

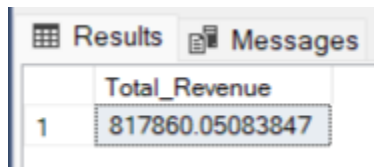
Pizza Sales SQL Queries

A.KPI'S

1. Total revenue:

The sum of the total price of all pizza orders

```
Select SUM(total_price) as Total_Revenue from pizza_sales;
```



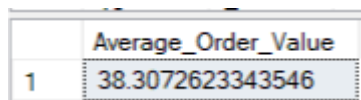
A screenshot of a SQL query results window. It has two tabs: 'Results' and 'Messages'. The 'Results' tab is active, showing a table with one column 'Total_Revenue' and one row with the value '817860.05083847'.

	Total_Revenue
1	817860.05083847

2. Average Order Value:

The average amount spent per order =total revenue / total number of orders

```
Select SUM(total_price) / COUNT(DISTINCT order_id) AS Average_Order_Value from pizza_sales
```



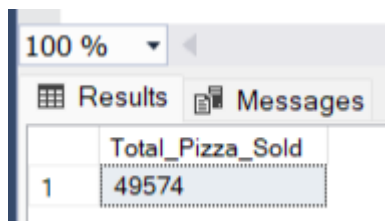
A screenshot of a SQL query results window. It shows a table with one column 'Average_Order_Value' and one row with the value '38.3072623343546'.

	Average_Order_Value
1	38.3072623343546

3. Total Pizzas Sold:

The sum of quantities of all pizzas sold

```
Select SUM(quantity) as Total_Pizza_Sold from pizza_sales;
```



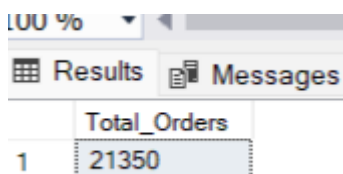
A screenshot of a SQL query results window. It has a zoom dropdown set to '100 %' and two tabs: 'Results' and 'Messages'. The 'Results' tab is active, showing a table with one column 'Total_Pizza_Sold' and one row with the value '49574'.

	Total_Pizza_Sold
1	49574

4. Total Orders:

The total numbers of orders placed

```
Select COUNT(distinct order_id) AS Total_Orders from pizza_sales;
```



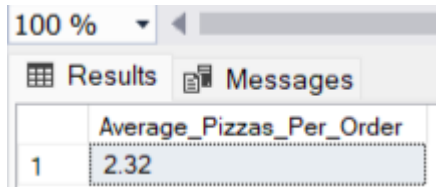
A screenshot of a SQL query results window. It has a zoom dropdown set to '100 %' and two tabs: 'Results' and 'Messages'. The 'Results' tab is active, showing a table with one column 'Total_Orders' and one row with the value '21350'.

	Total_Orders
1	21350

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

```
Select CAST(CAST(SUM(quantity) AS DECIMAL(10,2)) /  
CAST(count(distinct order_id) AS DECIMAL(10,2)) AS DECIMAL(10,2)) as  
Average_Pizzas_Per_Order FROM pizza_sales;
```



A screenshot of a SQL Server query results window. The window has a '100 %' zoom level and tabs for 'Results' and 'Messages'. The 'Results' tab is active, showing a single row with the column 'Average_Pizzas_Per_Order' and the value '2.32'.

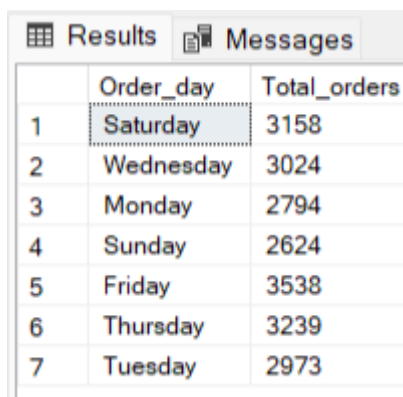
	Average_Pizzas_Per_Order
1	2.32

Chart requirement

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.

```
SELECT DATENAME(DW,order_date) as Order_day, count (distinct order_id) AS Total_orders  
from pizza_sales  
GROUP BY DATENAME(DW,order_date)
```



A screenshot of a SQL Server query results window. The window has tabs for 'Results' and 'Messages'. The 'Results' tab is active, showing a table with two columns: 'Order_day' and 'Total_orders'. The table contains seven rows of data for the days of the week.

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

2. Monthly Trends For Total Orders:

```
SELECT DATENAME(MONTH,order_date) as Month_name, count (distinct order_id) AS  
Total_orders from pizza_sales  
GROUP BY DATENAME(MONTH,order_date)  
ORDER BY Total_orders desc
```

100 %

Results	Messages
---------	----------

	Month_name	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

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.

```
select pizza_category, SUM (total_price) as Total_sales, SUM(total_price) * 100/
(select sum(total_price) from pizza_sales) as pct from pizza_sales
group by pizza_category
```

Results Messages

	pizza_category	Total_sales	pct
1	Classic	220053.100021362	26.9059602306976
2	Chicken	195919.5	23.9551375322885
3	Veggie	193690.451004028	23.6825910258677
4	Supreme	208196.99981308	25.4563112111462

4. Percentage Of 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.

```
select pizza_size, SUM (total_price) as Total_sales, SUM(total_price) * 100/ (select
sum(total_price) from pizza_sales) as pct from pizza_sales
group by pizza_size
```

100 %

	pizza_size	Total_sales	pct
1	L	375318.701004028	45.8903330244889
2	XXL	1006.6000213623	0.123077294254725
3	M	249382.25	30.492044420599
4	XL	14076	1.72107684995364
5	S	178076.49981308	21.7734684107037

With additional codes

```
select pizza_size,CAST(SUM (total_price) AS DECIMAL(10,2)) as Total_sales,
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 DESC
```

	pizza_size	Total_sales	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

With where clause

```
select pizza_size,CAST(SUM (total_price) AS DECIMAL(10,2)) as Total_sales,
CAST(SUM(total_price) * 100/ (select sum(total_price) from pizza_sales where
DATEPART(quarter,order_date)=1) AS DECIMAL(10,2)) as pct from pizza_sales
where DATEPART(quarter,order_date)=1
group by pizza_size
ORDER BY PCT DESC
```

	pizza_size	Total_sales	pct
1	L	95229.65	46.37
2	M	61159.00	29.78
3	S	45384.25	22.10
4	XL	3289.50	1.60
5	XXL	287.60	0.14

5. 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 and Total Orders. This chart will help us identify the most popular pizza options.

TOTAL REVENUE

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

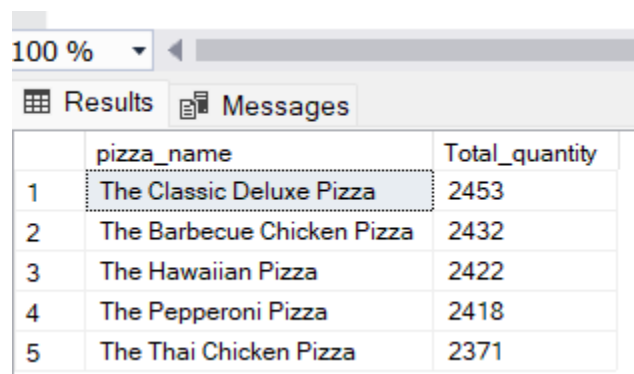


The screenshot shows a SQL Server query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'pizza_name' and 'Total_Revenue'. The table contains five rows of data, ordered by Total_Revenue in descending order. The first row is 'The Thai Chicken Pizza' with a revenue of 43434.25. The second row is 'The Barbecue Chicken Pizza' with a revenue of 42768. The third row is 'The California Chicken Pizza' with a revenue of 41409.5. The fourth row is 'The Classic Deluxe Pizza' with a revenue of 38180.5. The fifth row is 'The Spicy Italian Pizza' with a revenue of 34831.25.

	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

TOTAL QUANTITY

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

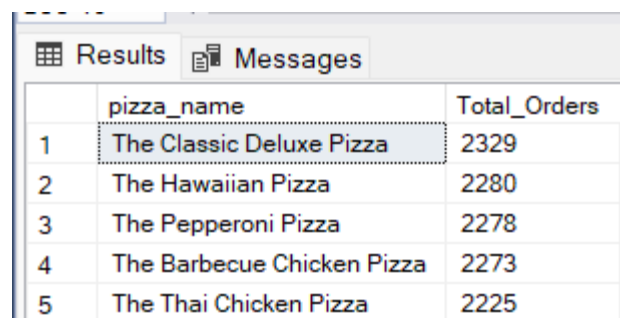


The screenshot shows a SQL Server query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'pizza_name' and 'Total_quantity'. The table contains five rows of data, ordered by Total_quantity in descending order. The first row is 'The Classic Deluxe Pizza' with a quantity of 2453. The second row is 'The Barbecue Chicken Pizza' with a quantity of 2432. The third row is 'The Hawaiian Pizza' with a quantity of 2422. The fourth row is 'The Pepperoni Pizza' with a quantity of 2418. The fifth row is 'The Thai Chicken Pizza' with a quantity of 2371.

	pizza_name	Total_quantity
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

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 Server query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'pizza_name' and 'Total_Orders'. The table contains five rows of data, ordered by Total_Orders in descending order. The first row is 'The Classic Deluxe Pizza' with 2329 orders. The second row is 'The Hawaiian Pizza' with 2280 orders. The third row is 'The Pepperoni Pizza' with 2278 orders. The fourth row is 'The Barbecue Chicken Pizza' with 2273 orders. The fifth row is 'The Thai Chicken Pizza' with 2225 orders.

	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

6. 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 and Total Orders. This chart will help us to identify underperforming or less popular pizza option.

TOTAL REVENUE

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

Results Messages		
	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

TOTAL QUANTITY

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

Results Messages		
	pizza_name	Total_quantity
1	The Brie Carre Pizza	490
2	The Mediterranean Pizza	934
3	The Calabrese Pizza	937
4	The Spinach Supreme Pizza	950
5	The Soppresata Pizza	961

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 ASC
```

Results		Messages
	pizza_name	Total_Orders
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

NOTES

If you want to apply the MONTH,QUARTER,WEEK filters to the above queries you can use where clause. Follow some of below examples

```
select pizza_category,SUM (total_price) as Total_sales, SUM(total_price) * 100/
(select sum(total_price) from pizza_sales where month(order_date)=1) as pct from
pizza_sales
where month(order_date)=1
group by pizza_category
```

*Here Month(order_date) =1 indicates that the output is for the month of January.

Month(order_date) =4 indicates that the output is for the month of April.

```
SELECT DATENAME(DW,order_date) as Order_day, count (distinct order_id) AS Total_orders
from pizza_sales
WHERE DATEPART(QUARTER, order_date)=1
GROUP BY DATENAME(DW,order_date)
```

*Here DATEPART(QUARTER, order_date)=1 indicates that the output is for the quarter 1.

DATEPART(QUARTER, order_date)=3 indicates that the output is for the quarter 3.

Power BI

KPI'S

1. Total Revenue= Total Revenue = `SUM(pizza_sales[total_price])`
2. Total Orders= Total Orders = `DISTINCTCOUNT(pizza_sales[order_id])`
3. Average Order Value= Average Order Value = `[Total Revenue]/[Total Orders]`
4. Total Pizzas Sold =Total Pizzas Sold = `sum(pizza_sales[quantity])`
5. Average pizzas per Order= Average Pizzas Per Order = `[Total Pizzas Sold]/[Total Orders]`

