Bookstore Project in SQL

CASE STUDY:

Q1. Write a SQL query to find the **Name** and **City** of all customers who are from the **country** 'Denmark'

Q2. Write a SQL query to find the total quantity of books each customer has ordered. Your result should display the `customer's Name` and the `Total Quantity` they have ordered. List the customers who have ordered the most books first.

Q3. Write a SQL query to find the total revenue generated from each book **genre** in the year **2023**. The results should show the `Genre` and the `Total Revenue`. Only include genres that generated more than **$500** in total revenue.

Q4. Write a SQL query to find the **second most recent order** for each customer. Your result should display the `Customer_ID`, `Order_ID`, and `Order_Date`. If a customer has only one order, they should not appear in the result.

Q5. Write a SQL query to analyze sales performance by genre. Your query should calculate two things for each book **genre**:

1. The total number of **distinct books** ordered.
2. A sales performance category based on the **total quantity of all books sold** in that genre. ('High': > 150 books, 'Medium': 50-150, 'Low': < 50). Your final output should display the `Genre`, the `Distinct_Book_Count`, and the `Sales_Performance` category.

Q6. Write a SQL query to find the **names of all customers who have never ordered a book from the 'Fantasy' genre**.

Q7. Write a SQL query to identify the **top-spending customer in each country**. If there's a tie in spending, include all customers with the top amount. Your final result should display the `Country`, the top customer's `Name`, and their `Total_Amount` spent.

Q8. Write a SQL query to calculate the number of days between each customer's consecutive orders. Your final result should show the `Customer_ID`, the `Order_ID`, the `Order_Date`, and a new column named `Days_Since_Last_Order.`

Q9. Write a SQL query to find all pairs of books that share the same **Genre** and were published in the same **Published_Year**. Each pair should only be listed once. A book should not be paired with itself.

Q10. Write a SQL query to calculate a running total of each customer's spending over time. Your result should show the customer's `Name`, the `Order_Date` of each order, the

`Total_Amount` for that single order, and a new column named `Running_Total_Spending`.

Q11. Write a single SQL query to create a customer spending summary. The report should show each customer's `Name` followed by their total spending for each year on record (`2022`, `2023`, `2024`), with each year's total in its own separate column. Also, include a `Grand_Total` column. Only include customers whose grand total spending is **more than $500**.

Q12. Write a SQL query to calculate the 3-day rolling average of total daily sales revenue.

Q13. Write a SQL query to find the **first-ever order date** for each customer and display that date next to **every order** that the customer has made.

Q14. Write a SQL query to segment customers into four spending **quartiles** based on their total lifetime spending. Quartile 1 should represent the top 25% of spenders.

Q15. Write a SQL query to count the number of **unique customers** who placed an order in each month of the year **2023**.

Q16. Write a SQL query to calculate the percentage growth in the number of **unique books sold** for each **genre**, month-over-month.

Q17. Write a SQL query to identify the top 5 most common email service providers. Group `'yahoo.com'` and `'hotmail.com'` into a single provider called `'Microsoft Live'`.

Q18. Write a SQL query to find all customers who have purchased **at least one book** that costs more than **$49.00**.

Q19. Write a SQL query to calculate what percentage of the total revenue each book genre contributes.

Q20. Identify "loyal customers" (those who made purchases in at least 3 different months) and then calculate their average number of days between consecutive orders.