In [1]:

```
import numpy as np
import pandas as pd
from numpy import mean,std
import matplotlib.pyplot as pp
from numpy import cov
from scipy.stats import pearsonr
from scipy.stats import spearmanr
```

# Data Set 1

In [41]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\2015 - 2015.csv")
```

# a) Find mean, median, mode and describe

In [3]:

```
print(a.mean())
```

```
Happiness Rank                   79.493671
Happiness Score                   5.375734
Standard Error                    0.047885
Economy (GDP per Capita)          0.846137
Family                            0.991046
Health (Life Expectancy)          0.630259
Freedom                           0.428615
Trust (Government Corruption)     0.143422
Generosity                        0.237296
Dystopia Residual                 2.098977
dtype: float64
```

In [4]:

```
print(a.median())
```

```
Happiness Rank                   79.500000
Happiness Score                   5.232500
Standard Error                    0.043940
Economy (GDP per Capita)          0.910245
Family                            1.029510
Health (Life Expectancy)          0.696705
Freedom                           0.435515
Trust (Government Corruption)     0.107220
Generosity                        0.216130
Dystopia Residual                 2.095415
dtype: float64
```

In [5]:

```
print(a.mode())
```

```
         Country              Region  Happiness Rank  Happiness Score  \
0    Afghanistan  Sub-Saharan Africa            82.0            5.192   
1        Albania                 NaN             NaN              NaN   
2        Algeria                 NaN             NaN              NaN   
3         Angola                 NaN             NaN              NaN   
4      Argentina                 NaN             NaN              NaN   
..           ...                 ...             ...              ...   
153    Venezuela                 NaN             NaN              NaN   
154      Vietnam                 NaN             NaN              NaN   
155        Yemen                 NaN             NaN              NaN   
156       Zambia                 NaN             NaN              NaN   
157     Zimbabwe                 NaN             NaN              NaN   

     Standard Error  Economy (GDP per Capita)   Family  \
0           0.03751                   0.00000  0.00000   
1           0.03780                   0.01530  0.13995   
2           0.04394                   0.01604  0.30285   
3           0.04934                   0.06940  0.35386   
4           0.05051                   0.07120  0.38174   
..              ...                       ...      ...   
153             NaN                   1.45900  1.34043   
154             NaN                   1.52186  1.34951   
155             NaN                   1.55422  1.36058   
156             NaN                   1.56391  1.36948   
157             NaN                   1.69042  1.40223   

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)  \
0                     0.92356  0.00000                        0.32524   
1                         NaN  0.07699                            NaN   
2                         NaN  0.09245                            NaN   
3                         NaN  0.10081                            NaN   
4                         NaN  0.10384                            NaN   
..                        ...      ...                            ...   
153                       NaN  0.65821                            NaN   
154                       NaN  0.65980                            NaN   
155                       NaN  0.66246                            NaN   
156                       NaN  0.66557                            NaN   
157                       NaN  0.66973                            NaN   

     Generosity  Dystopia Residual  
0       0.00000            0.32858  
1       0.00199            0.65429  
2       0.02641            0.67042  
3       0.05444            0.67108  
4       0.05547            0.89991  
..          ...                ...  
153     0.51535            3.10712  
154     0.51752            3.17728  
155     0.51912            3.19131  
156     0.57630            3.26001  
157     0.79588            3.60214  

[158 rows x 12 columns]
```

In [6]:

```
print(a.describe())
```

```
       Happiness Rank  Happiness Score  Standard Error  \
count      158.000000       158.000000      158.000000
mean        79.493671         5.375734        0.047885
std         45.754363         1.145010        0.017146
min          1.000000         2.839000        0.018480
25%         40.250000         4.526000        0.037268
50%         79.500000         5.232500        0.043940
75%        118.750000         6.243750        0.052300
max        158.000000         7.587000        0.136930

       Economy (GDP per Capita)      Family  Health (Life Expectancy)  \
count                158.000000  158.000000                158.000000
mean                   0.846137    0.991046                  0.630259
std                    0.403121    0.272369                  0.247078
min                    0.000000    0.000000                  0.000000
25%                    0.545808    0.856823                  0.439185
50%                    0.910245    1.029510                  0.696705
75%                    1.158448    1.214405                  0.811013
max                    1.690420    1.402230                  1.025250

          Freedom  Trust (Government Corruption)  Generosity  \
count  158.000000                     158.000000  158.000000
mean     0.428615                       0.143422    0.237296
std      0.150693                       0.120034    0.126685
min      0.000000                       0.000000    0.000000
25%      0.328330                       0.061675    0.150553
50%      0.435515                       0.107220    0.216130
75%      0.549092                       0.180255    0.309883
max      0.669730                       0.551910    0.795880

       Dystopia Residual
count         158.000000
mean            2.098977
std             0.553550
min             0.328580
25%             1.759410
50%             2.095415
75%             2.462415
max             3.602140
```

# b) Find sum(), cumsum(), count, min and max values

In [8]:

```
b=a[['Happiness Score','Standard Error']]
print(b.sum())
```

```
Happiness Score    849.36600
Standard Error       7.56579
dtype: float64
```

In [9]:

```
print(b.cumsum())
```

```
     Happiness Score  Standard Error
0              7.587         0.03411
1             15.148         0.08295
2             22.675         0.11623
3             30.197         0.15503
4             37.624         0.19056
..               ...             ...
153          837.276         7.32523
154          840.616         7.36179
155          843.622         7.41194
156          846.527         7.49852
157          849.366         7.56579

[158 rows x 2 columns]
```

In [10]:

```
print(b.count())
```

```
Happiness Score    158
Standard Error     158
dtype: int64
```

In [11]:

```
print(b.min())
```

```
Happiness Score    2.83900
Standard Error     0.01848
dtype: float64
```

In [12]:

```
print(b.max())
```

```
Happiness Score    7.58700
Standard Error     0.13693
dtype: float64
```

# c) Find covariance and correlation (spearman and pearsons)

In [38]:

```
d1=a['Happiness Score']
d2=a['Standard Error']
print(cov(d1,d2))
```

```
[[ 1.31104821e+00 -3.47994395e-03]
 [-3.47994395e-03  2.93991439e-04]]
```

In [39]:

```
print(pearsonr(d1,d2))
```

```
(-0.17725380900494764, 0.025878684792533323)
```

In [40]:

```
print(spearmanr(d1,d2))
```

```
SpearmanrResult(correlation=-0.21519846171732626, pvalue=0.006619286429972
024)
```

# Data Set 2

In [48]:

```
a1=pd.read_csv(r"C:\Users\user\Downloads\Vehicle.csv")
a1
```

Out[48]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | |
|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | lounge | 51.0 | 882.0 | 25000.0 | 1.0 | 44.907242 | 8.611 |
| **1** | 2.0 | pop | 51.0 | 1186.0 | 32500.0 | 1.0 | 45.666359 | 12.241 |
| **2** | 3.0 | sport | 74.0 | 4658.0 | 142228.0 | 1.0 | 45.503300 | 11 |
| **3** | 4.0 | lounge | 51.0 | 2739.0 | 160000.0 | 1.0 | 40.633171 | 17.634 |
| **4** | 5.0 | pop | 73.0 | 3074.0 | 106880.0 | 1.0 | 41.903221 | 12.495 |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **1544** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **1545** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **1546** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | Null |
| **1547** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **1548** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |

1549 rows × 11 columns

# a) Find mean, median, mode and describe

In [27]:

```
print(a1.mean())
```

```
ID                     769.500000
engine_power            51.904421
age_in_days           1650.980494
km                   53396.011704
previous_owners          1.123537
lat                     43.541361
Unnamed: 9                    NaN
dtype: float64
```

In [28]:

```
print(a1.median())
```

```
ID                     769.500000
engine_power            51.000000
age_in_days           1035.000000
km                   39031.000000
previous_owners          1.000000
lat                     44.394096
Unnamed: 9                    NaN
dtype: float64
```

In [29]:

```
print(a1.mode())
```

```
          ID   model  engine_power  age_in_days       km  previous_owners  \
0        1.0  lounge          51.0        366.0  17000.0              1.0
1        2.0     NaN           NaN        790.0      NaN              NaN
2        3.0     NaN           NaN          NaN      NaN              NaN
3        4.0     NaN           NaN          NaN      NaN              NaN
4        5.0     NaN           NaN          NaN      NaN              NaN
...      ...     ...           ...          ...      ...              ...
1533  1534.0     NaN           NaN          NaN      NaN              NaN
1534  1535.0     NaN           NaN          NaN      NaN              NaN
1535  1536.0     NaN           NaN          NaN      NaN              NaN
1536  1537.0     NaN           NaN          NaN      NaN              NaN
1537  1538.0     NaN           NaN          NaN      NaN              NaN

            lat          lon  price  Unnamed: 9 Unnamed: 10
0     41.903221  12.49565029  10500         NaN      >10000
1           NaN          NaN    NaN         NaN         NaN
2           NaN          NaN    NaN         NaN         NaN
3           NaN          NaN    NaN         NaN         NaN
4           NaN          NaN    NaN         NaN         NaN
...         ...          ...    ...         ...         ...
1533        NaN          NaN    NaN         NaN         NaN
1534        NaN          NaN    NaN         NaN         NaN
1535        NaN          NaN    NaN         NaN         NaN
1536        NaN          NaN    NaN         NaN         NaN
1537        NaN          NaN    NaN         NaN         NaN

[1538 rows x 11 columns]
```

In [30]:

```
print(a1.mode())
```

```
          ID   model  engine_power  age_in_days       km  previous_owners  \
0        1.0  lounge          51.0        366.0  17000.0              1.0
1        2.0     NaN           NaN        790.0      NaN              NaN
2        3.0     NaN           NaN          NaN      NaN              NaN
3        4.0     NaN           NaN          NaN      NaN              NaN
4        5.0     NaN           NaN          NaN      NaN              NaN
...      ...     ...           ...          ...      ...              ...
1533  1534.0     NaN           NaN          NaN      NaN              NaN
1534  1535.0     NaN           NaN          NaN      NaN              NaN
1535  1536.0     NaN           NaN          NaN      NaN              NaN
1536  1537.0     NaN           NaN          NaN      NaN              NaN
1537  1538.0     NaN           NaN          NaN      NaN              NaN

            lat          lon  price  Unnamed: 9 Unnamed: 10
0     41.903221  12.49565029  10500         NaN      >10000
1           NaN          NaN    NaN         NaN         NaN
2           NaN          NaN    NaN         NaN         NaN
3           NaN          NaN    NaN         NaN         NaN
4           NaN          NaN    NaN         NaN         NaN
...         ...          ...    ...         ...         ...
1533        NaN          NaN    NaN         NaN         NaN
1534        NaN          NaN    NaN         NaN         NaN
1535        NaN          NaN    NaN         NaN         NaN
1536        NaN          NaN    NaN         NaN         NaN
1537        NaN          NaN    NaN         NaN         NaN

[1538 rows x 11 columns]
```

# b) Find sum(), cumsum(), count, min and max values

In [32]:

```
b1=a1[['engine_power','km']]
print(b1.sum())
```

```
engine_power       79829.0
km              82123066.0
dtype: float64
```

In [33]:

```
print(b1.cumsum())
```

```
      engine_power        km
0             51.0   25000.0
1            102.0   57500.0
2            176.0  199728.0
3            227.0  359728.0
4            300.0  466608.0
...            ...       ...
1544           NaN       NaN
1545           NaN       NaN
1546           NaN       NaN
1547           NaN       NaN
1548           NaN       NaN

[1549 rows x 2 columns]
```

In [34]:

```
print(b1.count())
```

```
engine_power    1538
km              1538
dtype: int64
```

In [35]:

```
print(b1.min())
```

```
engine_power      51.0
km              1232.0
dtype: float64
```

In [36]:

```
print(b1.max())
```

```
engine_power        77.0
km              235000.0
dtype: float64
```

In [53]:

```
c=a1.fillna(value=3)
print(c)
```

```
        ID   model  engine_power  age_in_days        km  previous_owners  \
0      1.0  lounge          51.0        882.0   25000.0              1.0
1      2.0     pop          51.0       1186.0   32500.0              1.0
2      3.0   sport          74.0       4658.0  142228.0              1.0
3      4.0  lounge          51.0       2739.0  160000.0              1.0
4      5.0     pop          73.0       3074.0  106880.0              1.0
...    ...     ...           ...          ...       ...              ...
1544   3.0       3           3.0          3.0       3.0              3.0
1545   3.0       3           3.0          3.0       3.0              3.0
1546   3.0       3           3.0          3.0       3.0              3.0
1547   3.0       3           3.0          3.0       3.0              3.0
1548   3.0       3           3.0          3.0       3.0              3.0

            lat          lon     price  Unnamed: 9 Unnamed: 10
0     44.907242  8.611559868      8900         3.0           3
1     45.666359  12.24188995      8800         3.0           3
2     45.503300     11.41784      4200         3.0           3
3     40.633171  17.63460922      6000         3.0           3
4     41.903221  12.49565029      5700         3.0           3
...         ...          ...       ...         ...         ...
1544   3.000000       length         5         3.0           3
1545   3.000000       concat  lonprice         3.0           3
1546   3.000000  Null values        NO         3.0           3
1547   3.000000         find         1         3.0           3
1548   3.000000       search         1         3.0           3

[1549 rows x 11 columns]
```

# c) Find covariance and correlation (spearman and pearsons)

In [54]:

```
e1=c['ID']
e2=c['km']
print(cov(e1,e2))
```

```
[[1.99992116e+05 1.73315627e+05]
 [1.73315627e+05 1.61246637e+09]]
```

In [55]:

```
print(pearsonr(e1,e2))
```

```
(0.009651303080527938, 0.7042781545927824)
```

In [56]:

```
print(spearmanr(e1,e2))
```

```
SpearmanrResult(correlation=0.04484417407698012, pvalue=0.0776629557803589
9)
```

# Data Set 3

In [58]:

```
a2=pd.read_csv(r"C:\Users\user\Downloads\4_drug200.csv")
a2
```

Out[58]:

| | Age | Sex | BP | Cholesterol | Na_to_K | Drug |
|---|---|---|---|---|---|---|
| **0** | 23 | F | HIGH | HIGH | 25.355 | drugY |
| **1** | 47 | M | LOW | HIGH | 13.093 | drugC |
| **2** | 47 | M | LOW | HIGH | 10.114 | drugC |
| **3** | 28 | F | NORMAL | HIGH | 7.798 | drugX |
| **4** | 61 | F | LOW | HIGH | 18.043 | drugY |
| **...** | ... | ... | ... | ... | ... | ... |
| **195** | 56 | F | LOW | HIGH | 11.567 | drugC |
| **196** | 16 | M | LOW | HIGH | 12.006 | drugC |
| **197** | 52 | M | NORMAL | HIGH | 9.894 | drugX |
| **198** | 23 | M | NORMAL | NORMAL | 14.020 | drugX |
| **199** | 40 | F | LOW | NORMAL | 11.349 | drugX |

200 rows × 6 columns

# a) Find mean, median, mode and describe

In [59]:

```
print(a2.mean())
```

```
Age         44.315000
Na_to_K     16.084485
dtype: float64
```

In [60]:

```
print(a2.median())
```

```
Age         45.0000
Na_to_K     13.9365
dtype: float64
```

In [61]:

```
print(a2.mode())
```

```
    Age  Sex    BP Cholesterol  Na_to_K   Drug
0  47.0    M  HIGH        HIGH   12.006  drugY
1   NaN  NaN   NaN         NaN   18.295    NaN
```

In [62]:

```
print(a2.describe())
```

```
              Age     Na_to_K
count  200.000000  200.000000
mean    44.315000   16.084485
std     16.544315    7.223956
min     15.000000    6.269000
25%     31.000000   10.445500
50%     45.000000   13.936500
75%     58.000000   19.380000
max     74.000000   38.247000
```

# b) Find sum(), cumsum(), count, min and max values

In [63]:

```
b2=a2[['Age','Na_to_K']]
print(b2.sum())
```

```
Age        8863.000
Na_to_K    3216.897
dtype: float64
```

In [64]:

```
print(b2.cumsum())
```

```
      Age   Na_to_K
0      23    25.355
1      70    38.448
2     117    48.562
3     145    56.360
4     206    74.403
..    ...       ...
195  8732  3169.628
196  8748  3181.634
197  8800  3191.528
198  8823  3205.548
199  8863  3216.897

[200 rows x 2 columns]
```

In [65]:

```
print(b2.count())
```

```
Age        200
Na_to_K    200
dtype: int64
```

In [66]:

```
print(b2.min())
```

```
Age        15.000
Na_to_K     6.269
dtype: float64
```

In [67]:

```
print(b2.max())
```

```
Age        74.000
Na_to_K    38.247
dtype: float64
```

# c) Find covariance and correlation (spearman and pearsons)

In [68]:

```
f1=b2['Age']
f2=b2['Na_to_K']
print(cov(f1,f2))
```

```
[[273.71434673  -7.54375153]
 [ -7.54375153  52.18553348]]
```

In [69]:

```
print(pearsonr(f1,f2))
```

```
(-0.06311949726772592, 0.3745756399034559)
```

In [70]:

```
print(spearmanr(f1,f2))
```

```
SpearmanrResult(correlation=-0.047273882688479915, pvalue=0.50622005813874
18)
```

# Data Set 4

In [71]:

```
a3=pd.read_csv(r"C:\Users\user\Downloads\5_Instagram data.csv")
a3
```

Out[71]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | Comments | Shares | Likes | Profile Visits |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3920 | 2586 | 1028 | 619 | 56 | 98 | 9 | 5 | 162 | 35 |
| **1** | 5394 | 2727 | 1838 | 1174 | 78 | 194 | 7 | 14 | 224 | 48 |
| **2** | 4021 | 2085 | 1188 | 0 | 533 | 41 | 11 | 1 | 131 | 62 |
| **3** | 4528 | 2700 | 621 | 932 | 73 | 172 | 10 | 7 | 213 | 23 |
| **4** | 2518 | 1704 | 255 | 279 | 37 | 96 | 5 | 4 | 123 | 8 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | .. |
| **114** | 13700 | 5185 | 3041 | 5352 | 77 | 573 | 2 | 38 | 373 | 73 |
| **115** | 5731 | 1923 | 1368 | 2266 | 65 | 135 | 4 | 1 | 148 | 20 |
| **116** | 4139 | 1133 | 1538 | 1367 | 33 | 36 | 0 | 1 | 92 | 34 |
| **117** | 32695 | 11815 | 3147 | 17414 | 170 | 1095 | 2 | 75 | 549 | 148 |
| **118** | 36919 | 13473 | 4176 | 16444 | 2547 | 653 | 5 | 26 | 443 | 61 |

119 rows × 13 columns

# a) Find mean, median, mode and describe

In [72]:

```
print(a3.mean())
```

```
Impressions       5703.991597
From Home         2475.789916
From Hashtags     1887.512605
From Explore      1078.100840
From Other         171.092437
Saves              153.310924
Comments             6.663866
Shares               9.361345
Likes              173.781513
Profile Visits      50.621849
Follows             20.756303
dtype: float64
```

In [73]:

```
print(a3.median())
```

```
Impressions       4289.0
From Home         2207.0
From Hashtags     1278.0
From Explore       326.0
From Other          74.0
Saves              109.0
Comments             6.0
Shares               6.0
Likes              151.0
Profile Visits      23.0
Follows              8.0
dtype: float64
```

In [74]:

```
print(a3.mode())
```

```
    Impressions  From Home  From Hashtags  From Explore  From Other  Saves  \
0        5394.0     1975.0            116          45.0        34.0   40.0   
1           NaN        NaN            201          84.0         NaN  135.0   
2           NaN        NaN            278           NaN         NaN  144.0   
3           NaN        NaN            362           NaN         NaN    NaN   
4           NaN        NaN            411           NaN         NaN    NaN   
5           NaN        NaN            583           NaN         NaN    NaN   
6           NaN        NaN            655           NaN         NaN    NaN   
7           NaN        NaN            707           NaN         NaN    NaN   
8           NaN        NaN            771           NaN         NaN    NaN   
9           NaN        NaN            794           NaN         NaN    NaN   
10          NaN        NaN           1248           NaN         NaN    NaN   
11          NaN        NaN           1260           NaN         NaN    NaN   
12          NaN        NaN           1278           NaN         NaN    NaN   
13          NaN        NaN           1693           NaN         NaN    NaN   
14          NaN        NaN           1938           NaN         NaN    NaN   
15          NaN        NaN           2351           NaN         NaN    NaN   
16          NaN        NaN           2975           NaN         NaN    NaN   
17          NaN        NaN           3450           NaN         NaN    NaN   
18          NaN        NaN           3551           NaN         NaN    NaN   

    Comments  Shares  Likes  Profile Visits  Follows  \
0        6.0     3.0  114.0            19.0      2.0   
1        NaN     NaN  151.0            21.0      NaN   
2        NaN     NaN    NaN             NaN      NaN   
3        NaN     NaN    NaN             NaN      NaN   
4        NaN     NaN    NaN             NaN      NaN   
5        NaN     NaN    NaN             NaN      NaN   
6        NaN     NaN    NaN             NaN      NaN   
7        NaN     NaN    NaN             NaN      NaN   
8        NaN     NaN    NaN             NaN      NaN   
9        NaN     NaN    NaN             NaN      NaN   
10       NaN     NaN    NaN             NaN      NaN   
11       NaN     NaN    NaN             NaN      NaN   
12       NaN     NaN    NaN             NaN      NaN   
13       NaN     NaN    NaN             NaN      NaN   
14       NaN     NaN    NaN             NaN      NaN   
15       NaN     NaN    NaN             NaN      NaN   
16       NaN     NaN    NaN             NaN      NaN   
17       NaN     NaN    NaN             NaN      NaN   
18       NaN     NaN    NaN             NaN      NaN   

                                              Caption  \
0   Here are some of the best data science project...   
1   Here are some of the best websites that you ca...   
2                                                 NaN   
3                                                 NaN   
4                                                 NaN   
5                                                 NaN   
6                                                 NaN   
7                                                 NaN   
8                                                 NaN   
9                                                 NaN   
10                                                NaN   
11                                                NaN   
12                                                NaN   
13                                                NaN   
14                                                NaN   
15                                                NaN   
16                                                NaN   
```

```
17                                               NaN
18                                               NaN

                                            Hashtags
0   #data�#datascience�#dataanalysis�#dataanalytic...
1                                                NaN
2                                                NaN
3                                                NaN
4                                                NaN
5                                                NaN
6                                                NaN
7                                                NaN
8                                                NaN
9                                                NaN
10                                               NaN
11                                               NaN
12                                               NaN
13                                               NaN
14                                               NaN
15                                               NaN
16                                               NaN
17                                               NaN
18                                               NaN
```

In [75]:

```
print(a3.describe())
```

```
        Impressions     From Home  From Hashtags  From Explore   From Othe
r  \
count     119.000000    119.000000     119.000000    119.000000   119.00000
0
mean     5703.991597   2475.789916    1887.512605   1078.100840   171.09243
7
std      4843.780105   1489.386348    1884.361443   2613.026132   289.43103
1
min      1941.000000   1133.000000     116.000000      0.000000     9.00000
0
25%      3467.000000   1945.000000     726.000000    157.500000    38.00000
0
50%      4289.000000   2207.000000    1278.000000    326.000000    74.00000
0
75%      6138.000000   2602.500000    2363.500000    689.500000   196.00000
0
max     36919.000000  13473.000000   11817.000000  17414.000000  2547.00000
0

             Saves    Comments      Shares       Likes  Profile Visits  \
count   119.000000  119.000000  119.000000  119.000000      119.000000
mean    153.310924    6.663866    9.361345  173.781513       50.621849
std     156.317731    3.544576   10.089205   82.378947       87.088402
min      22.000000    0.000000    0.000000   72.000000        4.000000
25%      65.000000    4.000000    3.000000  121.500000       15.000000
50%     109.000000    6.000000    6.000000  151.000000       23.000000
75%     169.000000    8.000000   13.500000  204.000000       42.000000
max    1095.000000   19.000000   75.000000  549.000000      611.000000

          Follows
count  119.000000
mean    20.756303
std     40.921580
min      0.000000
25%      4.000000
50%      8.000000
75%     18.000000
max    260.000000
```

# b) Find sum(), cumsum(), count, min and max values

In [76]:

```
b3=a3[['Impressions','From Home']]
print(b3.sum())
```

```
Impressions    678775
From Home      294619
dtype: int64
```

In [77]:

```
print(b3.cumsum())
```

```
     Impressions  From Home
0           3920       2586
1           9314       5313
2          13335       7398
3          17863      10098
4          20381      11802
..           ...        ...
114       599291     266275
115       605022     268198
116       609161     269331
117       641856     281146
118       678775     294619

[119 rows x 2 columns]
```

In [78]:

```
print(b3.count())
```

```
Impressions    119
From Home      119
dtype: int64
```

In [79]:

```
print(b3.min())
```

```
Impressions    1941
From Home      1133
dtype: int64
```

In [80]:

```
print(b3.max())
```

```
Impressions    36919
From Home      13473
dtype: int64
```

# c) Find covariance and correlation (spearman and pearsons)

In [82]:

```
g1=a3['Saves']
g2=a3['Comments']
print(cov(g1,g2))
```

```
[[ 2.44352330e+04 -1.49115511e+01]
 [-1.49115511e+01  1.25640222e+01]]
```

In [83]:

```
print(pearsonr(g1,g2))
```

```
(-0.02691226370756101, 0.7714093067398262)
```

In [84]:

```
print(spearmanr(g1,g2))
```

```
SpearmanrResult(correlation=0.18289066665208123, pvalue=0.0464953934494190
5)
```

# Data Set 5

In [85]:

```
a4=pd.read_csv(r"C:\Users\user\Downloads\6_Salesworkload1.csv")
a4
```

Out[85]:

| | MonthYear | Time index | Country | StoreID | City | Dept_ID | Dept. Name | HoursOwn | Hour |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 1.0 | Dry | 3184.764 | |
| **1** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 2.0 | Frozen | 1582.941 | |
| **2** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 3.0 | other | 47.205 | |
| **3** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 4.0 | Fish | 1623.852 | |
| **4** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 5.0 | Fruits & Vegetables | 1759.173 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **7653** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 12.0 | Checkout | 6322.323 | |
| **7654** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 16.0 | Customer Services | 4270.479 | |
| **7655** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 11.0 | Delivery | 0 | |
| **7656** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 17.0 | others | 2224.929 | |
| **7657** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 18.0 | all | 39652.2 | |

7658 rows × 14 columns

# a) Find mean, median, mode and describe

In [86]:

```
print(a4.mean())
```

```
Time index     5.000000e+00
StoreID        6.199522e+04
Dept_ID        9.470588e+00
HoursLease     2.203608e+01
Sales units    1.076471e+06
Turnover       3.721393e+06
Customer                NaN
dtype: float64
```

In [87]:

```
print(a4.median())
```

```
Time index          5.0
StoreID         75400.5
Dept_ID             9.0
HoursLease          0.0
Sales units    293230.0
Turnover       931957.5
Customer            NaN
dtype: float64
```

In [88]:

```
print(a4.mode())
```

```
   MonthYear  Time index         Country  StoreID             City  Dept
_ID  \
0    01.2017         1.0          France  12227.0      Aalborg (I)
1.0
1    02.2017         2.0         Germany  15552.0     Aalborg (II)
2.0
2    03.2017         3.0  United Kingdom  16927.0        Amsterdam
3.0
3    04.2017         4.0             NaN  17647.0          Antwerp
4.0
4    05.2017         5.0             NaN  18808.0    Barcelona (I)
5.0
5    06.2017         6.0             NaN  19000.0   Barcelona (II)
6.0
6    10.2016         7.0             NaN  19340.0       Berlin (I)
7.0
7    11.2016         8.0             NaN  19769.0      Berlin (II)
8.0
8    12.2016         9.0             NaN  20166.0           Bilbao
```

In [89]:

```
print(a4.describe())
```

```
        Time index       StoreID      Dept_ID   HoursLease   Sales units  \
count  7650.000000   7650.000000  7650.000000  7650.000000  7.650000e+03
mean      5.000000  61995.220000     9.470588    22.036078  1.076471e+06
std       2.582158  29924.581631     5.337429   133.299513  1.728113e+06
min       1.000000  12227.000000     1.000000     0.000000  0.000000e+00
25%       3.000000  29650.000000     5.000000     0.000000  5.457125e+04
50%       5.000000  75400.500000     9.000000     0.000000  2.932300e+05
75%       7.000000  87703.000000    14.000000     0.000000  9.175075e+05
max       9.000000  98422.000000    18.000000  3984.000000  1.124296e+07

           Turnover  Customer
count  7.650000e+03       0.0
mean   3.721393e+06       NaN
std    6.003380e+06       NaN
min    0.000000e+00       NaN
25%    2.726798e+05       NaN
50%    9.319575e+05       NaN
75%    3.264432e+06       NaN
max    4.271739e+07       NaN
```

# b) Find sum(), cumsum(), count, min and max values

In [91]:

```
b4=a4[['StoreID','Dept_ID']]
print(b4.sum())
```

```
StoreID    474263433.0
Dept_ID        72450.0
dtype: float64
```

In [93]:

```
print(b4.cumsum())
```

```
          StoreID  Dept_ID
0         88253.0      1.0
1        176506.0      3.0
2        264759.0      6.0
3        353012.0     10.0
4        441265.0     15.0
...           ...      ...
7653  474144833.0  72388.0
7654  474174483.0  72404.0
7655  474204133.0  72415.0
7656  474233783.0  72432.0
7657  474263433.0  72450.0

[7658 rows x 2 columns]
```

In [94]:

```
print(b4.count())
```

```
StoreID    7650
Dept_ID    7650
dtype: int64
```

In [95]:

```
print(b4.min())
```

```
StoreID    12227.0
Dept_ID        1.0
dtype: float64
```

In [96]:

```
print(b4.max())
```

```
StoreID    98422.0
Dept_ID       18.0
dtype: float64
```

In [101]:

```
p=a4.fillna(value=7)
p
```

Out[101]:

| | MonthYear | Time index | Country | StoreID | City | Dept_ID | Dept. Name | HoursOwn | Hour |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 1.0 | Dry | 3184.764 | |
| **1** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 2.0 | Frozen | 1582.941 | |
| **2** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 3.0 | other | 47.205 | |
| **3** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 4.0 | Fish | 1623.852 | |
| **4** | 10.2016 | 1.0 | United Kingdom | 88253.0 | London (I) | 5.0 | Fruits & Vegetables | 1759.173 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **7653** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 12.0 | Checkout | 6322.323 | |
| **7654** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 16.0 | Customer Services | 4270.479 | |
| **7655** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 11.0 | Delivery | 0 | |
| **7656** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 17.0 | others | 2224.929 | |
| **7657** | 06.2017 | 9.0 | Sweden | 29650.0 | Gothenburg | 18.0 | all | 39652.2 | |

7658 rows × 14 columns

# c) Find covariance and correlation (spearman and pearsons)

In [102]:

```
x=p['Turnover']
y=p['StoreID']
print(cov(x,y))
```

```
[[3.60173739e+13 1.89289801e+09]
 [1.89289801e+09 8.98555466e+08]]
```

In [103]:

```
print(pearsonr(x,y))
```

```
(0.01052201088022699, 0.35722987709174453)
```

In [104]:

```
print(spearmanr(x,y))
```

```
SpearmanrResult(correlation=0.025346029206713392, pvalue=0.026553112421130
735)
```

In [ ]:

```
```