transaction\_ID INT PRIMARY KEY AUTO\_INCREMENT,

User\_ID INT NOT NULL,

Book\_ID INT NOT NULL,

Issue\_Date DATE NOT NULL,

Due\_Date DATE NOT NULL,

Return\_Date DATE DEFAULT NULL,

Status ENUM('Issued', 'Returned', 'Overdue') DEFAULT 'Issued',

Created\_At TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (User\_ID) REFERENCES Users(User\_ID) ON DELETE CASCADE,

FOREIGN KEY (Book\_ID) REFERENCES Books(Book\_ID) ON DELETE CASCADE

);

-- Insert Sample Data

INSERT INTO Books (Title, Author, Genre, ISBN, Availability) VALUES

('The Great Gatsby', 'F. Scott Fitzgerald', 'Classic', '9780743273565', TRUE),

('To Kill a Mockingbird', 'Harper Lee', 'Fiction', '9780061120084', TRUE),

('1984', 'George Orwell', 'Dystopian', '9780451524935', TRUE);

INSERT INTO Users (Name, Email, Contact) VALUES

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

('Bob Smith', 'bob@example.com', '0987654321');

-- Fetch all available books

SELECT Book\_ID, Title, Author, Genre, ISBN FROM Books WHERE Availability = TRUE ORDER BY Title;

-- Fetch books by a specific author

SELECT Book\_ID, Title, Genre FROM Books WHERE Author = 'George Orwell';

-- Fetch books by genre

SELECT Book\_ID, Title, Author FROM Books WHERE Genre = 'Fiction';

-- Check overdue books

SELECT T.Transaction\_ID, U.Name AS Borrower, B.Title AS BookTitle, T.Due\_Date

FROM Transactions T

JOIN Users U ON T.User\_ID = U.User\_ID

JOIN Books B ON T.Book\_ID = B.Book\_ID

WHERE T.Return\_Date IS NULL AND T.Due\_Date < CURDATE() AND T.Status = 'Issued';

-- Fetch user borrowing history

SELECT U.Name, B.Title, T.Issue\_Date, T.Return\_Date, T.Status

FROM Transactions T

JOIN Users U ON T.User\_ID = U.User\_ID

JOIN Books B ON T.Book\_ID = B.Book\_ID

WHERE U.User\_ID = 1

ORDER BY T.Issue\_Date DESC;

-- Update book availability on return

UPDATE Books

SET Availability = TRUE

WHERE Book\_ID = (SELECT Book\_ID FROM Transactions WHERE Transaction\_ID = 1);

-- Mark book as returned

UPDATE Transactions

SET Return\_Date = CURDATE(), Status = 'Returned'

WHERE Transaction\_ID = 1;

-- Delete a user and cascade delete their transactions

DELETE FROM Users WHERE User\_ID = 2;