Sales Data Analysis

In [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing
import matplotlib.pyplot as plt
import plotly.express as px
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-1-e17969570938> in <module>
      2 import pandas as pd # data processing
      3 import matplotlib.pyplot as plt
----> 4 import plotly.express as px

ModuleNotFoundError: No module named 'plotly'
```

In [2]:

```
df = pd.read_csv(r'C:\Users\User\Desktop\sales.csv')
df.head()
```

Out[2]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERD
0	10107	30	95.70	2	2871.00	2/24/2 (
1	10121	34	81.35	5	2765.90	05-07-2 0(
2	10134	41	94.74	2	3884.34	07-01-2 0(
3	10145	45	83.26	6	3746.70	8/25/2 (
4	10159	49	100.00	14	5205.27	10-10-2 0(

5 rows × 25 columns

In [3]:

```
# displaying the datatypes of the columns
df.dtypes
```

Out[3]:

```
ORDERNUMBER           int64
QUANTITYORDERED       int64
PRICEEACH           float64
ORDERLINENUMBER       int64
SALES               float64
ORDERDATE            object
STATUS               object
QTR_ID                int64
MONTH_ID              int64
YEAR_ID               int64
PRODUCTLINE          object
MSRP                  int64
PRODUCTCODE          object
CUSTOMERNAME         object
PHONE                object
ADDRESSLINE1         object
ADDRESSLINE2         object
CITY                 object
STATE                object
POSTALCODE           object
COUNTRY              object
TERRITORY            object
CONTACTLASTNAME      object
CONTACTFIRSTNAME     object
DEALSIZE             object
dtype: object
```

In [4]:

```
df.describe()
```

Out[4]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES
count	29.000000	29.000000	29.000000	29.000000	29.000000
mean	10242.379310	35.758621	93.372414	5.551724	4036.199655
std	98.165463	10.548037	13.672313	4.484785	1781.756937
min	10103.000000	20.000000	34.910000	1.000000	733.110000
25%	10159.000000	28.000000	94.740000	2.000000	2765.900000
50%	10237.000000	36.000000	100.000000	5.000000	3884.340000
75%	10318.000000	42.000000	100.000000	9.000000	4708.440000
max	10417.000000	66.000000	100.000000	14.000000	7737.930000

In [5]:

```
import numpy as np
import matplotlib.pyplot as plt
df.describe(include=[np.object])
```

```
<ipython-input-5-d37ed8e8c131>:3: DeprecationWarning: `np.object` is a depre
cated alias for the builtin `object`. To silence this warning, use `object`
by itself. Doing this will not modify any behavior and is safe.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/d
evdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdocs/rel
ease/1.20.0-notes.html#deprecations)
  df.describe(include=[np.object])
```

Out[5]:

	ORDERDATE	STATUS	PRODUCTLINE	PRODUCTCODE	CUSTOMERNAME	PHONE	AD
count	29	29	29	29	29	29	
unique	29	2	2	2	26	26	
top	11-02-2004 00:00	Shipped	Motorcycles	S10_1678	La Rochelle Gifts	07-98 9555	89
freq	1	28	26	26	2	2	

In [6]:

```
df.columns
```

Out[6]:

```
Index(['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'ORDERLINENUMBER',
       'SALES', 'ORDERDATE', 'STATUS', 'QTR_ID', 'MONTH_ID', 'YEAR_ID',
       'PRODUCTLINE', 'MSRP', 'PRODUCTCODE', 'CUSTOMERNAME', 'PHONE',
       'ADDRESSLINE1', 'ADDRESSLINE2', 'CITY', 'STATE', 'POSTALCODE',
       'COUNTRY', 'TERRITORY', 'CONTACTLASTNAME', 'CONTACTFIRSTNAME',
       'DEALSIZE'],
      dtype='object')
```

In [7]:

```
df.head()
```

Out[7]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERD
0	10107	30	95.70	2	2871.00	2/24/2 (
1	10121	34	81.35	5	2765.90	05-07-2 0(
2	10134	41	94.74	2	3884.34	07-01-2 0(
3	10145	45	83.26	6	3746.70	8/25/2 (
4	10159	49	100.00	14	5205.27	10-10-2 0(

5 rows × 25 columns

GENERATING DATA ANALYSIS

1.CHECKING ORDER DETAILS

In [24]:

```
len(df['ORDERNUMBER'].unique().tolist())
```

Out[24]:

```
29
```

Checking of order details with QTR_ID number.

In [26]:

```
SALES_data = df[df['QTR_ID']==3]
SALES_data.head()
```

Out[26]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERD
2	10134	41	94.74	2	3884.34	07-01- (
3	10145	45	83.26	6	3746.70	8/25/
14	10275	45	92.83	1	4177.35	7/23/
15	10285	36	100.00	6	4099.68	8/27/
16	10299	23	100.00	9	2597.39	9/30/

5 rows × 25 columns

In [28]:

```
Order_no = SALES_data.groupby(['ORDERNUMBER'])['SALES'].sum()
Order_no
```

Out[28]:

```
ORDERNUMBER
10134    3884.34
10145    3746.70
10275    4177.35
10285    4099.68
10299    2597.39
Name: SALES, dtype: float64
```

In [29]:

```
Order_no = Order_no .sort_values(ascending=False)
Order_no  = Order_no .head(20)
Order_no
```

Out[29]:

```
ORDERNUMBER
10275    4177.35
10285    4099.68
10134    3884.34
10145    3746.70
10299    2597.39
Name: SALES, dtype: float64
```

Visualization of sales using charts

In [17]:

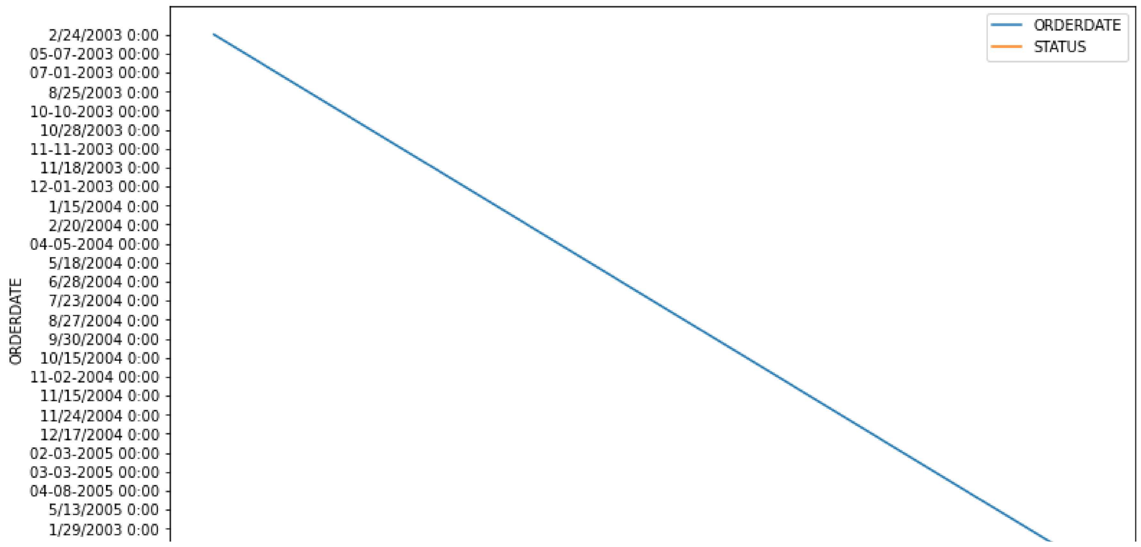
```
import numpy as np # linear algebra
import pandas as pd # data processing
import matplotlib.pyplot as plt
import seaborn as sns
```

In [18]:

```
plt.figure(figsize=(12,8))
sns.lineplot(data=df['ORDERDATE'],linewidth=1.5,label='ORDERDATE')
sns.lineplot(data=df['STATUS'], linewidth = 1.5, label ='STATUS')
```

Out[18]:

```
<AxesSubplot:ylabel='ORDERDATE'>
```

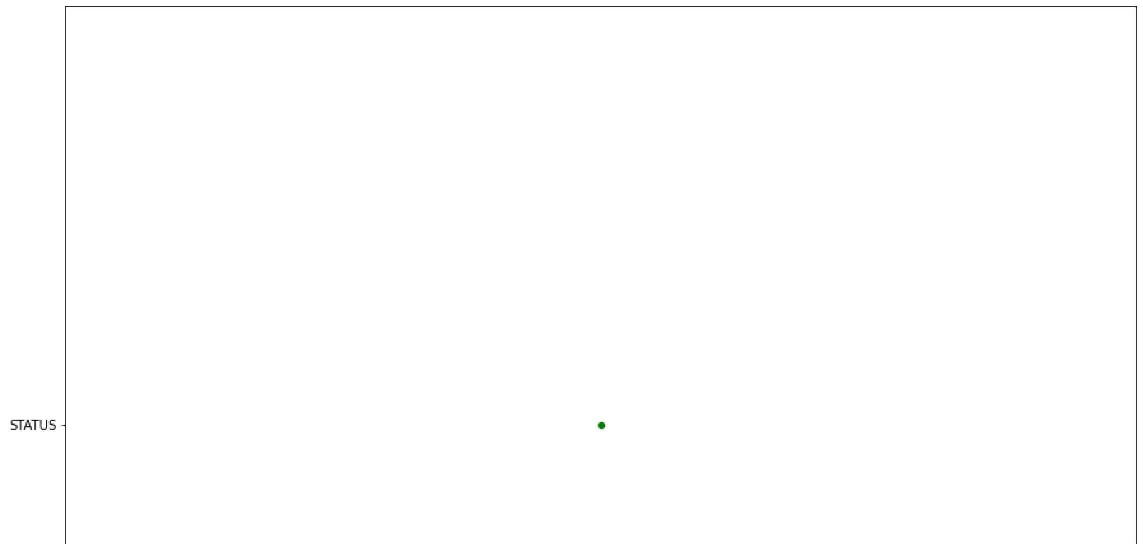


In [20]:

```
plt.figure(figsize = (15,12))
sns.scatterplot(x =['QUANTITYORDERED'], y =['STATUS'], color = 'green')
```

Out[20]:

```
<AxesSubplot:>
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: