

# Pizza Sales Analysis

## Importing necessary packages

```
In [1]: import numpy as np
import pandas as pd

import plotly.express as px
from plotly.offline import iplot

import warnings
warnings.filterwarnings('ignore')

import datetime
import calendar
```

## Importing the dataset

```
In [2]: data = pd.read_csv("pizza_sales.csv")
data.head()
```

Out[2]:

	pizza_id	order_id	pizza_name_id	quantity	order_date	order_time	unit_price	total_price
0	1	1	hawaiian_m	1	1/1/2015	11:38:36	13.25	13.25
1	2	2	classic_dlx_m	1	1/1/2015	11:57:40	16.00	16.00
2	3	2	five_cheese_l	1	1/1/2015	11:57:40	18.50	18.50
3	4	2	ital_supr_l	1	1/1/2015	11:57:40	20.75	20.75
4	5	2	mexicana_m	1	1/1/2015	11:57:40	16.00	16.00



```
In [3]: # Shape:
data.shape
```

Out[3]: (48620, 12)

There are **48,620** rows and **12** columns present in the dataset

```
In [4]: # Columns:
data.columns
```

```
Out[4]: Index(['pizza_id', 'order_id', 'pizza_name_id', 'quantity', 'order_date',
              'order_time', 'unit_price', 'total_price', 'pizza_size',
              'pizza_category', 'pizza_ingredients', 'pizza_name'],
              dtype='object')
```

```
In [5]: # Statistics of the data:
data.describe()
```

```
Out[5]:
```

	pizza_id	order_id	quantity	unit_price	total_price
<b>count</b>	48620.000000	48620.000000	48620.000000	48620.000000	48620.000000
<b>mean</b>	24310.500000	10701.479761	1.019622	16.494132	16.821474
<b>std</b>	14035.529381	6180.119770	0.143077	3.621789	4.437398
<b>min</b>	1.000000	1.000000	1.000000	9.750000	9.750000
<b>25%</b>	12155.750000	5337.000000	1.000000	12.750000	12.750000
<b>50%</b>	24310.500000	10682.500000	1.000000	16.500000	16.500000
<b>75%</b>	36465.250000	16100.000000	1.000000	20.250000	20.500000
<b>max</b>	48620.000000	21350.000000	4.000000	35.950000	83.000000

## Checking for null values:

```
In [6]: data.isna().sum()
```

```
Out[6]: pizza_id          0
order_id          0
pizza_name_id     0
quantity          0
order_date        0
order_time        0
unit_price        0
total_price       0
pizza_size        0
pizza_category    0
pizza_ingredients 0
pizza_name        0
dtype: int64
```

There is no null values

## Datatype of each column

```
In [7]: data.dtypes
```

```
Out[7]: pizza_id          int64
order_id          int64
pizza_name_id     object
quantity          int64
order_date        object
order_time        object
unit_price        float64
total_price       float64
pizza_size        object
pizza_category    object
pizza_ingredients object
pizza_name        object
dtype: object
```

## Column Description

### 1) pizza\_id

```
In [8]: data['pizza_id'].nunique()
```

```
Out[8]: 48620
```

- The number of unique items in **pizza\_id** column and the number of rows in the **dataset** are same.
- **pizza\_id** is like a serial number.

### 2) order\_id

```
In [9]: data['order_id'].nunique()
```

```
Out[9]: 21350
```

- The number of unique items in **order\_id** is less than total number of rows in the dataset.
- At a single order the customer can order more than one pizza (quantity) and more than one type of pizza (pizza\_name\_id).

### 3) pizza\_name\_id

```
In [10]: data['pizza_name_id'].nunique()
```

```
Out[10]: 91
```

- It gives the id of the pizza name

#### 4) quantity

```
In [11]: data['quantity'].nunique()
```

```
Out[11]: 4
```

```
In [12]: data['quantity'].unique()
```

```
Out[12]: array([1, 2, 3, 4], dtype=int64)
```

- It gives the quantity of a type of pizza(pizza\_name\_id) ordered in a particular order\_id.
- Maximum of **4** pizzas were ordered in a particular type of pizza(pizza\_name\_id).
- And minimum of **1** pizza is ordered in a particular type of pizza(pizza\_name\_id).

#### 5) order\_date

```
In [13]: data['order_date'].dtype
```

```
Out[13]: dtype('O')
```

```
In [14]: # Changing the datatype of order_date column into datetime datatype:  
data['order_date'] = pd.to_datetime(data['order_date'])
```

```
In [15]: data['order_date'].dtype
```

```
Out[15]: dtype('<M8[ns]')
```

- It gives the order date

#### 6) order\_time

```
In [16]: data['order_time'].dtype
```

```
Out[16]: dtype('O')
```

- It gives the order time

#### 7) unit\_price

- It gives the price of a single pizza in quantity column

## 8) total\_price

- It gives the total prize of all pizzas in quantity column

## 9) pizza\_size

```
In [17]: data['pizza_size'].unique()
```

```
Out[17]: array(['M', 'L', 'S', 'XL', 'XXL'], dtype=object)
```

- It gives the size of the pizza
1. M --> Medium
  2. L --> Large
  3. S --> Small
  4. XL -> Extra Large
  5. XXL > Extra Extra Large

## 10) pizza\_category

```
In [18]: data['pizza_category'].unique()
```

```
Out[18]: array(['Classic', 'Veggie', 'Supreme', 'Chicken'], dtype=object)
```

- It gives the category of the pizza

## 11) pizza\_ingredients

- It gives the ingredients used in the pizza

## 12) pizza\_name

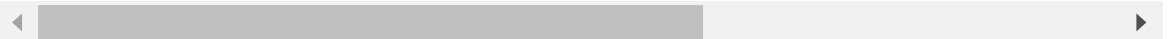
- It gives the name of the pizza

## Analysing the Data

In [19]: `data.head()`

Out[19]:

	pizza_id	order_id	pizza_name_id	quantity	order_date	order_time	unit_price	total_price
0	1	1	hawaiian_m	1	2015-01-01	11:38:36	13.25	13.25
1	2	2	classic_dlx_m	1	2015-01-01	11:57:40	16.00	16.00
2	3	2	five_cheese_l	1	2015-01-01	11:57:40	18.50	18.50
3	4	2	ital_supr_l	1	2015-01-01	11:57:40	20.75	20.75
4	5	2	mexicana_m	1	2015-01-01	11:57:40	16.00	16.00



## KPI's

### 1.) Total Revenue

```
In [20]: tot_rev = data['total_price'].sum()
print("Total Revenue: ${}".format(tot_rev))
```

Total Revenue: \$817860.05

### 2.) Total Orders

```
In [21]: tot_orders = data['order_id'].nunique()
print("Total Orders: {} orders".format(tot_orders))
```

Total Orders: 21350 orders

### 3.) Total Pizza Sold

```
In [22]: tot_pizza_sold = data['quantity'].sum()

print("Total Pizza Sold: {} pizzas".format(tot_pizza_sold))
```

Total Pizza Sold: 49574 pizzas

### 4.) Average Order Value

- Average amount spent per order

```
In [23]: avg_order_val = (tot_rev / tot_orders)

print("Average Order Value: ${}".format(round(avg_order_val, 2)))
```

Average Order Value: \$38.31

### 5.) Average Pizzas per Order

```
In [24]: avg_pizza_per_order = (tot_pizza_sold / tot_orders)

print("Average Pizzas per Order: {} pizzas/order".format(round(avg_pizza_pe
```

Average Pizzas per Order: 2.32 pizzas/order

## Charts

### 1.) Daily Trend for Total Orders

```
In [25]: days = []

for i in data['order_date']:
    day_of_week = i.dayofweek
    day = calendar.day_name[day_of_week]
    days.append(day)

data['Day'] = days
```

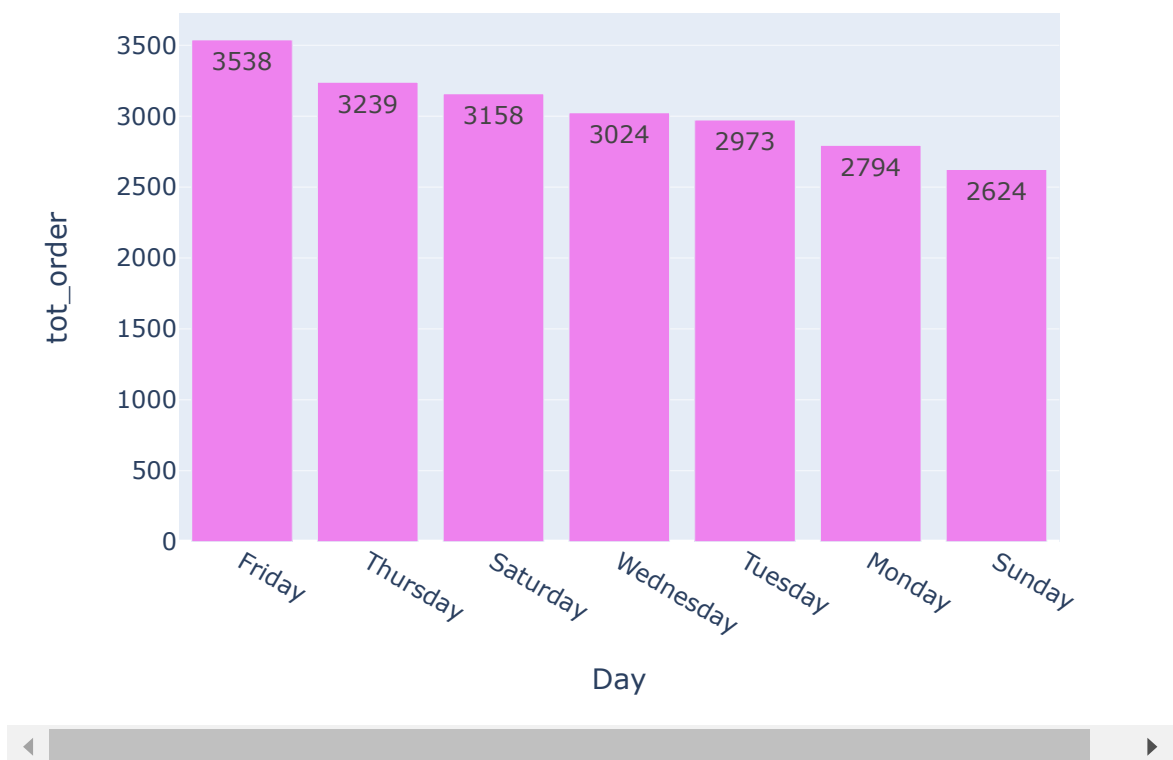
```
In [26]: df1 = data.groupby(by='Day').nunique().reset_index()[['Day', 'order_id']].so  
df1.rename(columns={'order_id': 'tot_order'}, inplace=True)  
df1
```

Out[26]:

	Day	tot_order
0	Friday	3538
4	Thursday	3239
2	Saturday	3158
6	Wednesday	3024
5	Tuesday	2973
1	Monday	2794
3	Sunday	2624

```
In [27]: fig1 = px.bar(df1, x='Day', y='tot_order', title='<b>Daily Trend for Total  
width=600, height=450, text_auto=True)  
  
fig1.update_traces(marker_color='violet', textposition='inside')  
iplot(fig1)
```

## Daily Trend for Total Orders





## 2.) Monthly Trend for Total Orders

```
In [28]: months = []

for i in data['order_date']:
    mnth = i.month
    month = calendar.month_name[mnth]
    months.append(month)

data['Month'] = months
```

```
In [29]: df2 = data.groupby(by='Month').nunique().reset_index()[['Month', 'order_id']]
df2.rename(columns={'order_id': 'tot_order'}, inplace=True)
df2
```

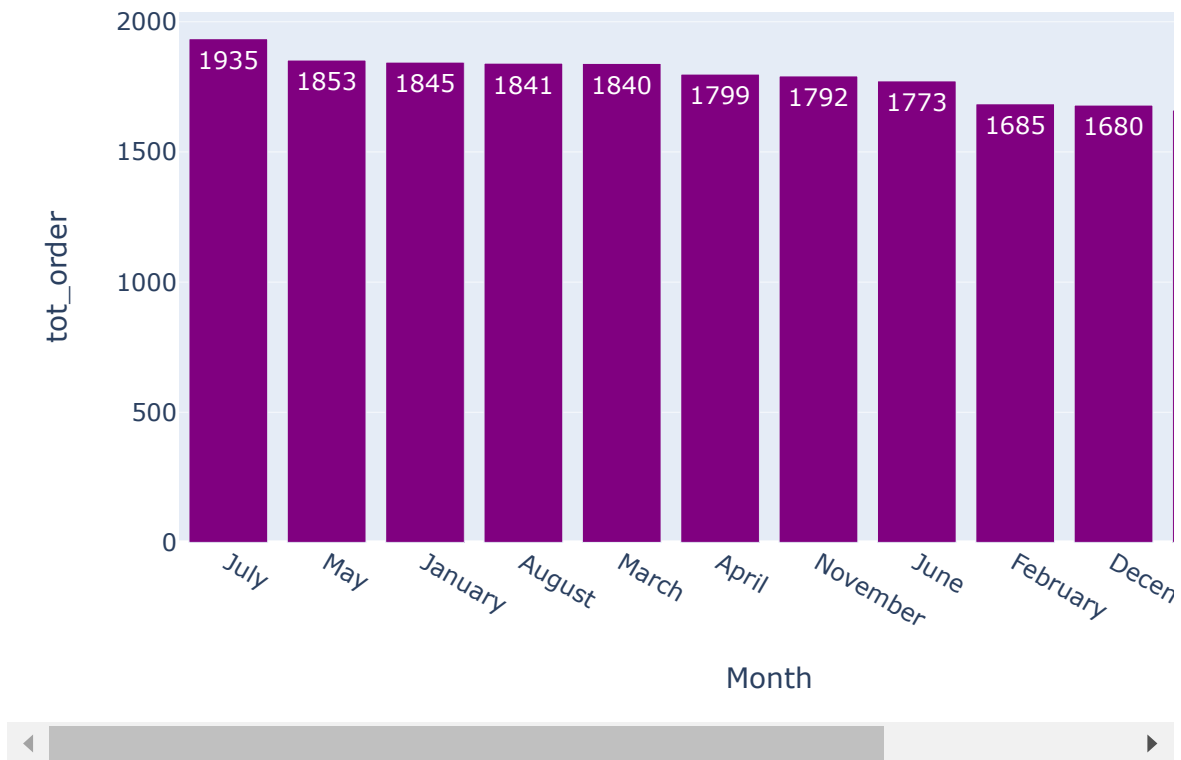
Out[29]:

	Month	tot_order
5	July	1935
8	May	1853
4	January	1845
1	August	1841
7	March	1840
0	April	1799
9	November	1792
6	June	1773
3	February	1685
2	December	1680
11	September	1661
10	October	1646

```
In [30]: fig2 = px.bar(df2, x='Month', y='tot_order', title='<b>Monthly Trend for To
          width=750, height=450, text_auto=True)

fig2.update_traces(marker_color='purple', textposition='inside')
iplot(fig2)
```

## Monthly Trend for Total Orders



## 3.) Percentage of Sales by Pizza Category

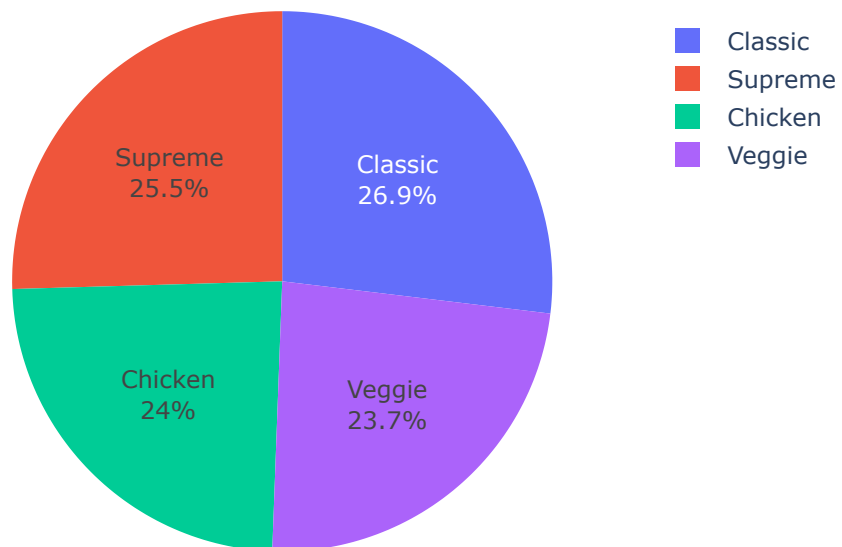
```
In [31]: df3 = data[['total_price', 'pizza_category']].groupby(by='pizza_category').s
df3['percentage_of_sales'] = round((df3['total_price']/df3['total_price'].s
df3
```

Out[31]:

	pizza_category	total_price	percentage_of_sales
0	Chicken	195919.50	23.96
1	Classic	220053.10	26.91
2	Supreme	208197.00	25.46
3	Veggie	193690.45	23.68

```
In [32]: fig3 = px.pie(df3, values='total_price', names='pizza_category',  
                    title="Percentage of Sales by Pizza Category",  
                    width=550, height=450, hole=0)  
  
fig3.update_traces(textposition='inside', textinfo='label+percent')  
iplot(fig3)
```

### Percentage of Sales by Pizza Category



### 4.) Percentage of Sales by Pizza Size

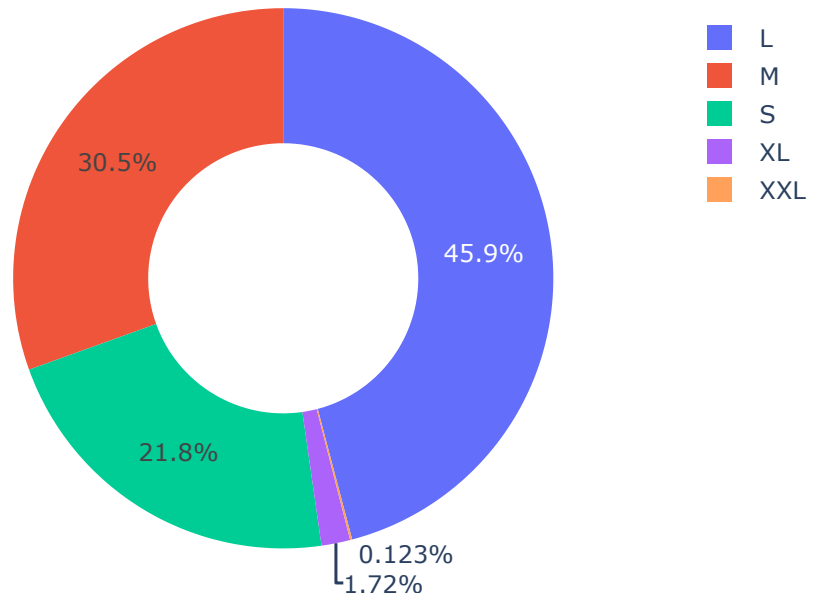
```
In [33]: df4 = data[['total_price', 'pizza_size']].groupby(by='pizza_size').sum().res  
df4['percentage_of_sales'] = round((df4['total_price']/df4['total_price'].s  
df4
```

Out[33]:

	pizza_size	total_price	percentage_of_sales
0	L	375318.70	45.89
1	M	249382.25	30.49
2	S	178076.50	21.77
3	XL	14076.00	1.72
4	XXL	1006.60	0.12

```
In [34]: fig4 = px.pie(df4, values='total_price', names='pizza_size',  
                      title='<b>Percentage of Sales by Pizza Size</b>',  
                      width=550, height=450, hole=0.5)  
  
iplot(fig4)
```

## Percentage of Sales by Pizza Size



## 5.) Total Pizzas Sold by Pizza Category

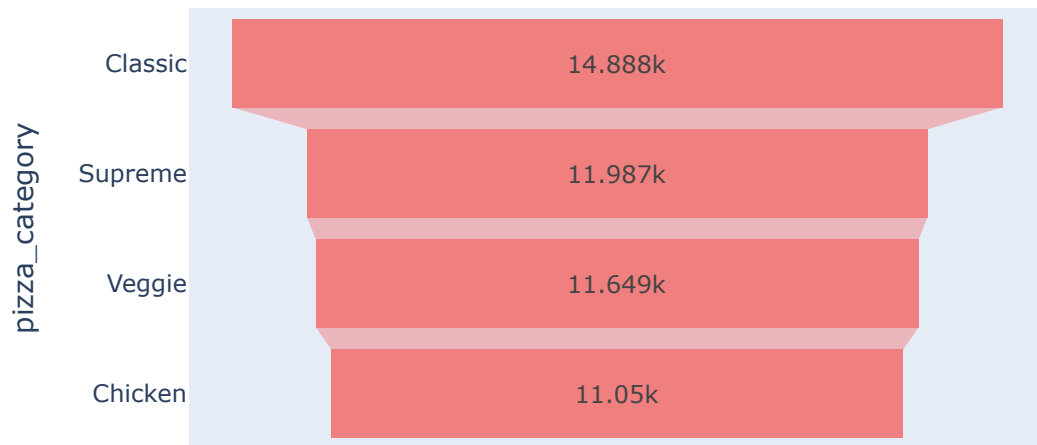
```
In [35]: df5 = data[['quantity', 'pizza_category']].groupby(by='pizza_category').sum(  
df5 = df5.sort_values(by='quantity', ascending=False)  
df5
```

Out[35]:

	pizza_category	quantity
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
0	Chicken	11050

```
In [36]: fig5 = px.funnel(df5, x='quantity', y='pizza_category',  
                        title='<b>Total Pizzas Sold by Pizza Category</b>',  
                        width=600, height=400)  
  
fig5.update_traces(marker_color='lightcoral')  
iplot(fig5)
```

## Total Pizzas Sold by Pizza Category

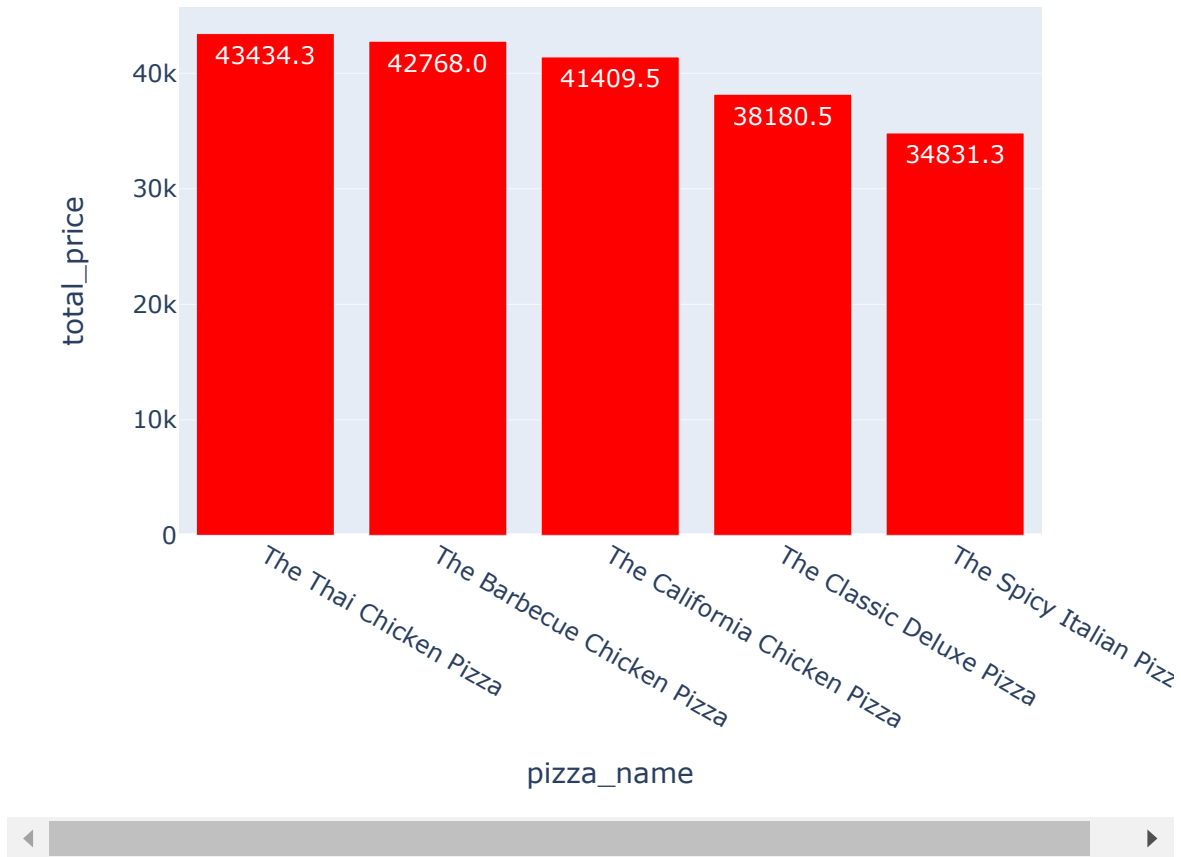


## 6.) Top 5 Best Sellers by Total Revenue, Total Quantity and Total Order

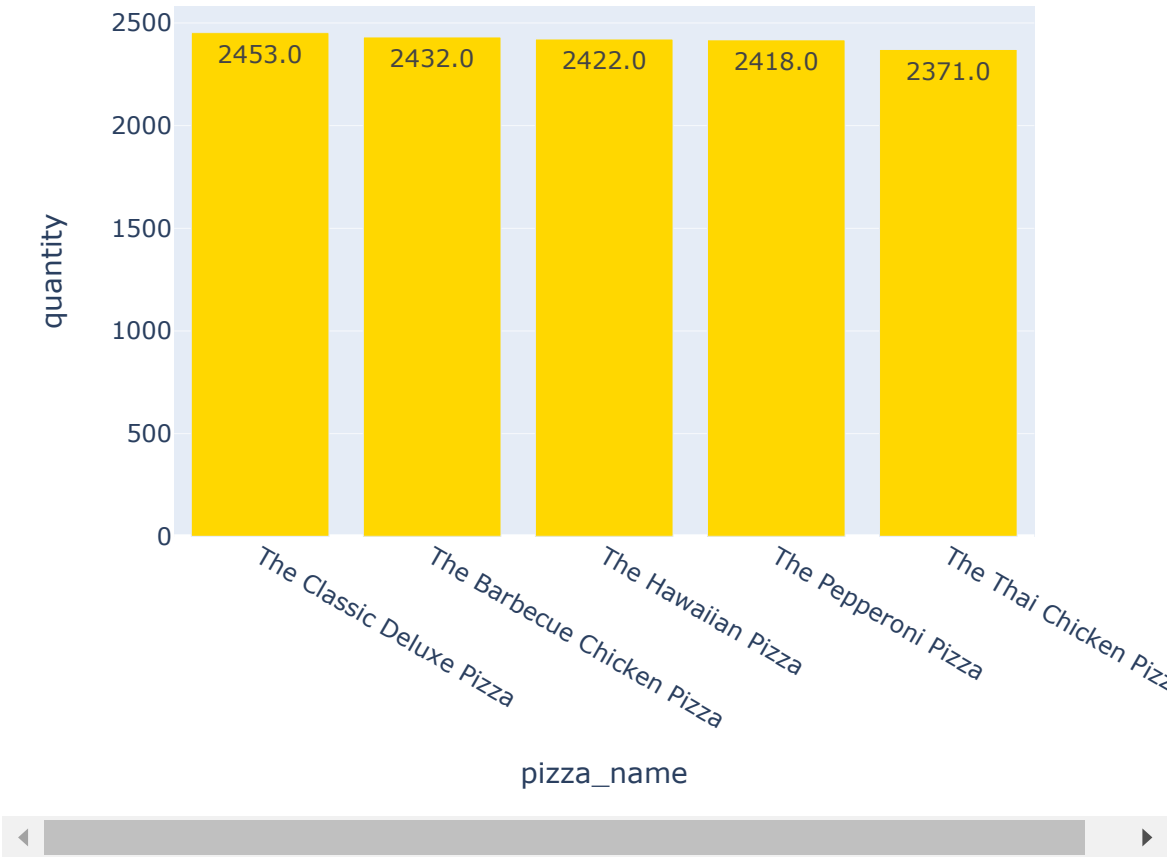
```
In [37]: def best_sellers_by(para,color):  
  
    if para=='tot_revenue':  
        col='total_price'  
    elif para=='tot_quantity':  
        col='quantity'  
    elif para=='tot_orders':  
        col='order_id'  
  
    if col=='order_id':  
        df6 = data[[col, 'pizza_name']].groupby(by='pizza_name').nunique().r  
    else:  
        df6 = data[[col, 'pizza_name']].groupby(by='pizza_name').sum().reset  
  
    df6 = df6.sort_values(by=col, ascending=False).head(5)  
  
    fig6 = px.bar(df6, x='pizza_name', y=col,  
                  title='Top 5 Best Sellers by '+para,  
                  width=600, height=500, text_auto='0.1f')  
  
    fig6.update_traces(textposition='inside', marker_color=color)  
    iplot(fig6)
```

```
In [38]: for i,j in zip(['tot_revenue', 'tot_quantity', 'tot_orders'], ['red', 'gold']  
           best_sellers_by(i,j))
```

