SELECT \* FROM coffee\_sales;

**--1.CREATING KPIS**

**/\*1.1.Total Sales Analysis\*/**

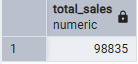
**--1.1.1Total Sales for each respective month**

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

--SELECT CONCAT(ROUND(SUM(unit\_price \* transaction\_qty)))/1000 , "K" AS Total\_Sales-- to adjust p.ex 98.835K

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 3; -- March Month



**--1.1.2 Determinate the month-on-month increase or decrease in sales**

**--1.1.3 Calculate the difference in sales between the selected month and the previous month**

WITH monthly\_sales AS (

SELECT

EXTRACT(MONTH FROM transaction\_date) AS month, -- number of the month

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

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) IN (4, 5) -- for months of April and May

GROUP BY EXTRACT(MONTH FROM transaction\_date)

)

SELECT

month,

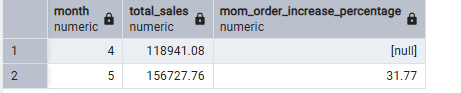
Total\_sales,

ROUND(((total\_sales - LAG(total\_sales) OVER (ORDER BY month))

/ LAG(total\_sales) OVER (ORDER BY month)::NUMERIC) \* 100,2) AS mom\_order\_increase\_percentage

FROM monthly\_sales

ORDER BY month;



-- Note: Month sales division% = CM-PM/PM\*100

**/\*1.2. Total orders analysis\*/**

**--1.2.1 Calculate total number of orders for each respective month**

SELECT COUNT(transaction\_id) AS Total\_orders

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5; -- May Month



**--1.2.2 Determinate the mom increase or decrease in the number of orders**

**--1.2.3 Calculate the difference in the number of orders between the selected month and previous month**

WITH monthly\_orders AS (

SELECT

EXTRACT(MONTH FROM transaction\_date) AS month, -- number of the months

COUNT(transaction\_id) AS total\_orders -- total orders

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) IN (4, 5) -- april and may

GROUP BY EXTRACT(MONTH FROM transaction\_date)

SELECT

month,

total\_orders,

ROUND(((total\_orders - LAG(total\_orders) OVER (ORDER BY month))

/ LAG(total\_orders) OVER (ORDER BY month)::NUMERIC) \* 100,2) AS mom\_order\_increase\_percentage

FROM monthly\_orders

ORDER BY month;

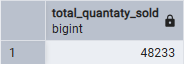
**/\*1.3.Total Quantity Sold Analysis\*/**

**--1.3.1 Calculate the total quantity sold for each respective month**

SELECT SUM(transaction\_qty) AS Total\_Quantaty\_Sold

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5; -- May Month



**--1.3.2 Determinate the month-on-month increase or decrease in the total quantity sold**

**--1.3.3 Calculate the difference in the total sold between the selected month and the previous month**

WITH monthly\_orders AS (

SELECT

EXTRACT(MONTH FROM transaction\_date) AS month, -- number of the months

SUM(transaction\_qty) AS total\_quantity\_sold -- total quantity sold

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) IN (4, 5) -- april and may

GROUP BY EXTRACT(MONTH FROM transaction\_date)

)

SELECT

month,

total\_quantity\_sold,

ROUND(((total\_quantity\_sold - LAG(total\_quantity\_sold) OVER (ORDER BY month))

/ LAG(total\_quantity\_sold) OVER (ORDER BY month)::NUMERIC) \* 100,2) AS mom\_quantity\_increase\_percentage

FROM monthly\_orders

ORDER BY month;

**--2.CHARTS REQUIREMENTS**

**/\*2.1.Calendar Heat Map**

💡To do in PBI:

1.1 Implement a calendar heat map that dynamically adjusts based on the select month from a slicer

1.2 Each day on the calendar will be color-coded to represent sales volume, with darker shades indicating higher sales

1.3 Implemt tooltips to display detailed metrics (sales,orders,quantity) qhen hovering over a specific day

**Creating the three metrics\*/**

SELECT

CONCAT(ROUND(SUM(unit\_price \* transaction\_qty)/1000.0, 1), 'K') AS Total\_Sales, -- 1000.0 forces the division to be decimal, avoiding truncation due to integer division.

CONCAT(ROUND(SUM(transaction\_qty)/1000.0,1), 'K') AS Total\_Qty\_Sold,

CONCAT(ROUND(COUNT(transaction\_id)/1000.0,1), 'K') AS Total\_Orders

FROM coffee\_sales

WHERE transaction\_date = '2023-03-27';



**/\*2.2.Sales Analysis by Weekdays and Weekends**

**2.2.1 Segment sales data into weekdays and weekend to analyze performance variations\*/**

--Note:EXTRACT(DOW FROM date) → retorns 0=Sunday, 1=Monday, ..., 6=Saturday in PostgreSQL

WITH sales\_with\_day\_type AS (

SELECT

CASE

WHEN EXTRACT(DOW FROM transaction\_date) IN (0,6) THEN 'Weekends'

ELSE 'Weekdays'

END AS day\_type,

unit\_price \* transaction\_qty AS sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 2 -- Feb

)

SELECT

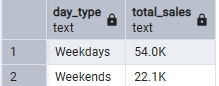
day\_type,

CONCAT(ROUND(SUM(sales)/1000.0, 1), 'K') AS total\_sales

FROM sales\_with\_day\_type

GROUP BY day\_type

ORDER BY day\_type;



**--2.3. Sales analysis by Store Location**

**--2.3.1 Visualize sales data by different store locations**

SELECT

store\_location,

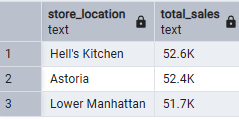
CONCAT(ROUND(SUM(unit\_price \* transaction\_qty)/1000.0, 1), 'K') AS total\_sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5 -- May

GROUP BY store\_location

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



**--2.3.2 Calculate increase or decrease between each month and percentage - sales, quantity and orders**

WITH monthly\_by\_store AS (

SELECT

store\_location,

EXTRACT(MONTH FROM transaction\_date) AS month,

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

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

COUNT(transaction\_id) AS total\_orders

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) IN (4, 5) -- April and May

GROUP BY store\_location, EXTRACT(MONTH FROM transaction\_date)

)

SELECT

store\_location,

month,

total\_sales,

total\_quantity\_sold,

total\_orders,

ROUND(

((total\_sales - LAG(total\_sales) OVER (PARTITION BY store\_location ORDER BY month))

/ LAG(total\_sales) OVER (PARTITION BY store\_location ORDER BY month)::NUMERIC) \* 100, 2

) AS mom\_sales\_percentage,

ROUND(

((total\_quantity\_sold - LAG(total\_quantity\_sold) OVER (PARTITION BY store\_location ORDER BY month))

/ LAG(total\_quantity\_sold) OVER (PARTITION BY store\_location ORDER BY month)::NUMERIC) \* 100, 2

) AS mom\_qty\_percentage,

ROUND(

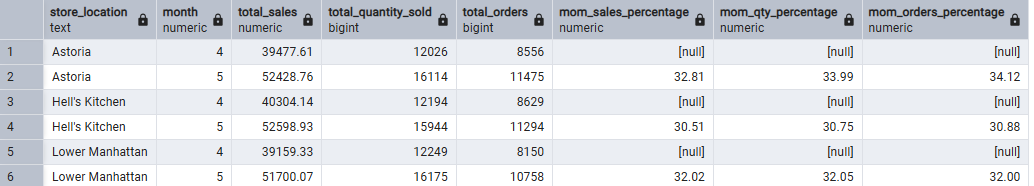
((total\_orders - LAG(total\_orders) OVER (PARTITION BY store\_location ORDER BY month))

/ LAG(total\_orders) OVER (PARTITION BY store\_location ORDER BY month)::NUMERIC) \* 100, 2

) AS mom\_orders\_percentage

FROM monthly\_by\_store

ORDER BY store\_location, month;



**/\* 2.4.Average sales**

**2.4.1 Display daily sales for the selected month with a line chart (avg)\*/**

**--first, only calculating total AVG**

SELECT

CONCAT(ROUND(AVG(total\_sales)/1000.0,1), 'K') AS Avg\_sales

FROM

(

SELECT

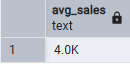
SUM(transaction\_qty \* unit\_price) AS total\_sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 4 -- april

GROUP BY transaction\_date

) AS Internal\_query;



**-- AVG Total sales by Day of month**

SELECT

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

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

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 4 - number of the month

GROUP BY EXTRACT(DAY FROM transaction\_date)

ORDER BY day\_of\_month;

WITH daily\_sales AS (

SELECT

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

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

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 4 – number of the month

GROUP BY EXTRACT(DAY FROM transaction\_date)

),

average\_sales AS (

SELECT AVG(total\_sales) AS avg\_sales

FROM daily\_sales

)

SELECT

ds.day\_of\_month,

ds.total\_sales,

CASE

WHEN ds.total\_sales > a.avg\_sales THEN 'Above Average'

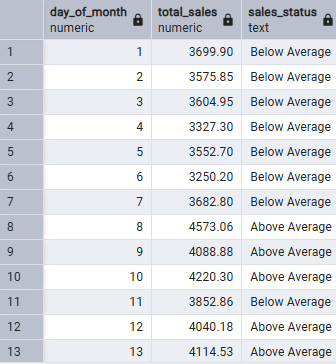
WHEN ds.total\_sales < a.avg\_sales THEN 'Below Average'

ELSE 'Equal to Average'

END AS sales\_status

FROM daily\_sales ds, average\_sales a

ORDER BY ds.day\_of\_month;



**/\*2.5. Sales analysis by product category**

**2.5.1 Analyze sales performance across different product categories to provide insights into wich product categories contribute the most to overall sales\*/**

SELECT

product\_category,

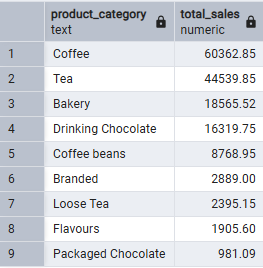
SUM(unit\_price \* transaction\_qty) AS total\_sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5

GROUP BY product\_category

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



**/\*2.6. Top 10 products by sales**

**2.6.1 identify and display the top 10 products based on sales volume and allow user to quicly visualize the best-performing products interms of sales\*/**

SELECT

product\_type,

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

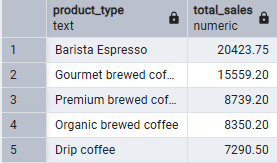
FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5 AND product\_category = 'Coffee' -- for especific product\_category

GROUP BY product\_type

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

LIMIT 10; -- to limit top 10



**/\*2.7. Sales analysis by days and hours**

**2.7.1 Utilize a heat map to visualize sales patterns by days and hours\*/**

SELECT

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

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

COUNT(\*) AS total\_orders

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5 -- May

AND EXTRACT(DOW FROM transaction\_date) = 0 -- sunday

AND EXTRACT(HOUR FROM transaction\_time) = 14 -- hour No 14



**-- Total sales by hours**

SELECT

EXTRACT(HOUR FROM transaction\_time) AS hour\_of\_day,

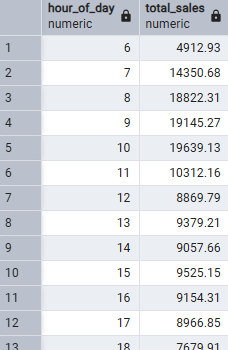
SUM(unit\_price \* transaction\_qty) AS total\_sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5 -- May

GROUP BY EXTRACT(HOUR FROM transaction\_time)

ORDER BY EXTRACT(HOUR FROM transaction\_time);



**-- + By store\_location**

SELECT

store\_location,

EXTRACT(HOUR FROM transaction\_time) AS hour\_of\_day,

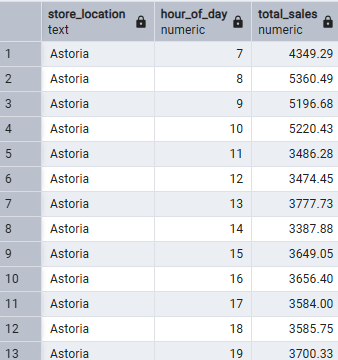
SUM(unit\_price \* transaction\_qty) AS total\_sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5 -- May

GROUP BY store\_location, EXTRACT(HOUR FROM transaction\_time)

ORDER BY store\_location, EXTRACT(HOUR FROM transaction\_time);



**--Total sales by day of week**

**-- Using TO\_CHAR - more simple**

SELECT

TO\_CHAR(transaction\_date, 'Day') AS day\_of\_week,

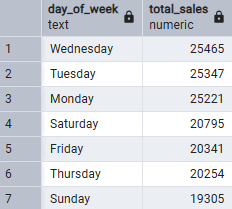
ROUND(SUM(unit\_price \* transaction\_qty)) AS total\_sales

FROM coffee\_sales

WHERE EXTRACT(MONTH FROM transaction\_date) = 5

GROUP BY TO\_CHAR(transaction\_date, 'Day')

ORDER BY total\_sales DESC;



**-- + by store\_id**

SELECT

TRIM(TO\_CHAR(transaction\_date, 'Day')) AS day\_of\_week,

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

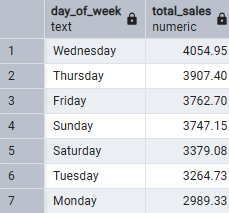
FROM coffee\_sales

WHERE store\_id = 3

AND EXTRACT(MONTH FROM transaction\_date) = 2

GROUP BY TRIM(TO\_CHAR(transaction\_date, 'Day'))

ORDER BY total\_sales DESC;



**--Using CASE if needed, to personalize or specify**

SELECT

CASE

WHEN EXTRACT(DOW FROM transaction\_date) = 1 THEN 'Monday'

WHEN EXTRACT(DOW FROM transaction\_date) = 2 THEN 'Tuesday'

WHEN EXTRACT(DOW FROM transaction\_date) = 3 THEN 'Wednesday'

WHEN EXTRACT(DOW FROM transaction\_date) = 4 THEN 'Thursday'

WHEN EXTRACT(DOW FROM transaction\_date) = 5 THEN 'Friday'

WHEN EXTRACT(DOW FROM transaction\_date) = 6 THEN 'Saturday'

ELSE 'Sunday'

END AS Day\_of\_week,

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

FROM coffee\_sales

WHERE store\_id=4 AND EXTRACT (MONTH FROM transaction\_date) = 5 -- May

GROUP BY

CASE

WHEN EXTRACT(DOW FROM transaction\_date) = 1 THEN 'Monday'

WHEN EXTRACT(DOW FROM transaction\_date) = 2 THEN 'Tuesday'

WHEN EXTRACT(DOW FROM transaction\_date) = 3 THEN 'Wednesday'

WHEN EXTRACT(DOW FROM transaction\_date) = 4 THEN 'Thursday'

WHEN EXTRACT(DOW FROM transaction\_date) = 5 THEN 'Friday'

WHEN EXTRACT(DOW FROM transaction\_date) = 6 THEN 'Saturday'

ELSE 'Sunday'

END

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

