pandas

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

In [3]:
```
pd.__version__
```

Out[3]:
```
'2.2.2'
```

In [4]:
```
store = pd.read_csv(r'C:\Users\jayes\OneDrive\Desktop\NareshIT\19_mar\19th, 20th - Pandas\19th, 20th - Pandas\Sample
```

In [5]:
```
store
```

Out[5]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segm
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consur
1	Office Supplies	Naperville	United States	Phillina Ober	GBC	04-01-2020	US-2020-112326	60540	GBC Standard Plastic Binding Systems Combs	Central	Ho Of
2	Office Supplies	Naperville	United States	Phillina Ober	Avery	04-01-2020	US-2020-112326	60540	Avery 508	Central	Ho Of
3	Office Supplies	Naperville	United States	Phillina Ober	SAFCO	04-01-2020	US-2020-112326	60540	SAFCO Boltless Steel Shelving	Central	Ho Of
4	Office Supplies	Philadelphia	United States	Mick Brown	Avery	05-01-2020	US-2020-141817	19143	Avery Hi-Liter EverBold Pen Style Fluorescent ...	East	Consur
...	...	...	...	...	...	...	...	...	...	...	
10189	Office Supplies	New York City	United States	Patrick O'Donnell	Wilson Jones	30-12-2023	US-2023-143259	10009	Wilson Jones Legal Size Ring Binders	East	Consur
10190	Office Supplies	Fairfield	United States	Erica Bern	GBC	30-12-2023	US-2023-115427	94533	GBC Binding covers	West	Corpor

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segm
10191	Office Supplies	Loveland	United States	Jill Matthias	Other	30-12-2023	US-2023-156720	80538	Bagged Rubber Bands	West	Consur
10192	Technology	New York City	United States	Patrick O'Donnell	Other	30-12-2023	US-2023-143259	10009	Gear Head AU3700S Headset	East	Consur
10193	Office Supplies	Charlottetown	Canada	Harry Olson	Wilson Jones	30-12-2023	CA-2023-143500	C0A	Wilson Jones Impact Binders	East	Consur

10194 rows × 19 columns

In [6]:
```
id(store)
```

Out[6]:
```
2118257331568
```

In [7]:
```
len(store)
```

Out[7]:
```
10194
```

In [8]:
```
store.shape
```

Out[8]:
```
(10194, 19)
```

In [9]:
```
store.columns
```

Out[9]:
```
Index(['Category', 'City', 'Country/Region', 'Customer Name', 'Manufacturer',
       'Order Date', 'Order ID', 'Postal Code', 'Product Name', 'Region',
       'Segment', 'Ship Date', 'Ship Mode', 'State/Province', 'Sub-Category',
       'Discount', 'Profit', 'Quantity', 'Sales'],
      dtype='object')
```

In [10]:
```
len(store.columns)
```

Out[10]: 19

In [11]:
```
store.isnull()
```

Out[11]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	Ship Date	Sh Mo
0	False	False	False	False	False	False	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	False	False	False	False	False	False	Fa
...	...	...	...	...	...	...	...	...	...	...	...	...	
10189	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10190	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10191	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10192	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10193	False	False	False	False	False	False	False	False	False	False	False	False	Fa

10194 rows × 19 columns

In [12]:
```
store.isnull().sum()
```

Out[12]:
```
Category          0
City              0
Country/Region    0
Customer Name     0
Manufacturer      0
Order Date        0
Order ID          0
Postal Code       0
Product Name      0
Region            0
Segment           0
Ship Date         0
Ship Mode         0
State/Province    0
Sub-Category      0
Discount          0
Profit            0
Quantity          0
Sales             0
dtype: int64
```

In [13]:
```
store[:]
```

Out[13]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segm
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consur
1	Office Supplies	Naperville	United States	Phillina Ober	GBC	04-01-2020	US-2020-112326	60540	GBC Standard Plastic Binding Systems Combs	Central	Ho Of
2	Office Supplies	Naperville	United States	Phillina Ober	Avery	04-01-2020	US-2020-112326	60540	Avery 508	Central	Ho Of
3	Office Supplies	Naperville	United States	Phillina Ober	SAFCO	04-01-2020	US-2020-112326	60540	SAFCO Boltless Steel Shelving	Central	Ho Of
4	Office Supplies	Philadelphia	United States	Mick Brown	Avery	05-01-2020	US-2020-141817	19143	Avery Hi-Liter EverBold Pen Style Fluorescent ...	East	Consur
...	...	...	...	...	...	...	...	...	...	...	
10189	Office Supplies	New York City	United States	Patrick O'Donnell	Wilson Jones	30-12-2023	US-2023-143259	10009	Wilson Jones Legal Size Ring Binders	East	Consur
10190	Office Supplies	Fairfield	United States	Erica Bern	GBC	30-12-2023	US-2023-115427	94533	GBC Binding covers	West	Corpor

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segm
10191	Office Supplies	Loveland	United States	Jill Matthias	Other	30-12-2023	US-2023-156720	80538	Bagged Rubber Bands	West	Consur
10192	Technology	New York City	United States	Patrick O'Donnell	Other	30-12-2023	US-2023-143259	10009	Gear Head AU3700S Headset	East	Consur
10193	Office Supplies	Charlottetown	Canada	Harry Olson	Wilson Jones	30-12-2023	CA-2023-143500	C0A	Wilson Jones Impact Binders	East	Consur

10194 rows × 19 columns

In [14]:
```
store[0:10]
```

Out[14]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	Shi Dat
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consumer	07 01 202
1	Office Supplies	Naperville	United States	Phillina Ober	GBC	04-01-2020	US-2020-112326	60540	GBC Standard Plastic Binding Systems Combs	Central	Home Office	08 01 202
2	Office Supplies	Naperville	United States	Phillina Ober	Avery	04-01-2020	US-2020-112326	60540	Avery 508	Central	Home Office	08 01 202
3	Office Supplies	Naperville	United States	Phillina Ober	SAFCO	04-01-2020	US-2020-112326	60540	SAFCO Boltless Steel Shelving	Central	Home Office	08 01 202
4	Office Supplies	Philadelphia	United States	Mick Brown	Avery	05-01-2020	US-2020-141817	19143	Avery Hi-Liter EverBold Pen Style Fluorescent ...	East	Consumer	12 01 202
5	Furniture	Henderson	United States	Maria Etezadi	Global	06-01-2020	US-2020-167199	42420	Global Deluxe High-Back Manager's Chair	South	Home Office	10 01 202
6	Office Supplies	Henderson	United States	Maria Etezadi	Rogers	06-01-2020	US-2020-167199	42420	Rogers Handheld Barrel Pencil Sharpener	South	Home Office	10 01 202

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	Shi Dat
7	Office Supplies	Athens	United States	Jack O'Briant	Dixon	06-01-2020	US-2020-106054	30605	Dixon Prang Watercolor Pencils, 10-Color Set w...	South	Corporate	0 0 202
8	Office Supplies	Henderson	United States	Maria Etezadi	Ibico	06-01-2020	US-2020-167199	42420	Ibico Hi-Tech Manual Binding System	South	Home Office	1 0 202
9	Office Supplies	Henderson	United States	Maria Etezadi	Alliance	06-01-2020	US-2020-167199	42420	Alliance Super-Size Bands, Assorted Sizes	South	Home Office	1 0 202

In [15]:
```
store[0:20:5]
```

Out[15]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	S D
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consumer	2
5	Furniture	Henderson	United States	Maria Etezadi	Global	06-01-2020	US-2020-167199	42420	Global Deluxe High-Back Manager's Chair	South	Home Office	2
10	Office Supplies	Henderson	United States	Maria Etezadi	Southworth	06-01-2020	US-2020-167199	42420	Southworth 25% Cotton Granite Paper & Envelopes	South	Home Office	2
15	Office Supplies	Huntsville	United States	Vivek Sundaresam	Acco	07-01-2020	US-2020-105417	77340	Acco Four Pocket Poly Ring Binder with Label H...	Central	Consumer	2

In [16]:
```
store.head()
```

Out[16]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	Shi Dat
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consumer	07 01 202
1	Office Supplies	Naperville	United States	Phillina Ober	GBC	04-01-2020	US-2020-112326	60540	GBC Standard Plastic Binding Systems Combs	Central	Home Office	08 01 202
2	Office Supplies	Naperville	United States	Phillina Ober	Avery	04-01-2020	US-2020-112326	60540	Avery 508	Central	Home Office	08 01 202
3	Office Supplies	Naperville	United States	Phillina Ober	SAFCO	04-01-2020	US-2020-112326	60540	SAFCO Boltless Steel Shelving	Central	Home Office	08 01 202
4	Office Supplies	Philadelphia	United States	Mick Brown	Avery	05-01-2020	US-2020-141817	19143	Avery Hi-Liter EverBold Pen Style Fluorescent ...	East	Consumer	12 01 202

In [17]:
```
store.head(3)
```

Out[17]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	Ship Date
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consumer	07-01-2020
1	Office Supplies	Naperville	United States	Phillina Ober	GBC	04-01-2020	US-2020-112326	60540	GBC Standard Plastic Binding Systems Combs	Central	Home Office	08-01-2020
2	Office Supplies	Naperville	United States	Phillina Ober	Avery	04-01-2020	US-2020-112326	60540	Avery 508	Central	Home Office	08-01-2020

In [18]:
```
store.tail()
```

Out[18]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segmen
10189	Office Supplies	New York City	United States	Patrick O'Donnell	Wilson Jones	30-12-2023	US-2023-143259	10009	Wilson Jones Legal Size Ring Binders	East	Consume
10190	Office Supplies	Fairfield	United States	Erica Bern	GBC	30-12-2023	US-2023-115427	94533	GBC Binding covers	West	Corporate
10191	Office Supplies	Loveland	United States	Jill Matthias	Other	30-12-2023	US-2023-156720	80538	Bagged Rubber Bands	West	Consume
10192	Technology	New York City	United States	Patrick O'Donnell	Other	30-12-2023	US-2023-143259	10009	Gear Head AU3700S Headset	East	Consume
10193	Office Supplies	Charlottetown	Canada	Harry Olson	Wilson Jones	30-12-2023	CA-2023-143500	C0A	Wilson Jones Impact Binders	East	Consume

In [19]:
```
store.isna()                # or store.isnull
```

Out[19]:

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segment	Ship Date	Sl Mo
0	False	False	False	False	False	False	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	False	False	False	False	False	False	Fa
...	...	...	...	...	...	...	...	...	...	...	...	...	
10189	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10190	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10191	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10192	False	False	False	False	False	False	False	False	False	False	False	False	Fa
10193	False	False	False	False	False	False	False	False	False	False	False	False	Fa

10194 rows × 19 columns

introduce statical concept in pandas

In [230…

```
store.describe()
```

Out[230…

	Discount	Profit	Quantity	Sales
count	10194.000000	10194.000000	10194.000000	10194.000000
mean	0.155385	28.673417	3.791838	228.225854
std	0.206249	232.465115	2.228317	619.906839
min	0.000000	-6599.978000	1.000000	0.444000
25%	0.000000	1.760800	2.000000	17.220000
50%	0.200000	8.690000	3.000000	53.910000
75%	0.200000	29.297925	5.000000	209.500000
max	0.800000	8399.976000	14.000000	22638.480000

In [22]:
```
store.columns
```

Out[22]:
```
Index(['Category', 'City', 'Country/Region', 'Customer Name', 'Manufacturer',
       'Order Date', 'Order ID', 'Postal Code', 'Product Name', 'Region',
       'Segment', 'Ship Date', 'Ship Mode', 'State/Province', 'Sub-Category',
       'Discount', 'Profit', 'Quantity', 'Sales'],
      dtype='object')
```

In [192…
```
store['Category']
```

Out[192…
```
0        Office Supplies
1        Office Supplies
2        Office Supplies
3        Office Supplies
4        Office Supplies
             ...
10189    Office Supplies
10190    Office Supplies
10191    Office Supplies
10192         Technology
10193    Office Supplies
Name: Category, Length: 10194, dtype: object
```

In [197…
```
store[['Category','City']]
```

Out[197…

	Category	City
0	Office Supplies	Houston
1	Office Supplies	Naperville
2	Office Supplies	Naperville
3	Office Supplies	Naperville
4	Office Supplies	Philadelphia
…	…	…
10189	Office Supplies	New York City
10190	Office Supplies	Fairfield
10191	Office Supplies	Loveland
10192	Technology	New York City
10193	Office Supplies	Charlottetown

10194 rows × 2 columns

dividing dataset into categorical and numerical data

In [199…
```
store.columns
```

Out[199…
```
Index(['Category', 'City', 'Country/Region', 'Customer Name', 'Manufacturer',
       'Order Date', 'Order ID', 'Postal Code', 'Product Name', 'Region',
       'Segment', 'Ship Date', 'Ship Mode', 'State/Province', 'Sub-Category',
       'Discount', 'Profit', 'Quantity', 'Sales'],
      dtype='object')
```

In [209…
```
store_cat=store[['Category', 'City', 'Country/Region', 'Customer Name', 'Manufacturer',
       'Order Date', 'Order ID', 'Postal Code', 'Product Name', 'Region',
       'Segment', 'Ship Date', 'Ship Mode', 'State/Province', 'Sub-Category']]
store_cat
```

Out[209...

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segm
0	Office Supplies	Houston	United States	Darren Powers	Message Book	03-01-2020	US-2020-103800	77015	Message Book, Wirebound, Four 5 1/2" X 4" Form...	Central	Consur
1	Office Supplies	Naperville	United States	Phillina Ober	GBC	04-01-2020	US-2020-112326	60540	GBC Standard Plastic Binding Systems Combs	Central	Ho Of
2	Office Supplies	Naperville	United States	Phillina Ober	Avery	04-01-2020	US-2020-112326	60540	Avery 508	Central	Ho Of
3	Office Supplies	Naperville	United States	Phillina Ober	SAFCO	04-01-2020	US-2020-112326	60540	SAFCO Boltless Steel Shelving	Central	Ho Of
4	Office Supplies	Philadelphia	United States	Mick Brown	Avery	05-01-2020	US-2020-141817	19143	Avery Hi-Liter EverBold Pen Style Fluorescent ...	East	Consur
...	...	...	...	...	...	...	...	...	...	...	
10189	Office Supplies	New York City	United States	Patrick O'Donnell	Wilson Jones	30-12-2023	US-2023-143259	10009	Wilson Jones Legal Size Ring Binders	East	Consur
10190	Office Supplies	Fairfield	United States	Erica Bern	GBC	30-12-2023	US-2023-115427	94533	GBC Binding covers	West	Corpor

	Category	City	Country/Region	Customer Name	Manufacturer	Order Date	Order ID	Postal Code	Product Name	Region	Segm
10191	Office Supplies	Loveland	United States	Jill Matthias	Other	30-12-2023	US-2023-156720	80538	Bagged Rubber Bands	West	Consur
10192	Technology	New York City	United States	Patrick O'Donnell	Other	30-12-2023	US-2023-143259	10009	Gear Head AU3700S Headset	East	Consur
10193	Office Supplies	Charlottetown	Canada	Harry Olson	Wilson Jones	30-12-2023	CA-2023-143500	C0A	Wilson Jones Impact Binders	East	Consur

10194 rows × 15 columns

In [211…

```
store_cat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10194 entries, 0 to 10193
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Category        10194 non-null  object
 1   City            10194 non-null  object
 2   Country/Region  10194 non-null  object
 3   Customer Name   10194 non-null  object
 4   Manufacturer    10194 non-null  object
 5   Order Date      10194 non-null  object
 6   Order ID        10194 non-null  object
 7   Postal Code     10194 non-null  object
 8   Product Name    10194 non-null  object
 9   Region          10194 non-null  object
 10  Segment         10194 non-null  object
 11  Ship Date       10194 non-null  object
 12  Ship Mode       10194 non-null  object
 13  State/Province  10194 non-null  object
 14  Sub-Category    10194 non-null  object
dtypes: object(15)
memory usage: 1.2+ MB
```

In [216…

```
store.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10194 entries, 0 to 10193
Data columns (total 19 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Category        10194 non-null  object
 1   City            10194 non-null  object
 2   Country/Region  10194 non-null  object
 3   Customer Name   10194 non-null  object
 4   Manufacturer    10194 non-null  object
 5   Order Date      10194 non-null  object
 6   Order ID        10194 non-null  object
 7   Postal Code     10194 non-null  object
 8   Product Name    10194 non-null  object
 9   Region          10194 non-null  object
 10  Segment         10194 non-null  object
 11  Ship Date       10194 non-null  object
 12  Ship Mode       10194 non-null  object
 13  State/Province  10194 non-null  object
 14  Sub-Category    10194 non-null  object
 15  Discount        10194 non-null  float64
 16  Profit          10194 non-null  float64
 17  Quantity        10194 non-null  int64
 18  Sales           10194 non-null  float64
dtypes: float64(3), int64(1), object(15)
memory usage: 1.5+ MB
```

In [218…
```
len(store.columns)
```

Out[218… 19

In [220…
```
len(store_cat.columns)
```

Out[220… 15

In [222…
```
store.columns
```

Out[222…
```
Index(['Category', 'City', 'Country/Region', 'Customer Name', 'Manufacturer',
       'Order Date', 'Order ID', 'Postal Code', 'Product Name', 'Region',
       'Segment', 'Ship Date', 'Ship Mode', 'State/Province', 'Sub-Category',
       'Discount', 'Profit', 'Quantity', 'Sales'],
      dtype='object')
```

In [233…
```
store_num=store[['Discount', 'Profit', 'Quantity', 'Sales']]
store_num
```

Out[233…

	Discount	Profit	Quantity	Sales
0	0.2	5.5512	2	16.448
1	0.8	-5.4870	2	3.540
2	0.2	4.2717	3	11.784
3	0.2	-64.7748	3	272.736
4	0.2	4.8840	3	19.536
…	…	…	…	…
10189	0.2	19.7910	3	52.776
10190	0.2	6.4750	2	20.720
10191	0.2	-0.6048	3	3.024
10192	0.0	2.7279	7	90.930
10193	0.2	-0.6048	3	3.024

10194 rows × 4 columns

In [235…
```
store_num.columns
```

Out[235…
```
Index(['Discount', 'Profit', 'Quantity', 'Sales'], dtype='object')
```

In [237…
```
store_num.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10194 entries, 0 to 10193
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Discount  10194 non-null  float64
 1   Profit    10194 non-null  float64
 2   Quantity  10194 non-null  int64  
 3   Sales     10194 non-null  float64
dtypes: float64(3), int64(1)
memory usage: 318.7 KB
```

In [245…
```
print(len(store.columns))
print(len(store_cat.columns))
print(len(store_num.columns))
```

```
19
15
4
```

In [248…
```
store.info()                          # gives which aare numerical and which are categorical
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10194 entries, 0 to 10193
Data columns (total 19 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Category        10194 non-null  object 
 1   City            10194 non-null  object 
 2   Country/Region  10194 non-null  object 
 3   Customer Name   10194 non-null  object 
 4   Manufacturer    10194 non-null  object 
 5   Order Date      10194 non-null  object 
 6   Order ID        10194 non-null  object 
 7   Postal Code     10194 non-null  object 
 8   Product Name    10194 non-null  object 
 9   Region          10194 non-null  object 
 10  Segment         10194 non-null  object 
 11  Ship Date       10194 non-null  object 
 12  Ship Mode       10194 non-null  object 
 13  State/Province  10194 non-null  object 
 14  Sub-Category    10194 non-null  object 
 15  Discount        10194 non-null  float64
 16  Profit          10194 non-null  float64
 17  Quantity        10194 non-null  int64  
 18  Sales           10194 non-null  float64
dtypes: float64(3), int64(1), object(15)
memory usage: 1.5+ MB
```

basic pandas completed

Dataframe in python and how to import the dataset

pandas are very good package for dataframes & it is perfect for dataset & very powerfull packages

In [24]:
```
import pandas as pd             # use for data frames
```

In [25]:
```
# how to read data set
stats = pd.read_csv(r'C:\Users\jayes\OneDrive\Desktop\NareshIT\20_mar\20th, 21st\20th, 21st\DataFrame_ Pandas\data.cs
```

In [26]:
```
stats
```

Out[26]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
...	...	...	...	...	...
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

195 rows × 5 columns

In [27]:
```
# Explore data in python
#1. Full dataframe
#2. How many rows & columns. you have to chk the row becuase the no. of raw should matched with client data

len(stats) #195 rows imported (this is for tracking later part )
```

Out[27]:
```
195
```

In [28]:
```
#3. see columns
stats.columns
```

Out[28]:
```
Index(['CountryName', 'CountryCode', 'BirthRate', 'InternetUsers',
       'IncomeGroup'],
      dtype='object')
```

In [251…
```
type(stats)
```

Out[251…
```
pandas.core.frame.DataFrame
```

In [29]:
```
#4. Number of columns

len(stats.columns)
```

Out[29]:
```
5
```

In [30]:
```
#5. top rows

stats.head() # it will print top 5 rows
```

Out[30]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [31]:
```
stats.head(2)
```

Out[31]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income

In [32]:
```
#6. Bottom rows
stats.tail() #last 5 rows
```

Out[32]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

In [33]:
```
stats.tail(3)
```

Out[33]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

In [34]:
```
#7. information of the column

stats.info()  #strings are called as object
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   CountryName    195 non-null    object 
 1   CountryCode    195 non-null    object 
 2   BirthRate      195 non-null    float64
 3   InternetUsers  195 non-null    float64
 4   IncomeGroup    195 non-null    object 
dtypes: float64(2), object(3)
memory usage: 7.7+ KB
```

In [35]:
```
#8. get stats on the columns


```

```
stats.describe() #it will work like a statistic fun
```

Out[35]:

	BirthRate	InternetUsers
count	195.000000	195.000000
mean	21.469928	42.076471
std	10.605467	29.030788
min	7.900000	0.900000
25%	12.120500	14.520000
50%	19.680000	41.000000
75%	29.759500	66.225000
max	49.661000	96.546800

In [36]:
```
stats.describe().transpose() #transpose convert column into rows
```

Out[36]:

	count	mean	std	min	25%	50%	75%	max
BirthRate	195.0	21.469928	10.605467	7.9	12.1205	19.68	29.7595	49.6610
InternetUsers	195.0	42.076471	29.030788	0.9	14.5200	41.00	66.2250	96.5468

In [37]:
```
# Renaming columns of a dataframe

stats.head()
```

Out[37]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [38]:
```
stats.columns
```

Out[38]:
```
Index(['CountryName', 'CountryCode', 'BirthRate', 'InternetUsers',
       'IncomeGroup'],
      dtype='object')
```

In [39]:
```
stats.columns = ['a','b','c','d','e']
stats.head()
```

Out[39]:

	a	b	c	d	e
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [40]:
```
stats.columns = ['CountryName', 'CountryCode', 'BirthRate', 'InternetUsers','IncomeGroup']
```

In [41]:
```
stats.head()
```

Out[41]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [257…
```
stats.dtypes
```

Out[257…
```
CountryName       object
CountryCode       object
BirthRate        float64
InternetUsers    float64
IncomeGroup       object
dtype: object
```

In [42]:
```
# subsetting a dataframes in pandas

#1. Rows
#2. Columns
#3. combine the two
```

In [43]:
```
# Rows:

stats[21:26] #how python know that only this is rows based on index
```

Out[43]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
21	Belize	BLZ	23.092	33.60	Upper middle income
22	Bermuda	BMU	10.400	95.30	High income
23	Bolivia	BOL	24.236	36.94	Lower middle income
24	Brazil	BRA	14.931	51.04	Upper middle income
25	Barbados	BRB	12.188	73.00	High income

In [44]:
```
stats[:]
```

Out[44]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
...	...	...	...	...	...
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

195 rows × 5 columns

In [45]:
```
stats[:10]
```

Out[45]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9000	High income
1	Afghanistan	AFG	35.253	5.9000	Low income
2	Angola	AGO	45.985	19.1000	Upper middle income
3	Albania	ALB	12.877	57.2000	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0000	High income
5	Argentina	ARG	17.716	59.9000	High income
6	Armenia	ARM	13.308	41.9000	Lower middle income
7	Antigua and Barbuda	ATG	16.447	63.4000	High income
8	Australia	AUS	13.200	83.0000	High income
9	Austria	AUT	9.400	80.6188	High income

In [46]:
```
stats.head(10)
```

Out[46]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9000	High income
1	Afghanistan	AFG	35.253	5.9000	Low income
2	Angola	AGO	45.985	19.1000	Upper middle income
3	Albania	ALB	12.877	57.2000	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0000	High income
5	Argentina	ARG	17.716	59.9000	High income
6	Armenia	ARM	13.308	41.9000	Lower middle income
7	Antigua and Barbuda	ATG	16.447	63.4000	High income
8	Australia	AUS	13.200	83.0000	High income
9	Austria	AUT	9.400	80.6188	High income

In [47]:
```
# How to reverse the dataframe

stats[ : : -1]
```

Out[47]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
194	Zimbabwe	ZWE	35.715	18.5	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
191	South Africa	ZAF	20.850	46.5	Upper middle income
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
...	...	...	...	...	...
4	United Arab Emirates	ARE	11.044	88.0	High income
3	Albania	ALB	12.877	57.2	Upper middle income
2	Angola	AGO	45.985	19.1	Upper middle income
1	Afghanistan	AFG	35.253	5.9	Low income
0	Aruba	ABW	10.244	78.9	High income

195 rows × 5 columns

In [48]:
```
stats
```

Out[48]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
...	...	...	...	...	...
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

195 rows × 5 columns

In [49]:
```
# get only every 20th row

stats[: : 20]
```

Out[49]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9000	High income
20	Belarus	BLR	12.500	54.1700	Upper middle income
40	Costa Rica	CRI	15.022	45.9600	Upper middle income
60	Gabon	GAB	30.555	9.2000	Upper middle income
80	India	IND	20.291	15.1000	Lower middle income
100	Libya	LBY	21.425	16.5000	Upper middle income
120	Mozambique	MOZ	39.705	5.4000	Low income
140	Poland	POL	9.600	62.8492	High income
160	Suriname	SUR	18.455	37.4000	Upper middle income
180	Uruguay	URY	14.374	57.6900	High income

In [50]:
```
# COLUMNS:

stats.columns
```

Out[50]:
```
Index(['CountryName', 'CountryCode', 'BirthRate', 'InternetUsers',
       'IncomeGroup'],
      dtype='object')
```

In [51]:
```
stats.head()
```

Out[51]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [52]:
```
stats['CountryName'].head()
```

Out[52]:
```
0                  Aruba
1            Afghanistan
2                 Angola
3                Albania
4    United Arab Emirates
Name: CountryName, dtype: object
```

In [53]:
```
['CountryName','BirthRate']
```

Out[53]:
```
['CountryName', 'BirthRate']
```

In [54]:
```
stats[['CountryName','BirthRate']].head()
```

Out[54]:

	CountryName	BirthRate
0	Aruba	10.244
1	Afghanistan	35.253
2	Angola	45.985
3	Albania	12.877
4	United Arab Emirates	11.044

In [55]:
```
stats.head()
```

Out[55]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [56]:
```
stats['BirthRate']
```

Out[56]:
```
0      10.244
1      35.253
2      45.985
3      12.877
4      11.044
        ...
190    32.947
191    20.850
192    42.394
193    40.471
194    35.715
Name: BirthRate, Length: 195, dtype: float64
```

In [57]:
```
# combine the two

stats[4:8][['CountryName','BirthRate']]
```

Out[57]:

	CountryName	BirthRate
4	United Arab Emirates	11.044
5	Argentina	17.716
6	Armenia	13.308
7	Antigua and Barbuda	16.447

In [58]:
```
stats [['CountryName','BirthRate']][4:8]
```

Out[58]:

	CountryName	BirthRate
4	United Arab Emirates	11.044
5	Argentina	17.716
6	Armenia	13.308
7	Antigua and Barbuda	16.447

In [59]:
```
df1 = stats [['CountryName','BirthRate']]
```

In [60]:
```
df1
```

Out[60]:

	CountryName	BirthRate
0	Aruba	10.244
1	Afghanistan	35.253
2	Angola	45.985
3	Albania	12.877
4	United Arab Emirates	11.044
...	...	...
190	Yemen, Rep.	32.947
191	South Africa	20.850
192	Congo, Dem. Rep.	42.394
193	Zambia	40.471
194	Zimbabwe	35.715

195 rows × 2 columns

In [61]:
```
df2 = stats[4:8]
df2
```

Out[61]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
4	United Arab Emirates	ARE	11.044	88.0	High income
5	Argentina	ARG	17.716	59.9	High income
6	Armenia	ARM	13.308	41.9	Lower middle income
7	Antigua and Barbuda	ATG	16.447	63.4	High income

In [62]:
```
# Basic operatioin of dataframe
stats.head()
```

Out[62]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [63]:
```
stats[['CountryCode','BirthRate','InternetUsers']][4:8] #subet dataframe
```

Out[63]:

	CountryCode	BirthRate	InternetUsers
4	ARE	11.044	88.0
5	ARG	17.716	59.9
6	ARM	13.308	41.9
7	ATG	16.447	63.4

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

Out[64]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [65]:
```
#Mathmetical operation =
stats.BirthRate * stats.InternetUsers
```

Out[65]:
```
0      808.2516
1      207.9927
2      878.3135
3      736.5644
4      971.8720
         ...
190    658.9400
191    969.5250
192     93.2668
193    623.2534
194    660.7275
Length: 195, dtype: float64
```

In [66]:
```
# Add a column

stats['myCalc'] = stats.BirthRate * stats.InternetUsers
stats.head()
```

Out[66]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup	myCalc
0	Aruba	ABW	10.244	78.9	High income	808.2516
1	Afghanistan	AFG	35.253	5.9	Low income	207.9927
2	Angola	AGO	45.985	19.1	Upper middle income	878.3135
3	Albania	ALB	12.877	57.2	Upper middle income	736.5644
4	United Arab Emirates	ARE	11.044	88.0	High income	971.8720

In [67]:
```
#Remove a column                                    # axis = 0 indicates row, axis = 1 indicates column

stats.drop('myCalc',axis = 1)                              # temporarily delete column myCalc
```

Out[67]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
...	...	...	...	...	...
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

195 rows × 5 columns

In [68]:
```
stats.head()
```

Out[68]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup	myCalc
0	Aruba	ABW	10.244	78.9	High income	808.2516
1	Afghanistan	AFG	35.253	5.9	Low income	207.9927
2	Angola	AGO	45.985	19.1	Upper middle income	878.3135
3	Albania	ALB	12.877	57.2	Upper middle income	736.5644
4	United Arab Emirates	ARE	11.044	88.0	High income	971.8720

In [69]:
```
stats = stats.drop('myCalc', axis=1)                    #  permenanetly delete column
```

In [70]:
```
stats.head()
```

Out[70]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [71]:
```
stats.drop(0, axis=0)                                   # axis = 0 delete column and axis = 1 delete row
```

Out[71]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
5	Argentina	ARG	17.716	59.9	High income
...	...	...	...	...	...
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

194 rows × 5 columns

In [72]:
```
stats.head()
```

Out[72]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [73]:
```
stats.columns[2]
```

Out[73]: 'BirthRate'

In [74]:
```
stats.InternetUsers<2 #we are checking given condition if its correct true or false
```

Out[74]:
```
0      False
1      False
2      False
3      False
4      False
       ...
190    False
191    False
192    False
193    False
194    False
Name: InternetUsers, Length: 195, dtype: bool
```

In [261…
```
Filter = stats.InternetUsers < 2
print(Filter)
print(type(Filter))
```

```
0      False
1      False
2      False
3      False
4      False
       ...
190    False
191    False
192    False
193    False
194    False
Name: InternetUsers, Length: 195, dtype: bool
<class 'pandas.core.series.Series'>
```

In [76]:
```
Filter.value_counts()
```

Out[76]:
```
InternetUsers
False    186
True       9
Name: count, dtype: int64
```

In [77]:
```
stats[3:7]
```

Out[77]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
5	Argentina	ARG	17.716	59.9	High income
6	Armenia	ARM	13.308	41.9	Lower middle income

In [78]:
```
stats[30:40]
```

Out[78]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
30	Canada	CAN	10.900	85.80	High income
31	Switzerland	CHE	10.200	86.34	High income
32	Chile	CHL	13.385	66.50	High income
33	China	CHN	12.100	45.80	Upper middle income
34	Cote d'Ivoire	CIV	37.320	8.40	Lower middle income
35	Cameroon	CMR	37.236	6.40	Lower middle income
36	Congo, Rep.	COG	37.011	6.60	Lower middle income
37	Colombia	COL	16.076	51.70	Upper middle income
38	Comoros	COM	34.326	6.50	Low income
39	Cabo Verde	CPV	21.625	37.50	Lower middle income

In [79]:
```
stats[Filter] # IT WILL take that row which are True
```

Out[79]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
11	Burundi	BDI	44.151	1.3	Low income
52	Eritrea	ERI	34.800	0.9	Low income
55	Ethiopia	ETH	32.925	1.9	Low income
64	Guinea	GIN	37.337	1.6	Low income
117	Myanmar	MMR	18.119	1.6	Lower middle income
127	Niger	NER	49.661	1.7	Low income
154	Sierra Leone	SLE	36.729	1.7	Low income
156	Somalia	SOM	43.891	1.5	Low income
172	Timor-Leste	TLS	35.755	1.1	Lower middle income

In [80]:
```
stats[Filter == False] # IT WILL take that row which are false
```

Out[80]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income
...	...	...	...	...	...
190	Yemen, Rep.	YEM	32.947	20.0	Lower middle income
191	South Africa	ZAF	20.850	46.5	Upper middle income
192	Congo, Dem. Rep.	COD	42.394	2.2	Low income
193	Zambia	ZMB	40.471	15.4	Lower middle income
194	Zimbabwe	ZWE	35.715	18.5	Low income

186 rows × 5 columns

In [81]:
```
Filter2 = stats.BirthRate>40
```

In [82]:
```
Filter2.value_counts()
```

Out[82]:
```
BirthRate
False    183
True      12
Name: count, dtype: int64
```

In [83]:
```
Filter & Filter2
```

Out[83]:
```
0      False
1      False
2      False
3      False
4      False
       ...
190    False
191    False
192    False
193    False
194    False
Length: 195, dtype: bool
```

In [84]:
```
stats[Filter & Filter2]
```

Out[84]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
11	Burundi	BDI	44.151	1.3	Low income
127	Niger	NER	49.661	1.7	Low income
156	Somalia	SOM	43.891	1.5	Low income

In [85]:
```
(Filter & Filter2).value_counts()
```

Out[85]:
```
False    192
True       3
Name: count, dtype: int64
```

In [86]:
```
stats[(stats.BirthRate > 40) & (stats.InternetUsers < 2)]
```

Out[86]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
11	Burundi	BDI	44.151	1.3	Low income
127	Niger	NER	49.661	1.7	Low income
156	Somalia	SOM	43.891	1.5	Low income

In [87]:
```
stats[(stats.BirthRate > 40) & (stats.InternetUsers < 2)]
```

Out[87]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
11	Burundi	BDI	44.151	1.3	Low income
127	Niger	NER	49.661	1.7	Low income
156	Somalia	SOM	43.891	1.5	Low income

In [88]:
```
# here 'stats' is dataframe which is a table of data. Filter is a boolean series which contain True or False rows
```

In [89]:
```
stats.head()
```

Out[89]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

In [90]:
```
stats[stats.IncomeGroup == 'Low income']
```

Out[90]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
1	Afghanistan	AFG	35.253	5.90	Low income
11	Burundi	BDI	44.151	1.30	Low income
13	Benin	BEN	36.440	4.90	Low income
14	Burkina Faso	BFA	40.551	9.10	Low income
29	Central African Republic	CAF	34.076	3.50	Low income
38	Comoros	COM	34.326	6.50	Low income
52	Eritrea	ERI	34.800	0.90	Low income
55	Ethiopia	ETH	32.925	1.90	Low income
64	Guinea	GIN	37.337	1.60	Low income
65	Gambia, The	GMB	42.525	14.00	Low income
66	Guinea-Bissau	GNB	37.503	3.10	Low income
77	Haiti	HTI	25.345	10.60	Low income
93	Cambodia	KHM	24.462	6.80	Low income
99	Liberia	LBR	35.521	3.20	Low income
111	Madagascar	MDG	34.686	3.00	Low income
115	Mali	MLI	44.138	3.50	Low income
120	Mozambique	MOZ	39.705	5.40	Low income
123	Malawi	MWI	39.459	5.05	Low income
127	Niger	NER	49.661	1.70	Low income
132	Nepal	NPL	20.923	13.30	Low income
148	Rwanda	RWA	32.689	9.00	Low income
154	Sierra Leone	SLE	36.729	1.70	Low income

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
156	Somalia	SOM	43.891	1.50	Low income
158	South Sudan	SSD	37.126	14.10	Low income
167	Chad	TCD	45.745	2.30	Low income
168	Togo	TGO	36.080	4.50	Low income
177	Tanzania	TZA	39.518	4.40	Low income
178	Uganda	UGA	43.474	16.20	Low income
192	Congo, Dem. Rep.	COD	42.394	2.20	Low income
194	Zimbabwe	ZWE	35.715	18.50	Low income

In [91]:
```
# How to get the unique categories

stats.IncomeGroup.unique()
```

Out[91]:
```
array(['High income', 'Low income', 'Upper middle income',
       'Lower middle income'], dtype=object)
```

In [263…
```
stats.IncomeGroup.nunique()
```

Out[263…
```
4
```

In [92]:
```
stats['IncomeGroup'].value_counts()
```

Out[92]:
```
IncomeGroup
High income            67
Lower middle income    50
Upper middle income    48
Low income             30
Name: count, dtype: int64
```

seaborn

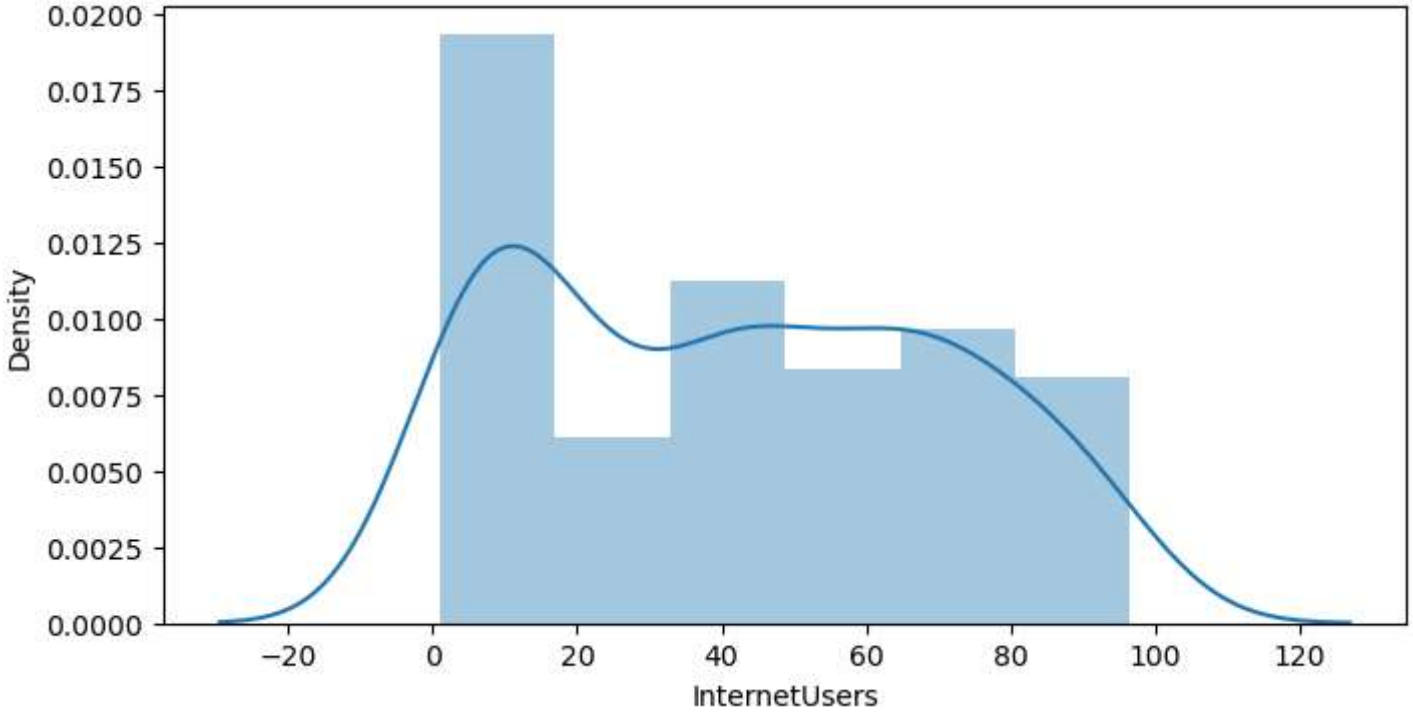
In [94]:
```
# Introduction to seaborn # seaborn is very powerfull visualizatio(STATISTIC VISULAIZATION) pkg in python

import matplotlib.pyplot as plt # visulaiztion
import seaborn as sns # distribution visualtion

%matplotlib inline
plt.rcParams['figure.figsize'] = 8,4                          #width and height of figure

import warnings
warnings.filterwarnings('ignore')
```

In [95]:
```
stats.head()
```

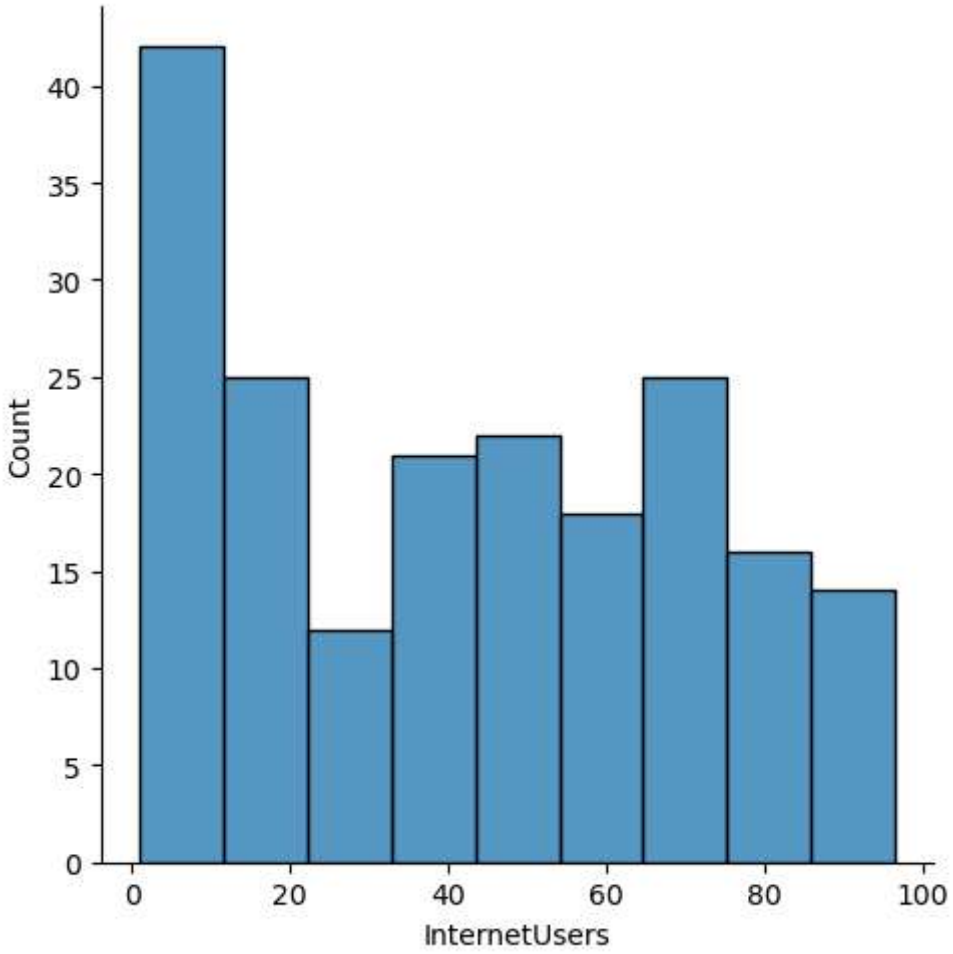
Out[95]:

	CountryName	CountryCode	BirthRate	InternetUsers	IncomeGroup
0	Aruba	ABW	10.244	78.9	High income
1	Afghanistan	AFG	35.253	5.9	Low income
2	Angola	AGO	45.985	19.1	Upper middle income
3	Albania	ALB	12.877	57.2	Upper middle income
4	United Arab Emirates	ARE	11.044	88.0	High income

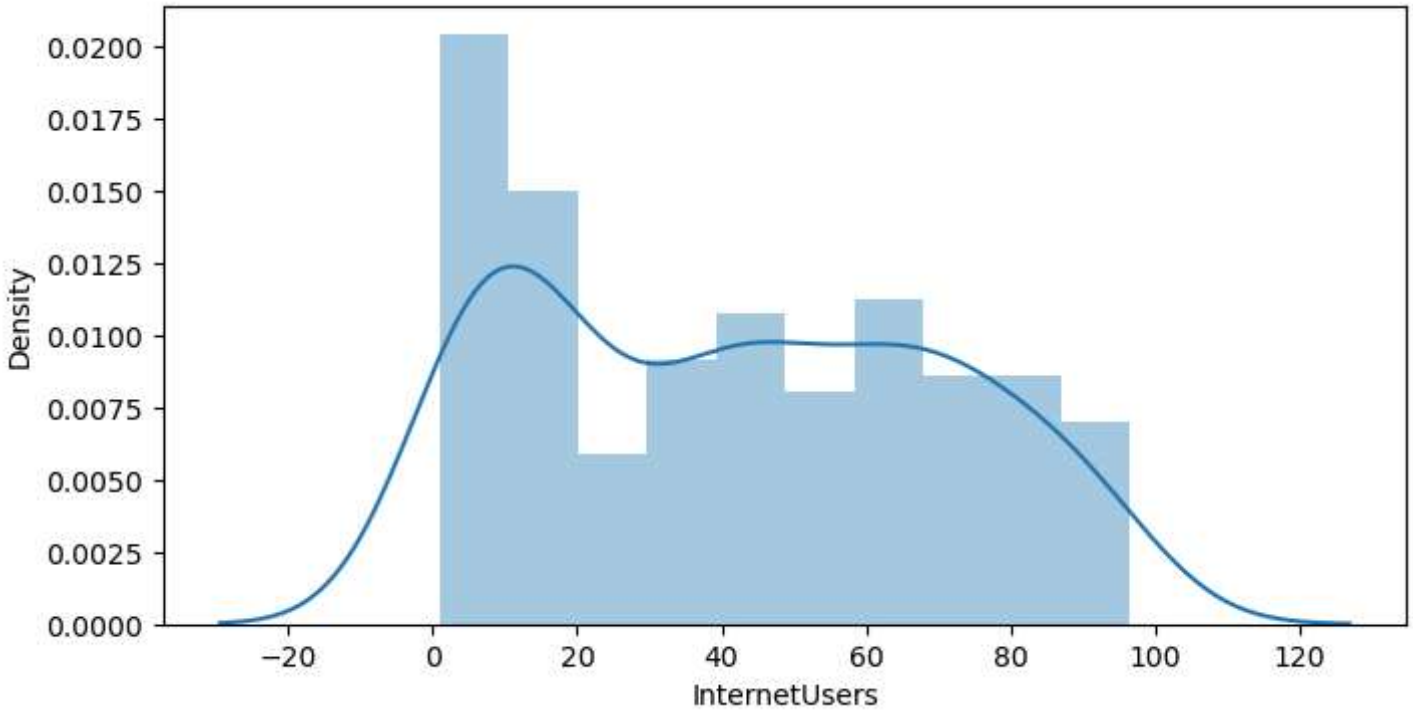
In [96]:
```
# Distributions:
sns.distplot(stats["InternetUsers"])                          # univariant analysis. others are twovariant and m
plt.show()
```



In [265…
```
sns.displot(stats["InternetUsers"])                    # remove line by avoiding t
plt.show()
```

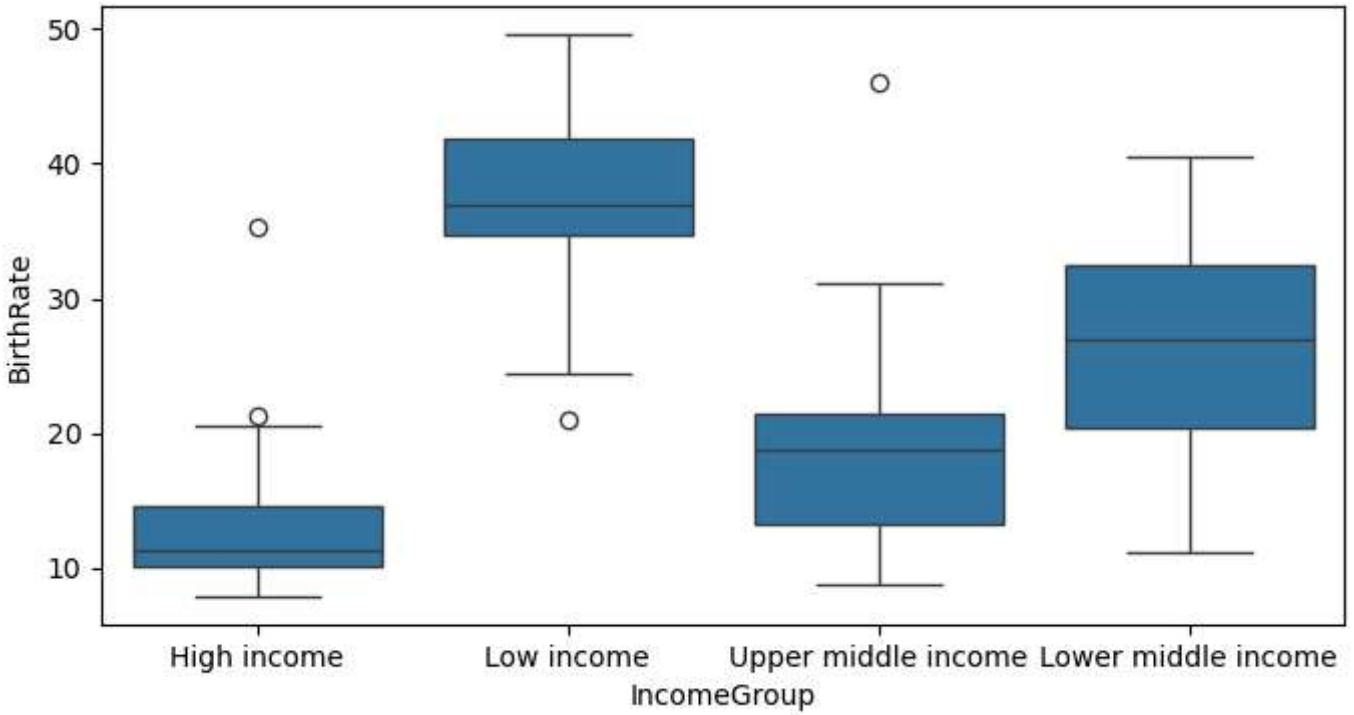


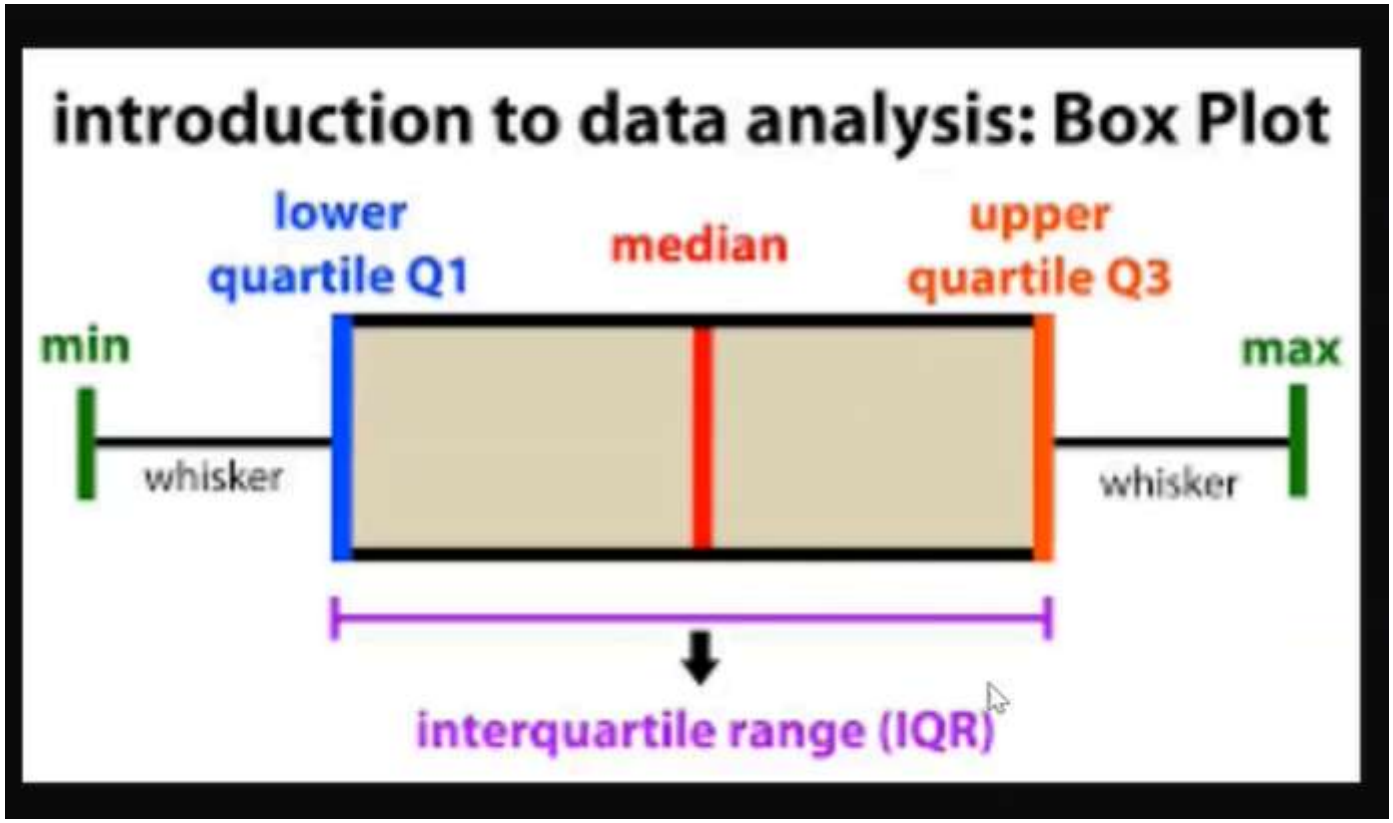
In [97]:
```
vis1 = sns.distplot(stats["InternetUsers"], bins=10)
plt.show()
```



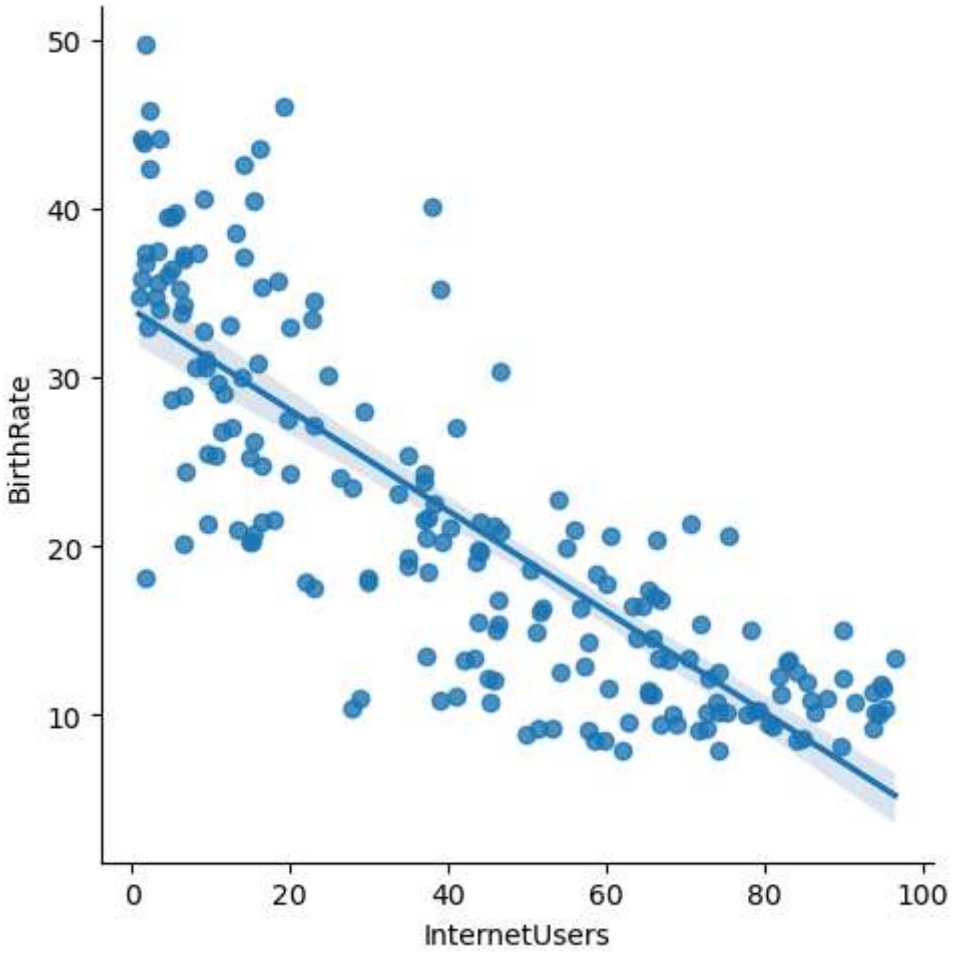
In [273…

```
vis2 = sns.boxplot(data = stats, x = 'IncomeGroup', y='BirthRate')              # Bivariant analysis
plt.show()
```


50
40
30
20
10
BirthRate
High income
Low income
Upper middle income
Lower middle income
IncomeGroup

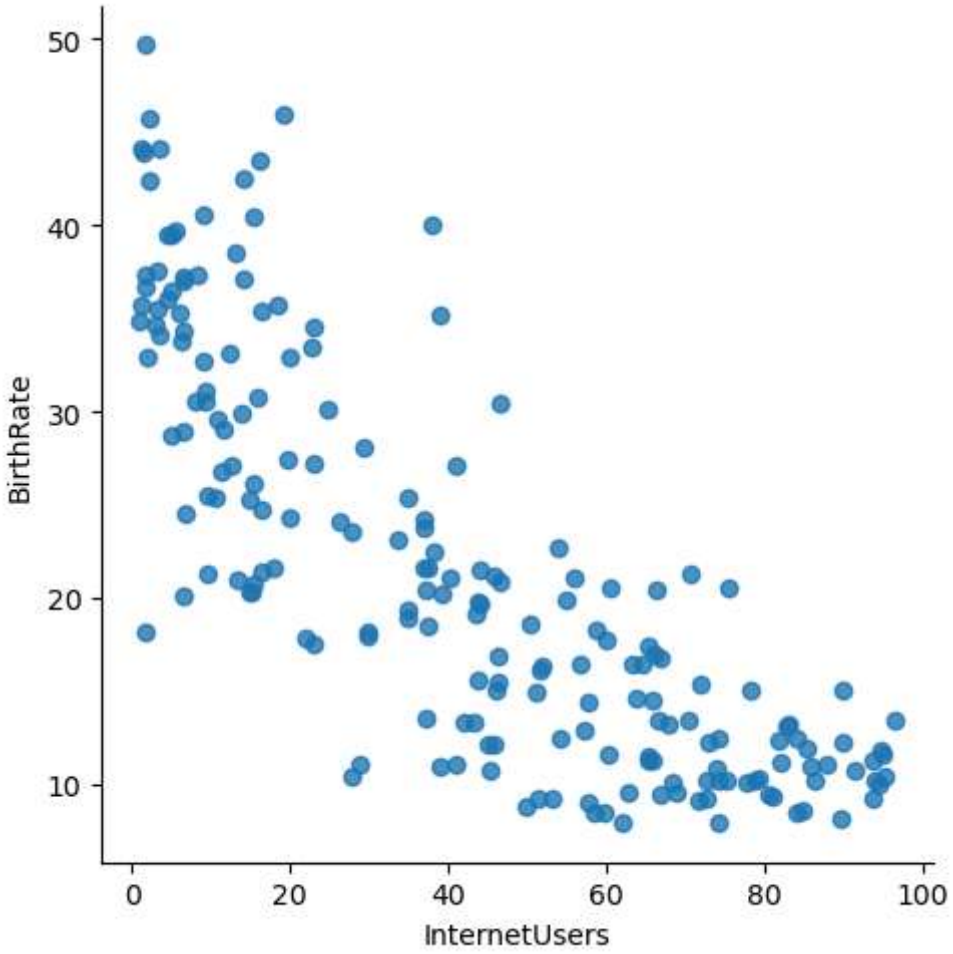


In [279…
```
vis3 = sns.lmplot(data = stats, x = 'InternetUsers', y = 'BirthRate')              #bivariate analysis
plt.show()
```



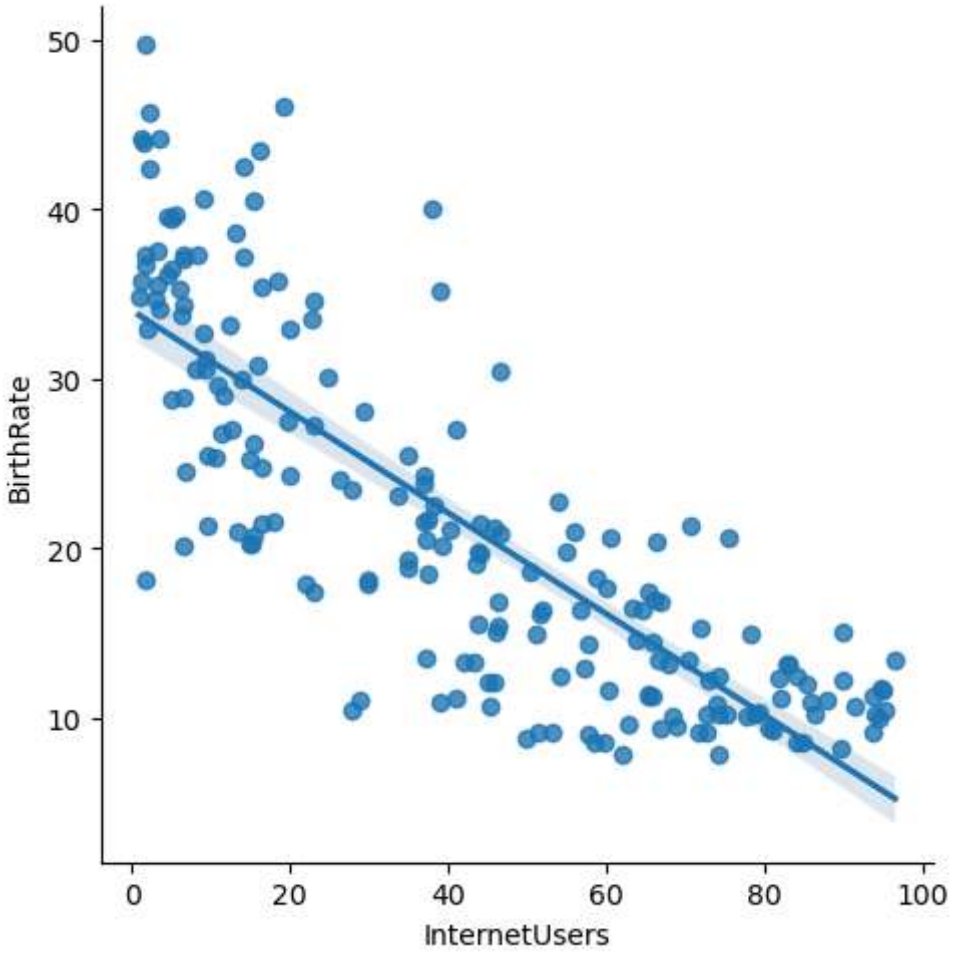
In [298…

```
vis4 = sns.lmplot(data = stats, x = 'InternetUsers', y = 'BirthRate', fit_reg = False)
plt.show()
```



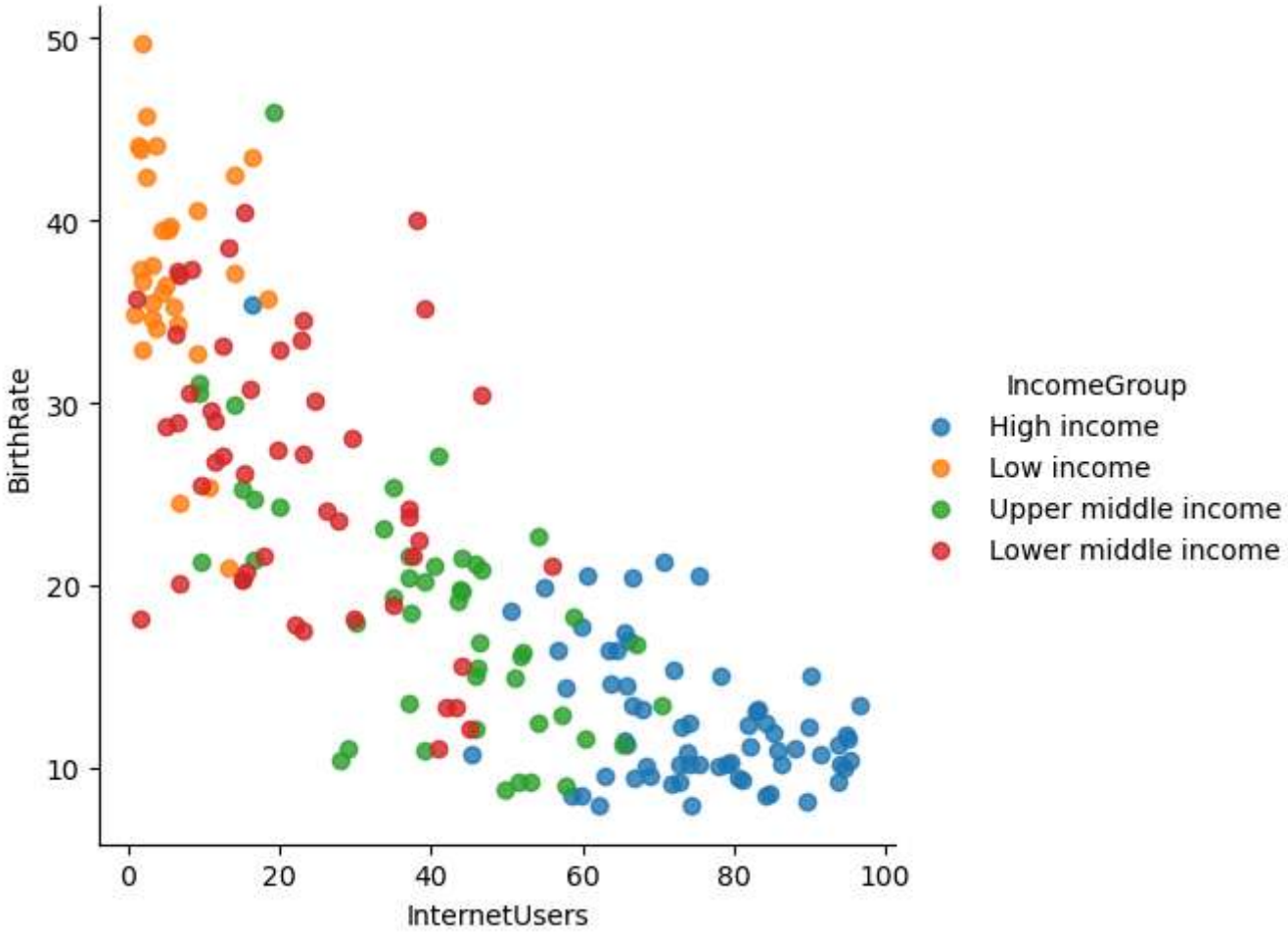
In [300…

```
vis5 = sns.lmplot(data = stats, x = 'InternetUsers', y = 'BirthRate', fit_reg = True)
plt.show()
```



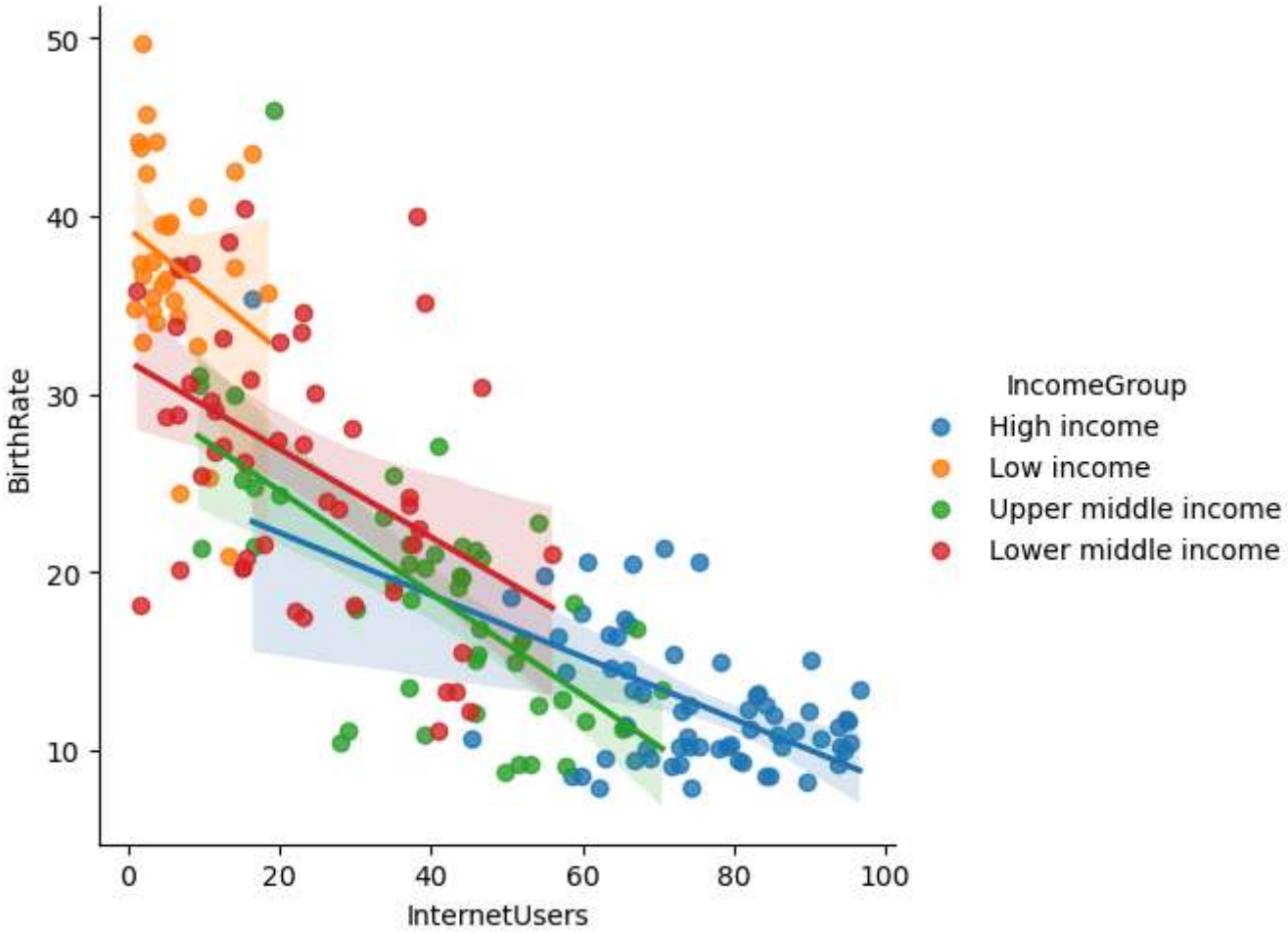
In [302…

```
vis6 = sns.lmplot(data = stats, x = 'InternetUsers', y = 'BirthRate', fit_reg = False, hue = 'IncomeGroup')
plt.show()
```



In [304…

```
vis7 = sns.lmplot(data = stats, x = 'InternetUsers', y = 'BirthRate', fit_reg = True, hue = 'IncomeGroup')
plt.show()
```



In this section we learned

- importing data into python
- Dataframe via panda
- exploring datasets: head()tail()info()describe()
- Renaming columns
- subsetting dataframes
- Basic operations with dataframe
- filtering data frames

- seaborn introduction