# Library Management System - SQL Queries

```
-- Professional SQL Queries for Library Management System

-- 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;
```