Customer & Sales Insights for a Retail Company Using SQL

DataBase Setup:- Mentioned DataBase Name, Tables, field & data types

The dataset used in this project contains **500 records** of customers and orders table from a mini retail company. It includes details about

➡️ **Customers** (Customer ID, Name, Region)
➡️ **Orders** (Order ID, Customer ID, Product ID, Sales, Profit, Discount, Region, Order Date)

DataSet HighLights:-

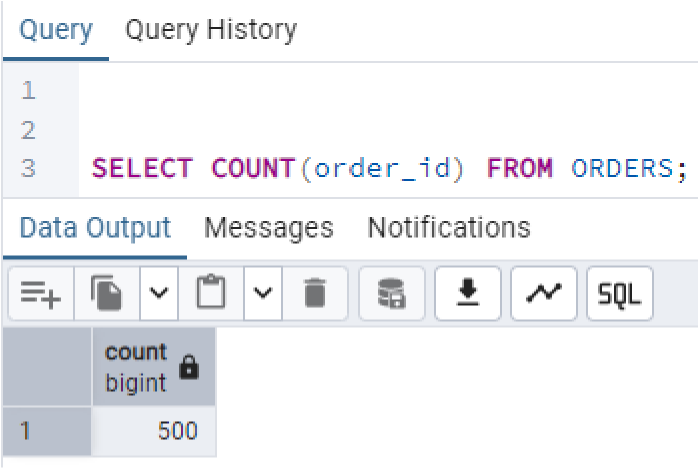
- Orders Range across Multiple Months and shipping modes
- Data is spread across **multiple regions** East, West, South, North

Objective:-

Help the retail company analyze **customer behavior, sales performance, and product trends**

1. Basic Exploration

1. Show the total number of orders



Query Query History

```
SELECT COUNT(order_id) FROM ORDERS;
```

Data Output Messages Notifications

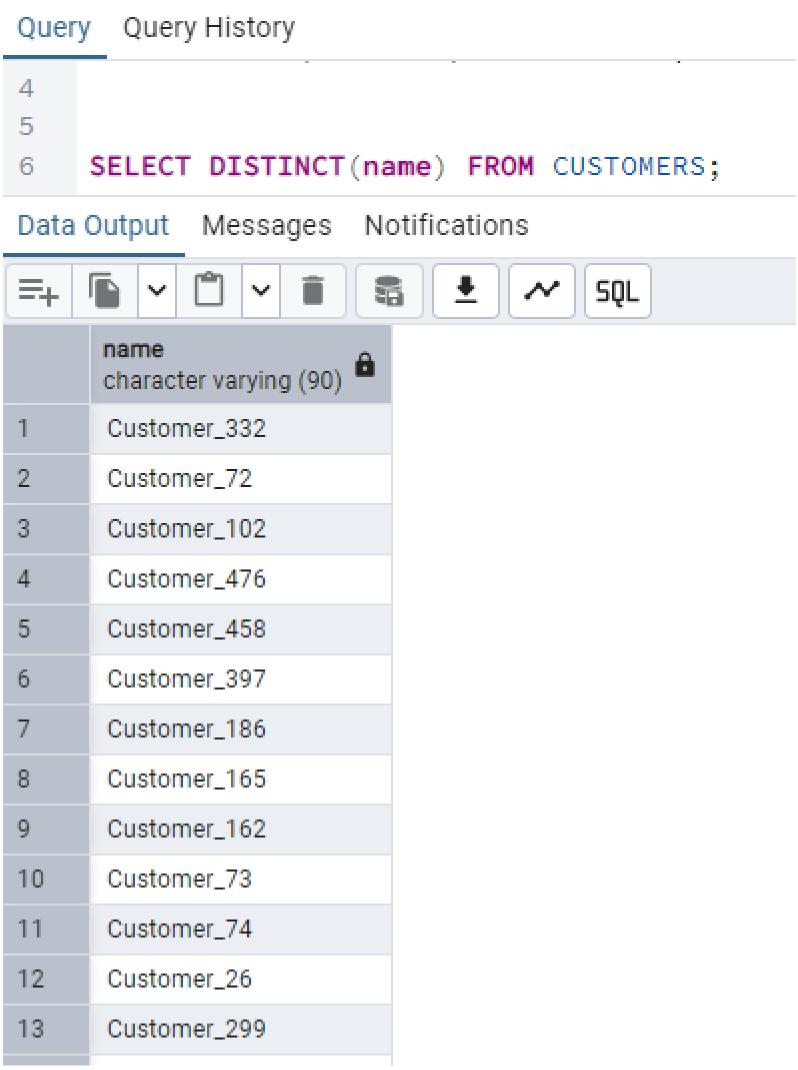
	count bigint
1	500

EXPLANATION:

This query returns the total number of orders present in the dataset.

➡️ The Dataset contains 500 orders, which forms basic of our analysis.

2. Count distinct customers



```
SELECT DISTINCT(name) FROM CUSTOMERS;
```

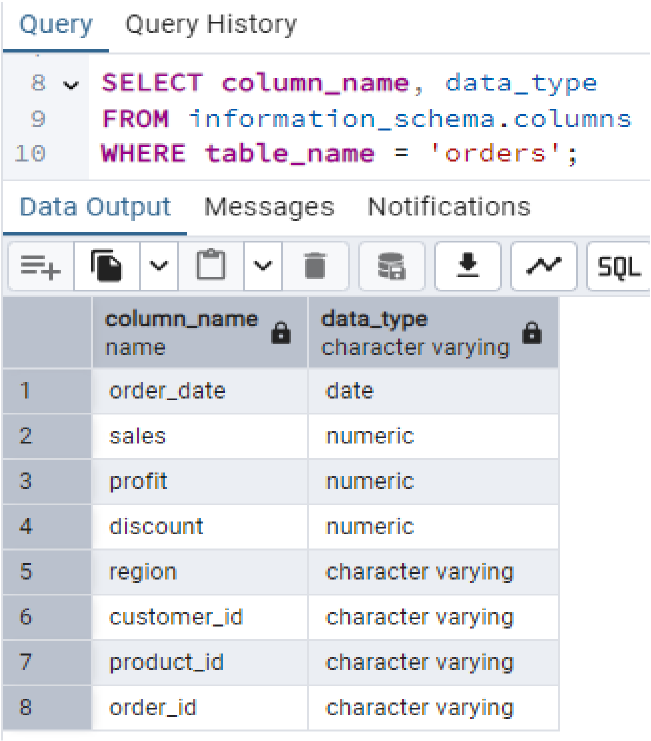
	name character varying (90)
1	Customer_332
2	Customer_72
3	Customer_102
4	Customer_476
5	Customer_458
6	Customer_397
7	Customer_186
8	Customer_165
9	Customer_162
10	Customer_73
11	Customer_74
12	Customer_26
13	Customer_299

EXPLANATION:

This query returns the total Unique customers present in the dataset.

➡️ The Dataset contains All 500 unique customer, in which I show you only 13 customer list on above .

3. **List all columns and their types from the `orders` table**



```sql
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'orders';
```

	column_name name	data_type character varying
1	order_date	date
2	sales	numeric
3	profit	numeric
4	discount	numeric
5	region	character varying
6	customer_id	character varying
7	product_id	character varying
8	order_id	character varying

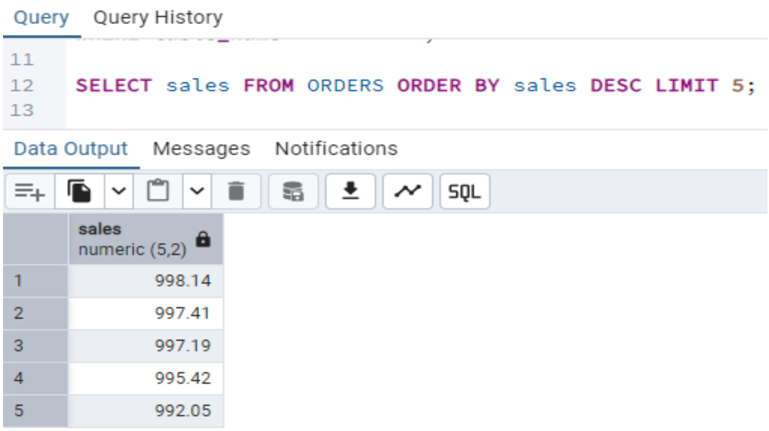
EXPLANATION:

This query returns the total name with there data type present in the dataset.

➡️ The Dataset contains only 8 column in which 3 column are Numeric, 4 column are varchar and 1 column are Date type.

2. SELECT, Filtering & Sorting Exploration

1. Show top 5 highest sales orders



```sql
SELECT sales FROM ORDERS ORDER BY sales DESC LIMIT 5;
```

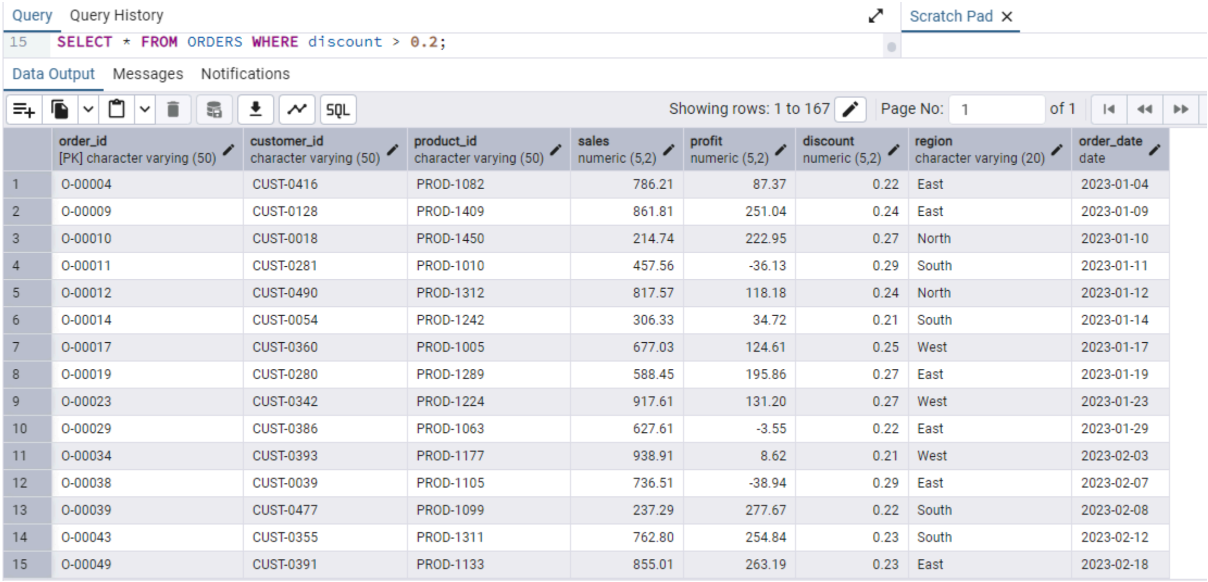
	sales numeric (5,2)
1	998.14
2	997.41
3	997.19
4	995.42
5	992.05

EXPLANATION:

This query returns the Top 5 Highest sales present in the dataset.

➡️ The Dataset contains the 998.14 as a Highest sales and soo on.

2. Filter orders where `discount > 0.2`



```sql
SELECT * FROM ORDERS WHERE discount > 0.2;
```

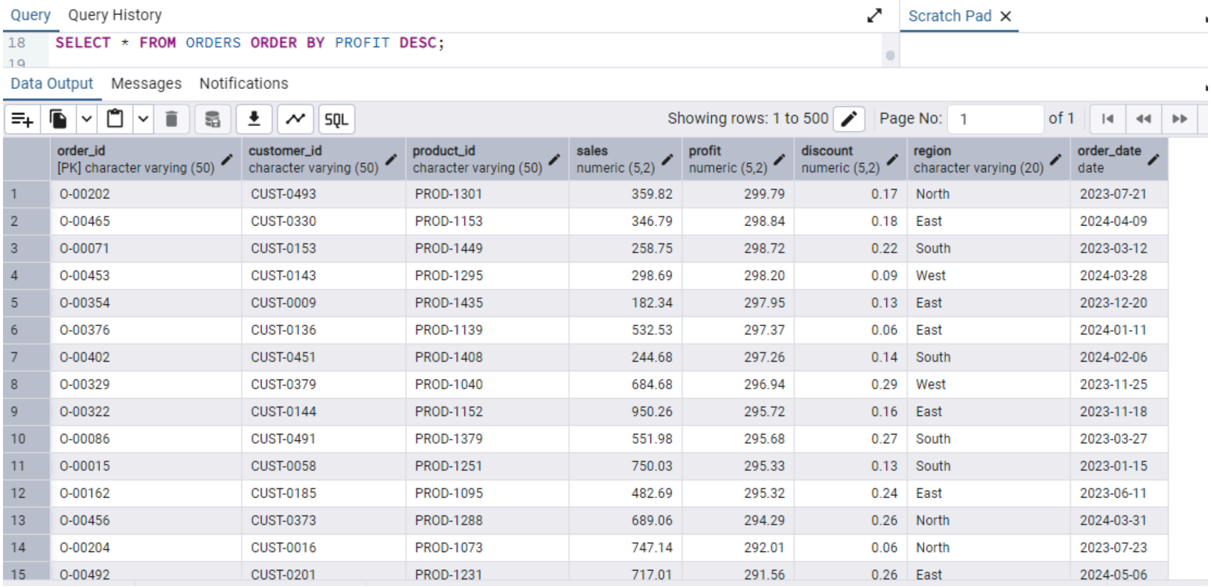
	order_id [PK] character varying (50)	customer_id character varying (50)	product_id character varying (50)	sales numeric (5,2)	profit numeric (5,2)	discount numeric (5,2)	region character varying (20)	order_date date
1	O-00004	CUST-0416	PROD-1082	786.21	87.37	0.22	East	2023-01-04
2	O-00009	CUST-0128	PROD-1409	861.81	251.04	0.24	East	2023-01-09
3	O-00010	CUST-0018	PROD-1450	214.74	222.95	0.27	North	2023-01-10
4	O-00011	CUST-0281	PROD-1010	457.56	-36.13	0.29	South	2023-01-11
5	O-00012	CUST-0490	PROD-1312	817.57	118.18	0.24	North	2023-01-12
6	O-00014	CUST-0054	PROD-1242	306.33	34.72	0.21	South	2023-01-14
7	O-00017	CUST-0360	PROD-1005	677.03	124.61	0.25	West	2023-01-17
8	O-00019	CUST-0280	PROD-1289	588.45	195.86	0.27	East	2023-01-19
9	O-00023	CUST-0342	PROD-1224	917.61	131.20	0.27	West	2023-01-23
10	O-00029	CUST-0386	PROD-1063	627.61	-3.55	0.22	East	2023-01-29
11	O-00034	CUST-0393	PROD-1177	938.91	8.62	0.21	West	2023-02-03
12	O-00038	CUST-0039	PROD-1105	736.51	-38.94	0.29	East	2023-02-07
13	O-00039	CUST-0477	PROD-1099	237.29	277.67	0.22	South	2023-02-08
14	O-00043	CUST-0355	PROD-1311	762.80	254.84	0.23	South	2023-02-12
15	O-00049	CUST-0391	PROD-1133	855.01	263.19	0.23	East	2023-02-18

EXPLANATION:

This query filter the data whose discount value is greater then 0.2

➡️ The Dataset contains the 167 data whose discount is more than 0.2 but I saw here only 15 data .

3. Sort orders by `profit` in descending order



```sql
SELECT * FROM ORDERS ORDER BY PROFIT DESC;
```

	order_id [PK] character varying (50)	customer_id character varying (50)	product_id character varying (50)	sales numeric (5,2)	profit numeric (5,2)	discount numeric (5,2)	region character varying (20)	order_date date
1	O-00202	CUST-0493	PROD-1301	359.82	299.79	0.17	North	2023-07-21
2	O-00465	CUST-0330	PROD-1153	346.79	298.84	0.18	East	2024-04-09
3	O-00071	CUST-0153	PROD-1449	258.75	298.72	0.22	South	2023-03-12
4	O-00453	CUST-0143	PROD-1295	298.69	298.20	0.09	West	2024-03-28
5	O-00354	CUST-0009	PROD-1435	182.34	297.95	0.13	East	2023-12-20
6	O-00376	CUST-0136	PROD-1139	532.53	297.37	0.06	East	2024-01-11
7	O-00402	CUST-0451	PROD-1408	244.68	297.26	0.14	South	2024-02-06
8	O-00329	CUST-0379	PROD-1040	684.68	296.94	0.29	West	2023-11-25
9	O-00322	CUST-0144	PROD-1152	950.26	295.72	0.16	East	2023-11-18
10	O-00086	CUST-0491	PROD-1379	551.98	295.68	0.27	South	2023-03-27
11	O-00015	CUST-0058	PROD-1251	750.03	295.33	0.13	South	2023-01-15
12	O-00162	CUST-0185	PROD-1095	482.69	295.32	0.24	East	2023-06-11
13	O-00456	CUST-0373	PROD-1288	689.06	294.29	0.26	North	2024-03-31
14	O-00204	CUST-0016	PROD-1073	747.14	292.01	0.06	North	2023-07-23
15	O-00492	CUST-0201	PROD-1231	717.01	291.56	0.26	East	2024-05-06

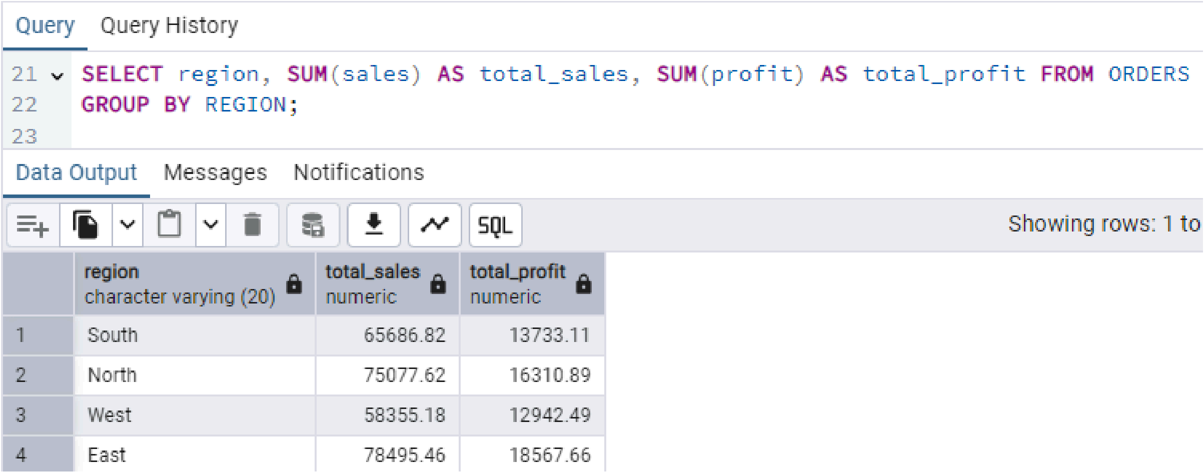
EXPLANATION:

This query return the data according to highest Profit.

➡️ The Dataset contain 299.79 as highest Profit and -48.27 as a Lowest Profit

3. Aggregates & GROUP BY Exploration

1. Show total sales and total profit by `region`



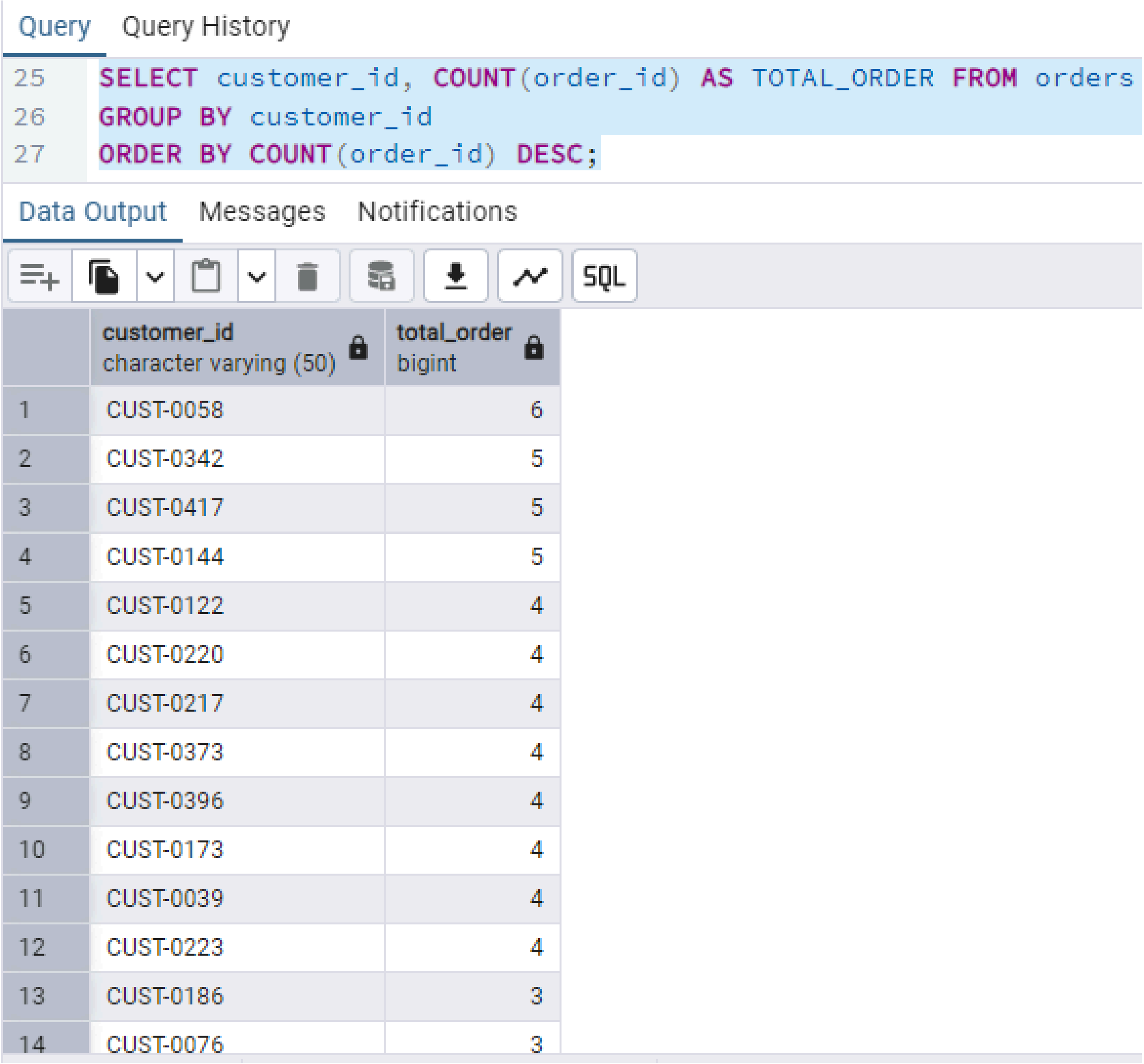
```sql
SELECT region, SUM(sales) AS total_sales, SUM(profit) AS total_profit FROM ORDERS
GROUP BY REGION;
```

	region character varying (20)	total_sales numeric	total_profit numeric
1	South	65686.82	13733.11
2	North	75077.62	16310.89
3	West	58355.18	12942.49
4	East	78495.46	18567.66

EXPLANATION:

This query return the total sales and total profit held in Each Region.
➡️ The Dataset contain four Region **EAST, WEST, NORTH AND SOUTH**
↪️ East Region have highest sales and profit: sales=78495.46 and Profit = 18567.66

2. Get the number of Orders per customer_id



```sql
SELECT customer_id, COUNT(order_id) AS TOTAL_ORDER FROM orders
GROUP BY customer_id
ORDER BY COUNT(order_id) DESC;
```

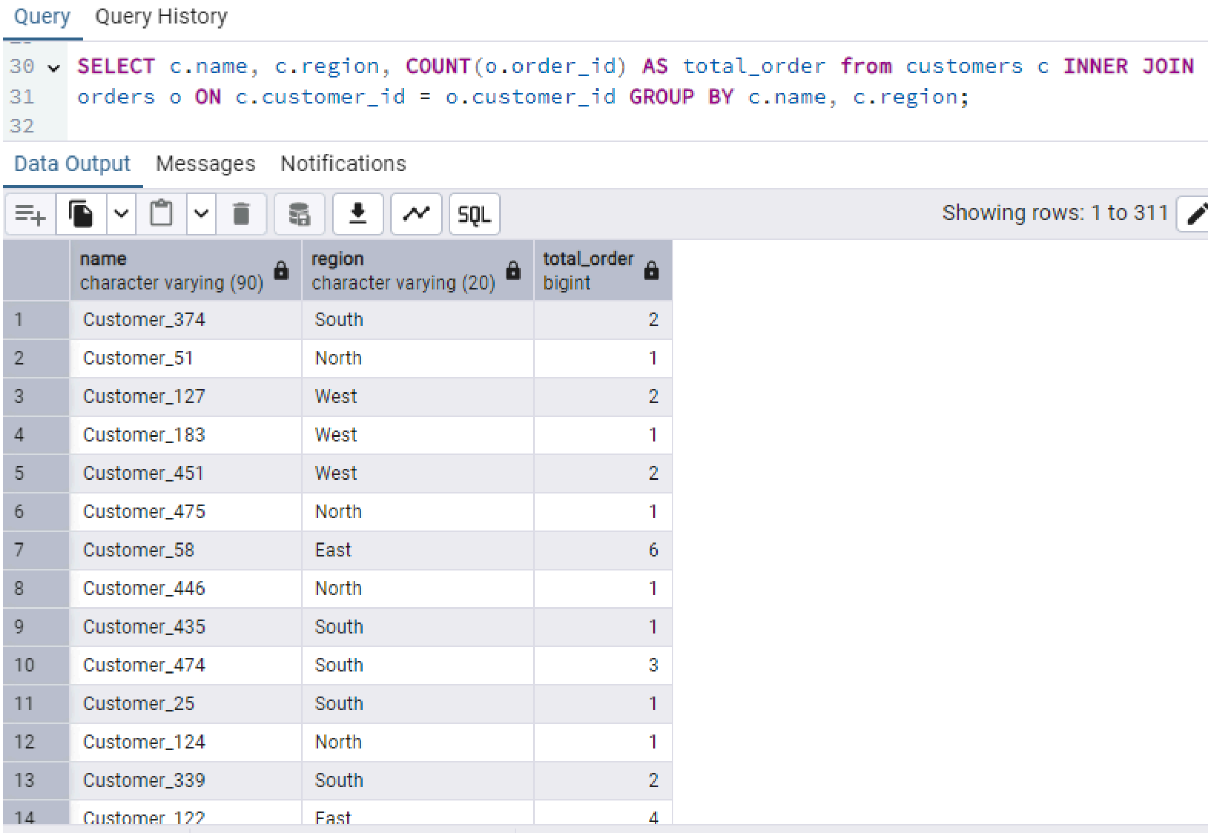
	customer_id character varying (50)	total_order bigint
1	CUST-0058	6
2	CUST-0342	5
3	CUST-0417	5
4	CUST-0144	5
5	CUST-0122	4
6	CUST-0220	4
7	CUST-0217	4
8	CUST-0373	4
9	CUST-0396	4
10	CUST-0173	4
11	CUST-0039	4
12	CUST-0223	4
13	CUST-0186	3
14	CUST-0076	3

EXPLANATION:

This query return the total orders placed by per customer
➡️ The Dataset contain highest number of order placed by only one Customer is 6.

4. Joins Exploration

1. List customer name, region, and their total orders



```
SELECT c.name, c.region, COUNT(o.order_id) AS total_order from customers c INNER JOIN
orders o ON c.customer_id = o.customer_id GROUP BY c.name, c.region;
```

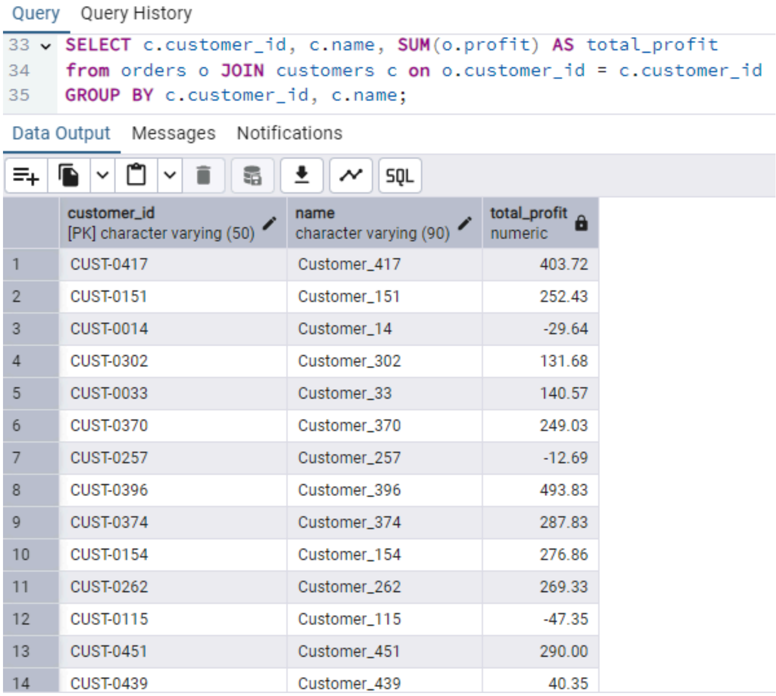
Showing rows: 1 to 311

	name character varying (90)	region character varying (20)	total_order bigint
1	Customer_374	South	2
2	Customer_51	North	1
3	Customer_127	West	2
4	Customer_183	West	1
5	Customer_451	West	2
6	Customer_475	North	1
7	Customer_58	East	6
8	Customer_446	North	1
9	Customer_435	South	1
10	Customer_474	South	3
11	Customer_25	South	1
12	Customer_124	North	1
13	Customer_339	South	2
14	Customer_122	East	4

EXPLANATION:

This query return the Customer Name, Region and Total Orders By per Customer
➡️ The Dataset contain Total 311 tatal data after filtter.
⏩ In this Query I join the Customer Table and order table.

2. Show total profit earned from each customer



```sql
SELECT c.customer_id, c.name, SUM(o.profit) AS total_profit
from orders o JOIN customers c on o.customer_id = c.customer_id
GROUP BY c.customer_id, c.name;
```

	customer_id [PK] character varying (50)	name character varying (90)	total_profit numeric
1	CUST-0417	Customer_417	403.72
2	CUST-0151	Customer_151	252.43
3	CUST-0014	Customer_14	-29.64
4	CUST-0302	Customer_302	131.68
5	CUST-0033	Customer_33	140.57
6	CUST-0370	Customer_370	249.03
7	CUST-0257	Customer_257	-12.69
8	CUST-0396	Customer_396	493.83
9	CUST-0374	Customer_374	287.83
10	CUST-0154	Customer_154	276.86
11	CUST-0262	Customer_262	269.33
12	CUST-0115	Customer_115	-47.35
13	CUST-0451	Customer_451	290.00
14	CUST-0439	Customer_439	40.35

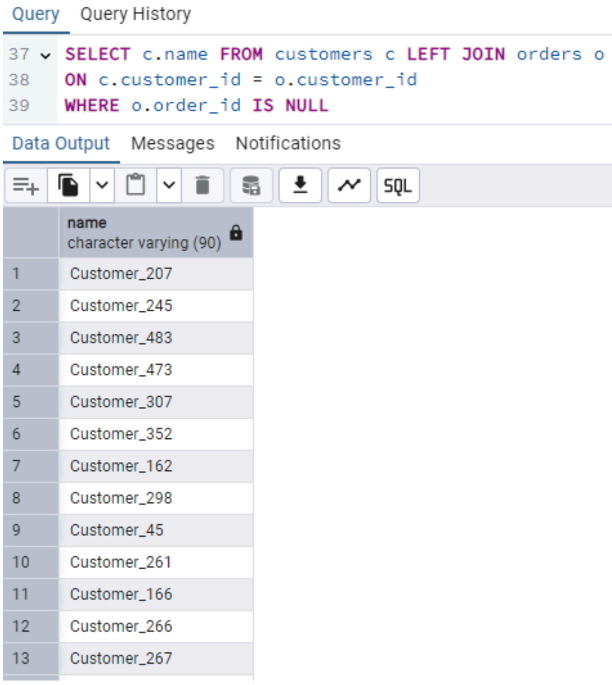
EXPLANATION:

This query return the Total Profit earned By each customer

➡️ The Dataset contain Total 311 tatal data after filtter.

⏩ In this Query I join the Customer Table and order table.

3. List all customers who placed no orders (`LEFT JOIN`)



```sql
SELECT c.name FROM customers c LEFT JOIN orders o
ON c.customer_id = o.customer_id
WHERE o.order_id IS NULL
```

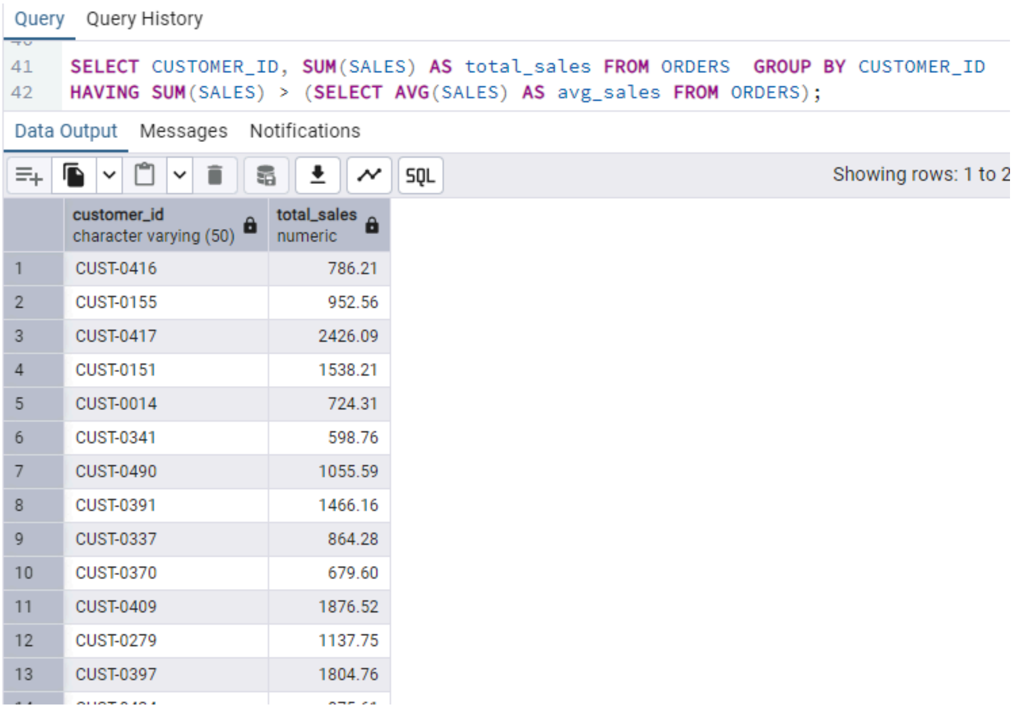
	name character varying (90)
1	Customer_207
2	Customer_245
3	Customer_483
4	Customer_473
5	Customer_307
6	Customer_352
7	Customer_162
8	Customer_298
9	Customer_45
10	Customer_261
11	Customer_166
12	Customer_266
13	Customer_267

EXPLANATION:

This query return the Customer Name who has No order
➡️ The Dataset contain Total 189 customer who have No order.
⏩ In this Query I use Left join the Customer Table and order table.

5. Subqueries Exploration:

1. Find customers whose total sales > average sales



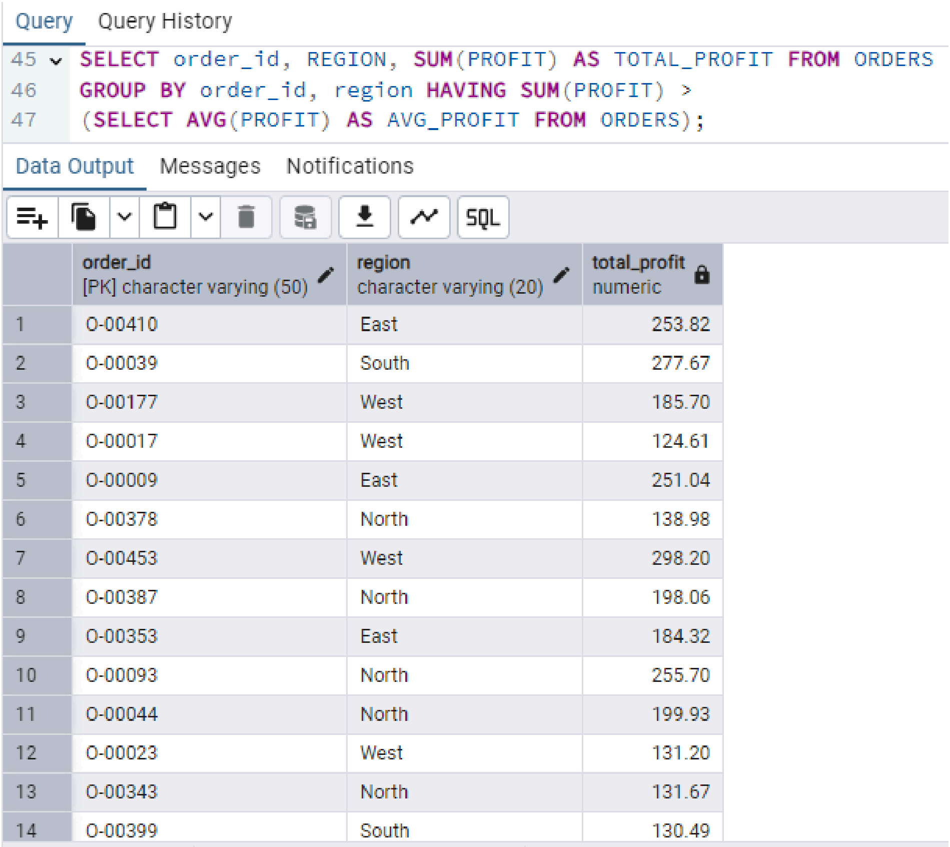
```
SELECT CUSTOMER_ID, SUM(SALES) AS total_sales FROM ORDERS  GROUP BY CUSTOMER_ID
HAVING SUM(SALES) > (SELECT AVG(SALES) AS avg_sales FROM ORDERS);
```

	customer_id character varying (50)	total_sales numeric
1	CUST-0416	786.21
2	CUST-0155	952.56
3	CUST-0417	2426.09
4	CUST-0151	1538.21
5	CUST-0014	724.31
6	CUST-0341	598.76
7	CUST-0490	1055.59
8	CUST-0391	1466.16
9	CUST-0337	864.28
10	CUST-0370	679.60
11	CUST-0409	1876.52
12	CUST-0279	1137.75
13	CUST-0397	1804.76

EXPLANATION:

This query return the Customer whose total sales is greater than avg sales
➡️ The Dataset contain Total 209 customer who have highest sales than avg sales.

2. Get orders where profit > average profit of their respective region



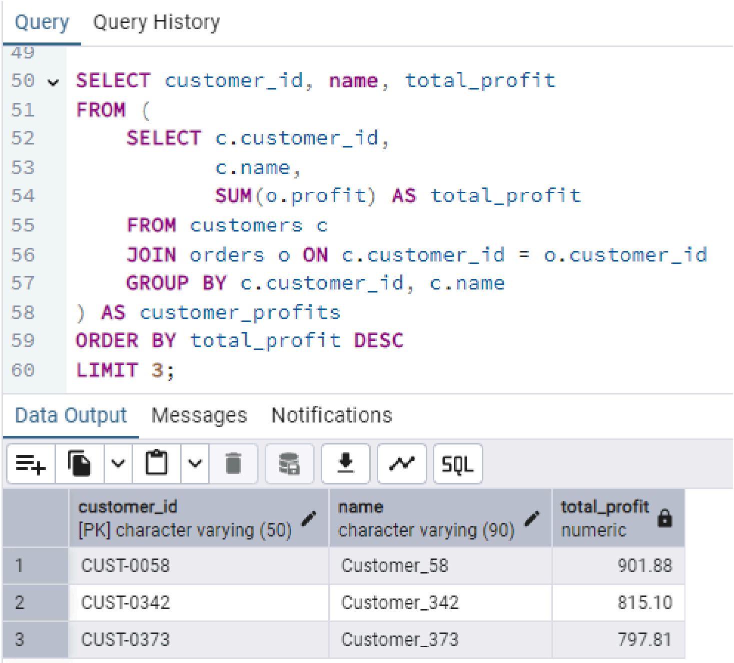
```
SELECT order_id, REGION, SUM(PROFIT) AS TOTAL_PROFIT FROM ORDERS
GROUP BY order_id, region HAVING SUM(PROFIT) >
(SELECT AVG(PROFIT) AS AVG_PROFIT FROM ORDERS);
```

	order_id [PK] character varying (50)	region character varying (20)	total_profit numeric
1	O-00410	East	253.82
2	O-00039	South	277.67
3	O-00177	West	185.70
4	O-00017	West	124.61
5	O-00009	East	251.04
6	O-00378	North	138.98
7	O-00453	West	298.20
8	O-00387	North	198.06
9	O-00353	East	184.32
10	O-00093	North	255.70
11	O-00044	North	199.93
12	O-00023	West	131.20
13	O-00343	North	131.67
14	O-00399	South	130.49

EXPLANATION:

This query return the orders whose total profit is greater than avg profit
➡️ The Dataset contain Total 254 orders who have highest profit than avg profit.

3. List top 3 customers with highest profit using subquery + `LIMIT`



```
SELECT customer_id, name, total_profit
FROM (
    SELECT c.customer_id,
           c.name,
           SUM(o.profit) AS total_profit
    FROM customers c
    JOIN orders o ON c.customer_id = o.customer_id
    GROUP BY c.customer_id, c.name
) AS customer_profits
ORDER BY total_profit DESC
LIMIT 3;
```

	customer_id [PK] character varying (50)	name character varying (90)	total_profit numeric
1	CUST-0058	Customer_58	901.88
2	CUST-0342	Customer_342	815.10
3	CUST-0373	Customer_373	797.81

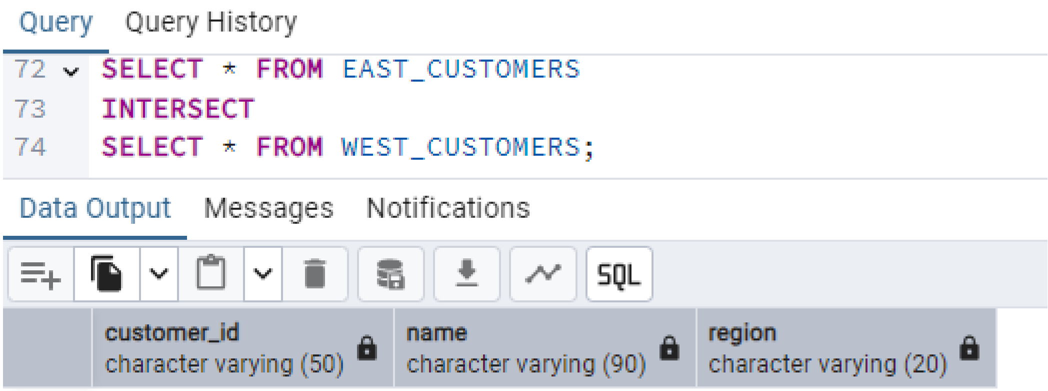
EXPLANATION:

This query return the Top 3 customer with highest Profit
➡️ The Dataset contain highest total profit is 901.88.

6. Set Operations Exploration:

1. Create two sample queries for customers from the East and West region

⏩ Use `INTERSECT` to find customers common to both

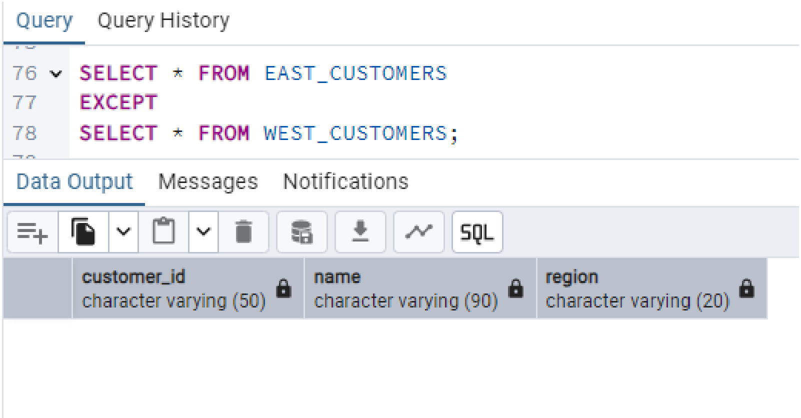


```sql
SELECT * FROM EAST_CUSTOMERS
INTERSECT
SELECT * FROM WEST_CUSTOMERS;
```

customer_id character varying (50)	name character varying (90)	region character varying (20)

EXPLANATION:
This query return 0 customer who common in both region

↪️ Use `EXCEPT` to find customers from East not in West



```sql
SELECT * FROM EAST_CUSTOMERS
EXCEPT
SELECT * FROM WEST_CUSTOMERS;
```

customer_id character varying (50)	name character varying (90)	region character varying (20)

EXPLANATION:
This query return 0 customer who common in both region so except query return 0.

🧠 Final Insights & Recommendations

1. **Customer Behavior:**
    - 311 out of 500 users placed orders → high engagement
    - Some regions (likely West/South) are stronger in both volume and profit
2. **Sales Strategy:**
    - Orders with **moderate discounts (~10-20%)** yield highest profits
    - **Top 3 products or customers** can be promoted for repeat purchases
3. **Growth Areas:**
    - Target inactive customers with marketing (via LEFT JOIN insights)
    - Analyze sub-category level (if available in future data)