In [2]:
```
import pandas as pd
```

In [5]:
```
pd.Series(["blue","red","green"], name="colors")
```

Out[5]:
```
0     blue
1      red
2    green
Name: colors, dtype: object
```

In [8]:
```
s1= pd.Series(["blue","red","green"], name="colors")
s2= pd.Series(["b", "r", "g"], name ="first_letter")

df= pd.DataFrame([s1,s2]).T

df
```

Out[8]:

	colors	first_letter
0	blue	b
1	red	r
2	green	g

In [9]:
```
import pandas as pd

df = pd.read_csv("2019.csv")
df
```

Out[9]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298
...	...	...	...	...	...	...	...	...	...
151	152	Rwanda	3.334	0.359	0.711	0.614	0.555	0.217	0.411
152	153	Tanzania	3.231	0.476	0.885	0.499	0.417	0.276	0.147
153	154	Afghanistan	3.203	0.350	0.517	0.361	0.000	0.158	0.025
154	155	Central African Republic	3.083	0.026	0.000	0.105	0.225	0.235	0.035
155	156	South Sudan	2.853	0.306	0.575	0.295	0.010	0.202	0.091

156 rows × 9 columns

In [10]:
```
df.head()
```

Out[10]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298

In [11]:
```
df.shape
```

Out[11]:
```
(156, 9)
```

In [12]:
```
df.dtypes
```

Out[12]:
```
Overall rank                      int64
Country or region                object
Score                           float64
GDP per capita                  float64
Social support                  float64
Healthy life expectancy         float64
Freedom to make life choices    float64
Generosity                      float64
Perceptions of corruption       float64
dtype: object
```

In [14]:
```
df.describe()
```

Out[14]:

	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Percepti corrupt
count	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000	156.000
mean	78.500000	5.407096	0.905147	1.208814	0.725244	0.392571	0.184846	0.110
std	45.177428	1.113120	0.398389	0.299191	0.242124	0.143289	0.095254	0.094
min	1.000000	2.853000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	39.750000	4.544500	0.602750	1.055750	0.547750	0.308000	0.108750	0.047
50%	78.500000	5.379500	0.960000	1.271500	0.789000	0.417000	0.177500	0.085
75%	117.250000	6.184500	1.232500	1.452500	0.881750	0.507250	0.248250	0.141
max	156.000000	7.769000	1.684000	1.624000	1.141000	0.631000	0.566000	0.453

In [15]:

```
df.describe(include ="all")
```

Out[15]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosit
count	156.000000	156	156.000000	156.000000	156.000000	156.000000	156.000000	156.00000
unique	NaN	156	NaN	NaN	NaN	NaN	NaN	Nal
top	NaN	Finland	NaN	NaN	NaN	NaN	NaN	Nal
freq	NaN	1	NaN	NaN	NaN	NaN	NaN	Nal
mean	78.500000	NaN	5.407096	0.905147	1.208814	0.725244	0.392571	0.18484
std	45.177428	NaN	1.113120	0.398389	0.299191	0.242124	0.143289	0.09525
min	1.000000	NaN	2.853000	0.000000	0.000000	0.000000	0.000000	0.00000
25%	39.750000	NaN	4.544500	0.602750	1.055750	0.547750	0.308000	0.10875
50%	78.500000	NaN	5.379500	0.960000	1.271500	0.789000	0.417000	0.17750
75%	117.250000	NaN	6.184500	1.232500	1.452500	0.881750	0.507250	0.24825
max	156.000000	NaN	7.769000	1.684000	1.624000	1.141000	0.631000	0.56600

In [16]:

```
df["Score"]
```

Out[16]:

```
0      7.769
1      7.600
2      7.554
3      7.494
4      7.488
       ...
151    3.334
152    3.231
153    3.203
154    3.083
155    2.853
Name: Score, Length: 156, dtype: float64
```

In [18]:

```
df[["Score","Generosity"]] #selecting multiple series in the data frame
```

Out[18]:

