SQL project of online book selling By Bodhisatta Banerjee on Postgre SQL

```
-- 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:\Postgre SQL\All Excel Practice Files\Books.csv'
CSV HEADER;

-- Import Data into customers Table
COPY Customers(Customer_id, Name, Email, Phone, City, Country)
FROM 'D:\Postgre SQL\All Excel Practice Files\Customers.csv'
CSV HEADER;

-- Import Data into Orders Table
COPY Orders(Order_id,Customer_id, Book_id, Order_date, Quantity, Total_amount)
FROM 'D:\Postgre SQL\All Excel Practice Files\Orders.csv'
CSV HEADER;


--Project Questions.

--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. Find books published after the year 1950:
SELECT * FROM customers
WHERE country = 'Canada';

--4. Show orders placed in November 2023:
SELECT * FROM orders
WHERE order_date BETWEEN '2023-11-1' 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
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. ist 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;



--12. Retrieve the total number of books sold for each genre:
SELECT * FROM orders;

SELECT b. Genre, SUM (o. Quantity) AS Total_sale FROM orders o
JOIN books b
ON o.book_id=b.book_id
GROUP BY Genre;

--13. Find the average price of books in the "Fantasy" genre:
SELECT * FROM books;

SELECT AVG (price) AS avg_price FROM books
WHERE genre = 'Fantasy';


--14. List customers who have placed at least 2 orders:
SELECT * FROM customers;
SELECT * FROM orders;

SELECT o. customer_id, c. name, COUNT (o. order_id) AS orders_per_customers FROM orders o
JOIN customers c
ON o.customer_id=c.customer_id
GROUP BY o.customer_id, c.name
HAVING COUNT(o.order_id)>=2;
```

--15. Find the most frequently ordered book:

```
SELECT * FROM books;
SELECT b. title, o.book_id, COUNT(o.order_id) AS frequently_ordered_book
FROM orders o
JOIN books b
ON o.book_id=b.book_id
GROUP BY b.title, o.book_id
ORDER BY frequently_ordered_book DESC
LIMIT 1;
```

--16. Show the top 3 most expensive books of 'Fantasy' Genre

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

--17. Retrieve the total quantity of books sold by each author

```
SELECT b. author, SUM(o.quantity) AS total_no_books_sold
FROM books b
JOIN orders o
ON b.book_id=o.book_id
GROUP BY b. author
ORDER BY total_no_books_sold DESC;
```

--18. List the cities where customers who spent over $30 are located:

```
SELECT DISTINCT c. city, country, o. total_amount
FROM orders o
JOIN customers c
ON o.customer_id=c.customer_id
WHERE total_amount>30
ORDER BY total_amount DESC;
```

--19. Find the customer who spent the most on orders:

```
SELECT c. name, c. customer_id, city, o. total_amount
FROM orders o
JOIN customers c
ON c.customer_id=o.customer_id
ORDER BY total_amount DESC
LIMIT 1;
```

--20. 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;
```

Thank you