-- Create Authors Table

CREATE TABLE Authors (

author\_id INT AUTO\_INCREMENT PRIMARY KEY,

author\_name VARCHAR(255)

);

-- Create Books Table

CREATE TABLE Books (

book\_id INT AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255),

author\_id INT,

genre VARCHAR(255),

publication\_date DATE,

isbn VARCHAR(13),

quantity\_available INT,

FOREIGN KEY (author\_id) REFERENCES Authors(author\_id)

);

-- Create Members Table

CREATE TABLE Members (

member\_id INT AUTO\_INCREMENT PRIMARY KEY,

member\_name VARCHAR(255),

member\_email VARCHAR(255),

member\_phone VARCHAR(15)

);

-- Create Borrowings Table

CREATE TABLE Borrowings (

borrowing\_id INT AUTO\_INCREMENT PRIMARY KEY,

book\_id INT,

member\_id INT,

borrowing\_date DATE,

return\_date DATE,

is\_returned BOOLEAN,

FOREIGN KEY (book\_id) REFERENCES Books(book\_id),

FOREIGN KEY (member\_id) REFERENCES Members(member\_id)

);

-- Create Publishers Table

CREATE TABLE Publishers (

publisher\_id INT AUTO\_INCREMENT PRIMARY KEY,

publisher\_name VARCHAR(255),

publisher\_country VARCHAR(255)

);

-- Create Book Copies Table

CREATE TABLE BookCopies (

copy\_id INT AUTO\_INCREMENT PRIMARY KEY,

book\_id INT,

copy\_number VARCHAR(10),

`condition` VARCHAR(50), -- Enclosing "condition" in backticks

shelf\_location VARCHAR(50),

FOREIGN KEY (book\_id) REFERENCES Books(book\_id)

);

-- Create Authors-Books Mapping Table

CREATE TABLE AuthorsBooks (

author\_book\_id INT AUTO\_INCREMENT PRIMARY KEY,

author\_id INT,

book\_id INT,

FOREIGN KEY (author\_id) REFERENCES Authors(author\_id),

FOREIGN KEY (book\_id) REFERENCES Books(book\_id)

);

-- Create Reviews Table

CREATE TABLE Reviews (

review\_id INT AUTO\_INCREMENT PRIMARY KEY,

book\_id INT,

member\_id INT,

rating FLOAT,

review\_text TEXT,

review\_date DATE,

FOREIGN KEY (book\_id) REFERENCES Books(book\_id),

FOREIGN KEY (member\_id) REFERENCES Members(member\_id)

);

-- Create Transactions Table

CREATE TABLE Transactions (

transaction\_id INT AUTO\_INCREMENT PRIMARY KEY,

member\_id INT,

transaction\_date DATE,

transaction\_type VARCHAR(50),

amount\_paid DECIMAL(10, 2),

FOREIGN KEY (member\_id) REFERENCES Members(member\_id)

);

-- Inserting data into the Books table

INSERT INTO Books (book\_id,title, author\_id, genre, publication\_date, isbn, quantity\_available)

VALUES

(1,'Harry Potter', 1, 'Fantasy', '1997-06-26', '9788700631625', 5),

(2,'To Kill a Mockingbird', 2, 'Fiction', '1960-07-11', '9780061120084', 3),

(3,'The Great Gatsby', 3, 'Classic', '1925-04-10', '9780743273565', 2);

-- Inserting data into the Authors table

INSERT INTO Authors (author\_id, author\_name)

VALUES

(1, 'J.K. Rowling'),

(2, 'Harper Lee'),

(3, 'F. Scott Fitzgerald');

-- Inserting data into the Members table

INSERT INTO Members (member\_id, member\_name, member\_email, member\_phone)

VALUES

(1,'John Doe', 'john.doe@example.com', '123-456-7890'),

(2, 'Jane Smith', 'jane.smith@example.com', '987-654-3210');

-- Inserting data into the Borrowings table

INSERT INTO Borrowings (book\_id, member\_id, borrowing\_id, borrowing\_date, return\_date, is\_returned)

VALUES

(1, 1, 1, '2024-02-10', '2024-02-20', TRUE),

(2, 2, 2, '2024-02-15', NULL, FALSE);

-- Inserting data into the Publishers table

INSERT INTO Publishers (publisher\_name, publisher\_country)

VALUES

('Penguin Random House', 'United States'),

('HarperCollins', 'United Kingdom');

-- Inserting data into the BookCopies table

INSERT INTO BookCopies (book\_id, copy\_number, `condition`, shelf\_location)

VALUES

(1, '001', 'Good', 'A1'),

(1, '002', 'Fair', 'B3');

-- Inserting data into the AuthorsBooks table

INSERT INTO AuthorsBooks (author\_id, book\_id)

VALUES

(1, 1),

(2, 2);

-- Inserting data into the Reviews table

INSERT INTO Reviews (book\_id, member\_id, rating, review\_text, review\_date)

VALUES

(1, 1, 4.5, 'A classic masterpiece.', '2024-02-12'),

(2, 2, 5.0, 'Absolutely loved it!', '2024-02-18');

-- Inserting data into the Transactions table

INSERT INTO Transactions (member\_id, transaction\_date, transaction\_type, amount\_paid)

VALUES

(1, '2024-02-10', 'Borrow', 0),

(2, '2024-02-15', 'Borrow', 0);

-- Question1-Ans

SELECT Books.title

FROM Books

JOIN Borrowings ON Books.book\_id = Borrowings.book\_id

WHERE Borrowings.member\_id = 1; -- Replace 1 with the actual member\_id

-- Question-2 answer

SELECT genre, COUNT(\*) AS count

FROM Books

GROUP BY genre

ORDER BY count DESC;

-- Question-3 answer

SELECT Books.title, AVG(Reviews.rating) AS average\_rating

FROM Books

LEFT JOIN Reviews ON Books.book\_id = Reviews.book\_id

GROUP BY Books.book\_id

ORDER BY average\_rating DESC;

-- Question4-answer

SELECT Members.member\_name

FROM Members

JOIN Borrowings ON Members.member\_id = Borrowings.member\_id

GROUP BY Members.member\_id

HAVING COUNT(\*) > 5;

-- Question-5 answer

SELECT Members.member\_name

FROM Members

JOIN Borrowings ON Members.member\_id = Borrowings.member\_id

GROUP BY Members.member\_id

HAVING COUNT(\*) < 5;

-- Question-6- answer

SELECT Books.title, AVG(Reviews.rating) AS average\_rating, COUNT(\*) AS review\_count

FROM Books

LEFT JOIN Reviews ON Books.book\_id = Reviews.book\_id

GROUP BY Books.book\_id

HAVING review\_count >= 5

ORDER BY average\_rating DESC;

-- Question-7 -answer

SELECT SUM(amount\_paid) AS total\_revenue

FROM Transactions

WHERE transaction\_type = 'Purchase';

-- Question-8 -answer

SET SQL\_SAFE\_UPDATES = 0;

UPDATE Books

SET publisher\_id = (

SELECT publisher\_id

FROM Publishers

WHERE publisher\_name = 'Penguin Random House'

)

WHERE title = 'Harry Potter'; -- Replace 'Harry Potter' with the appropriate book title

DESCRIBE Books;

SELECT Books.title, Authors.author\_name, Publishers.publisher\_name

FROM Books

JOIN Authors ON Books.author\_id = Authors.author\_id

JOIN Publishers ON Books.publisher\_id = Publishers.publisher\_id;

-- Question9 -answer

SELECT title

FROM Books

WHERE quantity\_available > 0;

-- Question 10 -answer

SELECT member\_id

FROM Borrowings

WHERE return\_date < CURRENT\_DATE AND is\_returned = FALSE;

-- Question 11-answer

SELECT Books.title, COUNT(\*) AS borrow\_count

FROM Books

JOIN Borrowings ON Books.book\_id = Borrowings.book\_id

GROUP BY Books.book\_id

ORDER BY borrow\_count DESC

LIMIT 10;

-- Question 12 -answer

SELECT AVG(DATEDIFF(return\_date, borrowing\_date)) AS average\_borrow\_duration

FROM Borrowings

WHERE is\_returned = TRUE;

-- Question 13 -answer

SELECT YEAR(publication\_date) AS publication\_year, COUNT(\*) AS book\_count

FROM Books

GROUP BY YEAR(publication\_date);

-- Question14 -answer

SELECT member\_id, COUNT(\*) AS borrow\_count

FROM Borrowings

GROUP BY member\_id

HAVING borrow\_count > 1;

-- Question 15 -answer

SELECT Books.title, Authors.author\_name, AVG(Reviews.rating) AS average\_rating

FROM Books

JOIN Authors ON Books.author\_id = Authors.author\_id

LEFT JOIN Reviews ON Books.book\_id = Reviews.book\_id

GROUP BY Books.book\_id;

-- Question-16 -answer

SELECT Books.title, COUNT(\*) AS copy\_count

FROM Books

JOIN BookCopies ON Books.book\_id = BookCopies.book\_id

GROUP BY Books.book\_id;

-- Question-17 -answer

CREATE VIEW TransactionView AS

SELECT member\_id, transaction\_date

FROM Transactions;

-- Grant SELECT privilege to another user

GRANT SELECT ON TransactionView TO puja;