In [1]:
```
import pandas as pd
import numpy as np
import numpy as np
import scipy.stats as sm
import pylab as py
import plotly.graph_objs as go
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import pandas as pd
df = pd.read_csv(r"Sample.csv",encoding= 'unicode_escape')
```

In [2]:
```
df.head(5)
```

Out[2]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Discount	Profit
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.9600	2	0.00	41.9136
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs,...	731.9400	3	0.00	219.5820
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters b...	14.6200	2	0.00	6.8714
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.0310
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold 'N Roll Cart System	22.3680	2	0.20	2.5164

5 rows × 21 columns

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
Row ID          0
Order ID        0
Order Date      0
Ship Date       0
Ship Mode       0
Customer ID     0
Customer Name   0
Segment         0
Country         0
City            0
State           0
Postal Code     0
Region          0
Product ID      0
Category        0
Sub-Category    0
Product Name    0
Sales           0
Quantity        0
Discount        0
Profit          0
dtype: int64
```

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
```

```
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   object
 3   Ship Date      9994 non-null   object
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

In [5]:
```
df.duplicated().sum()
```

Out[5]:
```
0
```

In [6]:
```
df.describe()
```

Out[6]:

	Row ID	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	55190.379428	229.858001	3.789574	0.156203	28.656896
std	2885.163629	32063.693350	623.245101	2.225110	0.206452	234.260108
min	1.000000	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	99301.000000	22638.480000	14.000000	0.800000	8399.976000

In [7]:
```
df.nunique()
```

Out[7]:
```
Row ID          9994
Order ID        5009
Order Date      1237
Ship Date       1334
Ship Mode          4
Customer ID      793
Customer Name    793
Segment            3
Country            1
City             531
State             49
Postal Code      631
Region             4
Product ID      1862
Category           3
```

```
Sub-Category       17