### Top 5 Best Sellers by tot\_revenue

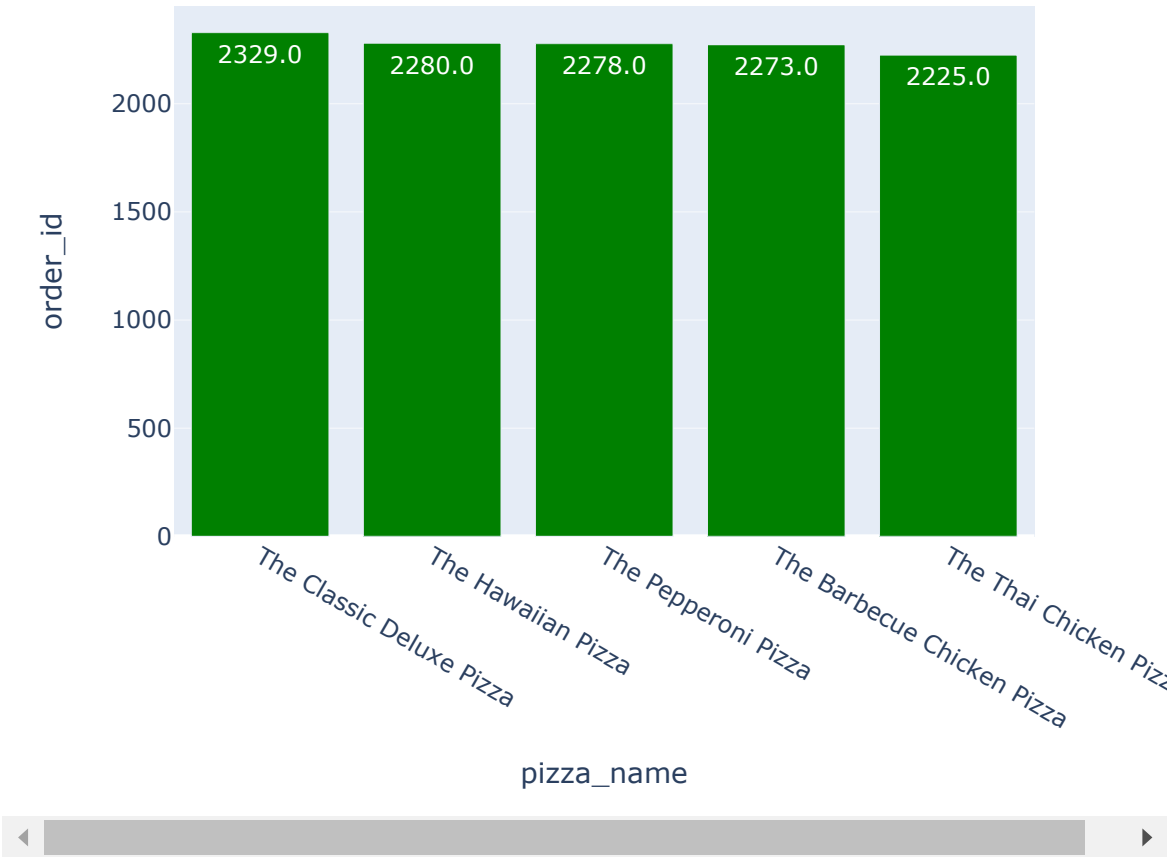


Top 5 Best Sellers by tot\_quantity





Top 5 Best Sellers by tot\_orders

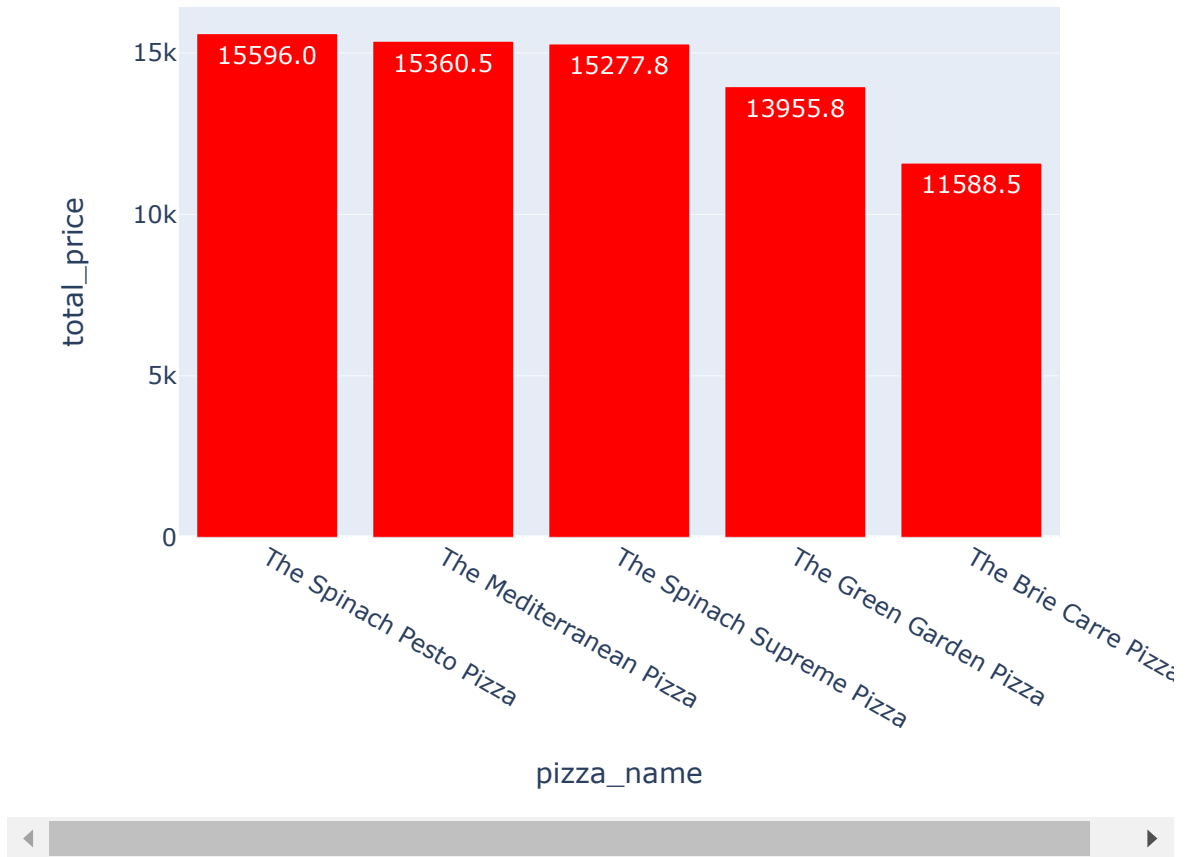


## 7.) Top 5 Worst Sellers by Total Revenue, Total Quantity and Total Order

```
In [39]: def worst_sellers_by(para,color):  
  
    if para=='tot_revenue':  
        col='total_price'  
    elif para=='tot_quantity':  
        col='quantity'  
    elif para=='tot_orders':  
        col='order_id'  
  
    if col=='order_id':  
        df7 = data[[col, 'pizza_name']].groupby(by='pizza_name').nunique().r  
    else:  