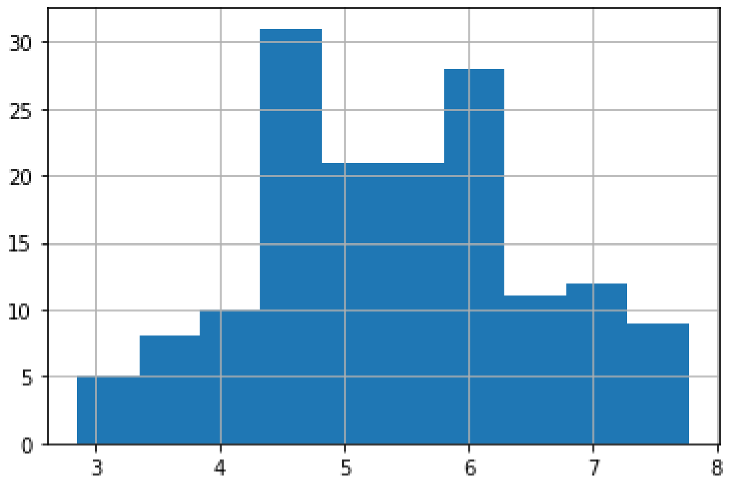
	Score	Generosity
0	7.769	0.153
1	7.600	0.252
2	7.554	0.271
3	7.494	0.354
4	7.488	0.322
...	...	...
151	3.334	0.217
152	3.231	0.276

	Score	Generosity
153	3.203	0.158
154	3.083	0.235
155	2.853	0.202

156 rows × 2 columns

In [20]:
```
df["Score"].hist()   #pandas use matplotlib for the histogram
```

Out[20]:
```
<AxesSubplot:>
```



In [21]:
```
score = df["Score"]
```

In [23]:
```
score.max() #Return a Series/DataFrame with absolute numeric value of each element.
```

Out[23]:
```
7.769
```

In [24]:
```
score.min()
```

Out[24]:
```
2.853
```

In [29]:
```
df = df.sort_values( by = "Healthy life expectancy", ascending = False) #sorting dat
```

In [30]:
```
df
```

Out[30]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
33	34	Singapore	6.262	1.572	1.463	1.141	0.556	0.271	0.453
75	76	Hong Kong	5.430	1.438	1.277	1.122	0.440	0.258	0.287
57	58	Japan	5.886	1.327	1.419	1.088	0.445	0.069	0.140

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
29	30	Spain	6.354	1.286	1.484	1.062	0.362	0.153	0.079
5	6	Switzerland	7.480	1.452	1.526	1.052	0.572	0.263	0.343
...	...	...	...	...	...	...	...	...	...
98	99	Ivory Coast	4.944	0.569	0.808	0.232	0.352	0.154	0.090
131	132	Chad	4.350	0.350	0.766	0.192	0.174	0.198	0.078
143	144	Lesotho	3.802	0.489	1.169	0.168	0.359	0.107	0.093
154	155	Central African Republic	3.083	0.026	0.000	0.105	0.225	0.235	0.035
134	135	Swaziland	4.212	0.811	1.149	0.000	0.313	0.074	0.135

156 rows × 9 columns

In [32]:
```
df = df.sort_index() #coming to the original order
```

In [33]:
```
df
```

Out[33]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298
...	...	...	...	...	...	...	...	...	...
151	152	Rwanda	3.334	0.359	0.711	0.614	0.555	0.217	0.411
152	153	Tanzania	3.231	0.476	0.885	0.499	0.417	0.276	0.147
153	154	Afghanistan	3.203	0.350	0.517	0.361	0.000	0.158	0.025
154	155	Central African Republic	3.083	0.026	0.000	0.105	0.225	0.235	0.035
155	156	South Sudan	2.853	0.306	0.575	0.295	0.010	0.202	0.091

156 rows × 9 columns

In [35]:
```
#data sorting
# create a boolean mask
```

```
#sorting values greater than score 7
greater_than_seven_bool_mask = df["Score"] >= 7
```

In [36]:

```
df[greater_than_seven_bool_mask]
```

Out[36]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298
5	6	Switzerland	7.480	1.452	1.526	1.052	0.572	0.263	0.343
6	7	Sweden	7.343	1.387	1.487	1.009	0.574	0.267	0.373
7	8	New Zealand	7.307	1.303	1.557	1.026	0.585	0.330	0.380
8	9	Canada	7.278	1.365	1.505	1.039	0.584	0.285	0.308
9	10	Austria	7.246	1.376	1.475	1.016	0.532	0.244	0.226
10	11	Australia	7.228	1.372	1.548	1.036	0.557	0.332	0.290
11	12	Costa Rica	7.167	1.034	1.441	0.963	0.558	0.144	0.093
12	13	Israel	7.139	1.276	1.455	1.029	0.371	0.261	0.082
13	14	Luxembourg	7.090	1.609	1.479	1.012	0.526	0.194	0.316
14	15	United Kingdom	7.054	1.333	1.538	0.996	0.450	0.348	0.278
15	16	Ireland	7.021	1.499	1.553	0.999	0.516	0.298	0.310

