

PIZZA Query Documentation

a. Total Revenue :

```
select sum(total_price) as Total_Revenue from pizza_sales;
```

Results		Messages
Total_Revenue		
1	817860.05083847	

b. Average order Value :

```
select sum(total_price) / count(distinct(order_id)) as Avg_order_value from pizza_sales;
```

100 %		Results	Messages
		Avg_order_value	
1	38.3072623343546		

c. Total Quantity Sold :

```
select sum(quantity) as Total_Pizza_Sold from pizza_sales;
```

100 %		Results	Messages
		Total_Pizza_Sold	
1	49574		

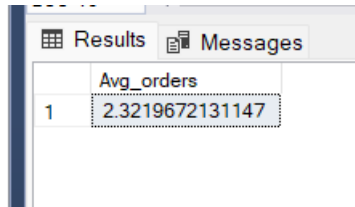
d. Total Orders Sold :

```
select count(distinct order_id) as total_orders from pizza_sales;
```

100 %		Results	Messages
		total_orders	
1	21350		

e. Avg Order Sold :

```
select cast( sum( quantity) as decimal(10,2)) / cast(count(distinct(order_id)) as decimal(10,2)) as Avg_orders from pizza_sales;
```



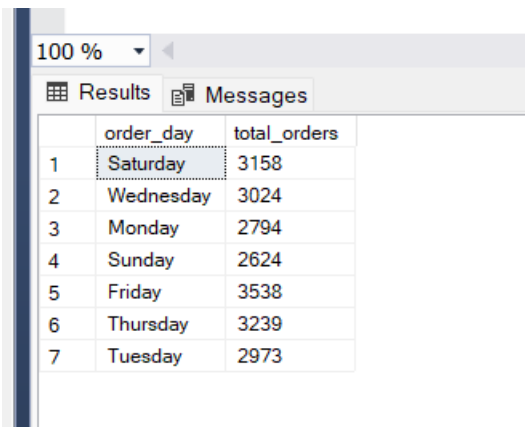
The screenshot shows a SQL Server query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a single row of data. The column is labeled 'Avg_orders' and the value is 2.3219672131147.

	Avg_orders
1	2.3219672131147

2. Total Orders wise Trends

a. Daily Trend of Orders :

```
select DATENAME(DW, order_date) as order_day, count(distinct(order_id)) as total_orders  
from pizza_sales  
group by DATENAME(DW, order_date);
```



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: 'order_day' and 'total_orders'. The table contains seven rows of data, one for each day of the week. The 'order_day' column is highlighted with a blue background.

	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

b. Monthly Trend of Orders :

```
select DATENAME(MM, order_date) as order_day, count(distinct(order_id)) as total_orders  
from pizza_sales  
group by DATENAME(MM, order_date);
```

100 %

Results Messages

	order_day	total_orders
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

c. Percentage of sales by Pizza Category :

-- Here we are figuring out the PCT for the pizza category and also added another condition of month wise 1 as for January.

```
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
Percentage_of_sales
from pizza_sales
where MONTH(order_date) = 1
group by pizza_category;
```

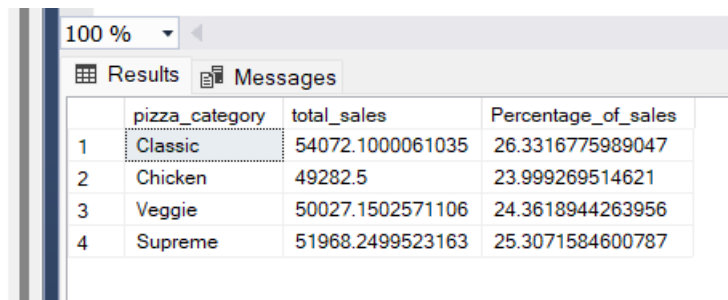
100 %

Results Messages

	pizza_category	total_sales	Percentage_of_sales
1	Classic	18619.4000015259	26.6779189176038
2	Chicken	16188.75	23.1952780348435
3	Veggie	17055.4000778198	24.4370162489706
4	Supreme	17929.7499866486	25.6897867985821

-- Here we are figuring out the PCT for the pizza category and also added another condition of quarter wise 1.

```
select pizza_category, sum(total_price) as total_sales, sum(total_price) * 100 /  
(Select sum(total_price) from pizza_sales where DATEPART(QUARTER, order_date) = 1) as  
Percentage_of_sales  
from pizza_sales  
where DATEPART(QUARTER, order_date) = 1  
group by pizza_category;
```

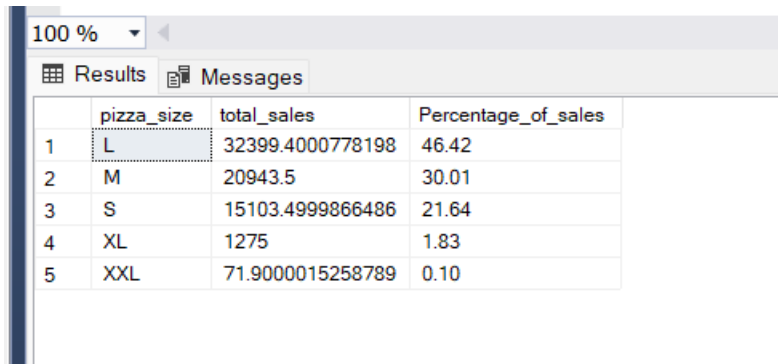


The screenshot shows a SQL Server query results window with a dropdown menu set to 100%. The window has two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with four columns: 'pizza_category', 'total_sales', and 'Percentage_of_sales'. The table contains four rows of data, numbered 1 to 4.

	pizza_category	total_sales	Percentage_of_sales
1	Classic	54072.1000061035	26.3316775989047
2	Chicken	49282.5	23.999269514621
3	Veggie	50027.1502571106	24.3618944263956
4	Supreme	51968.2499523163	25.3071584600787

d. Percentage of sales by Pizza Size :

```
select pizza_size, sum(total_price) as total_sales, cast( sum(total_price) * 100 /  
(Select sum(total_price) from pizza_sales where MONTH(order_date) = 1) as  
decimal(10,2)) as Percentage_of_sales  
from pizza_sales  
where MONTH(order_date) = 1  
group by pizza_size  
order by pizza_size;
```

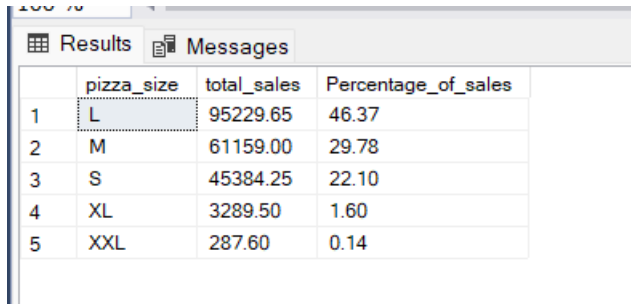


The screenshot shows a SQL Server query results window with a dropdown menu set to 100%. The window has two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with four columns: 'pizza_size', 'total_sales', and 'Percentage_of_sales'. The table contains five rows of data, numbered 1 to 5.

	pizza_size	total_sales	Percentage_of_sales
1	L	32399.4000778198	46.42
2	M	20943.5	30.01
3	S	15103.4999866486	21.64
4	XL	1275	1.83
5	XXL	71.9000015258789	0.10

-- Quarter Wise of Pizza Size Sales

```
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 Percentage_of_sales
from pizza_sales
where Datepart(quarter, order_date) = 1
group by pizza_size
order by pizza_size;
```



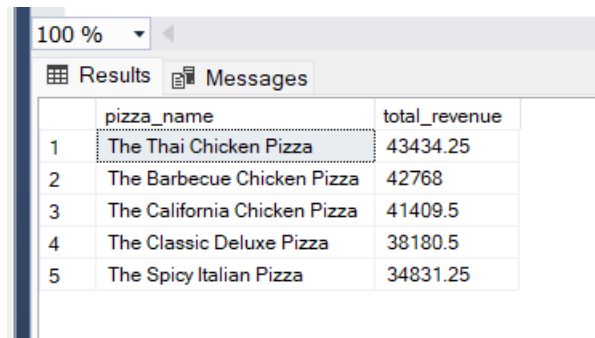
The screenshot shows a SQL Server query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with three columns: 'pizza_size', 'total_sales', and 'Percentage_of_sales'. The table contains five rows of data, ordered by pizza size. The first row is for size 'L' with a total sales of 95229.65 and a percentage of 46.37. The second row is for size 'M' with a total sales of 61159.00 and a percentage of 29.78. The third row is for size 'S' with a total sales of 45384.25 and a percentage of 22.10. The fourth row is for size 'XL' with a total sales of 3289.50 and a percentage of 1.60. The fifth row is for size 'XXL' with a total sales of 287.60 and a percentage of 0.14.

	pizza_size	total_sales	Percentage_of_sales
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

Top 5 Best Sellers by Revenue, Total Quantity and Total Orders

:

```
select TOP 5 pizza_name, sum(total_price) as total_revenue from pizza_sales
group by pizza_name
order by total_revenue desc;
```

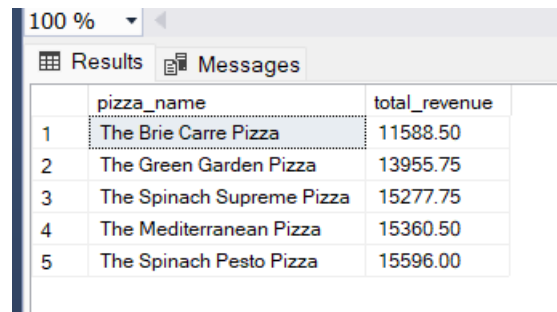


The screenshot shows a SQL Server query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'pizza_name' and 'total_revenue'. The table contains five rows of data, ordered by total revenue in descending order. The first row is for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

-- Bottom 5 Pizza

```
select TOP 5 pizza_name, cast(sum(total_price) as decimal(10,2)) as total_revenue
from pizza_sales
group by pizza_name
order by total_revenue;
```

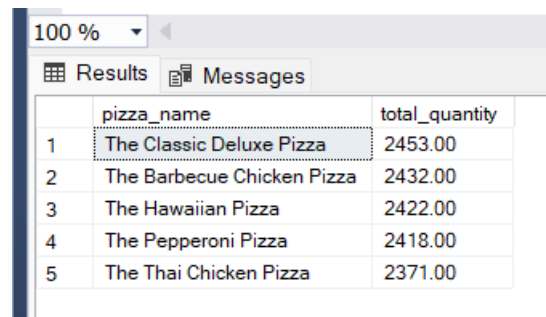


100 %

	pizza_name	total_revenue
1	The Brie Carre Pizza	11588.50
2	The Green Garden Pizza	13955.75
3	The Spinach Supreme Pizza	15277.75
4	The Mediterranean Pizza	15360.50
5	The Spinach Pesto Pizza	15596.00

-- Top 5 Pizza By quantity

```
select TOP 5 pizza_name, sum(quantity) as total_quantity from pizza_sales
group by pizza_name
order by total_quantity desc;
```

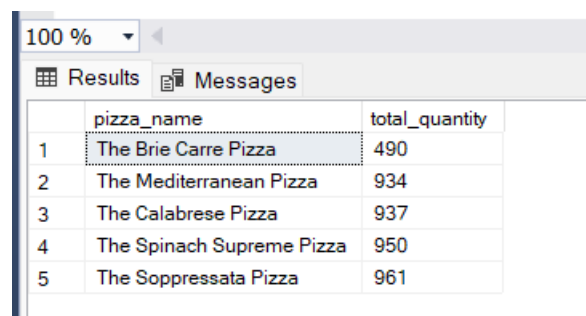


100 %

	pizza_name	total_quantity
1	The Classic Deluxe Pizza	2453.00
2	The Barbecue Chicken Pizza	2432.00
3	The Hawaiian Pizza	2422.00
4	The Pepperoni Pizza	2418.00
5	The Thai Chicken Pizza	2371.00

-- Bottom 5 Pizza By quantity

```
select TOP 5 pizza_name, sum(quantity) as total_quantity from pizza_sales
group by pizza_name
order by total_quantity;
```

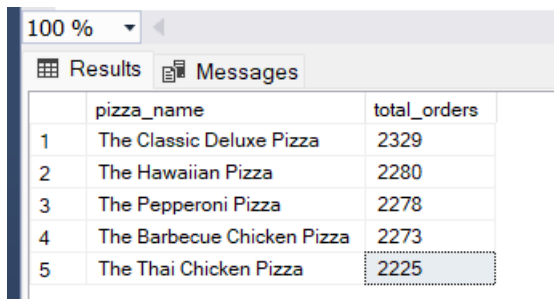


100 %

	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 Soppressata Pizza	961

-- Top 5 best sellers

```
select TOP 5 pizza_name, count(distinct(order_id)) as total_orders from pizza_sales
group by pizza_name
order by total_orders desc;
```

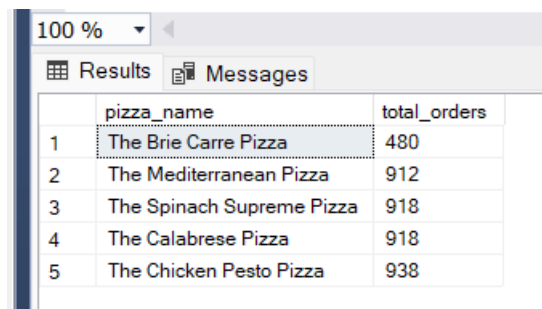


The screenshot shows a SQL Server query results window. At the top, there is a zoom level of 100% and two tabs: 'Results' (active) and 'Messages'. Below the tabs is a table with two columns: 'pizza_name' and 'total_orders'. The table contains five rows, numbered 1 to 5. The first row is 'The Classic 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. The fifth row is highlighted with a blue background.

	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

-- Worst 5 Pizza

```
select TOP 5 pizza_name, count(distinct(order_id)) as total_orders from pizza_sales
group by pizza_name
order by total_orders;
```



The screenshot shows a SQL Server query results window. At the top, there is a zoom level of 100% and two tabs: 'Results' (active) and 'Messages'. Below the tabs is a table with two columns: 'pizza_name' and 'total_orders'. The table contains five rows, numbered 1 to 5. The first row is 'The Brie Carre Pizza' with 480 orders. The second row is 'The Mediterranean Pizza' with 912 orders. The third row is 'The Spinach Supreme Pizza' with 918 orders. The fourth row is 'The Calabrese Pizza' with 918 orders. The fifth row is 'The Chicken Pesto Pizza' with 938 orders. The first row is highlighted with a blue background.

	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