        df7 = data[[col, 'pizza_name']].groupby(by='pizza_name').sum().reset  
  
    df7 = df7.sort_values(by=col, ascending=False).tail(5)  
  
    fig7 = px.bar(df7, x='pizza_name', y=col,  
                  title='Top 5 Worst Sellers by '+para,  
                  width=600, height=500, text_auto='0.1f')  
  
    fig7.update_traces(textposition='inside', marker_color=color)  
    iplot(fig7)
```

```
In [40]: for i,j in zip(['tot_revenue', 'tot_quantity', 'tot_orders'], ['red', 'gold']  
          worst_sellers_by(i,j))
```

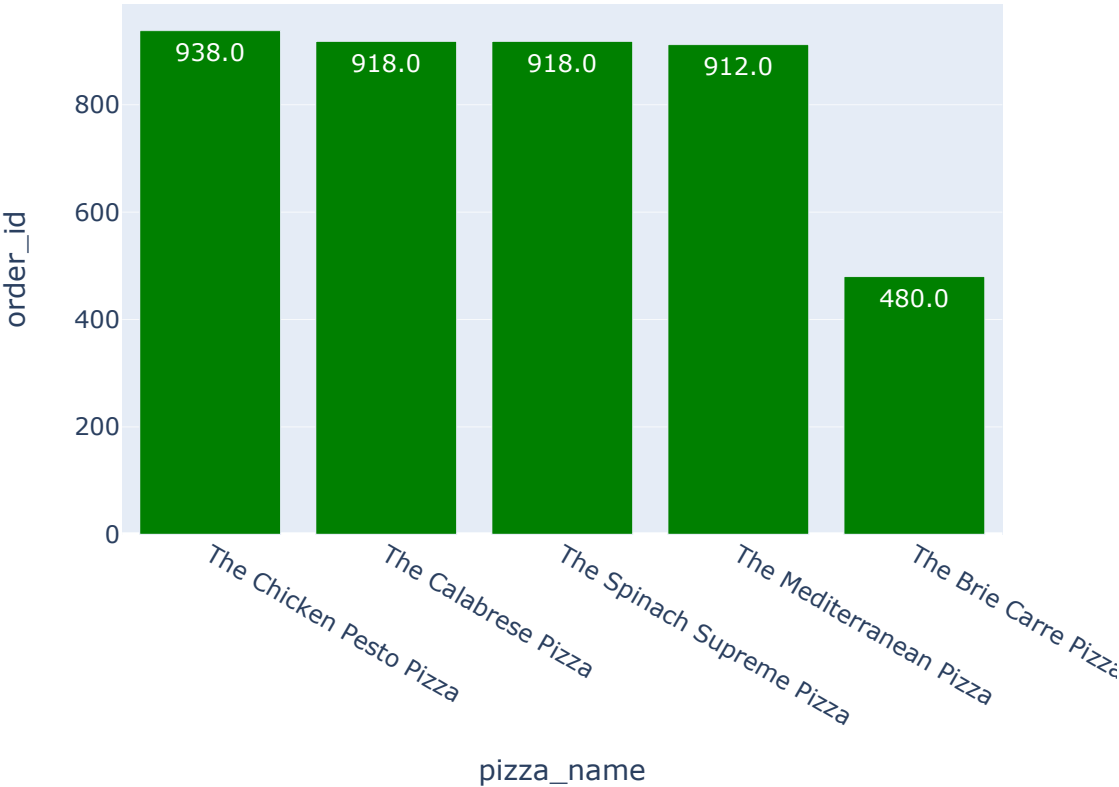
### Top 5 Worst Sellers by tot\_revenue



Top 5 Worst Sellers by tot\_quantity



Top 5 Worst Sellers by tot\_orders



In [ ]: