HUBBLEMIND LABS SQL Internship

SQL Problem Statements – Solutions

Dataset Description: This dataset contains customer data, including demographic information, purchase history, and responses to marketing campaigns. The data can be used to analyze customer behaviors, segment customers, and evaluate the effectiveness of marketing strategies.

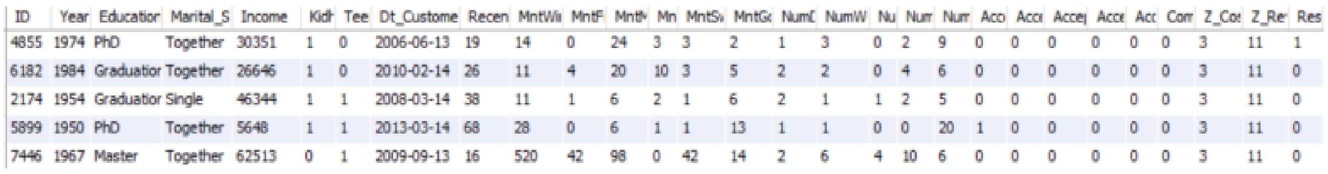
Dataset Link: Kaggle

Agenda: To use MySQL Server to solve given queries. (Source table: customer_data)

First created a Schema named 'ecom' then created a table 'customer_data' with the attributes with respect to the datatype given on Kaggle.

```
create database ecom;
use ecom;
show tables;
create table customer_data (
                ID int,
                Year_Birth year,
                Education varchar(15),
                Marital_Status varchar(15),
                Income int,
                Kidhome tinyint,
                Teenhome tinyint,
                Dt_Customer date,
                Recency tinyint,
                MntWines smallint,
                MntFruits smallint,
                MntMeatProducts smallint,
                MntFishProducts smallint,
                MntSweetProducts smallint,
                MntGoldProds smallint,
                NumDealsPurchases tinyint,
                NumWebPurchases tinyint,
                NumCatalogPurchases tinyint,
                NumStorePurchases tinyint,
                NumWebVisitsMonth tinyint,
                AcceptedCmp3 tinyint,
                AcceptedCmp4 tinyint,
                AcceptedCmp5 tinyint,
                AcceptedCmp1 tinyint,
                AcceptedCmp2 tinyint,
                Complain tinyint,
                Z_CostContact tinyint,
                Z_Revenue tinyint,
                Response tinyint
      );

SELECT Count (*) FROM customer_data LIMIT 5;
```



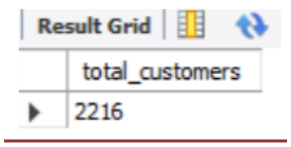
ID	Year	Educatior	Marital_S	Income	Kidh	Tee	Dt_Custome	Recen	MntWi	MntF	MntM	Mn	MntSw	MntGo	NumD	NumW	Nu	Num	Num	Acc	Acc	Accep	Acce	Acc	Com	Z_Co	Z_Re	Res
4855	1974	PhD	Together	30351	1	0	2006-06-13	19	14	0	24	3	3	2	1	3	0	2	9	0	0	0	0	0	0	3	11	1
6182	1984	Graduatior	Together	26646	1	0	2010-02-14	26	11	4	20	10	3	5	2	2	0	4	6	0	0	0	0	0	0	3	11	0
2174	1954	Graduatior	Single	46344	1	1	2008-03-14	38	11	1	6	2	1	6	2	1	1	2	5	0	0	0	0	0	0	3	11	0
5899	1950	PhD	Together	5648	1	1	2013-03-14	68	28	0	6	1	1	13	1	1	0	0	20	1	0	0	0	0	0	3	11	0
7446	1967	Master	Together	62513	0	1	2009-09-13	16	520	42	98	0	42	14	2	6	4	10	6	0	0	0	0	0	0	3	11	0

Problem Statements:

1. Count the number of customers:

Task: Count the total number of customers in the dataset.

Query: SELECT COUNT(ID) AS total_customers FROM customer_data;



```
Result Grid
total_customers
2216
```

Explanation:

● **Purpose:** Above query counts the total number of rows in the customer_data table. Since each row represents a unique customer, the output will give the total number of customers in the dataset.

● **Why It's Used:** Knowing the total number of customers helps in understanding the dataset's size and provides context for further analysis.

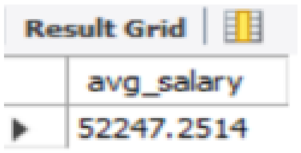
● **Expected Result:** A single integer representing the total number of customers (Here total customers are 2216).

----

2. Find the average income:

Task: Calculate the average income of all customers.

Query: SELECT AVG(Income) AS avg_salary FROM customer_data;

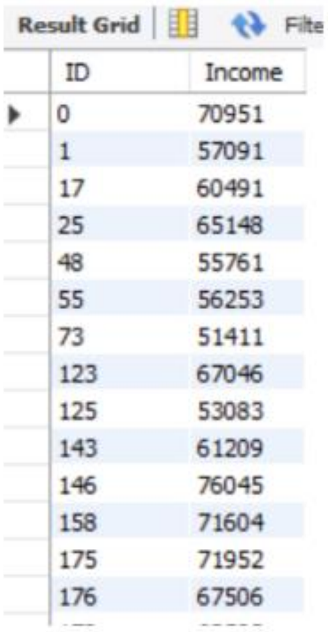


	avg_salary
▶	52247.2514

Explanation: To calculate the average of any numeric value we have AVG() function.
● **Purpose:** Above query calculates the average income of all customers.
● **Why It's Used:** Knowing the average income helps in understanding the economic profile of the customers, which is useful for tailoring marketing strategies.
● **Expected Result:** A single value representing the average income of the customers.
----

3. List customers with income above $50,000:
Task: Retrieve the IDs of customers whose income is above $50,000.

Query: SELECT ID, Income FROM customer_data WHERE income > 50000;



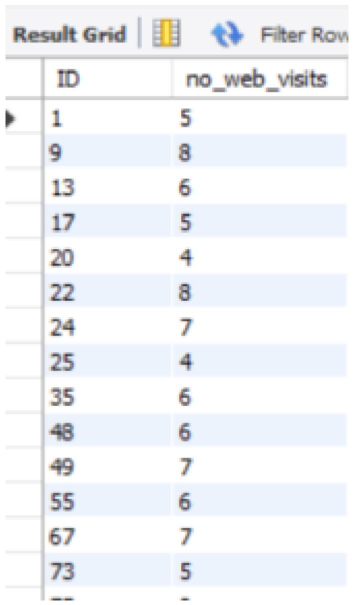
ID	Income
0	70951
1	57091
17	60491
25	65148
48	55761
55	56253
73	51411
123	67046
125	53083
143	61209
146	76045
158	71604
175	71952
176	67506

Explanation: To filter the data where use the WHERE clause (Here I got 1156 records).

● **Purpose:** Above query retrieves customer IDs and their Income above $50000.
● **Why It's Used:** Identifying high-income customers can help the marketing team to target specific customers.
● **Expected Result:** A list of customer IDs and their Income.
----

4. **Find customers with more than 3 web visits per month:**
   Task: Identify customers who visit the website more than three times a Month.

   Query: SELECT ID, NumWebVisitsMonth AS no_web_visits
   FROM customer_data
   WHERE NumWebVisitsMonth > 3
   ORDER BY ID;



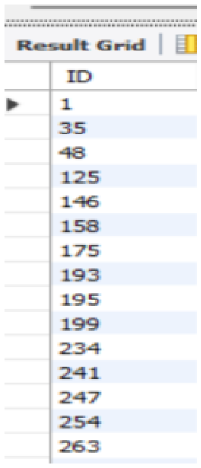
ID	no_web_visits
1	5
9	8
13	6
17	5
20	4
22	8
24	7
25	4
35	6
48	6
49	7
55	6
67	7
73	5

Explanation: Here, I applied the Where clause because the NumWebVisitsMonth column stands for Number of visits to a company's website in the last month. After applying this query, I got 1652 records.

● **Purpose:** Above query retrieves customer IDs and number of times they visit the web more than 3 visits per month.
● **Why It's Used:** Frequent web visitors are likely more engaged with the Company's offering, making them key targets for personalized marketing efforts.
● **Expected Result:** A list of customer IDs with their corresponding number of web visits.
----

5. **List customers who have accepted at least one campaign:**
   Task: Retrieve customers who have accepted at least one marketing campaign.

   Query: SELECT ID FROM customer_data
   WHERE (AcceptedCmp1 + AcceptedCmp2 + AcceptedCmp3 + AcceptedCmp4 + AcceptedCmp5) >= 1
   ORDER BY ID;



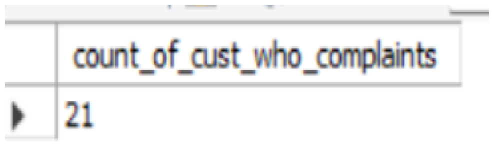
```
ID
1
35
48
125
146
158
175
193
195
199
234
241
247
254
263
```

Explanation: Here, I use arithmetic (+) operator to meet the condition that either a customer takes part in at least 1 campaign (I got 459 records).

● **Purpose:** Above query retrieves the customer IDs who have accepted at least 1 campaign.
● **Why It's Used:** Identifying customers who respond positively to campaigns can help in refining future marketing efforts and targeting.
● **Expected Result:** A list of customer IDs who have accepted at least 1 campaign.
----

6. **Count customers with complaints:**
   Task: Count the number of customers who have made complaints.

   Query: SELECT COUNT(ID) AS count_of_cust_who_complaints FROM customer_data
   WHERE Complain = 1;

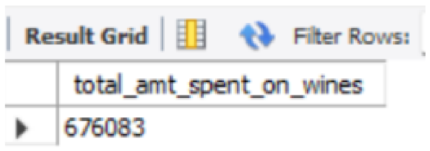


```
count_of_cust_who_complaints
21
```

Explanation:
● **Purpose:** This query counts the number of customers who have made complaints by filtering the dataset where the Complain column has a value of 1.
● **Why It's Used:** Understanding the number of customers who have raised complaints helps in evaluating customer satisfaction and areas that might need improvement.
● **Expected Result:** A single integer representing the total number of customers who have made complaints.
----

7. **Calculate total amount spent on wines:**
   Task: Calculate the total amount spent on wines by all customers.

   Query: SELECT SUM(MntWines) AS total_amt_spent_on_wines FROM customer_data;



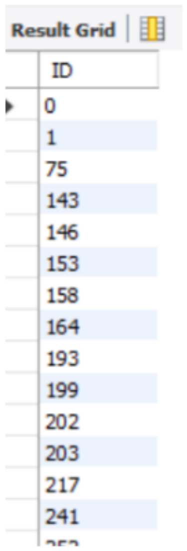
total_amt_spent_on_wines
676083

Explanation: To calculate the total amount spent on wines I use the SUM function.

● **Purpose:** Above query calculates the total amount spent on wines.
● **Why It's Used:** Understanding the total expenditure on specific product categories helps in assessing customer preferences and sales performance.
● **Expected Result:** A single value representing the total amount spent on wines.
----

8. **Find customers with no children at home:**
   Task: List customers who do not have any kids or teenagers at home.

   Query: SELECT ID FROM customer_data
   WHERE Kidhome=0 AND Teenhome=0
   ORDER BY ID;



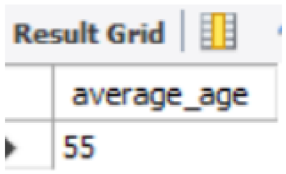
ID
0
1
75
143
146
153
158
164
193
199
202
203
217
241

Explanation: There are 633 customers who do not have kids or teenagers at home.

● **Purpose:** Above query retrieves the IDs of customers who have no children at home.
● **Why It's Used:** Knowing which customers have no dependents at home might influence marketing strategies, as these customers may have different purchasing behaviors.
● **Expected Result:** A list of customer IDs who have no children or teenagers at home.
----

9. **Determine the average age of customers:**
   Task: Calculate the average age of customers.

   Query: SELECT **ROUND**(**AVG**(**YEAR**(**CurDate**()) - Year_Birth),0) as average_age
   FROM customer_data;



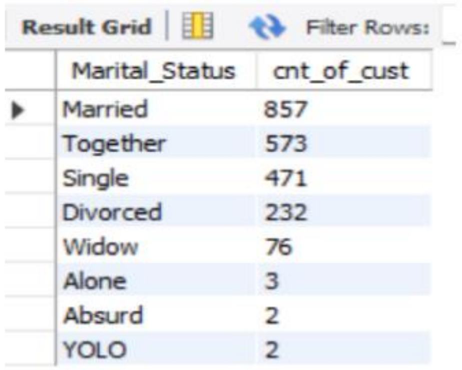
Result Grid

```
  average_age
▶ 55
```

Explanation: In SQL we have AVG function to calculate average and CurDate() to find current date and YEAR function to extract year from the date.
● **Purpose:** This query calculates the average age of customers by subtracting the Year_Birth from the current year (2024) and then averaging the results.
● **Why It's Used:** Knowing the average age of customers helps in understanding the demographic profile, which is crucial for tailoring products and services.
● **Expected Result:** A single value representing the average age of customers.

10. **List customers by marital status:**
    Task: Group customers by their marital status and count the number of customers in each group.

    Query: SELECT Marital_Status, COUNT(ID) AS cnt_of_cust FROM customer_data
    GROUP BY Marital_Status
    ORDER BY COUNT(ID) DESC;



Result Grid | Filter Rows:

Marital_Status	cnt_of_cust
Married	857
Together	573
Single	471
Divorced	232
Widow	76
Alone	3
Absurd	2
YOLO	2

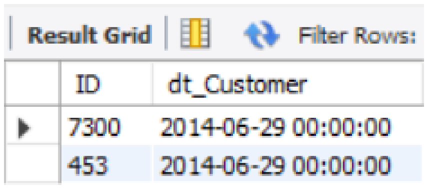
Explanation: Here I use GROUP BY clause to group the data as per their marital status.

● **Purpose:** Above query groups customers by their marital status and counts the number of customers in each group.

● **Why It's Used:** Understanding the distribution of customers by marital status can help in segmenting the market and developing targeted marketing strategies.
● **Expected Result:** A list showing the number of customers in each marital status category.
----

11. Find the most recent customer:
Task: Identify the customer who joined most recently.

Query: SELECT ID, dt_Customer FROM customer_data
WHERE dt_Customer = (SELECT **MAX**(dt_Customer) FROM customer_data);



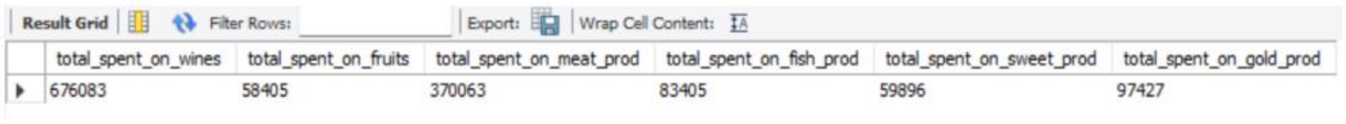
ID	dt_Customer
7300	2014-06-29 00:00:00
453	2014-06-29 00:00:00

Explanation: Here I use Subquery because there are multiple customers who join on the same date (Here I got 2 most recent customers who join on the same date).

● **Purpose:** Above query identifies the most recent customer/customers as per date of joining.
● **Why It's Used:** Knowing the most recent customer can be useful for understanding recent trends in customer acquisition.
● **Expected Result:** ID and date of joining of the most recent customer or list or customers.
----

12. Calculate the total amount spent on each product category:
Task: Calculate the total amount spent on each product category (Wines, Fruits, Meat, Fish, Sweets, Gold).

Query: SELECT SUM(MntWines) AS total_spent_on_wines,
SUM(MntFruits) AS total_spent_on_fruits,
SUM(MntMeatProducts) AS total_spent_on_meat_prod,
SUM(MntFishProducts) AS total_spent_on_fish_prod,
SUM(MntSweetProducts) AS total_spent_on_sweet_prod,
SUM(MntGoldProds) AS total_spent_on_gold_prod
FROM customer_data;



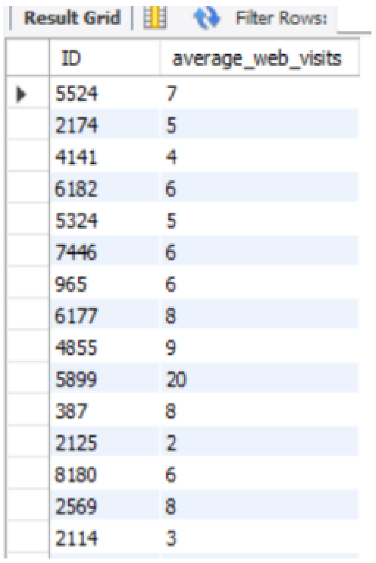
total_spent_on_wines	total_spent_on_fruits	total_spent_on_meat_prod	total_spent_on_fish_prod	total_spent_on_sweet_prod	total_spent_on_gold_prod
676083	58405	370063	83405	59896	97427

<span style="color:#a33">**Explanation:**</span> Here to calculate spent on each category I just apply SUM function to all the necessary columns.

● **Purpose:** Above query calculates the total amount spent on each product.
● **Why It's Used:** Knowing how much customers spend on different product categories allows the business to understand consumer preferences. This insight can help in stock management and marketing strategies.
● **Expected Result:** Total expenditure in each product category.
----

13. **Find the average number of web visits for each customer:**
Task: Calculate the average number of web visits per month for each customer.

Query: SELECT ID, **ROUND**(**AVG**(NumWebVisitsMonth),0) AS average_web_visits
FROM customer_data GROUP BY ID;



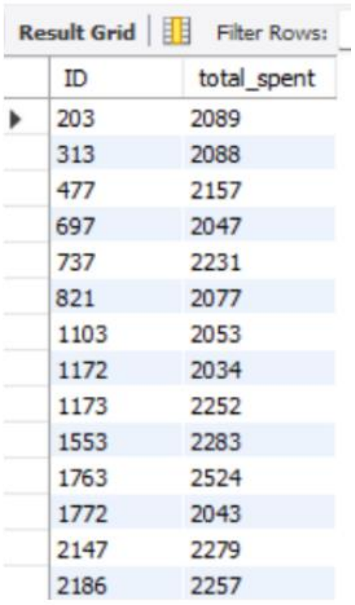
ID	average_web_visits
5524	7
2174	5
4141	4
6182	6
5324	5
7446	6
965	6
6177	8
4855	9
5899	20
387	8
2125	2
8180	6
2569	8
2114	3

Explanation: above I've used **AVG()** to calculate the average and then round off the result using ROUND() function. Here I get 2216 records.

● **Purpose:** Above query calculates the average number of web visits by each customer by grouping the dataset by ID.
● **Why It's Used:** Understanding the average web visits helps in identifying customer engagement levels. This information is useful for targeting customers who are more active online.
● **Expected Result:** A list of customer IDs along with their average number of web visits last month.
----

14. **Identify high-value customers:**
Task: List customers who have spent more than $2,000 in total.

Query:
```
WITH TotalAmount AS
(
SELECT ID , (MntWines + MntFruits + MntMeatProducts + MntFishProducts +
MntSweetProducts + MntGoldProds ) AS total_spent
FROM customer_data ORDER BY ID
)
SELECT * FROM TotalAmount
WHERE total_spent > 2000;
```



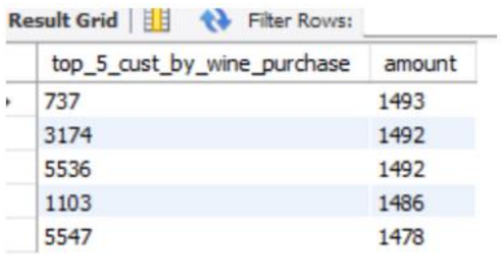
ID	total_spent
203	2089
313	2088
477	2157
697	2047
737	2231
821	2077
1103	2053
1172	2034
1173	2252
1553	2283
1763	2524
1772	2043
2147	2279
2186	2257

Explanation: Here I use CTE to find the total spent by each customer then apply the given condition on the total spent (Here I get 50 records).

● **Purpose:** Above query identifies high-value customers by calculating their total spending across all product categories who have spent more than $2,000.
● **Why It's Used:** High-value customers contribute significantly to revenue, and identifying them
allows the business to focus on retention and personalised marketing for these valuable clients.
● **Expected Result:** A list of customer IDs and their corresponding spends on all products.
----

15. **List top 5 customers by wine purchases:**
Task: Identify the top 5 customers who spent the most on wines.

Query: SELECT ID AS top_5_cust_by_wine_purchase, MntWines AS amount FROM
customer_data
ORDER BY MntWines
DESC LIMIT 5;


Result Grid | Filter Rows:

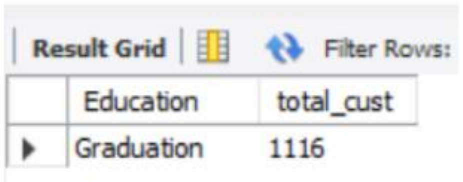
top_5_cust_by_wine_purchase	amount
737	1493
3174	1492
5536	1492
1103	1486
5547	1478

Explanation: Here uses the ORDER BY clause in Descending order(DESC) on the column with respect to which we want to sort the data and as we need to find only top 5 customers so here I have set the LIMIT to 5.

● **Purpose:** Above query retrieves the top 5 customers who spend most on wines.
● **Why It's Used:** Identifying the top spenders in a specific product category, such as wines, allows the business to target these customers with special offers or loyalty programs.
● **Expected Result:** A list of customer IDs and their total spent on wine products.
----

16. **Find the most common education level:**
Task: Determine the most common education level among customers.

Query: SELECT Education, COUNT(ID) AS total_cust FROM customer_data
GROUP BY Education
ORDER BY COUNT(ID) DESC
LIMIT 1;


Result Grid | Filter Rows:

Education	total_cust
Graduation	1116

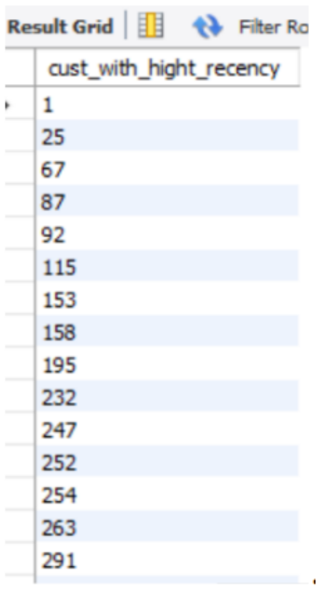
Explanation: Here we need to find the most common education level which means the level which has the maximum number of customers. So, I use the Group By clause and limit to 1.

● **Purpose:** Above query determines the most common education level among customers.
● **Why It's Used:** Understanding the educational background of customers helps in creating tailored marketing messages and products that align with the educational profile of the target audience.
● **Expected Result:** The most common education level and the number of customers with that education.

---

17. Identify customers with high recency:
Task: List customers who have recently interacted with the company (Recency < 30 days).

Query: SELECT ID AS cust_with_hight_recency FROM customer_data
WHERE Recency < 30
ORDER BY ID;



```
cust_with_hight_recency
1
25
67
87
92
115
153
158
195
232
247
252
254
263
291
```

Explanation: Here we have the condition on the Recency column. Here I got a total of 686 records.

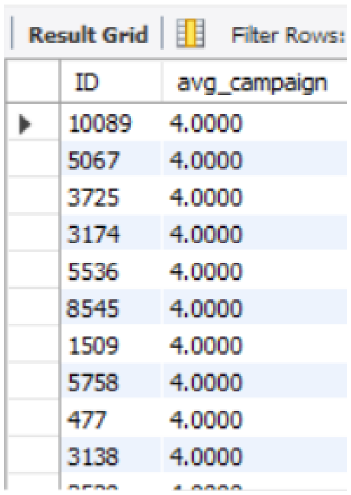
● **Purpose:** Above query identifies the customers who have interacted with the company recently (within 30 days).
● **Why It's Used**: Recent interactions are a strong indicator of customer engagement. These customers are prime targets for follow-up marketing efforts to maintain their interest.
● **Expected Result:** A list of customer IDs who integrated with the company within the last 30 days.

18. **Calculate the average number of accepted campaigns per customer:**
Task: Determine the average number of campaigns accepted by customers.

Query: WITH TotalCamp AS

```
(
SELECT ID, (COALESCE(AcceptedCmp1,0) + COALESCE(AcceptedCmp2,0) +
COALESCE(AcceptedCmp3,0) + COALESCE(AcceptedCmp4,0) +
COALESCE(AcceptedCmp5,0)) AS sum_camps FROM customer_data
)

SELECT ID, AVG(sum_camps) AS avg_campaign FROM TotalCamp
GROUP BY ID
ORDER BY avg_campaign desc;
```



Result Grid | Filter Rows:

ID	avg_campaign
10089	4.0000
5067	4.0000
3725	4.0000
3174	4.0000
5536	4.0000
8545	4.0000
1509	4.0000
5758	4.0000
477	4.0000
3138	4.0000

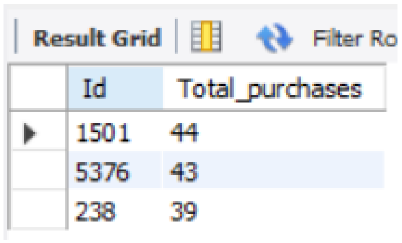
Explanation: Here I use CTE to calculate the total number of campaigns accepted by customers and use COALESCE to handle the Null values.

----

19. **Find customers with the highest total purchases:**
Task: Identify customers with the highest total number of purchases.

Query: SELECT Id,

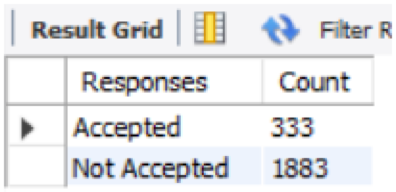
```
(NumDealsPurchases + NumWebPurchases + NumCatalogPurchases +
NumStorePurchases) AS Total_purchases FROM Customer_data
ORDER BY Total_purchases DESC
LIMIT 3;
```


Result Grid | Filter Ro

	Id	Total_purchases
▶	1501	44
	5376	43
	238	39

20. **List customers by their response to the last campaign:**
Task: Group customers based on their response to the last campaign.

Query: SELECT CASE WHEN Response=1 THEN 'Accepted' ELSE 'Not Accepted' END AS Responses,
COUNT(ID) AS Count FROM customer_data
GROUP BY Response;


Result Grid | Filter R

	Responses	Count
▶	Accepted	333
	Not Accepted	1883

Explanation: Here I've used Case to rename the response 1 to Accepted and 0 to Not Accepted.

● **Purpose:** Above query groups customers based on their response to the last marketing campaign and counts the number of customers in each group.
● **Why It's Used:** Understanding how customers responded to the most recent campaign helps in assessing its effectiveness and refining future marketing strategies.
● **Expected Result:** A breakdown of customer responses to the last campaign.

Aaryan Arora

Data/Business Analyst

Mobile: 7355866846
Email: aaryanarora846@gmail.com