Analyzing Amazon sales data

Problem Statement:

Sales management has gained importance to meet increasing competition and the need for improved methods of distribution to reduce cost and to increase profits. Sales management today is the most important function in a commercial and business enterprise. Do ETL: Extract-Transform-Load some Amazon dataset and find for me Sales-trend -> month-wise, year-wise, yearly_month-wise Find key metrics and factors and show the meaningful relationships between attributes. Do your own research and come up with your findings.

Importing the necessary libraries

In [2]:
```
import numpy as np # For numpy Array
import pandas as pd # Pandas Data frame
import matplotlib.pyplot as plt # For Matplotlib Visualization
import seaborn as sns # for seaborn visualization
%matplotlib inline
sns.set(color_codes=True)
from scipy import stats # for statastics
import warnings # For warning
warnings.filterwarnings("ignore")
```

Load the dataset into dataframe

In [3]:
```
# Load CSV File
df =pd.read_csv('Amazon Sales data.csv')
df.head(100)
```

Out[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	Tot Pro
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.
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.
2	Europe	Russia	Office Supplies	Offline	L	5/2/2014	341417157	5/8/2014	1779	651.21	524.96	1158502.59	933903.84	224598.
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	7/5/2014	8102	9.33	6.92	75591.66	56065.84	19525.
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	115456712	2/6/2013	5062	651.21	524.96	3296425.02	2657347.52	639077.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
95	Sub-Saharan Africa	Mali	Clothes	Online	M	7/26/2011	512878119	9/3/2011	888	109.28	35.84	97040.64	31825.92	65214.
96	Asia	Malaysia	Fruits	Offline	L	11/11/2011	810711038	12/28/2011	6267	9.33	6.92	58471.11	43367.64	15103.
97	Sub-Saharan Africa	Sierra Leone	Vegetables	Offline	C	6/1/2016	728815257	6/29/2016	1485	154.06	90.93	228779.10	135031.05	93748.
98	North America	Mexico	Personal Care	Offline	M	7/30/2015	559427106	8/8/2015	5767	81.73	56.67	471336.91	326815.89	144521.
99	Sub-Saharan Africa	Mozambique	Household	Offline	L	2/10/2012	665095412	2/15/2012	5367	668.27	502.54	3586605.09	2697132.18	889472.

100 rows × 14 columns

Observation

1: Read the dataset

In [4]:
```
## Check the Data Shape of df
df.shape ## 100 row , 14 col
```

Out[4]:
```
(100, 14)
```

In [5]:
```
df.columns
```

Out[5]:
```
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')
```

Now we observe the each features present in the dataset.

`Region:` The Region feature is contain regoin name.
`Country:` The Country feature contain Country name.
`Item Type:` The Item Type describes which type of Item has been available for sales.
`Sales Channel:` Sales Channel elaborate sale mode which online or offline mode.
`Order Priority:` It's say order priority.
`Order Date:` It contain order date.
`Order ID:` These columns contain order ids.
`Ship Date:` These column contain shipping date of orders.
`Units Sold:` These columns contain information of how many unit sold.
`Unit Price:` These columns contain unit price of each unit.
`Unit Cost:` It's say about the unit cost.
`Total Revenue:` The feature says about total revenue of amazon.
`Total Cost:` These columns having total cost of each unit products.
`Total Profit:` These column contain total profit of each products.

Check the Datatypes

In [6]:
```
# Get the datatypes of each columns number of records in each column.
df.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
```

Observation

1: In these data frame the data types are `int` , `float` , `Object`

Check Missing Values

In [7]:
```
# Check Missing Values
df.isnull().sum()
```

Out[7]:
```
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
```

Observation

1: Here we observe No null values are available in dataset

2: So no need to remove null Values

In [8]:
```
# Generating descriptive statistics
df.describe()
```

Out[8]:

	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

Observation

1: Here we got descriptive statastics of all numerical features

In [9]:
```
df.describe(include='all')
```

Out[9]:

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	T
count	100	100	100	100	100	100	1.000000e+02	100	100.000000	100.000000	100.000000	1.000000e+02	1.00
unique	7	76	12	2	4	100	NaN	99	NaN	NaN	NaN	NaN	
top	Sub-Saharan Africa	The Gambia	Clothes	Offline	H	5/28/2010	NaN	11/17/2010	NaN	NaN	NaN	NaN	
freq	36	4	13	50	30	1	NaN	2	NaN	NaN	NaN	NaN	
mean	NaN	NaN	NaN	NaN	NaN	NaN	5.550204e+08	NaN	5128.710000	276.761300	191.048000	1.373488e+06	9.31
std	NaN	NaN	NaN	NaN	NaN	NaN	2.606153e+08	NaN	2794.484562	235.592241	188.208181	1.460029e+06	1.08
min	NaN	NaN	NaN	NaN	NaN	NaN	1.146066e+08	NaN	124.000000	9.330000	6.920000	4.870260e+03	3.61
25%	NaN	NaN	NaN	NaN	NaN	NaN	3.389225e+08	NaN	2836.250000	81.730000	35.840000	2.687212e+05	1.68
50%	NaN	NaN	NaN	NaN	NaN	NaN	5.577086e+08	NaN	5382.500000	179.880000	107.275000	7.523144e+05	3.63
75%	NaN	NaN	NaN	NaN	NaN	NaN	7.907551e+08	NaN	7369.000000	437.200000	263.330000	2.212045e+06	1.61
max	NaN	NaN	NaN	NaN	NaN	NaN	9.940222e+08	NaN	9925.000000	668.270000	524.960000	5.997055e+06	4.50

