In [1]:

```
import os
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Merge 12 months of sales data into a single csv file

In [125]:

```
files =[file for file in os.listdir("F:/EDA_projects/Sales_Analysis/Sal
esAnalysis/Sales_Data")]
for file in files:
    print(file)
```

```
Sales_April_2019.csv
Sales_August_2019.csv
Sales_December_2019.csv
Sales_February_2019.csv
Sales_January_2019.csv
Sales_July_2019.csv
Sales_June_2019.csv
Sales_March_2019.csv
Sales_May_2019.csv
Sales_November_2019.csv
Sales_October_2019.csv
Sales_September_2019.csv
```

In [141]:

```
path = "F:/EDA_projects/Sales_Analysis/SalesAnalysis/Sales_Data"

#blank dataframe
all_data = pd.DataFrame()

for file in files:
    current_df = pd.read_csv(path+"/"+file)
    all_data = pd.concat([all_data, current_df])
```

```
all_data.shape
```

Out[141]:
```
(186850, 6)
```

convert it into dataset

In [ ]:
```
all_data.to_csv('F:/EDA_projects/Sales_Analysis/SalesAnalysis/Sales_Data/all_data.csv',index=False)
```

Data cleaning and formatting

In [142]:
```
all_data.dtypes
```

Out[142]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
dtype: object
```

In [128]:
```
all_data.head()
```

Out[128]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
1	NaN	NaN	NaN	NaN	NaN	NaN
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001

In [129]:
```
all_data.isnull().sum()
```

Out[129]:
```
Order ID            585
Product             585
Quantity Ordered    585
Price Each          585
Order Date          585
Purchase Address    585
dtype: int64
```

In [143]:
```
all_data = all_data.dropna(how='all')
all_data.shape
```

Out[143]:
```
(186305, 6)
```

What is the best month for sale?

In [10]:
```
'04/19/19 08:46'.split('/')[0]
```

Out[10]:
```
'04'
```

In [144]:
```
def month(x):
    return x.split('/')[0]
```

add month col

In [145]:
```
all_data['Month']=all_data['Order Date'].apply(month)
```

In [134]:
```
all_data.dtypes
```

```
Out[134]: Order ID            object
          Product             object
          Quantity Ordered    object
          Price Each          object
          Order Date          object
          Purchase Address    object
          Month               object
          dtype: object
```

```
In [14]: all_data['Month']=all_data['Month'].astype(int)
```

```
----------------------------------------------------------------------
----
ValueError                               Traceback (most recent call l
ast)
<ipython-input-14-188a0bea4a2e> in <module>
----> 1 all_data['Month']=all_data['Month'].astype(int)

~\Anaconda3\lib\site-packages\pandas\core\generic.py in astype(self, dt
ype, copy, errors)
   5535         else:
   5536             # else, only a single dtype is given
-> 5537             new_data = self._mgr.astype(dtype=dtype, copy=copy,
 errors=errors,)
   5538             return self._constructor(new_data).__finalize__(sel
f, method="astype")
   5539

~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in asty
pe(self, dtype, copy, errors)
    565         self, dtype, copy: bool = False, errors: str = "raise"
    566     ) -> "BlockManager":
--> 567         return self.apply("astype", dtype=dtype, copy=copy, err
ors=errors)
    568
    569     def convert(

~\Anaconda3\lib\site-packages\pandas\core\internals\managers.py in appl
y(self, f, align_keys, **kwargs)
```

```
    394                     applied = b.apply(f, **kwargs)
    395                 else:
--> 396                     applied = getattr(b, f)(**kwargs)
    397                 result_blocks = _extend_blocks(applied, result_bloc
ks)
    398 

~\Anaconda3\lib\site-packages\pandas\core\internals\blocks.py in astype
(self, dtype, copy, errors)
    588                 vals1d = values.ravel()
    589                 try:
--> 590                     values = astype_nansafe(vals1d, dtype, copy=Tru
e)
    591                 except (ValueError, TypeError):
    592                     # e.g. astype_nansafe can fail on object-dtype
 of strings

~\Anaconda3\lib\site-packages\pandas\core\dtypes\cast.py in astype_nans
afe(arr, dtype, copy, skipna)
    964         # work around NumPy brokenness, #1987
    965         if np.issubdtype(dtype.type, np.integer):
--> 966             return lib.astype_intsafe(arr.ravel(), dtype).resha
pe(arr.shape)
    967 
    968         # if we have a datetime/timedelta array of objects

pandas\_libs\lib.pyx in pandas._libs.lib.astype_intsafe()

ValueError: invalid literal for int() with base 10: 'Order Date'
```

In [147]:
```
all_data['Month'].unique()
```

Out[147]:
```
array(['04', '05', 'Order Date', '08', '09', '12', '01', '02', '03', '0
7',
       '06', '11', '10'], dtype=object)
```

In [148]:
```
filter=all_data['Month']=='Order Date'
len(all_data[~filter])
```

Out[148]:
```
185950
```

In [149]:
```
all_data=all_data[~filter]
```

In [150]:
```
all_data.shape
```

Out[150]:
```
(185950, 7)
```

In [151]:
```
all_data.head()
```

Out[151]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	04
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	04
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	04

In [152]:
```
all_data['Month']=all_data['Month'].astype(int)
```

In [29]:
```
all_data.dtypes
```

Out[29]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
Month                int32
dtype: object
```

In [153]:

```
all_data['Price Each']=all_data['Price Each'].astype(float)
```

In [154]:

```
all_data['Quantity Ordered']=all_data['Quantity Ordered'].astype(int)
```

In [155]:

```
all_data['sales']=all_data['Quantity Ordered']*all_data['Price Each']
all_data.head(5)
```

Out[155]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	sales
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99

In [156]:

```
all_data.groupby('Month')['sales'].sum()
```

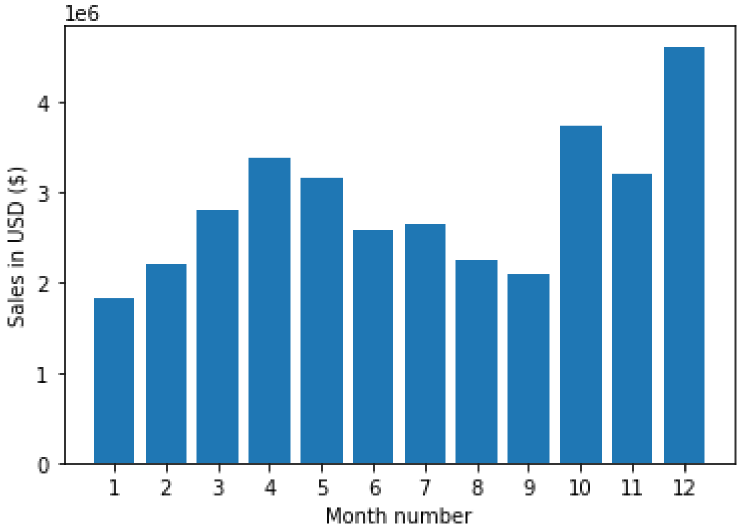
Out[156]:

```
Month
1       1.822257e+06
2       2.202022e+06
3       2.807100e+06
4       3.390670e+06
5       3.152607e+06
6       2.577802e+06
7       2.647776e+06
8       2.244468e+06
9       2.097560e+06
10      3.736727e+06
11      3.199603e+06
```

```
12    4.613443e+06
Name: sales, dtype: float64
```

In [157]:

```
months=range(1,13)
plt.bar(months,all_data.groupby('Month')['sales'].sum())
plt.xticks(months)
plt.ylabel('Sales in USD ($)')
plt.xlabel('Month number')
plt.show()
```



Which city has max order

In [43]:

```
'917 1st St, Dallas, TX 75001'.split(',')[1]
```

Out[43]:

```
' Dallas'
```

In [158]:

```
def city(x):
    return x.split(',')[1]
```

In [159]:

```
all_data['city']=all_data['Purchase Address'].apply(city)
```

In [160]:

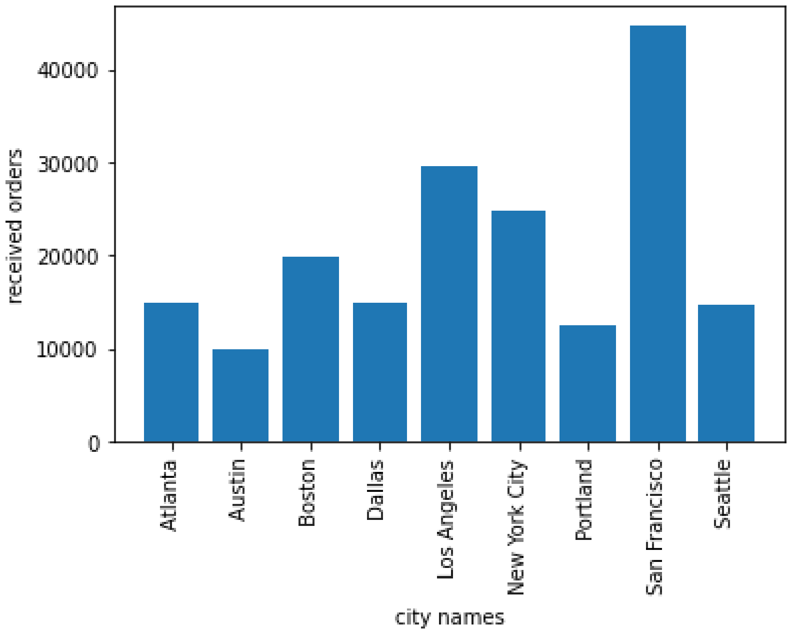
```
all_data.groupby('city')['city'].count()
```

Out[160]:

```
city
 Atlanta           14881
 Austin             9905
 Boston            19934
 Dallas            14820
 Los Angeles       29605
 New York City     24876
 Portland          12465
 San Francisco     44732
 Seattle           14732
Name: city, dtype: int64
```

In [161]:

```
plt.bar(all_data.groupby('city')['city'].count().index,all_data.groupby
('city')['city'].count())
plt.xticks(rotation='vertical')
plt.ylabel('received orders')
plt.xlabel('city names')
plt.show()
```



In [ ]:

```
```

What time should we display advertisements to maximise for product purchase?

In [ ]:

```
```

In [59]:

```
all_data['Order Date'][0].dtype
```

Out[59]:

```
dtype('O')
```

In [162]:

```
all_data['Hour'] = pd.to_datetime(all_data['Order Date']).dt.hour
```

In [163]:

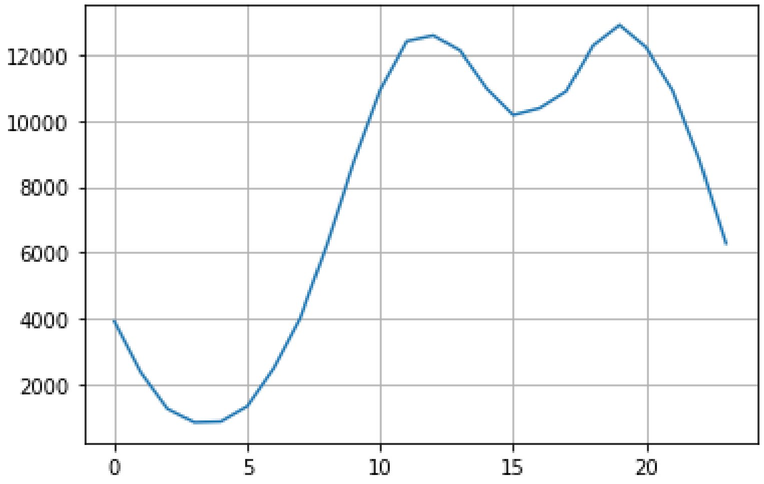
```
keys=[]
hour=[]
for key,hour_df in all_data.groupby('Hour'):
    keys.append(key)
    hour.append(len(hour_df))
```

In [164]:

```
plt.grid()
plt.plot(keys,hour)
```

Out[164]:

```
[<matplotlib.lines.Line2D at 0xe5aa76f08>]
```



between 12pm and 7pm is probably the best time to advertise to maximise product purchase

In [ ]:

```

```

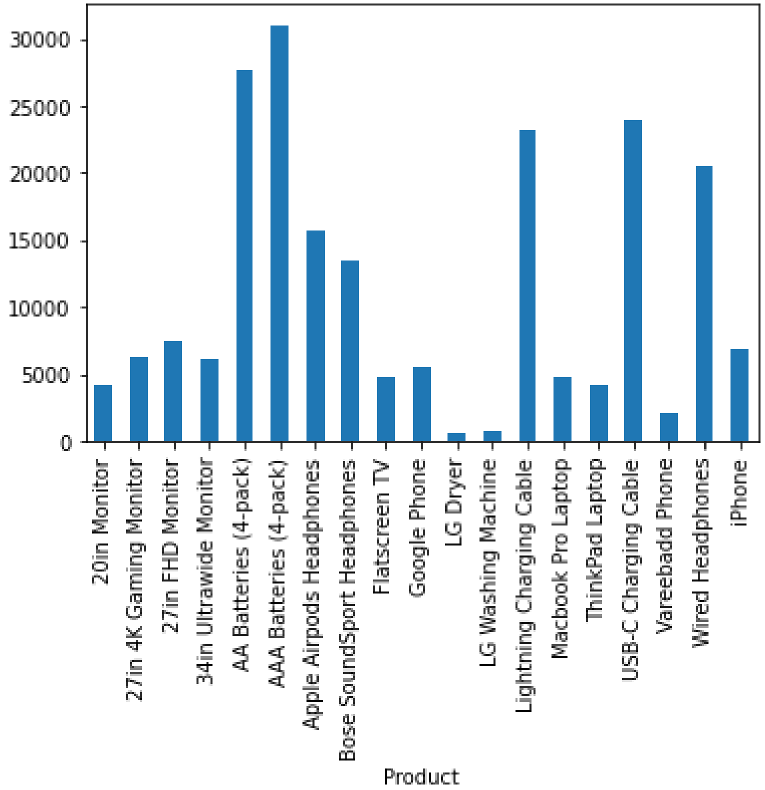
What product sold the most? & Why?

In [165]:

```
all_data.groupby('Product')['Quantity Ordered'].sum().plot(kind='bar')
```

Out[165]:

```
<matplotlib.axes._subplots.AxesSubplot at 0xe60459c48>
```



In [166]:

```
all_data.groupby('Product')['Price Each'].mean()
```

Out[166]:

```
Product
20in Monitor                  109.99
27in 4K Gaming Monitor        389.99
27in FHD Monitor              149.99
34in Ultrawide Monitor        379.99
AA Batteries (4-pack)           3.84
AAA Batteries (4-pack)          2.99
Apple Airpods Headphones      150.00
Bose SoundSport Headphones     99.99
Flatscreen TV                 300.00
Google Phone                  600.00
LG Dryer                      600.00
LG Washing Machine            600.00
```

```
Lightning Charging Cable       14.95
Macbook Pro Laptop           1700.00
ThinkPad Laptop               999.99
USB-C Charging Cable           11.95
Vareebadd Phone               400.00
Wired Headphones               11.99
iPhone                        700.00
Name: Price Each, dtype: float64
```

In [167]:

```
products=all_data.groupby('Product')['Quantity Ordered'].sum().index
quantity=all_data.groupby('Product')['Quantity Ordered'].sum()
prices=all_data.groupby('Product')['Price Each'].mean()
```

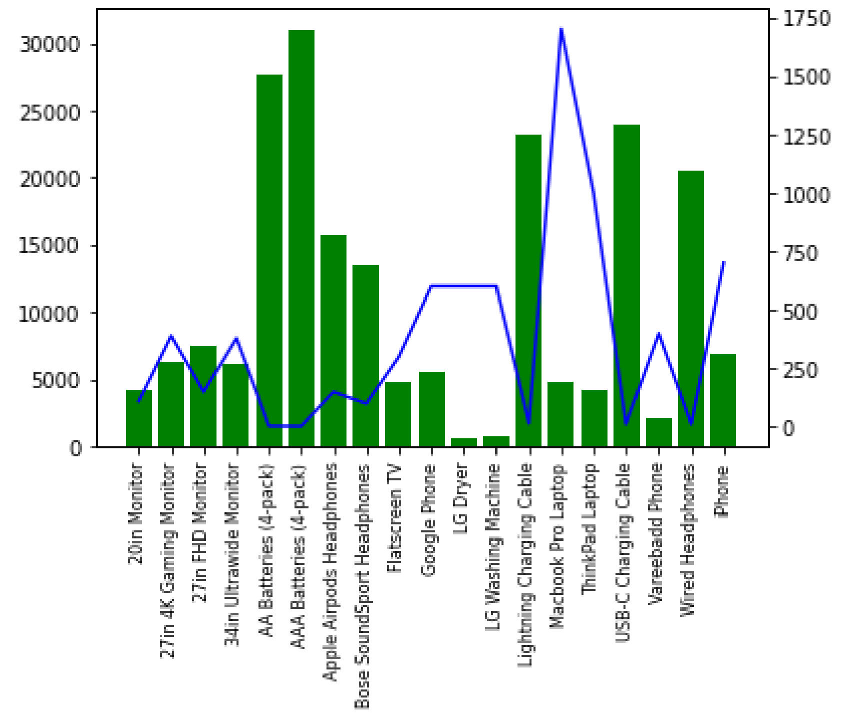
In [168]:

```
plt.figure(figsize=(40,24))
fig,ax1 = plt.subplots()
ax2=ax1.twinx()
ax1.bar(products, quantity, color='g')
ax2.plot(products, prices, 'b-')
ax1.set_xticklabels(products, rotation='vertical', size=8)
```

Out[168]:

```
[Text(0, 0, '20in Monitor'),
 Text(0, 0, '27in 4K Gaming Monitor'),
 Text(0, 0, '27in FHD Monitor'),
 Text(0, 0, '34in Ultrawide Monitor'),
 Text(0, 0, 'AA Batteries (4-pack)'),
 Text(0, 0, 'AAA Batteries (4-pack)'),
 Text(0, 0, 'Apple Airpods Headphones'),
 Text(0, 0, 'Bose SoundSport Headphones'),
 Text(0, 0, 'Flatscreen TV'),
 Text(0, 0, 'Google Phone'),
 Text(0, 0, 'LG Dryer'),
 Text(0, 0, 'LG Washing Machine'),
 Text(0, 0, 'Lightning Charging Cable'),
 Text(0, 0, 'Macbook Pro Laptop'),
 Text(0, 0, 'ThinkPad Laptop'),
 Text(0, 0, 'USB-C Charging Cable'),
 Text(0, 0, 'Vareebadd Phone'),
 Text(0, 0, 'Wired Headphones'),
 Text(0, 0, 'iPhone')]
```

```
<Figure size 2880x1728 with 0 Axes>
```



The top selling product is 'AAA Batteries'. The top selling products seem to have a correlation with the price of the product. The cheaper the product higher the quantity ordered and vice versa.

In [123]:
```
all_data.shape
```

Out[123]:
```
(197579, 10)
```

What products are most often sold together?

note: keep orders that have same order Id,are sold mostly together

In [169]:

```
df=all_data[all_data['Order ID'].duplicated(keep=False)]
df.head(20)
```

Out[169]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	sales	city	Hour
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles	14
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles	14
18	176574	Google Phone	1	600.00	04/03/19 19:42	20 Hill St, Los Angeles, CA 90001	4	600.00	Los Angeles	19
19	176574	USB-C Charging Cable	1	11.95	04/03/19 19:42	20 Hill St, Los Angeles, CA 90001	4	11.95	Los Angeles	19
30	176585	Bose SoundSport Headphones	1	99.99	04/07/19 11:31	823 Highland St, Boston, MA 02215	4	99.99	Boston	11
31	176585	Bose SoundSport Headphones	1	99.99	04/07/19 11:31	823 Highland St, Boston, MA 02215	4	99.99	Boston	11
32	176586	AAA Batteries (4-pack)	2	2.99	04/10/19 17:00	365 Center St, San Francisco, CA 94016	4	5.98	San Francisco	17

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	sales	city	Hour
33	176586	Google Phone	1	600.00	04/10/19 17:00	365 Center St, San Francisco, CA 94016	4	600.00	San Francisco	17
119	176672	Lightning Charging Cable	1	14.95	04/12/19 11:07	778 Maple St, New York City, NY 10001	4	14.95	New York City	11
120	176672	USB-C Charging Cable	1	11.95	04/12/19 11:07	778 Maple St, New York City, NY 10001	4	11.95	New York City	11
129	176681	Apple Airpods Headphones	1	150.00	04/20/19 10:39	331 Cherry St, Seattle, WA 98101	4	150.00	Seattle	10
130	176681	ThinkPad Laptop	1	999.99	04/20/19 10:39	331 Cherry St, Seattle, WA 98101	4	999.99	Seattle	10
138	176689	Bose SoundSport Headphones	1	99.99	04/24/19 17:15	659 Lincoln St, New York City, NY 10001	4	99.99	New York City	17
139	176689	AAA Batteries (4-pack)	2	2.99	04/24/19 17:15	659 Lincoln St, New York City, NY 10001	4	5.98	New York City	17
189	176739	34in Ultrawide Monitor	1	379.99	04/05/19 17:38	730 6th St, Austin, TX 73301	4	379.99	Austin	17

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	sales	city	Hour
190	176739	Google Phone	1	600.00	04/05/19 17:38	730 6th St, Austin, TX 73301	4	600.00	Austin	17
225	176774	Lightning Charging Cable	1	14.95	04/25/19 15:06	372 Church St, Los Angeles, CA 90001	4	14.95	Los Angeles	15
226	176774	USB-C Charging Cable	1	11.95	04/25/19 15:06	372 Church St, Los Angeles, CA 90001	4	11.95	Los Angeles	15
233	176781	iPhone	1	700.00	04/03/19 07:37	976 Hickory St, Dallas, TX 75001	4	700.00	Dallas	7
234	176781	Lightning Charging Cable	1	14.95	04/03/19 07:37	976 Hickory St, Dallas, TX 75001	4	14.95	Dallas	7

In [171]:

```
df.head()
```

Out[171]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	sales	city	Hour
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles	14
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles	14

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	sales	city	Hour	
18	176574	Google Phone	1	600.00	04/03/19 19:42	20 Hill St, Los Angeles, CA 90001	4	600.00	Los Angeles	19	
19	176574	USB-C Charging Cable	1	11.95	04/03/19 19:42	20 Hill St, Los Angeles, CA 90001	4	11.95	Los Angeles	19	
30	176585	Bose SoundSport Headphones	1	99.99	04/07/19 11:31	823 Highland St, Boston, MA 02215	4	99.99	Boston	11	B He So

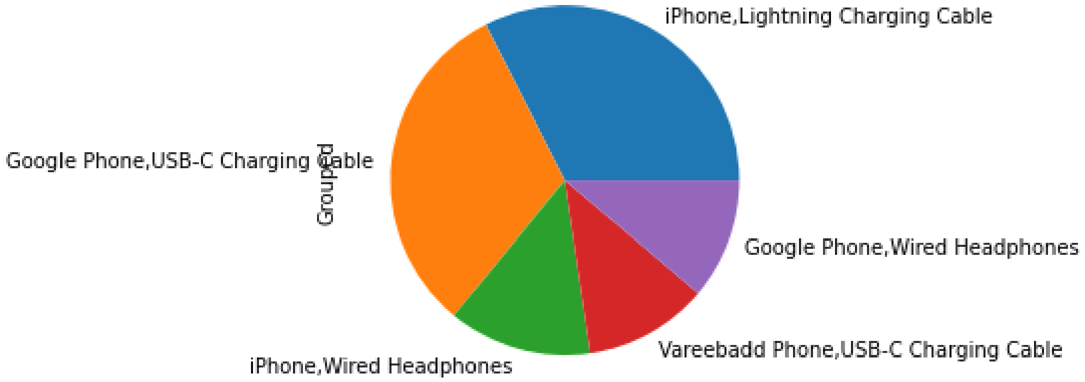
```
In [172]: df.shape
```

```
Out[172]: (14649, 11)
```

```
In [174]: #lets drop out all duplicate Order ID
          df2 = df.drop_duplicates(subset=['Order ID'])
```

```
In [179]: df2['Grouped'].value_counts()[0:5].plot.pie()
```

```
Out[179]: <matplotlib.axes._subplots.AxesSubplot at 0xe60fac2c8>
```



```
In [180]: import plotly.graph_objs as go
          from plotly.offline import iplot
```

```
In [185]: values=df2['Grouped'].value_counts()[0:5]
          labels=df['Grouped'].value_counts()[0:5].index
```

```
In [186]: trace=go.Pie(labels=labels, values=values,
                         hoverinfo='label+percent', textinfo='value',
                         textfont=dict(size=25),
                        pull=[0, 0, 0,0.2, 0]
                         )
```

```
In [187]: iplot([trace])
```