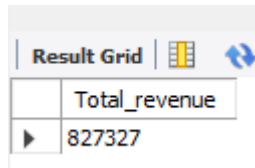


**1. The sum of the total price of all pizza orders.**

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

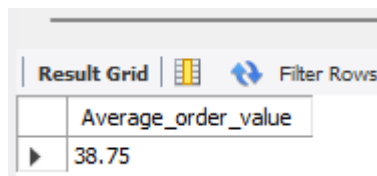


The screenshot shows a SQL query result grid with the following data:

	Total_revenue
▶	827327

**2. The average amount spent 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 FROM pizza_sales;
```

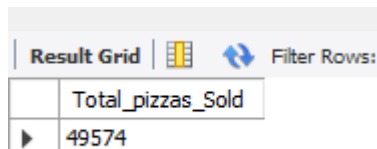


The screenshot shows a SQL query result grid with the following data:

	Average_order_value
▶	38.75

**3. The sum of the quantities of all pizzas sold.**

```
SELECT SUM(quantity) AS Total_pizzas_Sold FROM pizza_sales;
```

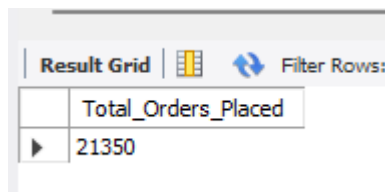


The screenshot shows a SQL query result grid with the following data:

	Total_pizzas_Sold
▶	49574

**4. The total number of orders placed.**

```
SELECT COUNT(DISTINCT order_id) AS Total_Orders_Placed FROM pizza_sales;
```



The screenshot shows a SQL query result grid with the following data:

	Total_Orders_Placed
▶	21350

5. The average number of pizzas sold per order, calculated by dividing the total number of pizzas sold by the total number of orders.

```
SELECT ROUND(SUM(quantity) / COUNT(DISTINCT order_id), 2) AS AVG_Pizzas_Per_Order FROM pizza_sales;
```

Result Grid		Filter Rows:
	Total_Numbers_Placed_Sold	
▶	2.32	

6. Hourly Trend for Total Pizzas Sold:

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



```
SELECT HOUR(order_time) AS Hour, SUM(quantity) AS Total_pizzas_Sold  
  
FROM pizza_sales  
  
GROUP BY HOUR(order_time)  
  
ORDER BY HOUR(order_time);
```

Result Grid		Filter Rows:
	Hour	Total_pizzas_Sold
▶	9	4
	10	18
	11	2728
	12	6776
	13	6413
	14	3613
	15	3216
	16	4239
	17	5211
	18	5417
	19	4406
	20	3534
	21	2545
	22	1386
	23	68

## 7. Weekly Trend for Total Orders:

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

```
SELECT WEEK(order_date) AS WEEK,  
  
       YEAR(order_date) AS YEAR,  
  
       COUNT(DISTINCT order_id) AS Total_orders  
  
FROM pizza_sales  
  
GROUP BY WEEK(order_date), YEAR(order_date)  
  
ORDER BY WEEK(order_date), YEAR(order_date);
```

Result Grid   Filter Rows:			
	WEEK	YEAR	Total_orders
▶	0	2015	202
	1	2015	427
	2	2015	401
	3	2015	422
	4	2015	393
	5	2015	456
	6	2015	421
	7	2015	411
	8	2015	397
	9	2015	407
	10	2015	417
	11	2015	428
	12	2015	410
	13	2015	437
	14	2015	405
	15	2015	417

## 8. 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 AS Category, SUM(total_price) AS Total_Price, SUM(total_price) / (SELECT SUM(total_price) FROM pizza_sales) * 100 AS PCT
```

```
FROM pizza_sales
```

```
GROUP BY Category;
```

Result Grid			
Filter Rows:			
	Category	Total_Price	PCT
▶	Classic	222979	26.9517
	Veggie	194653	23.5279
	Supreme	211019	25.5061
	Chicken	198676	24.0142

## 9. 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.

```
SELECT pizza_size,
```

```
SUM(total_price) AS Total_Price,
```

```
ROUND(SUM(total_price) / (SELECT SUM(total_price) FROM pizza_sales) * 100, 2) AS PCT
```

```
FROM pizza_sales
```

```
GROUP BY pizza_size
```

```
ORDER BY PCT DESC;
```

Result Grid			
Filter Rows:			
	pizza_size	Total_Price	PCT
▶	L	378992	45.81
	M	252436	30.51
	S	180547	21.82
	XL	14344	1.73
	XXL	1008	0.12



#### 10. 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.

```
SELECT pizza_category, SUM(quantity) AS Total_Pizzas_Sold
```



```
FROM pizza_sales
```

```
GROUP BY pizza_category;
```

Result Grid     Filter Rows: <input type="text"/>		
	pizza_category	Total_Pizzas_Sold
▶	Classic	14888
	Veggie	11649
	Supreme	11987
	Chicken	11050

#### 11. 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.

Result Grid     Filter Rows: <input type="text"/>		
	pizza_name	Revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5
	The Classic Deluxe Pizza	38180.5
	The Spicy Italian Pizza	34831.25

#### 12. . 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.