Observation

1: Here we got descriptive stats of categorical columns

In [10]:
```
df.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
```

In [11]:
```
## Check duplicate value in df
df.duplicated().sum()
```

Out[11]:
```
0
```

Observation

1: In the dataset we saw there is no duplicated values

Finding outliers using statistical methods

In [12]:
```
##Finding outliers using statistical methods
def find_outlier(df):
    q1=df.quantile(0.25)
    q3=df.quantile(0.75)
    IQR =q3-q1
    outliers=df[((df<(q1-1.5*IQR))|(df>(q3+1.5*IQR)))]
    return outliers
```

In [13]:
```
outliers = find_outlier(df)
print(f'number of outliers:{(len(outliers))}')
```

```
number of outliers:100
```

Observation

1: Above observation we found 100 outliers in whole dataset.

In [14]:
```
## Check the dataset Shape
df.shape
```

Out[14]:
```
(100, 14)
```

In [15]:
```
df.head()
```

Out[15]:

	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	5/2/2014	341417157	5/8/2014	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	7/5/2014	8102	9.33	6.92	75591.66	56065.84	19525.82
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	115456712	2/6/2013	5062	651.21	524.96	3296425.02	2657347.52	639077.50

Checking Correlation of colunms

In [17]:
```
plt.figure(figsize=(12,5))  # Here we have given figure size
sns.heatmap(df.corr(method='pearson'), annot=True, vmin=-1, vmax=1,cmap='YlGnBu',linewidths=0.9,linecolor='white')
```

Out[17]: <AxesSubplot:>

Observation of heatmap

Heatmap

The heatmap is two dimensional representation of data in which values by representaion by colour.

1: See there We have +1 to -1 scale for colour.In these graph dark blue color represent the max values present there and faint color value represent the min value present at there.

2: Heatmap is used to checking the which feature is **+ve or -ve** correlated to each other.

3: In these heatmap Total profit is highly correlated to unit sold,unit price,unit cost,total Revenue,total cost.

In [18]:
```
## Copy The df data into df1
df1 =df.copy()
df1.head() # print Head of df1
```

Out[18]:

	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	5/2/2014	341417157	5/8/2014	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	7/5/2014	8102	9.33	6.92	75591.66	56065.84	19525.82
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	115456712	2/6/2013	5062	651.21	524.96	3296425.02	2657347.52	639077.50

In [19]:
```
## Check the data info
df1.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
```

In [20]:
```
## Check Null Values overhere
df1.isnull().sum()
```

Out[20]:
```
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
```

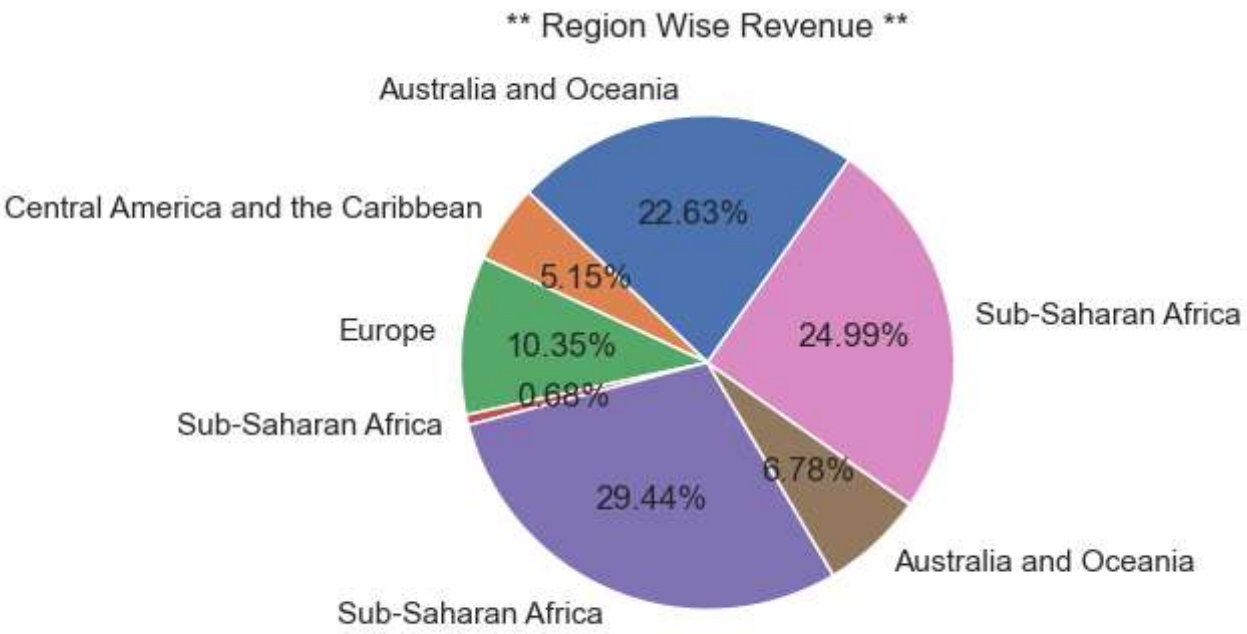
In [21]:
```
### Top Revenue Region
top10 = df1.groupby('Region').sum().sort_values('Total Revenue', ascending = False)
top10 = df1.reset_index().head(7)
```

In [22]:
```
top10
```

Out[22]:

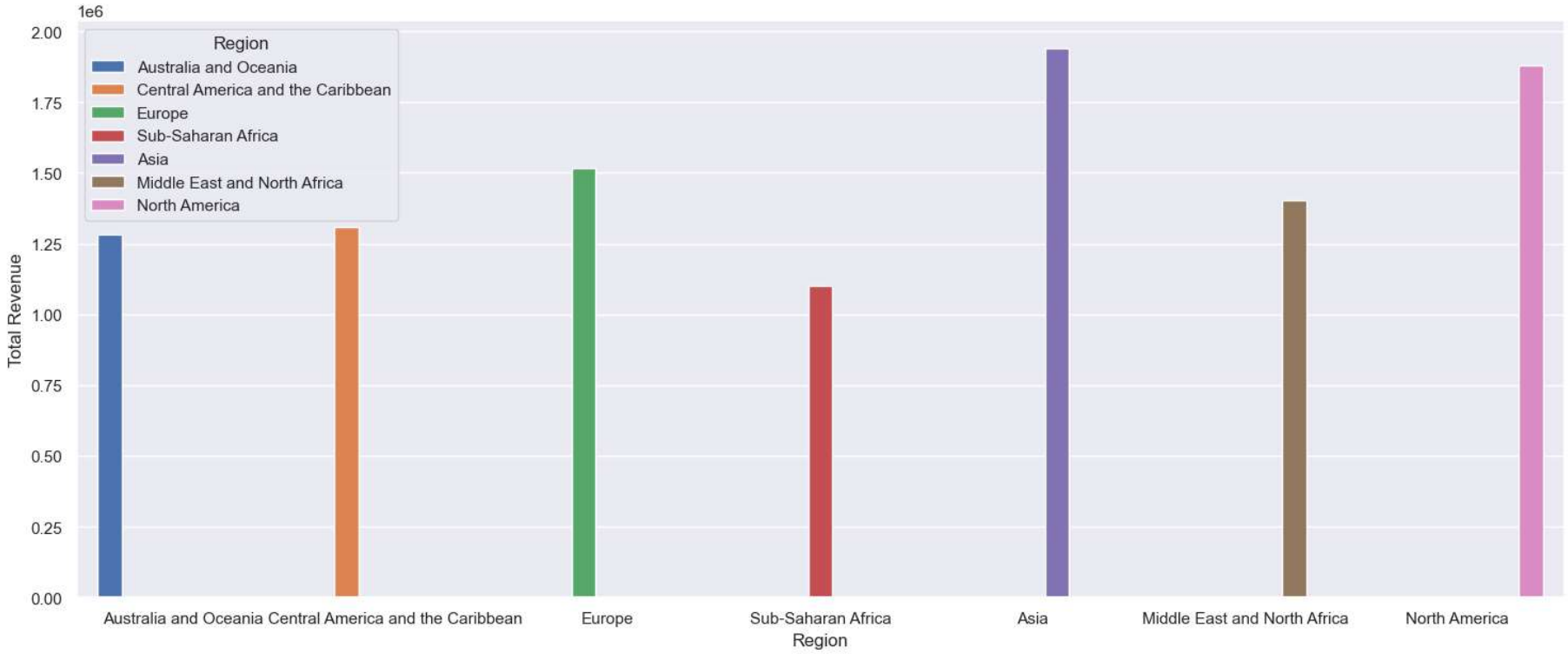
	index	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	To Pr
0	0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010	9925	255.28	159.42	2533654.00	1582243.50	951410
1	1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012	2804	205.70	117.11	576782.80	328376.44	248406
2	2	Europe	Russia	Office Supplies	Offline	L	5/2/2014	341417157	5/8/2014	1779	651.21	524.96	1158502.59	933903.84	224598
3	3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	7/5/2014	8102	9.33	6.92	75591.66	56065.84	19525
4	4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	115456712	2/6/2013	5062	651.21	524.96	3296425.02	2657347.52	639077
5	5	Australia and Oceania	Solomon Islands	Baby Food	Online	C	2/4/2015	547995746	2/21/2015	2974	255.28	159.42	759202.72	474115.08	285087
6	6	Sub-Saharan Africa	Angola	Household	Offline	M	4/23/2011	135425221	4/27/2011	4187	668.27	502.54	2798046.49	2104134.98	693911

In [108]:
```
## Here We present Region wise sale data
plt.figure(figsize=(12,4))
plt.pie('Total Revenue',labels='Region',data =top10,
        startangle=55,autopct='%1.2f%%')
plt.title('** Region Wise Revenue **')
plt.show()
```



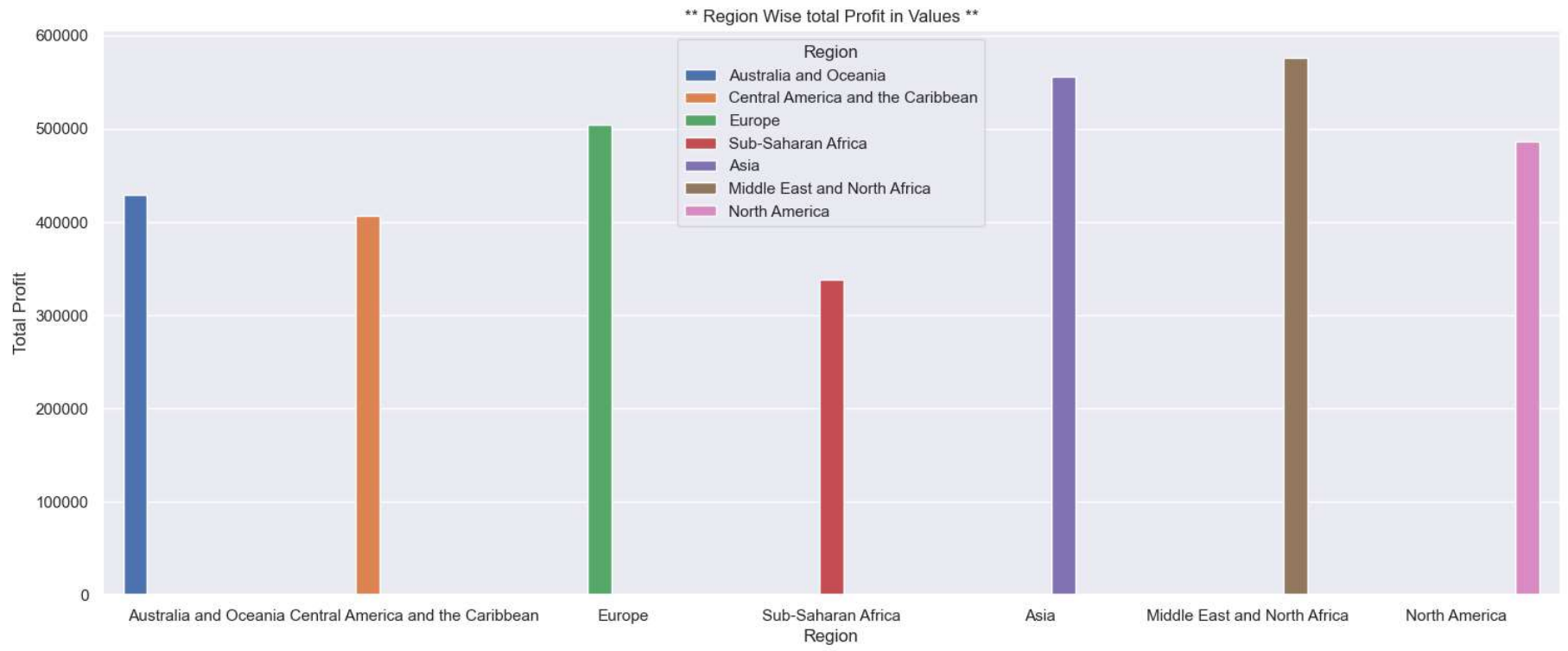
Here we show Bar plot regoin wise Revenue

In [82]:
```
plt.figure(figsize=(18,7))
sns.barplot(x='Region',y='Total Revenue',data=df1,hue='Region',ci=0,n_boot=1000,saturation=10)
plt.xlabel('Region')
plt.ylabel('Total Revenue')
plt.show()
```



