Pizza Sales Analysis Using MySQL®

By: Sushant Jha

Introduction

- Leveraging SQL for pizza sales analysis offers significant business intelligence. We can glean customer preferences, identify high-demand items, and uncover temporal sales trends.
- Furthermore, by constructing a data dashboard, stakeholders gain a visually compelling and readily interpretable means to grasp these insights.

EXCEL DASHBOARD

SQL ANALYSIS

Details of the Project

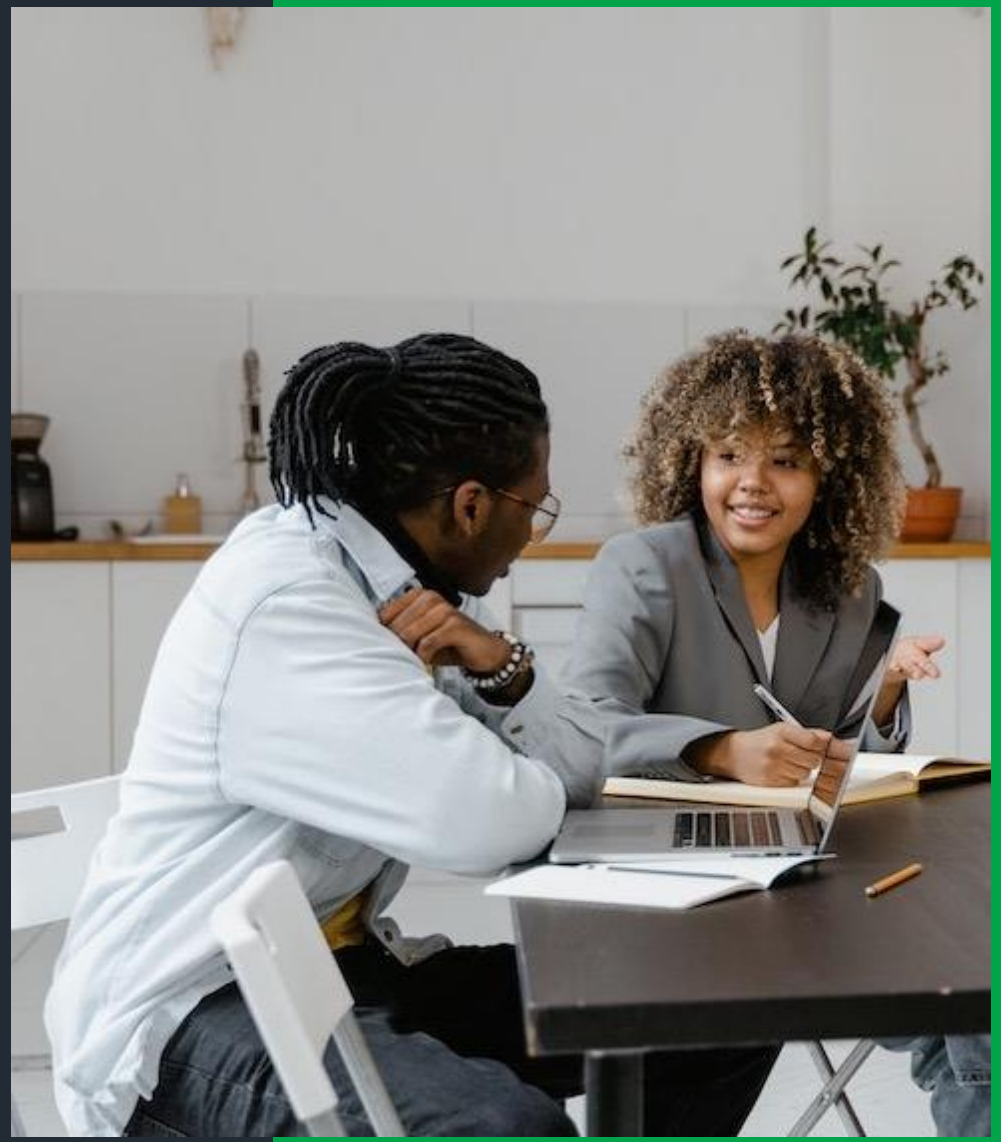
Source of Data

- The data used is sourced from Kaggle published by Nehar Tiwari.
- Link: https://www.kaggle.com/datasets/miniyadav/pizza-sales-case-study

Dataset Information

The dataset contains 4 csv files, name as follows with the following columns:

- orders.csv has columns : order_id, date, time
- order_details.csv has columns : order_details_id, order_id, pizza_id, quantity
- pizza_types.csv has columns : pizza_type_id, name, category, ingredients
- pizzas.csv has columns : pizza_id, pizza_type_id, size, price



kaggle

Details of the Project

KPI's

1. **Total Revenue:** This metric represents the sum of the total price for all pizza orders within the specified timeframe.
2. **Average Order Value:** This KPI reflects the average amount spent per order. It is calculated by dividing the total revenue by the total number of orders.
3. **Total Pizzas Sold:** This metric represents the total number of pizzas sold within the specified timeframe.
4. **Total Orders:** This KPI reflects the total number of orders placed within the specified timeframe.
5. **Average Pizzas per Order:** This metric indicates the average number of pizzas sold per order. It is calculated by dividing the total pizzas sold by the total number of orders.

By monitoring these KPIs, we can gain a comprehensive understanding of our pizza sales performance and identify areas for improvement.

Let’s Start With
SQL Queries

Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Result Grid

	total_orders
▶	21350

Calculate the total revenue generated from pizza sales.

```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
ORDER BY total_revenue DESC;
```

Result Grid

total_revenue
817860.05

Calculate the total pizza sold.

```sql
SELECT
    SUM(quantity) AS Total_Pizza_Sold
FROM
    order_details;
```

Result Grid

	Total_Pizza_Sold
▶	49574

Identify the highest-priced pizza.

```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

Result Grid | Filter Ro

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

```sql
SELECT
    pz.size AS pizza_size,
    COUNT(pz.size) AS ordered_by_people,
    sum(od.quantity) as Total_Ordered
FROM
    order_details AS od
        JOIN
    pizzas AS pz
    ON od.pizza_id = pz.pizza_id
GROUP BY pizza_size
ORDER BY Total_Ordered DESC
limit 1;
```

Result Grid | Filter Rows:

pizza_size	ordered_by_people	Total_Ordered
L	18526	18956

Calculate the percentage sales by pizza size

```sql
WITH cte AS (
  SELECT
    pz.size AS pizza_size,
    ROUND(SUM(od.quantity * pz.price), 2) AS total_price
  FROM
    pizza_types AS pzt
  JOIN
    pizzas AS pz ON pzt.pizza_type_id = pz.pizza_type_id
  JOIN
    order_details AS od ON od.pizza_id = pz.pizza_id
  GROUP BY pz.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 cte) AS DECIMAL(10,2)) AS PCT
FROM cte
GROUP BY pizza_size
ORDER BY pizza_size;
```

pizza_size	Total_Revenue	PCT
L	375318.70	45.89
M	249382.25	30.49
S	178076.50	21.77
XL	14076.00	1.72
XXL	1006.60	0.12

Calculate the total pizza sold by each pizza category.