Product Name     1850
Sales            5825
Quantity           14
Discount           12
Profit           7287
dtype: int64
```

In [8]:
```
# df = df.drop('Row ID', axis=1)
# df = df.drop('Country',axis = 1)
# df = df.drop('Customer Name',axis = 1)
```

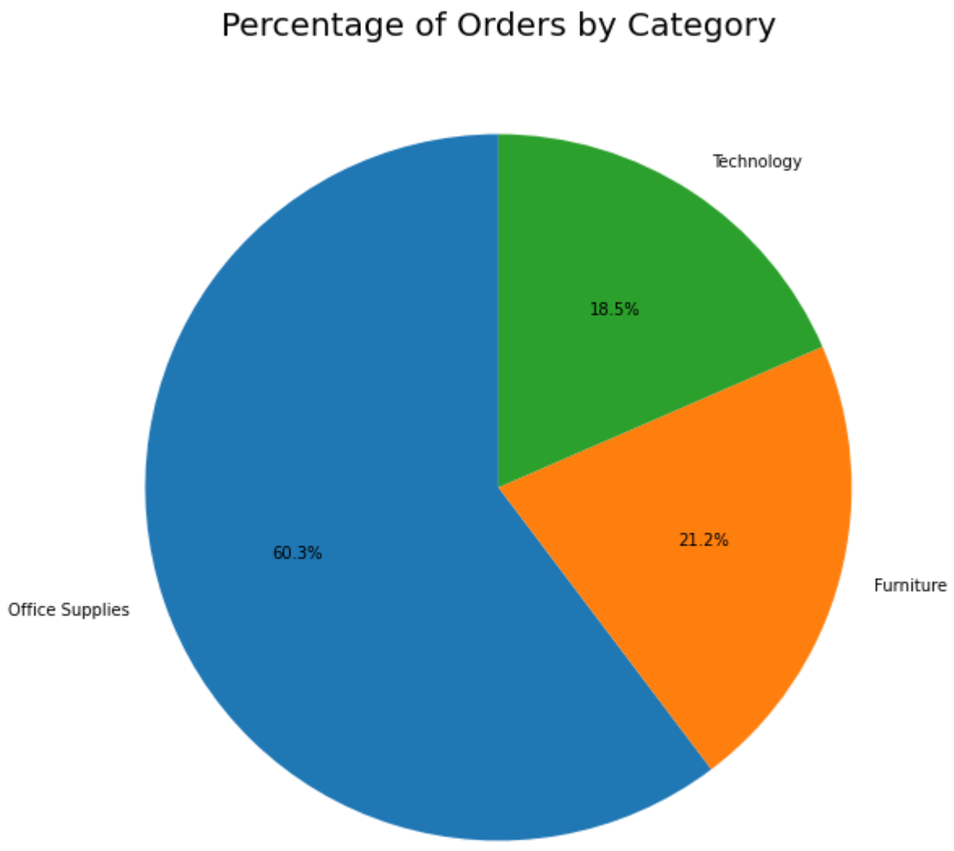
In [9]:
```
df['Order Date'] = pd.to_datetime(df['Order Date'], format='%m/%d/%Y')
df['Ship Date'] = pd.to_datetime(df['Ship Date'], format='%m/%d/%Y')
df
```

Out[9]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Discount	Profit
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.9600	2	0.00	41.9136
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs,...	731.9400	3	0.00	219.5820
2	3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters b...	14.6200	2	0.00	6.8714
3	4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.0310
4	5	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold 'N Roll Cart System	22.3680	2	0.20	2.5164
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9989	9990	CA-2014-110422	2014-01-21	2014-01-23	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United States	Miami	...	33180	South	FUR-FU-10001889	Furniture	Furnishings	Ultra Door Pull Handle	25.2480	3	0.20	4.1028
9990	9991	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627	West	FUR-FU-10000747	Furniture	Furnishings	Tenex B1-RE Series Chair Mats for Low Pile Car...	91.9600	2	0.00	15.6332
9991	9992	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627	West	TEC-PH-10003645	Technology	Phones	Aastra 57i VoIP phone	258.5760	2	0.20	19.3932
9992	9993	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627	West	OFF-PA-10004041	Office Supplies	Paper	It's Hot Message Books with Stickers, 2 3/4" x 5"	29.6000	4	0.00	13.3200
9993	9994	CA-2017-119914	2017-05-04	2017-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westminster	...	92683	West	OFF-AP-10002684	Office Supplies	Appliances	Acco 7-Outlet Masterpiece Power Center, Wihtou...	243.1600	2	0.00	72.9480

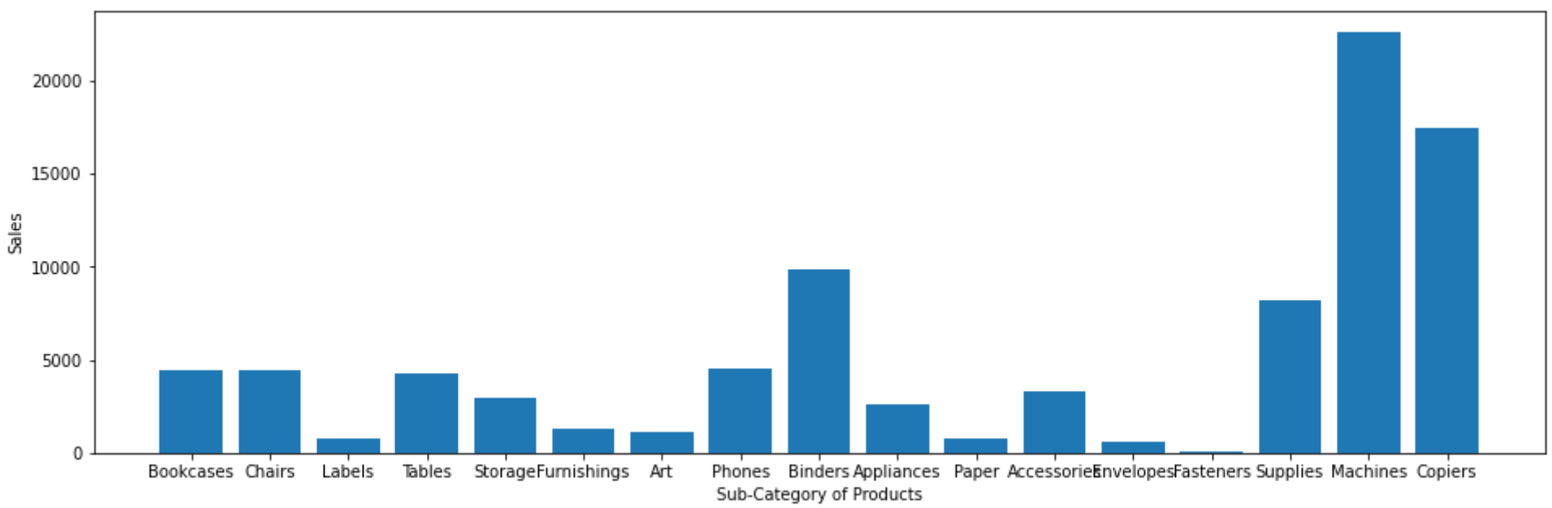
9994 rows × 21 columns

In [10]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(10,10))
df['Category'].value_counts().plot.pie(autopct='%1.1f%%', startangle=90)
plt.title("Percentage of Orders by Category", size=20)
plt.ylabel(None)
plt.show()
```



In [11]:

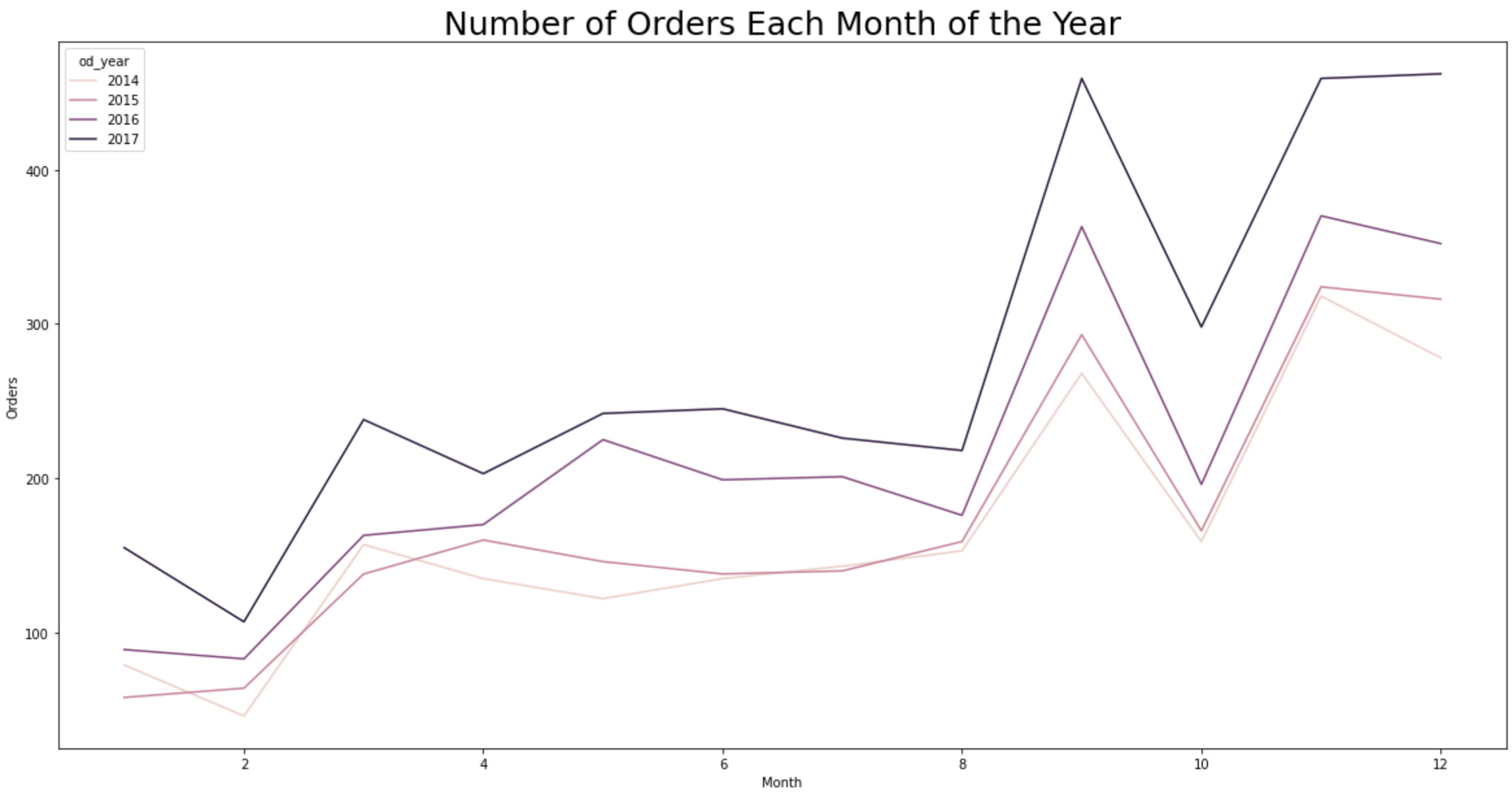
```
plt.figure(figsize = (16,5))
plt.bar(df["Sub-Category"],df["Sales"])
plt.xlabel("Sub-Category of Products")
plt.ylabel("Sales")
plt.show()
```



In [12]:

```
import seaborn as sns
#Number of orders on each month of the year

df['od_year'] = pd.DatetimeIndex(df['Order Date']).year
```

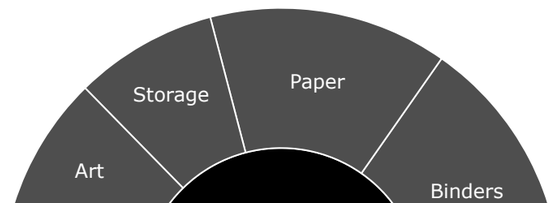
```
df['od_month'] = pd.DatetimeIndex(df['Order Date']).month

years = df.groupby(['od_year', 'od_month']).count().reset_index()

plt.figure(figsize=(20,10))
sns.lineplot(data = years, x = "od_month", y="Order ID", hue = 'od_year')
plt.title("Number of Orders Each Month of the Year", size=25)
plt.xlabel('Month')
plt.ylabel('Orders')
plt.show()
```

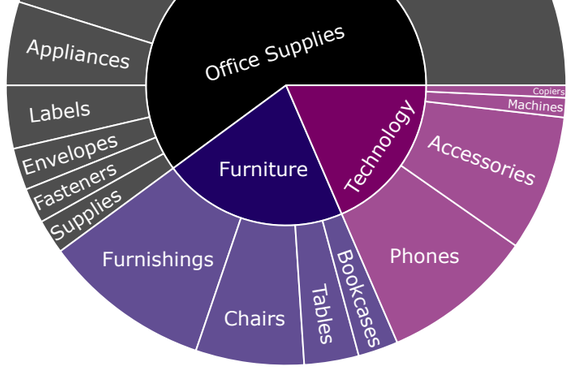


In [13]:

```
import plotly.express as px
fig = px.sunburst(data_frame = df[['Category','Sub-Category','Row ID']].groupby(['Category','Sub-Category']).sum().reset_index(), path=['Category', 'Sub-Category'], values='Row ID', title='Frequency of cate
fig.show()
```

Frequency of category occurences.

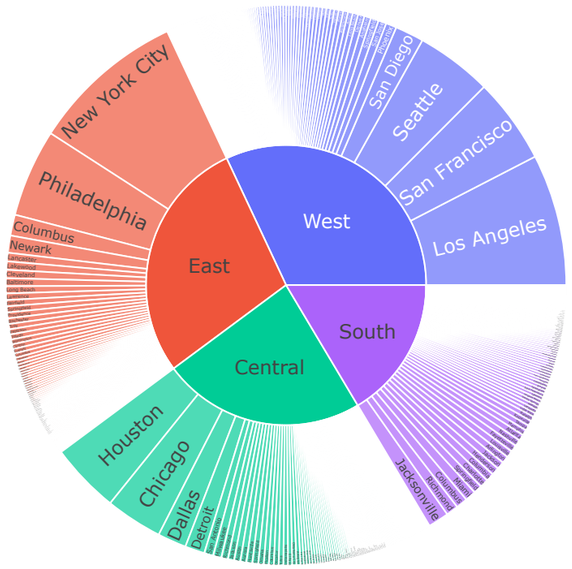




In [14]:

```
fig = px.sunburst(data_frame = df[['Region','City','Row ID']].groupby(['Region','City']).sum().reset_index(), path=['Region', 'City'], values='Row ID', title='Frequency of regions and cities occurences.')
fig.show()
```

Frequency of regions and cities occurences.

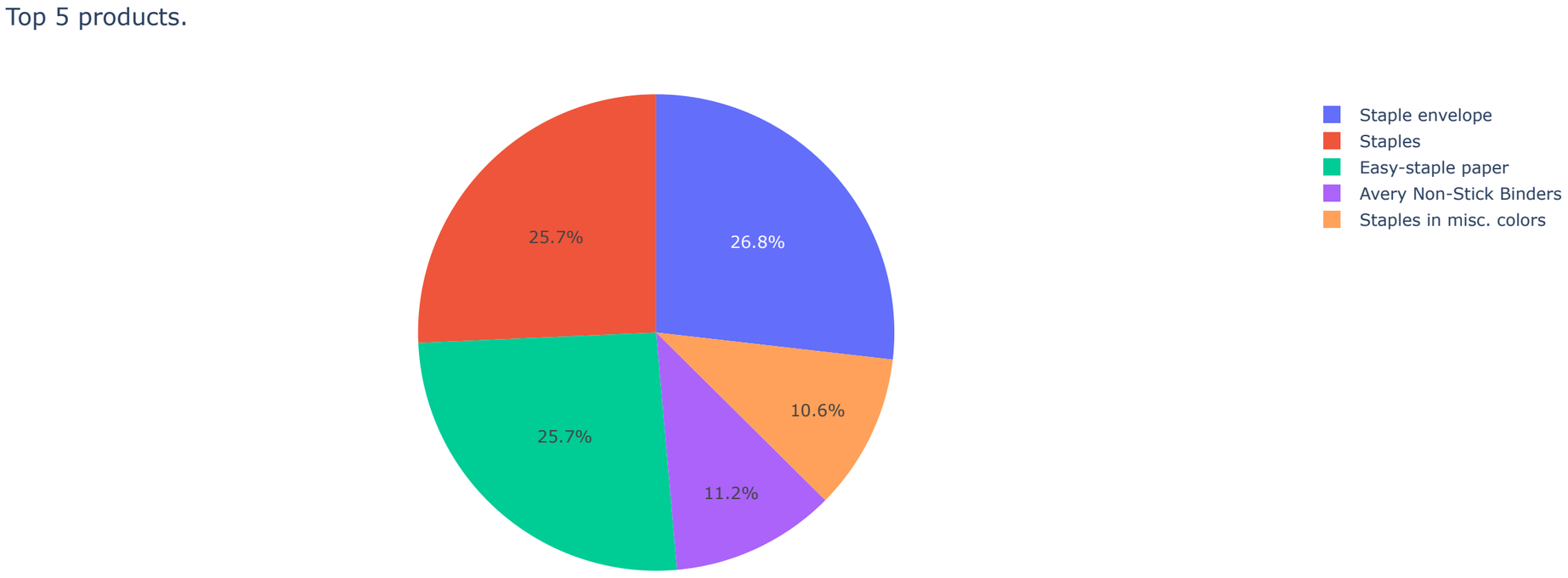


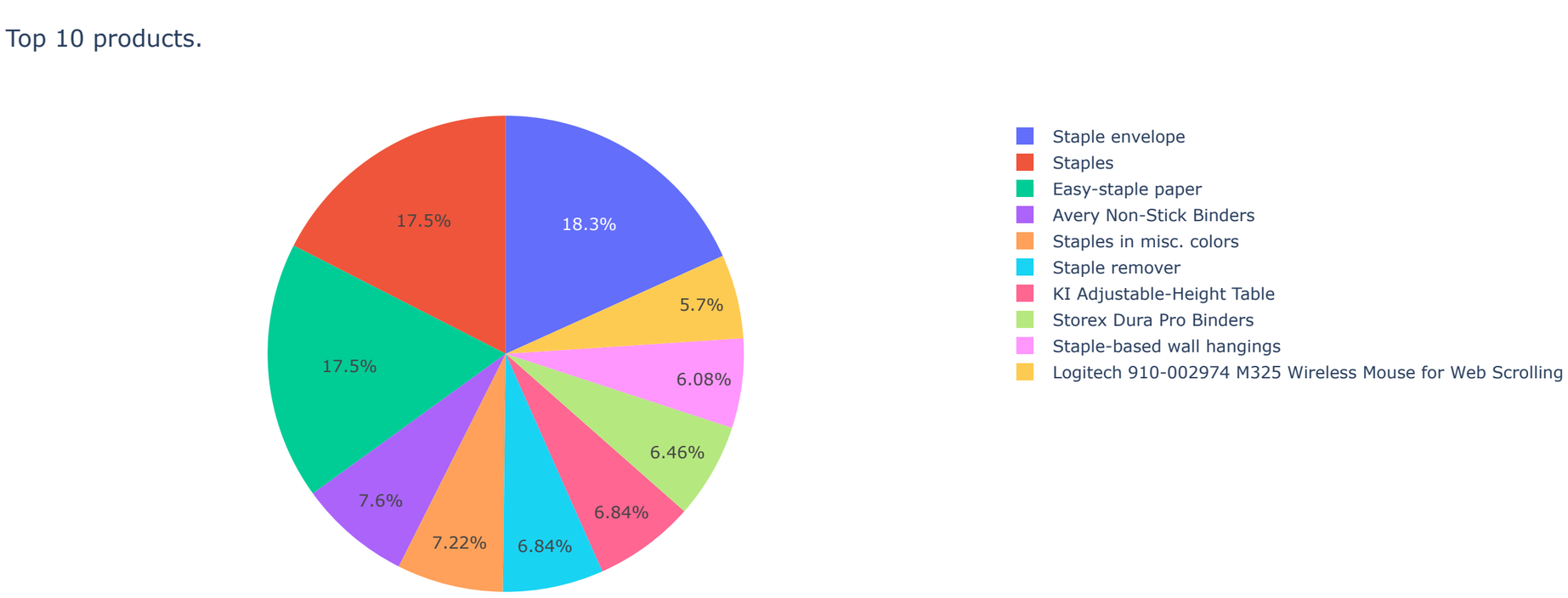
In [15]:

```
print("Total count of products: ",len(list(df['Product Name'].unique())))
top_prod = df.groupby('Product Name').size().reset_index().rename(columns={0: 'total'}).sort_values('total', ascending=False).head(5)
fig = px.pie(top_prod, values='total', names='Product Name', title='Top 5 products.')
fig.show()
top_prod = df.groupby('Product Name').size().reset_index().rename(columns={0: 'total'}).sort_values('total', ascending=False).head(10)
fig = px.pie(top_prod, values='total', names='Product Name', title='Top 10 products.')
fig.show()
top_prod = df.groupby('Product Name').size().reset_index().rename(columns={0: 'total'}).sort_values('total', ascending=False).head(20)
```

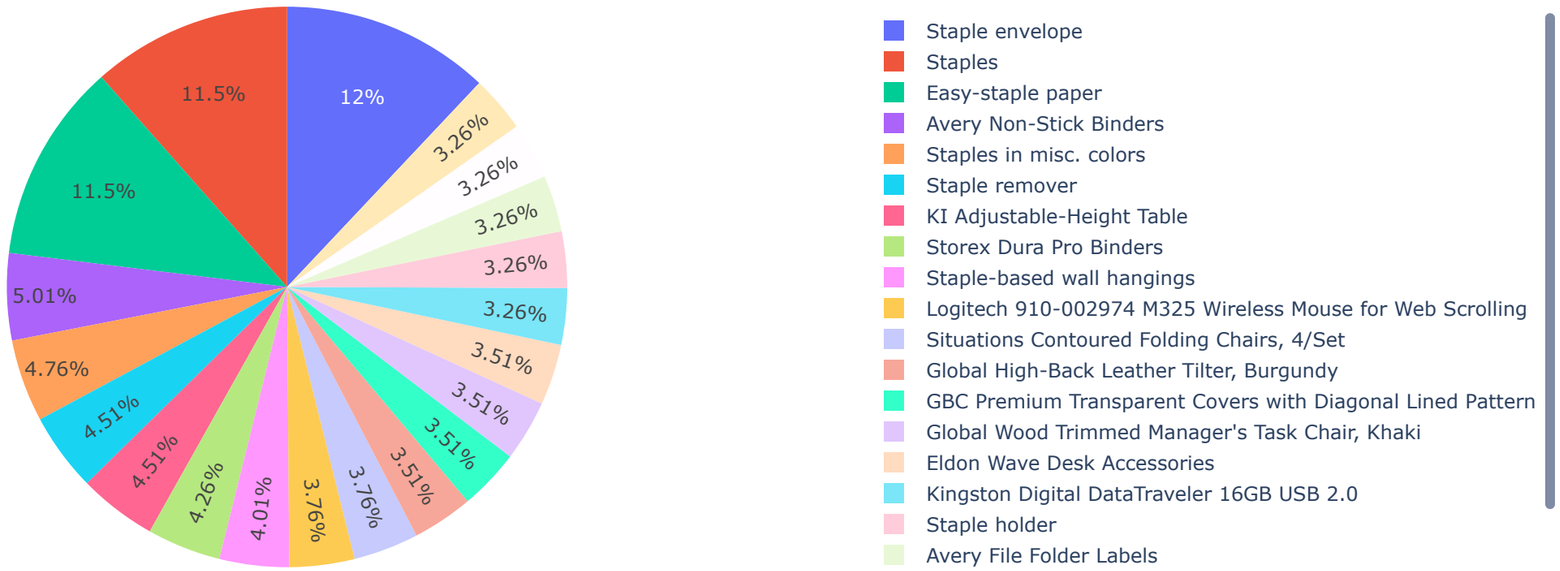
```
fig = px.pie(top_prod, values='total', names='Product Name', title='Top 20 products.')
fig.show()
```

```
Total count of products:  1850
```



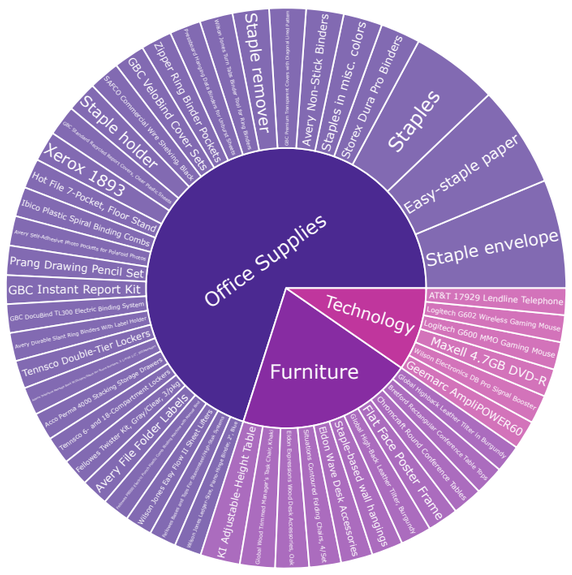


Top 20 products.



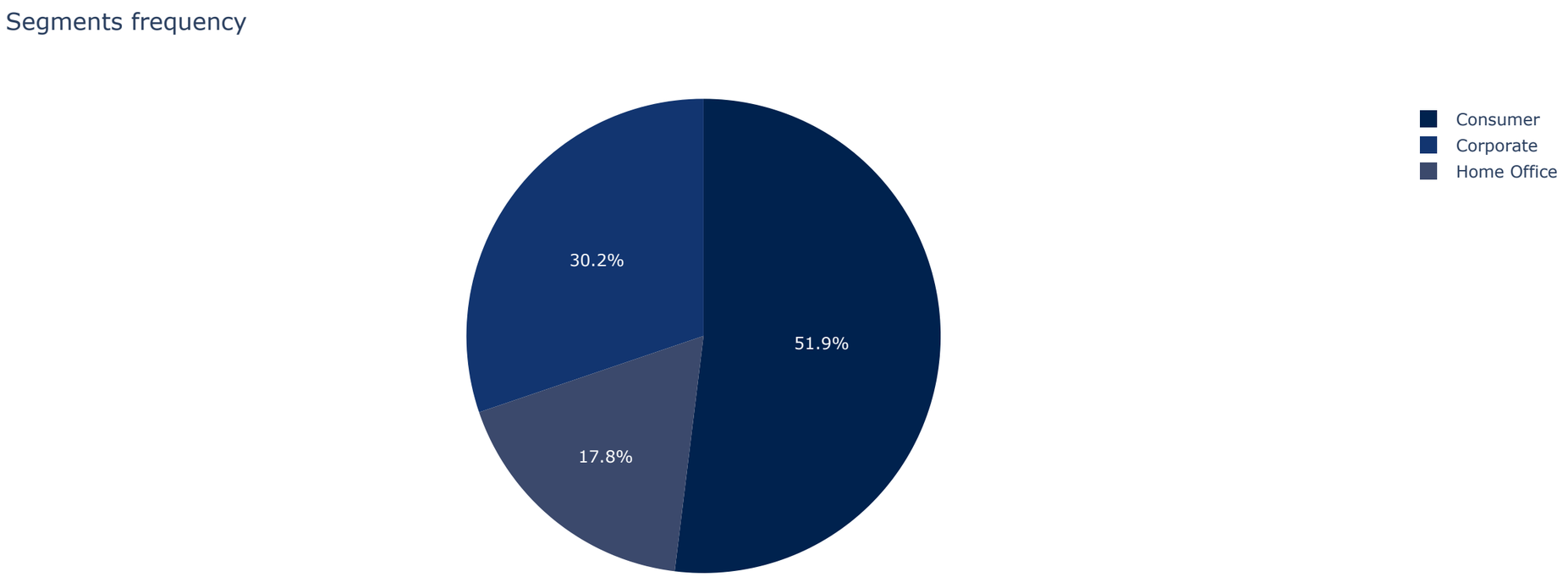
In [16]:
```
fig = px.sunburst(data_frame = df[['Category','Product Name','Row ID']].groupby(['Category','Product Name']).sum().reset_index().sort_values('Row ID',ascending=False).head(50), path=['Category', 'Product Na
fig.show()
```

Top 50 products and their distribution acorss categories



In [17]:

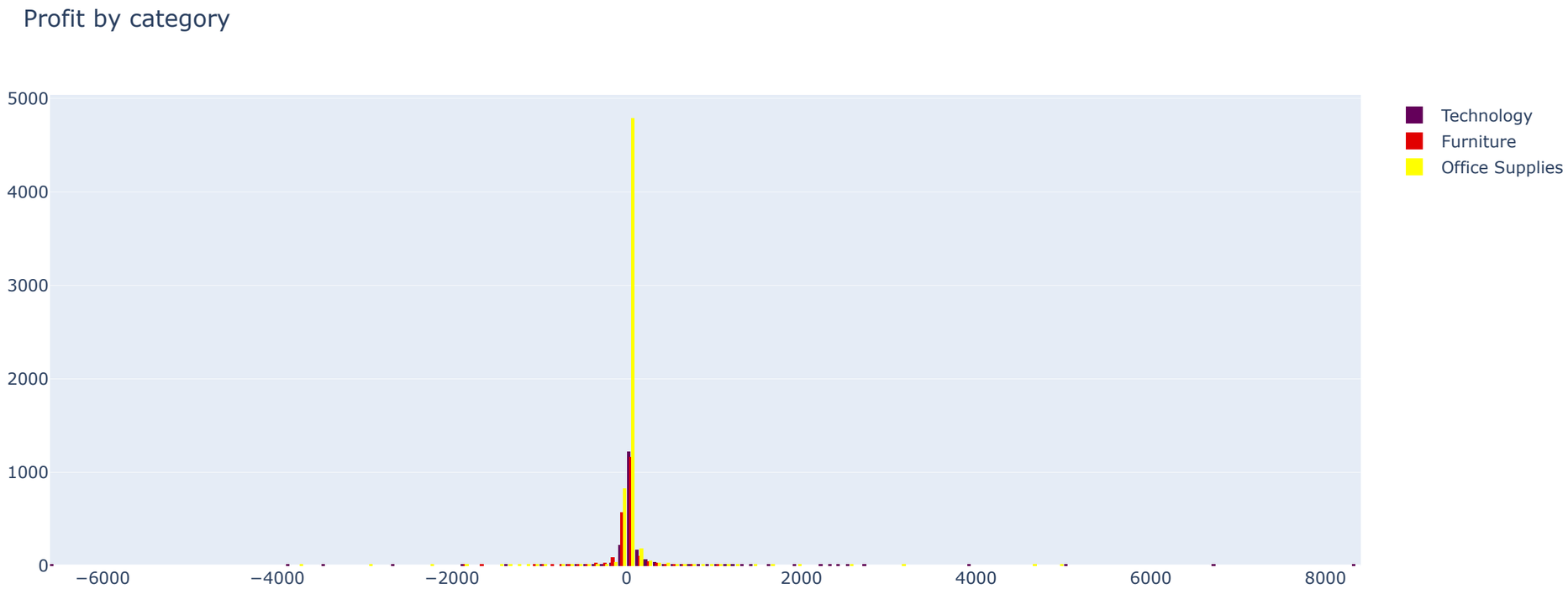
```
segment = df.groupby('Segment').size().reset_index().rename(columns={0: 'total'})
fig = px.pie(segment, values='total', names='Segment', color_discrete_sequence=px.colors.sequential.Cividis, title='Segments frequency')
fig.show()
```



