

```

-- Create Database
CREATE DATABASE OnlineBookstore;

-- Switch to the database
\c OnlineBookstore;

-- Create Tables
DROP TABLE IF EXISTS Books;
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
);
DROP TABLE IF EXISTS customers;
CREATE TABLE Customers (
    Customer_ID SERIAL PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(100),
    Phone VARCHAR(15),
    City VARCHAR(50),
    Country VARCHAR(150)
);
DROP TABLE IF EXISTS orders;
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 Table
COPY Books(Title, Author, Genre, Published_Year, Price, Stock)
FROM 'D:\Course Updates\30 Day Series\SQL\CSV\Books.csv'
CSV HEADER;

-- Import Data into Customers Table
COPY Customers(Name, Email, Phone, City, Country)
FROM 'D:\Course Updates\30 Day Series\SQL\CSV\Customers.csv'
CSV HEADER;

-- Import Data into Orders Table
COPY Orders(Customer_ID, Book_ID, Order_Date, Quantity, Total_Amount)
FROM 'D:\Course Updates\30 Day Series\SQL\CSV\Orders.csv'
CSV HEADER;

```

-- Basic Queries

-- 1) Retrieve all books in the "Fiction" genre:

```
SELECT * FROM Books WHERE Genre='Fiction';
```

-- 2) Find books published after the 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_Stock From Books;
```

-- Advanced Queries

-- 1) Retrieve the total number 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 average price of books in the "Fantasy" genre:

```
SELECT AVG(price) AS Average_Price FROM Books WHERE Genre = 'Fantasy';
```

-- 3) List customers who have placed at least 2 orders:

```
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 the 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 the top 3 most expensive books of 'Fantasy' Genre:

```
SELECT * FROM books WHERE genre ='Fantasy' ORDER BY price DESC LIMIT 3;
```

-- 6) Retrieve the total quantity of books sold by each author:

```
SELECT b.author, SUM(o.quantity) AS Total_Books_Sold
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, total_amount FROM orders o JOIN customers c ON o.customer_id=c.customer_id
```

-- 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 the 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_Quantity
FROM books b LEFT JOIN orders o ON b.book_id=o.book_id GROUP BY b.book_id ORDER BY b.book_id;
```