```sql
SELECT
    pzt.category AS 'pizza_category',
    SUM(od.quantity) AS Total_Quantity_Sold
FROM
    pizza_types pzt
        JOIN
    pizzas pz ON pzt.pizza_type_id = pz.pizza_type_id
        JOIN
    order_details od ON pz.pizza_id = od.pizza_id
        JOIN
    orders o ON od.order_id = o.order_id
WHERE
    MONTH(o.order_date) = 2
GROUP BY pizza_category
ORDER BY Total_Quantity_Sold DESC;
```

pizza_category	Total_Quantity_Sold
Classic	1178
Supreme	964
Veggie	944
Chicken	875

List the top 5 most ordered pizza types along with their quantities.

```sql
SELECT
    pzt.name AS pizza_name, SUM(od.quantity) AS pizza_quantity
FROM
    pizza_types AS pzt
        JOIN
    pizzas AS pz ON pzt.pizza_type_id = pz.pizza_type_id
        JOIN
    order_details AS od ON od.pizza_id = pz.pizza_id
GROUP BY pizza_name
ORDER BY pizza_quantity DESC
LIMIT 5
;
```

Result Grid | Filter Rows:

pizza_name	pizza_quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

List the least 5 ordered pizza types along with their quantities.

```sql
SELECT
    pzt.name AS pizza_name, SUM(od.quantity) AS pizza_quantity
FROM
    pizza_types AS pzt
        JOIN
    pizzas AS pz ON pzt.pizza_type_id = pz.pizza_type_id
        JOIN
    order_details AS od ON od.pizza_id = pz.pizza_id
GROUP BY pizza_name
ORDER BY pizza_quantity ASC
LIMIT 5
;
```

pizza_name	pizza_quantity
The Brie Carre Pizza	490
The Mediterranean Pizza	934
The Calabrese Pizza	937
The Spinach Supreme Pizza	950
The Soppressata Pizza	961

Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
SELECT
    distinct pzt.category, SUM(od.quantity) AS total_ordered
FROM
    pizza_types AS pzt
        JOIN
    pizzas AS pz ON pzt.pizza_type_id = pz.pizza_type_id
        JOIN
    order_details AS od ON od.pizza_id = pz.pizza_id
GROUP BY pzt.category
ORDER BY total_ordered DESC;
```

Result Grid | Filter Row

category	total_ordered
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Join the necessary tables to find the percentage sales by each pizza category.

```sql
with cte as (
SELECT
    pzt.category as 'pizza_category',
    ROUND(SUM(od.quantity * pz.price),2) as 'total_price'
FROM
    pizza_types AS pzt
        JOIN
    pizzas AS pz ON pzt.pizza_type_id = pz.pizza_type_id
        JOIN
    order_details AS od ON od.pizza_id = pz.pizza_id
GROUP BY pzt.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 cte ) AS DECIMAL(10,2)) AS PCT
FROM cte
GROUP BY pizza_category;
```

	pizza_category	Total_Revenue	PCT
▶	Classic	220053.10	26.91
	Veggie	193690.45	23.68
	Supreme	208197.00	25.46
	Chicken	195919.50	23.96

Determine the daily trend of orders

```sql
SELECT
    DAYNAME(o.order_date) AS Order_Day,
    COUNT(DISTINCT od.order_id) AS Total_Orders
FROM
    orders o
        JOIN
    order_details od ON o.order_id = od.order_id
GROUP BY DAYNAME(o.order_date)
ORDER BY DAYNAME(o.order_date);
```

Result Grid | Filter Row

Order_Day	Total_Orders
Friday	3538
Monday	2794
Saturday	3158
Sunday	2624
Thursday	3239
Tuesday	2973
Wednesday	3024

Determine the hourly trend of orders

```sql
SELECT
    HOUR(o.order_time) AS hours,
    COUNT(DISTINCT o.order_id) AS total_orders,
    SUM(od.quantity) AS total_quantity
FROM
    orders o
        JOIN
    order_details od ON o.order_id = od.order_id
GROUP BY HOUR(o.order_time)
ORDER BY hours ASC;
```

Result Grid | Filter Rows:

hours	total_orders	total_quantity
9	1	4
10	8	18
11	1231	2728
12	2520	6776
13	2455	6413
14	1472	3613
15	1468	3216
16	1920	4239
17	2336	5211
18	2399	5417
19	2009	4406
20	1642	3534
21	1198	2545
22	663	1386
23	28	68

Determine the distribution of orders by hour of the day.

```sql
SELECT
    HOUR(order_time) AS Order_Hour,
    COUNT(order_id) AS Order_id
FROM
    orders
GROUP BY HOUR(order_time);
```

Result Grid | Filter R

Order_Hour	Order_id
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

Join relevant tables to find the category-wise distribution of pizzas.

```sql
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

	category	COUNT(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

Calculate the average number of pizzas ordered per day.

```sql
with cte as (
select o.order_date as O_Date, sum(od.quantity) as qty
from orders o
join order_details od
on o.order_id = od.order_id
group by O_Date)

select round(avg(qty),0) as avg_pizza_ordered_per_day from cte;
```

Result Grid | Filter Rows:

avg_pizza_ordered_per_day
138

Calculate the average order per value.

```
with cte as(
select
ROUND(SUM(od.quantity * pz.price),2) as 'total_price',
count(distinct od.order_id) as 'total_orders'
from  order_details od
join
pizzas pz
on od.pizza_id = pz.pizza_id)


select (total_price/total_orders) as Avg_Order_Value from cte;
```

Result Grid | Filter Rows

	Avg_Order_Value
▶	38.30726229508197

Calculate the average pizzas per order

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

Result Grid | Filter Rows:

	Avg_Pizzas_Per_Order
▶	2.32

Determine the top 3 most ordered pizza types based on revenue.

