Pandas_Assignment

April 1, 2023

Q1. Create a Pandas Series that contains the following data: 4, 8, 15, 16, 23, and 42. Then, print the series.

[1]:
```
import pandas as pd
```

[2]:
```
data=[4,8,15,16,23,42]
df=pd.Series(data)
df
```

[2]:
```
0     4
1     8
2    15
3    16
4    23
5    42
dtype: int64
```

Q2. Create a variable of list type containing 10 elements in it, and apply pandas.Series function on the variable print it.

[6]:
```
import pandas as pd
my_list=[1,2,3,4,5,6,7,8,9,10]
series=pd.Series(my_list)
series
```

[6]:
```
0     1
1     2
2     3
3     4
4     5
5     6
6     7
7     8
8     9
9    10
dtype: int64
```

Q.3 Create a Pandas DataFrame that contains the following data:

Then, print the DataFrame.

[7]:
```
dict={"Name":['Alice','Bob','Claire'],'Age':[25,30,27] ,'Gender':
 ↪['Female','male','Female']}
df= pd.DataFrame(dict)
```

[8]:
```
df
```

[8]:
```
     Name  Age  Gender
0   Alice   25  Female
1     Bob   30    male
2  Claire   27  Female
```

[10]:
```
df.loc[0]
```

[10]:
```
Name       Alice
Age           25
Gender    Female
Name: 0, dtype: object
```

Q.4

In pandas, a DataFrame is a 2-dimensional labeled data structure with columns of potentially different data types. It is similar to a spreadsheet or SQL table, where the data is organized in rows and columns, with each column representing a variable or feature and each row representing an observation or instance.

On the other hand, a pandas series is a one-dimensional labeled array capable of holding any data type, such as integers, floats, strings, and objects, among others.

The primary difference between a DataFrame and a Series is that a DataFrame can hold multiple columns of data, while a Series can only hold a single column of data. A Series can be thought of as a single column of a DataFrame.

[11]:
```
#pandas series
```

[12]:
```
import pandas as pd
list=[1,2,3,4,5]
df=pd.Series(list)
df
```

[12]:
```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

```
[13]: # Data Frame
      import pandas as pd
      dict={"Name":['Alice','Bob','Claire'],'Age':[25,30,27] ,'Gender':
       ↪['Female','male','Female']}
      df= pd.DataFrame(dict)
      df
```

```
[13]:      Name  Age  Gender
      0   Alice   25  Female
      1     Bob   30    male
      2  Claire   27  Female
```

Q5. What are some common functions you can use to manipulate data in a Pandas DataFrame? Can you give an example of when you might use one of these functions?

head() and tail(): These functions are used to view the top and bottom rows of the DataFrame, respectively. For example, you might use these functions to get a quick glimpse of the data or to check that the data has been loaded correctly.

```
[15]: import pandas as pd

      # Creating a pandas DataFrame object
      df = pd.DataFrame({
          'Name': ['John', 'Mary', 'Bob', 'Alice', 'Tom'],
          'Age': [25, 30, 20, 35, 40],
          'Gender': ['M', 'F', 'M', 'F', 'M']})
```

```
[16]: df.head()
```

```
[16]:     Name  Age Gender
      0   John   25      M
      1   Mary   30      F
      2    Bob   20      M
      3  Alice   35      F
      4    Tom   40      M
```

```
[18]: df.tail(2)
```

```
[18]:     Name  Age Gender
      3  Alice   35      F
      4    Tom   40      M
```

```
[19]: df.describe()
```

```
[19]:              Age
      count   5.000000
      mean   30.000000
      std     7.905694
```

```
min    20.000000
25%    25.000000
50%    30.000000
75%    35.000000
max    40.000000
```

[23]:

```
df.sort_values(by='Age')
```

[23]:

```
    Name  Age Gender
2    Bob   20      M
0   John   25      M
1   Mary   30      F
3  Alice   35      F
4    Tom   40      M
```

Q6. Which of the following is mutable in nature Series, DataFrame, Panel?

In Pandas, Series and DataFrame are mutable, whereas Panel is mutable up to version 0.25.0. From version 0.26.0, the Panel was deprecated and removed from the library, and developers were recommended to use MultiIndex or xarray for similar functionality.

Q7. Create a DataFrame using multiple Series. Explain with an example.

We can create a DataFrame using multiple Series in Pandas by passing the Series objects as a dictionary to the DataFrame constructor. Each key in the dictionary represents a column name, and the corresponding value represents the data for that column as a Series.

[24]:

```
import pandas as pd

names = pd.Series(['John', 'Mary', 'Bob', 'Alice', 'Tom'])
ages = pd.Series([25, 30, 20, 35, 40])
genders = pd.Series(['M', 'F', 'M', 'F', 'M'])
df = pd.DataFrame({'Name': names, 'Age': ages, 'Gender': genders})

print(df)
```

```
    Name  Age Gender
0   John   25      M
1   Mary   30      F
2    Bob   20      M
3  Alice   35      F
4    Tom   40      M
```

[ ]:

```

```