**MS SQL QUERIES**

**COFFEE SHOP SALES PROJECT**

**CONVERT DATE (transaction\_date) COLUMN TO PROPER DATE FORMAT**

UPDATE coffee\_shop\_sales

SET transaction\_date = STR\_TO\_DATE(transaction\_date, '%d-%m-%Y');

**ALTER DATE (transaction\_date) COLUMN TO DATE DATA TYPE**

ALTER TABLE coffee\_shop\_sales

MODIFY COLUMN transaction\_date DATE;

**CONVERT TIME (transaction\_time) COLUMN TO PROPER DATE FORMAT**

UPDATE coffee\_shop\_sales

SET transaction\_time = STR\_TO\_DATE(transaction\_time, '%H:%i:%s');

**ALTER TIME (transaction\_time) COLUMN TO DATE DATA TYPE**

ALTER TABLE coffee\_shop\_sales

MODIFY COLUMN transaction\_time TIME;

**DATA TYPES OF DIFFERENT COLUMNS**

DESCRIBE coffee\_shop\_sales;

****

**CHANGE COLUMN NAME `ï»¿transaction\_id` to transaction\_id**

ALTER TABLE coffee\_shop\_sales

CHANGE COLUMN `ï»¿transaction\_id` transaction\_id INT;

**TOTAL SALES by Month**

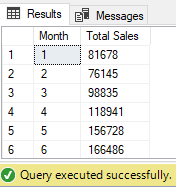
-- Total Sales by Month

SELECT DISTINCT MONTH(transaction\_date) AS "Month", ROUND(SUM(unit\_price \* transaction\_qty),0) AS "Total Sales"

FROM [dbo].[Coffee\_Shop\_Sales]

GROUP BY MONTH(transaction\_date)

ORDER BY "Month" ASC



**TOTAL SALES KPI - MOM DIFFERENCE AND MOM GROWTH**

WITH MonthlySales AS (

SELECT

MONTH(transaction\_date) AS "Month",

SUM(unit\_price \* transaction\_qty) AS "Total Sales"

FROM

[dbo].[Coffee\_Shop\_Sales]

GROUP BY

MONTH(transaction\_date)

)

, SalesWithLag AS (

SELECT

"Month",

"Total Sales",

LAG("Total Sales", 1) OVER (ORDER BY "Month") AS PreviousMonthSales -- Obtains sales from the previous month

FROM

MonthlySales

)

SELECT

"Month",

"Total Sales",

PreviousMonthSales,

"Total Sales" - PreviousMonthSales AS SalesDifference,

CASE

WHEN PreviousMonthSales IS NULL THEN NULL

ELSE ROUND((("Total Sales" - PreviousMonthSales) / PreviousMonthSales) \* 100, 2)

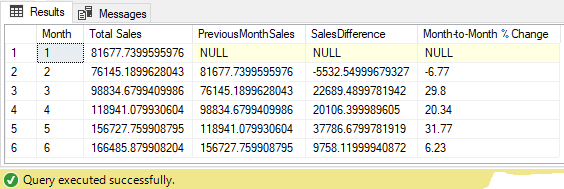
END AS "Month-to-Month % Change"

FROM

SalesWithLag

ORDER BY

"Month";



**Explanation**

**SELECT clause:**

* MONTH(transaction\_date) AS month: Extracts the month from the transaction\_date column and renames it as month.
* ROUND(SUM(unit\_price \* transaction\_qty)) AS total\_sales: Calculates the total sales by multiplying unit\_price and transaction\_qty, then sums the result for each month. The ROUND function rounds the result to the nearest integer.
* (SUM(unit\_price \* transaction\_qty) - LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date))) / LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date)) \* 100 AS mom\_increase\_percentage with the functions used:
  + SUM(unit\_price \* transaction\_qty): This calculates the total sales for the current month. It multiplies the unit\_price by the transaction\_qty for each transaction and then sums up these values.
  + LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date)): This function retrieves the value of the total sales for the previous month. It uses the LAG window function to get the value of the SUM(unit\_price \* transaction\_qty) from the previous row (previous month) ordered by the transaction\_date.
  + (SUM(unit\_price \* transaction\_qty) - LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date))): This part calculates the difference between the total sales of the current month and the total sales of the previous month.
  + LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date)): This function retrieves the value of the total sales for the previous month again. It's used in the denominator to calculate the percentage increase.
  + (SUM(unit\_price \* transaction\_qty) - LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date))) / LAG(SUM(unit\_price \* transaction\_qty), 1) OVER (ORDER BY MONTH(transaction\_date)): This calculates the ratio of the difference in sales between the current and previous months to the total sales of the previous month. It represents the percentage increase or decrease in sales compared to the previous month.
  + 100: This part multiplies the ratio by 100 to convert it to a percentage.
* FROM clause:

coffee\_shop\_sales: Specifies the table from which data is being selected.

* WHERE clause:

MONTH(transaction\_date) IN (4, 5): Filters the data to include only transactions from April and May.

* GROUP BY clause:

MONTH(transaction\_date): Groups the results by month.

* ORDER BY clause:

MONTH(transaction\_date): Orders the results by month.

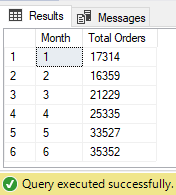
**TOTAL ORDERS by Month**

SELECT DISTINCT MONTH(transaction\_date) AS "Month", COUNT(transaction\_id) AS "Total Orders"

FROM [dbo].[Coffee\_Shop\_Sales]

GROUP BY MONTH(transaction\_date)

ORDER BY "Month" ASC



**Month-to-month increase/decrease in orders**

WITH MonthlyOrders AS (

SELECT

MONTH(transaction\_date) AS "Month",

COUNT(transaction\_id) AS "Total Orders"

FROM

[dbo].[Coffee\_Shop\_Sales]

GROUP BY

MONTH(transaction\_date)

)

, OrdersWithLag AS (

SELECT

"Month",

"Total Orders",

LAG("Total Orders", 1) OVER (ORDER BY "Month") AS "PreviousMonthOrders"

FROM

MonthlyOrders

)

SELECT

"Month",

"Total Orders",

"PreviousMonthOrders",

"Total Orders" - PreviousMonthOrders AS OrdersDifference,

CASE

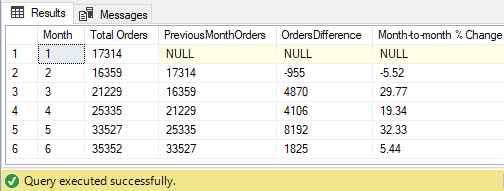
WHEN "PreviousMonthOrders" IS NULL THEN NULL

ELSE ROUND((CAST("Total Orders" AS FLOAT) - CAST("PreviousMonthOrders" AS FLOAT)) / CAST("PreviousMonthOrders" AS FLOAT) \* 100, 2)

END AS "Month-to-month % Change"

FROM OrdersWithLag

ORDER BY "Month";



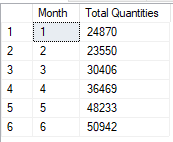
**TOTAL QUANTITY SOLD by Month**

SELECT DISTINCT MONTH(transaction\_date) AS "Month", SUM(transaction\_qty) AS "Total Quantities"

FROM [dbo].[Coffee\_Shop\_Sales]

GROUP BY MONTH(transaction\_date)

ORDER BY "Month" ASC



**Month-to-month increase/decrease in quantities sold**

WITH MonthlyOrders AS (

SELECT

MONTH(transaction\_date) AS "Month",

SUM(transaction\_qty) AS "Total Quantities"

FROM

[dbo].[Coffee\_Shop\_Sales]

GROUP BY

MONTH(transaction\_date)

)

, OrdersWithLag AS (

SELECT

"Month",

"Total Quantities",

LAG("Total Quantities", 1) OVER (ORDER BY "Month") AS "PreviousMonthQuantities"

FROM

MonthlyOrders

)

SELECT

"Month",

"Total Quantities",

"PreviousMonthQuantities",

"Total Quantities" - PreviousMonthQuantities AS QuantitiesDifference,

CASE

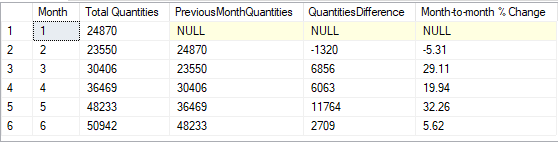
WHEN "PreviousMonthQuantities" IS NULL THEN NULL

ELSE ROUND((CAST("Total Quantities" AS FLOAT) - CAST("PreviousMonthQuantities" AS FLOAT)) / CAST("PreviousMonthQuantities" AS FLOAT) \* 100, 2)

END AS "Month-to-month % Change"

FROM OrdersWithLag

ORDER BY "Month";



**CALENDAR TABLE – DAILY SALES, QUANTITY and TOTAL ORDERS**

SELECT

SUM(unit\_price \* transaction\_qty) AS total\_sales,

SUM(transaction\_qty) AS total\_quantity\_sold,

COUNT(transaction\_id) AS total\_orders

FROM

coffee\_shop\_sales

WHERE

transaction\_date = '2023-05-18'; --For 18 May 2023



***If you want to get exact Rounded off values then use below query to get the result:***

SELECT

CONCAT(ROUND(SUM(unit\_price \* transaction\_qty) / 1000, 1),'K') AS total\_sales,

CONCAT(ROUND(COUNT(transaction\_id) / 1000, 1),'K') AS total\_orders,

CONCAT(ROUND(SUM(transaction\_qty) / 1000, 1),'K') AS total\_quantity\_sold

FROM

coffee\_shop\_sales

WHERE

transaction\_date = '2023-05-18'; --For 18 May 2023



# Sales analysis by weekday and weekend

SELECT

CASE

WHEN DATEPART(WEEKDAY, transaction\_date) IN (1,7) THEN 'Weekend'

ELSE 'Weekday'

END AS "Day Type",

SUM(transaction\_qty \* unit\_price) AS "Total Sales"

FROM [dbo].[Coffee\_Shop\_Sales]

WHERE MONTH(transaction\_date) = 5

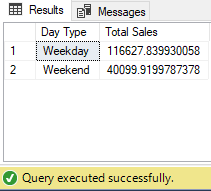
GROUP BY

CASE

WHEN DATEPART(WEEKDAY, transaction\_date) IN (1,7) THEN 'Weekend'

ELSE 'Weekday'

END



# SALES TREND OVER PERIOD

SELECT AVG(total\_sales) AS average\_sales

FROM (

SELECT

SUM(unit\_price \* transaction\_qty) AS total\_sales

FROM

coffee\_shop\_sales

WHERE

MONTH(transaction\_date) = 5 -- Filter for May

GROUP BY

transaction\_date

) AS internal\_query;

***Query Explanation:***

* This inner subquery calculates the total sales (unit\_price \* transaction\_qty) for each date in May. It filters the data to include only transactions that occurred in May by using the MONTH() function to extract the month from the transaction\_date column and filtering for May (month number 5).
* The GROUP BY clause groups the data by transaction\_date, ensuring that the total sales are aggregated for each individual date in May.
* The outer query calculates the average of the total sales over all dates in May. It references the result of the inner subquery as a derived table named internal\_query.
* The AVG() function calculates the average of the total\_sales column from the derived table, giving us the average sales for May.



# DAILY SALES FOR MONTH SELECTED

SELECT

DAY(transaction\_date) AS day\_of\_month,

ROUND(SUM(unit\_price \* transaction\_qty),1) AS total\_sales

FROM

coffee\_shop\_sales

WHERE

MONTH(transaction\_date) = 5 -- Filter for May

GROUP BY

DAY(transaction\_date)

ORDER BY

DAY(transaction\_date);

***COMPARING DAILY SALES WITH AVERAGE SALES – IF GREATER THAN “ABOVE AVERAGE” and LESSER THAN “BELOW AVERAGE”***

SELECT

day\_of\_month,

CASE

WHEN total\_sales > avg\_sales THEN 'Above Average'

WHEN total\_sales < avg\_sales THEN 'Below Average'

ELSE 'Average'

END AS sales\_status,

total\_sales

FROM (

SELECT

DAY(transaction\_date) AS day\_of\_month,

SUM(unit\_price \* transaction\_qty) AS total\_sales,

AVG(SUM(unit\_price \* transaction\_qty)) OVER () AS avg\_sales

FROM

coffee\_shop\_sales

WHERE

MONTH(transaction\_date) = 5 -- Filter for May

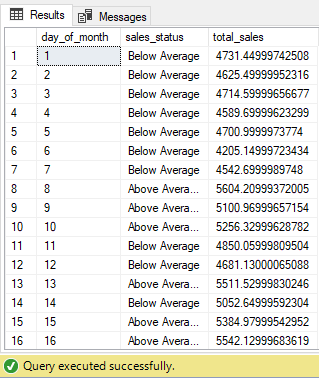
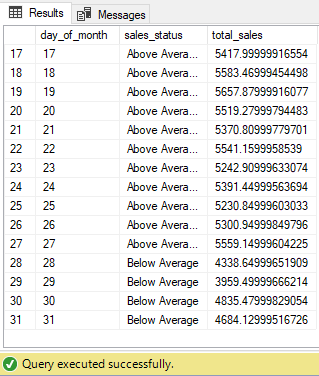
GROUP BY

DAY(transaction\_date)

) AS sales\_data

ORDER BY

day\_of\_month;

**Sales value by store location for a particular month**

SELECT

DISTINCT(store\_location) AS "Store Location",

CONCAT(ROUND(SUM(transaction\_qty \* unit\_price)/1000, 2), 'K') AS "Total Sales"

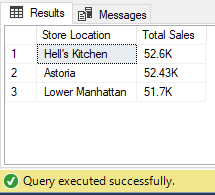
FROM

[dbo].[Coffee\_Shop\_Sales]

WHERE MONTH(transaction\_date) IN (5)

GROUP BY store\_location

ORDER BY "Total Sales" DESC



**Sales by product category**

SELECT

product\_category,

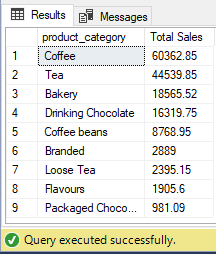
ROUND(SUM(transaction\_qty \* unit\_price),2) AS "Total Sales"

FROM [dbo].[Coffee\_Shop\_Sales]

WHERE MONTH(transaction\_date) IN (5)

GROUP BY product\_category

ORDER BY "Total Sales" DESC



**SALES BY PRODUCTS (TOP 10)**

SELECT

product\_type,

ROUND(SUM(unit\_price \* transaction\_qty),1) as Total\_Sales

FROM coffee\_shop\_sales

WHERE

MONTH(transaction\_date) = 5

GROUP BY product\_type

ORDER BY SUM(unit\_price \* transaction\_qty) DESC

LIMIT 10



**SALES BY DAY | HOUR**

SELECT

ROUND(SUM(unit\_price \* transaction\_qty)) AS Total\_Sales,

SUM(transaction\_qty) AS Total\_Quantity,

COUNT(\*) AS Total\_Orders

FROM

coffee\_shop\_sales

WHERE

DAYOFWEEK(transaction\_date) = 3 -- Filter for Tuesday (1 is Sunday, 2 is Monday, ..., 7 is Saturday)

AND HOUR(transaction\_time) = 8 -- Filter for hour number 8

AND MONTH(transaction\_date) = 5; -- Filter for May (month number 5)



# Sales value by day of the week

SELECT

CASE

WHEN DATEPART(WEEKDAY, transaction\_date) = 1 THEN 'Sunday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 2 THEN 'Monday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 3 THEN 'Tuesday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 4 THEN 'Wednesday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 5 THEN 'Thursday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 6 THEN 'Friday'

ELSE 'Saturday'

END AS "Day of the week",

ROUND(SUM(transaction\_qty \*unit\_price), 0) AS "Total Sales"

FROM Coffee\_Shop\_Sales

WHERE MONTH(transaction\_date) IN (5)

GROUP BY

CASE

WHEN DATEPART(WEEKDAY, transaction\_date) = 1 THEN 'Sunday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 2 THEN 'Monday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 3 THEN 'Tuesday'

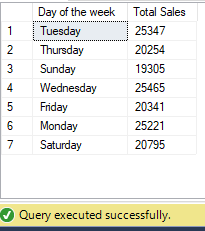
WHEN DATEPART(WEEKDAY, transaction\_date) = 4 THEN 'Wednesday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 5 THEN 'Thursday'

WHEN DATEPART(WEEKDAY, transaction\_date) = 6 THEN 'Friday'

ELSE 'Saturday'

END



# Sales values by hours

SELECT

DISTINCT DATEPART(hour, transaction\_time) AS "Day",

ROUND(SUM(transaction\_qty \* unit\_price), 2) AS "Total Sales"

FROM Coffee\_Shop\_Sales

WHERE MONTH(transaction\_date) IN (5)

GROUP BY DATEPART(hour, transaction\_time)

ORDER BY "Day"