```sql
SELECT
    pzt.name AS 'P_name',
    ROUND(SUM(pz.price * od.quantity), 2) AS 'Revenue'
FROM
    pizza_types AS pzt
        JOIN
    pizzas AS pz ON pz.pizza_type_id = pzt.pizza_type_id
        JOIN
    order_details AS od ON od.pizza_id = pz.pizza_id
GROUP BY P_name
ORDER BY Revenue DESC
LIMIT 3;
```

Result Grid Filter Rows:

P_name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Calculate the percentage contribution of each pizza type to total revenue.

```sql
WITH
  revenue AS (
    SELECT
      pzt.name AS 'P_name',
      ROUND(SUM(pz.price * od.quantity), 2) AS 'rev'
    FROM
      pizza_types AS pzt
      JOIN pizzas AS pz ON pz.pizza_type_id = pzt.pizza_type_id
      JOIN order_details AS od ON od.pizza_id = pz.pizza_id
    GROUP BY
      P_name
  ),
  total_sales AS (
    SELECT
      ROUND(SUM(od.quantity * pz.price), 2) AS ts
    FROM
      order_details AS od
      JOIN pizzas AS pz ON pz.pizza_id = od.pizza_id
  )
SELECT
  P_name,
  ROUND((rev / (SELECT ts FROM total_sales)) * 100, 2) AS percentage
FROM
  revenue;
```

Result Grid | Filter Rows:

P_name	percentage
The Hawaiian Pizza	3.95
The Classic Deluxe Pizza	4.67
The Five Cheese Pizza	3.19
The Italian Supreme Pizza	4.09
The Mexicana Pizza	3.27
The Thai Chicken Pizza	5.31
The Prosciutto and Arugula Pizza	2.96
The Barbecue Chicken Pizza	5.23
The Greek Pizza	3.48
The Spinach Supreme Pizza	1.87
The Green Garden Pizza	1.71
The Italian Capocollo Pizza	3.07
The Spicy Italian Pizza	4.26
The Spinach Pesto Pizza	1.91
The Vegetables + Vegetables Pi...	2.98
The Southwest Chicken Pizza	4.24
The California Chicken Pizza	5.06
The Pepperoni Pizza	3.69
The Chicken Pesto Pizza	2.04
The Big Meat Pizza	2.81
The Soppressata Pizza	2.01
The Four Cheese Pizza	3.95
The Napolitana Pizza	2.95
The Calabrese Pizza	1.95
The Italian Vegetables Pizza	1.96
The Mediterranean Pizza	1.88
The Pepper Salami Pizza	3.12
The Spinach and Feta Pizza	2.85
The Sicilian Pizza	3.78
The Chicken Alfredo Pizza	2.07
The Pepperoni, Mushroom, and ...	2.3
The Brie Carre Pizza	1.42

Analyze the revenue generated per day and the percentage contribution to total revenue.

```sql
with rvpd as (
select
date(o.order_date) as Ord_date,
round(sum(od.quantity*pz.price),2) as revenue_per_day
from
orders as o
join
order_details as od
on o.order_id = od.order_id
join
pizzas as pz
on od.pizza_id = pz.pizza_id
group by Ord_date),

total_sales AS (
    SELECT
      ROUND(SUM(od.quantity * pz.price), 2) AS ts
    FROM
      order_details AS od
      JOIN pizzas AS pz ON pz.pizza_id = od.pizza_id
  )

  SELECT
  Ord_date, revenue_per_day,
  ROUND((revenue_per_day / (SELECT ts FROM total_sales)) * 100, 2) AS percentage
FROM
  rvpd;
```

Result Grid | Filter Rows:

Ord_date	revenue_per_day	percentage
2015-01-01	2713.85	0.33
2015-01-02	2731.9	0.33
2015-01-03	2662.4	0.33
2015-01-04	1755.45	0.21
2015-01-05	2065.95	0.25
2015-01-06	2428.95	0.3
2015-01-07	2202.2	0.27
2015-01-08	2838.35	0.35
2015-01-09	2127.35	0.26
2015-01-10	2463.95	0.3
2015-01-11	1872.3	0.23
2015-01-12	1919.05	0.23
2015-01-13	2049.6	0.25
2015-01-14	2527.4	0.31
2015-01-15	1984.8	0.24
2015-01-16	2594.15	0.32
2015-01-17	2064.1	0.25
2015-01-18	1976.85	0.24
2015-01-19	2387.15	0.29
2015-01-20	2397.9	0.29

Result 1

Analyze the cumulative revenue generated over time.

```sql
with sales as (
select
o.order_date as Ord_date,
round(sum(od.quantity*pz.price),2) as revenue_per_day
from
orders as o
join
order_details as od
on o.order_id = od.order_id
join
pizzas as pz
on od.pizza_id = pz.pizza_id
group by Ord_date)

select Ord_date,
sum(revenue_per_day) over(order by Ord_date) as cum_revenue from sales;
```

Result Grid | Filter Rows:

Ord_date	cum_revenue
2015-01-01	2713.85
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.399999999998
2015-01-10	23990.35
2015-01-11	25862.649999999998
2015-01-12	27781.699999999997
2015-01-13	29831.299999999996
2015-01-14	32358.699999999997
2015-01-15	34343.5
2015-01-16	36937.65
2015-01-17	39001.75
2015-01-18	40978.6
2015-01-19	43365.75
2015-01-20	45763.65
2015-01-21	47804.200000000004
2015-01-22	50300.9
2015-01-23	52724.6
2015-01-24	55013.85
2015-01-25	56631.4
2015-01-26	58515.8

Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
with ranking as (
with cte as (
select  pzt.category as 'cat', pzt.name 'pname',
round(sum(od.quantity * pz.price),2) as revenue
from
pizza_types as pzt
join
pizzas as pz
on pzt.pizza_type_id = pz.pizza_type_id
join order_details as od
on od.pizza_id = pz.pizza_id
group by cat,pname)

select cat,pname,revenue,
rank() over(partition by cat order by revenue desc) as rnk
from cte)

select * from ranking
where rnk<=3;
```

Result Grid | Filter Rows: | Export:

cat	pname	revenue	rnk
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.7	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3

Let’s Start With
Creating Dashboard in

KPIs Creation

Total Revenue	Total Orders	Total Pizza Sold
Sum of total_price	**Sum of total_ordes**	**Sum of quantity**
817860.05	21350	49574

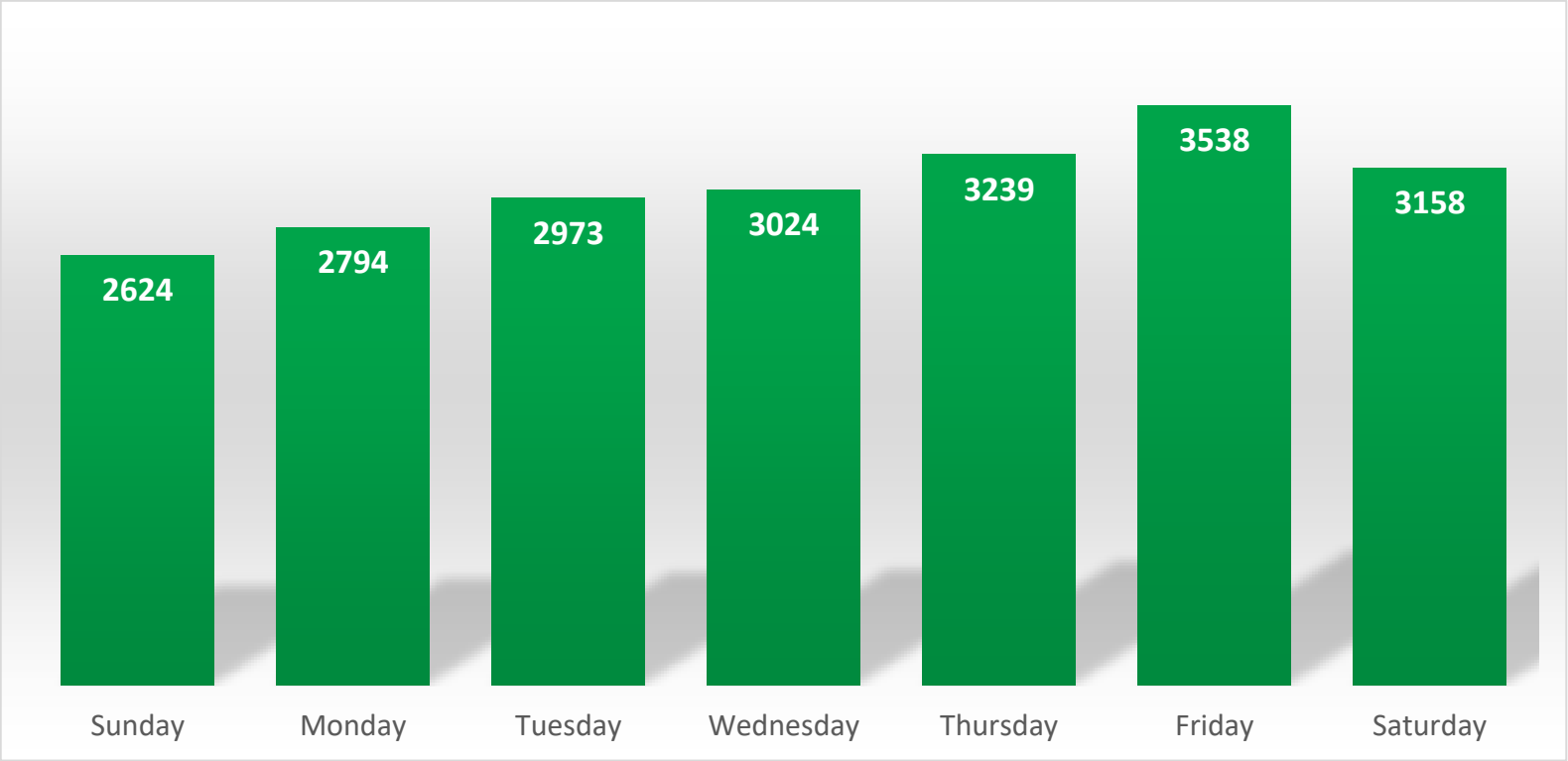
Avg.Order Value
$38.31

= [Total Revenue]/[Total Orders]

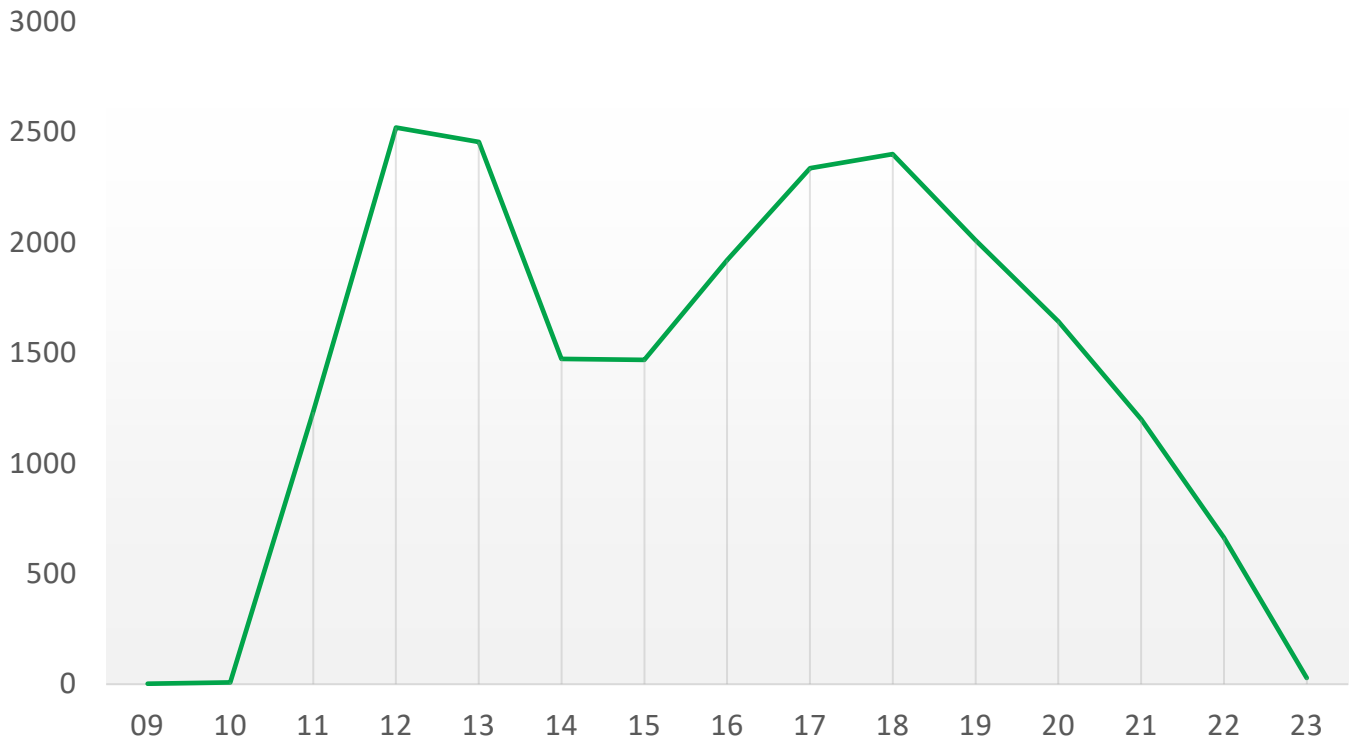
Avg. Pizza Per Order
2.32

= [Total Pizza Sold]/[Total Orders]

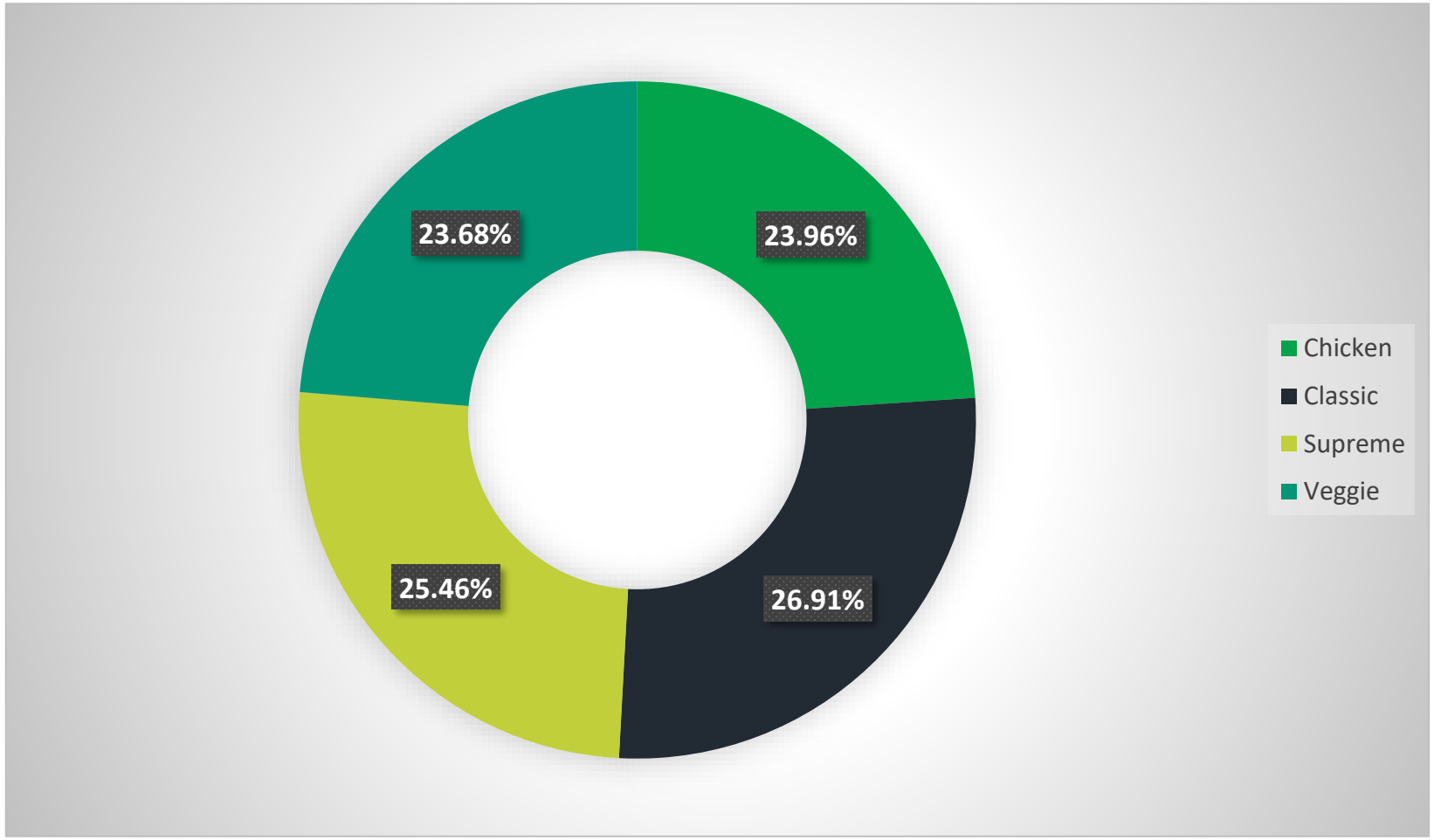
Daily Trends for Total Orders



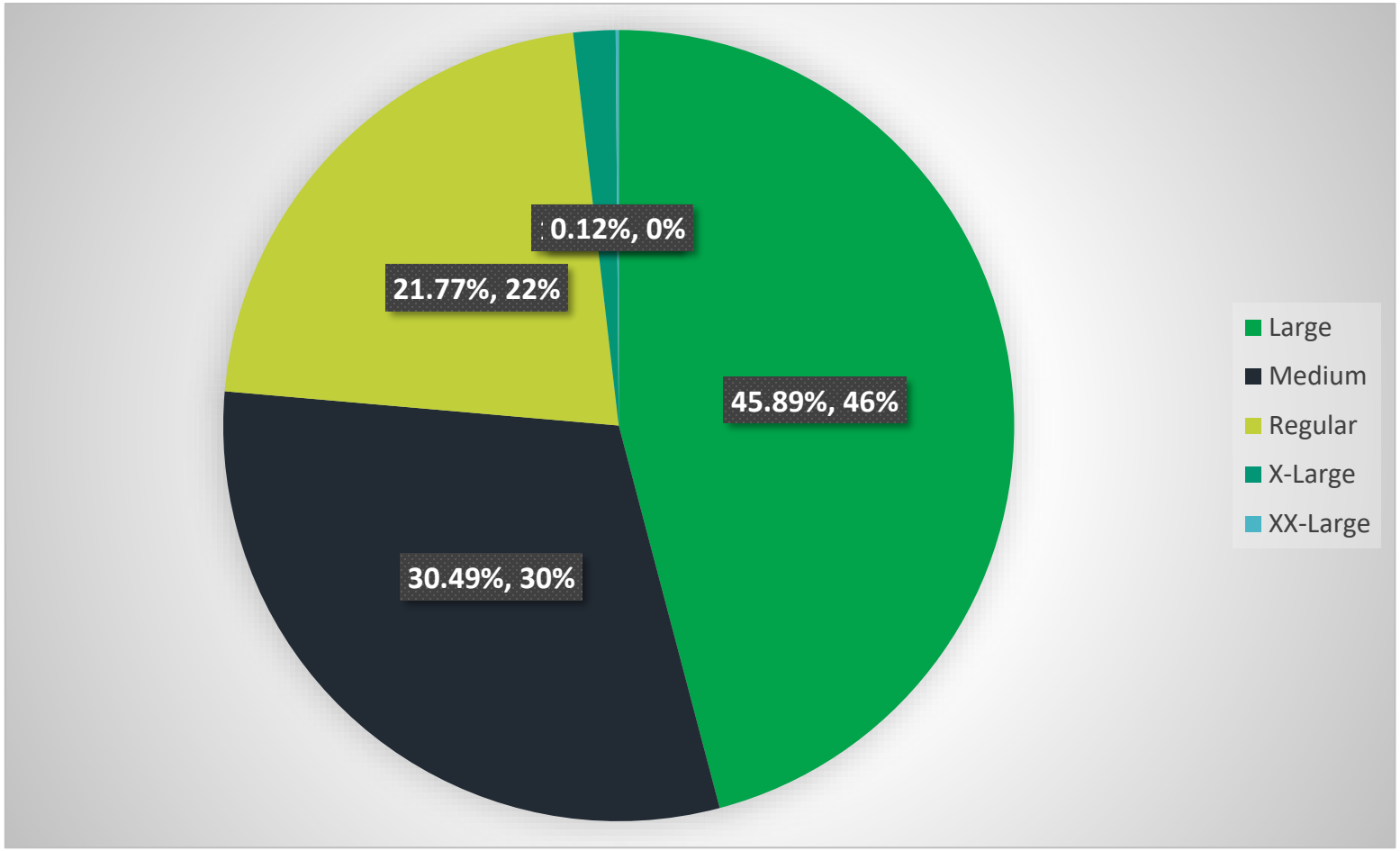
Hourly Trends for Total Orders



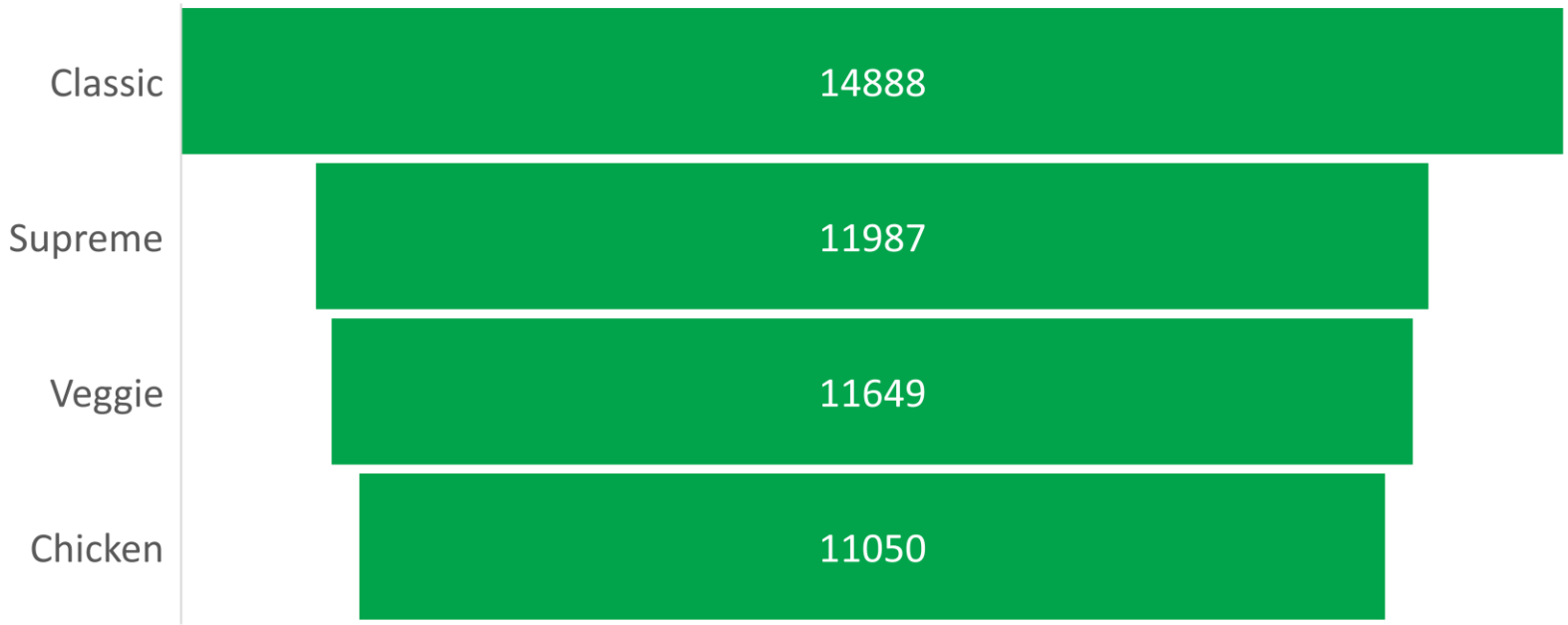
%age of Sales by Pizza Category



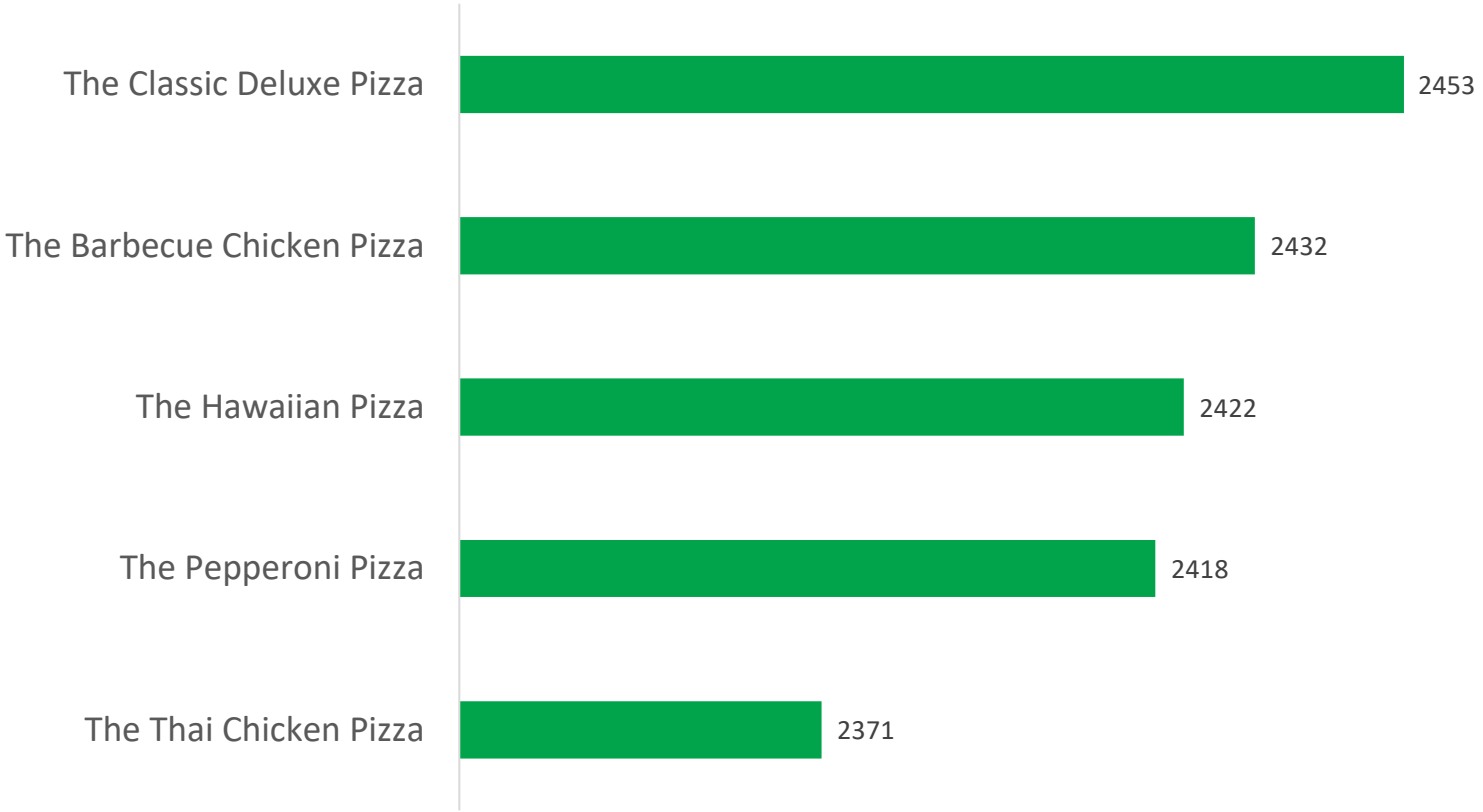
%age of Sales by Pizza Size



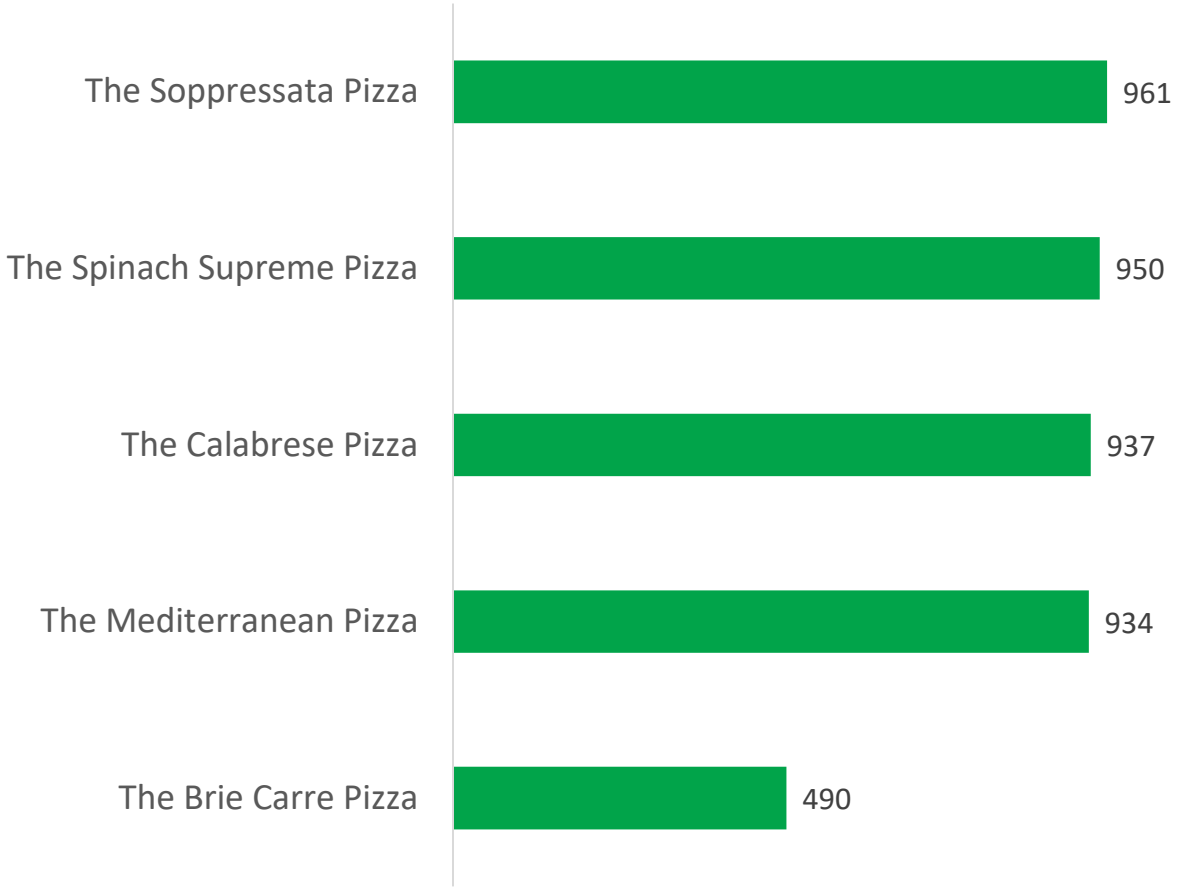
Total Pizza Sold by Pizza Category



Top 5 Selling



Worst 5 Selling



Sales Dashboard

Pizza Sales Dashboard

Total Revenue	Avg. Order Value	Total Pizza Sold	Total Orders	Avg. Pizza Per Order
$8,17,860	$38.31	49,574	21,350	2.32

Busiest Days & Times

DAYS
Orders are **highest** on **Friday/Saturday** evenings.

TIMES
The **maximum** orders are between **12 to 1pm & 4 to 8pm**

Sales by Category & Size

CATEGORY
Classic category contributes to **maximum** sales & total orders.

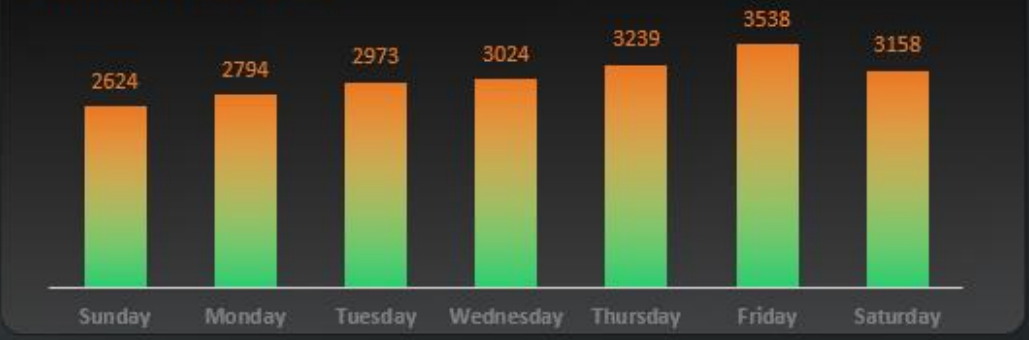
SIZE
Large size pizza contributes to **maximum** sales.

Best & Worst Sellers

BEST
Classic Deluxe & Chicken pizzas are the sellers and revenue generators

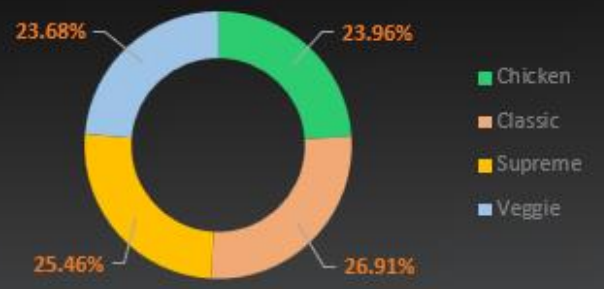
WORST
The Brie Carre is at the bottom in both orders and revenue

Daily Trends for Total Orders

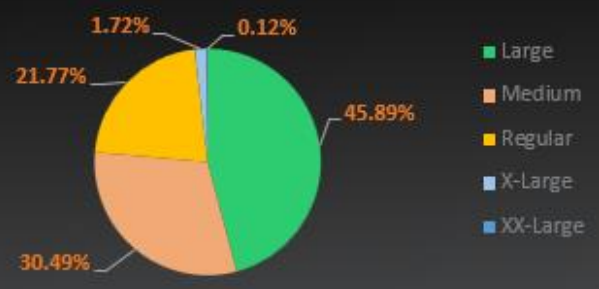


Hourly Trend for Total Orders

% of Sales by Pizza Category

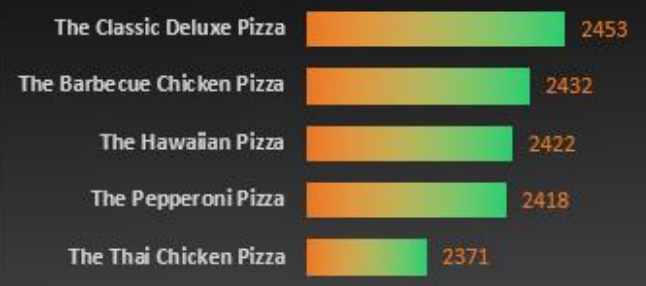


% of Sales by Pizza Size

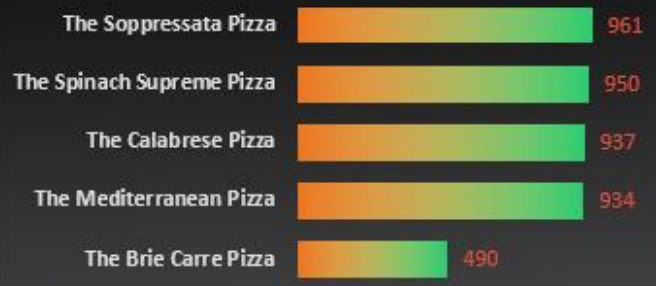


Total Pizza Sold by Pizza Category

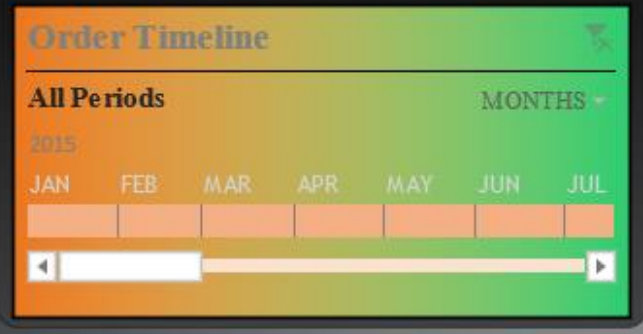
Top 5 Best Sellers by Total Pizzas Sold



Bottom 5 Worst Sellers by Total Pizzas Sold



Filter



Important Links

- .csv files used
- Image Resources Used
- Questions & SQL Queries
- Dashboard .xlsx file
- Github Repository

Thank You for
Staying till end!!