AMAZON SALES DATA ANALYSIS

Importing the csv file

```
from google.colab import files
uploaded=files.upload()
```

Choose Files Amazon_Sales_data.csv

- **Amazon_Sales_data.csv**(text/csv) - 12727 bytes, last modified: 1/5/2024 - 100% done

```
Saving Amazon_Sales_data.csv to Amazon_Sales_data.csv
```

Importing Python libraries

```
import pandas as pd
import matplotlib.pyplot as plt
```

Reading the file

```
data=pd.read_csv('Amazon_Sales_data.csv')
data
```

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Sh Da
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/20
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/20
2	Europe	Russia	Office Supplies	Offline	L	05-02-2014	341417157	05-0 20
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	07-0 20
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	02-01-2013	115456712	02-0 20
...	...	...	...	...	...	...	...	
95	Sub-Saharan Africa	Mali	Clothes	Online	M	7/26/2011	512878119	09-0 20
96	Asia	Malaysia	Fruits	Offline	L	11-11-2011	810711038	12/28/20
	Sub-					06-01-		

Converting columns of Order Date and Ship Date to similar DateTime format

```
data['Order Date']=pd.to_datetime(data['Order Date'])
data['Ship Date']=pd.to_datetime(data['Ship Date'])

data['month']=data['Order Date'].dt.month
data['year']=data['Order Date'].dt.year
data['yearly_month']=data['Order Date'].dt.strftime('%Y-%m')
```

Calculating Sales for every month

```
month_sales=data.groupby('month')['Total Revenue'].sum()
month_sales
```

```
month
1     10482467.12
```

```
2     24740517.77
3      2274823.87
4     16187186.33
5     13215739.99
6      5230325.77
7     15669518.50
8      1128164.91
9      5314762.56
10    15287576.61
11    20568222.76
12     7249462.12
Name: Total Revenue, dtype: float64
```

Calculating sales for every year

```
year_sales=data.groupby('year')['Total Revenue'].sum()
year_sales
```

```
year
2010    19186024.92
2011    11129166.07
2012    31898644.52
2013    20330448.66
2014    16630214.43
2015    12427982.86
2016    12372867.22
2017    13373419.63
Name: Total Revenue, dtype: float64
```

Calculating Sales for every year month wise

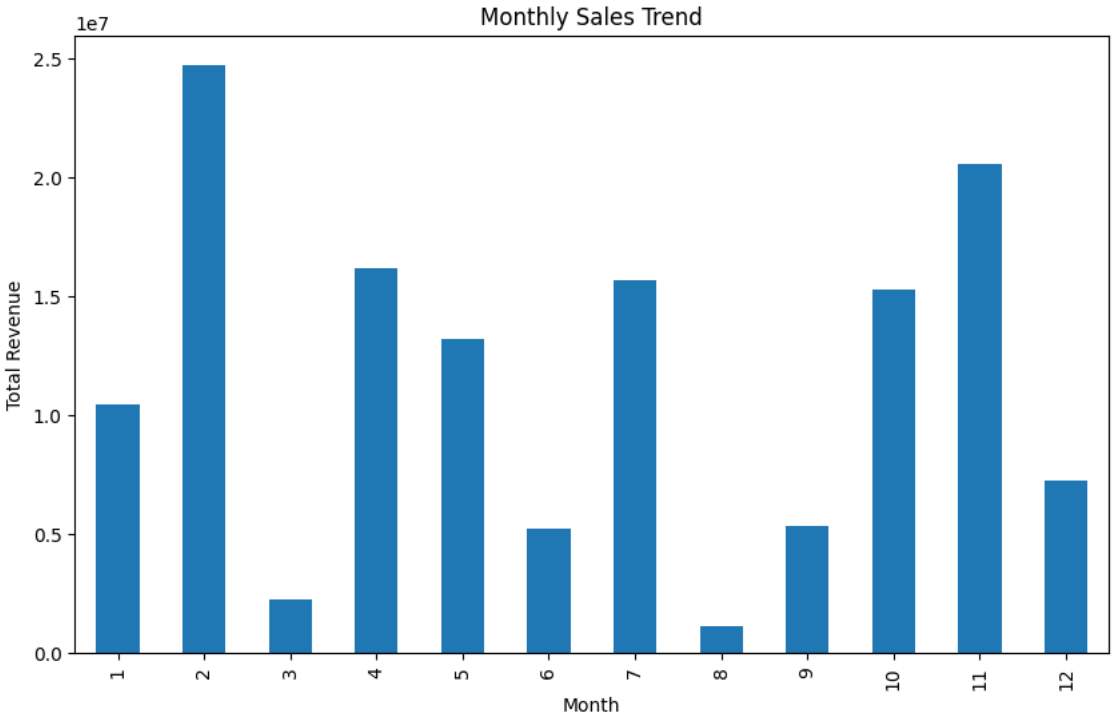
```
yearly_month_sales=data.groupby('yearly_month')['Total Revenue'].sum()
yearly_month_sales
```

```
yearly_month
2010-02    3410661.12
2010-05    2587973.26
2010-06    1082418.40
2010-10    6064933.75
2010-11    3458252.00
2010-12    2581786.39
2011-01    1042225.35
2011-02     387002.20
2011-04    2798046.49
2011-05     272410.45
2011-06      19103.44
2011-07      97040.64
2011-09     574951.92
2011-11    5938385.58
2012-01    1012884.00
2012-02    6707849.42
2012-03     994765.42
2012-04    4556012.38
2012-05    3782781.82
2012-06    2132075.27
2012-07    4445093.92
2012-08     576782.80
2012-09    4648152.72
2012-10    3042246.77
2013-02    3296425.02
2013-03     835759.10
2013-04    3262562.10
2013-06    1352867.40
2013-07    8545511.20
2013-08      89623.98
2013-09      71253.21
2013-10    2702770.40
2013-12     173676.25
2014-02    1819660.25
2014-04    4510578.10
2014-05    3060338.59
2014-06      75591.66
2014-07     688641.85
2014-08     455479.04
2014-09      20404.71
2014-10    1352370.65
2014-11    4647149.58
2015-01    5513227.50
2015-02    2003911.12
2015-04    1059987.26
2015-07    1292409.45
2015-08       6279.09
2015-10    1904138.04
2015-11     648030.40
```

```
2016-03     197883.40
2016-05     414371.10
2016-06     568269.60
2016-07     600821.44
2016-10     221117.00
2016-11    5876405.20
2016-12    4493999.48
```

Monthly Sales Vs Total Revenue

```
plt.figure(figsize=(10,6))
month_sales.plot(kind='bar')
plt.title('Monthly Sales Trend')
plt.xlabel('Month')
plt.ylabel('Total Revenue')
plt.show()
```



Total Sales Revenue of the given dataset

```
total_sales=data['Total Revenue'].sum()
total_sales
```

```
137348768.31
```

Average Sales Revenue of the given dataset

```
average_sales=data['Total Revenue'].mean()
average_sales
```

```
1373487.6831
```

Top 5 Demanded Items by the customers/Clients

```
demand_items=data.groupby('Item Type')['Total Revenue'].sum().nlargest(5)
demand_items
```

```
Item Type
Cosmetics          36601509.60
Office Supplies    30585380.07
Household          29889712.29
Baby Food          10350327.60
Clothes             7787292.80
Name: Total Revenue, dtype: float64
```

Correlation Matrix of the dataset

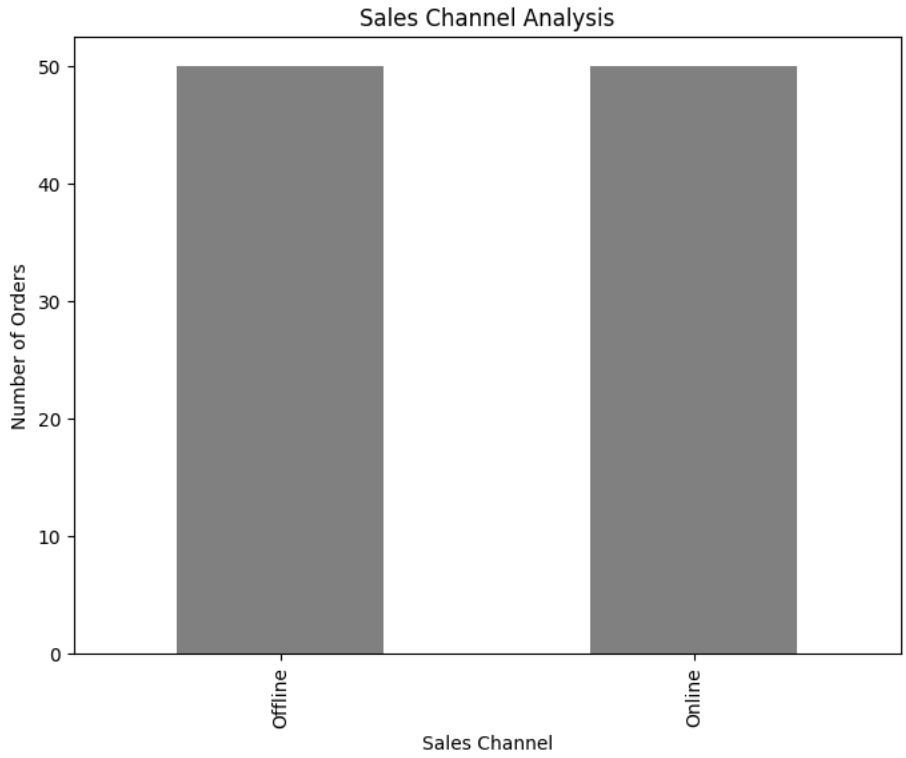
```
correlation=data.corr()
correlation
```

```
<ipython-input-29-d7a18ccdee06>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future ver
  correlation=data.corr()
```

	Order ID	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit	month	year
Order ID	1.000000	-0.222907	-0.190941	-0.213201	-0.314688	-0.328944	-0.234638	-0.111219	0.081752
Units Sold	-0.222907	1.000000	-0.070486	-0.092232	0.447784	0.374746	0.564550	-0.007995	0.012455
Unit Price	-0.190941	-0.070486	1.000000	0.987270	0.752360	0.787905	0.557365	-0.031917	-0.061791
Unit Cost	-0.213201	-0.092232	0.987270	1.000000	0.715623	0.774895	0.467214	-0.042016	-0.071567
Total Revenue	-0.314688	0.447784	0.752360	0.715623	1.000000	0.983928	0.897327	0.003835	-0.037128
Total Cost	-0.328944	0.374746	0.787905	0.774895	0.983928	1.000000	0.804091	-0.015617	-0.050899
Total Profit	-0.234638	0.564550	0.557365	0.467214	0.897327	0.804091	1.000000	0.051366	0.002196
month	-0.111219	-0.007995	-0.031917	-0.042016	0.003835	-0.015617	0.051366	1.000000	-0.106715
year	0.081752	0.012455	-0.061791	-0.071567	-0.037128	-0.050899	0.002196	-0.106715	1.000000

Sales Channel Analysis

```
sales_channel_analysis=data['Sales Channel'].value_counts()
sales_channel_analysis

plt.figure(figsize=(8,6))
sales_channel_analysis.plot(kind='bar', color='grey')
plt.title('Sales Channel Analysis')
plt.xlabel('Sales Channel')
plt.ylabel('Number of Orders')
plt.show()
```



Average Profit Margin for Amazon Sales data

```
data['Profit_Margin']=(data['Total Profit']/data['Total Revenue'])*100

average_profit=data['Profit_Margin'].mean()
round(average_profit,3)
```

```
36.212
```

Maximum Profit attained

```
max_profit=data['Profit_Margin'].max()
round(max_profit,3)
```

```
67.204
```

Minimum Profit obtained

```
min_profit=data['Profit_Margin'].min()
round(min_profit,3)
```

```
13.558
```

Average Time taken for delivery

```
data['Delivery Time']=(data['Ship Date']-data['Order Date']).dt.days

average_time=data['Delivery Time'].mean()
round(average_time,3)
```

```
23.36
```

```
max_time=data['Delivery Time'].max()
round(max_time,3)
```

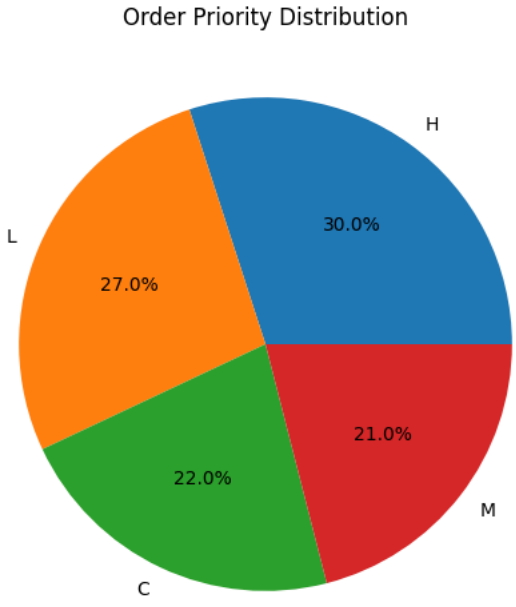
```
50
```

```
min_time=data['Delivery Time'].min()
round(min_time,3)
```

```
0
```

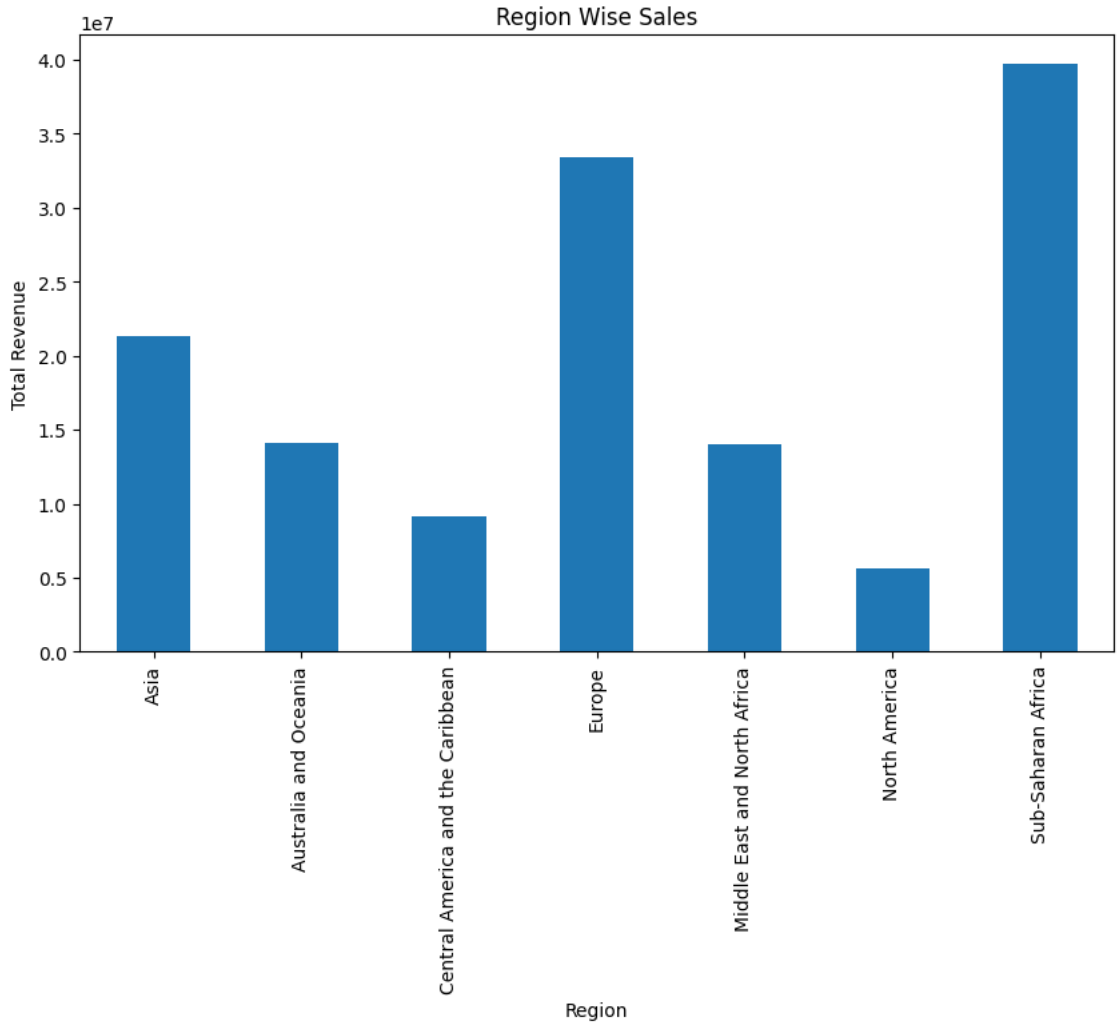
Order Priority Distribution

```
order_priority=data['Order Priority'].value_counts()

plt.figure(figsize=(8,6))
order_priority.plot(kind='pie',autopct='%1.1f%%')
plt.title('Order Priority Distribution')
plt.ylabel('')
plt.show()
```



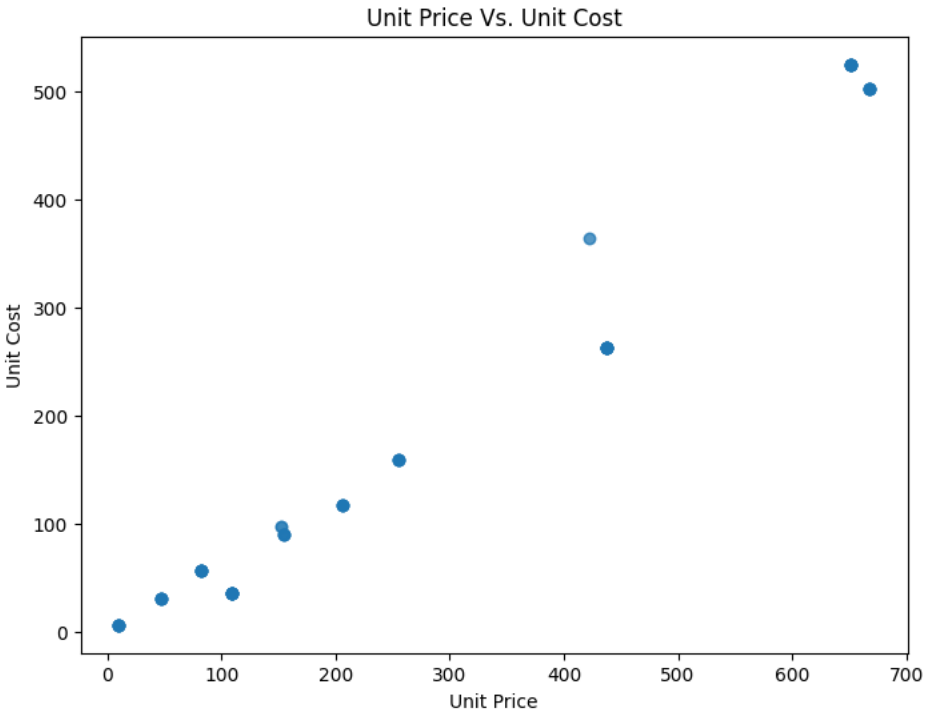
Region Wise Sales Analysis

```
region_sales=data.groupby('Region')['Total Revenue'].sum()

plt.figure(figsize=(10,6))
region_sales.plot(kind='bar')
plt.title('Region Wise Sales')
plt.xlabel('Region')
plt.ylabel('Total Revenue')
plt.show()
```



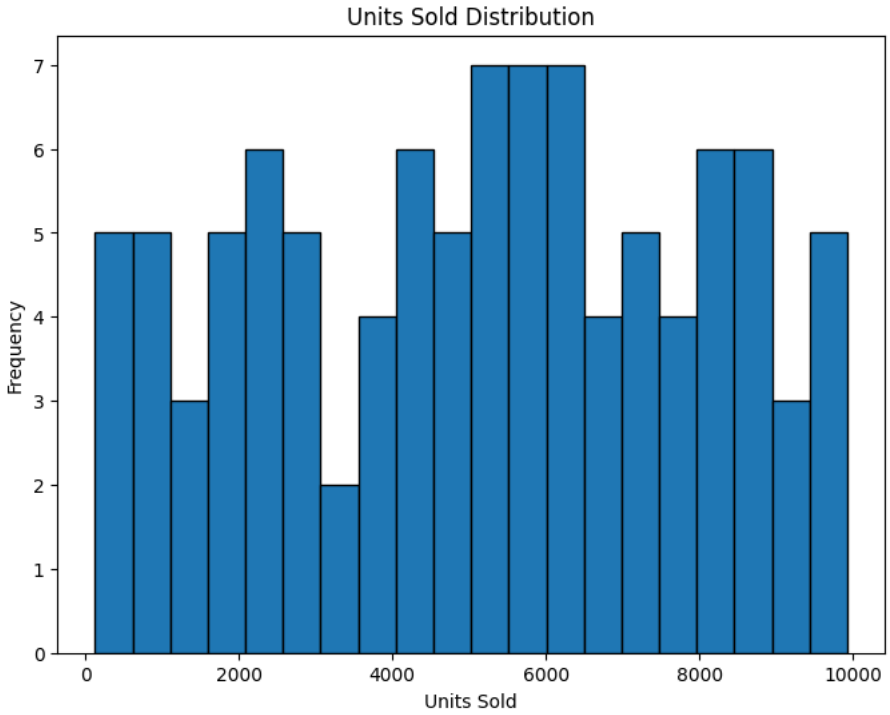
Comparison of Unit Price Vs Unit Cost

```
plt.figure(figsize=(8,6))
plt.scatter(data['Unit Price'],data['Unit Cost'],alpha=0.5)
plt.title('Unit Price Vs. Unit Cost')
plt.xlabel('Unit Price')
plt.ylabel('Unit Cost')
plt.show()
```



Histogram Distribution of Units Sold

```
plt.figure(figsize=(8,6))
plt.hist(data['Units Sold'],bins=20,edgecolor='black')
plt.title('Units Sold Distribution')
plt.xlabel('Units Sold')
plt.ylabel('Frequency')
plt.show()
```



Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.