# BOOKSTORE ANALYSIS ON SQL

```
-- 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;

-- When importing the data using CSV file, If '\' doesnt work then use '/'

-- Import Data into Books Table
COPY Books(Book_ID, Title, Author, Genre, Published_Year, Price, Stock)
FROM 'D:\PROGRAMMING\DATA SCIENCE\DATA ANALYTICS\SQL\Project\Bookstore\Books.csv'
CSV HEADER;

-- Import Data into Customers Table
COPY Customers(Customer_ID, Name, Email, Phone, City, Country)
FROM 'D:\PROGRAMMING\DATA SCIENCE\DATA ANALYTICS\SQL\Project\Bookstore\Customers.csv'
CSV HEADER;

-- Import Data into Orders Table
COPY Orders(Order_ID, Customer_ID, Book_ID, Order_Date, Quantity, Total_Amount)
FROM 'D:\PROGRAMMING\DATA SCIENCE\DATA ANALYTICS\SQL\Project\Bookstore\Orders.csv'
CSV HEADER;


-- 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 order_date BETWEEN '2023-11-01' AND '2023-11-30';
```

**-- 5) Retrieve the total stock of books available:**

```
SELECT SUM(stock) AS Total_Stocks
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;
```

**-- 9) List all genres available in the Books table:**

```
SELECT DISTINCT(Genre) AS Genres
FROM Books;
```

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

```
SELECT *
FROM Books
ORDER BY stock
LIMIT 1;
```

**-- 11) Calculate the total revenue generated from all orders:**

```
SELECT SUM(total_amount) AS revenue
FROM Orders;
```

## -- Advance Questions :

**-- 1) Retrieve the total number of books sold for each genre:**

```
SELECT b.Genre, SUM(o.Quantity) AS total_books_sold
FROM Books b
JOIN Orders o
ON o.Book_ID = b.Book_ID
GROUP BY b.Genre;
```

**-- 2) Find the average price of books in the "Fantasy" genre:**

```
SELECT AVG(b.price) AS average_book_price
FROM Books b
```

```
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 c.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 o.book_id, b.title, COUNT(o.order_id) as order_count
FROM Orders o
JOIN Books b
ON b.book_id = o.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 Books b
JOIN Orders o
ON o.book_id = b.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 o.total_amount > 30

ORDER BY o.total_amount;
```

**-- 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 c.customer_id = o.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

FROM Orders o

RIGHT JOIN Books b

ON b.book_id = o.book_id

GROUP BY b.book_id

ORDER BY b.book_id;
```

*--created by vivek sahu*

*-- get the project on-* *"https://github.com/VIVEKSAHU-06/Data-Analysis/tree/main/SQL/Bookstore%20Sales"*