In [3]:

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

In [4]:

```
sales = pd.read_csv(
    'sales_data.csv',
    parse_dates=['Date'])
```

In [6]:

```
sales.head()
```

Out[6]:

	Date	Day	Month	Year	Customer_Age	Age_Group	Customer_Gender	Country	State	Proc
0	2013-11-26	26	November	2013	19	Youth (<25)	M	Canada	British Columbia	
1	2015-11-26	26	November	2015	19	Youth (<25)	M	Canada	British Columbia	
2	2014-03-23	23	March	2014	49	Adults (35-64)	M	Australia	New South Wales	
3	2016-03-23	23	March	2016	49	Adults (35-64)	M	Australia	New South Wales	
4	2014-05-15	15	May	2014	47	Adults (35-64)	F	Australia	New South Wales	

In [22]:

```
sales.shape #tells how many rows and columns we had
```

Out[22]:

```
(113036, 18)
```

In [24]:

```
sales.info() #showing the datatypes of the variables of the entire data set
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 113036 entries, 0 to 113035
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Date              113036 non-null  datetime64[ns]
 1   Day               113036 non-null  int64
 2   Month             113036 non-null  object
 3   Year              113036 non-null  int64
 4   Customer_Age      113036 non-null  int64
 5   Age_Group         113036 non-null  object
 6   Customer_Gender   113036 non-null  object
 7   Country           113036 non-null  object
 8   State             113036 non-null  object
 9   Product_Category  113036 non-null  object
 10  Sub_Category      113036 non-null  object
 11  Product           113036 non-null  object
 12  Order_Quantity    113036 non-null  int64
 13  Unit_Cost         113036 non-null  int64
 14  Unit_Price        113036 non-null  int64
 15  Profit            113036 non-null  int64
 16  Cost              113036 non-null  int64
 17  Revenue           113036 non-null  int64
dtypes: datetime64[ns](1), int64(9), object(8)
memory usage: 15.5+ MB
```

In [25]:

```
sales.describe() #showing numeric visualization of entire data's statistical properties
```

Out[25]:

	Day	Year	Customer_Age	Order_Quantity	Unit_Cost	Unit_Price	
count	113036.000000	113036.000000	113036.000000	113036.000000	113036.000000	113036.000000	113
mean	15.665753	2014.401739	35.919212	11.901660	267.296366	452.938427	
std	8.781567	1.272510	11.021936	9.561857	549.835483	922.071219	
min	1.000000	2011.000000	17.000000	1.000000	1.000000	2.000000	
25%	8.000000	2013.000000	28.000000	2.000000	2.000000	5.000000	
50%	16.000000	2014.000000	35.000000	10.000000	9.000000	24.000000	
75%	23.000000	2016.000000	43.000000	20.000000	42.000000	70.000000	
max	31.000000	2016.000000	87.000000	32.000000	2171.000000	3578.000000	15

In [26]:

```
sales['Unit_Cost'].mean() ## showing the mean unit cost
```

Out[26]:

```
267.296365759581
```

In [27]:

```
sales['Unit_Cost'].median()## median of the unit cost
```

Out[27]:

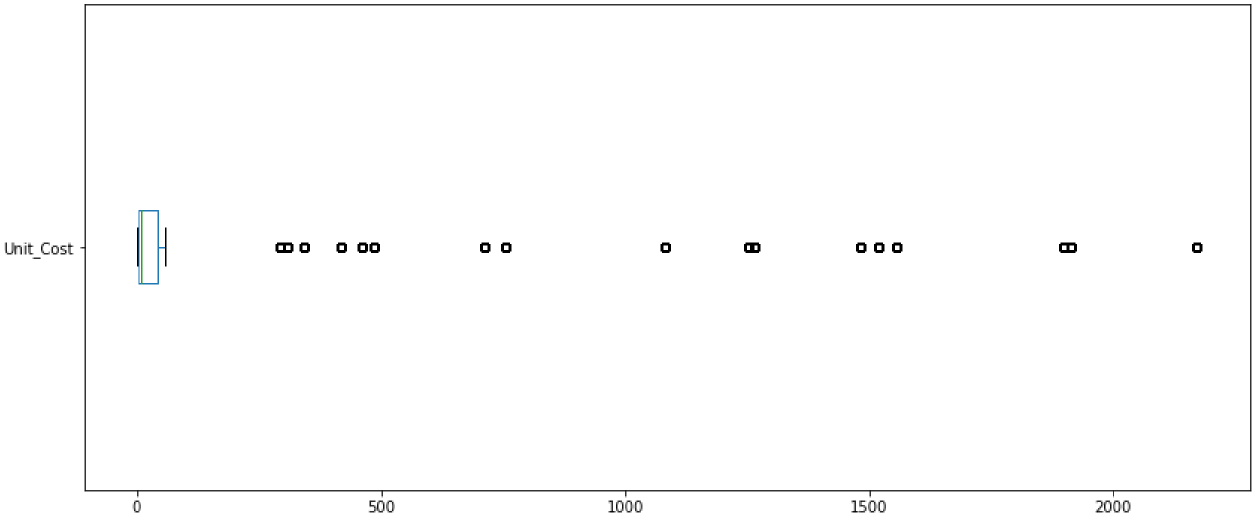
```
9.0
```

In [31]:

```
sales['Unit_Cost'].plot(kind='box', vert=False, figsize=(14,6))
```

