Aniruddha Talekar

Pandas Series

In [2]:
```
#import pandas and numpy libraries
import pandas as pd
import numpy as np
```

In [3]:
```
#create an empty series
empty_series = pd.Series()
```

```
C:\Users\ANI\AppData\Local\Temp/ipykernel_27792/1892664345.py:1: DeprecationWarning: The default dtype for empty Series will be 'object' instead of 'float64' in a future version. Specify a dtype e
xplicitly to silence this warning.
  empty_series = pd.Series()
```

In [4]:
```
print(empty_series)
```

```
Series([], dtype: float64)
```

In [5]:
```
#create two series
s1 = pd.Series([1,10,15,23,7])
s2 = pd.Series([4,15,160,65,37])
```

In [6]:
```
#perform addition, subtraction, multiplication and division operations on the series.
s_add = s1 + s2
s_sub = s1 - s2
s_mul = s1 * s2
s_div = s1 / s2
print("Addition of 2 series =","\n",s_add)
print("Subtraction of 2 series = ","\n",s_sub)
print("Multiplication of 2 series = ","\n",s_mul)
print("Division of 2 series = ","\n",s_div)
```

```
Addition of 2 series =
 0      5
1     25
2    175
3     88
4     44
dtype: int64
Subtraction of 2 series =
 0     -3
1     -5
2   -145
3    -42
4    -30
dtype: int64
Multiplication of 2 series =
 0       4
1     150
2    2400
3    1495
4     259
dtype: int64
Division of 2 series =
 0    0.250000
```

```
1    0.666667
2    0.093750
3    0.353846
4    0.189189
dtype: float64
```

In [7]:
```
#create two series
ps1 = pd.Series([3, 5.5, 15, 30, 76.2])
ps2 = pd.Series([45, 5.5, 30, 30, 7.62])
print("Ps1: ","\n",ps1,"\n","Ps2: ","\n", ps2)
```

```
Ps1:
 0     3.0
1     5.5
2    15.0
3    30.0
4    76.2
dtype: float64
 Ps2:
 0    45.00
1     5.50
2    30.00
3    30.00
4     7.62
dtype: float64
```

In [8]:
```
#check if the elements are same in both the series
print(ps1 == ps2)
```

```
0    False
1     True
2    False
3     True
4    False
dtype: bool
```

In [9]:
```
series = pd.Series([1,2,'Python', 2.0, True, 100])
```

In [10]:
```
series
```

Out[10]:
```
0         1
1         2
2    Python
3       2.0
4      True
5       100
dtype: object
```

In [11]:
```
#import and read the csv file from the local drive.
#squeeze is "True" so that series is craeted instead of dataframe.
bs = pd.read_csv("batsman_runs_series.csv", squeeze=True)
```

In [12]:
```
#display the series
bs
```

Out[12]:

	batter	batsman_run
0	A Ashish Reddy	280

	batter	batsman_run
1	A Badoni	161
2	A Chandila	4
3	A Chopra	53
4	A Choudhary	25
...	...	...
600	Yash Dayal	0
601	Yashpal Singh	47
602	Younis Khan	3
603	Yuvraj Singh	2754
604	Z Khan	117

605 rows × 2 columns

In [13]:
```
#get the top 10 run scorers
top10 = bs.sort_values(by="batsman_run",ascending=False)
```

In [14]:
```
top10.head(10)
```

Out[14]:

	batter	batsman_run
569	V Kohli	6634
462	S Dhawan	6244
130	DA Warner	5883
430	RG Sharma	5881
493	SK Raina	5536
27	AB de Villiers	5181
108	CH Gayle	4997
339	MS Dhoni	4978
452	RV Uthappa	4954
256	KD Karthik	4377

In [15]:
```
#check how many times the runs scored is more than 3000.
above_3k = bs[bs["batsman_run"] > 3000].size
```

In [16]:
```
above_3k
```

Out[16]:
```
40
```

In [17]:
```
#check how many times the runs scored is above the mean value of runs scored.
above_mean = bs[bs["batsman_run"] > bs["batsman_run"].mean()].size
```

In [18]:
```
above_mean
```

Out[18]:
```
256
```

In [19]:
```
#import and read the csv file from the local drive.
#squeeze is "True" so that series is craeted instead of dataframe.
items =pd.read_csv("items.csv",index_col="item_name", squeeze=True)
```

In [20]:
```
#display the first 5 rows
items.head()
```

Out[20]:
```
item_name
Chips and Fresh Tomato Salsa              $2.39
Izze                                      $3.39
Nantucket Nectar                          $3.39
Chips and Tomatillo-Green Chili Salsa     $2.39
Chicken Bowl                             $16.98
Name: item_price, dtype: object
```

In [21]:
```
#check  how  many rows have  null values
items.isnull().sum()
```

Out[21]:
```
50
```

In [22]:
```
items.describe()
```

Out[22]:
```
count       4572
unique        78
top       $8.75
freq         724
Name: item_price, dtype: object
```

In [119…
```
#name  the  col as item price
list_of_items = items.reset_index(name="item_price")
```

In [120…
```
#chcek the  first 10 rows.
list_of_items.head(10)
```

Out[120…

	item_name	item_price
0	Chips and Fresh Tomato Salsa	$2.39
1	Izze	$3.39
2	Nantucket Nectar	$3.39
3	Chips and Tomatillo-Green Chili Salsa	$2.39
4	Chicken Bowl	$16.98
5	Chicken Bowl	$10.98
6	Side of Chips	NaN
7	Steak Burrito	$11.75
8	Steak Soft Tacos	NaN

	item_name	item_price
9	Steak Burrito	$9.25

In [121…
```
#remove the  "$"  from  the item   price columnn.
list_of_items["item_price"] = list_of_items["item_price"].replace({"\$":""}, regex = True)
```

In [122…
```
list_of_items.head(10)
```

Out[122…

	item_name	item_price
0	Chips and Fresh Tomato Salsa	2.39
1	Izze	3.39
2	Nantucket Nectar	3.39
3	Chips and Tomatillo-Green Chili Salsa	2.39
4	Chicken Bowl	16.98
5	Chicken Bowl	10.98
6	Side of Chips	NaN
7	Steak Burrito	11.75
8	Steak Soft Tacos	NaN
9	Steak Burrito	9.25

In [123…
```
#check the data  type  of  the  entities.
list_of_items.dtypes
```

Out[123…
```
item_name     object
item_price    object
dtype: object
```

In [124…
```
#connvert the dtype of item price  from  obj   i.e  string  to  float   i.e  numeric
list_of_items["item_price"] = pd.to_numeric(list_of_items["item_price"])
```

In [125…
```
#check  the  dtypes again
list_of_items.dtypes
```

Out[125…
```
item_name      object
item_price    float64
dtype: object
```

In [127…
```
list_of_items.head(10)
```

Out[127…

	item_name	item_price
0	Chips and Fresh Tomato Salsa	2.39
1	Izze	3.39
2	Nantucket Nectar	3.39
3	Chips and Tomatillo-Green Chili Salsa	2.39

	item_name	item_price
4	Chicken Bowl	16.98
5	Chicken Bowl	10.98
6	Side of Chips	NaN
7	Steak Burrito	11.75
8	Steak Soft Tacos	NaN
9	Steak Burrito	9.25

In [131…
```
#fill the  null  values  with the  mean.
list_of_items=list_of_items.fillna(list_of_items["item_price"].mean())
```

In [132…
```
list_of_items.head(10)
```

Out[132…

	item_name	item_price
0	Chips and Fresh Tomato Salsa	2.390000
1	Izze	3.390000
2	Nantucket Nectar	3.390000
3	Chips and Tomatillo-Green Chili Salsa	2.390000
4	Chicken Bowl	16.980000
5	Chicken Bowl	10.980000
6	Side of Chips	7.463031
7	Steak Burrito	11.750000
8	Steak Soft Tacos	7.463031
9	Steak Burrito	9.250000

In [133…
```
list_of_items.dtypes
```

Out[133…
```
item_name      object
item_price    float64
dtype: object
```

In [140…
```
list_of_items.head(10)
```

Out[140…

	item_name	item_price
0	Chips and Fresh Tomato Salsa	2.390000
1	Izze	3.390000
2	Nantucket Nectar	3.390000
3	Chips and Tomatillo-Green Chili Salsa	2.390000
4	Chicken Bowl	16.980000
5	Chicken Bowl	10.980000
6	Side of Chips	7.463031

	item_name	item_price
7	Steak Burrito	11.750000
8	Steak Soft Tacos	7.463031
9	Steak Burrito	9.250000

In [196…
```
#since we  ahve removed the  "$",  convert  the USD to  INR
list_of_items["item_price"]   = list_of_items["item_price"] * 83
```

In [206…
```
list_of_items.head(10)
```

Out[206…

	item_name	item_price
0	Chips and Fresh Tomato Salsa	198.370000
1	Izze	281.370000
2	Nantucket Nectar	281.370000
3	Chips and Tomatillo-Green Chili Salsa	198.370000
4	Chicken Bowl	1409.340000
5	Chicken Bowl	911.340000
6	Side of Chips	619.431614
7	Steak Burrito	975.250000
8	Steak Soft Tacos	619.431614
9	Steak Burrito	767.750000

