

# PIZZA SALES ANALYSIS USING SQL AND POWER BI

PRESENTED BY:  
SUJAL JAIN



# OBJECTIVE

- To analyze pizza sales data and derive insights to answer various business questions.
- To analyze customer preferences and demographics to tailor marketing strategies.

# SOFTWARE USED

- MS OFFICE/ EXCEL
- MS SQL SERVER
- SQL SERVER MANAGEMENT STUDIO
- POWER BI

# PROBLEM STATEMENT

## KPI's REQUIREMENT

We need to analyze key indicators for our pizza sales data to gain insights into our business performance. Specifically, we want to calculate the following metrics:

- 1. Total Revenue:** The sum of the total price of all pizza orders.
- 2. Average Order Value:** The average amount spent per order, calculated by dividing the total revenue by the total number of orders.
- 3. Total Pizzas Sold:** The sum of the quantities of all pizzas sold.
- 4. Total Orders:** The total number of orders placed.
- 5. Average Pizzas Per Order:** The average number of pizzas sold per order, calculated by dividing the total number of pizzas sold by the total number of orders.

# **PROBLEM STATEMENT**

## **CHARTS REQUIREMENT**

We would like to visualize various aspects of our pizza sales data to gain insights and understand key trends.

### **1. Daily Trend for Total Orders:**

Create a bar chart that displays the daily trend of total orders over a specific time period. This chart will help us identify any patterns or fluctuations in order volumes on a daily basis.

### **2. Monthly Trend for Total Orders:**

Create a line chart that illustrates the hourly trend of total orders throughout the day. This chart will allow us to identify peak hours or periods of high order activity.

# **PROBLEM STATEMENT**

## **CHARTS REQUIREMENT**

### **3. Percentage of Sales by Pizza Category:**

Create a pie chart that shows the distribution of sales across different pizza categories. This chart will provide insights into the popularity of various pizza categories and their contribution to overall sales.

### **4. Percentage of Sales by Pizza Size:**

Generate a pie chart that represents the percentage of sales attributed to different pizza sizes. This chart will help us understand customer preferences for pizza sizes and their impact on sales.

# **PROBLEM STATEMENT**

## **CHARTS REQUIREMENT**

### **5. Total Pizzas Sold by Pizza Category:**

Create a funnel chart that presents the total number of pizzas sold for each pizza category. This chart will allow us to compare the sales performance of different pizza categories.

### **6. Top 5 Best Sellers by Revenue, Total Quantity and Total Orders:**

Create a bar chart highlighting the top 5 best-selling pizzas based on the Revenue, Total Quantity, Total Orders. This chart will help us identify the most popular pizza options.

# PROBLEM STATEMENT

## CHARTS REQUIREMENT

### **7. Bottom 5 Best Sellers by Revenue, Total Quantity and Total Orders:**

Create a bar chart showcasing the bottom 5 worst-selling pizzas based on the Revenue, Total Quantity, Total Orders. This chart will enable us to identify underperforming or less popular pizza options.

---

# PIZZA SALES SQL QUERIES

## A. KPI's

### 1. Total Revenue:

```
SELECT SUM(total_price) AS TOTAL_REVENUET FROM  
pizza_sales;
```

The screenshot shows a SQL query results window. At the top, there are two tabs: 'Results' (selected) and 'Messages'. The results table has one row with the following data:

	TOTAL_REVENUET
1	817860.05083847

# PIZZA SALES SQL QUERIES

## A. KPI's

### 2. Average Order Value

```
SELECT (SUM(total_price) / COUNT(DISTINCT order_id)) AS  
Avg_order_Value FROM pizza_sales
```

The screenshot shows a SQL query results window. At the top, there are two tabs: 'Results' (selected) and 'Messages'. The results table has one row with two columns. The first column is labeled 'Avg\_order\_Value' and contains the value '38.3072623343546'. The second column is empty.

	Avg_order_Value
1	38.3072623343546

# PIZZA SALES SQL QUERIES

## A. KPI's

### 3. Total Pizzas Sold

```
SELECT SUM(quantity) AS Total_pizza_sold FROM pizza_sales
```

The screenshot shows a SQL query results window. At the top, there are two tabs: 'Results' (selected) and 'Messages'. The 'Results' tab displays a single row of data in a table format. The table has one column labeled 'Total\_pizza\_sold' with the value '49574'.

