# KPI

## 1. Sales

-- 1. Sales

SELECT

SUM(Unit\_Price\*Sales\_Volume) AS 'Sales'

FROM Forte\_Grocery



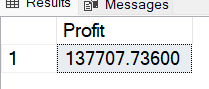
## 2. Profit

-- 2. Profit

SELECT

SUM(((Unit\_Price \* 0.4) \* Sales\_Volume)) AS 'Profit'

FROM Forte\_Grocery

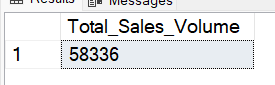


## 3. Total Sales Volume

-- 3. Total Sales Volume

SELECT SUM(Sales\_Volume) AS Total\_Sales\_Volume

FROM Forte\_Grocery



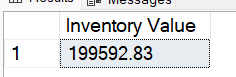
## 4. Inventory Value

-- 4. Inventory Value

SELECT

CAST(SUM((Unit\_Price\*0.6)\*Stock\_Quantity) AS DECIMAL (10,2)) AS 'Inventory Value'

FROM Forte\_Grocery;



## 5. Inventory Units

--5. Inventory Units

SELECT

SUM(Stock\_Quantity) AS 'Inventory Unit(s)'

FROM Forte\_Grocery



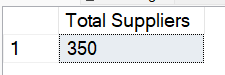
## 6. No. of Suppliers

-- 6. No. of Suppliers

SELECT

COUNT(DISTINCT Supplier\_Name) AS 'Total Suppliers'

FROM Forte\_Grocery



## 7. To get the Total Current Reorder Cost

-- 7. To get the Total Current Reorder Cost

WITH ReorderDetail AS (

SELECT

Product\_Name,

Reorder\_Quantity,

CAST(SUM((Unit\_Price\*0.6)\*Reorder\_Quantity) AS DECIMAL (10,2)) AS Reorder\_Cost\_Each\_Product,

Supplier\_Name

FROM Forte\_Grocery

WHERE Stock\_Quantity < Reorder\_Level AND Status <> 'Discontinued'

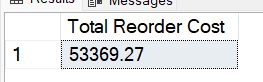
GROUP BY Product\_Name, Reorder\_Quantity, Supplier\_Name

)

SELECT

SUM(Reorder\_Cost\_Each\_Product) AS 'Total Reorder Cost'

FROM ReorderDetail



## 8. Number of Products to Reorder

-- 8. Number of Products to Reorder

WITH ReorderDetail AS (

SELECT

Product\_Name,

Reorder\_Quantity,

CAST(SUM(Unit\_Price\*Reorder\_Quantity) AS DECIMAL (10,2)) AS Reorder\_Cost\_Each\_Product,

Supplier\_Name

FROM Forte\_Grocery

WHERE Stock\_Quantity < Reorder\_Level AND Status <> 'Discontinued'

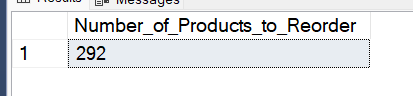
GROUP BY Product\_Name, Reorder\_Quantity, Supplier\_Name

)

SELECT

COUNT (Product\_Name) AS Number\_of\_Products\_to\_Reorder

FROM ReorderDetail



## 9. Total Stock Quantity vs Total Reorder Level

-- 9. Total Stock Quantity vs Total Reorder Level

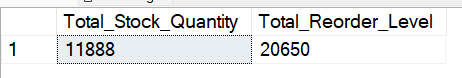
SELECT

SUM(Stock\_Quantity) AS Total\_Stock\_Quantity,

SUM(Reorder\_Level) AS Total\_Reorder\_Level

FROM Forte\_Grocery

WHERE Stock\_Quantity < Reorder\_Level AND Status <> 'Discontinued';



## 10. Average Delivery Time (Days) for All Suppliers

-- 10. Average Delivery Time (Days) for All Suppliers

WITH DeliveryData AS (

SELECT

Supplier\_Name,

SUM(Reorder\_Quantity) AS Total\_Orders,

ROUND(AVG(CASE

WHEN Date\_Received >= Last\_Order\_Date THEN DATEDIFF(DAY, Last\_Order\_Date, Date\_Received)

ELSE NULL

END), 2) AS Avg\_Delivery\_Time\_Days

FROM Forte\_Grocery

WHERE Last\_Order\_Date IS NOT NULL

AND Date\_Received IS NOT NULL

AND Date\_Received >= Last\_Order\_Date

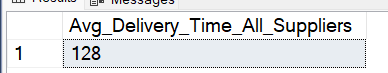
GROUP BY Supplier\_Name

)

SELECT

AVG(Avg\_Delivery\_Time\_Days) AS Avg\_Delivery\_Time\_All\_Suppliers

FROM DeliveryData;



# CHART

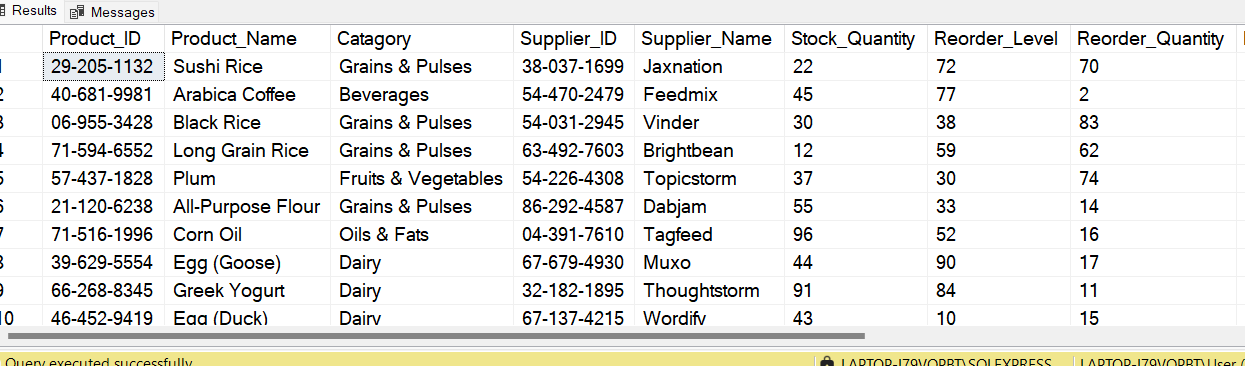
## Fixing Error (Correct Header Spelling and Replace Null Value)

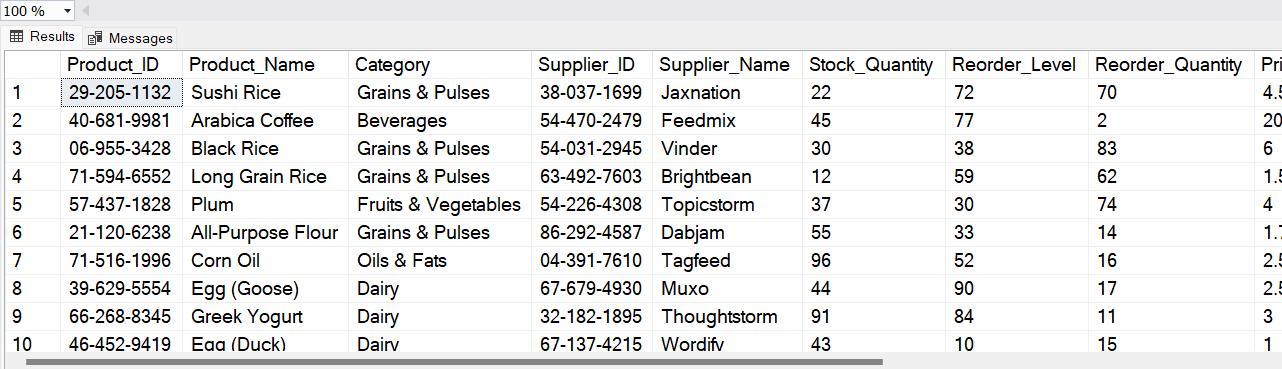
**In order to get the Profit Estimation by Category, I just realised there are some errors in the data and it needed to be fixed.**

1. I found out a column header is misspelled as ‘Catagory’ instead of ‘Category’

--Renaming column ‘Catagory’ to ‘Category’

sp\_rename 'Forte\_Grocery.Catagory', 'Category', 'COLUMN';





2. I found out that there is a NULL value in a cell under the ‘Category’ column. The ‘Product\_Name’ is ‘Cabbage’ so the NULL value should be ‘Fruits & Vegetables.

--When quering for then average price per category, I found a NULL value under category.

SELECT

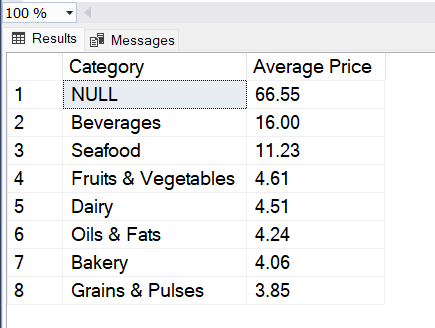
Category,

ROUND(AVG(Unit\_Price),2) AS 'Average Price'

FROM Forte\_Grocery

GROUP BY Category

ORDER BY 'Average Price' DESC;



--Finding which cell under the category column is null, with its corresponding product name.

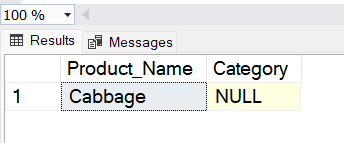
SELECT

Product\_Name,

Category

FROM Forte\_Grocery

WHERE Category IS NULL;



--Replacing a Null Value in Category to ‘Fruits & Vegetables

Update Forte\_Grocery

SET Category = 'Fruits and Vegetables'

WHERE Category IS NULL

## 1. Estimated Profit by Category

-- 1. Estimated Profit by Category

SELECT

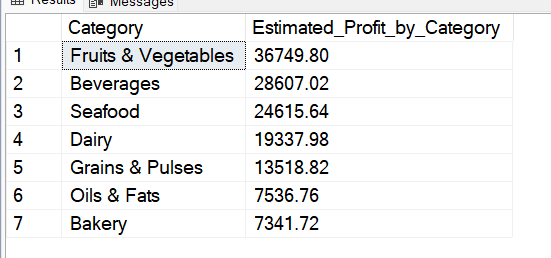
Category,

CAST(SUM((Unit\_Price - (Unit\_Price \* 0.6))\*Sales\_Volume) AS DECIMAL (10,2)) AS Estimated\_Profit\_by\_Category

FROM Forte\_Grocery

GROUP BY Category

ORDER BY Estimated\_Profit\_by\_Category DESC;



## 2. Percentage of Estimated Profit by Category

-- 2. Percentage of Profit Estimation by Category

WITH ProfitData AS (

SELECT

Category,

SUM((Unit\_Price - (Unit\_Price\*0.6))\*Sales\_Volume) AS Estimated\_Profit\_by\_Category

FROM Forte\_Grocery

GROUP BY Category

)

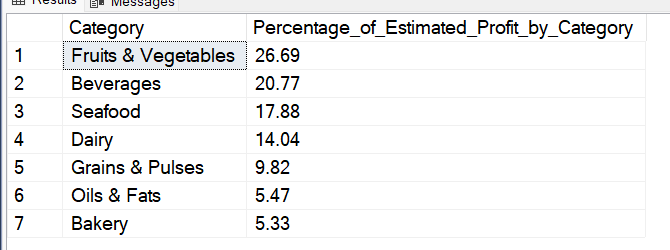
SELECT

Category,

CAST((Estimated\_Profit\_by\_Category / (SELECT SUM(Estimated\_Profit\_by\_Category) FROM ProfitData))\*100 AS DECIMAL (10,2)) AS Percentage\_of\_Estimated\_Profit\_by\_Category

FROM ProfitData

ORDER BY Percentage\_of\_Estimated\_Profit\_by\_Category DESC;



## 3. Sales Volume by Category

-- 3. Sales Volume by Category

SELECT

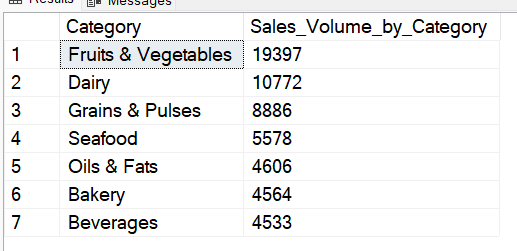
Category,

SUM(Sales\_Volume) AS Sales\_Volume\_by\_Category

FROM Forte\_Grocery

GROUP BY Category

ORDER BY Sales\_Volume\_By\_Category DESC;



## 4. Percentage of Sales Volume by Category

-- 4. Percentage of Sales Volume by Category

WITH SalesData AS(

SELECT

Category,

SUM(Sales\_Volume) AS Sales\_Volume\_by\_Category

FROM Forte\_Grocery

GROUP BY Category

)

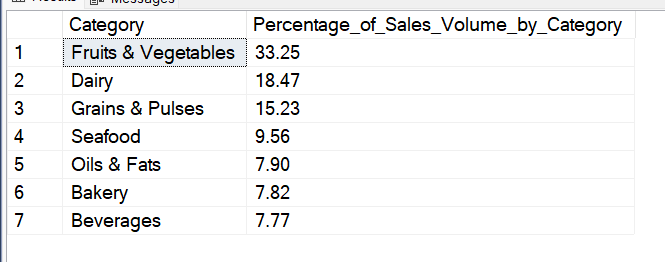
SELECT

Category,

CAST((Sales\_Volume\_by\_Category/cast((SELECT SUM(Sales\_Volume\_by\_Category) FROM SalesData) AS DECIMAL (10,2)))\*100 AS DECIMAl (10,2)) AS Percentage\_of\_Sales\_Volume\_by\_Category

FROM SalesData

ORDER BY Percentage\_of\_Sales\_Volume\_by\_Category DESC;



## 5. Average Price per Category

-- 5. Average Price per Category

SELECT

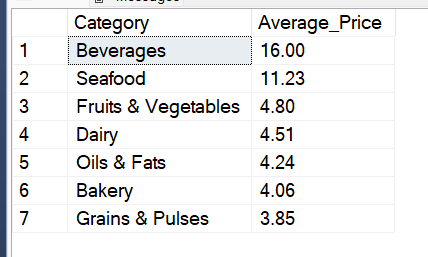
Category,

ROUND(AVG(Unit\_Price), 2) AS Average\_Price

FROM Forte\_Grocery

GROUP BY Category

ORDER BY Average\_Price DESC;



## 6. Top 20 Profitable Products

-- 6. Top 20 Profitable Products

SELECT

TOP 20 Product\_Name,

Category,

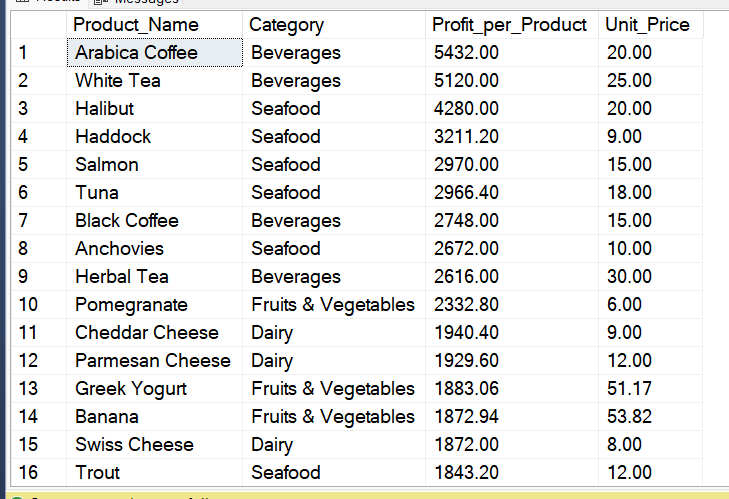
CAST(SUM((Unit\_Price - (Unit\_Price \* 0.6))\*Sales\_Volume) AS DECIMAL (10,2)) AS Profit\_per\_Product,

Unit\_Price

FROM Forte\_Grocery

GROUP BY Product\_Name, Category, Unit\_Price

ORDER BY Profit\_per\_Product DESC;



## 7. Top 20 Selling Products (Highest Sale Volume)

-- 7. Top 20 Selling Products (Highest Sale Volume)

SELECT

TOP 20 Product\_Name,

Category,

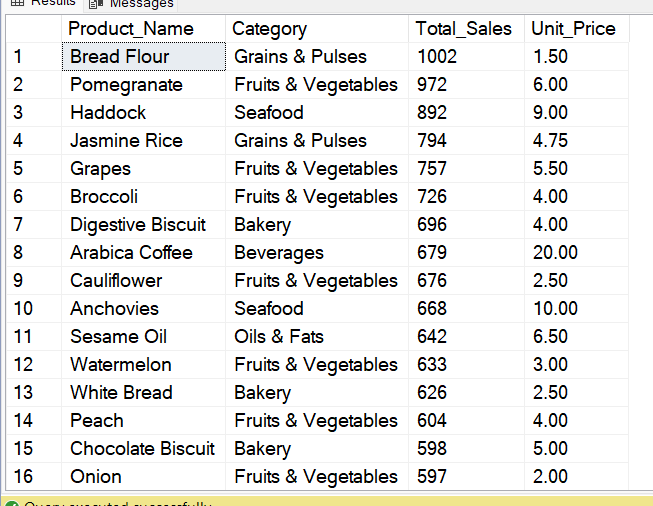
SUM(Sales\_Volume) AS Total\_Sales,

Unit\_Price

FROM Forte\_Grocery

GROUP BY Product\_Name, Category, Unit\_Price

ORDER BY Total\_Sales DESC;



## 8. Stock Levels & Reorder Alerts

-- 8. Stock Levels & Reorder Alerts (Products Needing Replenishment)

SELECT

Product\_Name,

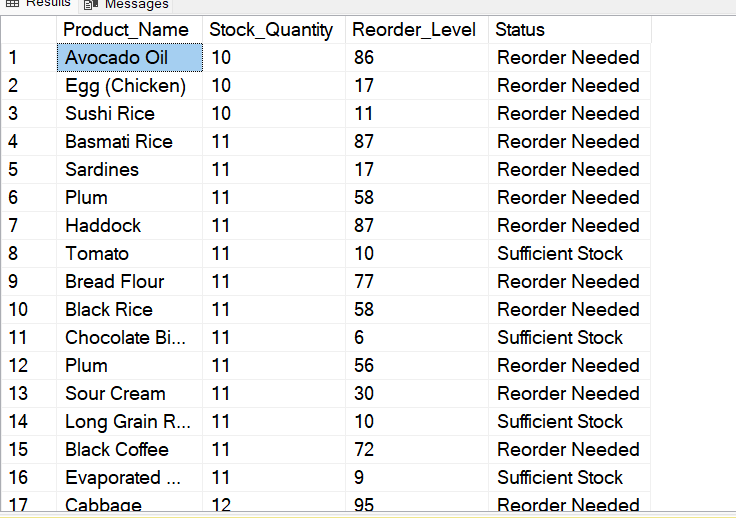
Stock\_Quantity,

Reorder\_Level,

CASE WHEN Stock\_Quantity <= Reorder\_Level THEN 'Reorder Needed' ELSE 'Sufficient Stock' END AS Status

FROM Forte\_Grocery

ORDER BY Stock\_Quantity ASC;



## 9. Reorder Quantity and Reorder Cost for Each Product

-- 9. Reorder Quantity and Reorder Cost for Each Product

SELECT

Product\_Name,

Reorder\_Quantity,

CAST(SUM((Unit\_Price\*0.6)\*Reorder\_Quantity) AS DECIMAL (10,2)) AS Reorder\_Cost\_Each\_Product,

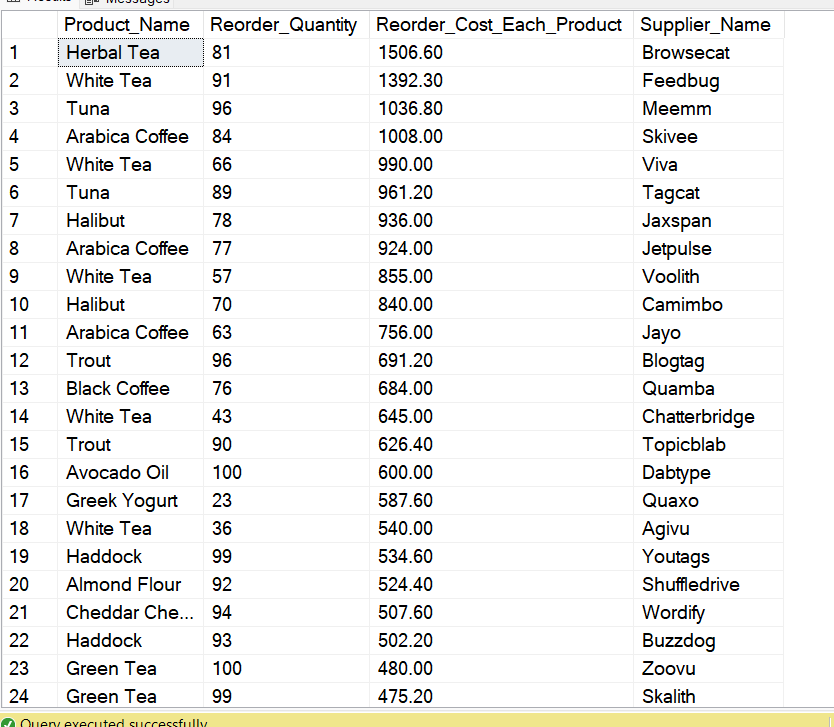
Supplier\_Name

FROM Forte\_Grocery

WHERE Stock\_Quantity < Reorder\_Level AND Status <> 'Discontinued'

GROUP BY Product\_Name, Reorder\_Quantity, Supplier\_Name

ORDER BY Reorder\_Cost\_Each\_Product DESC;



--To get the Total Current Reorder Cost

WITH ReorderDetail AS (

SELECT

Product\_Name,

Reorder\_Quantity,

CAST(SUM((Unit\_Price\*0.6)\*Reorder\_Quantity) AS DECIMAL (10,2)) AS Reorder\_Cost\_Each\_Product,

Supplier\_Name

FROM Forte\_Grocery

WHERE Stock\_Quantity < Reorder\_Level AND Status <> 'Discontinued'

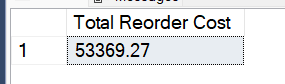
GROUP BY Product\_Name, Reorder\_Quantity, Supplier\_Name

)

SELECT

SUM(Reorder\_Cost\_Each\_Product) AS 'Total Reorder Cost'

FROM ReorderDetail



## 10. Total Stock Quantity, Total Reorder Level, Total Reorder Quantity, Estimated Stock After

-- 10. Total Stock Quantity, Total Reorder Level, Total Reorder Quantity, Estimated Stock After Reorder

SELECT

SUM(Stock\_Quantity) AS Total\_Stock\_Quantity,

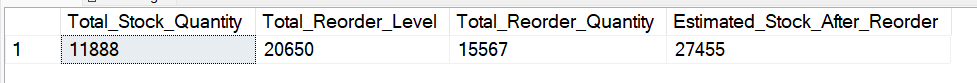
SUM(Reorder\_Level) AS Total\_Reorder\_Level,

SUM(Reorder\_Quantity) AS Total\_Reorder\_Quantity,

(SUM(Stock\_Quantity) + SUM(Reorder\_Quantity)) AS Estimated\_Stock\_After\_Reorder

FROM Forte\_Grocery

WHERE Stock\_Quantity < Reorder\_Level AND Status <> 'Discontinued';



## 11. Average Delivery Time (Days) by Supplier

-- 11. Average Delivery Time (Days) by Supplier

SELECT

Supplier\_Name,

SUM(Reorder\_Quantity) AS Total\_Orders,

ROUND(AVG(CASE

WHEN Date\_Received >= Last\_Order\_Date THEN DATEDIFF(DAY, Last\_Order\_Date, Date\_Received)

ELSE NULL

END), 2) AS Avg\_Delivery\_Time\_Days

FROM Forte\_Grocery

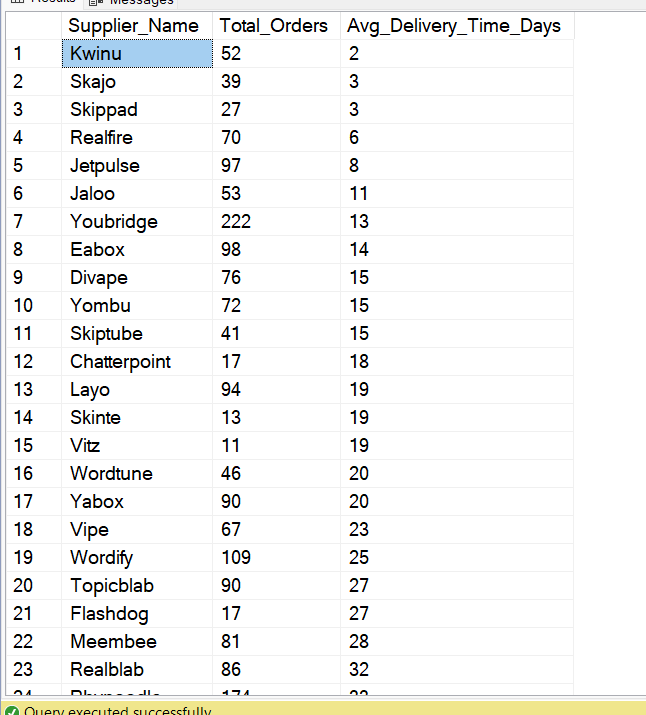
WHERE Last\_Order\_Date IS NOT NULL

AND Date\_Received IS NOT NULL

AND Date\_Received >= Last\_Order\_Date

GROUP BY Supplier\_Name

ORDER BY Avg\_Delivery\_Time\_Days ASC;



## 12. Average Delivery Time (Days) and Unit Cost for Particular Product by Supplier

-- 12. Average Delivery Time (Days) and Unit Cost for Particular Product by Supplier

SELECT

Product\_Name,

Supplier\_Name,

CAST((Unit\_Price\*0.6) AS DECIMAL(10,2)) AS Unit\_Cost,

ROUND(AVG(DATEDIFF(DAY, Last\_Order\_Date, Date\_Received)), 2) AS Avg\_Delivery\_Time\_Days,

SUM(Reorder\_Quantity) AS Total\_Orders

FROM Forte\_Grocery

WHERE Last\_Order\_Date IS NOT NULL

AND Date\_Received IS NOT NULL

AND Date\_Received >= Last\_Order\_Date

GROUP BY Product\_Name, Supplier\_Name, Unit\_Price

ORDER BY Product\_Name, Avg\_Delivery\_Time\_Days ASC;

# DAX FUNCTION

## Total Sales

Total Sales = SUMX (Forte\_Grocery, Forte\_Grocery[Unit\_Price]\*Forte\_Grocery[Sales\_Volume])

## Total Profit

Total Profit = SUMX (Forte\_Grocery, (Forte\_Grocery[Unit\_Price]\*0.4)\*Forte\_Grocery[Sales\_Volume])

## Total Reorder

Total Reorders =

CALCULATE(

SUM('Forte\_Grocery'[Reorder\_Quantity]),

FILTER(

'Forte\_Grocery',

NOT(ISBLANK('Forte\_Grocery'[Last\_Order\_Date])) &&

NOT(ISBLANK('Forte\_Grocery'[Date\_Received])) &&

'Forte\_Grocery'[Date\_Received] >= 'Forte\_Grocery'[Last\_Order\_Date]

)

)

## Total Reorder Cost

Total Reorder Cost =

CALCULATE(

SUMX(

'Forte\_Grocery',

('Forte\_Grocery'[Unit\_Price] \* 0.6) \* 'Forte\_Grocery'[Reorder\_Quantity]

),

'Forte\_Grocery'[Stock\_Quantity] < 'Forte\_Grocery'[Reorder\_Level],

'Forte\_Grocery'[Status] <> "Discontinued"

)

## Number of Product to Reorder

Number of Products to Reorder =

CALCULATE(

DISTINCTCOUNT('Forte\_Grocery'[Product\_Name]),

'Forte\_Grocery'[Stock\_Quantity] < 'Forte\_Grocery'[Reorder\_Level],

'Forte\_Grocery'[Status] <> "Discontinued"

)

## Inventory Value

Inventory Value = SUMX(Forte\_Grocery, (Forte\_Grocery[Unit\_Price]\*0.6)\*Forte\_Grocery[Stock\_Quantity])

## Average Delivery Time (Days)

Avg Delivery Time (Days) =

VAR AvgDays =

CALCULATE(

AVERAGEX(

FILTER(

'Forte\_Grocery',

'Forte\_Grocery'[Date\_Received] >= 'Forte\_Grocery'[Last\_Order\_Date]

),

DATEDIFF('Forte\_Grocery'[Last\_Order\_Date], 'Forte\_Grocery'[Date\_Received], DAY)

)

)

RETURN IF(ISBLANK(AvgDays), 0, AvgDays)

## Delivery Days

Delivery\_Days =

IF(

'Forte\_Grocery'[Date\_Received] >= 'Forte\_Grocery'[Last\_Order\_Date],

DATEDIFF('Forte\_Grocery'[Last\_Order\_Date], 'Forte\_Grocery'[Date\_Received], DAY),

BLANK())

## Search Filter

SearchFilter =

'Forte\_Grocery'[Product\_Name] & " " &

'Forte\_Grocery'[Supplier\_Name] & " " &

'Forte\_Grocery'[Status]

## Stock Level Alert

Stock Level Alert =

IF(

Forte\_Grocery[Stock\_Quantity]<Forte\_Grocery[Reorder\_Level],

"Reorder Needed",

"Sufficient Stock"

)