```
-- Create Database
CREATE DATABASE OnlineBookstore;

-- Switch to the database



-- 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;

-- 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 the Canada:

select * from customers where Country="Canada";
-- 4) Show orders placed in November 2023:

select * from orders where year(order_date) ='2023' and month(order_date)='11';
-- or
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;
-- 6) Find the details of the most expensive book:

select * from books  where price=(select max(price) from books);
--  or
```

```
select * from books order by price DESC limit 1;

-- 7) Show all customers who ordered more than 1 quantity of a book:

select c.Name ,o.Quantity from Customers c inner join orders o
on c.Customer_ID=o.Customer_ID where Quantity>1 ;

-- 8) Retrieve all orders where the total amount exceeds $20:

select * from orders where Total_Amount>20;
-- 9) List all genres available in the Books table:

select Genre from Books group by genre;
-- or
select DISTINCT Genre from Books;

-- 10) Find the book with the lowest stock:

select Title,stock from Books where stock=(select min(stock) from Books);
-- or
select * from Books order by stock 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 b.genre , sum(o.quantity) from books b join orders o on b.book_id=o.Book_id group by
b.Genre;
-- 2) Find the average price of books in the "Fantasy" genre:
```

```
select avg(Price)  Fantasy_avg from books where Genre="Fantasy";


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


select c.Customer_id, C.Name ,count(o.Customer_id) order_placed
from Customers c inner join orders o
on c.customer_id= o.Customer_id
 group by c.Customer_id  having count(c.Customer_id)>=2;
select * from orders;
-- 4) Find the most frequently ordered book:
select
b.book_id, b.title,count(o.book_id) order_count
from
 books b join orders o on b.book_id=o.book_id
 group by b.book_id
 order by order_count desc limit 1  ;


-- 5) Show the top 3 most expensive books of 'Fantasy' Genre :


select book_id,Title , price 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)  total_book_sold
from books b join orders o on b.book_id = o.book_id
group by Author ;


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


select distinct c.City ,o.Total_amount
```

```
from customers c join orders o on c.Customer_ID=o.Customer_ID
WHERE total_amount >30;

-- 8) Find the customer who spent the most on orders:

select  o.customer_id ,c.Name , sum(o.Total_Amount) as total_amount
from orders o join customers c on o.customer_id=c.Customer_ID
group by  o.customer_id ,c.Name
order by Total_Amount Desc
limit 1;
-- 9) Calculate the stock remaining after fulfilling all orders:

select b.book_id,b.stock, coalesce(sum(quantity),0) as order_quantity,
b.stock-coalesce(sum(quantity),0) as reamining_Orders
from books b left join orders o
 on b.book_id=o.book_id
 group by b.book_id
 order by b.book_id;
```