SQL Queries

**1. Data Cleaning Queries**

**Query 1: Adding a New Column for Order Date**

ALTER TABLE pizza\_shop\_sales

ADD COLUMN new\_order\_date DATE;

**Description:** This query adds a new column named new\_order\_date of type DATE to the pizza\_shop\_sales table. It will store the transformed values of the order\_date column in a proper date format.

**Query 2: Converting Order Date Format**

UPDATE pizza\_shop\_sales

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

**Description:** This query updates the new\_order\_date column by converting values from order\_date using the STR\_TO\_DATE function, which transforms date strings from DD-MM-YYYY format into MySQL’s standard DATE format.

**Query 3: Dropping the Old Order Date Column**

ALTER TABLE pizza\_shop\_sales

DROP COLUMN order\_date;

**Description:** This query removes the original order\_date column, as its values have been successfully converted and stored in new\_order\_date.

**Query 4: Renaming the New Column to Order Date**

ALTER TABLE pizza\_shop\_sales

CHANGE COLUMN new\_order\_date order\_date DATE;

**Description:** This query renames the new\_order\_date column back to order\_date, ensuring consistency while keeping the correct format.

**Query 5: Converting Order Time Format**

UPDATE pizza\_shop\_sales

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

**Description:** This query converts the order\_time values from string format into MySQL’s TIME format using STR\_TO\_DATE.

**Query 6: Modifying Order Time Column Type**

ALTER TABLE pizza\_shop\_sales

MODIFY COLUMN order\_time TIME;

**Description:** This query ensures that the order\_time column is explicitly set as a TIME data type after conversion.

**Query 7: Standardizing Order ID Column**

ALTER TABLE pizza\_shop\_sales

CHANGE COLUMN order\_id Order\_id INT;

**Description:** This query ensures that the order\_id column remains an integer and applies proper naming conventions by changing its case to Order\_id.

**2. Total Sales Analysis**

**Query 1: Calculating Total Sales for March**

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

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) = 3; -- March month

**Description:** This query calculates the total sales in March by multiplying unit\_price by quantity for each order and summing the results. The ROUND function ensures that the total sales value is rounded.

**Query 2: Month-over-Month Sales Growth**

SELECT

MONTH(order\_date) AS month, -- Number of the month

ROUND(SUM(unit\_price \* quantity)) AS total\_sales, -- Total sales column

(SUM(unit\_price \* quantity) - LAG(SUM(unit\_price \* quantity),1)

OVER (ORDER BY MONTH(order\_date))) / LAG(SUM(unit\_price \* quantity),1)

OVER (ORDER BY MONTH(order\_date)) \* 100 AS MoM\_increase\_percentage -- Percentage change

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) IN (4,5) -- Comparing April (PM) and May (CM)

GROUP BY MONTH(order\_date)

ORDER BY MONTH(order\_date);

**Description:** This query calculates the total sales for each month and determines the month-over-month (MoM) percentage increase. The LAG function helps compare the sales of the current month with the previous month to determine growth.

**3. Total Orders Analysis**

**Query 1: Calculating Total Orders for May**

SELECT COUNT(order\_id) AS Total\_orders

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) = 5; -- Month of May

**Description:** This query counts the total number of orders placed in May by counting the occurrences of order\_id where the order date falls within the fifth month.

**Query 2: Month-over-Month Order Growth**

SELECT

MONTH(order\_date) AS month, -- Number of the month

ROUND(COUNT(order\_id)) AS total\_orders, -- Total orders column

(COUNT(order\_id) - LAG(COUNT(order\_id),1)

OVER (ORDER BY MONTH(order\_date))) / LAG(COUNT(order\_id),1)

OVER (ORDER BY MONTH(order\_date)) \* 100 AS MoM\_increase\_percentage -- Percentage change

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) IN (4,5) -- Comparing April (PM) and May (CM)

GROUP BY MONTH(order\_date)

ORDER BY MONTH(order\_date);

**Description:** This query calculates the total number of orders for each month and determines the month-over-month (MoM) percentage increase. The LAG function helps compare the order count of the current month with the previous month to determine growth.

**4. Total Quantity Analysis**

**Query 1: Calculating Total Quantity Sold in June**

SELECT SUM(quantity) AS Total\_Qty

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) = 6; -- Month of June

**Description:** This query calculates the total quantity of pizzas sold in June by summing the quantity column where the order\_date falls within the sixth month.

**Query 2: Month-over-Month Quantity Growth**

SELECT

MONTH(order\_date) AS month, -- Number of the month

ROUND(SUM(quantity)) AS total\_qty, -- Total quantity column

(SUM(quantity) - LAG(SUM(quantity),1)

OVER (ORDER BY MONTH(order\_date))) / LAG(SUM(quantity),1)

OVER (ORDER BY MONTH(order\_date)) \* 100 AS MoM\_increase\_percentage -- Percentage change

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) IN (4,5) -- Comparing April (PM) and May (CM)

GROUP BY MONTH(order\_date)

ORDER BY MONTH(order\_date);

**Description:** This query calculates the total quantity of pizzas sold for each month and determines the month-over-month (MoM) percentage increase. The LAG function helps compare the quantity of the current month with the previous month to analyse growth.

**5. Heat Map and Top 10 Analysis**

**Query 1: Heat Map Analysis for a Specific Date**

SELECT

SUM(unit\_price \* quantity) AS total\_sales,

SUM(quantity) AS total\_qty,

COUNT(order\_id) AS total\_order

FROM pizza\_shop\_sales

WHERE order\_date = '2015-05-18';

**Description:** This query calculates the total sales, total quantity sold, and total orders placed on a specific date, providing key data for a heat map analysis.

**Query 2: Top 10 Best-Selling Pizzas**

SELECT pizza\_name,

ROUND(SUM(unit\_price \* quantity)) AS total\_sales

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) = 5

GROUP BY pizza\_name

ORDER BY SUM(unit\_price \* quantity) DESC

LIMIT 10;

**Description:** This query retrieves the top 10 best-selling pizzas in May by calculating total sales and ranking them in descending order.

**Query 3: Top 10 Best-Selling Pizza Categories**

SELECT pizza\_category,

ROUND(SUM(unit\_price \* quantity)) AS total\_sales

FROM pizza\_shop\_sales

WHERE MONTH(order\_date) = 5

GROUP BY pizza\_category

ORDER BY SUM(unit\_price \* quantity) DESC

LIMIT 10;

**Description:** This query retrieves the top 10 best-selling pizza categories in May by calculating total sales and ranking them in descending order.