	Total_pizza_sold
1	49574

# PIZZA SALES SQL QUERIES

## A. KPI's

### 4. Total Orders

```
SELECT COUNT(DISTINCT order_id) AS Total_Orders FROM  
pizza_sales
```

The image shows a screenshot of a SQL query results window. At the top, there are two tabs: "Results" (selected) and "Messages". The "Results" tab displays a single row of data in a table format. The table has one column labeled "Total\_Orders" with the value "21350".

	Total_Orders
1	21350

# PIZZA SALES SQL QUERIES

## A. KPI's

### 5. 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 pizza_sales
```

The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a single row of data. The column header is 'Avg\_Pizzas\_per\_order' and the value is '2.32'. The 'Messages' tab is also visible.

	Avg_Pizzas_per_order
1	2.32

# PIZZA SALES SQL QUERIES

## B. Daily Trend for Total Orders

```
SELECT DATENAME(DW,ORDER_DATE) AS  
ORDER_DAY,COUNT(DISTINCT ORDER_ID) AS  
TOTAL_ORDER FROM PIZZA_SALES GROUP BY  
DATENAME(DW,ORDER_DATE)
```

The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with seven rows, each representing a day of the week and its corresponding total number of orders. The table has three columns: a row number, the day name, and the total order count. The days are listed in descending order of total sales, with Saturday having the highest value at 3158.

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

# PIZZA SALES SQL QUERIES

## C. Monthly Trend for Orders

```
select DATENAME(MONTH, order_date) as ORDER_MONTH,  
COUNT(DISTINCT order_id) as Total_Orders  
from pizza_sales  
GROUP BY DATENAME(MONTH, order_date)
```

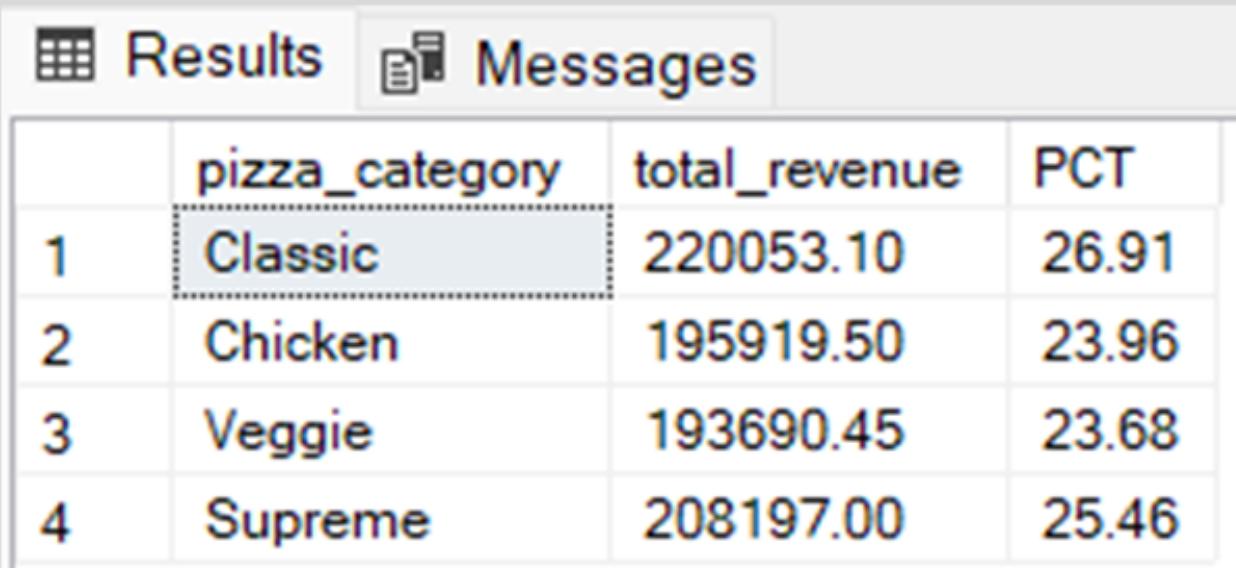
The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with 12 rows, each representing a month and its total number of orders. The columns are labeled 'ORDER\_MONTH' and 'TOTAL\_ORDER'. The data is as follows:

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

# PIZZA SALES SQL QUERIES

## D. % of Sales by Pizza Category

