**QUERIES FOR Ivan’s Restaurant PIZZA SALE**

1. **Ivan’s Pizza Queries**
2. **Total Revenue**

SELECT ROUND(SUM(total\_price),2) AS total\_revenue

FROM dbo.pizza\_sales



1. **Average Order value**

WITH avg\_order\_value AS (

SELECT ROUND(SUM(total\_price) / COUNT(DISTINCT order\_id), 2) AS average\_order\_value

FROM dbo.pizza\_sales

)

SELECT \* FROM avg\_order\_value;

// Can be simplier but I like to do this one



1. Total Pizzas Sold

SELECT SUM(quantity) AS total\_pizza\_sold

FROM dbo.pizza\_sales



1. **Total Orders**

SELECT COUNT(DISTINCT order\_id) AS total\_pizza\_sold

FROM dbo.pizza\_sales



1. **Average Pizzas Per order**

SELECT CAST(CAST(SUM(quantity) AS DECIMAL(10,2))

/ CAST(COUNT(DISTINCT order\_id) AS DECIMAL(10,2))

AS DECIMAL (10,2))

AS avg\_pizzas\_per\_order

FROM DBO.pizza\_sales



But if full is

SELECT CAST(SUM(quantity) AS DECIMAL(10,2))

/ CAST(COUNT(DISTINCT order\_id) AS DECIMAL(10,2))

AS avg\_pizzas\_per\_order

FROM DBO.pizza\_sales



**CHARTS & their Queries**

1. **Daily Trend**

SELECT order\_day, total\_orders

FROM (

SELECT

DATENAME(WEEKDAY, order\_date) AS order\_day,

COUNT(DISTINCT order\_id) AS total\_orders,

CASE DATENAME(WEEKDAY, order\_date)

WHEN 'Monday' THEN 1

WHEN 'Tuesday' THEN 2

WHEN 'Wednesday' THEN 3

WHEN 'Thursday' THEN 4

WHEN 'Friday' THEN 5

WHEN 'Saturday' THEN 6

WHEN 'Sunday' THEN 7

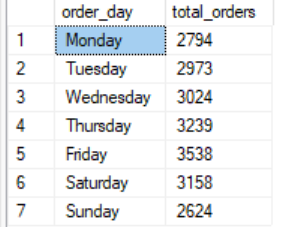
END AS weekday\_number

FROM dbo.pizza\_sales

GROUP BY DATENAME(WEEKDAY, order\_date)

) AS temp

ORDER BY weekday\_number;



-- used a subquery to not show the weekday\_number in the final output

-- Made a CASE WHEN expression so it can be used in the order by clause. The reason for this use is because the output of DATENAME is a weekday name and using order by will Order alphabetically

1. **HOURLY TREND**

WITH HourDay AS (

SELECT DATENAME(HOUR, order\_time) AS time\_of\_hour, order\_id

FROM dbo.pizza\_sales

)

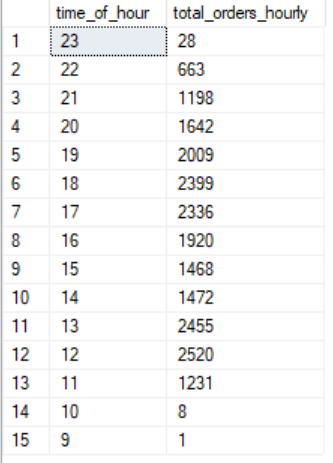
SELECT time\_of\_hour, COUNT(DISTINCT order\_id) AS total\_orders\_hourly

FROM HourDay

GROUP BY time\_of\_hour

ORDER BY CAST(time\_of\_hour AS INT) DESC

--casted as INT because there is a problem when DESC it stays at the top.



1. **Percentage of sales By Pizza Category**

SELECT DISTINCT pizza\_category

from dbo.pizza\_sales

-- Query to check all pizza categories

SELECT pizza\_category, SUM(total\_price) \* 100 / (SELECT SUM(total\_price) from pizza\_sales) AS total\_contribution\_percentage\_wholeYear

FROM dbo.pizza\_sales

GROUP BY pizza\_category;

-- Query to find percentage of total sales of which category / how much each category contributes to total revenue.

WITH total\_sales AS (

SELECT pizza\_category, SUM(total\_price) AS category\_total

FROM dbo.pizza\_sales

WHERE MONTH(order\_date) = 2

GROUP BY pizza\_category

),

overall\_total\_sales AS (

SELECT SUM(total\_price) AS overall

FROM dbo.pizza\_sales

WHERE MONTH(order\_date) = 2

)

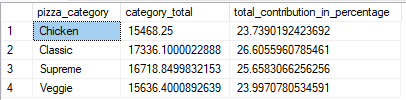
SELECT DISTINCT ts.pizza\_category, ts.category\_total,

ts.category\_total \* 100 / ots.overall AS total\_contribution\_in\_percentage

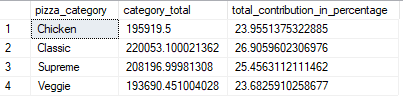
FROM total\_sales ts, overall\_total\_sales ots, pizza\_sales

-- Query to find percentage of total sales of which category / how much each category contributes to total revenue in A month (change the MONTH(order\_date) to a specified number of month

This is set to febuary of that year.



**For whole year**



1. **Percentage of Sales By Pizza Size**

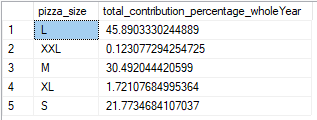
**Same as the Pizza Category**

SELECT pizza\_size, SUM(total\_price) \* 100 / (SELECT SUM(total\_price) from pizza\_sales) AS total\_contribution\_percentage\_wholeYear

FROM dbo.pizza\_sales

GROUP BY pizza\_size;

-- Query to find percentage of total sales of which size / how much each size contributes to total revenue.



WITH total\_sales AS (

SELECT pizza\_size, SUM(total\_price) AS size\_total\_earning

FROM dbo.pizza\_sales

GROUP BY pizza\_size

),

-- calculates total revenue per size

overall\_total\_sales AS (

SELECT SUM(total\_price) AS overall\_earning

FROM dbo.pizza\_sales

)

-- calculates the total revenue across all sizes

SELECT pizza\_size, size\_total\_earning, total\_contribution\_in\_percentage

FROM (

SELECT

ts.pizza\_size,

ts.size\_total\_earning,

ROUND(ts.size\_total\_earning \* 100.0 / ots.overall\_earning, 2) AS total\_contribution\_in\_percentage,

CASE ts.pizza\_size

WHEN 'S' THEN 1

WHEN 'M' THEN 2

WHEN 'L' THEN 3

WHEN 'XL' THEN 4

WHEN 'XXL' THEN 5

ELSE 6

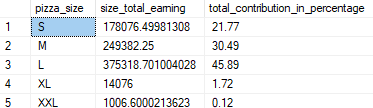
END AS sort\_order

FROM total\_sales ts

CROSS JOIN overall\_total\_sales ots

) AS sorted\_table

ORDER BY sort\_order;



1. **Total pizza sold by category**

WITH total\_pizza\_sold AS (

SELECT pizza\_category, SUM(quantity) AS count\_sales

FROM dbo.pizza\_sales

GROUP BY pizza\_category

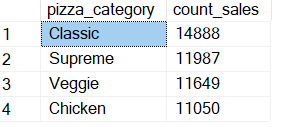
)

SELECT ts.\*

from total\_pizza\_sold ts

ORDER BY ts.count\_sales DESC;

--classic contains the most sales for the whole year.



1. **Top 5 Best Sellers by Total Pizza’s Sold and 7. Bottom**

SELECT DISTINCT pizza\_name

FROM dbo.pizza\_sales

-- Top 5 best sellers by total pizza Sold

SELECT TOP 5 pizza\_name, SUM(quantity) AS total\_pizza\_sold

FROM dbo.pizza\_sales

GROUP BY pizza\_name

ORDER BY total\_pizza\_sold DESC;

-- Bottom 5 Worst Sellers

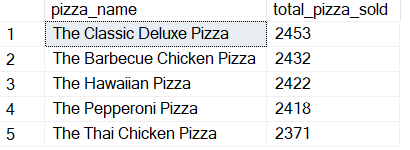
SELECT TOP 5 pizza\_name, SUM(quantity) AS total\_pizza\_sold

FROM dbo.pizza\_sales

GROUP BY pizza\_name

ORDER BY total\_pizza\_sold ASC

**TOP 5 BEST SELLERS**



**BOTTOM 5 WORST SELLERS**

