OMKAR MORE

**HUBBLEMIND 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 analyse customer behaviour, segment customers, and evaluate the effectiveness of marketing strategies.

**Dataset Link:** [**Kaggle**](https://www.kaggle.com/datasets/krsign/ecom-dataset)

**GitHub Link:**[**Solution**](https://github.com/omkar3751/Hubblemind)

**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,

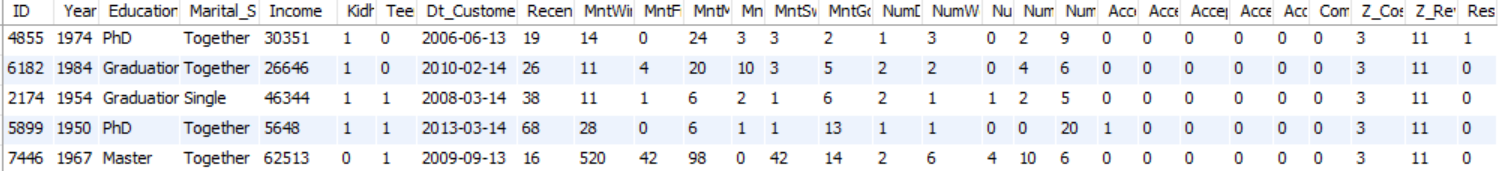
AcceptedCmp4 tinyint, AcceptedCmp5 tinyint, AcceptedCmp1 tinyint,

AcceptedCmp2 tinyint, Complain tinyint, Z\_CostContact tinyint,

Z\_Revenue tinyint, Response tinyint

);

select count(\*) from customer\_data limit 5;

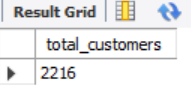
****

**Problem Statements:**

**1. Count the number of customers:**

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

**Query:** **SELECT COUNT**(ID) **FROM** customer\_data;



**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

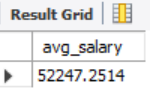
* **Expected Result**: A single integer representing the total number of customers (Here total customers are 2216).

------------------------------------------------------------------------------------------------------------------------------------

**2. Find the average income:**

**Task:** Calculate the average income of all customers.

**Query: SELECT AVG**(Income) **AS** avg\_salary **FROM** customer\_data;



**Explanation:** To calculate the average of any numeric value we have AVG() function.

* **Purpose**: Above query calculates the average income of all customers.
* **Expected Result**: A single value representing the average income of the

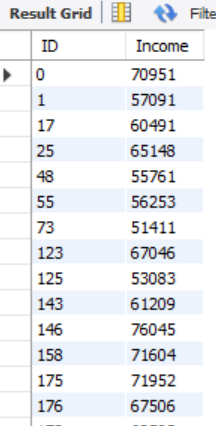
customers.

----------------------------------------------------------------------------------------------------------------------------

**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;



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

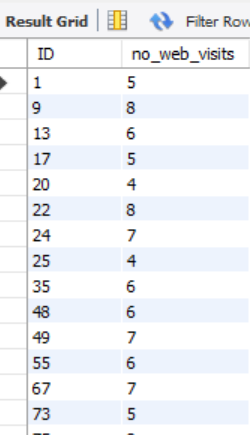
* **Purpose**: Above query retrieves customer IDs and their Income above $50000.
* **Expected Result**: A list of customer IDs and their Income.

----------------------------------------------------------------------------------------------------------------------------

**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;



**Explanation:** Here ,I applied the Where clause because the NumWebVisitsMonthcolumn 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.
* **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.

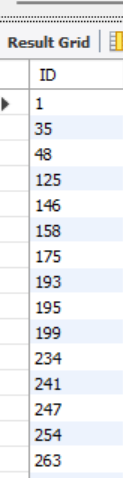
**Query1: SELECT** ID **FROM** customer\_data **WHERE** AcceptedCmp1 = 1 **OR**

AcceptedCmp2 = 1 **OR**  AcceptedCmp3 = 1 **OR** AcceptedCmp4 = 1 **OR**

AcceptedCmp5 = 1 **ORDER BY** ID;

**Query2: SELECT** ID **FROM** customer\_data **WHERE** 1 **IN** (AcceptedCmp1, AcceptedCmp2, AcceptedCmp3, AcceptedCmp4, AcceptedCmp5)

**ORDER BY** ID**;**



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

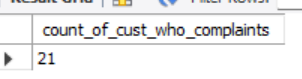
* **Purpose**: Above query retrieves the customer IDs who have accepted at least 1 campaign.
* **Expected Result**: A list of customer IDs who have accepted at least 1 campaign.

----------------------------------------------------------------------------------------------------------------------------

**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;

****

**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.

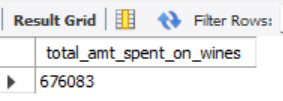
* **Expected Result**: A single integer representing the total number of customers who have made complaints

----------------------------------------------------------------------------------------------------------------------------

**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;

****

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

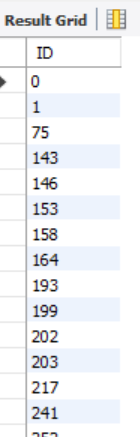
* **Purpose**: Above query calculates the total amount spent on wines.
* **Expected Result**: A single value representing the total amount spent on wines.

----------------------------------------------------------------------------------------------------------------------------

**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;

****

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

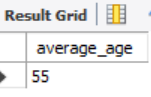
* **Purpose**: Above query retrieves the IDs of customers who have no children at home.
* **Expected Result**: A list of customer IDs who have no children or teenagers at home.

----------------------------------------------------------------------------------------------------------------------------

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

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

**Query: SELECT ROUND((AVG**((**YEAR**(**NOW()**) - Year\_Birth)),0) **AS** average\_age **FROM** customer\_data;

****

**Explanation:** In SQL we have AVG function to calculate average and NOW() 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.
* **Expected Result**: A single value representing the average age of customers.

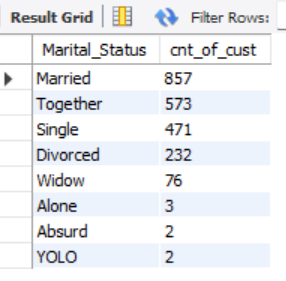
----------------------------------------------------------------------------------------------------------------------------

**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**;



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

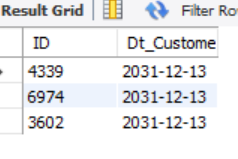
* **Purpose**: Above query groups customers by their marital status and counts the number of customers in each group.
* **Expected Result**: A list showing the number of customers in each marital status category.

----------------------------------------------------------------------------------------------------------------------------

**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);



**Explanation:** Here I use Subquery because there are multiple customers who join on the same date (Here i got 3 most recent customers who join on the same date).

* **Purpose**: Above query identifies the most recent customer/customers as per date of joining.
* **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,

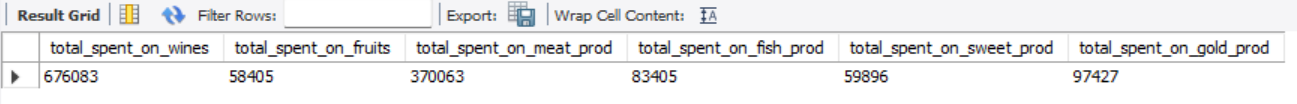
**SUM**(MntMeatProducts) **AS** total\_spent\_on\_meat\_prod,

**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;



**Explanation:** 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.
* **Expected Result**: Total expenditure in each product category.

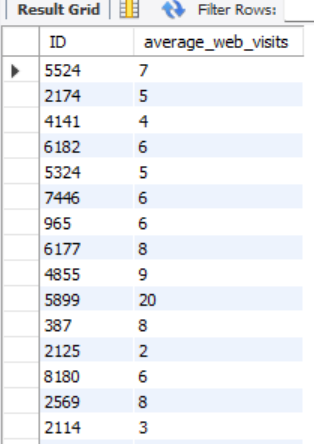
----------------------------------------------------------------------------------------------------------------------------

**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;

****

**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.
* **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

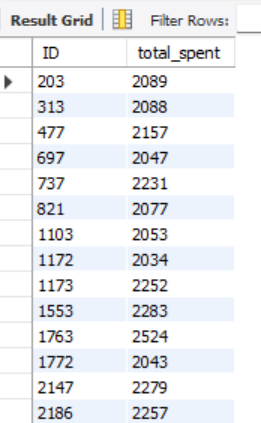
+ MntSweetProducts + MntGoldProds ) **AS** total\_spent

**FROM** customer\_data **ORDER BY** ID

)

**SELECT** \* **FROM** TotalAmount

**WHERE** total\_spent > 2000;



**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).

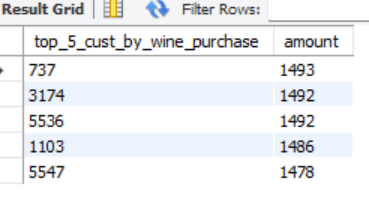
* **Purpose**: Above query identifies high-value customers by calculating their total spending across all product categories who have spent more than $2,000.
* **Expected Result**: A list of customer IDs and their corresponding spends on all products.

----------------------------------------------------------------------------------------------------------------------------

**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;



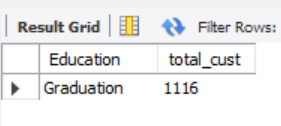
**Explanation:** Here uses the ORDER BY clause in Descending order(DESC) on the column with respect to which we want to sort the data and as we need to find only top 5 customers so here I have set the LIMIT to 5.

* **Purpose**: Above query retrieves the top 5 customers who spend most on wines.
* **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;



**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.

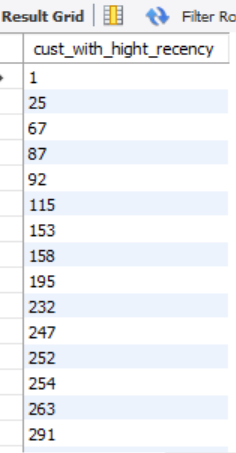
* **Purpose**: Above query determines the most common education level among customers.
* **Expected Result**: The most common education level and the number of customers with that education.

----------------------------------------------------------------------------------------------------------------------------

**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;

**......**

**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).
* **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** cnt\_camp

**FROM** customer\_data

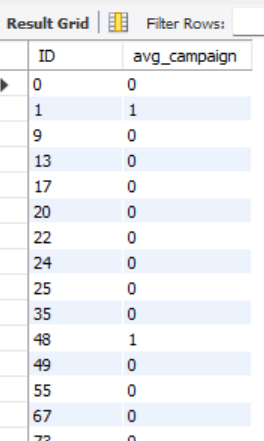
)

**SELECT** ID, **CEIL**(**AVG**(cnt\_camp)) **AS** avg\_campaign

**FROM** TotalCamp

**GROUP BY** ID

**ORDER BY** ID;

.....

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

* **Purpose**: Above query calculates the average number of campaigns accepted by each customer.
* **Expected Result**: A list of customer IDs with average number of campaign accepted by them.

----------------------------------------------------------------------------------------------------------------------------

**19. Find customers with the highest total purchases:**

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

**Query: WITH** TotalPurchase **AS** (

**SELECT** ID,

(

**COALESCE**(NumDealsPurchases,0) +

**COALESCE**(NumWebPurchases,0) +

**COALESCE**(NumCatalogPurchases,0) +

**COALESCE**(NumStorePurchases,0)

) **AS** ttl\_pur

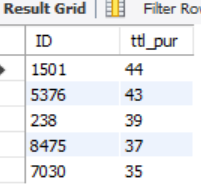
**FROM** customer\_data

)

**SELECT** \* **FROM** TotalPurchase

**ORDER BY** ttl\_pur **DESC**

**LIMIT** 5;

****

**Explanation:** Here are the Top 5 customers who made the highest purchase.

* **Purpose**: Above query identifies customers with the highest number of purchases

across all product categories.

* **Expected Result**: A list of the top 5 customers with their total number of purchases.

----------------------------------------------------------------------------------------------------------------------------

**20. List customers by their response to the last campaign:**

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

**Query:** **SELECT CASE**

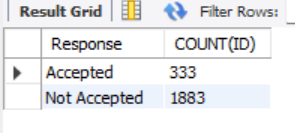
**WHEN** Response=1 **THEN** 'Accepted'

**ELSE** 'Not Accepted'

**END AS** Response

, **COUNT**(ID)

**FROM** customer\_data **GROUP BY** Response;



**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.

****