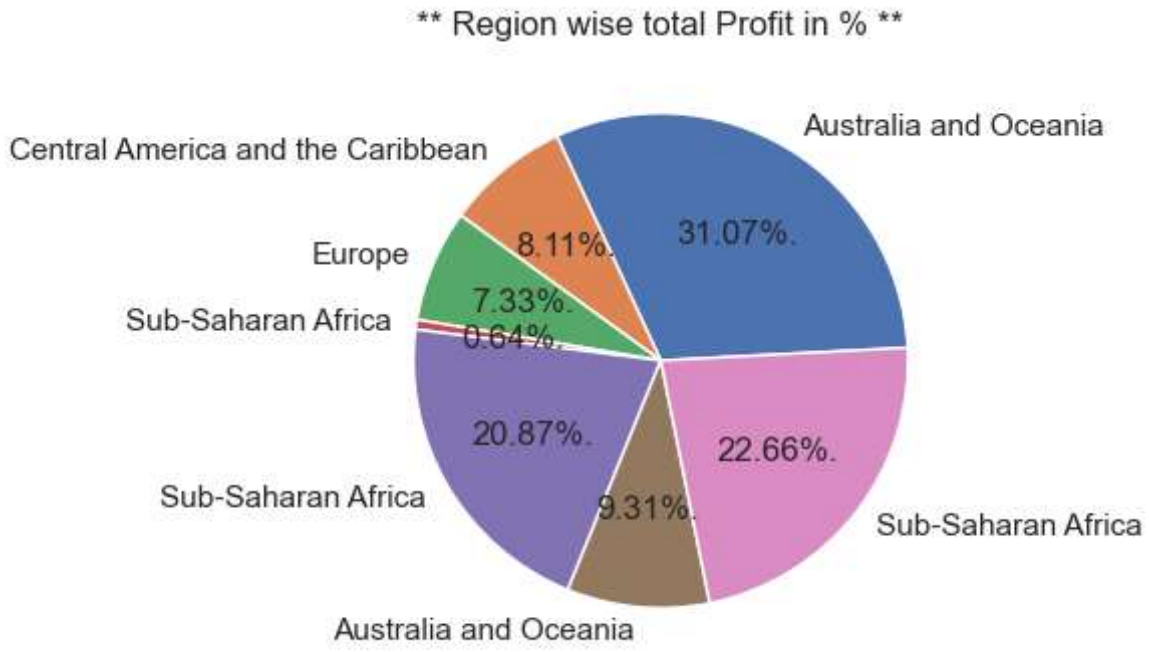
Here We show Region wise Profit

In [114]:
```
plt.figure(figsize=(18,7))
sns.barplot(x='Region',y='Total Profit',data=df1,hue='Region',ci=0,n_boot=1000,saturation=10)
plt.xlabel('Region')
plt.ylabel('Total Profit')
plt.title('** Region Wise total Profit in Values **')
plt.show()
```



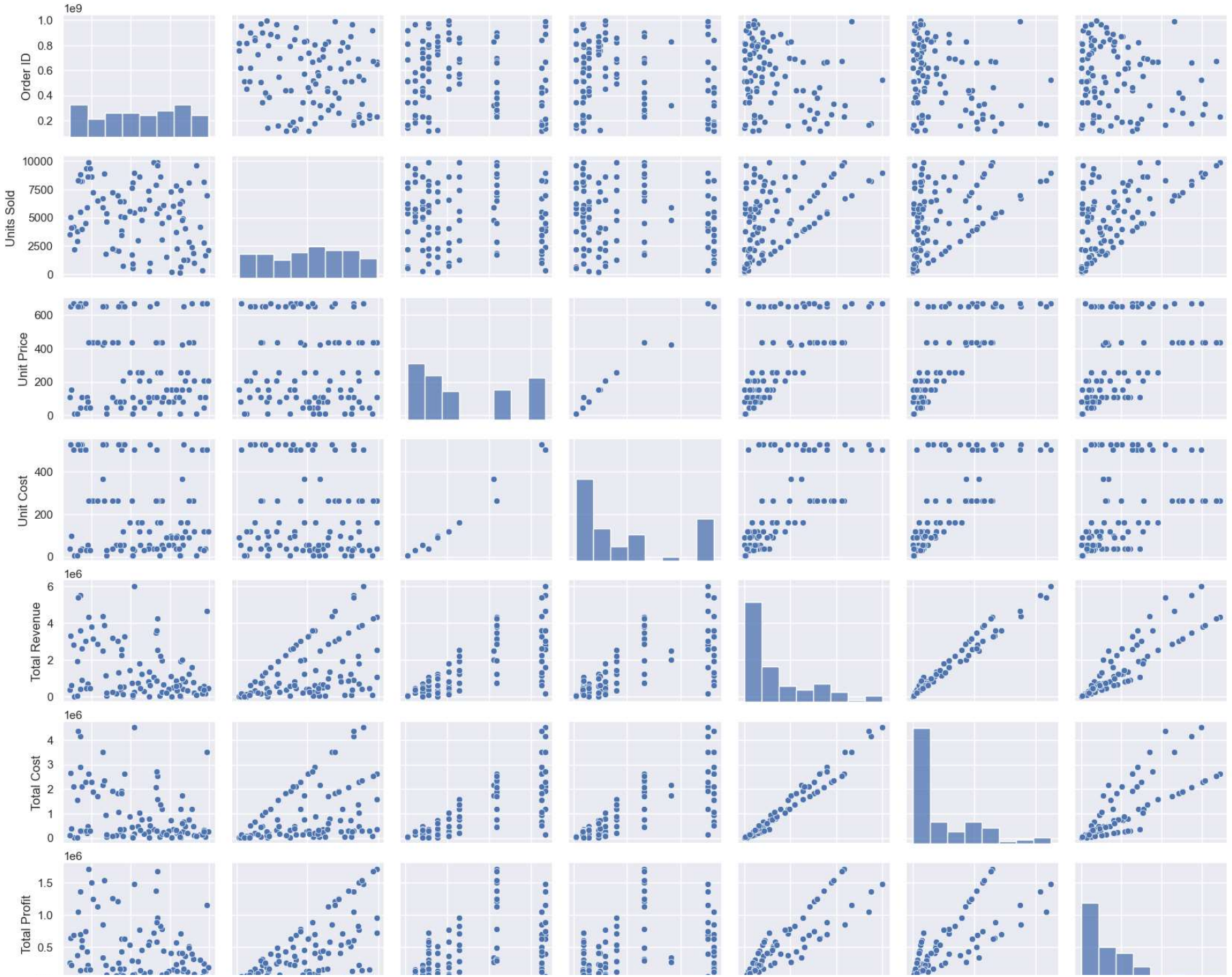
In [94]:
```
# Here I got Top5 Regoin Vs Total Profit
top5 = df1.groupby('Region').sum().sort_values('Total Profit', ascending = False)
top5 = df1.reset_index().head(7)
```