In [18]:

```
import plotly.graph_objs as go
import numpy as np
trace = go.Histogram(x=df[df.Category=='Technology'].Profit, xbins=dict(start=np.min(df.Profit), size=100, end=np.max(df.Profit)),
                     marker=dict(color='rgb(100, 0, 90)'),name='Technology')

trace2  = go.Histogram(x=df[df.Category=='Furniture'].Profit, xbins=dict(start=np.min(df.Profit), size=100, end=np.max(df.Profit)),
                     marker=dict(color='rgb(225, 0, 0)'),name='Furniture')
trace3 = go.Histogram(x=df[df.Category=='Office Supplies'].Profit, xbins=dict(start=np.min(df.Profit), size=100, end=np.max(df.Profit)),
                     marker=dict(color='rgb(255, 255, 0)'),name='Office Supplies')


layout = go.Layout(title="Profit by category")
fig = go.Figure(data=go.Data([trace,trace2,trace3]), layout=layout)
fig.show()
print('Technology products mean profit',df[df.Category=='Technology'].Profit.mean(),'Technology products profit std', df[df.Category=='Technology'].Profit.std(),'Technology products profit median', df[df.Ca
print('Furniture products mean profit',df[df.Category=='Furniture'].Profit.mean(),'Furniture products profit std', df[df.Category=='Furniture'].Profit.std(),'Furniture products profit median', df[df.Categor
print('Office Supplies products mean profit',df[df.Category=='Office Supplies'].Profit.mean(),'Office Supplies products profit std', df[(df.Category=='Office supplies')].dropna().Profit.std(),'Office Suppli
```

```
c:\users\admin\appdata\local\programs\python\python39\lib\site-packages\plotly\graph_objs\_deprecations.py:31: DeprecationWarning:

plotly.graph_objs.Data is deprecated.
Please replace it with a list or tuple of instances of the following types
  - plotly.graph_objs.Scatter
  - plotly.graph_objs.Bar
  - plotly.graph_objs.Area
  - plotly.graph_objs.Histogram
  - etc.
```

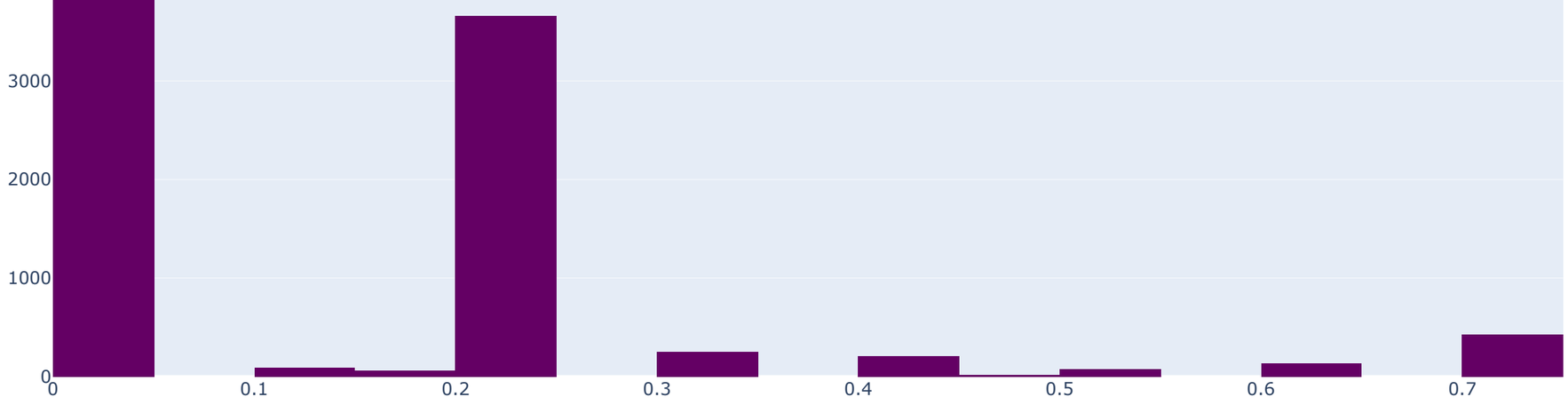



```
Technology products mean profit 78.75200221981592 Technology products profit std 428.8166330051747 Technology products profit median 25.0182
Furniture products mean profit 8.699327109853845 Furniture products profit std 136.04924643905227 Furniture products profit median 7.7748
Office Supplies products mean profit 20.3270495851311 Office Supplies products profit std nan Office Supplies products profit median nan
```

In [19]:

```
trace = go.Histogram(x=df.Discount, xbins=dict(start=np.min(df.Discount), size=0.05, end=np.max(df.Discount)),marker=dict(color='rgb(100, 0, 100)'))

layout = go.Layout(title="Discounts distribution!")

fig = go.Figure(data=go.Data([trace]), layout=layout)
fig.show()
```

```
c:\users\admin\appdata\local\programs\python\python39\lib\site-packages\plotly\graph_objs\_deprecations.py:31: DeprecationWarning:

plotly.graph_objs.Data is deprecated.
Please replace it with a list or tuple of instances of the following types
  - plotly.graph_objs.Scatter
  - plotly.graph_objs.Bar
  - plotly.graph_objs.Area
  - plotly.graph_objs.Histogram
  - etc.
```



In [20]:

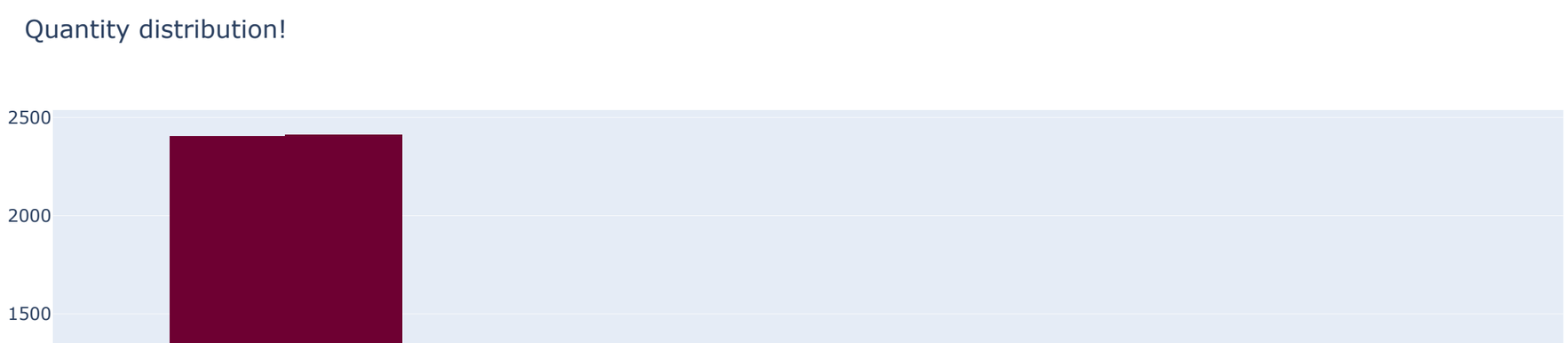
```
profit = df[df['Profit'] > 0]['Customer ID'].count()/df['Customer ID'].count()*100
loss = df[df['Profit'] < 0]['Customer ID'].count()/df['Customer ID'].count()*100
print("Profit % = ",profit)
print("Loss % =",loss)
```

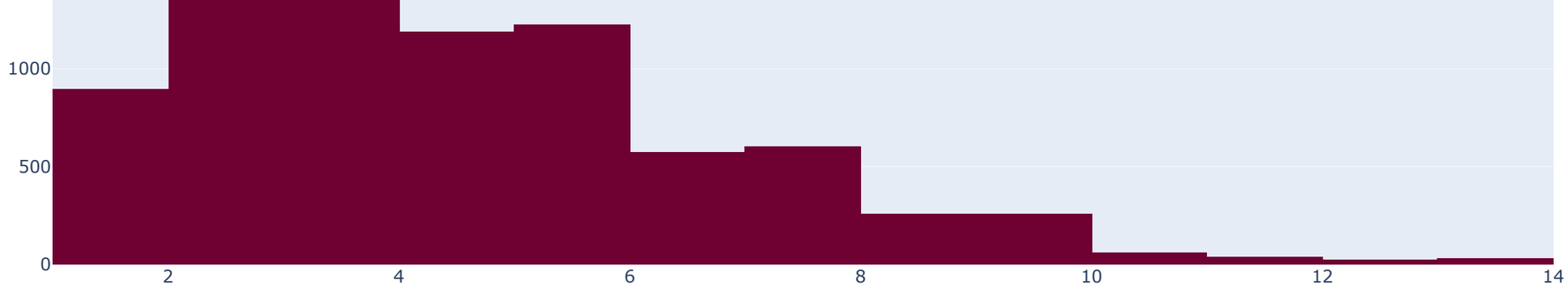
```
Profit % =  80.62837702621573
Loss % = 18.721232739643785
```

In [21]:

```
trace = go.Histogram(x=df.Quantity, xbins=dict(start=np.min(df.Quantity), size=1, end=np.max(df.Quantity)),
                     marker=dict(color='rgb(110, 0, 50)'))

layout = go.Layout(
    title="Quantity distribution!"
)

fig = go.Figure(data=go.Data([trace]), layout=layout)
fig.show()
```

```
c:\users\admin\appdata\local\programs\python\python39\lib\site-packages\plotly\graph_objs\_deprecations.py:31: DeprecationWarning:

plotly.graph_objs.Data is deprecated.
Please replace it with a list or tuple of instances of the following types
  - plotly.graph_objs.Scatter
  - plotly.graph_objs.Bar
  - plotly.graph_objs.Area
  - plotly.graph_objs.Histogram
  - etc.
```





In [ ]:

In [ ]: