-----------------------Combining RFM Metrics--------------------------

WITH RecencyCTE AS (

SELECT

CustomerID,

DATEDIFF(DAY, SubscriptionDate, GETDATE()) AS RecencyDays

FROM Customers

),

FrequencyCTE AS (

SELECT

CustomerID,

1 AS Frequency -- Placeholder for lack of frequency data

FROM Customers

),

MonetaryCTE AS (

SELECT

CustomerID,

DATEDIFF(DAY, SubscriptionDate, GETDATE()) \* 10 AS Monetary -- Proxy metric

FROM Customers

)

SELECT

r.CustomerID,

CONCAT(c.FirstName, ' ', c.LastName) AS CustomerName,

r.RecencyDays,

f.Frequency,

m.Monetary

FROM

RecencyCTE r

JOIN

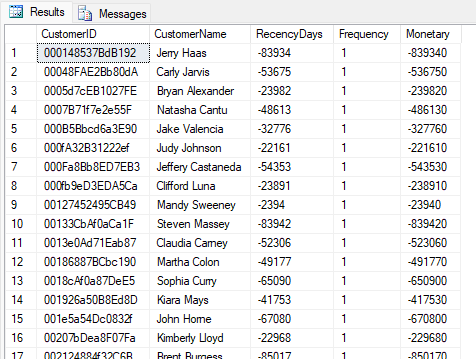
FrequencyCTE f ON r.CustomerID = f.CustomerID

JOIN

MonetaryCTE m ON r.CustomerID = m.CustomerID

JOIN

Customers c ON r.CustomerID = c.CustomerID;



-------------------------------------Segmentation Query Adapted------------------------------

-- Step 1: Calculate RFM Metrics using Common Table Expressions (CTEs)

WITH RecencyCTE AS (

SELECT

CustomerID,

DATEDIFF(DAY, SubscriptionDate, GETDATE()) AS RecencyDays

FROM Customers

),

FrequencyCTE AS (

SELECT

CustomerID,

1 AS Frequency -- Placeholder for frequency if not available

FROM Customers

),

MonetaryCTE AS (

SELECT

CustomerID,

DATEDIFF(DAY, SubscriptionDate, GETDATE()) \* 10 AS Monetary -- Proxy metric for engagement

FROM Customers

)

-- Step 2: Combine RFM Metrics into a Unified Table

SELECT

c.CustomerID,

CONCAT(c.FirstName, ' ', c.LastName) AS CustomerName,

r.RecencyDays,

f.Frequency,

m.Monetary,

-- Step 3: Segment Customers Based on RFM Metrics

CASE

WHEN r.RecencyDays < 30 AND m.Monetary > 500 THEN 'High-Value'

WHEN r.RecencyDays > 60 THEN 'At-Risk'

ELSE 'Other'

END AS Segment

FROM

RecencyCTE r

JOIN

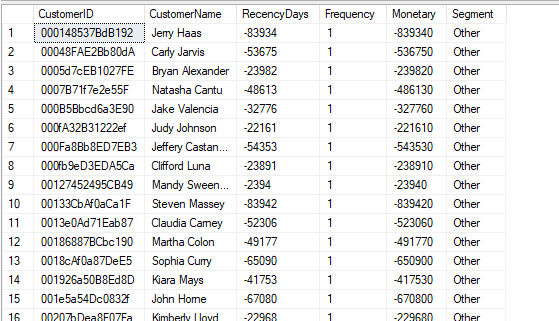
FrequencyCTE f ON r.CustomerID = f.CustomerID

JOIN

MonetaryCTE m ON r.CustomerID = m.CustomerID

JOIN

Customers c ON r.CustomerID = c.CustomerID;



------------------------Identify Customers with Multiple Contact Numbers---------------------

SELECT

Country,

COUNT(CustomerID) AS CustomersWithMultipleContacts

FROM

Customers

WHERE

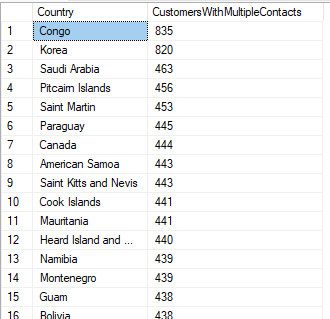
Phone1 IS NOT NULL AND Phone2 IS NOT NULL

GROUP BY

Country

ORDER BY

CustomersWithMultipleContacts DESC;



------------------------Domain Analysis for Customer Emails---------------

SELECT

RIGHT(Email, LEN(Email) - CHARINDEX('@', Email)) AS EmailDomain,

COUNT(CustomerID) AS CustomerCount

FROM

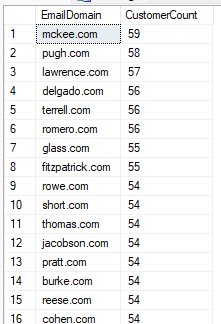
Customers

GROUP BY

RIGHT(Email, LEN(Email) - CHARINDEX('@', Email))

ORDER BY

CustomerCount DESC;



----------------------Analyze Subscription Patterns by Month-----------------

SELECT

FORMAT(SubscriptionDate, 'yyyy-MM') AS SubscriptionMonth,

COUNT(CustomerID) AS Subscriptions

FROM

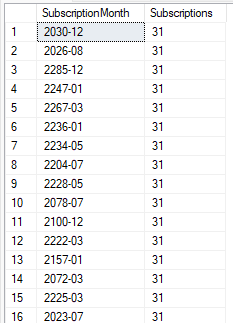
Customers

GROUP BY

FORMAT(SubscriptionDate, 'yyyy-MM')

ORDER BY

Subscriptions DESC;



-----------------------------Generate Full Customer Contact List-----------------

SELECT

CustomerID,

CONCAT(FirstName, ' ', LastName) AS FullName,

CONCAT(Phone1, ', ', Phone2) AS ContactNumbers,

Email,

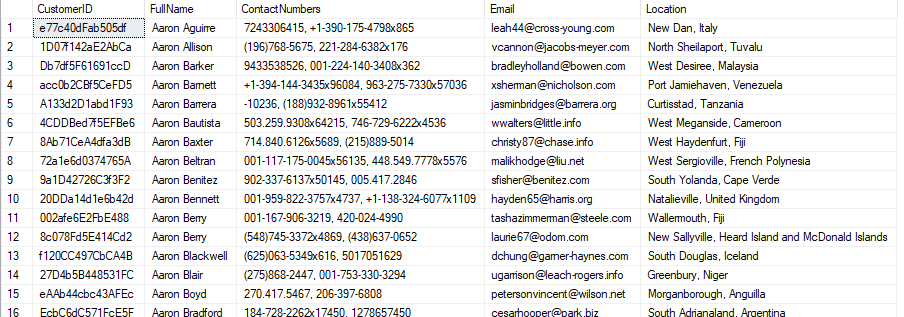
CONCAT(City, ', ', Country) AS Location

FROM

Customers

ORDER BY

FullName ASC;



---------------Country-wise Most Recent Subscriber--------------

WITH RankedSubscribers AS (

SELECT

CustomerID,

CONCAT(FirstName, ' ', LastName) AS FullName,

Country,

SubscriptionDate,

ROW\_NUMBER() OVER (PARTITION BY Country ORDER BY SubscriptionDate DESC) AS Rank

FROM

Customers

)

SELECT

CustomerID,

FullName,

Country,

SubscriptionDate

FROM

RankedSubscribers

WHERE

Rank = 1;



------Scenario: Find the latest subscription date for each country-------

---------Fast Query (Using JOIN and Aggregation)-----

SELECT

C.[ID],

C.[CustomerID],

C.[FirstName],

C.[LastName],

C.[Company],

C.[City],

C.[Country],

C.[Phone1],

C.[Phone2],

C.[Email],

C.[SubscriptionDate],

C.[Website]

FROM

[Health\_Food\_Flask].[dbo].[Customers] C

INNER JOIN (

SELECT

[Country],

MAX([SubscriptionDate]) AS MaxSubscriptionDate

FROM

[Health\_Food\_Flask].[dbo].[Customers]

GROUP BY

[Country]

) AS SubQuery ON

C.[Country] = SubQuery.[Country] AND

C.[SubscriptionDate] = SubQuery.MaxSubscriptionDate

ORDER BY

C.[Country];



---------Slow Query (Using HAVING with Correlated Subquery)-------------

SELECT

[ID],

[CustomerID],

[FirstName],

[LastName],

[Company],

[City],

[Country],

[Phone1],

[Phone2],

[Email],

[SubscriptionDate],

[Website]

FROM

[Health\_Food\_Flask].[dbo].[Customers] C1

GROUP BY

[ID],

[CustomerID],

[FirstName],

[LastName],

[Company],

[City],

[Country],

[Phone1],

[Phone2],

[Email],

[SubscriptionDate],

[Website]

HAVING

[SubscriptionDate] = (

SELECT

MAX([SubscriptionDate])

FROM

[Health\_Food\_Flask].[dbo].[Customers] C2

WHERE

C2.[Country] = C1.[Country]

)

ORDER BY

[Country];

