2

April 10, 2024

[5]:
```
import pandas as pd
```

Write a Pandas program to create and display a one-dimensional array-like object containing an array of data using Pandas module.

[5]:
```
x1 = pd.Series([1,2,3])
x1
```

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

Write a Pandas program to convert a Panda module Series to Python list and it's type.

[18]:
```
x2 = x1
x2list = x2.to_list()
print(x2list, x2.dtype)
```

```
[1, 2, 3] int64
```

Write a Pandas program to add, subtract, multiple and divide two Pandas Series. Sample Series: [2, 4, 6, 8, 10], [1, 3, 5, 7, 9]

[12]:
```
x3 = pd.Series([2, 4, 6, 8, 10])
x4 = pd.Series([1, 3, 5, 7, 9])
print(x3 + x4)
print(x3 - x4)
print(x3 * x4)
print(x3 / x4)
```

```
0     3
1     7
2    11
3    15
4    19
dtype: int64
0    1
1    1
2    1
```

```
3    1
4    1
dtype: int64
0     2
1    12
2    30
3    56
4    90
dtype: int64
0    2.000000
1    1.333333
2    1.200000
3    1.142857
4    1.111111
dtype: float64
```

Write a Pandas program to compare the elements of the two Pandas Series. Sample Series: [2, 4, 6, 8, 10], [1, 3, 5, 7, 10]

[6]:
```
x5 = pd.Series([2, 4, 6, 8, 10])
x6 = pd.Series([1, 3, 5, 7, 10])
for i in range(x5.size):
    if x5[i] < x6[i]:
        print("Element " + str(i + 1) + " from series 2 is bigger by " +␣
 ↪str(x6[i] - x5[i]))
    if x6[i] < x5[i]:
        print("Element " + str(i + 1) + " from series 1 is bigger by " +␣
 ↪str(x5[i] - x6[i]))
    else:
        print("Element " + str(i + 1) + " is equal")
```

```
Element 1 from series 1 is bigger by 1
Element 2 from series 1 is bigger by 1
Element 3 from series 1 is bigger by 1
Element 4 from series 1 is bigger by 1
Element 5 is equal
```

[8]:
```
equal = x5 == x6

greater_than = x5 > x6

less_than = x5 < x6

comparison_results = pd.DataFrame({
    'Equal': equal,
    'Greater than': greater_than,
    'Less than': less_than
```

```
})

comparison_results
```

[8]:

	Equal	Greater than	Less than
0	False	True	False
1	False	True	False
2	False	True	False
3	False	True	False
4	True	False	False

Write a Pandas program to convert a dictionary to a Pandas series. Sample Series: Original dictionary: {'a': 100, 'b': 200, 'c': 300, 'd': 400, 'e': 800} Converted series: a 100 b 200 c 300 d 400 e 800 dtype: int64

[14]:
```
x7 = {'a': 100, 'b': 200, 'c': 300, 'd': 400, 'e': 800}
x7series = pd.Series(x7)
x7series
```

[14]:
```
a    100
b    200
c    300
d    400
e    800
dtype: int64
```

Write a Pandas program to convert a NumPy array to a Pandas series. Sample Series: NumPy array: [10 20 30 40 50] Converted Pandas series: 0 10 1 20 2 30 3 40 4 50 dtype: int64

[4]:
```
import numpy as np
x8 = np.array([10, 20, 30, 40, 50])
x8series = pd.Series(x8)
x8series
```

[4]:
```
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

Write a Pandas program to change the data type of given a column or a Series. Sample Series: Original Data Series: 0 100 1 200 2 python 3 300.12 4 400 dtype: object Change the said data type to numeric: 0 100.00 1 200.00 2 NaN 3 300.12 4 400.00 dtype: float64

[33]:
```
x9 = pd.Series([100, 200, "python", 300.12, 400])
print(pd.to_numeric(x9, "coerce"))
```

```
0    100.00
1    200.00
```

```
2       NaN
3    300.12
4    400.00
dtype: float64
```

Write a Pandas program to convert the first column of a DataFrame as a Series. Original DataFrame col1 col2 col3 0 1 4 7 1 2 5 5 2 3 6 8 3 4 9 12 4 7 5 1 5 11 0 11 1st column as a Series: 0 1 1 2 2 3 3 4 4 7 5 11 Name: col1, dtype: int64

[61]:
```
x10 = pd.DataFrame([
                    [1,4,7],
                    [2,5,5],
                    [3,6,8],
                    [4,9,12],
                    [7,5,1],
                    [11,0,11],
                    ],
                    columns = ["col1","col2","col3"]
                   )
x10col1 = x10.loc[:,'col1']
x10col1
```

[61]:
```
0     1
1     2
2     3
3     4
4     7
5    11
Name: col1, dtype: int64
```

Write a Pandas program to convert a given Series to an array. Sample Output: Original Data Series: 0 100 1 200 2 python 3 300.12 4 400 dtype: object Series to an array [‘100’ ‘200’ ‘python’ ‘300.12’ ‘400’]

[63]:
```
x11 = pd.Series([100, 200, "python", 300.12, 400])
print(x11.to_list())
```

```
[100, 200, 'python', 300.12, 400]
```

Write a Pandas program to create a dataframe from a dictionary and display it. Sample data: {‘X’:[78,85,96,80,86], ‘Y’:[84,94,89,83,86],‘Z’:[86,97,96,72,83]}

[64]:
```
x12 = pd.DataFrame({'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':
↪[86,97,96,72,83]})
x12
```

[64]:

	X	Y	Z
0	78	84	86
1	85	94	97
2	96	89	96

```
3  80  83  72
4  86  86  83
```

Write a Pandas program to create and display a DataFrame from a specified dictionary data which has the index labels. Write a Pandas program to display a summary of the basic information about a specified DataFrame and its data.

[36]:
```
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily',␣
 ↪'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

x13 = pd.DataFrame(exam_data, index=labels)
x13.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10 entries, a to j
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   name      10 non-null     object
 1   score     8 non-null      float64
 2   attempts  10 non-null     int64
 3   qualify   10 non-null     object
dtypes: float64(1), int64(1), object(2)
memory usage: 400.0+ bytes
```

Write a Pandas program to get the first 3 rows of a given DataFrame.

[8]:
```
x13[0:3]
```

[8]:
```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
```

Write a Pandas program to select the 'name' and 'score' columns from the following DataFrame.

[26]:
```
x13.get(["name", "score"])
```

[26]:
```
        name  score
a  Anastasia   12.5
b       Dima    9.0
c  Katherine   16.5
d      James    NaN
e      Emily    9.0
f    Michael   20.0
```

```
g    Matthew   14.5
h      Laura    NaN
i      Kevin    8.0
j      Jonas   19.0
```

Write a Pandas program to select the specified columns and rows from a given data frame. Sample Python dictionary data and list labels:

[27]:
```
x14 = pd.DataFrame(x13.get(["name", "score", "qualify"]))
x14.iloc[[1,3,5,6]]
```

[27]:
```
      name  score qualify
b     Dima    9.0      no
d    James    NaN      no
f  Michael   20.0     yes
g  Matthew   14.5     yes
```

Write a Pandas program to select the rows where the number of attempts in the examination is greater than 2.

[16]:
```
x13[x13["attempts"] > 2]
```

[16]:
```
      name  score  attempts qualify
b     Dima    9.0         3      no
d    James    NaN         3      no
f  Michael   20.0         3     yes
```

Write a Pandas program to count the number of rows and columns of a DataFrame.

[17]:
```
nrows, ncols = x13.shape

print("Number of rows:", nrows)
print("Number of columns:", ncols)
```

```
Number of rows: 10
Number of columns: 4
```

Write a Pandas program to select the rows where the score is missing, i.e. is NaN.

[22]:
```
x13[x13["score"].isna()]
```

[22]:
```
    name  score  attempts qualify
d  James    NaN         3      no
h  Laura    NaN         1      no
```

Write a Pandas program to select the rows the score is between 15 and 20 (inclusive). Sample Python dictionary data and list labels:

[37]:
```
xfinal = pd.merge(x13.loc[x13["score"] >= 15, ["score"]], x13.loc[x13["score"]␣
↪<= 20, ["score"]], how="inner")
xfinal
```

[37]:

	score
0	16.5
1	20.0
2	19.0