Pandas - II

🎯 Session Objectives:

- ✅ Differentiate Pandas Series vs NumPy Arrays
- ✅ Create Series from scalar, list, array, and dictionary
- ✅ Access Series elements using indexing and slicing
- ✅ Understand attributes of Series
- ✅ Learn basic mathematical operations on Series
- ✅ Understand the key attributes of a DataFrame
- ✅ Use various DataFrame methods to explore and manipulate data
- ✅ Perform row and column operations
- ✅ Use joining, merging, and concatenation techniques across DataFrames

```
# DataFrame # 2D Labelled Data Structure in Python [rows,cols]
# Spreadsheet in Excel & SQL Matrix
# Each columns contains different dtypes [object,int,float,str,boolean]
```

```
import numpy as np
import pandas as pd
empty_df = pd.DataFrame()
empty_df
```

```
_
```

```
print(empty_df)
```

```
Empty DataFrame
Columns: []
Index: []
```

```python
# Let's create a DataFrame which includes person data
# Name | Age | State
personal_info = np.array([
    ['Rajat Singh',28,'Delhi'],
    ['Shyam Sundar Rao',26,'Andhra Pradesh'],
    ['Vaibhav Ashtekar',32,'Telangana'],
    ['Paramjeet Kaur', 28 , 'Karnataka'],
    ['Salman Khan', 55, 'Maharashtra'],
    ['John Rambo',30,'Maharashtra'],
    ['Shahzain Sher Shan' ,26 ,'Madhya Pradesh'],
    ['Pragya Chowdhury', 27, 'West Bengal'],
    ['Virat Kohli', 35, 'Delhi'],
])
person_df = pd.DataFrame(personal_info)
person_df
```

	0	1	2
0	Rajat Singh	28	Delhi
1	Shyam Sundar Rao	26	Andhra Pradesh
2	Vaibhav Ashtekar	32	Telangana
3	Paramjeet Kaur	28	Karnataka
4	Salman Khan	55	Maharashtra
5	John Rambo	30	Maharashtra
6	Shahzain Sher Shan	26	Madhya Pradesh
7	Pragya Chowdhury	27	West Bengal
8	Virat Kohli	35	Delhi

```python
person_df = pd.DataFrame(personal_info , columns = ['Name','Age','State'])
person_df
```

	Name	Age	State
0	Rajat Singh	28	Delhi
1	Shyam Sundar Rao	26	Andhra Pradesh
2	Vaibhav Ashtekar	32	Telangana
3	Paramjeet Kaur	28	Karnataka
4	Salman Khan	55	Maharashtra
5	John Rambo	30	Maharashtra
6	Shahzain Sher Shan	26	Madhya Pradesh
7	Pragya Chowdhury	27	West Bengal
8	Virat Kohli	35	Delhi

```python
person_df.head() # First Five Rows
```

	Name	Age	State
0	Rajat Singh	28	Delhi
1	Shyam Sundar Rao	26	Andhra Pradesh
2	Vaibhav Ashtekar	32	Telangana
3	Paramjeet Kaur	28	Karnataka
4	Salman Khan	55	Maharashtra

```
person_df.head(7) # First Seven Rows
```

	Name	Age	State
0	Rajat Singh	28	Delhi
1	Shyam Sundar Rao	26	Andhra Pradesh
2	Vaibhav Ashtekar	32	Telangana
3	Paramjeet Kaur	28	Karnataka
4	Salman Khan	55	Maharashtra
5	John Rambo	30	Maharashtra
6	Shahzain Sher Shan	26	Madhya Pradesh

```
person_df.tail() # Return the last `n` rows. [Default = 5]
```

	Name	Age	State
4	Salman Khan	55	Maharashtra
5	John Rambo	30	Maharashtra
6	Shahzain Sher Shan	26	Madhya Pradesh
7	Pragya Chowdhury	27	West Bengal
8	Virat Kohli	35	Delhi

```
person_df.describe() # Statistical info
```

	Name	Age	State
count	9	9	9
unique	9	7	7
top	Rajat Singh	28	Delhi
freq	1	2	2

```
count -> non-null values
unique -> number of distinct values
top -> Most Frequent Value [Mode] [Index Return]
freq -> Frequency of most frequent value
```

```
person_df['Age']
```

```
0    28
1    26
2    32
3    28
4    55
5    30
6    26
7    27
8    35
Name: Age, dtype: object
```

```
type(person_df['Age']) # series
```

```
pandas.core.series.Series
```

```
type(person_df) # DataFrame [rows , cols]
```

```
pandas.core.frame.DataFrame
```

```
person_df['Age'].value_counts()
```

```
Age
28    2
26    2
32    1
55    1
30    1
27    1
35    1
Name: count, dtype: int64
```

```
person_df['State'].value_counts()
```

```
State
Delhi             2
Maharashtra       2
Andhra Pradesh    1
Telangana         1
Karnataka         1
Madhya Pradesh    1
West Bengal       1
Name: count, dtype: int64
```

```
person_df['Name'].values
```

```
array(['Rajat Singh', 'Shyam Sundar Rao', 'Vaibhav Ashtekar',
       'Paramjeet Kaur', 'Salman Khan', 'John Rambo',
       'Shahzain Sher Shan', 'Pragya Chowdhury', 'Virat Kohli'],
      dtype=object)
```

```
person_df['Age'].values.astype('int')
```

```
array([28, 26, 32, 28, 55, 30, 26, 27, 35])
```

```
person_df['State'].values
```

```
array(['Delhi', 'Andhra Pradesh', 'Telangana', 'Karnataka', 'Maharashtra',
       'Maharashtra', 'Madhya Pradesh', 'West Bengal', 'Delhi'],
      dtype=object)
```

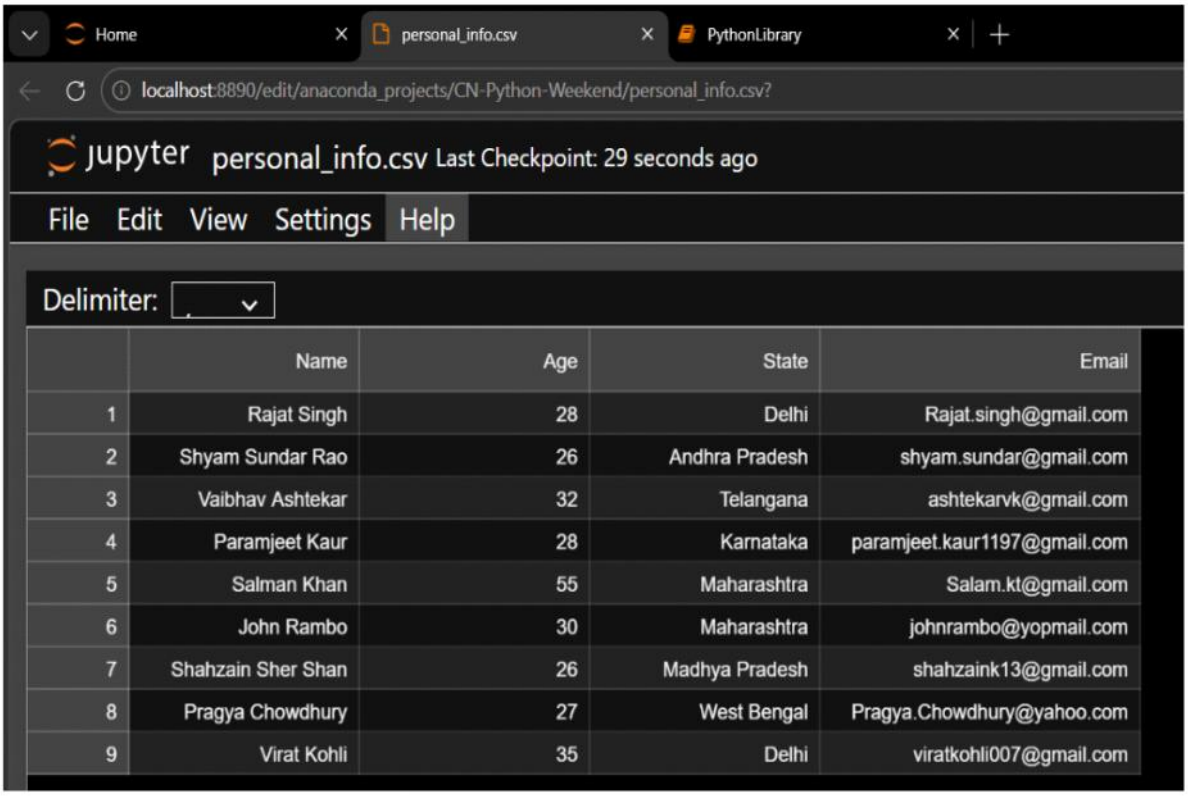
```
# person_df with use of Dictionary
# Name | Age | State | Email
# Two-dimensional, size-mutable, potentially heterogeneous tabular data.
personal_data = {
    'Name' : ['Rajat Singh', 'Shyam Sundar Rao', 'Vaibhav Ashtekar',
       'Paramjeet Kaur', 'Salman Khan', 'John Rambo',
       'Shahzain Sher Shan', 'Pragya Chowdhury', 'Virat Kohli'],
    'Age' : [28, 26, 32, 28, 55, 30, 26, 27, 35],
    'State' : ['Delhi', 'Andhra Pradesh', 'Telangana', 'Karnataka', 'Maharashtra',
       'Maharashtra', 'Madhya Pradesh', 'West Bengal', 'Delhi'],
    'Email' : ['Rajat.singh@gmail.com', 'shyam.sundar@gmail.com', 'ashtekarvk@gmail.com',
       'paramjeet.kaur1197@gmail.com', 'Salam.kt@gmail.com', 'johnrambo@yopmail.com',
       'shahzaink13@gmail.com', 'Pragya.Chowdhury@yahoo.com', 'viratkohli007@gmail.com']
}
person_df = pd.DataFrame(personal_data)
person_df
```

	Name	Age	State	Email
0	Rajat Singh	28	Delhi	Rajat.singh@gmail.com
1	Shyam Sundar Rao	26	Andhra Pradesh	shyam.sundar@gmail.com
2	Vaibhav Ashtekar	32	Telangana	ashtekarvk@gmail.com
3	Paramjeet Kaur	28	Karnataka	paramjeet.kaur1197@gmail.com
4	Salman Khan	55	Maharashtra	Salam.kt@gmail.com
5	John Rambo	30	Maharashtra	johnrambo@yopmail.com
6	Shahzain Sher Shan	26	Madhya Pradesh	shahzaink13@gmail.com
7	Pragya Chowdhury	27	West Bengal	Pragya.Chowdhury@yahoo.com
8	Virat Kohli	35	Delhi	viratkohli007@gmail.com

```python
# Exporting a clean file into our local system .csv(Comma Seperated Values)
person_df.to_csv('personal_info.csv' , index = False)
```

```python
# Exporting to Excel
person_df.to_excel('personal_data.xlsx' , index = False)
```

```python
# Importing the DataSet.....
try:
    customers = pd.read_excel('customers.xlsx')
    products = pd.read_excel('products.xlsx')
    purchases = pd.read_excel('purchases.xlsx')
except FileNotFoundError:
    print("File Not Found!")
```

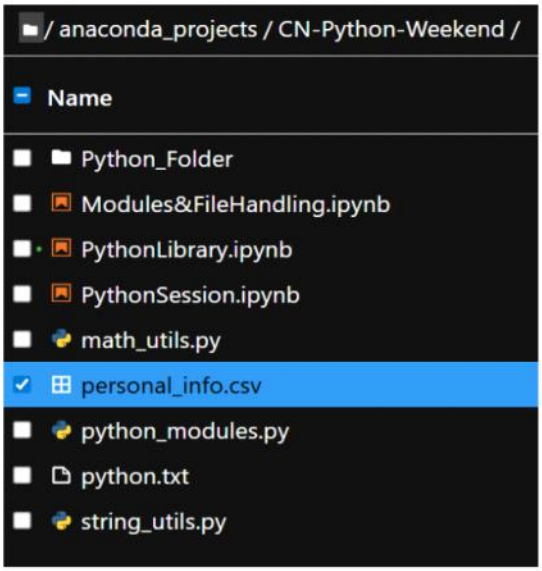


	Name	Age	State	Email
1	Rajat Singh	28	Delhi	Rajat.singh@gmail.com
2	Shyam Sundar Rao	26	Andhra Pradesh	shyam.sundar@gmail.com
3	Vaibhav Ashtekar	32	Telangana	ashtekarvk@gmail.com
4	Paramjeet Kaur	28	Karnataka	paramjeet.kaur1197@gmail.com
5	Salman Khan	55	Maharashtra	Salam.kt@gmail.com
6	John Rambo	30	Maharashtra	johnrambo@yopmail.com
7	Shahzain Sher Shan	26	Madhya Pradesh	shahzaink13@gmail.com
8	Pragya Chowdhury	27	West Bengal	Pragya.Chowdhury@yahoo.com
9	Virat Kohli	35	Delhi	viratkohli007@gmail.com

```
person_df = pd.DataFrame(personal_data)
person_df
```

	Name	Age	State	Email
0	Rajat Singh	28	Delhi	Rajat.singh@gmail.com
1	Shyam Sundar Rao	26	Andhra Pradesh	shyam.sundar@gmail.com
2	Vaibhav Ashtekar	32	Telangana	ashtekarvk@gmail.com
3	Paramjeet Kaur	28	Karnataka	paramjeet.kaur1197@gmail.com
4	Salman Khan	55	Maharashtra	Salam.kt@gmail.com
5	John Rambo	30	Maharashtra	johnrambo@yopmail.com
6	Shahzain Sher Shan	26	Madhya Pradesh	shahzaink13@gmail.com
7	Pragya Chowdhury	27	West Bengal	Pragya.Chowdhury@yahoo.com
8	Virat Kohli	35	Delhi	viratkohli007@gmail.com

```
# Exporting a clean file into our local system .csv(Comma Seperated Values)
person_df.to_csv('personal_info.csv' , index = False)
```




```
customers
```

last_name	email	gender	street_num	street_name	street_suffix	city	state	postcode
Southcott	rsouthcott0@clickbank.net	Male	1.0	Trailsway	Road	San Diego	California	92127.0
Molyneaux	cmolyneaux1@wiley.com	Male	NaN	NaN	NaN	El Paso	Texas	NaN
Westrip	bwestrip2@symantec.com	Female	4057.0	Arkansas	Circle	San Antonio	Texas	78220.0
Pankettman	rpankettman3@wiley.com	Male	74.0	Debs	Point	Memphis	Tennessee	38150.0
Althorpe	malthorpe4@51.la	NaN	NaN	2nd	Drive	NaN	NaN	83732.0
...	...	...	...	...	...	...	...	...
Alman	malmanrn@cornell.edu	Female	0.0	Thompson	Way	Reading	Pennsylvania	19610.0
Heckle	whecklero@fc2.com	Female	701.0	Rowland	Hill	Indianapolis	Indiana	46278.0
Birt	NaN	Male	2.0	Basil	Road	Waterbury	Connecticut	6721.0
Standbrooke	istandbrookerq@yellowpages.com	Male	NaN	Kenwood	Drive	Savannah	Georgia	31405.0
Malpas	NaN	Male	205.0	Farwell	Park	Atlanta	Georgia	30386.0

products

	Unnamed: 0	id	product	cost	company
0	0	1	Liners - Baking Cups	$6.36	Skipfire
1	1	2	Nori Sea Weed - Gold Label	$85.74	Dynazzy
2	2	3	Bar Bran Honey Nut	$65.40	Ntag
3	3	4	Soup - Campbells Beef Stew	$68.16	Photojam
4	4	5	Wine - Shiraz Wolf Blass Premium	$87.39	Eare
5	5	6	Wine - White, Riesling, Semi - Dry	$99.22	Livepath
6	6	7	Brandy - Bar	$13.83	Oloo
7	7	8	Onions - White	$42.19	Oozz
8	8	9	Lettuce - Baby Salad Greens	$30.01	Meevee

purchases

	Unnamed: 0	id	purch_date	customer_num	product_num	amount	paid
0	0	1	2019-01-03 00:00:00	823	27	12	$568.92
1	1	2	2019-01-03 00:00:00	606	28	14	$395.36
2	2	3	2019-01-03 00:00:00	955	9	17	$510.17
3	3	4	2019-01-03 00:00:00	577	19	3	$68.49
4	4	5	2019-01-03 00:00:00	429	8	18	$759.42
...	...	...	...	...	...	...	...
5995	5995	5996	06/20/2019	893	33	5	$411.10
5996	5996	5997	06/20/2019	566	23	11	$178.97
5997	5997	5998	06/20/2019	114	19	9	$205.47
5998	5998	5999	06/20/2019	404	11	20	$429.40
5999	5999	6000	06/20/2019	88	57	4	$274.52

6000 rows × 7 columns

```
# Importing the DataSet.....
try:
    customers = pd.read_csv('customers.csv')
    products = pd.read_csv('products.csv')
    purchases = pd.read_csv('purchases.csv')
except FileNotFoundError:
    print("File Not Found!")
```

```
# Finding the Statistical Analysis of Customer Table
customers.describe()
```

	id	street_num	postcode
count	1000.000000	738.000000	843.000000
mean	500.500000	10536.439024	52669.548043
std	288.819436	23050.537603	28140.041026
min	1.000000	0.000000	214.000000
25%	250.750000	21.000000	29279.500000
50%	500.500000	445.500000	48232.000000
75%	750.250000	6918.500000	78337.500000
max	1000.000000	99918.000000	99812.000000

```
products.describe()
```

	Unnamed: 0	id
count	60.000000	60.000000
mean	29.500000	30.500000
std	17.464249	17.464249
min	0.000000	1.000000
25%	14.750000	15.750000
50%	29.500000	30.500000
75%	44.250000	45.250000
max	59.000000	60.000000

```
purchases.describe()
```

	Unnamed: 0	id	customer_num	product_num	amount
count	6000.000000	6000.000000	6000.000000	6000.000000	6000.000000
mean	2999.500000	3000.500000	500.889333	30.140667	10.576167
std	1732.195139	1732.195139	288.377188	17.249613	5.768889
min	0.000000	1.000000	1.000000	1.000000	1.000000
25%	1499.750000	1500.750000	244.000000	15.000000	6.000000
50%	2999.500000	3000.500000	511.000000	30.000000	11.000000
75%	4499.250000	4500.250000	751.000000	45.000000	16.000000
max	5999.000000	6000.000000	1000.000000	60.000000	20.000000

```
customers['gender'].describe() # Series ['Categorical Col']
```

```
count      957
unique       2
top       Male
freq       485
Name: gender, dtype: object
```

```
customers['gender'].value_counts()
```

```
gender
Male      485
Female    472
Name: count, dtype: int64
```

```
customers.describe(include = [object]) # Generate descriptive statistics.
# Descriptive statistics include those that summarize the central tendency, dispersion and shape of a
# dataset's distribution, excluding ``NaN`` values.
```

	first_name	last_name	email	gender	street_name	street_suffix	city	state
count	1000	1000	878	957	963	963	921	920
unique	932	993	878	2	427	21	296	48
top	Berty	Sedworth	rsouthcott0@clickbank.net	Male	Arizona	Place	Washington	Texas
freq	4	3	1	485	6	57	29	107

```
# Top 10 states as per the total number of customers [high to low]
customers['state'].value_counts().head(10)
```

```
state
Texas                   107
California              101
Florida                  75
New York                 43
Ohio                     41
Pennsylvania             35
District of Columbia     33
Georgia                  28
Virginia                 27
Illinois                 23
Name: count, dtype: int64
```

```
customers['last_name'].value_counts().head()
```

```
last_name
Sedworth     3
Swatridge    2
Robardey     2
Truitt       2
Sunnex       2
Name: count, dtype: int64
```

```
customers['first_name'].value_counts().head()
```

```
first_name
Berty      4
Leland     3
Abel       3
Gunther    2
Silvain    2
Name: count, dtype: int64
```

```
customers['street_suffix'].value_counts().head()
```

```
street_suffix
Place      57
Street     52
Hill       52
Parkway    51
Circle     51
Name: count, dtype: int64
```

```
customers['city'].value_counts().head()
```

```
city
Washington       29
El Paso          19
Dallas           15
Houston          14
New York City    13
Name: count, dtype: int64
```

```
customers.describe(include = 'all')
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix
count	1000.000000	1000	1000	878	957	738.000000	963	963
unique	NaN	932	993	878	2	NaN	427	21
top	NaN	Berty	Sedworth	rsouthcott0@clickbank.net	Male	NaN	Arizona	Place
freq	NaN	4	3	1	485	NaN	6	57
mean	500.500000	NaN	NaN	NaN	NaN	10536.439024	NaN	NaN
std	288.819436	NaN	NaN	NaN	NaN	23050.537603	NaN	NaN
min	1.000000	NaN	NaN	NaN	NaN	0.000000	NaN	NaN
25%	250.750000	NaN	NaN	NaN	NaN	21.000000	NaN	NaN
50%	500.500000	NaN	NaN	NaN	NaN	445.500000	NaN	NaN
75%	750.250000	NaN	NaN	NaN	NaN	6918.500000	NaN	NaN
max	1000.000000	NaN	NaN	NaN	NaN	99918.000000	NaN	NaN

```
# .info() feature -> DESC TABLENAME in SQL
customers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             1000 non-null   int64
 1   first_name     1000 non-null   object
 2   last_name      1000 non-null   object
 3   email          878 non-null    object
 4   gender         957 non-null    object
 5   street_num     738 non-null    float64
 6   street_name    963 non-null    object
 7   street_suffix  963 non-null    object
 8   city           921 non-null    object
 9   state          920 non-null    object
 10  postcode       843 non-null    float64
dtypes: float64(2), int64(1), object(8)
memory usage: 86.1+ KB
```

```
products.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 60 entries, 0 to 59
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  60 non-null     int64
 1   id          60 non-null     int64
 2   product     60 non-null     object
 3   cost        60 non-null     object
 4   company     55 non-null     object
dtypes: int64(2), object(3)
memory usage: 2.5+ KB
```

```
purchases.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6000 entries, 0 to 5999
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Unnamed: 0    6000 non-null   int64 
 1   id            6000 non-null   int64 
 2   purch_date    6000 non-null   object
 3   customer_num  6000 non-null   int64 
 4   product_num   6000 non-null   int64 
 5   amount        6000 non-null   int64 
 6   paid          6000 non-null   object
dtypes: int64(5), object(2)
memory usage: 328.3+ KB
```

```
# .iloc[Positional Based Indexing] vs .loc[Labelled Based Indexing] [Inclusive]
# 2D Matrix [rows , cols]
# Find all the records of Customers with FirstName Columns
FirstName = customers.iloc[: , 1]
FirstName # Series
```

```
0       Romain
1       Cosimo
2        Bambi
3       Roarke
4      Mikaela
        ...
995    Merrili
996    Winonah
997      Tobit
998     Issiah
999     Elmore
Name: first_name, Length: 1000, dtype: object
```

```
type(FirstName)
```

```
pandas.core.series.Series
```

```
# .loc [label Based] [rows , cols]
FirstName = customers.loc[: , 'first_name']
FirstName # Series
```

```
0       Romain
1       Cosimo
2        Bambi
3       Roarke
4      Mikaela
        ...
995    Merrili
996    Winonah
997      Tobit
998     Issiah
999     Elmore
Name: first_name, Length: 1000, dtype: object
```

```
Email = customers.loc[: , 'email']
Email
```

```
0            rsouthcott0@clickbank.net
1                cmolyneaux1@wiley.com
2               bwestrip2@symantec.com
3               rpankettman3@wiley.com
4                    malthorpe4@51.la
                        ...
995              malmanrn@cornell.edu
996                 whecklero@fc2.com
997                               NaN
998    istandbrookerq@yellowpages.com
999                               NaN
Name: email, Length: 1000, dtype: object
```

```
# Provide the first 10 rows of Customers table with some specific columns
# Feature Selections
customers_info = customers.loc[1:10 , ['first_name' , 'last_name' , 'email' , 'gender' , 'city']]
customers_info # DataFrame
```

	first_name	last_name	email	gender	city
1	Cosimo	Molyneaux	cmolyneaux1@wiley.com	Male	El Paso
2	Bambi	Westrip	bwestrip2@symantec.com	Female	San Antonio
3	Roarke	Pankettman	rpankettman3@wiley.com	Male	Memphis
4	Mikaela	Althorpe	malthorpe4@51.la	NaN	NaN
5	Magdalena	Cullip	mcullip5@tiny.cc	Female	Baltimore
6	Marietta	Heball	mheball6@blog.com	Female	Carol Stream
7	Tine	McSperrin	tmcsperrin7@statcounter.com	NaN	Kansas City
8	Enrichetta	de Villier	edevillier8@ox.ac.uk	Female	Fairbanks
9	Sari	Poulden	spoulden9@xing.com	Female	New Orleans
10	Natale	Martina	nmartinaa@wordpress.com	Male	Bradenton

```
customers_info = customers.iloc[1:11 , 1:5] # [1st till 10th rows] [1st till 4th columns]
customers_info # DataFrame
```

	first_name	last_name	email	gender
1	Cosimo	Molyneaux	cmolyneaux1@wiley.com	Male
2	Bambi	Westrip	bwestrip2@symantec.com	Female
3	Roarke	Pankettman	rpankettman3@wiley.com	Male
4	Mikaela	Althorpe	malthorpe4@51.la	NaN
5	Magdalena	Cullip	mcullip5@tiny.cc	Female
6	Marietta	Heball	mheball6@blog.com	Female
7	Tine	McSperrin	tmcsperrin7@statcounter.com	NaN
8	Enrichetta	de Villier	edevillier8@ox.ac.uk	Female
9	Sari	Poulden	spoulden9@xing.com	Female
10	Natale	Martina	nmartinaa@wordpress.com	Male

```
customers_info = customers.iloc[1:11 , [1,3,4,8,9]]
customers_info
```

	first_name	email	gender	city	state
1	Cosimo	cmolyneaux1@wiley.com	Male	El Paso	Texas
2	Bambi	bwestrip2@symantec.com	Female	San Antonio	Texas
3	Roarke	rpankettman3@wiley.com	Male	Memphis	Tennessee
4	Mikaela	malthorpe4@51.la	NaN	NaN	NaN
5	Magdalena	mcullip5@tiny.cc	Female	Baltimore	Maryland
6	Marietta	mheball6@blog.com	Female	Carol Stream	Illinois
7	Tine	tmcsperrin7@statcounter.com	NaN	Kansas City	Missouri
8	Enrichetta	edevillier8@ox.ac.uk	Female	Fairbanks	Alaska
9	Sari	spoulden9@xing.com	Female	New Orleans	Louisiana
10	Natale	nmartinaa@wordpress.com	Male	Bradenton	Florida

```
# First 10 records with all columns
customers_info = customers.iloc[:10 , :]
customers_info
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city	state
0	1	Romain	Southcott	rsouthcott0@clickbank.net	Male	1.0	Trailsway	Road	San Diego	California
1	2	Cosimo	Molyneaux	cmolyneaux1@wiley.com	Male	NaN	NaN	NaN	El Paso	Texas
2	3	Bambi	Westrip	bwestrip2@symantec.com	Female	4057.0	Arkansas	Circle	San Antonio	Texas
3	4	Roarke	Pankettman	rpankettman3@wiley.com	Male	74.0	Debs	Point	Memphis	Tennessee
4	5	Mikaela	Althorpe	malthorpe4@51.la	NaN	NaN	2nd	Drive	NaN	NaN
5	6	Magdalena	Cullip	mcullip5@tiny.cc	Female	9190.0	Packers	Drive	Baltimore	Maryland
6	7	Marietta	Heball	mheball6@blog.com	Female	12.0	Mallory	Center	Carol Stream	Illinois
7	8	Tine	McSperrin	tmcsperrin7@statcounter.com	NaN	530.0	Erie	Plaza	Kansas City	Missouri
8	9	Enrichetta	de Villier	edevillier8@ox.ac.uk	Female	745.0	Annamark	Street	Fairbanks	Alaska
9	10	Sari	Poulden	spoulden9@xing.com	Female	2.0	Pond	Hill	New Orleans	Louisiana

```
# Slicing
customers_info = customers.iloc[1:11 , 1:15:2] # Starts from first_name and fetch every alternative columns
customers_info
```

	first_name	email	street_num	street_suffix	state
1	Cosimo	cmolyneaux1@wiley.com	NaN	NaN	Texas
2	Bambi	bwestrip2@symantec.com	4057.0	Circle	Texas
3	Roarke	rpankettman3@wiley.com	74.0	Point	Tennessee
4	Mikaela	malthorpe4@51.la	NaN	Drive	NaN
5	Magdalena	mcullip5@tiny.cc	9190.0	Drive	Maryland
6	Marietta	mheball6@blog.com	12.0	Center	Illinois
7	Tine	tmcsperrin7@statcounter.com	530.0	Plaza	Missouri
8	Enrichetta	edevillier8@ox.ac.uk	745.0	Street	Alaska
9	Sari	spoulden9@xing.com	2.0	Hill	Louisiana
10	Natale	nmartinaa@wordpress.com	221.0	Pass	Florida

```
# Accessing a single row
# using .loc [labelled based]
customers.loc[999]
```

```
id                  1000
first_name         Elmore
last_name          Malpas
email                 NaN
gender               Male
street_num          205.0
street_name       Farwell
street_suffix        Park
city              Atlanta
state             Georgia
postcode          30386.0
Name: 999, dtype: object
```

```
type(customers.loc[999])
```

```
pandas.core.series.Series
```

```
customers.iloc[999]
```

```
id                  1000
first_name         Elmore
last_name          Malpas
email                 NaN
gender               Male
street_num          205.0
street_name       Farwell
street_suffix        Park
city              Atlanta
state             Georgia
postcode          30386.0
Name: 999, dtype: object
```

```
# Accessing a single row -> DataFrame
# using .loc [label based indexing]
# Select * From Customers Where id = 1000
customers.loc[customers['id'] == 1000]
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city	state	postcode
999	1000	Elmore	Malpas	NaN	Male	205.0	Farwell	Park	Atlanta	Georgia	30386.0

```
type(customers.loc[customers['id'] == 1000])
```

```
pandas.core.frame.DataFrame
```

```
# Boolean Returns [True / False]
customers.loc[customers['id'] == 1001]
```

id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city	state	postcode

```
customers['last_name'].value_counts().head()
```

```
last_name
Sedworth     3
Swatridge    2
Robardey     2
Truitt       2
Sunnex       2
Name: count, dtype: int64
```

```
# -> Filtering Rows
customers_info = customers[customers['last_name'] == 'Sedworth']
customers_info
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city	state
554	555	Kimble	Sedworth	ksedworthfe@parallels.com	Male	1581.0	Morningstar	Trail	Denver	Colorado
631	632	Leif	Sedworth	lsedworthhj@myspace.com	Male	27823.0	Laurel	Lane	Shreveport	Louisiana
923	924	Xymenes	Sedworth	xsedworthpn@so-net.ne.jp	Male	4148.0	Lotheville	Hill	Charlotte	North Carolina

```
customers['state'].value_counts().head()
```

```
state
Texas         107
California    101
Florida        75
New York       43
Ohio           41
Name: count, dtype: int64
```

```
customers['state'] == 'Texas' # Series
```

```
0       False
1        True
2        True
3       False
4       False
        ...
995     False
996     False
997     False
998     False
999     False
Name: state, Length: 1000, dtype: bool
```

```
# Filtering Rows
customers[customers['state'] == 'Texas']
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city	state
1	2	Cosimo	Molyneaux	cmolyneaux1@wiley.com	Male	NaN	NaN	NaN	El Paso	Texas
2	3	Bambi	Westrip	bwestrip2@symantec.com	Female	4057.0	Arkansas	Circle	San Antonio	Texas
17	18	Reinhold	Woolforde	rwoolfordeh@cbc.ca	Male	NaN	Northwestern	Way	Amarillo	Texas
19	20	Rubina	Hustings	rhustingsj@wikimedia.org	Female	249.0	Esker	Hill	San Antonio	Texas
20	21	Karim	Woosnam	kwoosnamk@ifeng.com	Male	NaN	Grayhawk	Place	Houston	Texas
...	...	...	...	...	...	...	...	...	...	...
956	957	Ruddie	Eckhard	reckhardqk@free.fr	Male	853.0	Little Fleur	Trail	Abilene	Texas
969	970	Georgetta	Bartoszek	gbartoszekqx@is.gd	Female	4023.0	Donald	Street	NaN	Texas

```
customers['city'].value_counts().head()
```

```
city
Washington       29
El Paso          19
Dallas           15
Houston          14
New York City    13
Name: count, dtype: int64
```

```
customers[customers['city'] == 'Washington']
```

last_name	email	gender	street_num	street_name	street_suffix	city	state	postcode
Thraves	dthravesd@ibm.com	Female	71865.0	Valley Edge	Place	Washington	District of Columbia	20456.0
Massinger	NaN	Male	9.0	Victoria	Trail	Washington	District of Columbia	20409.0
Armal	earmalz@dailymail.co.uk	Male	4.0	Susan	Center	Washington	District of Columbia	20231.0
Mendus	emendus10@ocn.ne.jp	Female	127.0	Mockingbird	Hill	Washington	District of Columbia	NaN

```
customers[customers['city'] == 'Washington'].shape
```

```
(29, 11)
```

```
customers[customers['city'] == 'Washington'].ndim
```

```
2
```

```
# Find out the customers who are Male and lives in California [AND Logic]
customers_info = customers[(customers['gender'] == 'Male') & (customers['state'] == 'California')]
customers_info
```

last_name	email	gender	street_num	street_name	street_suffix	city	state	postcode
Southcott	rsouthcott0@clickbank.net	Male	1.0	Trailsway	Road	San Diego	California	92127.0
Pantridge	NaN	Male	6124.0	Kropf	Street	Los Angeles	California	90065.0
Athowe	bathowe24@dagondesign.com	Male	9500.0	Coleman	Terrace	Pasadena	California	91131.0
Pladen	gpladen2x@cornell.edu	Male	13793.0	Oakridge	Place	Inglewood	California	90398.0
Gladyer	jgladyer3p@cafepress.com	Male	48173.0	Arizona	Street	San Francisco	California	94132.0
Aleshintsev	faleshintsev3q@google.it	Male	463.0	Sunbrook	Point	Sacramento	California	95813.0
Swaddle	cswaddle4c@redcross.org	Male	7.0	Ridgeview	Place	Los Angeles	California	90040.0

```
customers_info.shape
```

```
(57, 11)
```

```
# SELECT * FROM Customers WHERE Gender = 'female' and state = 'Florida'
gender_filter = (customers['gender'] == 'Female')
state_filter = (customers['state'] == 'Florida')
customers_info = customers[gender_filter & state_filter]
customers_info
```

ame	last_name	email	gender	street_num	street_name	street_suffix	city	state	postcode
ielyn	Naisey	jnaisey29@naver.com	Female	3115.0	Pearson	Park	Miami	Florida	33233.0
Dorie	Dutt	ddutt2t@hatena.ne.jp	Female	NaN	Dwight	Center	Sarasota	Florida	34276.0
aren	Sings	NaN	Female	NaN	Stoughton	Circle	Lakeland	Florida	33805.0
Gray	Gagan	NaN	Female	1122.0	Hooker	Road	Pompano Beach	Florida	33075.0
ndie	Fawdrey	cfawdrey62@umn.edu	Female	3295.0	Gale	Place	Lakeland	Florida	33811.0
Nedi	Pettisall	npettisall72@simplemachines.org	Female	72.0	Cambridge	Pass	Miami	Florida	33158.0
Berty	Vaillant	bvaillant8m@blog.com	Female	0.0	Drewry	Terrace	Saint Petersburg	Florida	33705.0
nata	Inde	NaN	Female	6044.0	Straight	Point	Pinellas	Florida	34665.0

```
customers_info.shape
```

```
(35, 11)
```

```
products['company'].isna() # Series
```

```
44    False
45     True
46    False
47    False
48    False
49    False
50    False
51    False
52    False
53    False
54    False
55    False
56    False
57    False
58    False
59    False
Name: company, dtype: bool
```

```
products['company'].isna().sum()
```

```
5
```

```
products[products['company'].isna()] # DataFrame
```

	Unnamed: 0	id	product	cost	company
15	15	16	Sausage - Chorizo	$55.45	NaN
21	21	22	Scotch - Queen Anne	$60.26	NaN
26	26	27	Spaghetti Squash	$47.41	NaN
27	27	28	Wine - Niagara,vqa Reisling	$28.24	NaN
45	45	46	Aromat Spice / Seasoning	$61.34	NaN

```
# pick 5 company from product DataFrame and apply the filter
Skipfire_filter = products['company'] == 'Skipfire'
Ntag_filter = products['company'] == 'Ntag'
Livepath_filter = products['company'] == 'Livepath'
Zoombox_filter = products['company'] == 'Zoombox'
Quatz_filter = products['company'] == 'Quatz'

products[Skipfire_filter | Ntag_filter | Livepath_filter | Zoombox_filter | Quatz_filter]
```

	Unnamed: 0	id	product	cost	company
0	0	1	Liners - Baking Cups	$6.36	Skipfire
2	2	3	Bar Bran Honey Nut	$65.40	Ntag
5	5	6	Wine - White, Riesling, Semi - Dry	$99.22	Livepath
9	9	10	Sambuca - Ramazzotti	$88.99	Livepath
22	22	23	Puree - Blackcurrant	$16.27	Zoombox
30	30	31	Wine - Carmenere Casillero Del	$55.77	Ntag
35	35	36	Bread - English Muffin	$13.83	Quatz

```
Boolean Filters Summary :
df[df['column'] == 'value'] # Equals
df[(df['column'] == 'value') & df['column'] == 'value')] # And Logic
df[(df['column'] == 'value') | df['column'] == 'value')] # OR Logic
df[~(df['column'] == 'value')] # Not Logic
```

```
customers['gender'].unique()
```

```
array(['Male', 'Female', nan], dtype=object)
```

```
# Don't include counts of rows that contain NA values.
customers['gender'].value_counts()
```

```
gender
Male      485
Female    472
Name: count, dtype: int64
```

```
customers[customers['gender'] == 'Male'] # DataFrame # 485 records
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city
0	1	Romain	Southcott	rsouthcott0@clickbank.net	Male	1.0	Trailsway	Road	San Diego
1	2	Cosimo	Molyneaux	cmolyneaux1@wiley.com	Male	NaN	NaN	NaN	El Paso
3	4	Roarke	Pankettman	rpankettman3@wiley.com	Male	74.0	Debs	Point	Memphis
10	11	Natale	Martina	nmartinaa@wordpress.com	Male	221.0	Sauthoff	Pass	Bradenton
12	13	Mikol	MacWhan	mmacwhanc@patch.com	Male	9.0	Eagle Crest	Center	Bronx
...	...	...	...	...	...	...	...	...	...
984	985	Tailor	Sealeaf	tsealeafrc@irs.gov	Male	NaN	Summer Ridge	Pass	Beaufor
986	987	Eric	Mountford	emountfordre@blogspot.com	Male	NaN	6th	Court	Las Vegas
997	998	Tobit	Birt	NaN	Male	2.0	Basil	Road	Waterbury

```
# Not Logic # ~(customers['gender'] == 'Male') # 'Female','NaN' # 515 records
customers[~(customers['gender'] == 'Male')]
```

	id	first_name	last_name	email	gender	street_num	street_name	street_suffix	city	
2	3	Bambi	Westrip	bwestrip2@symantec.com	Female	4057.0	Arkansas	Circle	San Antonio	
4	5	Mikaela	Althorpe	malthorpe4@51.la	NaN	NaN	2nd	Drive	NaN	
5	6	Magdalena	Cullip	mcullip5@tiny.cc	Female	9190.0	Packers	Drive	Baltimore	M
6	7	Marietta	Heball	mheball6@blog.com	Female	12.0	Mallory	Center	Carol Stream	
7	8	Tine	McSperrin	tmcsperrin7@statcounter.com	NaN	530.0	Erie	Plaza	Kansas City	M
...	...	...	...	...	...	...	...	...	...	
992	993	Johnath	Clancy	jclancyrk@smugmug.com	Female	7.0	Washington	Crossing	Juneau	
993	994	Binnie	Dearth	bdearthrl@ed.gov	NaN	8993.0	Elgar	Trail	Minneapolis	Mir
994	995	Brana	Dixon	bdixonrm@myspace.com	Female	97.0	Truax	Avenue	Maple Plain	Mir
995	996	Merrili	Alman	malmanrn@cornell.edu	Female	0.0	Thompson	Way	Reading	Penns
996	997	Winonah	Heckle	whecklero@fc2.com	Female	701.0	Rowland	Hill	Indianapolis	

515 rows × 11 columns

```
customers[~(customers['gender'] == 'Male')].shape
```

```
(515, 11)
```

```
1000 - (485 + 472)
```

```
43
```

```
472 + 43
```

```
515
```

```
# Like Keyword in SQL # Wildcard [% , _]
# products['product'] LIKE '%Nut%',"%NuT%","%nuT%"
products[products['product'].str.contains('NUT' , case = False)] # case-sensitive
```

	Unnamed: 0	id	product	cost	company
2	2	3	Bar Bran Honey Nut	$65.40	Ntag
57	57	58	Bar Bran Honey Nut	$73.05	Yakijo
58	58	59	Nut - Almond, Blanched, Whole	$74.28	Eazzy

```
# Like Keyword in SQL # Wildcard [% , _]
# products['product'] LIKE 'Nut%' -> startswith
products[products['product'].str.startswith('Nut')]
```

	Unnamed: 0	id	product	cost	company
58	58	59	Nut - Almond, Blanched, Whole	$74.28	Eazzy

```
# Like Keyword in SQL # Wildcard [% , _]
# products['product'] LIKE '%Nut' -> endswith
products[products['product'].str.endswith('Nut')]
```

	Unnamed: 0	id	product	cost	company
2	2	3	Bar Bran Honey Nut	$65.40	Ntag
57	57	58	Bar Bran Honey Nut	$73.05	Yakijo

```
customers['gender'].isna()
```

```
0      False
1      False
2      False
3      False
4       True
       ...
995    False
996    False
997    False
998    False
999    False
Name: gender, Length: 1000, dtype: bool
```

```
customers['gender'].isna().sum()
```

```
43
```