



My name is **BABAFEMI, OLUWASEYI GRACE**. I'm a junior data analyst. I wanted to implement my data analysis skill and I decided to analyse a pizza sales dataset. I chose pizza because it's a well-known Fast-food with high purchasing rate and everybody, well almost everybody loves pizza.

I got my dataset from Kaggle which also came with some vital questions for insight, and I also added mine to spice things up.

PROBLEM STATEMENT

KPI's Requirement: I generated some specific metrics for calculations, this allows me to analyse key indicators for the pizza sales data to gain insight into business performance.

1. Total Revenue
2. Average order value
3. Total pizzas sold
4. Average pizzas per order

Charts Requirement: I would like to visualise various part of the pizza sales data to gain insight and understand key trends. I have identified the following requirements for creating charts.

1. Daily trend of total order.
2. Monthly trend of total orders.
3. Percentage of sales by pizza category.
4. Percentage of sales by pizza size.
5. Total pizza sold by pizzas category.
6. Top 5 best sellers by Revenue, Total Quantity and Total Orders.
7. Bottom 5 best sellers by Revenue, Total Quantity and Total Orders.
8. Top 5 Pizzas by Quantity
9. Bottom 5 Pizzas by Quantity
10. Top 5 Pizzas by Total Orders
11. Bottom 5 Pizzas by Total Orders

The Data

The dataset was in a form of a csv file (comma separated value). Here is an image of the original data (first 10 rows):

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	order_details_id	order_id	pizza_id	quantity	order_date	order_time	unit_price	total_price	pizza_size	pizza_category	pizza_ingredients	pizza_name						
2	1,1	hawaiian_m	1,1	1/2015,11:38:36	13.25	13.25	M,Classic	"Sliced Ham, Pineapple, Mozzarella Cheese"				The Hawaiian Pizza						
3	2,2	classic_dlx_m	1,1	1/2015,11:57:40	16,16	M,Classic	"Pepperoni, Mushrooms, Red Onions, Red Peppers, Bacon"					The Classic Deluxe Pizza						
4	3,2	five_cheese_l	1,1	1/2015,11:57:40	18.5	18.5	L,Veggie	"Mozzarella Cheese, Provolone Cheese, Smoked Gouda Cheese, Romano Cheese, Blue Cheese, Garlic"				The Five Cheese Pizza						
5	4,2	ital_supr_l	1,1	1/2015,11:57:40	20.75	20.75	L,Supreme	"Calabrese Salami, Capocollo, Tomatoes, Red Onions, Green Olives, Garlic"				The Italian Supreme Pizza						
6	5,2	mexicana_m	1,1	1/2015,11:57:40	16,16	M,Veggie	"Tomatoes, Red Peppers, Jalapeno Peppers, Red Onions, Cilantro, Corn, Chipotle Sauce, Garlic"					The Mexicana Pizza						
7	6,2	thai_chn_l	1,1	1/2015,11:57:40	20.75	20.75	L,Chicken	"Chicken, Pineapple, Tomatoes, Red Peppers, Thai Sweet Chilli Sauce"				The Thai Chicken Pizza						
8	7,3	ital_supr_m	1,1	1/2015,12:12:28	16.5	16.5	M,Supreme	"Calabrese Salami, Capocollo, Tomatoes, Red Onions, Green Olives, Garlic"				The Italian Supreme Pizza						
9	8,3	prsc_argla_l	1,1	1/2015,12:12:28	20.75	20.75	L,Supreme	"Prosciutto di San Daniele, Arugula, Mozzarella Cheese"				The Prosciutto and Arugula Pizza						
10	9,4	ital_supr_m	1,1	1/2015,12:16:31	16.5	16.5	M,Supreme	"Calabrese Salami, Capocollo, Tomatoes, Red Onions, Green Olives, Garlic"				The Italian Supreme Pizza						

There are 11 total columns:

- pizza_id
- order_id
- quantity
- order_date
- order_time
- unit_price
- total_price
- pizza_size
- pizza_category
- pizza_ingredients
- pizza_name

KPI's Requirement:

1. Total revenue: The sum of all total price of all pizza orders.

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


100 %	Results	Messages
	Total_Revenue	
1	817860.05083847	

2. Average Order Value: Calculated by dividing the total revenue by the total number of orders.

```
SELECT AVG(total_price) AS average_order_value
FROM pizza_sales
    SELECT order_id, SUM(unit_price) AS total_order_value
    FROM pizza_sales
    GROUP BY order_id
ORDER BY order_id
```

Results		Messages
	average_order_value	
1	16.8214736906308	

	order_id	total_order_value
1	1	13.25
2	2	92
3	3	37.25
4	4	16.5
5	5	16.5
6	6	24.75
7	7	12.5
8	8	12.5
9	9	143.25

 Query executed successfully.

1. The inner subquery groups the rows by "order_id" and calculates the sum of "unit_price" for each unique order.
2. The outer query then calculates the average of the "total_price" calculated in the subquery, giving me the AOV.

This query will give me the average order value considering all the unique order IDs, even if there are duplicate order IDs with different unit price.

3. Total Pizzas Sold: This is the sum of the quantities of all pizzas sold.

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

Results		Messages
	Total_pizza_sold	
1	49574	

4. Total Orders: This is the total number of orders placed.

```
SELECT COUNT(DISTINCT order_id) AS total_orders
FROM pizza sales
```

100 %	
Results	Messages
total_orders	
1	21350

To calculate the total number of orders when my order_id has duplicate numbers but different purchases, I performed a COUNT of distinct order_id values to ensure I am not double-counting orders with the same ID.

5. Average Pizza sold:

Charts Requirement

1. Daily trend of total order.

```
SELECT DATEPART(HOUR, order_time) as order_hours, SUM(quantity) as
total_pizzas_sold
from pizza_sales
group by DATEPART(HOUR, order_time)
order by DATEPART(HOUR, order_time)
```

Output

	order_hours	total_pizzas_sold
1	9	4
2	10	18
3	11	2728
4	12	6776
5	13	6413
6	14	3613
7	15	3216
8	16	4239
9	17	5211
10	18	5417
11	19	4406
12	20	3534
13	21	2545
14	22	1386
15	23	68

2. Monthly trend of total orders.

```
SELECT DATEPART(ISO_WEEK, order_date) AS WeekNumber, YEAR(order_date) AS Year, COUNT(DISTINCT order_id) AS Total_orders
FROM pizza_sales
GROUP BY DATEPART(ISO_WEEK, order_date), YEAR(order_date)
ORDER BY Year, WeekNumber;
```

	WeekNumber	Year	Total_orders
1	1	2015	254
2	2	2015	427
3	3	2015	400
4	4	2015	415
5	5	2015	436
6	6	2015	422
7	7	2015	423
8	8	2015	393
9	9	2015	409
10	10	2015	420
11	11	2015	404
12	12	2015	416
13	13	2015	427
14	14	2015	433
15	15	2015	408
16	16	2015	414
17	17	2015	437
18	18	2015	423
19	19	2015	399
20	20	2015	458
21	21	2015	414
22	22	2015	390
23	23	2015	423
24	24	2015	418
25	25	2015	410
26	26	2015	416
27	27	2015	474

28	28	2015	417
29	29	2015	420
30	30	2015	433
31	31	2015	419
32	32	2015	426
33	33	2015	435
34	34	2015	407
35	35	2015	394
36	36	2015	397
37	37	2015	435
38	38	2015	423
39	39	2015	288
40	40	2015	433
41	41	2015	334
42	42	2015	386
43	43	2015	352
44	44	2015	371
45	45	2015	394
46	46	2015	400
47	47	2015	392
48	48	2015	491
49	49	2015	424
50	50	2015	417
51	51	2015	430
52	52	2015	298
53	53	2015	171

3. Percentage 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
```

	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

4. Percentage 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 pizza_size
```

	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

5.Total pizza sold by pizzas category.

```
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
```

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

6. 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
```

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

7. Bottom 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 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

8. 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

9. 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
```

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 Soppresata Pizza	961

10. 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

11. Bottom 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
```

Results Messages		
	pizza_name	Total_Orders
1	The Brie Carré Pizza	480
2	The Mediterranean Pizza	912
3	The Spinach Supreme Pizza	918
4	The Calabrese Pizza	918
5	The Chicken Pesto Pizza	938