In [37]:

```
#Finding score which is greater than or equal to 7 and Healthy life happiness less t
df["Score"] >= 7
```

Out[37]:

```
0       True
1       True
2       True
3       True
4       True
       ...
151    False
152    False
153    False
154    False
155    False
Name: Score, Length: 156, dtype: bool
```

In [38]:

```
df["Healthy life expectancy"] < 1
```

Out[38]:

```
0       True
```

```
1       True
2      False
3      False
4       True
       ...
151     True
152     True
153     True
154     True
155     True
Name: Healthy life expectancy, Length: 156, dtype: bool
```

In [40]:

```
#Finding score which is greater than or equal to 7 and Healthy life happiness less t
score_greater_7_healthy_life_less_one =(df["Score"] >= 7) & (df["Healthy life expect
```

In [41]:

```
df[score_greater_7_healthy_life_less_one] #printing the countries
```

Out[41]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298
11	12	Costa Rica	7.167	1.034	1.441	0.963	0.558	0.144	0.093
14	15	United Kingdom	7.054	1.333	1.538	0.996	0.450	0.348	0.278
15	16	Ireland	7.021	1.499	1.553	0.999	0.516	0.298	0.310

In [43]:

```
#Alternative method Which is prettey much simple.
df[  (df["Score"] >= 7)
      & (df["Healthy life expectancy"] < 1 )
]
```

Out[43]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298
11	12	Costa Rica	7.167	1.034	1.441	0.963	0.558	0.144	0.093
14	15	United Kingdom	7.054	1.333	1.538	0.996	0.450	0.348	0.278
15	16	Ireland	7.021	1.499	1.553	0.999	0.516	0.298	0.310

In [44]:

```
#modifying series and collumn or adding something
df["Score"] * 10
```

Out[44]:
```
0      77.69
1      76.00
2      75.54
3      74.94
4      74.88
       ...
151    33.34
152    32.31
153    32.03
154    30.83
155    28.53
Name: Score, Length: 156, dtype: float64
```

In [45]:
```
#but hear we are adding a new collumn or assinging a new collumn a name
df["score_times_ten"] = df["Score"] * 10
```

In [46]:
```
df.head()
```

Out[46]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	sco
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393	
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410	
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341	
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118	
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298	

In [48]:
```
#multiplying two collumn to create new collumn
df["Generositydp"] = df["GDP per capita"] * df["Generosity"]
```

In [49]:
```
df.head()
```

Out[49]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	sco
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393	
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410	
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341	
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118	
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298	

In [50]:
```
#complicated function with maps
df["Score"]
```

Out[50]:
```
0      7.769
1      7.600
2      7.554
3      7.494
4      7.488
       ...  
151    3.334
152    3.231
153    3.203
154    3.083
155    2.853
Name: Score, Length: 156, dtype: float64
```

In [62]:
```
#complicated function with maps
def map_store_to_category(score):
    if score >=7:
         return "High score"
    elif score <4:
         return " Low score"
    else:
        return "Medium score"
```

In [61]:
```
#complicated function with maps
df["Score"].map(map_store_to_category)
```

Out[61]:
```
0      High score
1      High score
2      High score
3      High score
4      High score
          ...    
151     Low score
152     Low score
153     Low score
154     Low score
155     Low score
Name: Score, Length: 156, dtype: object
```

In [60]:
```
#complicated function with maps with value count
df["Score"].map(map_store_to_category).value_counts()
```

Out[60]:
```
Medium score    124
High score       16
 Low score       16
Name: Score, dtype: int64
```

In [63]:
```
df["score_category"] = df["Score"].map(map_store_to_category)
```

In [64]:
```
df.head()
```

Out[64]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	sco

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	sco
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393	
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410	
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341	
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118	
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298	

In [65]:
```
#lambda syntax in python
df[df["Country or region"].map(lambda x:x[0] == "S")]
```

Out[65]:

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	sc
5	6	Switzerland	7.480	1.452	1.526	1.052	0.572	0.263	0.343	
6	7	Sweden	7.343	1.387	1.487	1.009	0.574	0.267	0.373	
27	28	Saudi Arabia	6.375	1.403	1.357	0.795	0.439	0.080	0.132	
29	30	Spain	6.354	1.286	1.484	1.062	0.362	0.153	0.079	
33	34	Singapore	6.262	1.572	1.463	1.141	0.556	0.271	0.453	
37	38	Slovakia	6.198	1.246	1.504	0.881	0.334	0.121	0.014	
43	44	Slovenia	6.118	1.258	1.523	0.953	0.564	0.144	0.057	
53	54	South Korea	5.895	1.301	1.219	1.036	0.159	0.175	0.056	
69	70	Serbia	5.603	1.004	1.383	0.854	0.282	0.137	0.039	
105	106	South Africa	4.722	0.960	1.351	0.469	0.389	0.130	0.055	
110	111	Senegal	4.681	0.450	1.134	0.571	0.292	0.153	0.072	
111	112	Somalia	4.668	0.000	0.698	0.268	0.559	0.243	0.270	
128	129	Sierra Leone	4.374	0.268	0.841	0.242	0.309	0.252	0.045	
129	130	Sri Lanka	4.366	0.949	1.265	0.831	0.470	0.244	0.047	
134	135	Swaziland	4.212	0.811	1.149	0.000	0.313	0.074	0.135	
148	149	Syria	3.462	0.619	0.378	0.440	0.013	0.331	0.141	
155	156	South Sudan	2.853	0.306	0.575	0.295	0.010	0.202	0.091	

In [66]:
```
#rounded score
df["Score"].round()
```

Out[66]:
```
0      8.0
1      8.0
2      8.0
3      7.0
4      7.0
      ...
151    3.0
152    3.0
153    3.0
154    3.0
155    3.0
Name: Score, Length: 156, dtype: float64
```

In [67]:
```
#rounded score value count
df["Score"].round().value_counts()
```

Out[67]:
```
6.0    49
5.0    48
4.0    26
7.0    21
3.0     9
8.0     3
Name: Score, dtype: int64
```

In [69]:
```
#check pivot table function
#Group by data
df["round_score"] = df["Score"].round()
```

In [74]:
```
gb = df.groupby("round_score")
```

In [75]:
```
gb
```

Out[75]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001AA0E6E42E0>
```

In [76]:
```
gb.groups
```

Out[76]:
```
{3.0: [147, 148, 149, 150, 151, 152, 153, 154, 155], 4.0: [121, 122, 123, 124, 125,
126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
143, 144, 145, 146], 5.0: [73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 8
7, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 10
6, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120], 6.0: [24,
25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66,
67, 68, 69, 70, 71, 72], 7.0: [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
18, 19, 20, 21, 22, 23], 8.0: [0, 1, 2]}
```

In [78]:
```
#The average score for all the countries
gb.mean()
```

Out[78]:

	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	s

round_score	Overall rank	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
round_score								
3.0	152.0	3.271556	0.406111	0.659333	0.423333	0.251222	0.196667	0.124000
4.0	134.5	4.179923	0.489269	0.978154	0.468885	0.301885	0.202038	0.085115
5.0	97.5	4.985292	0.795792	1.123104	0.661021	0.362208	0.175562	0.085625
6.0	49.0	6.009041	1.102204	1.373898	0.851000	0.430020	0.153041	0.082204
7.0	14.0	7.082143	1.352857	1.487476	0.985667	0.518238	0.248143	0.221095
8.0	2.0	7.641000	1.403667	1.580667	1.003333	0.597000	0.225333	0.381333

In [79]:
```
gb.mean()["GDP per capita"]
```

Out[79]:
```
round_score
3.0    0.406111
4.0    0.489269
5.0    0.795792
6.0    1.102204
7.0    1.352857
8.0    1.403667
Name: GDP per capita, dtype: float64
```

In [80]:
```
df.to_csv("2019-updated.csv", index = False)
```

In [ ]:

In [ ]: