SQL Project - Books, Customers & Orders Database

```
CREATE TABLE books(
    Book_ID SERIAL PRIMARY KEY,
    Title VARCHAR(100),
    Author VARCHAR(100),
    Genre VARCHAR(50),
    Published_Year INT,
    Price NUMERIC(10,2),
    Stock INT
);

CREATE TABLE Customers(
    Customer_ID SERIAL PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(100),
    Phone VARCHAR(15),
    City VARCHAR(50),
    Country VARCHAR(150)
);

CREATE TABLE Orders(
    Order_ID SERIAL PRIMARY KEY,
    Customer_ID INT REFERENCES Customers(Customer_ID),
    Book_ID INT REFERENCES Books(Book_ID),
    Order_Date DATE,
    Quantity INT,
    Total_Amount NUMERIC(10, 2)
);

SELECT * FROM Books;
SELECT * FROM Customers;
SELECT * FROM Orders;

-- IMPORT DATA INTO BOOKS
COPY Books(Book_ID, Title, Author, Genre, Published_Year, Price, Stock)
FROM 'C:/Users/nandi/OneDrive/Desktop/Books.csv'
CSV HEADER;

-- 1 Retrieve all books in Fiction genre
SELECT * FROM Books WHERE genre = 'Fiction';

-- 2 Find books published after year 1950
SELECT * FROM Books WHERE published_year > 1950;

-- 3 List all customers from Canada
SELECT * FROM Customers WHERE Country = 'Canada';

-- 4 Show orders placed in November 2023
SELECT * FROM Orders WHERE order_date BETWEEN '2023-11-01' AND '2023-11-30';

-- 5 Retrieve the total stock of books available
```

```
SELECT SUM(stock) AS total_books FROM Books;

-- 6 Find the details of the most expensive book
SELECT * FROM Books ORDER BY price DESC LIMIT 1;

-- 7 Show all customers who ordered more than 1 qty of book
SELECT * FROM Orders WHERE quantity > 1;

-- 8 Retrieve all orders where total amount exceed $20
SELECT * FROM Orders WHERE total_amount > 20;

-- 9 List all genres available in the books table
SELECT DISTINCT genre FROM Books;

-- 10 Find the book with lowest stock
SELECT * FROM Books ORDER BY stock ASC LIMIT 5;

-- 11 Calculate the total revenue generated from all orders
SELECT SUM(total_amount) AS revenue_allorders FROM Orders;

-- Advanced Queries
-- 1 Retrieve the total no of books sold for each genre
SELECT b.genre, SUM(o.quantity) AS total_books_sold
FROM Orders o
JOIN Books b ON o.book_id = b.book_id
GROUP BY b.Genre;

-- 2 Find the avg price of books in the 'Fantasy' genre
SELECT AVG(price) AS avg_price FROM Books WHERE Genre='Fantasy';

-- 3 List customers who have placed at least 2 orders
SELECT customer_id, COUNT(order_id) AS order_count
FROM Orders GROUP BY customer_id HAVING COUNT(order_id) >= 2;

-- With customer name
SELECT o.customer_id, c.name, COUNT(o.order_id) AS order_count
FROM Orders o JOIN Customers c ON o.customer_id=c.customer_id
GROUP BY o.customer_id, c.name HAVING COUNT(order_id) >=2;

-- 4 Find most frequently ordered book
SELECT o.book_id, b.title, COUNT(o.order_id) AS order_count
FROM Orders o JOIN Books b ON o.book_id=b.book_id
GROUP BY o.book_id, b.title ORDER BY order_count DESC LIMIT 1;

-- 5 Show top 3 most expensive books by Fantasy genre
SELECT * FROM Books WHERE genre='Fantasy' ORDER BY price DESC LIMIT 3;

-- 6 Retrieve total quantity of books sold by each author
SELECT b.author, SUM(o.quantity) AS total_qty
FROM Orders o JOIN Books b ON o.book_id=b.book_id
GROUP BY b.author;

-- 7 List the cities where customers who spent over $30 are located
```

```
SELECT DISTINCT c.city
FROM Customers c JOIN Orders o ON c.customer_id=o.customer_id
WHERE o.total_amount > 30;

-- 8 Find the customer who spent the most on orders
SELECT c.customer_id, c.name, SUM(o.total_amount) AS total_spent
FROM Orders o JOIN Customers c ON o.customer_id=c.customer_id
GROUP BY c.customer_id, c.name ORDER BY total_spent DESC LIMIT 1;

-- 9 Calculate stock remaining after fulfilling all orders
SELECT b.book_id, b.title, b.stock,
       COALESCE(SUM(o.quantity),0) AS order_quantity,
       b.stock - COALESCE(SUM(o.quantity),0) AS remaining_qty
FROM Books b LEFT JOIN Orders o ON b.book_id=o.book_id
GROUP BY b.book_id ORDER BY b.book_id;
```