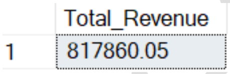
Pizza Sales SQL Queries: Extracting Key Performance Indicators

1. Find the total revenue of total pizza orders
2. Find the Average order value (per order) calculated by dividing the total revenue by the total number of orders
3. Find the total number of pizzas sold
4. Find the total number of orders ordered.
5. Find the Average number of pizzas ordered/sold per order, calculated by dividing the total number of pizzas sold by the total number of orders?
6. Find the What are the busiest Days & MONTHS & QUARTER and Times
7. Find the Percentage of Sales by each Pizza Category
8. Find the Percentage of Sales by each 'Pizza Size '
9. Find the Top 5 Best Sellers (pizza name)
10. Find the Lowest 5 Worst Sellers (pizza name) (bottom 5 pizzas sold)

1. Find the total revenue of total pizza orders

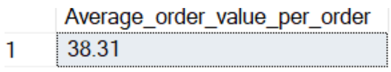
```
SELECT ROUND(SUM(total_price),2) AS Total_Revenue

FROM Pizza_sales_2023;
```



	Total_Revenue
1	817860.05

2. Find the Average order value (per order) calculated by dividing the total revenue by the total number of orders

```
SELECT

        ROUND(SUM(total_price)/ COUNT(DISTINCT(order_id)),2) AS
Average_order_value_per_order

FROM Pizza_sales_2023;
```

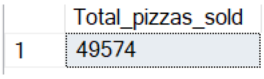


	Average_order_value_per_order
1	38.31

3. Find the total number of pizzas sold

```
SELECT
```

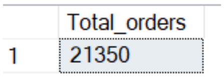
```
        SUM(quantity) AS Total_pizzas_sold
FROM Pizza_sales_2023;
```



	Total_pizzas_sold
1	49574

4. Find the total number of orders ordered.

```
SELECT
        COUNT(DISTINCT(order_id)) AS Total_orders
FROM Pizza_sales_2023;
```

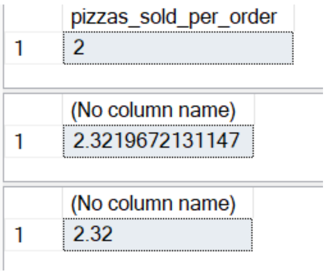


	Total_orders
1	21350

5. Find the Average number of pizzas ordered/sold per order. calculated by dividing the total number of pizzas sold by the total number of orders?

```
                --- This query gives the Integer output without decimal values
SELECT
        SUM(quantity) /COUNT(DISTINCT(order_id)) AS pizzas_sold_per_order
FROM Pizza_sales_2023;


                ---- The CAST function is used to convert a value from one data type to
        another


SELECT
        CAST(SUM(quantity) AS DECIMAL(10,2)) /
        CAST(COUNT(DISTINCT(order_id)) AS DECIMAL(10,2))
FROM Pizza_sales_2023;


                --------- USE CAST() especially if you expect fractional averages
SELECT
        CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) /
        CAST(COUNT(DISTINCT(order_id)) AS DECIMAL(10,2)) AS DECIMAL(10,2))
```

```
FROM Pizza_sales_2023;
```



	pizzas_sold_per_order
1	2

	(No column name)
1	2.3219672131147

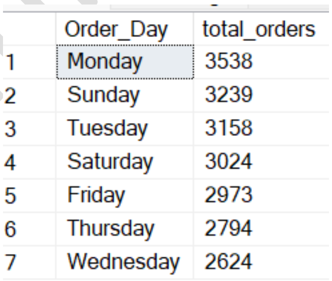
	(No column name)
1	2.32

6. Find the What are the busiest Days & MONTHS & QUARTER and Times

```
--DW for Day of the Week (e.g., 'Monday', 'Tuesday')
--MONTH for the Month Name (e.g., 'January', 'February')
--YEAR for the Year (e.g., '2023')
--DAY for the Day of the Month (e.g., '1', '15', '31')
--QUARTER for the Quarter (e.g., '1', '2', '3', '4')
```

-- DAILY TREND IN A WEEK OF PIZZA SALES

```
SELECT
        DATENAME(DW, order_date) AS Order_Day,
        COUNT(DISTINCT order_id) AS total_orders
FROM Pizza_sales_2023
GROUP BY DATENAME(DW, order_date)
ORDER BY COUNT(DISTINCT order_id) DESC;
```

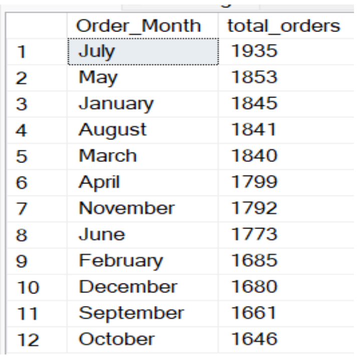


	Order_Day	total_orders
1	Monday	3538
2	Sunday	3239
3	Tuesday	3158
4	Saturday	3024
5	Friday	2973
6	Thursday	2794
7	Wednesday	2624

-- MONTHLY TREND IN A WEEK OF PIZZA SALES

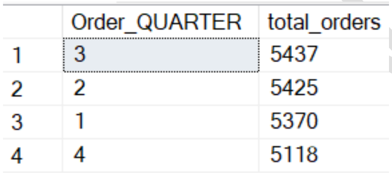
```
SELECT
        DATENAME(MONTH, order_date) AS Order_Month,
COUNT(DISTINCT order_id) AS total_orders
FROM Pizza_sales_2023
```

```
GROUP BY DATENAME(MONTH, order_date)

ORDER BY COUNT(DISTINCT order_id) DESC;
```



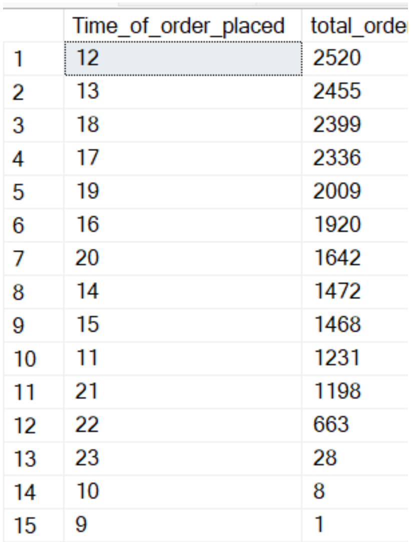
	Order_Month	total_orders
1	July	1935
2	May	1853
3	January	1845
4	August	1841
5	March	1840
6	April	1799
7	November	1792
8	June	1773
9	February	1685
10	December	1680
11	September	1661
12	October	1646

```
        -- Quarterly TREND OF PIZZA SALES

SELECT

        DATENAME(QUARTER, order_date) AS Order_QUARTER,

        COUNT(DISTINCT order_id) AS total_orders

FROM Pizza_sales_2023

GROUP BY DATENAME(QUARTER, order_date)

ORDER BY COUNT(DISTINCT order_id) DESC;
```

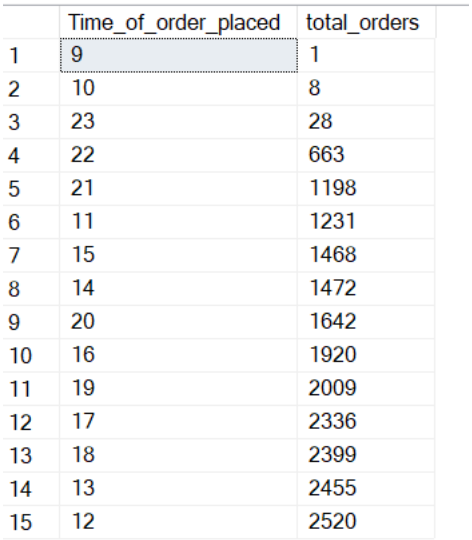


	Order_QUARTER	total_orders
1	3	5437
2	2	5425
3	1	5370
4	4	5118

```
        -- HOURLY TREND OF PIZZA SALES

SELECT

        DATEPART(HOUR, order_time) AS Time_of_order_placed,

        COUNT(DISTINCT(order_id)) AS total_orders

FROM Pizza_sales_2023

GROUP BY DATEPART(HOUR, order_time)

ORDER BY COUNT(DISTINCT(order_id)) DESC;
```



	Time_of_order_placed	total_orde
1	12	2520
2	13	2455
3	18	2399
4	17	2336
5	19	2009
6	16	1920
7	20	1642
8	14	1472
9	15	1468
10	11	1231
11	21	1198
12	22	663
13	23	28
14	10	8
15	9	1

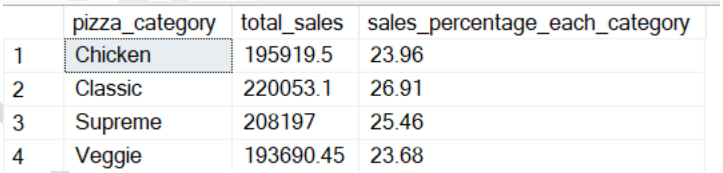


	Time_of_order_placed	total_orders
1	9	1
2	10	8
3	23	28
4	22	663
5	21	1198
6	11	1231
7	15	1468
8	14	1472
9	20	1642
10	16	1920
11	19	2009
12	17	2336
13	18	2399
14	13	2455
15	12	2520

7. Find the Percentage of Sales by each Pizza Category

--- Query 1 produces values in terms of the 'percentage of total sales' that each pizza category contributes.

```
SELECT
        pizza_category,
        ROUND(SUM(total_price),2) AS total_sales,
        ROUND(SUM(total_price)*100/ (SELECT SUM(total_price) FROM
Pizza_sales_2023),2) AS sales_percentage_each_category
FROM Pizza_sales_2023
GROUP BY pizza_category
ORDER BY pizza_category ;
```

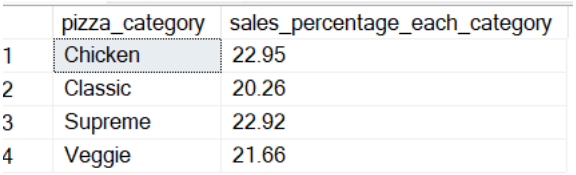


	pizza_category	total_sales	sales_percentage_each_category
1	Chicken	195919.5	23.96
2	Classic	220053.1	26.91
3	Supreme	208197	25.46
4	Veggie	193690.45	23.68

----Query 2 produces values in terms of 'average revenue per order' for each pizza category.

```
SELECT
        pizza_category,
        ROUND(SUM(total_price)/COUNT(DISTINCT order_id),2) AS
sales_percentage_each_category
FROM Pizza_sales_2023
GROUP BY pizza_category
```

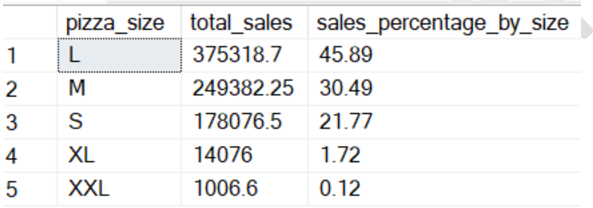
```
    ORDER BY pizza_category ;
```



	pizza_category	sales_percentage_each_category
1	Chicken	22.95
2	Classic	20.26
3	Supreme	22.92
4	Veggie	21.66

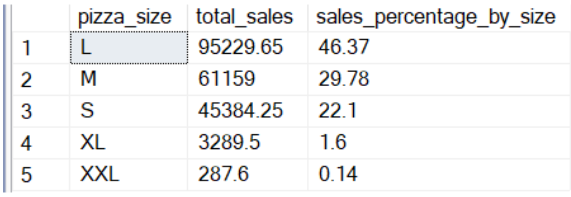
8. Find the Percentage of Sales by each 'Pizza Size '

```
    SELECT
        pizza_size,
        ROUND(SUM(total_price),2) AS total_sales,
        ROUND(SUM(total_price)*100/ (SELECT SUM(total_price) FROM
Pizza_sales_2023),2) AS sales_percentage_by_size
    FROM Pizza_sales_2023
    GROUP BY pizza_size
    ORDER BY  sales_percentage_by_size DESC;
```



	pizza_size	total_sales	sales_percentage_by_size
1	L	375318.7	45.89
2	M	249382.25	30.49
3	S	178076.5	21.77
4	XL	14076	1.72
5	XXL	1006.6	0.12

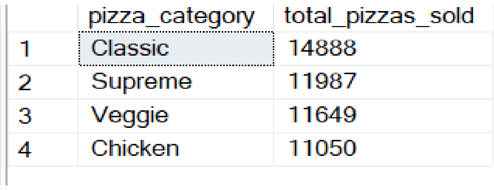
```
    -- Finding the values for 1st quarter
    SELECT
        pizza_size,
        ROUND(SUM(total_price),2) AS total_sales,
        ROUND(SUM(total_price)*100/ (SELECT SUM(total_price) FROM
Pizza_sales_2023        WHERE DATEPART(quarter, order_date) = 1),2) AS
sales_percentage_by_size
    FROM Pizza_sales_2023
    WHERE DATEPART(quarter, order_date) = 1
    GROUP BY pizza_size
    ORDER BY  sales_percentage_by_size DESC;
```



	pizza_size	total_sales	sales_percentage_by_size
1	L	95229.65	46.37
2	M	61159	29.78
3	S	45384.25	22.1
4	XL	3289.5	1.6
5	XXL	287.6	0.14

9. Find the Total number of Pizzas sold by Pizza Category

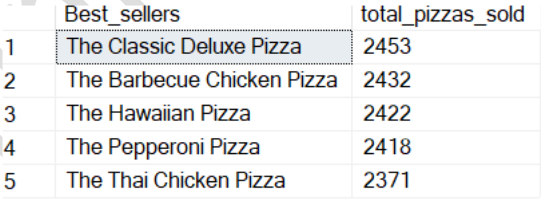
```
SELECT
        pizza_category,
        SUM(quantity) AS total_pizzas_sold
FROM Pizza_sales_2023
GROUP BY pizza_category
ORDER BY total_pizzas_sold DESC;
```



	pizza_category	total_pizzas_sold
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

9. Find the Top 5 Best Sellers (pizza name)

```
SELECT TOP 5
        pizza_name AS Best_sellers,
        SUM(quantity) AS total_pizzas_sold
FROM Pizza_sales_2023
GROUP BY pizza_name
ORDER BY total_pizzas_sold DESC;
```

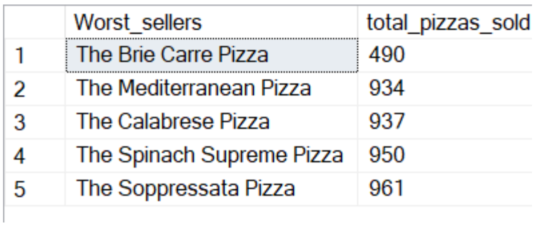


	Best_sellers	total_pizzas_sold
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

10. Find the Lowest 5 Worst Sellers (pizza name) (bottom 5 pizzas sold)

```
SELECT TOP 5
        pizza_name AS Worst_sellers,
        SUM(quantity) AS total_pizzas_sold
FROM Pizza_sales_2023
GROUP BY pizza_name
```

```
ORDER BY total_pizzas_sold;
```



	Worst_sellers	total_pizzas_sold
1	The Brie Carre Pizza	490
2	The Mediterranean Pizza	934
3	The Calabrese Pizza	937
4	The Spinach Supreme Pizza	950
5	The Soppressata Pizza	961

We've successfully addressed 10 business/client inquiries using SQL Server.

Next, we'll transition to Excel for data visualization and further analysis.