Customers Books Data Analysis

This SQL project analyses book sales, customer data, and inventory management using a structured customer database. It involves querying books, customers, and orders to extract insights such as customer purchase trends, stock availability, revenue generation, and more. By utilizing SQL JOINs, Aggregate Functions, and Data Analysis techniques, I was able to create meaningful reports for better decision-making.

```
CREATE DATABASE Customersdata;

USE Customersdata;


SELECT * FROM books;

SELECT * FROM customers;

SELECT * FROM orders;
```

Questions

1) Retrieve all books in the "Fiction" genre.

```
SELECT * FROM books

WHERE genre = 'Fiction';
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Book_ID	Title	Author	Genre	Published_Year	Price	Stock
4	Customizable 24hour product	Christopher Andrews	Fiction	2020	43.52	8
22	Multi-layered optimizing migration	Wesley Escobar	Fiction	1908	39.23	78
28	Expanded analyzing portal	Lisa Coffey	Fiction	1941	37.51	79
29	Quality-focused multi-tasking challenge	Katrina Underwood	Fiction	1905	31.12	100
31	Implemented encompassing conglomeration	Melissa Taylor	Fiction	2010	21.23	44
39	Optimized national process improvement	Megan Goodwin	Fiction	1978	10.99	42
40	Adaptive didactic interface	Natalie Gonzalez	Fiction	1923	25.97	94
47	Reverse-engineered directional conglomeration	John Christian	Fiction	2006	20.37	90

books 1

Output

2) Find books published after the year 1950.

```
SELECT * FROM books

WHERE published_year > 1950;
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Book_ID	Title	Author	Genre	Published_Year	Price	Stock
2	Persevering reciprocal knowledge user	Mario Moore	Fantasy	1971	35.8	19
4	Customizable 24hour product	Christopher Andrews	Fiction	2020	43.52	8
5	Adaptive 5thgeneration encoding	Juan Miller	Fantasy	1956	10.95	16
6	Advanced encompassing implementation	Bryan Morgan	Biography	1985	6.56	2
8	Persistent local encoding	Troy Cox	Science Fiction	2019	48.99	84
9	Optimized interactive challenge	Colin Buckley	Fantasy	1987	14.33	70
10	Ergonomic national hub	Samantha Ruiz	Mystery	2015	24.63	25
11	Secured zero tolerance time-frame	Denise Barnes	Fantasy	1998	35.95	10

books 2

Output

3) List all customers FROM the Canada.

```
SELECT * FROM customers
WHERE Country = 'Canada';
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_ID	Name	Email	Phone	City	Country
38	Nicholas Harris	christine93@perkins.com	1234567928	Davistown	Canada
415	James Ramirez	robert54@hall.com	1234568305	Maxwelltown	Canada
468	David Hart	stokesrebecca@gmail.com	1234568358	Thompsonfurt	Canada

4) Show orders placed in November 2023.

```
SELECT * FROM orders
WHERE order_date BETWEEN '01-11-2023' AND '30-11-2023';
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Order_ID	Customer_ID	Book_ID	Order_Date	Quantity	Total_Amount
1	84	169	26-05-2023	8	188.56
2	137	301	23-01-2023	10	216.6
3	216	261	27-05-2024	6	85.5
4	433	343	25-11-2023	7	301.21
5	14	431	26-07-2023	7	136.36
6	439	119	11-10-2024	5	249.4
7	195	467	23-10-2023	6	82.92
8	32	159	07-05-2024	4	144.84
9	109	407	04-01-2024	9	379.71

orders 4

5) Retrieve the total stock of books available.

```
SELECT SUM(stock) AS Total_Stock
FROM books;
```

Output :-

Result Grid

	Total_Stock
▶	25056

6) Find the details of the most expensive book.

```
SELECT * FROM books
ORDER BY price DESC
LIMIT 1;
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

	Book_ID	Title	Author	Genre	Published_Year	Price	Stock
▶	340	Proactive system-worthy orchestration	Robert Scott	Mystery	1907	49.98	88

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

```
SELECT * FROM orders
WHERE quantity>1;
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Order_ID	Customer_ID	Book_ID	Order_Date	Quantity	Total_Amount
▶	1	84	169	26-05-2023	8	188.56
	2	137	301	23-01-2023	10	216.6
	3	216	261	27-05-2024	6	85.5
	4	433	343	25-11-2023	7	301.21
	5	14	431	26-07-2023	7	136.36
	6	439	119	11-10-2024	5	249.4
	7	195	467	23-10-2023	6	82.92
	8	32	159	07-05-2024	4	144.84
	9	109	407	04-01-2024	9	379.71

orders 7 ×

Output

8) Retrieve all orders WHERE the total amount exceeds $20.

```
SELECT * FROM orders
WHERE total_amount>20
ORDER BY total_amount;
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Order_ID	Customer_ID	Book_ID	Order_Date	Quantity	Total_Amount
307	368	133	17-11-2023	1	20.96
100	207	63	14-07-2023	1	22.38
260	112	170	07-05-2024	2	22.56
319	172	296	27-06-2024	2	22.98
345	325	253	04-07-2023	2	23.32
494	117	127	23-11-2024	2	23.32
288	6	128	13-11-2023	1	24.04
188	46	425	13-03-2023	1	24.23
394	299	425	14-04-2024	1	24.23

orders 8

Output

9) List all genres available in the Books table.

```
SELECT DISTINCT genre FROM books;
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

genre
Biography
Fantasy
Non-Fiction
Fiction
Romance
Science Fiction
Mystery

10) Find the book with the lowest stock.

```
SELECT * FROM books
ORDER BY stock ASC
LIMIT 1;
```

Output :-

Book_ID	Title	Author	Genre	Published_Year	Price	Stock
378	Future-proofed heuristic function	Samantha Mcclain	Romance	1903	6.01	0

11) Calculate the total revenue generated FROM all orders.

```
SELECT SUM(price) AS Total_Revenue
FROM books;
```

Output :-

Total_Revenue
13683.71999999999

Advance QuestiONs -

1) Retrieve the total number of books sold for each genre.

JOIN

```
SELECT b.genre, SUM(o.quantity) AS Total_Book_Sold
FROM orders o
JOIN books b ON o.book_id = b.book_id
GROUP BY b.genre;
```

Output :-

genre	Total_Book_Sold
Biography	285
Non-Fiction	351
Fantasy	446
Romance	439
Science Fiction	447
Mystery	504
Fiction	225

2) Find the average price of books in the "Fantasy" genre.

```
SELECT AVG(price) AS AVG_Price_of_Books
FROM books
```

```
WHERE genre = 'Fantasy';
```

Output :-

Result Grid | Filter Rows:

	AVG_Price_of_Books
▶	25.981690140845064

3) List customers who have placed at least 2 orders.

```
SELECT customer_id, COUNT(order_id) AS Order_COUNT
FROM orders
GROUP BY customer_id
HAVING COUNT (order_id) >=2
ORDER BY customer_id;
```

Output :-

Result Grid | Filter Rows:

	customer_id	Order_Count
▶	2	2
	6	2
	8	2
	13	2
	14	2
	15	2
	16	3
	21	2
	22	3

Result 14 ×

4) Find the most frequently ordered book.

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

Output :-

Result Grid | Filter Rows: | Export:

book_id	title	Book_Count
31	Implemented encompassing conglomeration	4

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

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

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Book_ID	Title	Author	Genre	Published_Year	Price	Stock
240	Stand-alone content-based hub	Lisa Ellis	Fantasy	1957	49.9	41
462	Innovative 3rdgeneration database	Allison Contreras	Fantasy	1988	49.23	62
238	Optimized even-keeled analyzer	Sherri Griffith	Fantasy	1975	48.97	72

6) Retrieve the total quantity of books sold by each author.

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

Output :-

Result Grid | Filter Rows:

author	Total_Book_Qty
Joseph Crane	3
Derrick Howard	5
Juan Miller	8
Jacqueline Young	5
Troy Cox	3
Samantha Ruiz	1
Denise Barnes	5
Jaclyn Miller	9
Christopher Price	1

Result 17

7) List the cities WHERE customers who spent over $30 are located.

```
SELECT DISTINCT c.city, total_amount
```

```
FROM orders o
JOIN customers c ON o.customer_id = c.customer_id
WHERE o.total_amount > 30;
```

Output :-

Result Grid | Filter Rows:

city	total_amount
East Derekberg	298.06
Hamiltonstad	148.02
Kirstenborough	95.85
Kirstenborough	44.61
Lake Benjamin	192.12
West Monicabury	221.8
South Ashleychester	39.51
Lake Robert	48.8
Lake Robert	290.94

Result 18

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 o.customer_id = c.customer_id
GROUP BY c.customer_id, c.name
ORDER BY Total_Spent DESC
LIMIT 1;
```

Output :-

Result Grid | Filter Rows:

customer_id	name	Total_Spent
457	Kim Turner	1398.8999999999999

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_Quantity
FROM books b
LEFT JOIN orders o ON b.book_id = o.book_id
GROUP BY b.book_id, b.title, b.stock;
```

Output :-

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

book_id	title	stock	Order_Quantity	Remaining_Quantity
1	Configurable modular throughput	100	3	97
2	Persevering reciprocal knowledge user	19	0	19
3	Streamlined coherent initiative	27	5	22
4	Customizable 24hour product	8	0	8
5	Adaptive 5thgeneration encoding	16	8	8
6	Advanced encompassing implementation	2	0	2
7	Open-architected exuding structure	95	5	90
8	Persistent local encoding	84	3	81
9	Optimized interactive challenge	70	0	70

Result 20