COFFEE SHOP SALES PROJECT

**MY SQL QUERIES**

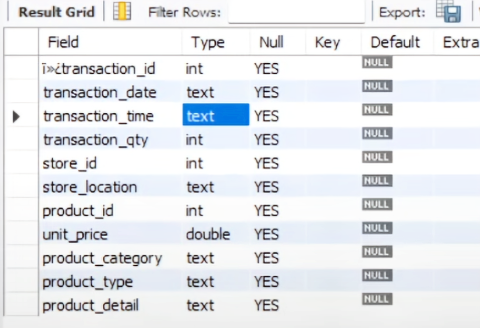
The dataset is called "Coffee." The first steps involve data cleaning and formatting specific columns properly. Once the data is organized, we can perform queries related to the business problems.

create database coffee\_database;

use coffee\_database;

select\*from coffee;

describe coffee;



The transaction\_date and transaction\_time columns are currently in text format, but they need to be converted to Date and Time formats. I’ll use SQL queries to change them into the correct Date and Time format.

1.For ( transaction\_date).

-- Step 1: Add a new column with the DATE type

ALTER TABLE coffee

ADD COLUMN new\_transaction\_date DATE;

-- Step 2: Update the new column with converted values

UPDATE coffee

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

-- Step 3: Drop the old column

ALTER TABLE coffee

DROP COLUMN transaction\_date;

-- Step 4: Rename the new column to the original column name

ALTER TABLE coffee

RENAME COLUMN new\_transaction\_date TO transaction\_date;

2.For (transaction\_time).

-- Step 1: Add a new column with the TIME type

ALTER TABLE coffee

ADD COLUMN new\_transaction\_time TIME;

-- Step 2: Update the new column with converted values

UPDATE coffee

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

-- Step 3: Drop the old column

ALTER TABLE coffee

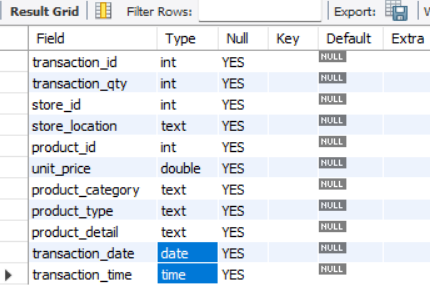
DROP COLUMN transaction\_time;

-- Step 4: Rename the new column to the original column name

ALTER TABLE coffee

RENAME COLUMN new\_transaction\_time TO transaction\_time;

DESCRIBE coffee;



NOTE- Both columns are now displayed in the proper format.

3. Rename the transaction\_id.

select\*from coffee;

alter table coffee

change column ï»¿transaction\_id transaction\_id int;

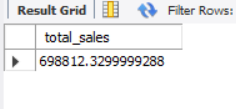
4. Now solve the business Problems statements.

**TOTAL SALES ANALYSIS**

**• Calculate the total sales for each respective month.**

select\*from coffee;

select sum(unit\_price \* transaction\_qty) as total\_sales

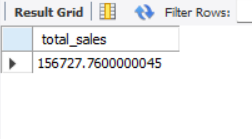
from coffee;

select sum(unit\_price \* transaction\_qty) as total\_sales

from coffee

where

month(transaction\_date) = 5;



select round(sum(unit\_price \* transaction\_qty)) as total\_sales

from coffee

where

A screenshot of a computer

Description automatically generatedmonth(transaction\_date) = 3;

**• Determine the month-on-month increase or decrease in sales.**

SELECT

MONTH(transaction\_date) AS month,

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

(SUM(unit\_price \* transaction\_qty) - LAG(SUM(unit\_price \* transaction\_qty), 1) -- month sales different

OVER (ORDER BY MONTH(transaction\_date))) / LAG(SUM(unit\_price \* transaction\_qty), 1) -- divison by PM sales

OVER (ORDER BY MONTH(transaction\_date)) \* 100 AS mom\_increase\_percentage -- convert in percentage

FROM \* coffee

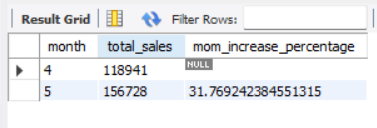
WHERE

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

GROUP BY

MONTH(transaction\_date)

ORDER BY

 MONTH(transaction\_date);

**• Calculate the difference in sales between the selected month and the previous month.**

Ans= 31.76%

**TOTAL ORDER’S ANALYSIS**

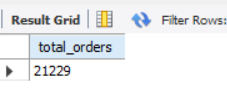
**• Calculate the total number of orders for each respective month.**

select count(transaction\_id) as total\_orders

from coffee

where

month(transaction\_date)=3;

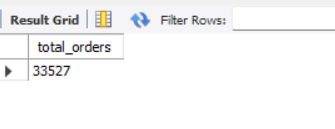


select count(transaction\_id) as total\_orders

from coffee

where

month(transaction\_date) = 5;



**• Determine the month-on-month increase or decrease in the number of orders.**

SELECT

MONTH(transaction\_date) AS month,

ROUND(count(unit\_price \* transaction\_id)) AS total\_sales,

(SUM(unit\_price \* transaction\_id) - LAG(SUM(unit\_price \* transaction\_id), 1) -- month sales different

OVER (ORDER BY MONTH(transaction\_date))) / LAG(SUM(unit\_price \* transaction\_id), 1) -- divison by PM sales

OVER (ORDER BY MONTH(transaction\_date)) \* 100 AS mom\_increase\_percentage -- convert in percentage

FROM

coffee

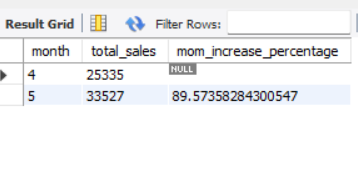
WHERE

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

GROUP BY

MONTH(transaction\_date)

ORDER BY

 MONTH(transaction\_date);

**• Calculate the difference in the number of orders between the selected month and the previous month.**

Ans- 89.57%

**TOTAL QUANTITY SOLD ANALYSIS**

**• Calculate the total quantity sold for each respective month.**

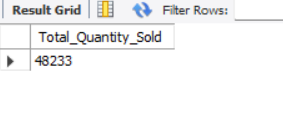
**• Calculate the difference in the total quantity sold between the selected month and the previous month.**

SELECT SUM(transaction\_qty) as Total\_Quantity\_Sold

FROM coffee

WHERE

MONTH(transaction\_date) = 5; -- for month of (CM-May)



SELECT SUM(transaction\_qty) as Total\_Quantity\_Sold

FROM coffee

A screenshot of a computer

Description automatically generatedWHERE MONTH (transaction\_date) = 6 -- for month of (CM-May)

**• Determine the month-on-month increase or decrease in the total quantity sold.**

SELECT

MONTH(transaction\_date) AS month,

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

(SUM(transaction\_qty) - LAG(SUM(transaction\_qty), 1)

OVER (ORDER BY MONTH(transaction\_date))) / LAG(SUM(transaction\_qty), 1)

OVER (ORDER BY MONTH(transaction\_date)) \* 100 AS mom\_increase\_percentage

FROM

coffee

WHERE

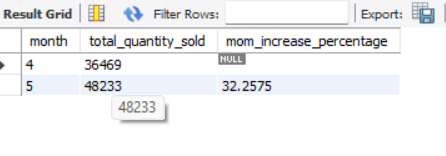
MONTH(transaction\_date) IN (4, 5) -- for April and May

GROUP BY

MONTH(transaction\_date)

ORDER BY

MONTH(transaction\_date);



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

SELECT

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

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

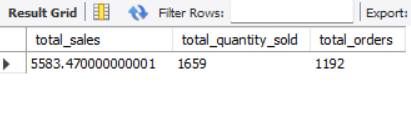
COUNT(transaction\_id) AS total\_orders

FROM

coffee

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,

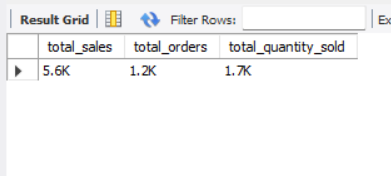
CONCAT(ROUND(SUM(transaction\_qty) / 1000, 1),'K') AS total\_quantity\_sold

FROM

coffee

WHERE

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



**--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

WHERE

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

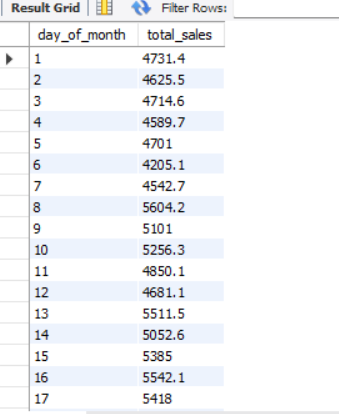
GROUP BY

DAY(transaction\_date)

ORDER BY

DAY(transaction\_date);

A screenshot of a data

Description automatically generated

**-- 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

WHERE

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

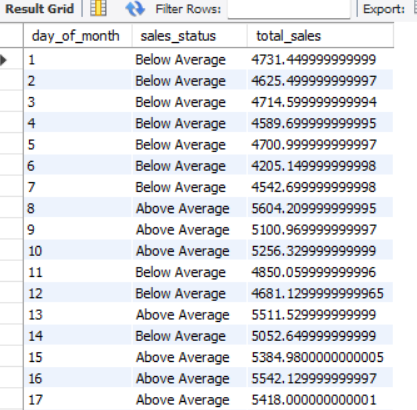
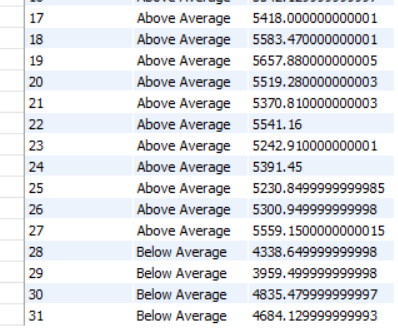
GROUP BY

DAY(transaction\_date)

) AS sales\_data

ORDER BY

day\_of\_month;



**-- SALES BY WEEKDAY / WEEKEND**

SELECT

CASE

WHEN DAYOFWEEK(transaction\_date) IN (1, 7) THEN 'Weekends'

ELSE 'Weekdays'

END AS day\_type,

ROUND(SUM(unit\_price \* transaction\_qty),2) AS total\_sales

FROM

coffee

WHERE

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

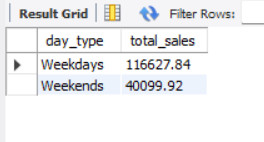
GROUP BY

CASE

WHEN DAYOFWEEK(transaction\_date) IN (1, 7) THEN 'Weekends'

ELSE 'Weekdays'

END;



**-- SALES BY STORE LOCATION**

SELECT

store\_location,

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

FROM coffee

WHERE

MONTH(transaction\_date) =5

GROUP BY store\_location

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

A screenshot of a computer

Description automatically generated

**-- SALES BY PRODUCT CATEGORY**

SELECT

product\_category,

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

FROM coffee

WHERE

MONTH(transaction\_date) = 5

GROUP BY product\_category

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

A screenshot of a computer

Description automatically generated

**-- SALES BY PRODUCTS (TOP 10)**

SELECT

product\_type,

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

FROM coffee

WHERE

MONTH(transaction\_date) = 5

GROUP BY product\_type

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

LIMIT 10;

A screenshot of a computer

Description automatically generated

**-- 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

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);

A screenshot of a computer

Description automatically generated

**-- TO GET SALES FROM MONDAY TO SUNDAY FOR MONTH OF MAY**

SELECT

CASE

WHEN DAYOFWEEK(transaction\_date) = 2 THEN 'Monday'

WHEN DAYOFWEEK(transaction\_date) = 3 THEN 'Tuesday'

WHEN DAYOFWEEK(transaction\_date) = 4 THEN 'Wednesday'

WHEN DAYOFWEEK(transaction\_date) = 5 THEN 'Thursday'

WHEN DAYOFWEEK(transaction\_date) = 6 THEN 'Friday'

WHEN DAYOFWEEK(transaction\_date) = 7 THEN 'Saturday'

ELSE 'Sunday'

END AS Day\_of\_Week,

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

FROM

coffee

WHERE

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

GROUP BY

CASE

WHEN DAYOFWEEK(transaction\_date) = 2 THEN 'Monday'

WHEN DAYOFWEEK(transaction\_date) = 3 THEN 'Tuesday'

WHEN DAYOFWEEK(transaction\_date) = 4 THEN 'Wednesday'

WHEN DAYOFWEEK(transaction\_date) = 5 THEN 'Thursday'

WHEN DAYOFWEEK(transaction\_date) = 6 THEN 'Friday'

WHEN DAYOFWEEK(transaction\_date) = 7 THEN 'Saturday'

ELSE 'Sunday'

END;

A screenshot of a computer

Description automatically generated

**-- TO GET SALES FOR ALL HOURS FOR MONTH OF MAY**

SELECT

HOUR(transaction\_time) AS Hour\_of\_Day,

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

FROM

coffee

WHERE

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

GROUP BY

HOUR(transaction\_time)

ORDER BY

HOUR(transaction\_time);

A screenshot of a computer

Description automatically generated

**THE END**