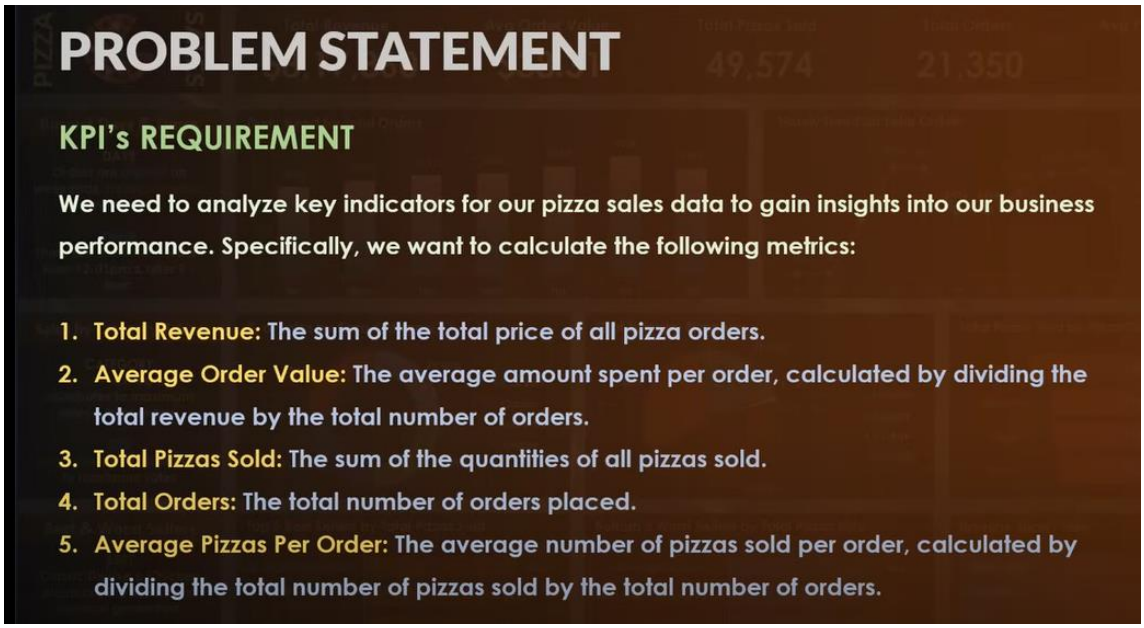
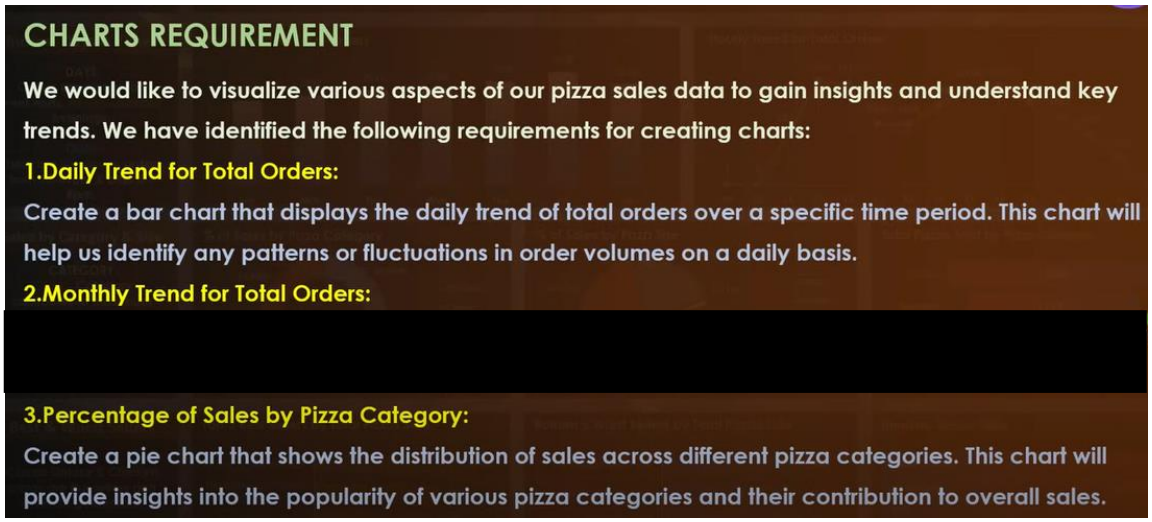
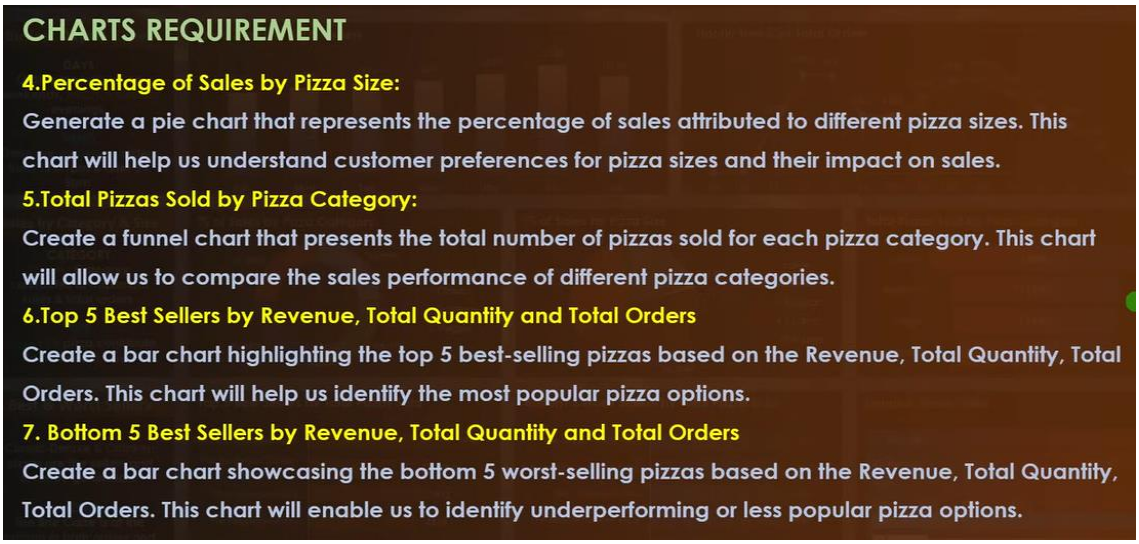
PIZZA SALES SQL QUERIES

A. KPI's

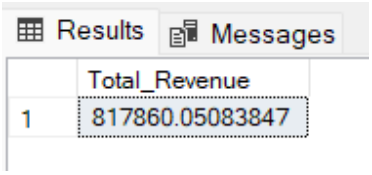






1. Total Revenue:

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

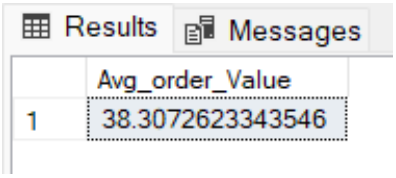


Results Messages

	Total_Revenue
1	817860.05083847

2. Average Order Value

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

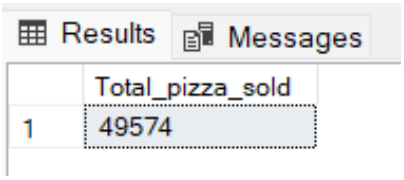


Results Messages

	Avg_order_Value
1	38.3072623343546

3. Total Pizzas Sold

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

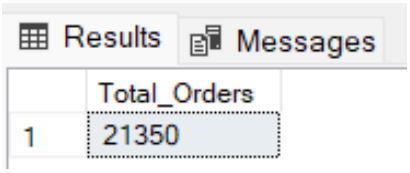


Results Messages

	Total_pizza_sold
1	49574

4. Total Orders

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

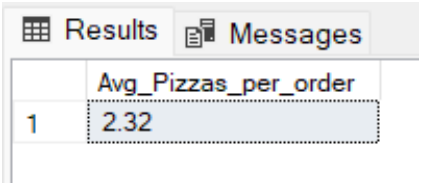


Results Messages

	Total_Orders
1	21350

5. Average Pizzas Per Order

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



Results Messages

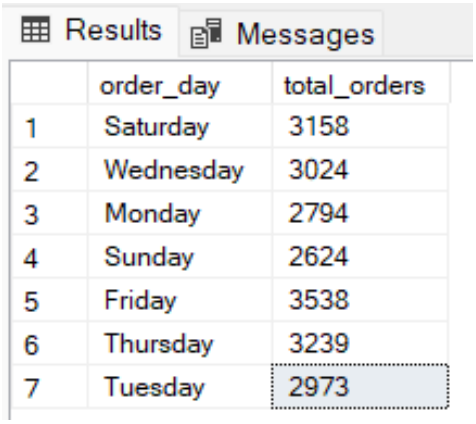
	Avg_Pizzas_per_order
1	2.32

In SQL, when you perform arithmetic operations on integers, the result is also treated as an integer. To get a result with non-zero decimal places, you need to ensure that at least one of the operands is of a decimal or floating-point type. You can achieve this by casting one of the operands to a decimal before performing the division.

B. Daily Trend for Total 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)
```

Output:



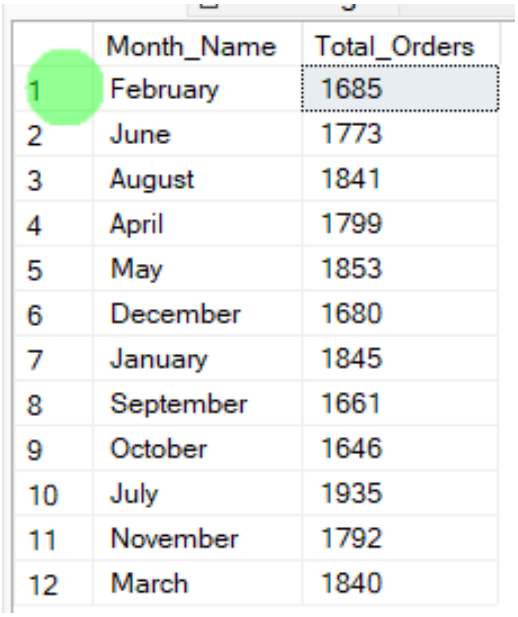
Results | Messages

	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

C. Monthly Trend for 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)
```

Output



	Month_Name	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

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
```

Output

Results Messages

	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

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 DESC;
```

Output

Results Messages

	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

F. Total Pizzas Sold by Pizza Category

```sql
SELECT pizza_category, SUM(quantity) AS Total_Quantity_Sold
FROM pizza_sales
--WHERE MONTH(order_date) = 2
GROUP BY pizza_category
ORDER BY Total_Quantity_Sold DESC;
```

Output

Results Messages

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

G. Top 5 Pizzas by Revenue

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

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

H. Bottom 5 Pizzas by Revenue

```sql
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

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;
```

Output

	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

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;
```

Output

Results Messages

	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

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;
```

Results | Messages

	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

L. Borrom 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 ASC;
```

	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

NOTE

To apply the pizza_category or pizza_size filters to the above queries:

```
SELECT Top 5 pizza_name, COUNT(DISTINCT order_id) AS Total_Orders
FROM pizza_sales
WHERE pizza_category = 'Classic'
GROUP BY pizza_name
ORDER BY Total_Orders ASC
```