```
SELECT pizza_category, CAST(SUM(total_price) AS  
DECIMAL(10,2)) as total_revenue,  
CAST(SUM(total_price) * 100 / (SELECT SUM(total_price) from  
pizza_sales) AS DECIMAL(10,2)) AS PCT  
FROM pizza_sales  
GROUP BY pizza_category
```



The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with four columns: 'pizza\_category', 'total\_revenue', and 'PCT'. The table has 4 rows of data. The first row, for 'Classic' pizza, is highlighted with a dotted border.

	pizza_category	total_revenue	PCT
1	Classic	220053.10	26.91
2	Chicken	195919.50	23.96
3	Veggie	193690.45	23.68
4	Supreme	208197.00	25.46

# PIZZA SALES SQL QUERIES

## E. % of Sales by Pizza Size

```
SELECT pizza_size, CAST(SUM(total_price) AS DECIMAL(10,2)) as  
total_revenue,  
CAST(SUM(total_price) * 100 / (SELECT SUM(total_price) from  
pizza_sales) AS DECIMAL(10,2)) AS PCT  
FROM pizza_sales  
GROUP BY pizza_size  
ORDER BY PCT
```

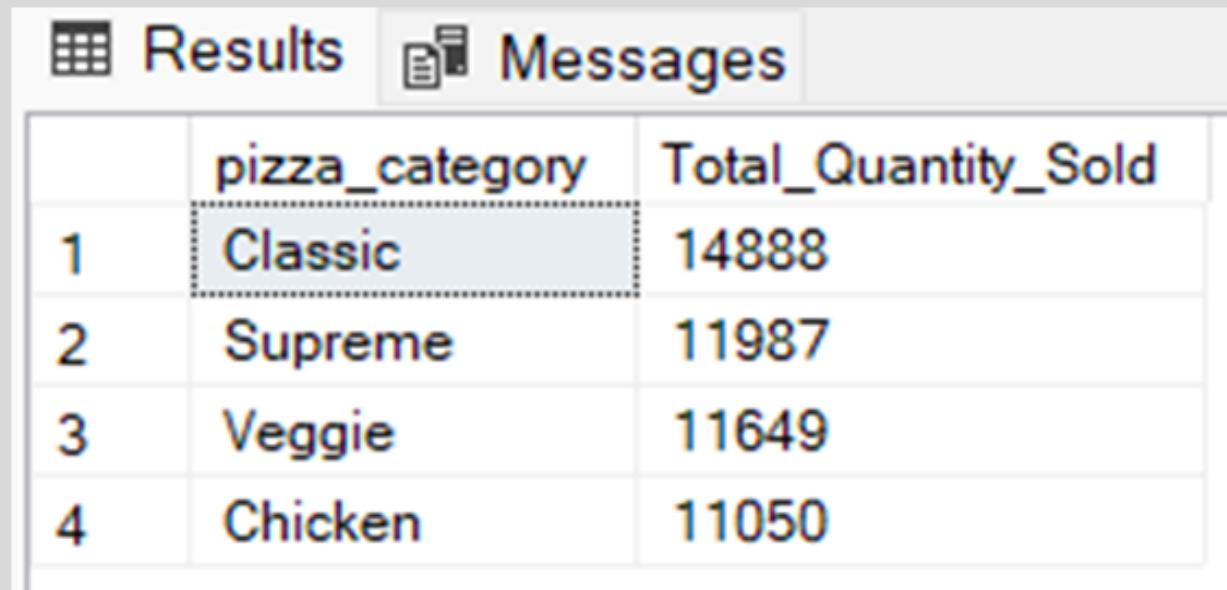
The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with four columns: 'pizza\_size', 'total\_revenue', and 'PCT'. The table has 5 rows, indexed 1 to 5. Row 1 shows 'L' as the pizza size, \$375318.70 as total revenue, and 45.89% as PCT. Rows 2 through 5 show M, S, XL, and XXL sizes respectively, with their corresponding revenue and PCT values.

	pizza_size	total_revenue	PCT
1	L	375318.70	45.89
2	M	249382.25	30.49
3	S	178076.50	21.77
4	XL	14076.00	1.72
5	XXL	1006.60	0.12

# PIZZA SALES SQL QUERIES

## F. Total Pizzas Sold by Pizza Category

```
SELECT pizza_category, SUM(quantity) as Total_Quantity_Sold  
FROM pizza_sales  
GROUP BY pizza_category  
ORDER BY Total_Quantity_Sold DESC
```



The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with four rows. The table has two columns: 'pizza\_category' and 'Total\_Quantity\_Sold'. The data is as follows:

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

# PIZZA SALES SQL QUERIES

## G. Top 5 Pizzas by Revenue

```
SELECT Top 5 pizza_name, SUM(total_price) AS Total_Revenue  
FROM pizza_sales  
GROUP BY pizza_name  
ORDER BY Total_Revenue DESC
```

The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with five rows, each representing a pizza and its total revenue. The table has two columns: 'pizza\_name' and 'Total\_Revenue'. The data is as follows:

	pizza_name	Total_Revenue
1	The Thai Chicken Pizza	43434.25
2	The Barbecue Chicken Pizza	42768
3	The California Chicken Pizza	41409.5
4	The Classic Deluxe Pizza	38180.5
5	The Spicy Italian Pizza	34831.25

# PIZZA SALES SQL QUERIES

## H. Bottom 5 Pizzas by Revenue

```
SELECT Top 5 pizza_name, SUM(total_price) AS Total_Revenue  
FROM pizza_sales  
GROUP BY pizza_name  
ORDER BY Total_Revenue ASC
```

	pizza_name	Total_Revenue
1	The Brie Carre Pizza	11588.4998130798
2	The Green Garden Pizza	13955.75
3	The Spinach Supreme Pizza	15277.75
4	The Mediterranean Pizza	15360.5
5	The Spinach Pesto Pizza	15596

# PIZZA SALES SQL QUERIES

## I. Top 5 Pizzas by Quantity

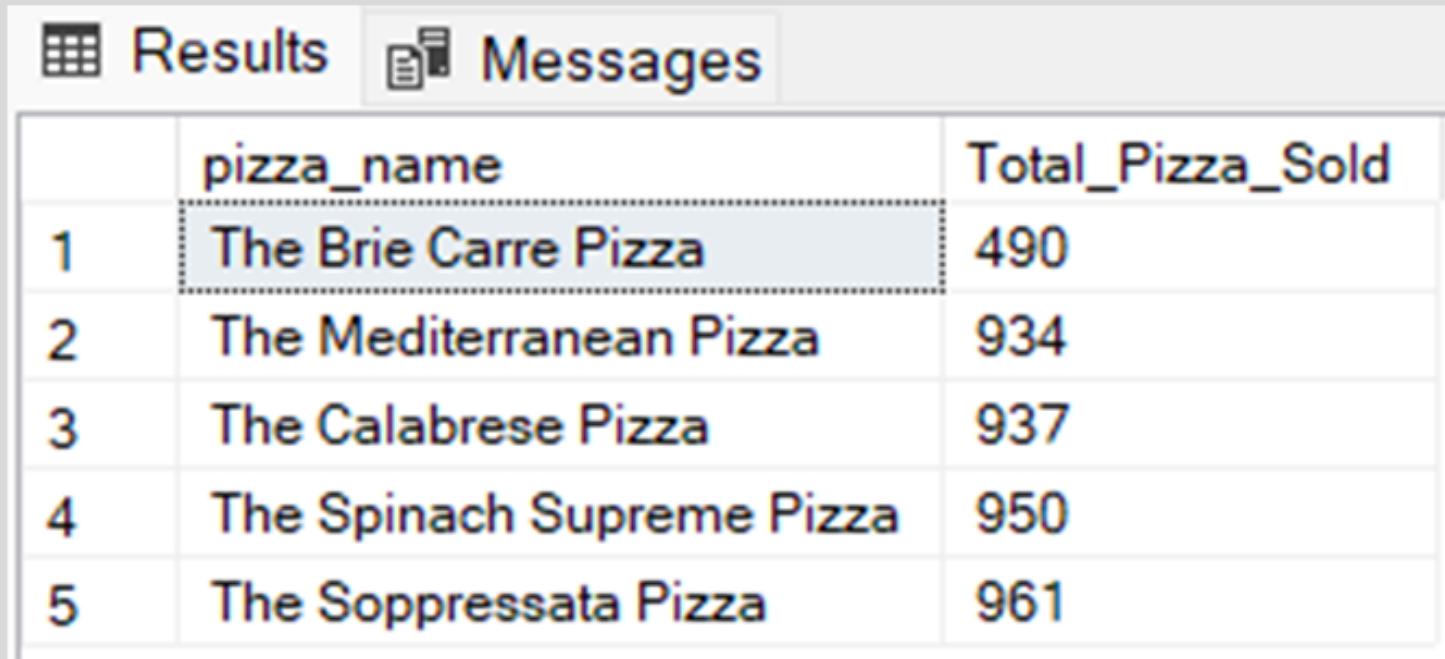
```
SELECT Top 5 pizza_name, SUM(quantity) AS Total_Pizza_Sold  
FROM pizza_sales  
GROUP BY pizza_name  
ORDER BY Total_Pizza_Sold DESC
```

	pizza_name	Total_Pizza_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

# PIZZA SALES SQL QUERIES

## J. Bottom 5 Pizzas by Quantity

```
SELECT TOP 5 pizza_name, SUM(quantity) AS Total_Pizza_Sold  
FROM pizza_sales  
GROUP BY pizza_name  
ORDER BY Total_Pizza_Sold ASC
```



The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected, displaying a table with five rows. The table has two columns: 'pizza\_name' and 'Total\_Pizza\_Sold'. The data is as follows:

	pizza_name	Total_Pizza_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

# PIZZA SALES SQL QUERIES

## K. Top 5 Pizzas by Total Orders

```
SELECT Top 5 pizza_name, COUNT(DISTINCT order_id) AS  
Total_Orders  
FROM pizza_sales  
GROUP BY pizza_name  
ORDER BY Total_Orders DESC
```

The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with five rows, each representing a pizza name and its total number of orders. The table has three columns: 'pizza\_name' (ranked 1 to 5), 'Total\_Orders' (values 2329, 2280, 2278, 2273, 2225), and a third column which is currently empty. The first row, 'The Classic Deluxe Pizza', is highlighted with a dashed border.

	pizza_name	Total_Orders
1	The Classic Deluxe Pizza	2329
2	The Hawaiian Pizza	2280
3	The Pepperoni Pizza	2278
4	The Barbecue Chicken Pizza	2273
5	The Thai Chicken Pizza	2225

# PIZZA SALES SQL QUERIES

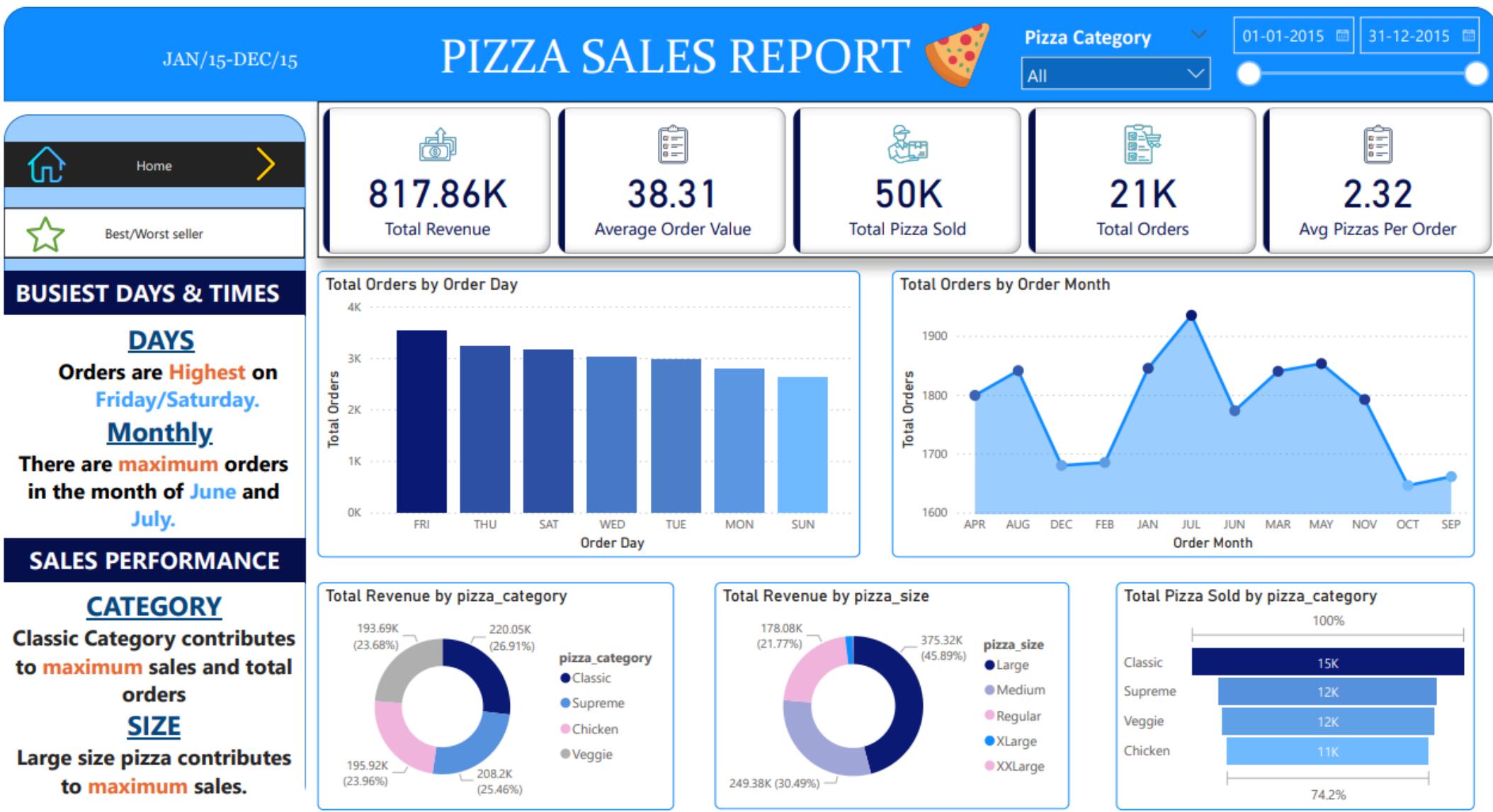
## L. Bottom 5 Pizzas by Total Orders

```
SELECT Top PIZZA_NAME, COUNT(DISTINCT order_id) AS  
TOTAL_QUANTITY FROM pizza_sales GROUP BY PIZZA_NAME  
ORDER BY TOTAL_QUANTITY ASC
```

The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with five rows. The table has two columns: 'PIZZA\_NAME' and 'TOTAL\_QUANTITY'. The data is as follows:

	PIZZA_NAME	TOTAL_QUANTITY
1	The Brie Carre Pizza	480
2	The Mediterranean Pizza	912
3	The Spinach Supreme Pizza	918
4	The Calabrese Pizza	918
5	The Chicken Pesto Pizza	938

# POWER BI DASHBOARD



# POWER BI DASHBOARD

JAN/15-DEC/15

## PIZZA SALES REPORT

Pizza Category: All | Date Range: 01-01-2015 - 31-12-2015

 Home

 Best/Worst seller >

**BUSIEST DAYS & TIMES**

**DAYS**  
Orders are **Highest** on **Friday/Saturday**.  
**Monthly**.

There are **maximum** orders in the month of **June** and **July**.

**SALES PERFORMANCE**

**CATEGORY**  
Classic Category contributes to **maximum** sales and total orders

**SIZE**  
Large size pizza contributes to **maximum** sales.

**817.86K** Total Revenue

**38.31** Average Order Value

**50K** Total Pizza Sold

**21K** Total Orders

**2.32** Avg Pizzas Per Order

**Top 5 Pizzas by Revenue**

pizza_name	Total Revenue
The Thai Chic...	~45K
The Barbecu...	~35K
The Californi...	~30K
The Classic D...	~25K
The Spicy Ital...	~15K

**Top 5 Pizzas by Quantity**

pizza_name	Total Pizza Sold
The Classic D...	~1800
The Barbecu...	~1500
The Hawaii...	~1200
The Peppero...	~1000
The Thai Chi...	~800

**Top 5 Pizzas by Total Orders**

pizza_name	Total Orders
The Classic D...	~2500
The Hawaiian...	~2000
The Pepper...	~1800
The Barbecue...	~1500
The Thai Chic...	~1200

**Bottom 5 Pizzas by Revenue**

pizza_name	Total Revenue
The Spinach ...	~15K
The Mediterr...	~12K
The Spinach ...	~10K
The Green G...	~8K
The Brie Carr...	~5K

**Bottom 5 Pizzas by Quantity**

pizza_name	Total Pizza Sold
The Soppres...	~900
The Spinach ...	~800
The Calabres...	~700
The Mediterr...	~600
The Brie Carr...	~500

**Bottom 5 Pizzas by Total Orders**

pizza_name	Total Orders
The Chicken ...	~2500
The Calabres...	~2000
The Spinach ...	~1800
The Mediterr...	~1500
The Brie Carr...	~1000

*Thank  
You*