Out[31]:

```
<AxesSubplot:>
```

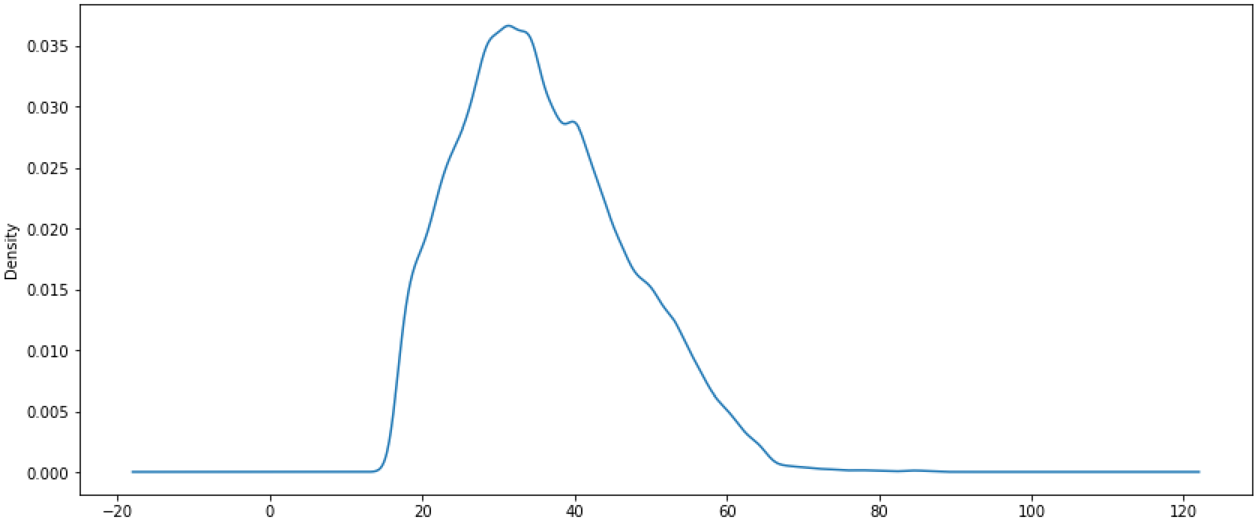


In [34]:

```
sales['Customer_Age'].plot(kind='kde', figsize=(14,6)) ##show a density (KDE) and a box
```

Out[34]:

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

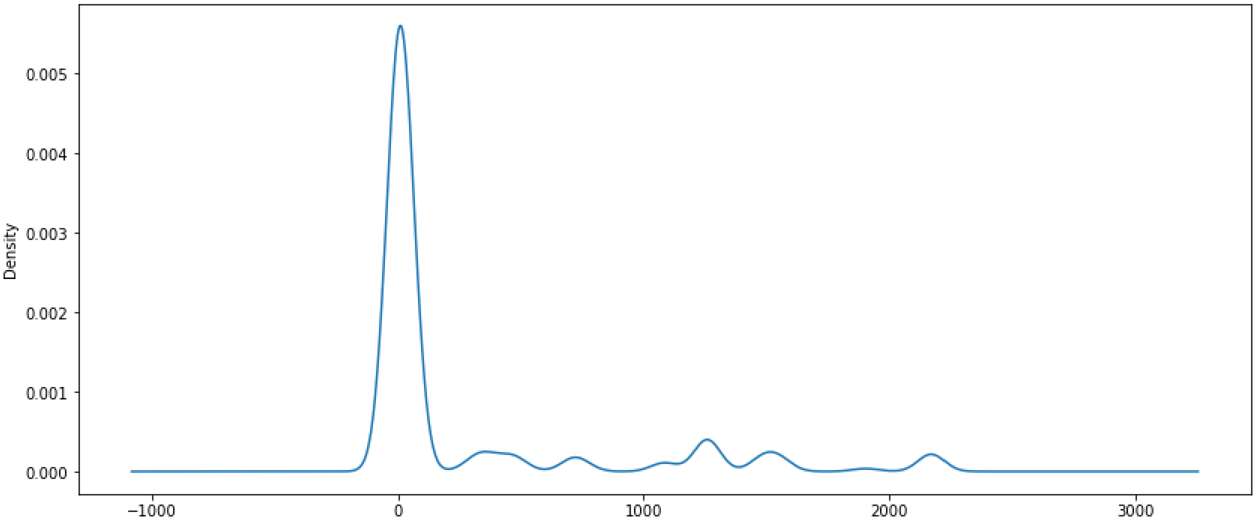


In [35]:

```
sales['Unit_Cost'].plot(kind='density', figsize=(14,6))
```

Out[35]:

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

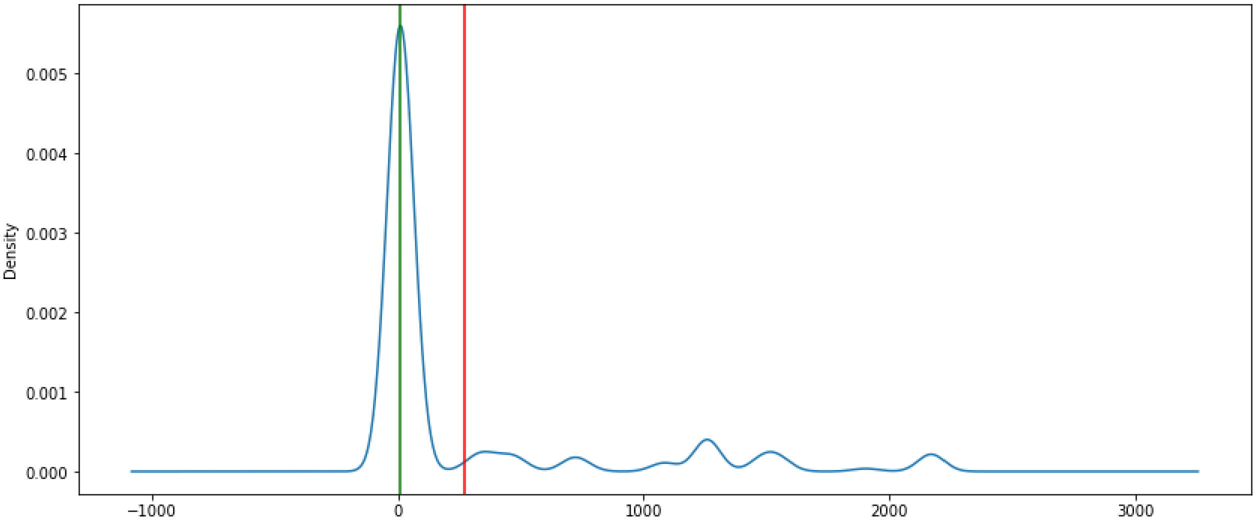


In [38]:

```
ax = sales['Unit_Cost'].plot(kind='density', figsize=(14,6))
ax.axvline(sales['Unit_Cost'].mean(), color='red') ## finding the mean of the cost
ax.axvline(sales['Unit_Cost'].median(), color='green') ##showing median of the unit cos
```

Out[38]:

```
<matplotlib.lines.Line2D at 0x2a5acc65eb0>
```

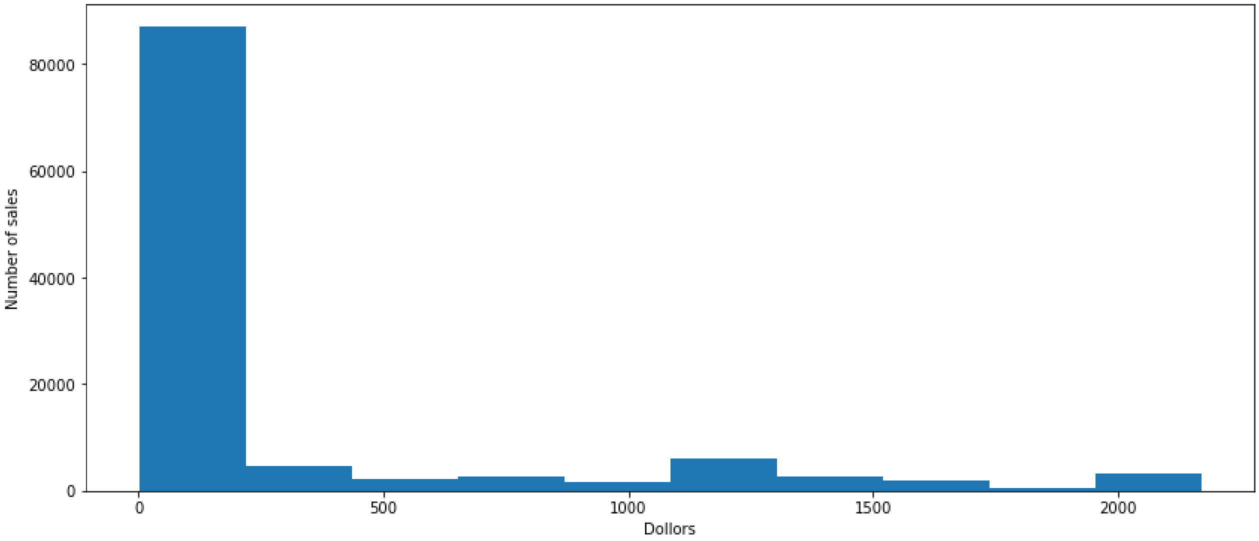


In [39]:

```
ax = sales['Unit_Cost'].plot(kind="hist", figsize=(14,6))
ax.set_ylabel('Number of sales')
ax.set_xlabel('Dollors')
```

Out[39]:

```
Text(0.5, 0, 'Dollors')
```



In [40]:

```
sales.head()
```

Out[40]:

	Date	Day	Month	Year	Customer_Age	Age_Group	Customer_Gender	Country	State	Proc
0	2013-11-26	26	November	2013	19	Youth (<25)	M	Canada	British Columbia	
1	2015-11-26	26	November	2015	19	Youth (<25)	M	Canada	British Columbia	
2	2014-03-23	23	March	2014	49	Adults (35-64)	M	Australia	New South Wales	
3	2016-03-23	23	March	2016	49	Adults (35-64)	M	Australia	New South Wales	
4	2014-05-15	15	May	2014	47	Adults (35-64)	F	Australia	New South Wales	

In [42]:

```
sales['Age_Group'].value_counts()
```

Out[42]:

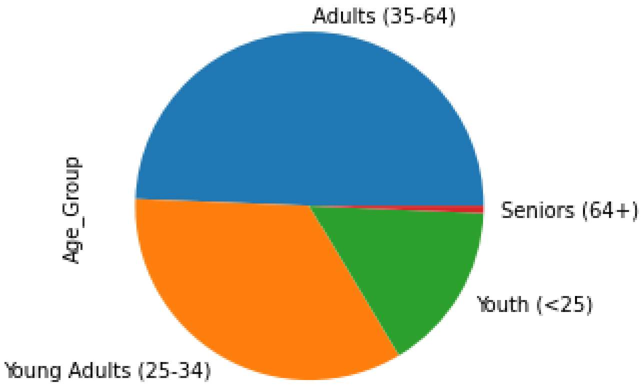
```
Adults (35-64)          55824
Young Adults (25-34)    38654
Youth (<25)             17828
Seniors (64+)             730
Name: Age_Group, dtype: int64
```

In [48]:

```
sales['Age_Group'].value_counts().plot(kind='pie', figsize=(16,4)) ##you need to put va
#because youre counting the total list
```

Out[48]:

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

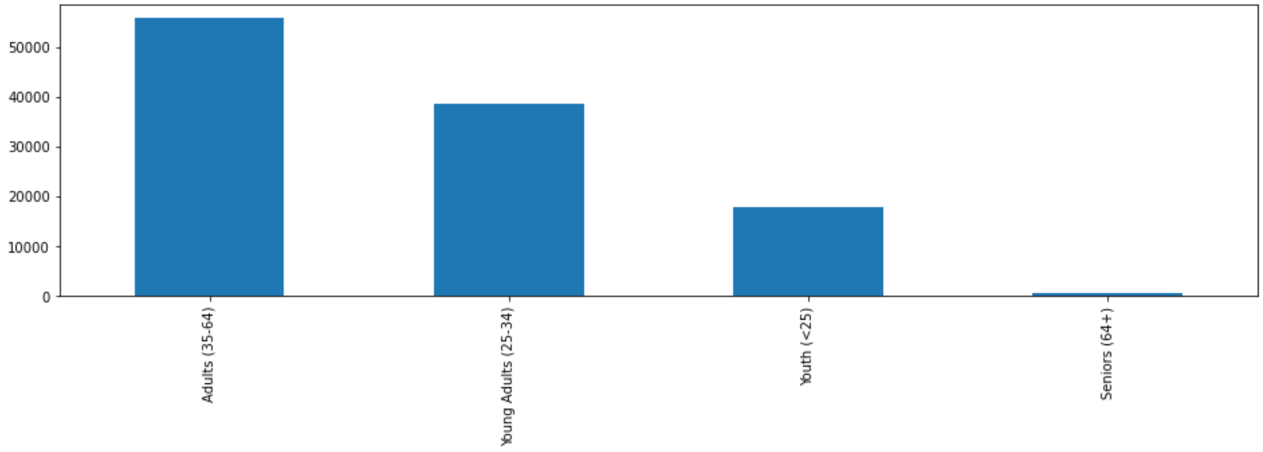


In [54]:

```
sales['Age_Group'].value_counts().plot(kind='bar', figsize=(16,4))
```

Out[54]:

```
<AxesSubplot:>
```



In [44]:

```
sales['Age_Group'].value_counts()
```

Out[44]:

```
California                22450
British Columbia          14116
England                   13620
Washington                11264
New South Wales           10412
Victoria                   6016
Oregon                     5286
Queensland                 5220
Saarland                   2770
Nordrhein-Westfalen        2484
Hessen                     2384
Seine (Paris)              2328
Hamburg                    1836
Seine Saint Denis          1684
Nord                       1670
South Australia            1564
Bayern                     1426
Hauts de Seine             1084
Essonne                     994
Yveline                     954
Tasmania                    724
Seine et Marne              394
Moselle                     386
```

```
Loiret                  382
Val d'Oise              264
Garonne (Haute)         208
Brandenburg             198
Val de Marne            158
Charente-Maritime       148
Somme                   134
Loir et Cher            120
Pas de Calais            90
Alberta                  56
Texas                    30
Illinois                 28
Ohio                     28
New York                 20
Florida                  14
Kentucky                 10
Utah                     10
South Carolina           10
Wyoming                   8
Georgia                   8
Montana                   6
Minnesota                 6
Ontario                   6
Missouri                  6
Alabama                   4
North Carolina            4
Arizona                   4
Mississippi               4
Virginia                  4
Massachusetts             2
Name: State, dtype: int64
```

In [55]:

```
# RELATIONSHIP BETWEEN THE COMLUNMS
```

In [59]:

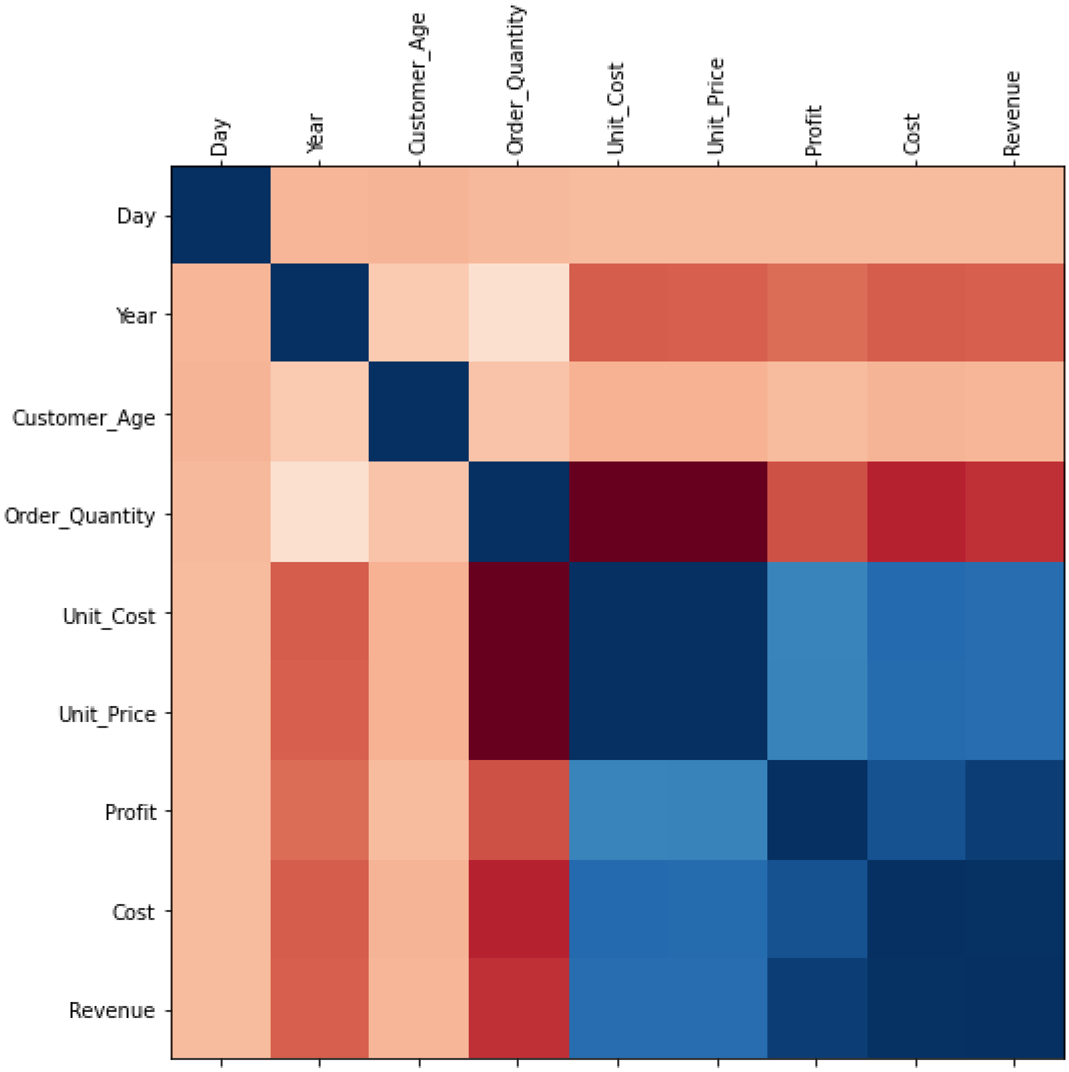
```
corr = sales.corr()
corr
```

Out[59]:

	Day	Year	Customer_Age	Order_Quantity	Unit_Cost	Unit_Price	Profit
Day	1.000000	-0.007635	-0.014296	-0.002412	0.003133	0.003207	0.004623
Year	-0.007635	1.000000	0.040994	0.123169	-0.217575	-0.213673	-0.181525
Customer_Age	-0.014296	0.040994	1.000000	0.026887	-0.021374	-0.020262	0.004319
Order_Quantity	-0.002412	0.123169	0.026887	1.000000	-0.515835	-0.515925	-0.238863
Unit_Cost	0.003133	-0.217575	-0.021374	-0.515835	1.000000	0.997894	0.741020
Unit_Price	0.003207	-0.213673	-0.020262	-0.515925	0.997894	1.000000	0.749870
Profit	0.004623	-0.181525	0.004319	-0.238863	0.741020	0.749870	1.000000
Cost	0.003329	-0.215604	-0.016013	-0.340382	0.829869	0.826301	0.902233
Revenue	0.003853	-0.208673	-0.009326	-0.312895	0.817865	0.818522	0.956572

In [62]:

```
fig =  plt.figure(figsize=(8,8)) ##size of figure
plt.matshow(corr, cmap='RdBu', fignum=fig.number)
plt.xticks(range(len(corr.columns)), corr.columns, rotation='vertical');
plt.yticks(range(len(corr.columns)), corr.columns);
```

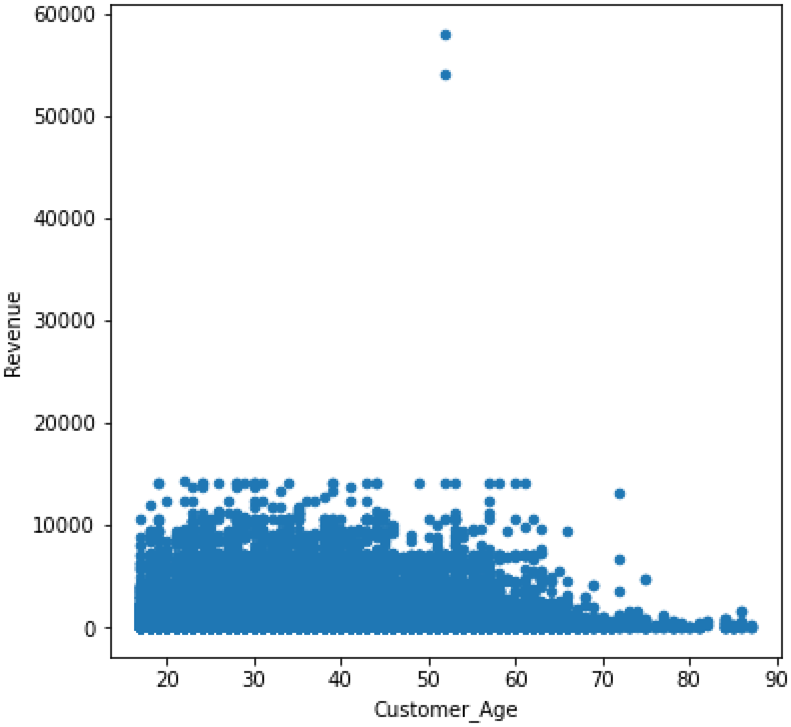


In [68]:

```
sales.plot(kind='scatter', x='Customer_Age', y='Revenue', figsize=(6,6)) ## coleration
```

Out[68]:

```
<AxesSubplot:xlabel='Customer_Age', ylabel='Revenue'>
```

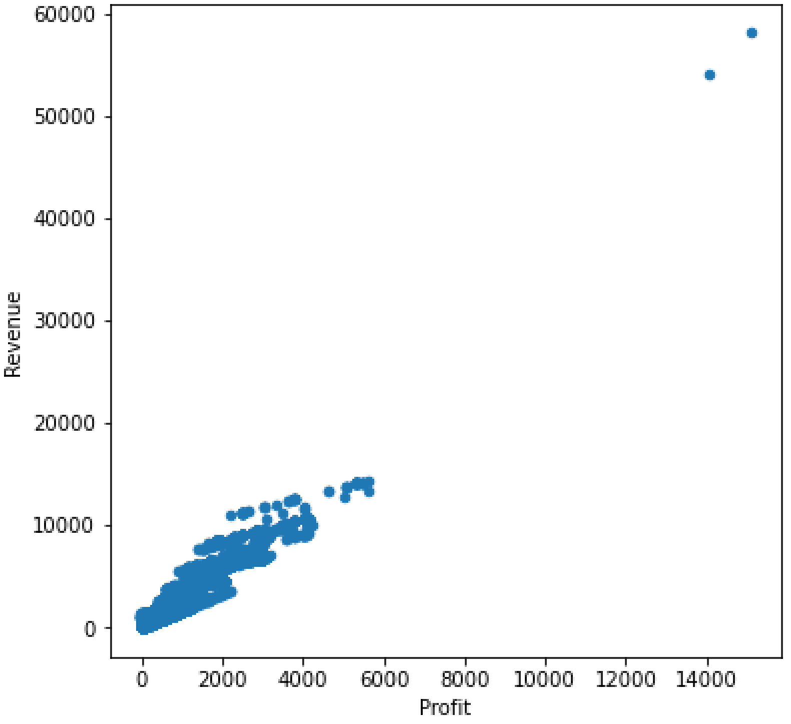


In [69]:

```
sales.plot(kind='scatter', x='Profit', y='Revenue', figsize=(6,6))
```

Out[69]:

```
<AxesSubplot:xlabel='Profit', ylabel='Revenue'>
```

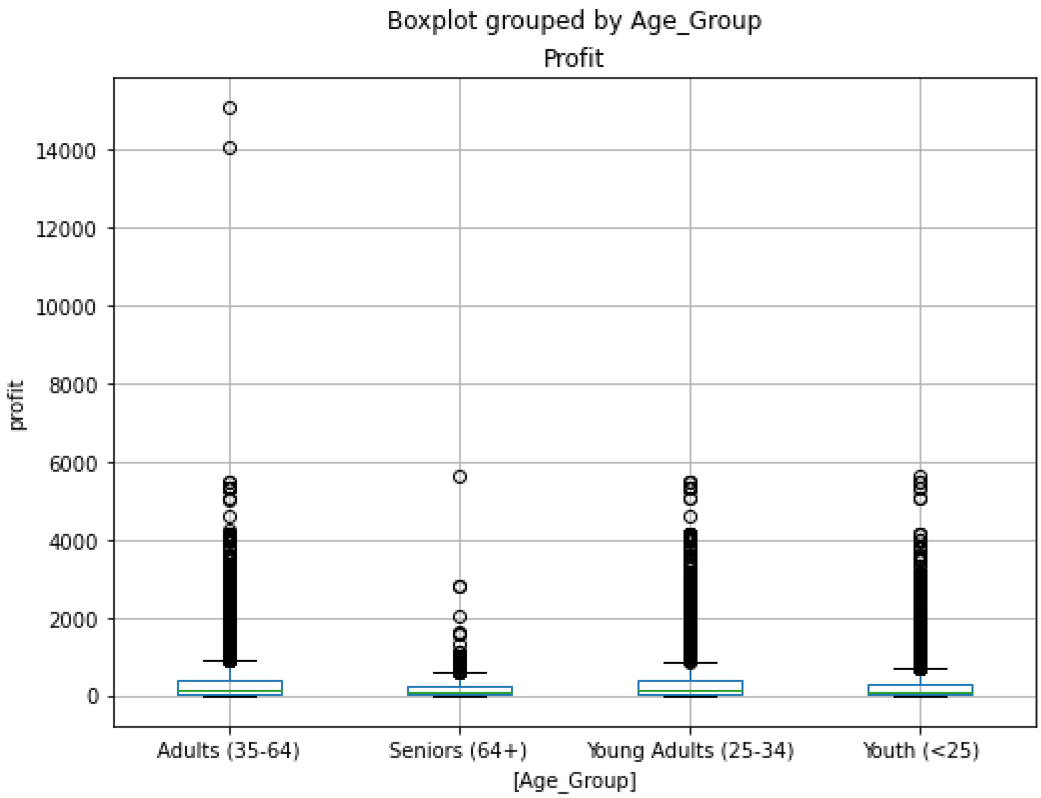


In [72]:

```
ax =  sales [['Profit', 'Age_Group']].boxplot(by='Age_Group', figsize=(8,6))
ax.set_ylabel('profit') #PROFIT BY AGE GROUP
```

Out[72]:

```
Text(0, 0.5, 'profit')
```



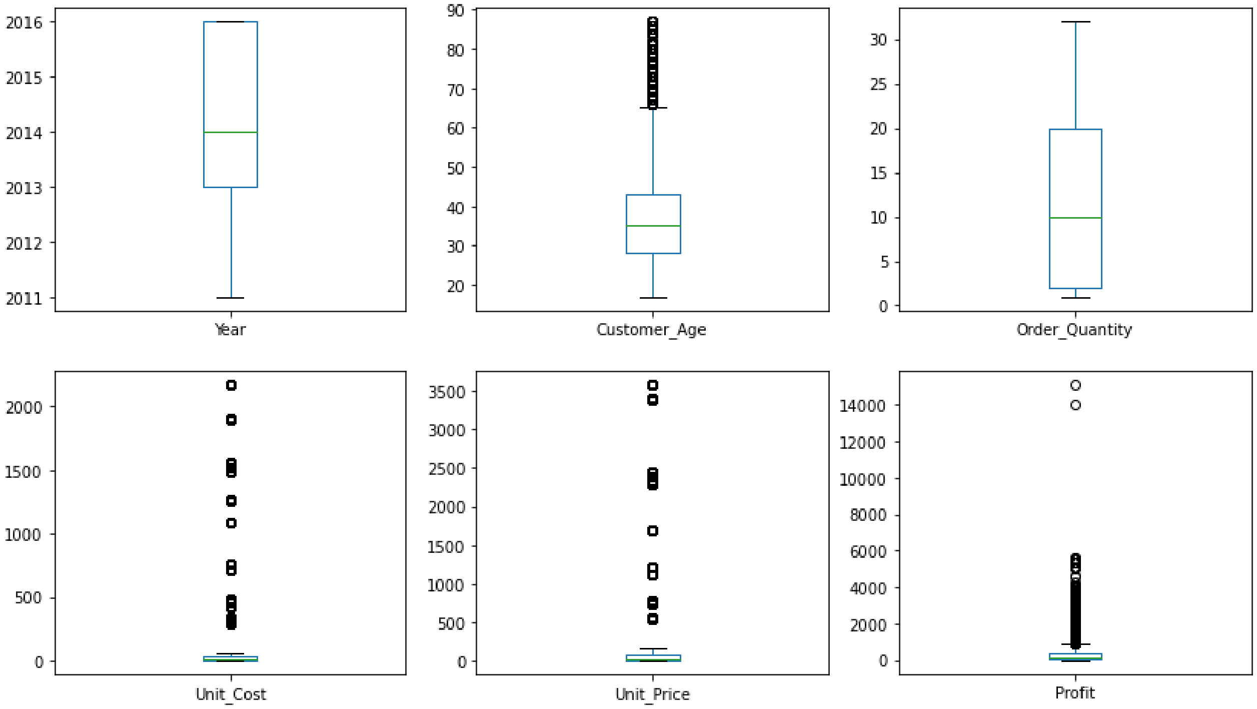
In [75]:

```
boxplot_cols = ['Year', 'Customer_Age', 'Order_Quantity', 'Unit_Cost', 'Unit_Price', 'P

sales[boxplot_cols].plot(kind='box', subplots=True, layout=(2,3), figsize=(14,8))
```

Out[75]:

```
Year                 AxesSubplot(0.125,0.536818;0.227941x0.343182)
Customer_Age      AxesSubplot(0.398529,0.536818;0.227941x0.343182)
Order_Quantity    AxesSubplot(0.672059,0.536818;0.227941x0.343182)
Unit_Cost               AxesSubplot(0.125,0.125;0.227941x0.343182)
Unit_Price           AxesSubplot(0.398529,0.125;0.227941x0.343182)
Profit               AxesSubplot(0.672059,0.125;0.227941x0.343182)
dtype: object
```



In [104…

```
sales['Revenue_per_Age'] = sales['Revenue']/sales['Customer_Age']
sales['Revenue_per_Age'].head()
```

Out[104…

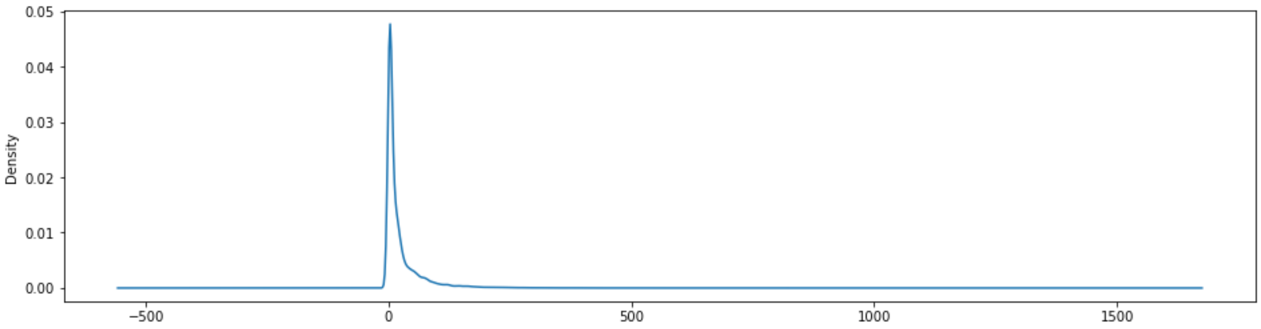
```
0    50.000000
1    50.000000
2    49.000000
3    42.612245
4     8.893617
Name: Revenue_per_Age, dtype: float64
```

In [100…

```
sales['Revenue_per_Age'].plot(kind='density', figsize=(16,4))
```

Out[100…

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

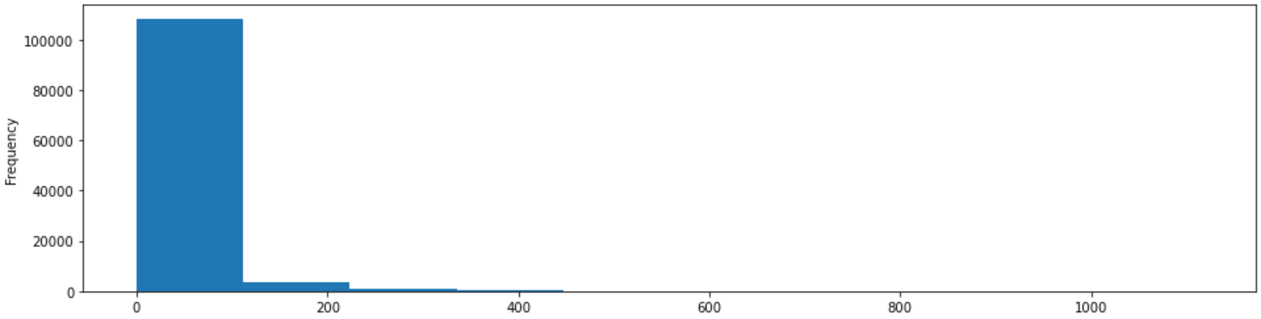


In [101…

```
sales['Revenue_per_Age'].plot(kind='hist', figsize=(16,4))
```

Out[101…

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



In [10]:

```
sales['Calculated_cost'] = sales['Unit_Cost'] * sales['Order_Quantity']
sales['Calculated_cost'].head()
```

Out[10]:

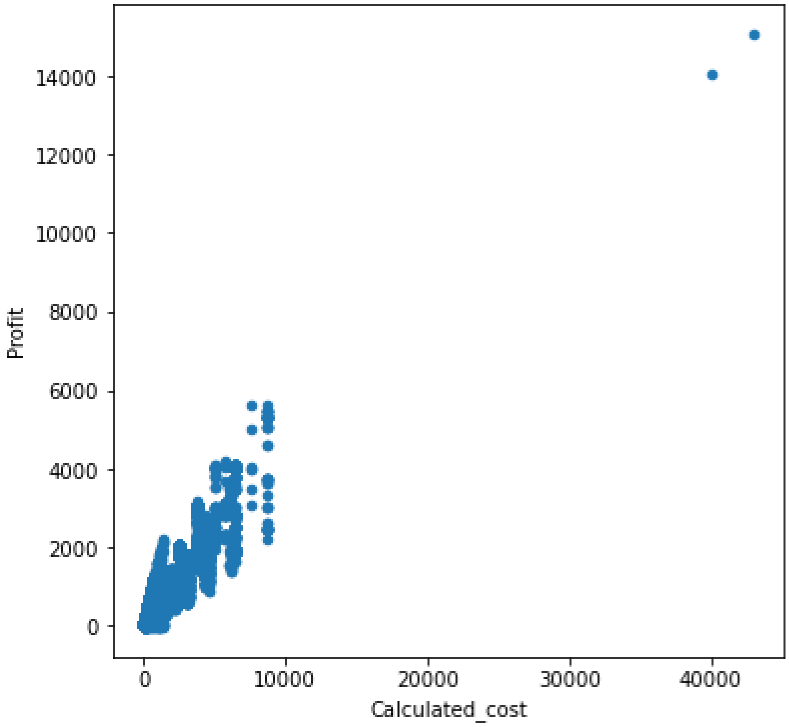
```
0     360
1     360
2    1035
3     900
4     180
Name: Calculated_cost, dtype: int64
```

In [11]:

```
sales.plot(kind='scatter', x='Calculated_cost', y='Profit', figsize=(6,6))
#We can see the relationship between Cost and Profit using a scatter plot:
```

Out[11]:

```
<AxesSubplot:xlabel='Calculated_cost', ylabel='Profit'>
```



In [12]:

```
(sales['Calculated_cost'] != sales['Cost']).sum() # to check if your calculation from c
```

Out[12]:

```
0
```

In [21]:

```
sales['Calculated_revenue']=  sales['Cost']  +  sales['Profit']
sales['Calculated_revenue'].head()
```

Out[21]:

```
0     950
1     950
2    2401
3    2088
4     418
Name: Calculated_revenue, dtype: int64
```

In [24]:

```
(sales['Calculated_revenue'] != sales['Revenue']).sum() # to check if your calculation 
```

Out[24]:

```
0
```

In [25]:

```
sales.head()
```

Out[25]:

	Date	Day	Month	Year	Customer_Age	Age_Group	Customer_Gender	Country	State	Proc
0	2013-11-26	26	November	2013	19	Youth (<25)	M	Canada	British Columbia	
1	2015-11-26	26	November	2015	19	Youth (<25)	M	Canada	British Columbia	

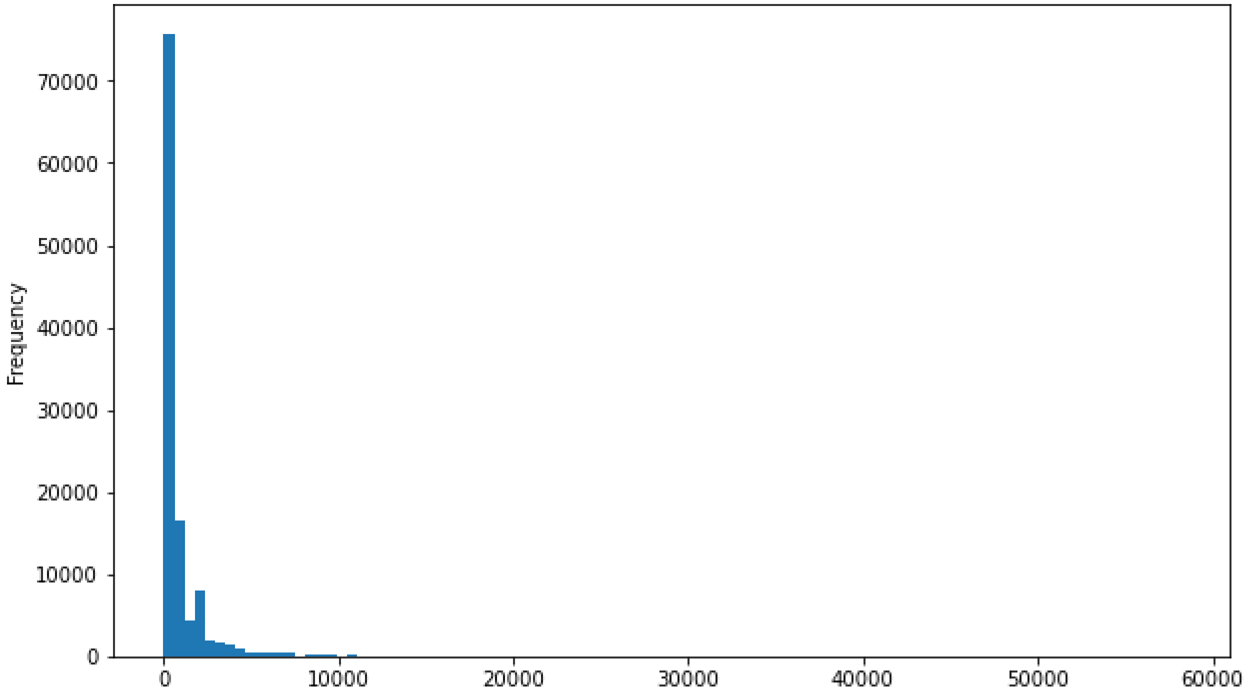
	Date	Day	Month	Year	Customer_Age	Age_Group	Customer_Gender	Country	State	Proc
2	2014-03-23	23	March	2014	49	Adults (35-64)	M	Australia	New South Wales	
3	2016-03-23	23	March	2016	49	Adults (35-64)	M	Australia	New South Wales	
4	2014-05-15	15	May	2014	47	Adults (35-64)	F	Australia	New South Wales	

In [26]:

```
sales['Revenue'].plot(kind='hist', bins=100, figsize=(10,6))
```

Out[26]:

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



In [5]:

```
sales['Unit_Price'].head()
```

Out[5]:

```
0    120
1    120
2    120
3    120
4    120
Name: Unit_Price, dtype: int64
```

In [6]:

```
sales['Country'].head()
```

Out[6]:

```
0       Canada
1       Canada
2    Australia
3    Australia
```

```
4    Australia
Name: Country, dtype: object
```

In [7]:

```
sales.loc[sales['Age_Group'] == 'Adults (35-64)', 'Revenue'].mean()
# GET MEAN REVENUE OF THE ADULTS
```

Out[7]:

```
762.8287654055604
```

In [12]:

```
sales.loc[(sales['Age_Group'] == 'Adults (35-64)') & (sales['Country']== 'United States
#GET THE REVENUE OF ADULTS 35+-64 YEARS OLD INN UNITED STATES
```

Out[12]:

```
726.7260473588342
```

In [16]:

```
sales.loc[sales['Country'] == 'France', 'Revenue'].head()
# get the revenue from particular country
```

Out[16]:

```
50     787
51     787
52    2957
53    2851
60     626
Name: Revenue, dtype: int64
```

In [17]:

```
#INCREASE THE REVENUE BY 10% TO EVERY SALES MADE IN FRANCE
sales.loc[sales['Country'] == 'France', 'Revenue'] *= 1.1
```

In [18]:

```
sales.loc[sales['Country'] == 'France', 'Revenue'].head()
```

Out[18]:

```
50     865.7
51     865.7
52    3252.7
53    3136.1
60     688.6
Name: Revenue, dtype: float64
```

In [ ]: