```
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(Book_ID, 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(Customer_ID, 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(Order_ID, Customer_ID, Book_ID, Order_Date, Quantity, Total_Amount)
FROM 'D:\Course Updates\30 Day Series\SQL\CSV\Orders.csv'
CSV HEADER;

select * from Books;
select * from customers;
select * from orders;

-- 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'
order by Published_Year desc;

-- 3) List all customers from the 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_Bookstocks
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 quantity of a book:
select * from orders
where quantity>1;

-- 8) Retrieve all orders where the total amount exceeds $20:
select * from orders
```

```
where total_amount>20
order by total_amount asc;

-- 9) List all genres available in the Books table:
select distinct genre from books;

-- 10) Find the book with the lowest stock:
select * from Books
order by stock asc limit 1;

-- 11) Calculate the total revenue generated from all orders:
select sum(total_amount) as Total_Revenue from Orders;

-- Advance Questions :

-- 1) Retrieve the total number of books sold for each genre:
SELECT * FROM Orders;

select b.Genre, sum(o.Quantity)
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 Avg_Price from Books
where Genre='Fantasy';

-- 3) List customers who have placed at least 2 orders:
SELECT * FROM Customers;
SELECT * FROM Orders;
```

```
select customer_id, count(order_id) as order_count
from Orders
group by customer_id
having count(order_id)>2;

--WITH JOIN
select o.customer_id,c.name, count(o.order_id) as order_count
from Orders o
join Customers c
on o.customer_id=o.customer_id
group by o.customer_id, c.name
having count(order_id)>2;

-- 4) Find the most frequently ordered book:
SELECT * FROM Orders;

select book_id, count(order_id) as order_count
from Orders
group by book_id
order by order_count desc limit 1;

--WITH JOIN
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;

-- 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_book_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 * FROM Customers;
SELECT * FROM Orders;

select distinct c.city,o.total_amount
from orders o
join customers c
on o.customer_id=c.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 customers c
join orders o
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, coalesce(sum(quantity),0) as Quantity_ordered,b.stock
from Books b
left join Orders o
on b.book_id=o.book_id
group by b.book_id;
```