In [197…
```
#check   the  mean  again  after  the null values  are  replaced with  mean.
#not much  impact on  the  mean value  since  there   are 4622 entries   in  total  and   only  50 had null values.
list_of_items["item_price"].mean()
```

Out[197…
```
619.4316141732339
```

In [198…
```
#calculate  the 30th percentile.
np.percentile(list_of_items["item_price"], 30)
```

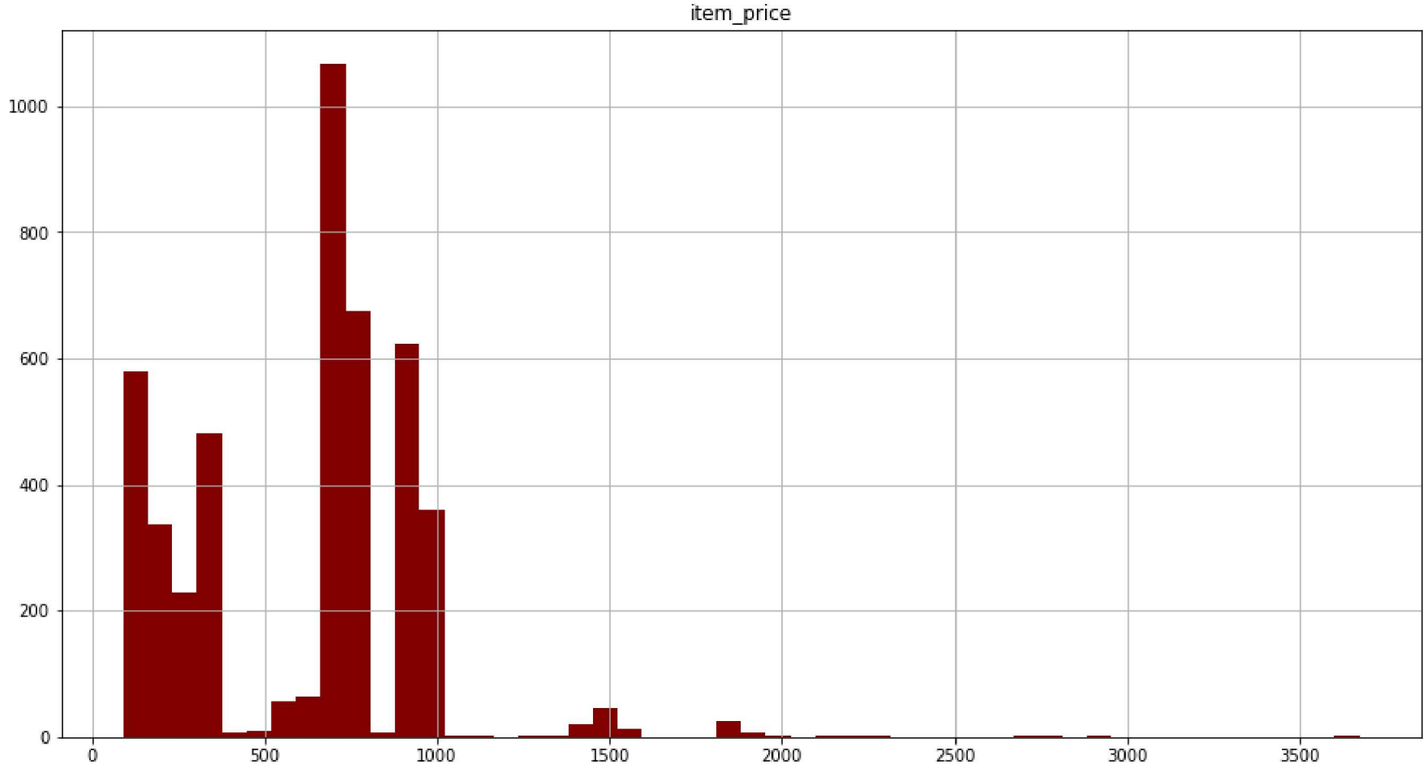
Out[198…
```
369.35
```

In [199…
```
#calculate the 6th  percentile.
np.percentile(list_of_items["item_price"], 6)
```

Out[199…
```
103.75
```

In [200…
```
#import  the matplotlib library
import matplotlib.pyplot as plt
```

In [207…
```
#create   an  histogram
list_of_items.hist(figsize=(15,8), bins=50, color = "maroon")
plt.show()
```



In [202…

```
#find  the  max  value  of  item price.
list_of_items["item_price"].max()
```

Out[202…

```
3672.75
```

In [205…

```
#find how many items have price between 1000 and 2000
list_of_items[list_of_items["item_price"].between(1000,2000)].count()
```

Out[205…

```
item_name     116
item_price    116
dtype: int64
```