```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data= pd.read_csv('Amazon Sales data.csv')
data= pd.DataFrame(data= data)
data
```

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010	9925	255.28	159.42	2533654.00	1582243.50	951
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012	2804	205.70	117.11	576782.80	328376.44	248
2	Europe	Russia	Office Supplies	Offline	L	05-02-2014	341417157	05-08-2014	1779	651.21	524.96	1158502.59	933903.84	224
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	07-05-2014	8102	9.33	6.92	75591.66	56065.84	19
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	02-01-2013	115456712	02-06-2013	5062	651.21	524.96	3296425.02	2657347.52	639
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
95	Sub-Saharan Africa	Mali	Clothes	Online	M	7/26/2011	512878119	09-03-2011	888	109.28	35.84	97040.64	31825.92	65
96	Asia	Malaysia	Fruits	Offline	L	11-11-2011	810711038	12/28/2011	6267	9.33	6.92	58471.11	43367.64	15
97	Sub-Saharan Africa	Sierra Leone	Vegetables	Offline	C	06-01-2016	728815257	6/29/2016	1485	154.06	90.93	228779.10	135031.05	93
98	North America	Mexico	Personal Care	Offline	M	7/30/2015	559427106	08-08-2015	5767	81.73	56.67	471336.91	326815.89	144
99	Sub-Saharan Africa	Mozambique	Household	Offline	L	02-10-2012	665095412	2/15/2012	5367	668.27	502.54	3586605.09	2697132.18	889

100 rows × 14 columns

Next steps: Generate code with data View recommended plots New interactive sheet

```
data.head()
```

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010	9925	255.28	159.42	2533654.00	1582243.50	951410.50
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012	2804	205.70	117.11	576782.80	328376.44	248406.36
2	Europe	Russia	Office Supplies	Offline	L	05-02-2014	341417157	05-08-2014	1779	651.21	524.96	1158502.59	933903.84	224598.75
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	07-05-2014	8102	9.33	6.92	75591.66	56065.84	19525.82

Next steps: Generate code with data View recommended plots New interactive sheet

```
data.columns
```

```
Index(['Region', 'Country', 'Item Type', 'Sales Channel', 'Order Priority',
       'Order Date', 'Order ID', 'Ship Date', 'Units Sold', 'Unit Price',
       'Unit Cost', 'Total Revenue', 'Total Cost', 'Total Profit'],
      dtype='object')
```

```
data.shape
```

```
(100, 14)
```

```
data.size
```

```
1400
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Region          100 non-null    object 
 1   Country         100 non-null    object 
 2   Item Type       100 non-null    object 
 3   Sales Channel   100 non-null    object 
 4   Order Priority  100 non-null    object 
 5   Order Date      100 non-null    object 
 6   Order ID        100 non-null    int64  
 7   Ship Date       100 non-null    object 
 8   Units Sold      100 non-null    int64  
 9   Unit Price      100 non-null    float64
 10  Unit Cost       100 non-null    float64
 11  Total Revenue   100 non-null    float64
 12  Total Cost      100 non-null    float64
 13  Total Profit    100 non-null    float64
dtypes: float64(5), int64(2), object(7)
memory usage: 11.1+ KB
```

```
data.describe()
```

	Order ID	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
count	1.000000e+02	100.000000	100.000000	100.000000	1.000000e+02	1.000000e+02	1.000000e+02
mean	5.550204e+08	5128.710000	276.761300	191.048000	1.373488e+06	9.318057e+05	4.416820e+05
std	2.606153e+08	2794.484562	235.592241	188.208181	1.460029e+06	1.083938e+06	4.385379e+05
min	1.146066e+08	124.000000	9.330000	6.920000	4.870260e+03	3.612240e+03	1.258020e+03
25%	3.389225e+08	2836.250000	81.730000	35.840000	2.687212e+05	1.688680e+05	1.214436e+05
50%	5.577086e+08	5382.500000	179.880000	107.275000	7.523144e+05	3.635664e+05	2.907680e+05
75%	7.907551e+08	7369.000000	437.200000	263.330000	2.212045e+06	1.613870e+06	6.358288e+05
max	9.940222e+08	9925.000000	668.270000	524.960000	5.997055e+06	4.509794e+06	1.719922e+06

```
data.isna().sum()
```

```
                     0
        Region       0
       Country       0
     Item Type       0
 Sales Channel       0
Order Priority       0
    Order Date       0
      Order ID       0
     Ship Date       0
    Units Sold       0
    Unit Price       0
     Unit Cost       0
 Total Revenue       0
    Total Cost       0
  Total Profit       0

dtype: int64
```

```
data.dtypes
```

```
                       0
        Region    object
       Country    object
     Item Type    object
 Sales Channel    object
Order Priority    object
    Order Date    object
      Order ID     int64
     Ship Date    object
    Units Sold     int64
    Unit Price   float64
     Unit Cost   float64
 Total Revenue   float64
    Total Cost   float64
  Total Profit   float64

dtype: object
```

```
data = data.astype({'Ship Date': 'datetime64[ns]','Order Date':'datetime64[ns]'})
```

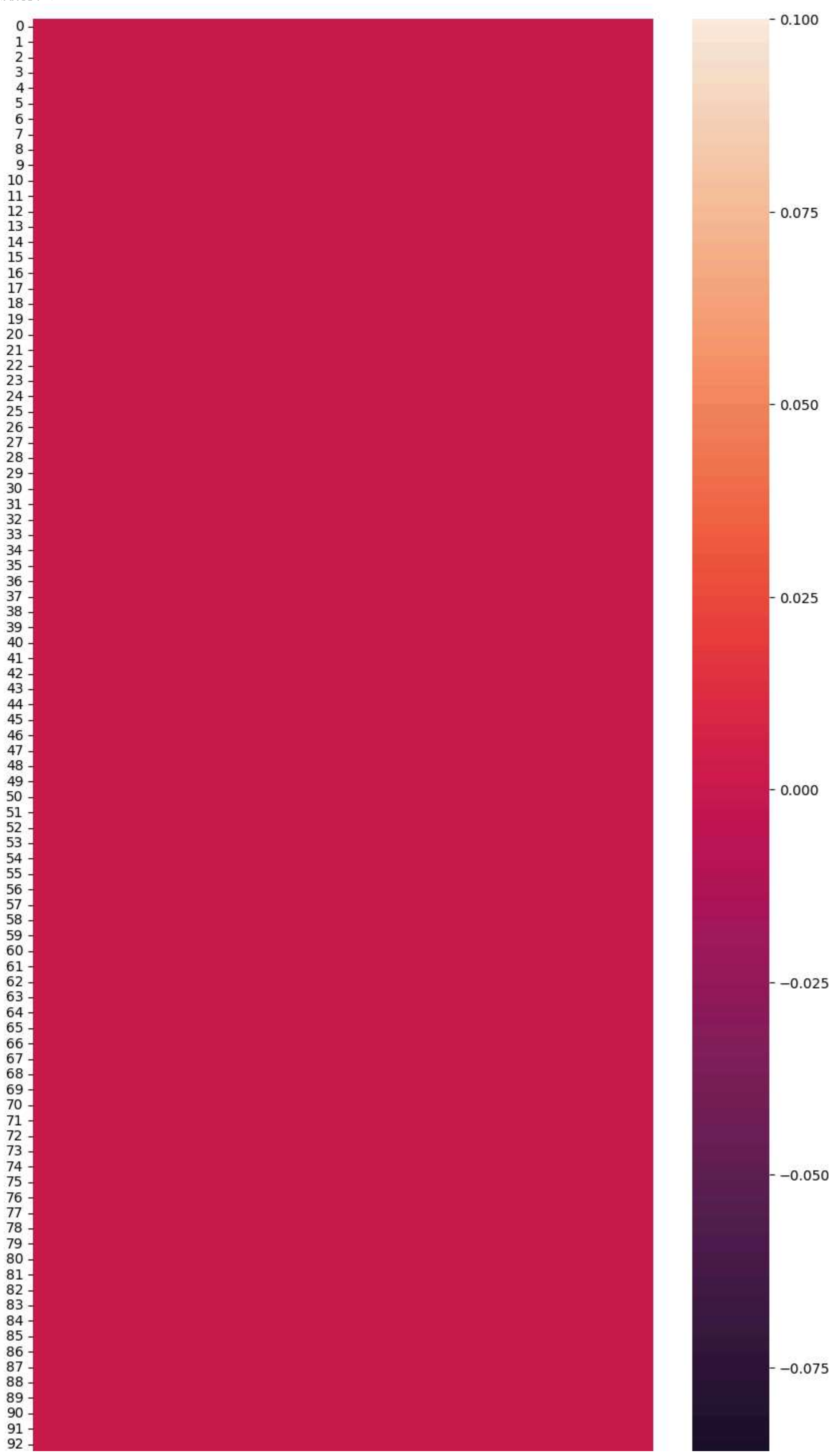
```
data.dtypes
```

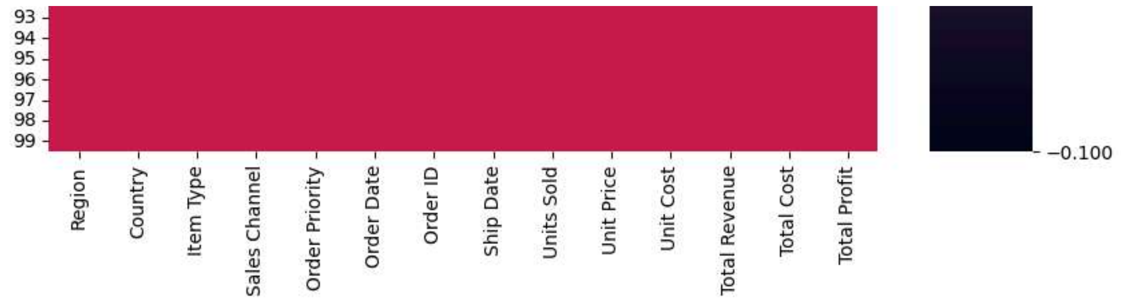
	0
Region	object
Country	object
Item Type	object
Sales Channel	object
Order Priority	object
Order Date	datetime64[ns]
Order ID	int64
Ship Date	datetime64[ns]
Units Sold	int64
Unit Price	float64
Unit Cost	float64
Total Revenue	float64
Total Cost	float64
Total Profit	float64

dtype: object

```
plt.figure(figsize=(10,20))
sns.heatmap(data.isnull()) # NO ANY NULL VALUE PRESENT IN OUR DATASET.
```

```
<Axes: >
```


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
−0.075

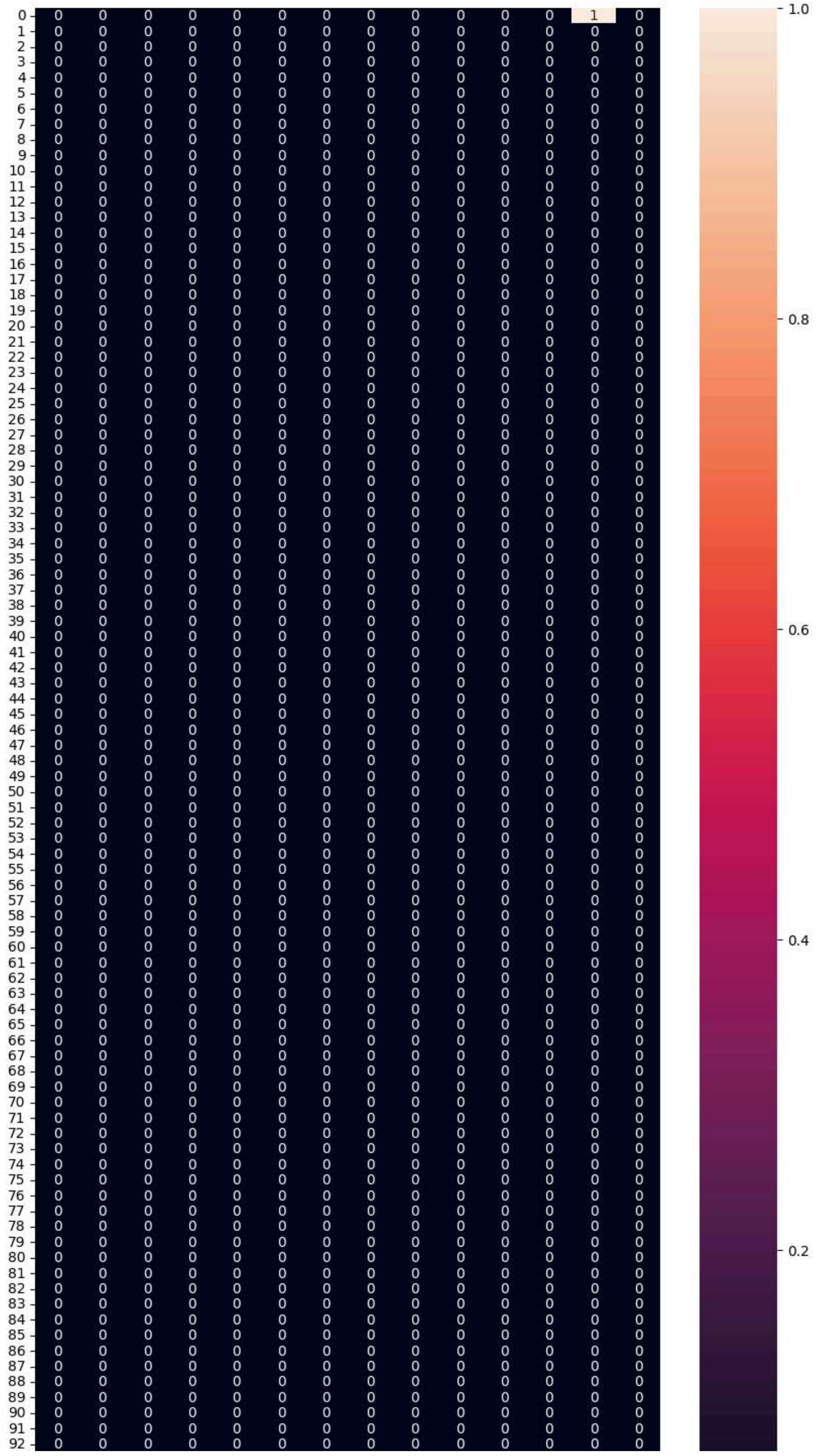


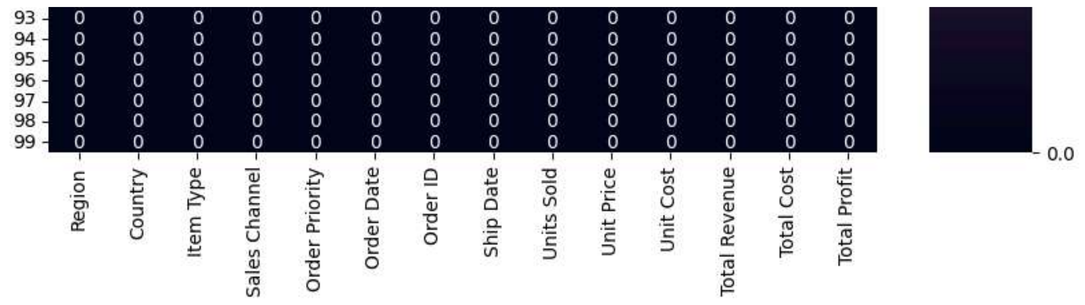
```
test = data.iloc[0, 12] = np.nan   # ADDING NULL VALUE JUST FOR DEMO
test
```

```
nan
```

```
plt.figure(figsize=(10,20))
sns.heatmap(data.isnull(),annot= True) #NULL VALUE FOUND IN 'TOTAL COST' COLUMN
```

```
<Axes: >
```


1.0
0.8
0.6
0.4
0.2



```
data = data.fillna(data.mean())   #FILL MEAN WHERE NULL VALUE PRESENT
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-17-6303f0ce277a> in <cell line: 1>()
----> 1 data = data.fillna(data.mean())   #FILL MEAN WHERE NULL VALUE PRESENT

                                    10 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/nanops.py in _ensure_numeric(x)
   1684             if inferred in ["string", "mixed"]:
   1685                 # GH#44008, GH#36703 avoid casting e.g. strings to numeric
-> 1686                 raise TypeError(f"Could not convert {x} to numeric")
   1687             try:
   1688                 x = x.astype(np.complex128)

TypeError: Could not convert ['Australia and OceaniaCentral America and the CaribbeanEuropeSub-Saharan AfricaSub-Saharan
AfricaAustralia and OceaniaSub-Saharan AfricaSub-Saharan AfricaSub-Saharan AfricaSub-Saharan AfricaAsiaSub-Saharan AfricaAsiaCentral
America and the CaribbeanAsiaEuropeAsiaSub-Saharan AfricaAsiaAustralia and OceaniaEuropeEuropeCentral America and the
CaribbeanAustralia and OceaniaEuropeEuropeAustralia and OceaniaSub-Saharan AfricaEuropeSub-Saharan AfricaEuropeSub-Saharan
AfricaAustralia and OceaniaAsiaSub-Saharan AfricaCentral America and the CaribbeanMiddle East and North AfricaSub-Saharan
AfricaAsiaEuropeSub-Saharan AfricaMiddle East and North AfricaSub-Saharan AfricaEuropeAsiaSub-Saharan AfricaEuropeEuropeEuropeSub-
Saharan AfricaEuropeSub-Saharan AfricaMiddle East and North AfricaSub-Saharan AfricaSub-Saharan AfricaSub-Saharan AfricaAustralia and
OceaniaEuropeEuropeSub-Saharan AfricaAustralia and OceaniaEuropeSub-Saharan AfricaMiddle East and North AfricaCentral America and the
CaribbeanSub-Saharan AfricaSub-Saharan AfricaCentral America and the CaribbeanEuropeSub-Saharan AfricaAsiaMiddle East and North
AfricaSub-Saharan AfricaSub-Saharan AfricaMiddle East and North AfricaNorth AmericaAustralia and OceaniaAsiaEuropeAustralia and
OceaniaEuropeMiddle East and North AfricaMiddle East and North AfricaSub-Saharan AfricaSub-Saharan AfricaNorth AmericaSub-Saharan
AfricaSub-Saharan AfricaMiddle East and North AfricaEuropeSub-Saharan AfricaAustralia...
```

Next steps: Explain error

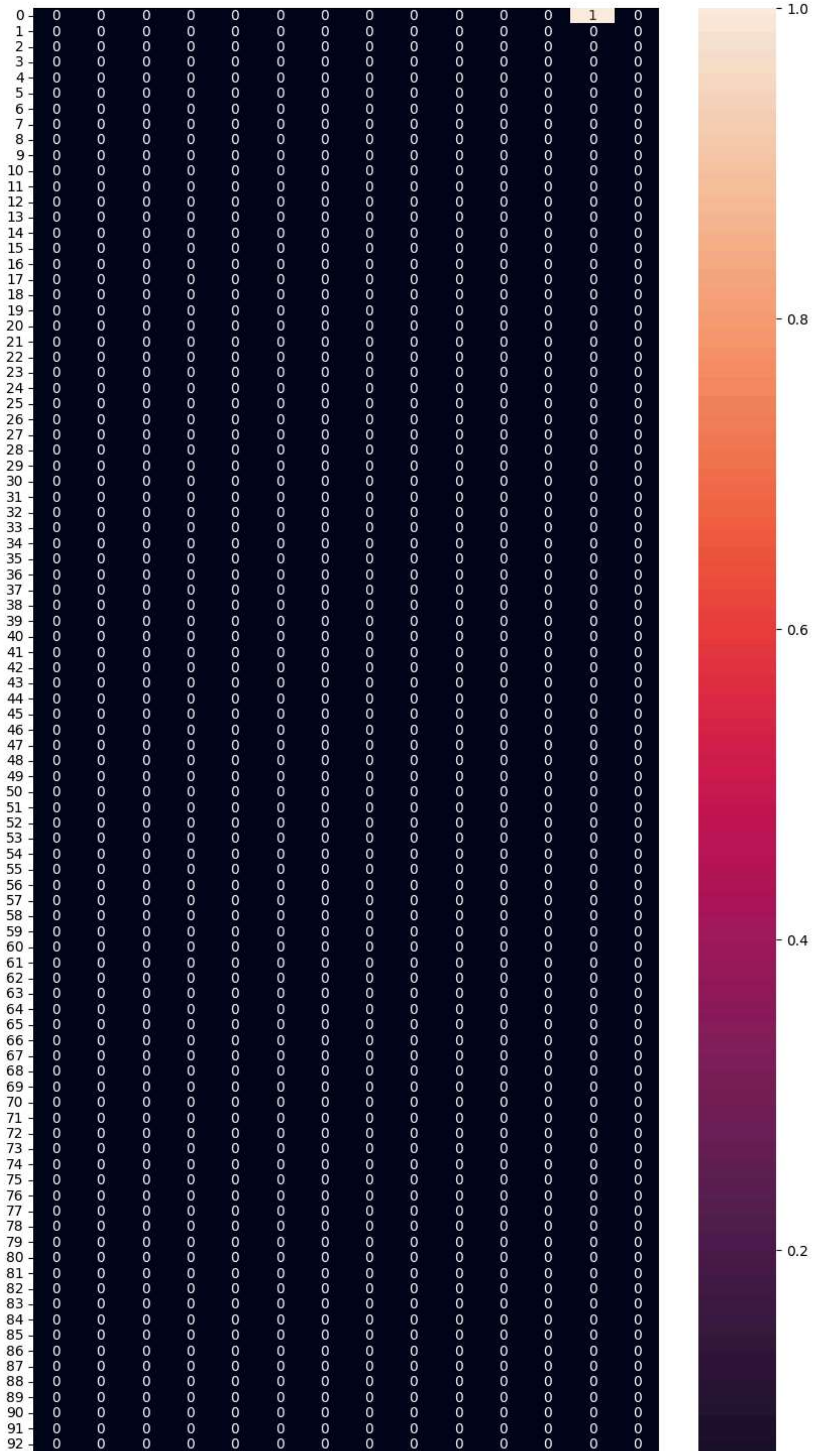
```
data['Total Cost']= data['Total Cost'].astype('Float64')
data
```

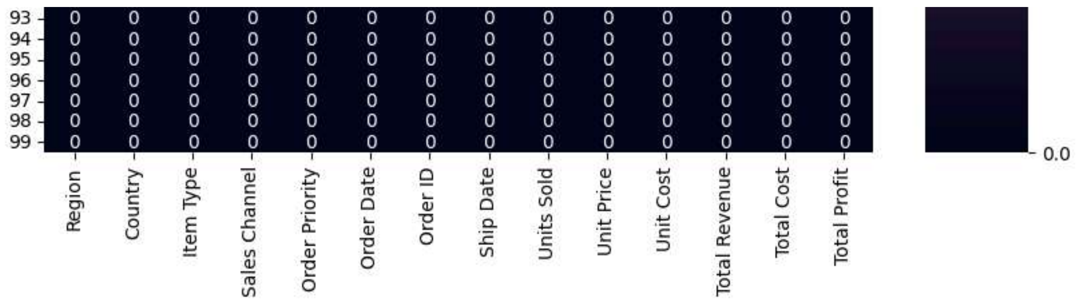
	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	2010-05-28	669165933	2010-06-27	9925	255.28	159.42	2533654.00	<NA>	951410.50
1	Central America and the Caribbean	Grenada	Cereal	Online	C	2012-08-22	963881480	2012-09-15	2804	205.70	117.11	576782.80	328376.44	248406.36
2	Europe	Russia	Office Supplies	Offline	L	2014-05-02	341417157	2014-05-08	1779	651.21	524.96	1158502.59	933903.84	224598.75
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	2014-06-20	514321792	2014-07-05	8102	9.33	6.92	75591.66	56065.84	19525.82
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2013-02-01	115456712	2013-02-06	5062	651.21	524.96	3296425.02	2657347.52	639077.50
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	Sub-Saharan Africa	Mali	Clothes	Online	M	2011-07-26	512878119	2011-09-03	888	109.28	35.84	97040.64	31825.92	65214.72
96	Asia	Malaysia	Fruits	Offline	L	2011-11-11	810711038	2011-12-28	6267	9.33	6.92	58471.11	43367.64	15103.47
97	Sub-Saharan Africa	Sierra Leone	Vegetables	Offline	C	2016-06-01	728815257	2016-06-29	1485	154.06	90.93	228779.10	135031.05	93748.05
98	North	Mexico	Personal	Offline	M	2015-	559427106	2015-	5767	81.73	56.67	471336.91	326815.89	144521.02

Next steps: Generate code with `data` View recommended plots New interactive sheet

```
plt.figure(figsize=(10,20))
sns.heatmap(data.isnull(),annot= True)   # NO NULL VALUES
```

```
<Axes: >
```


0
1
...
92
1.0
0.8
0.6
0.4
0.2



```
data.head(3)
```

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	2010-05-28	669165933	2010-06-27	9925	255.28	159.42	2533654.00	<NA>	951410.50
1	Central America and the Caribbean	Grenada	Cereal	Online	C	2012-08-22	963881480	2012-09-15	2804	205.70	117.11	576782.80	328376.44	248406.36
2	Europe	Russia	Office	Offline	L	2014-	341417157	2014-	1779	651.21	524.96	1158502.59	933903.84	224598.75

Next steps: Generate code with data View recommended plots New interactive sheet

Data Analysis:

Queries:

Which regions have the highest total sales revenue?

What is the average unit price and unit cost for each item type?

Which country has the highest total profit?

How does the sales channel affect the order priority distribution?

What is the average order processing time (duration between order and ship dates) for each sales channel?

Which item types have the highest and lowest total sales?

How does the order priority vary across different regions?

What is the correlation between unit price and total profit?

Are there any seasonal trends or patterns in the sales data?

How does the number of units sold vary across different countries?

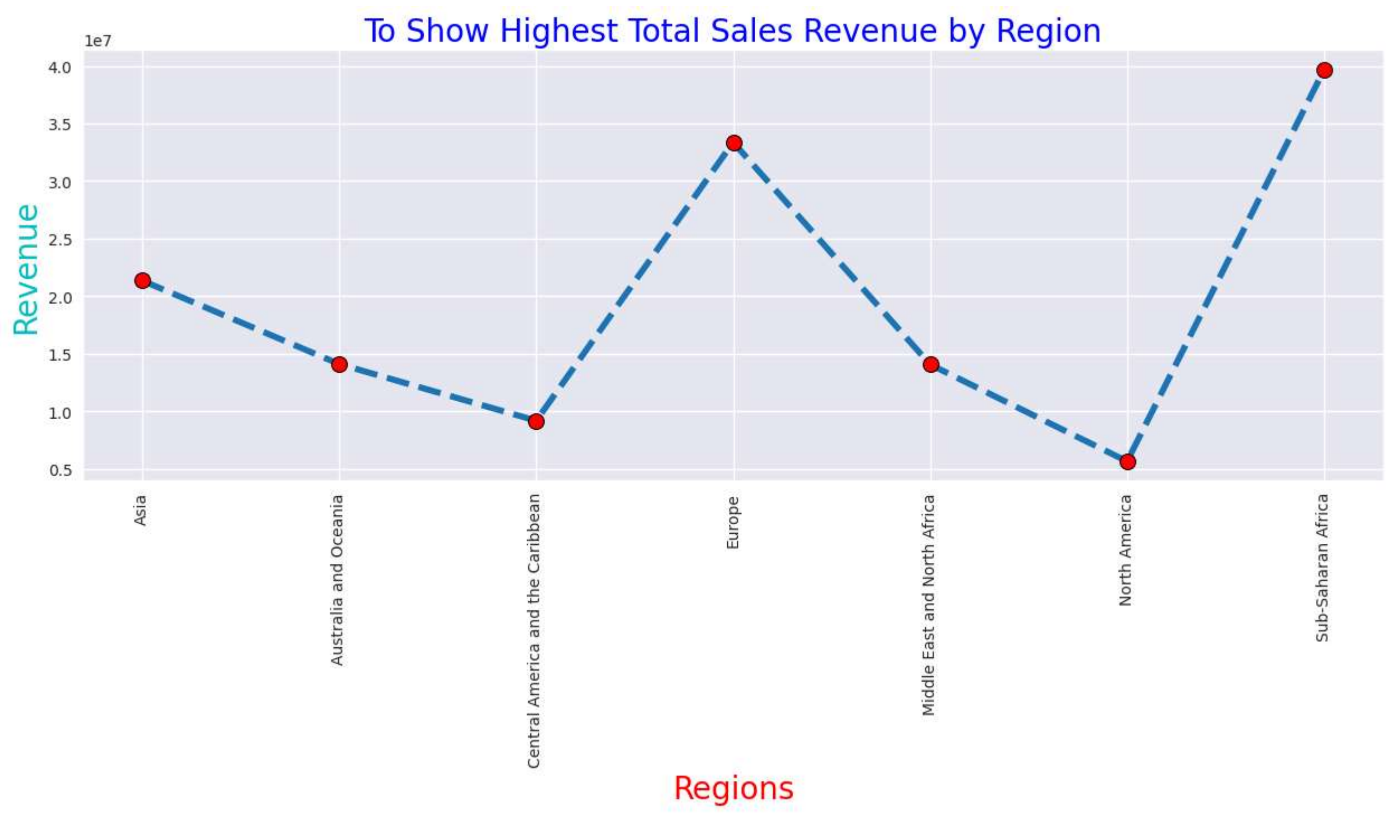
1- Which regions have the highest total sales revenue?

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

```
'Sub-Saharan Africa'
```

```
group_data= data.groupby(data['Region'])['Total Revenue'].sum()
sns.set_style('darkgrid')
plt.figure(figsize=(15,5))
sns.lineplot(data= group_data, linestyle= '--' ,linewidth= 4 , marker= 'o', markersize= 10,
             markerfacecolor='red', markeredgecolor='black')

plt.xticks(rotation= 90)
plt.title('To Show Highest Total Sales Revenue by Region', fontsize= 20, color= 'Blue')
plt.xlabel('Regions', fontsize= 20, color= 'red')
plt.ylabel('Revenue', fontsize= 20, color= 'c')
plt.show()
```

```
# 1e7 is scientific form. it means 1*10**7= 10,000,000
```



2- What is the average unit price and unit cost for each item type?

```
Avg_Unit_Price= data.groupby(data['Item Type'])['Unit Price'].mean()
Avg_Unit_Cost= data.groupby(data['Item Type'])['Unit Cost'].mean()

Avg_Price_Cost= pd.DataFrame({'Average Unit Price': Avg_Unit_Price,
                              'Average Unit Cost': Avg_Unit_Cost})

Avg_Price_Cost
```

	Average Unit Price	Average Unit Cost
Item Type		
Baby Food	255.28	159.42
Beverages	47.45	31.79
Cereal	205.70	117.11
Clothes	109.28	35.84
Cosmetics	437.20	263.33
Fruits	9.33	6.92
Household	668.27	502.54
Meat	421.89	364.69
Office Supplies	651.21	524.96
Personal Care	81.73	56.67
Snacks	152.58	97.44
Vegetables	154.06	90.93

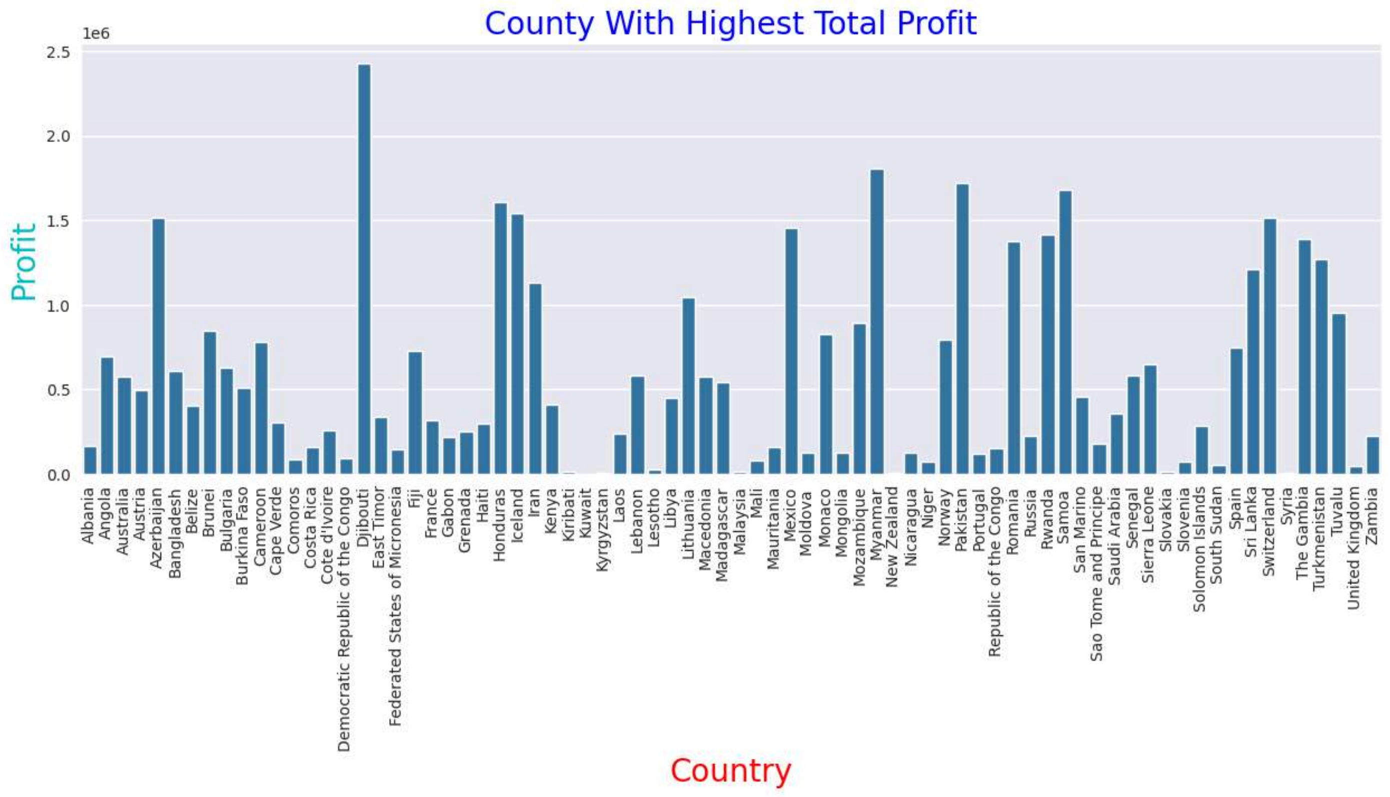
Next steps: Generate code with Avg_Price_Cost | View recommended plots | New interactive sheet

3- Which country has the highest total profit?

```
Total_Profit_By_Comapany= data.groupby(data['Country']) ['Total Profit'].sum()
Highest_Total_Profit_County= Total_Profit_By_Comapany.idxmax()

print("Country with the highest total profit:",Highest_Total_Profit_County)
```

```
Country with the highest total profit: Djibouti
```

```
group_data= data.groupby(data['Country']) ['Total Profit'].sum()
sns.set_style('darkgrid')
plt.figure(figsize=(15,5))
sns.barplot(x= group_data.index, y= group_data )

plt.xticks(rotation= 90)
plt.title('County With Highest Total Profit', fontsize= 20, color= 'Blue')
plt.xlabel('Country', fontsize= 20, color= 'red')
plt.ylabel('Profit', fontsize= 20, color= 'c')
plt.show()
```



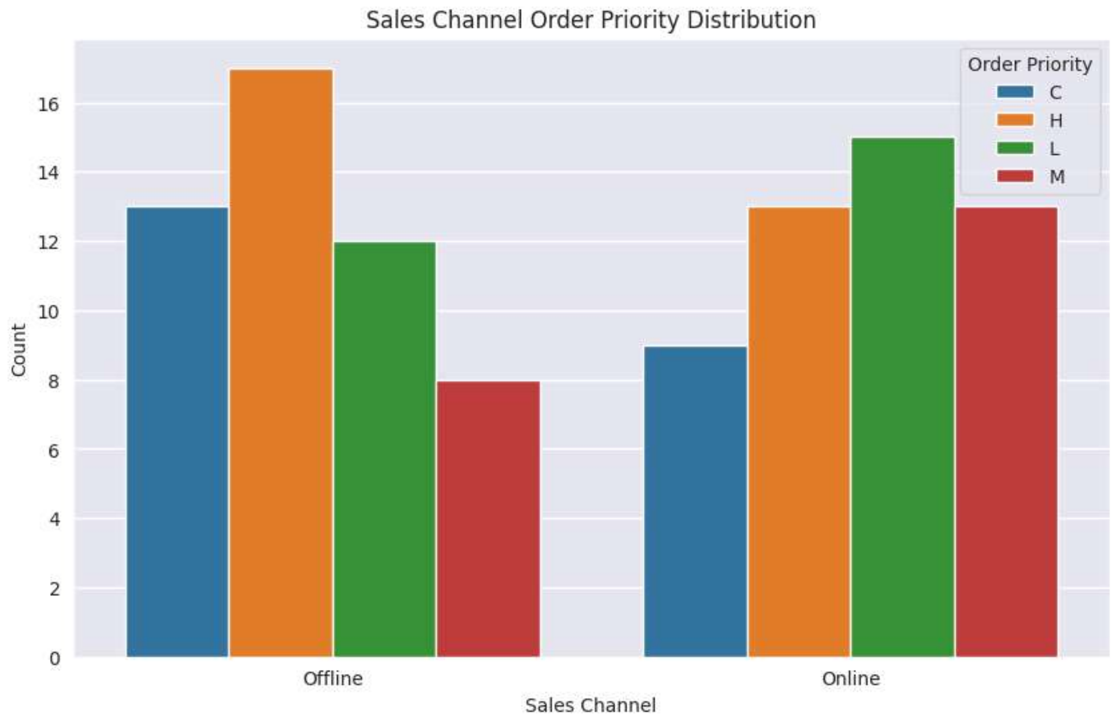
4- How does the sales channel affect the order priority distribution?

```
Sales_Channel_Order_Priority_Distribution= data.groupby(data['Sales Channel']) ['Order Priority'].value_counts()
Sales_Channel_Order_Priority_Distribution
```

```
                              count
Sales Channel  Order Priority
Offline        H                 17
               C                 13
               L                 12
               M                  8
Online         L                 15
               H                 13
               M                 13
               C                  9

dtype: int64
```

```
Sales_Channel_Order_Priority_Distribution = data.groupby(['Sales Channel', 'Order Priority'])['Order Priority'].count()

# Reset the index to convert the grouped data into a DataFrame
Sales_Channel_Order_Priority_Distribution = Sales_Channel_Order_Priority_Distribution.reset_index(name='Count')

# Set the style
sns.set_style('darkgrid')

# Create the bar plot
plt.figure(figsize=(10, 6))
sns.barplot(x='Sales Channel', y='Count', hue='Order Priority', data=Sales_Channel_Order_Priority_Distribution)

# Add labels and title
plt.xlabel('Sales Channel')
plt.ylabel('Count')
plt.title('Sales Channel Order Priority Distribution')

# Display the plot
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
```

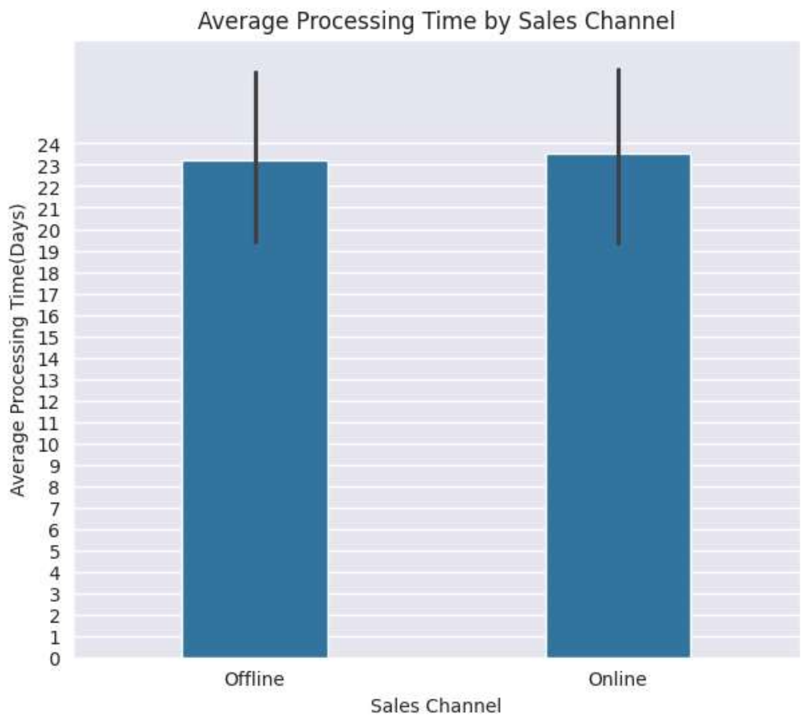


5- What is the average order processing time (duration between order and ship dates) for each sales channel?

```
data['Processing Time']= data['Ship Date']-data['Order Date']

Avg_Processing_Time= data.groupby(data['Sales Channel'])['Processing Time'].mean()
Avg_Processing_Time
```

```
                Processing Time
 Sales Channel
       Offline  23 days 04:48:00
        Online  23 days 12:28:48

dtype: timedelta64[ns]
```

```
plt.figure(figsize=(7, 6))

sns.barplot(data= data, x= data['Sales Channel'], y=data['Processing Time'].dt.days, width= 0.4 )

plt.title('Average Processing Time by Sales Channel')
plt.xlabel('Sales Channel')
plt.yticks(np.arange(0,25,1))
plt.ylabel('Average Processing Time(Days)')

plt.show()
```

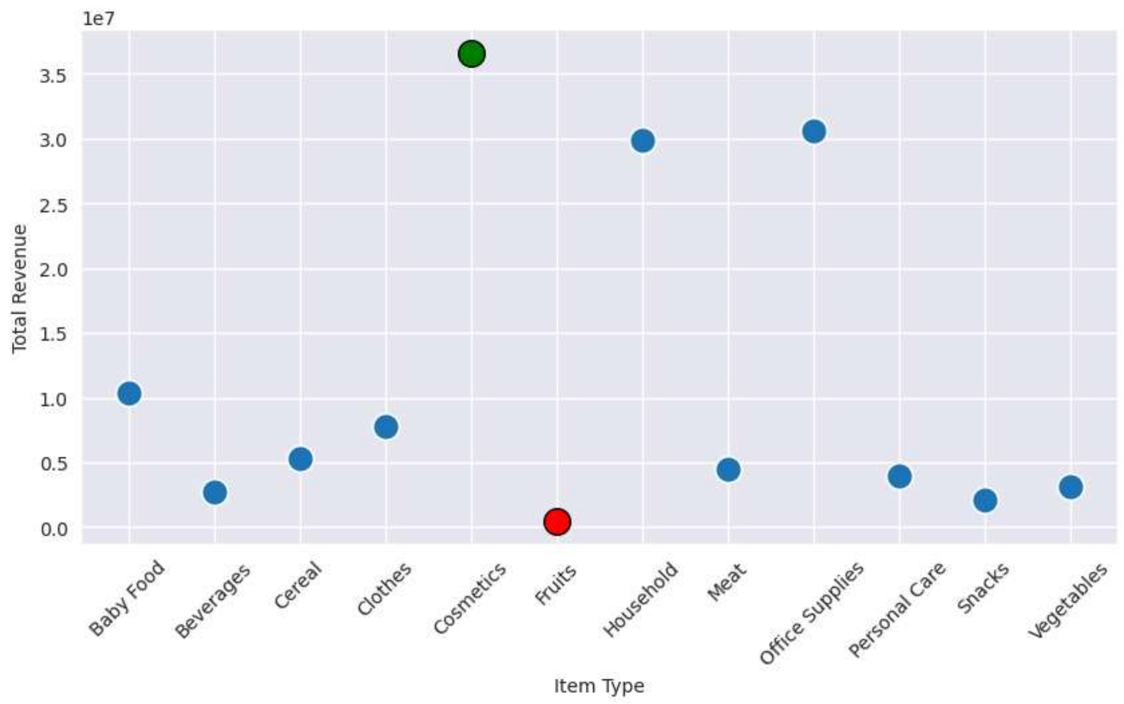


6- Which item types have the highest and lowest total sales?

```
group_item_type= data.groupby(data['Item Type'])['Total Revenue'].sum()

highest_sales_revenue_item_type= group_item_type.idxmax()
lowest_sales_revenue_item_type= group_item_type.idxmin()

print("{'Highest Sales Revenue By Item Type':", highest_sales_revenue_item_type, "\n'Lowest Sales Revenue By Item Type':", lowest_sales_reve
```

```
{'Highest Sales Revenue By Item Type': Cosmetics
'Lowest Sales Revenue By Item Type': Fruits }
```

```
plt.figure(figsize=(10,5))

# Highlight Max Value
sns.scatterplot(x=group_item_type.index, y=group_item_type, s=200)
max_index = group_item_type.idxmax()
plt.scatter(x=max_index, y=group_item_type[max_index], s=200, color='Green', edgecolor='black')

# Highlight the minimum value
min_index = group_item_type.idxmin()
plt.scatter(x=min_index, y=group_item_type[min_index], s=200, color='RED', edgecolor='black')

plt.yticks(rotation= 0)
plt.xticks(rotation= 45)
plt.show()
```



7- How does the order priority vary across different regions?

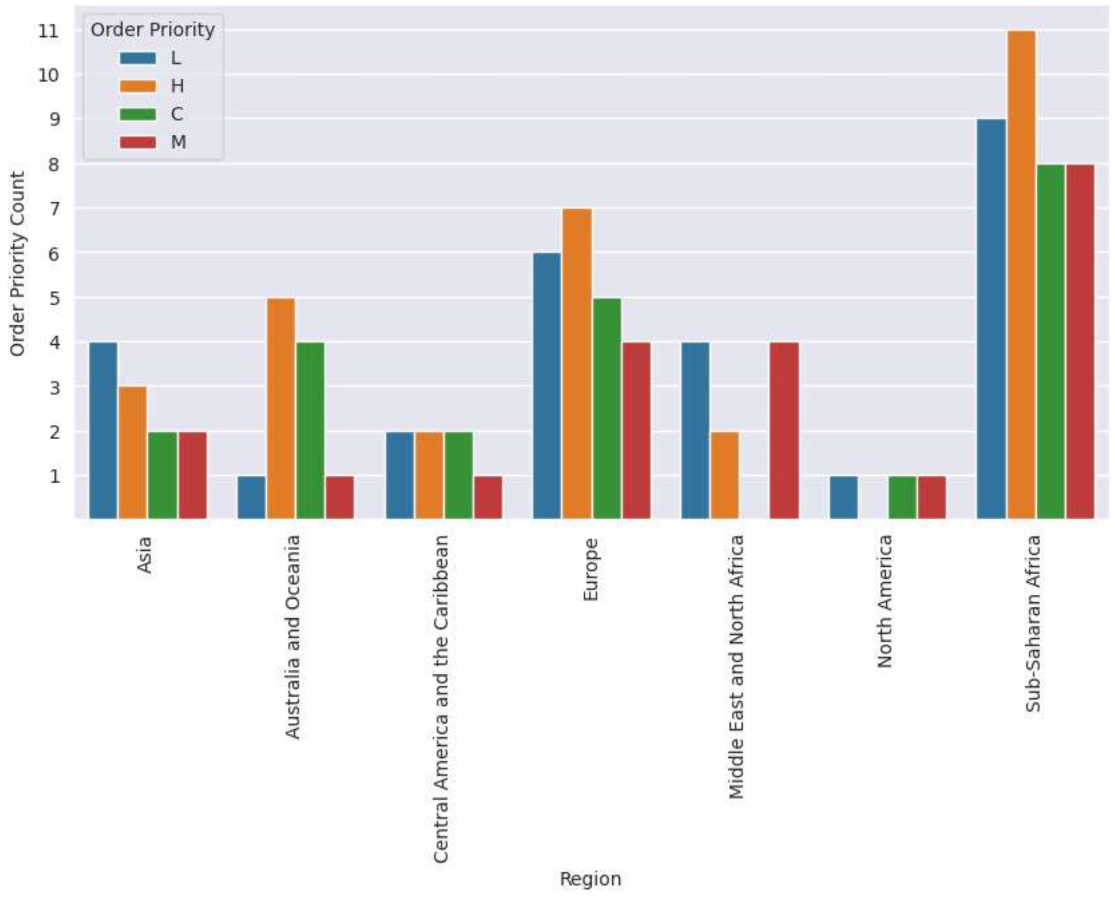
```
Diff_regions_by_order_priority= data.groupby(data['Region'])['Order Priority'].value_counts()
Diff_regions_by_order_priority
```

Region	Order Priority	count
Asia	L	4
	H	3
	C	2
	M	2
Australia and Oceania	H	5
	C	4
	L	1
	M	1
Central America and the Caribbean	C	2
	H	2
	L	2
	M	1
Europe	H	7
	L	6
	C	5
	M	4
Middle East and North Africa	L	4
	M	4
	H	2
North America	C	1
	L	1
	M	1
Sub-Saharan Africa	H	11
	L	9
	C	8
	M	8

dtype: int64

```
Diff_regions_by_order_priority= data.groupby(data['Region'])['Order Priority'].value_counts().reset_index(name='Order Priority Count')
plt.figure(figsize= (10,5))
sns.barplot(data= Diff_regions_by_order_priority, x= 'Region', y= 'Order Priority Count', hue= 'Order Priority')
plt.xticks(rotation= 90)
plt.yticks(np.arange(1,12,1))

plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
```



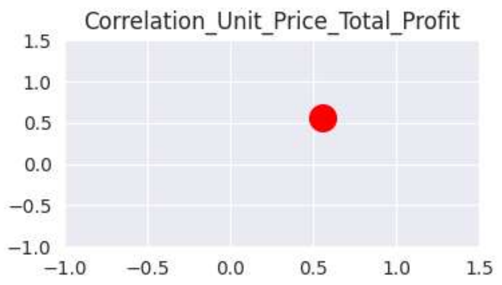
8- What is the correlation between unit price and total profit?

```
Correlation_Unit_Price_Total_Profit= data['Unit Price'].corr(data['Total Profit'])

print("Correlation between Unit Price and Total Profit:", Correlation_Unit_Price_Total_Profit)
```

```
Correlation between Unit Price and Total Profit: 0.5573652488121267
```

```
plt.figure(figsize=(4,2))
plt.scatter(x= Correlation_Unit_Price_Total_Profit, y= Correlation_Unit_Price_Total_Profit, s= 200, color= 'RED' )
plt.xticks(np.arange(-1,2,0.5))
plt.yticks(np.arange(-1,2,0.5))
plt.title('Correlation_Unit_Price_Total_Profit')

plt.show
```

```
matplotlib.pyplot.show
def show(*args, **kwargs)

/usr/local/lib/python3.10/dist-packages/matplotlib/pyplot.py
Display all open figures.

Parameters
----------
block : bool, optional
```



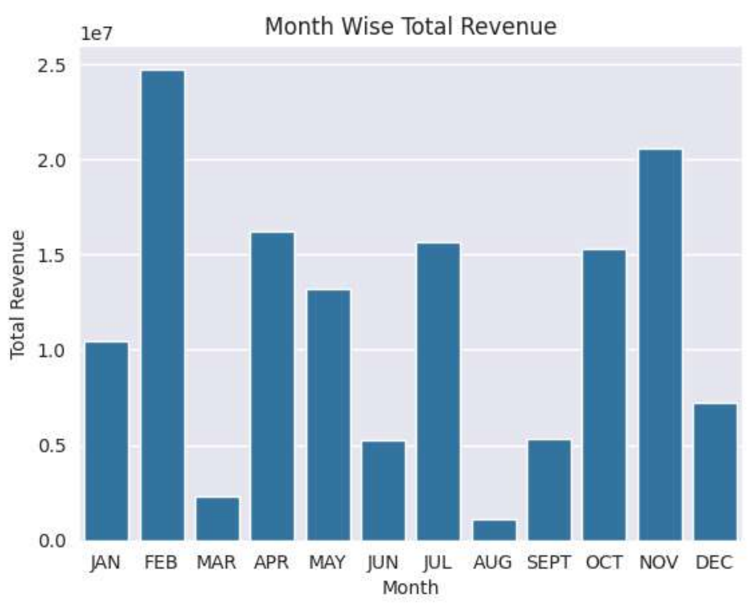
9- Are there any seasonal trends or patterns in the sales data?

```
month_names= {1: 'JAN',
              2: 'FEB',
              3: 'MAR',
              4: 'APR',
              5: 'MAY',
              6: 'JUN',
              7: 'JUL',
              8: 'AUG',
              9: 'SEPT',
             10: 'OCT',
             11: 'NOV',
             12: 'DEC'}
monthly_sales = data.groupby(data['Order Date'].dt.month)['Total Revenue'].sum()
monthly_sales.index= monthly_sales.index.map(month_names)

monthly_sales
```

Order Date	Total Revenue
JAN	10482467.12
FEB	24740517.77
MAR	2274823.87
APR	16187186.33
MAY	13215739.99
JUN	5230325.77
JUL	15669518.50
AUG	1128164.91
SEPT	5314762.56
OCT	15287576.61
NOV	20568222.76
DEC	7249462.12

dtype: float64

```
sns.barplot(x= monthly_sales.index, y= monthly_sales)
plt.title('Month Wise Total Revenue')
plt.xlabel('Month')
plt.ylabel('Total Revenue')
plt.show()
```



10- How does the number of units sold vary across different countries?

```
Diff_countries_by_unit_sold= data.groupby(data['Country'])['Units Sold'].sum().reset_index(name= 'Unit Sold')
pd.set_option('display.max_rows',None)
Diff_countries_by_unit_sold
```

	Country	Unit Sold
0	Albania	2269
1	Angola	4187
2	Australia	12995
3	Austria	2847
4	Azerbaijan	9255
5	Bangladesh	8263
6	Belize	5498
7	Brunei	6708
8	Bulgaria	5660
9	Burkina Faso	8082
10	Cameroon	10948
11	Cape Verde	4168
12	Comoros	962
13	Costa Rica	6409
14	Cote d'Ivoire	3482
15	Democratic Republic of the Congo	5741
16	Djibouti	23198
17	East Timor	5908
18	Federated States of Micronesia	9379
19	Fiji	9905
20	France	1815
21	Gabon	8656
22	Grenada	2804
23	Haiti	1705
24	Honduras	11199
25	Iceland	8867
26	Iran	6489
27	Kenya	6457
28	Kiribati	5398
29	Kuwait	522
30	Kyrgyzstan	124
31	Laos	3732
32	Lebanon	7884
33	Lesotho	9606
34	Libya	6789
35	Lithuania	8287
36	Macedonia	7842
37	Madagascar	7342
38	Malaysia	6267
39	Mali	6710
40	Mauritania	1266
41	Mexico	19143
42	Moldova	5070
43	Monaco	8614
44	Mongolia	4901
45	Mozambique	5367