In [110]:
```
## Ploting the Pie Chart
plt.figure(figsize=(12,4))
plt.pie('Total Profit',
    labels='Region',
    autopct='%1.2f%%.',data=top5,startangle=3)
plt.title('** Region wise total Profit in % **')
plt.show()
```



In [132]:
```
## Pairplot
sns.pairplot(data=df1,height=2, aspect=1.2)
```

Out[132]:
```
<seaborn.axisgrid.PairGrid at 0x26d6afa4dc0>
```


1e9
Order ID
1.0
0.8
0.6
0.4
0.2
Units Sold
10000
7500
5000
2500
0
Unit Price
600
400
200
0
Unit Cost
400
200
0
1e6
Total Revenue
6
4
2
0
1e6
Total Cost
4
3
2
1
0
1e6
Total Profit
1.5
1.0
0.5

In [145]:
```
plt.scatter(x='Total Cost',
    y='Total Profit',
    alpha=1,
    data=df1)
plt.xlabel('Total Cost')
plt.ylabel('Total Profit')
plt.title('Total Cost VS Total Profit')
plt.show()
```

In [159]:
```
df1.head(2)
```

Out[159]:

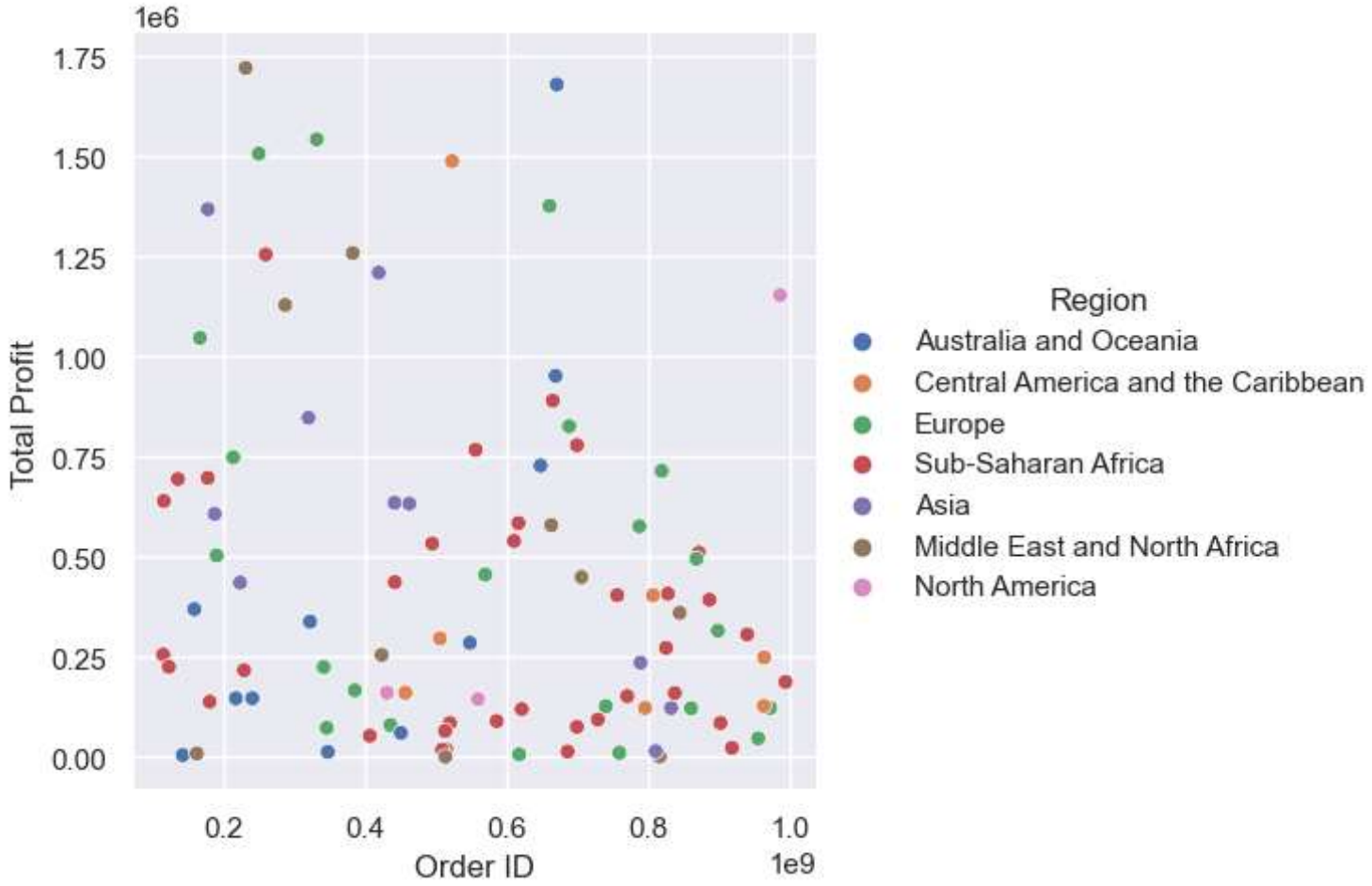
	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010	9925	255.28	159.42	2533654.0	1582243.50	951410.50
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012	2804	205.70	117.11	576782.8	328376.44	248406.36

In [183]:
```
plt.figure(figsize=(10,4))
sns.catplot(x='Sales Channel',
    y='Total Profit',
    hue='Region',
    data=df1,legend='auto',ci=500,
    n_boot=10)
plt.title('** Sales Channel by Total Profit **')
```

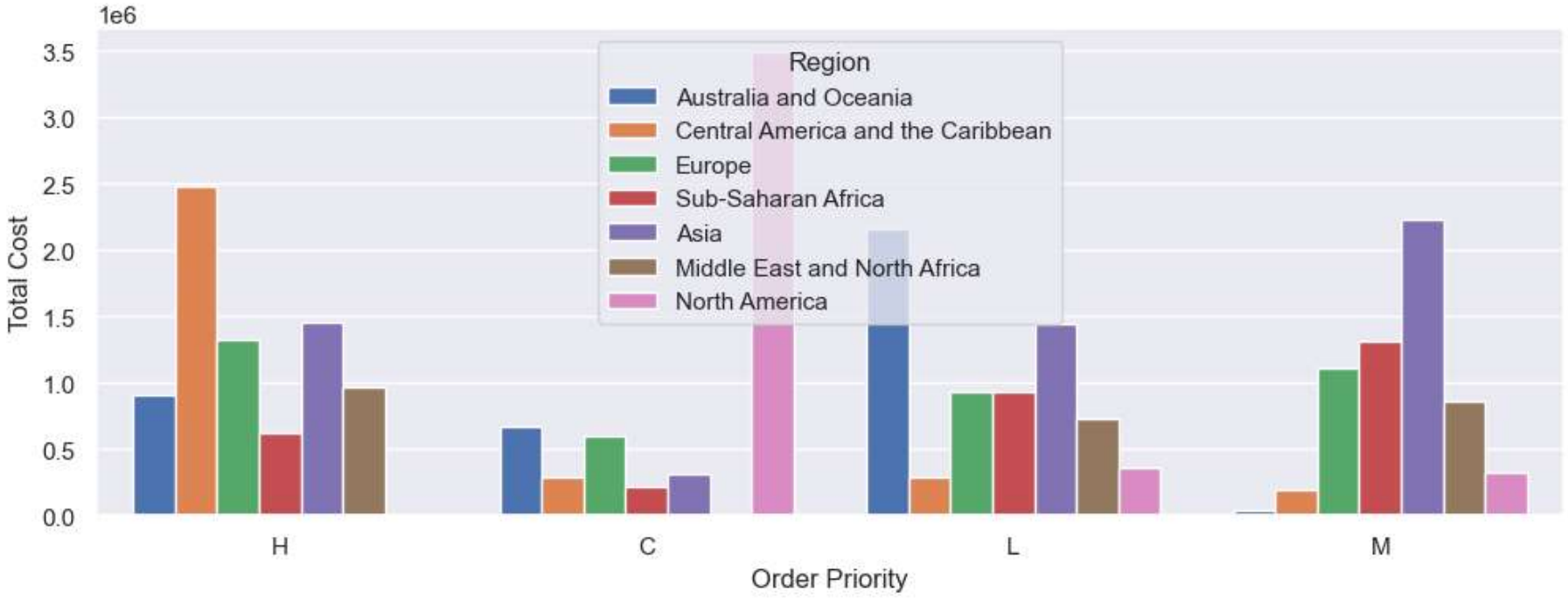
Out[183]:
```
Text(0.5, 1.0, '** Sales Channel by Total Profit **')

<Figure size 1000x400 with 0 Axes>
```

In [158]:
```
sns.relplot(x='Order ID',
    y='Total Profit',
    hue='Region',
    data=df1,legend='auto')
```

Out[158]: <seaborn.axisgrid.FacetGrid at 0x26d8f991eb0>

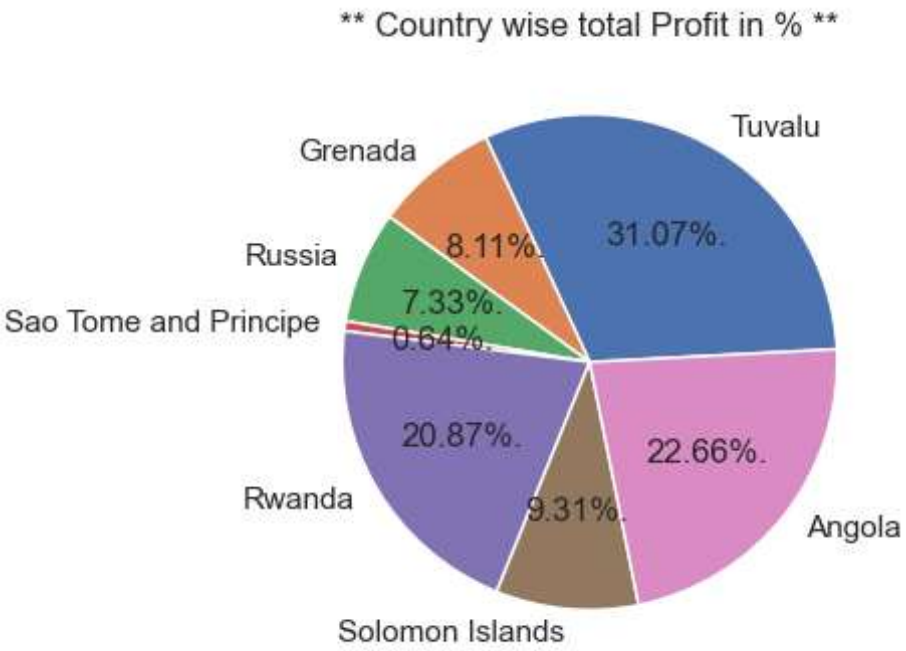


In [189]:
```
plt.figure(figsize=(12,4))
sns.barplot(x='Order Priority',y='Total Cost',data=df1,hue='Region',ci=0,n_boot=1000,saturation=10)
plt.xlabel('Order Priority')
plt.ylabel('Total Cost')
plt.show()
```



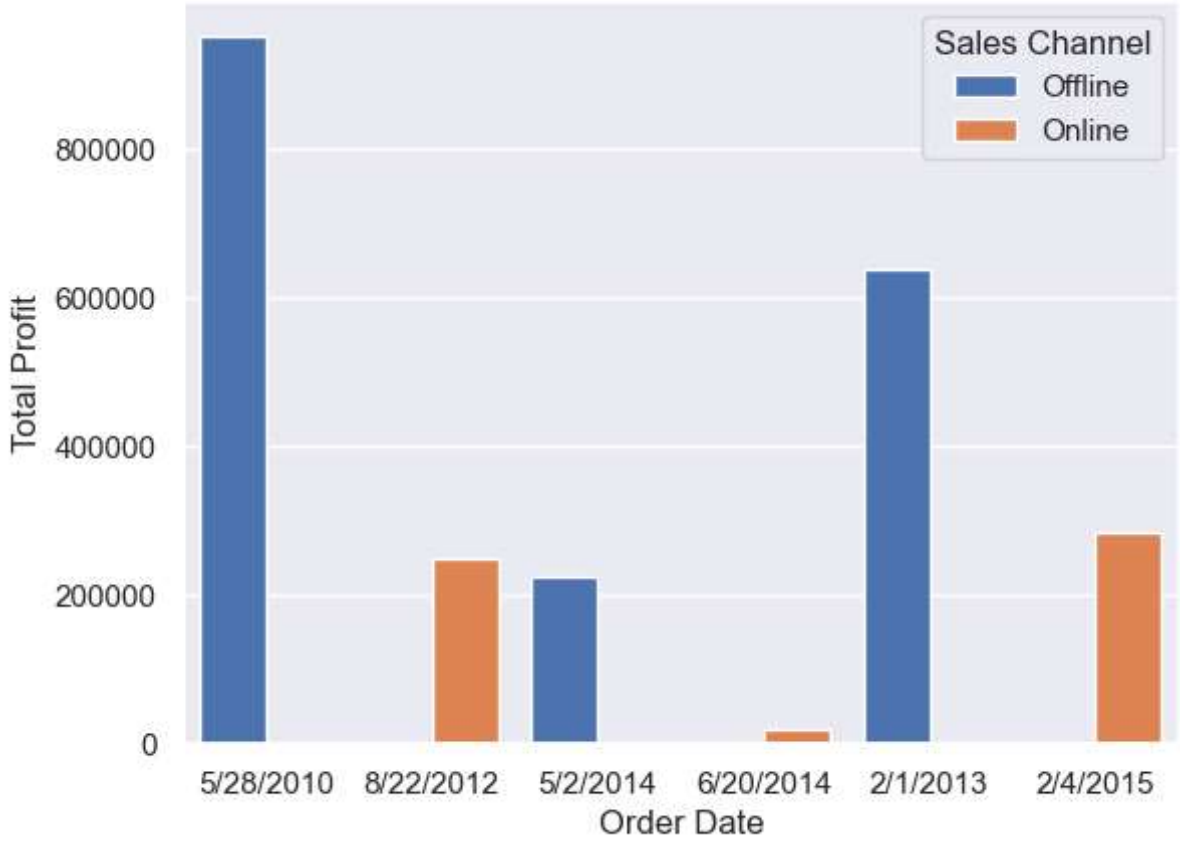
In [190]:
```
top5Country = df1.groupby('Country').sum().sort_values('Total Profit', ascending = False)
top5Country = df1.reset_index().head(7)
```

In [191]:
```
## Ploting the Pie Chart
plt.figure(figsize=(12,4))
plt.pie('Total Profit',
    labels='Country',
    autopct='%1.2f%%.',data=top5Country,startangle=3)
plt.title('** Country wise total Profit in % **')
plt.show()
```



In [203]:
```
sns.barplot(x=df1['Order Date'].head(6),
    y='Total Profit',
    hue='Sales Channel',
    data=df1,saturation=10)
```

Out[203]: <AxesSubplot:xlabel='Order Date', ylabel='Total Profit'>



In [209]:
```
sns.pointplot(x=df1['Order Date'].head(),
    y='Total Profit',
    data=df1,ci=95,
    n_boot=1000,)
```

Out[209]: <AxesSubplot:xlabel='Order Date', ylabel='Total Profit'>

In [ ]:

In [ ]:

In [ ]: