What is Pandas in Python?

- Pandas is a Python library that is a simple yet powerful tool for Data Science. Python Pandas is one of the most widely used Python packages.
- This package comprises many data structures and tools for effective data manipulation and analysis. If we are going to work with data using Python, we need to learn Python Pandas as well.
- Python Pandas is used everywhere including commercial and academic sectors and in fields like economics, finance, analytics, statistics, etc.

Features of Python Pandas

Some of the key features of Python Pandas are as follows:

- It provides DataFrame objects with default and customized indexing which is very fast and efficient.
- There are tools available for loading data of different file formats into in-memory data objects.
- It is easy to perform data alignment and integrated handling of missing data in Python Pandas.
- It is very simple to perform pivoting and reshaping of data sets in Pandas.
- It also provides indexing, label-based slicing, and sub-setting of large data sets.
- We can easily insert and delete columns from a data structure.
- Data aggregation and transformations can be done using group by.
- High-performance merging and joining of data can be done using Pandas.
- It also provides time-series functionality.

Pandas 20 popular and the most important functions:

- **read_csv():** This function reads a CSV file and creates a pandas data frame. It has several options to handle various data formats and separators.
- **head():** This function returns the first n rows of a data frame. By default, it returns the first five rows.
- **tail():** This function returns the last n rows of a data frame. By default, it returns the last five rows.
- **info():** This function provides a summary of a data frame, including the number of rows, columns, data types, and memory usage.
- **describe():** This function provides a statistical summary of a data frame, including the mean, standard deviation, minimum, maximum, and quartile values of each numerical column.
- **dropna():** This function removes rows with missing values from a data frame. It has several options to handle various missing data formats.

- **fillna():** This function fills missing values in a data frame with a specified value or method.
- **groupby():** This function groups a data frame by one or more columns and applies a function to each group.
- **pivot_table():** This function creates a pivot table from a data frame, which summarizes the data by one or more columns.
- **merge():** This function combines two or more data frames based on a common column or index.
- **apply():** This function applies a function to each element of a data frame or series.
- **map():** This function applies a function to each element of a series and returns a new series.
- **value_counts():** This function returns the count of unique values in a series.
- **unique():** This function returns an array of unique values in a series.
- **loc[]:** This function is used to access a group of rows and columns by label(s) or a boolean array.
- **iloc[]:** This function is used to access a group of rows and columns by integer location(s).
- **set_index():** This function sets a column or a list of columns as the index of a data frame.
- **reset_index():** This function resets the index of a data frame to the default integer index.
- **sort_values():** This function sorts a data frame by one or more columns.
- **to_csv():** This function writes a data frame to a CSV file. It has several options to handle various data formats and separators.

In [ ]:

In [1]:
```
!pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\admin\anaconda3\lib\site-packages (1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\admin\anaconda3\lib\site-packages (from
pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib\site-packages (from pandas)
(2022.7)
Requirement already satisfied: numpy>=1.21.0 in c:\users\admin\anaconda3\lib\site-packages (from pandas)
(1.24.3)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site-packages (from python-dateut
il>=2.8.1->pandas) (1.16.0)
```

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

Series

In [3]:
```
lst = [4,7,2,4,1,6,8]
lst
```

Out[3]:
```
[4, 7, 2, 4, 1, 6, 8]
```

In [4]:
```
type(lst)
```

Out[4]:
```
list
```

In [7]:
```
ser = pd.Series([4,7,2,4,1,6,8])
print(ser)
```

```
0    4
1    7
2    2
3    4
4    1
5    6
6    8
dtype: int64
```

In [13]:
```
type(ser)
```

Out[13]:
```
pandas.core.series.Series
```

In [8]:
```
ser = pd.Series([4,7,2,4,1,6,8], name='Roll No')
ser
```

Out[8]:
```
0    4
1    7
2    2
3    4
4    1
5    6
6    8
Name: Roll No, dtype: int64
```

In [9]:
```
ser = pd.Series([4,7,2,1,6,8], index=['a', 'b', 'c', 'd', 'e', 'f'], name='Roll No')
ser
```

Out[9]:
```
a    4
b    7
c    2
d    1
e    6
f    8
Name: Roll No, dtype: int64
```

In [12]:
```
ser = pd.Series(15, index=['a', 'b', 'c', 'd', 'e', 'f'], name='Roll No')
ser
```

Out[12]:
```
a    15
b    15
c    15
d    15
e    15
f    15
Name: Roll No, dtype: int64
```

In [ ]:
```

```

DataFrame

Using List

In [14]:
```
lst_of_lst = [['apple', 8], ['banana', 12], ['mango', 6]]
lst_of_lst
```

Out[14]:
```
[['apple', 8], ['banana', 12], ['mango', 6]]
```

In [15]:
```
lst_of_lst[0]
```

Out[15]:
```
['apple', 8]
```

In [16]:
```
lst_of_lst[0][1]
```

Out[16]: 8

In [17]:
```
lst_of_lst[2][0]
```

Out[17]: 'mango'

In [ ]:
```

```

In [18]:
```
lst_of_lst = [['apple', 8], ['banana', 12], ['mango', 6]]

df = pd.DataFrame(lst_of_lst)
df
```

Out[18]:

	0	1
0	apple	8
1	banana	12
2	mango	6

In [19]:
```
type(df)
```

Out[19]: pandas.core.frame.DataFrame

In [21]:
```
lst_of_lst = [['apple', 8], ['banana', 12], ['mango', 6], ['orange', 9]]

df = pd.DataFrame(lst_of_lst, columns=['Fruits', 'Quantities'])
df
```

Out[21]:

	Fruits	Quantities
0	apple	8
1	banana	12
2	mango	6
3	orange	9

In [23]:
```
lst_of_lst = [['apple', 8, 'L'], ['banana', 12, 'D'], ['mango', 6, 'L'], ['orange', 9, 'D']]

df = pd.DataFrame(lst_of_lst, columns=['Fruits', 'Quantities', 'Like/Dislike'])
df
```

Out[23]:

	Fruits	Quantities	Like/Dislike
0	apple	8	L
1	banana	12	D
2	mango	6	L
3	orange	9	D

In [33]:
```
df.ndim
```

Out[33]:
```
2
```

In [24]:
```
type(df)
```

Out[24]:
```
pandas.core.frame.DataFrame
```

In [27]:
```
df['Fruits']
```

Out[27]:
```
0     apple
1    banana
2     mango
3    orange
Name: Fruits, dtype: object
```

In [28]:
```
df.Fruits
```

Out[28]:
```
0     apple
1    banana
2     mango
3    orange
Name: Fruits, dtype: object
```

In [26]:
```
type(df['Fruits'])
```

Out[26]:
```
pandas.core.series.Series
```

In [29]:
```
df['Fruits'].ndim
```

Out[29]:
```
1
```

In [ ]:
```

```

In [31]:
```
d  = df[['Fruits', 'Quantities']]
d
```

Out[31]:

	Fruits	Quantities
0	apple	8
1	banana	12
2	mango	6
3	orange	9

In [32]:
```
d.ndim
```

Out[32]:
```
2
```

In [ ]:
```

```

Same Dataframe - Using Dictionary

	Fruits	Quantities	Like/Dislike
0	apple	8	L
1	banana	12	D
2	mango	6	L
3	orange	9	D

In [34]:
```
d = {

    'Fruits': ['apple', 'banana', 'mango', 'orange'],
    'Quantities': [8, 12, 6, 9],
    'Like/Dislike': ['L', 'D', 'L', 'D']

    }

d
```

Out[34]:
```
{'Fruits': ['apple', 'banana', 'mango', 'orange'],
 'Quantities': [8, 12, 6, 9],
 'Like/Dislike': ['L', 'D', 'L', 'D']}
```

In [35]:
```
df1  = pd.DataFrame(d)
df1
```

Out[35]:

	Fruits	Quantities	Like/Dislike
0	apple	8	L
1	banana	12	D
2	mango	6	L
3	orange	9	D

In [36]:
```
df1['Like/Dislike']
```

Out[36]:
```
0    L
1    D
2    L
3    D
Name: Like/Dislike, dtype: object
```

In [37]:
```
df1['Price']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3802, in Index.get_loc(self, key, metho
d, tolerance)
   3801 try:
-> 3802     return self._engine.get_loc(casted_key)
   3803 except KeyError as err:

File ~\anaconda3\Lib\site-packages\pandas\_libs\index.pyx:138, in pandas._libs.index.IndexEngine.get_l
oc()

File ~\anaconda3\Lib\site-packages\pandas\_libs\index.pyx:165, in pandas._libs.index.IndexEngine.get_l
oc()

File pandas\_libs\hashtable_class_helper.pxi:5745, in pandas._libs.hashtable.PyObjectHashTable.get_ite
m()

File pandas\_libs\hashtable_class_helper.pxi:5753, in pandas._libs.hashtable.PyObjectHashTable.get_ite
m()
```

In [ ]:
```

```

In [38]:
```
df1['Price'] = [100, 40, 60, 30]
```

In [39]:
```
df1
```

Out[39]:

	Fruits	Quantities	Like/Dislike	Price
0	apple	8	L	100
1	banana	12	D	40
2	mango	6	L	60
3	orange	9	D	30

In [40]:
```
df1['Price']
```

Out[40]:
```
0    100
1     40
2     60
3     30
Name: Price, dtype: int64
```

In [41]:
```
df1.Price
```

Out[41]:
```
0    100
1     40
2     60
3     30
Name: Price, dtype: int64
```

In [ ]:
```

```

Arithmatic Operations

In [43]:
```
d1 = pd.DataFrame({'A': [1,2,3,4,5], 'B':[6,7,8,9,10]})
d1
```

Out[43]:

	A	B
0	1	6
1	2	7
2	3	8
3	4	9
4	5	10

In [47]:
```
d1['C'] = d1['A'] * d1['B']
```

In [48]:
```
d1
```

Out[48]:

	A	B	C
0	1	6	6
1	2	7	14
2	3	8	24
3	4	9	36
4	5	10	50

In [49]:
```
d1['D'] = d1['A'] + d1['C']
```

In [50]:
```
d1
```

Out[50]:

	A	B	C	D
0	1	6	6	7
1	2	7	14	16
2	3	8	24	27
3	4	9	36	40
4	5	10	50	55

In [ ]:
```

```

Insert and Delete

In [51]:
```
d1
```

Out[51]:

	A	B	C	D
0	1	6	6	7
1	2	7	14	16
2	3	8	24	27
3	4	9	36	40
4	5	10	50	55

In [ ]:
```
# syntax: d1.insert(index_position, col_name, value)
```

In [53]:
```
d1.insert(1, 'Alphabets', ['a', 'b', 'c', 'd', 'e'])
```

In [54]:
```
d1
```

Out[54]:

	A	Alphabets	B	C	D
0	1	a	6	6	7
1	2	b	7	14	16
2	3	c	8	24	27
3	4	d	9	36	40
4	5	e	10	50	55

In [ ]:
```

```

In [ ]:
```
# Delete
```

In [56]:
```
d1.pop('C')
```

Out[56]:
```
0     6
1    14
2    24
3    36
4    50
Name: C, dtype: int64
```

In [57]:
```
d1
```

Out[57]:

	A	Alphabets	B	D
0	1	a	6	7
1	2	b	7	16
2	3	c	8	27
3	4	d	9	40
4	5	e	10	55

In [58]:
```
del d1['Alphabets']
```

In [59]:
```
d1
```

Out[59]:

	A	B	D
0	1	6	7
1	2	7	16
2	3	8	27
3	4	9	40
4	5	10	55

In [60]:
```
del d1
```

In [61]:
```
d1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[61], line 1
----> 1 d1

NameError: name 'd1' is not defined
```

In [ ]:
```

```

Data Format - Excel, CSV, JSON, XML, HTML etc

Reading a data

In [62]:
```
data = pd.read_csv('petrol_consumption.csv', nrows=10)
data
```

Out[62]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
0	9.0	3571.0	1976.0	0.525	541.0
1	9.0	4092.0	1250.0	0.572	524.0
2	9.0	NaN	NaN	0.580	561.0
3	7.5	4870.0	2351.0	0.529	NaN
4	8.0	4399.0	431.0	0.544	410.0
5	10.0	5342.0	1333.0	0.571	457.0
6	8.0	5319.0	11868.0	0.451	344.0
7	8.0	5126.0	2138.0	0.553	467.0
8	8.0	4447.0	8577.0	0.529	464.0
9	7.0	4512.0	8507.0	0.552	498.0

In [63]:
```
data = pd.read_csv('petrol_consumption.csv', nrows=5, usecols=['Petrol_tax', 'Average_income'])
data
```

Out[63]:

	Petrol_tax	Average_income
0	9.0	3571.0
1	9.0	4092.0
2	9.0	NaN
3	7.5	4870.0
4	8.0	4399.0

In [66]:
```
d = pd.read_csv('D:\TB Resources\petrol_consumption.csv')  # Reading a local file
d
```

Out[66]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
0	9.00	3571.0	1976.0	0.525	NaN
1	9.00	4092.0	1250.0	0.369	524.0
2	9.00	3865.0	1586.0	0.580	561.0
3	7.50	4870.0	2351.0	0.529	414.0
4	8.00	NaN	431.0	0.544	410.0
5	10.00	5342.0	1333.0	0.571	457.0
6	8.00	5319.0	11868.0	0.451	344.0
7	8.00	5126.0	2138.0	0.553	467.0
8	8.00	NaN	8577.0	0.529	464.0
9	7.00	4512.0	8507.0	0.552	498.0
10	8.00	4391.0	5939.0	0.530	580.0

In [ ]:

In [ ]:

Reading the entire Data

In [64]:
```
data = pd.read_csv('petrol_consumption.csv')
data
```

Out[64]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
0	9.00	3571.0	1976.0	0.525	541.0
1	9.00	4092.0	1250.0	0.572	524.0
2	9.00	NaN	NaN	0.580	561.0
3	7.50	4870.0	2351.0	0.529	NaN
4	8.00	4399.0	431.0	0.544	410.0
5	10.00	5342.0	1333.0	0.571	457.0
6	8.00	5319.0	11868.0	0.451	344.0
7	8.00	5126.0	2138.0	0.553	467.0
8	8.00	4447.0	8577.0	0.529	464.0
9	7.00	4512.0	8507.0	0.552	498.0
10	8.00	4391.0	5939.0	0.530	580.0

In [ ]:

In [ ]:

In [ ]:

In [ ]: