# -- VIEW: BorrowedBooks

CREATE VIEW BorrowedBooks AS

SELECT

b.Title,

b.Author,

br.FirstName,

br.LastName,

l.LoanDate,

ls.StatusName

FROM Loans l

JOIN Books b ON l.BookID = b.BookID

JOIN Borrower br ON l.BorrowerID = br.BorrowerID

JOIN LoanStatus ls ON l.LoanStatusID = ls.LoanStatusID

WHERE ls.StatusName = 'borrowed';

# -- QUERY: Overdue Books (Assuming 30-day borrow period)

SELECT

b.Title, br.FirstName, br.LastName, l.LoanDate

FROM Loans l

JOIN Books b ON l.BookID = b.BookID

JOIN Borrower br ON l.BorrowerID = br.BorrowerID

WHERE l.LoanStatusID = 1

AND l.LoanDate < NOW() - INTERVAL 30 DAY;

-- STORED PROCEDURE: Lend a Book

DELIMITER //

CREATE PROCEDURE sp\_LendBook (

IN in\_BookID INT,

IN in\_BorrowerID INT

)

BEGIN

INSERT INTO Loans (BorrowerID, BookID, LoanDate, LoanStatusID)

VALUES (in\_BorrowerID, in\_BookID, NOW(), 1);

UPDATE Books

SET Quantity = Quantity - 1,

InStock = IF(Quantity - 1 > 0, 1, 0)

WHERE BookID = in\_BookID;

END;//

DELIMITER ;

# -- STORED PROCEDURE: Return a Book

DELIMITER //

CREATE PROCEDURE sp\_ReturnBook (

IN in\_LoanID INT

)

BEGIN

UPDATE Loans

SET ReturnDate = NOW(), LoanStatusID = 2

WHERE LoanID = in\_LoanID;

UPDATE Books

SET Quantity = Quantity + 1,

InStock = 1

WHERE BookID = (SELECT BookID FROM Loans WHERE LoanID = in\_LoanID);

END;//

DELIMITER ;

-- FUNCTION: Count of Books Borrowed by a User

DELIMITER //

CREATE FUNCTION fn\_BooksBorrowedByUser(in\_BorrowerID INT)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE bookCount INT;

SELECT COUNT(\*) INTO bookCount

FROM Loans

WHERE BorrowerID = in\_BorrowerID AND LoanStatusID = 1;

RETURN bookCount;

END;//

DELIMITER ;