# Pandas Basic

Assignment

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

import pandas as pd

data = [4, 8, 15, 16, 23, 42]

series = pd.Series(data)

print(series)

Output:

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.

To create a list variable with 10 elements and convert it to a pandas Series, we need to follow these steps:

1. Import the pandas library.
2. Create a list variable with 10 elements.
3. Convert the list to a pandas Series using the pandas.Series() function.
4. Print the resulting Series.

Here's the code to accomplish this:

import pandas as pd

# Create a list with 10 elements

my\_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Convert the list to a pandas Series

my\_series = pd.Series(my\_list)

# Print the resulting Series

print(my\_series)

When you run this code, it will print the pandas Series containing the elements from the list:

0 1

1 2

2 3

3 4

4 5

5 6

6 7

7 8

8 9

9 10

dtype: int64

# Q3. Create a Pandas Data Frame that contains the following data:

Name  Age Gender

Alice 25 Female

Bob 30 Male

Claire 27 Female

Then,print the DataFrame.

import pandas as pd

data = {

'Name': ['Alice', 'Bob', 'Claire'],

'Age': [25, 30, 27],

'Gender': ['Female', 'Male', 'Female']

}

df = pd.DataFrame(data)

print(df)

Output:

Name Age Gender

0 Alice 25 Female

1 Bob 30 Male

2 Claire 27 Female

# Q4. What is ‘DataFrame’ in pandas and how is it different from pandas.series? Explain with an example.

In pandas, a DataFrame is a two-dimensional data structure that can store and manipulate tabular data. It consists of rows and columns, similar to a spreadsheet or a SQL table. A DataFrame is one of the primary data structures provided by the pandas library, and it offers various operations and functionalities for data analysis and manipulation.

On the other hand, a pandas Series is a one-dimensional array-like object that can hold any data type. It is designed to represent a single column or row of data within a DataFrame. A Series has an index that labels each element in the array, allowing for easy and efficient access to the data.

To illustrate the difference between a DataFrame and a Series, let's consider an example related to student grades:

import pandas as pd

# Create a DataFrame

data = {

'Name': ['Alice', 'Bob', 'Charlie', 'David'],

'Math': [80, 75, 90, 85],

'Science': [85, 70, 95, 80]

}

df = pd.DataFrame(data)

# Print the DataFrame

print(df)

Output:

Name Math Science

0 Alice 80 85

1 Bob 75 70

2 Charlie 90 95

3 David 85 80

In this example, we create a DataFrame named df that contains information about student grades. Each column in the DataFrame represents a different attribute, such as the student's name, math grade, and science grade. The index along the left side of the DataFrame helps identify each row.

Now, let's say we want to access only the math grades of the students. We can do this using the column name and obtain a pandas Series:

# Access the 'Math' column as a Series

math\_series = df['Math']

# Print the Series

print(math\_series)

Output:

0 80

1 75

2 90

3 85

Name: Math, dtype: int64

# 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?

Pandas is a powerful library in Python for data manipulation and analysis. It provides numerous functions to manipulate data in a Pandas DataFrame. Here are some common functions:

1. head() and tail(): These functions allow you to view the first or last few rows of a DataFrame, respectively. They are often used to quickly inspect the data.

Example:

import pandas as pd

df = pd.read\_csv('data.csv')

print(df.head()) # View the first 5 rows of the DataFrame

print(df.tail(10)) # View the last 10 rows of the DataFrame

1. ‘info()’: This function provides a summary of the DataFrame, including the column names, data types, and non-null counts. It is useful for understanding the structure and content of the DataFrame.

Example:

import pandas as pd

df = pd.read\_csv('data.csv')

df.info() # Get information about the DataFrame

1. ‘describe()’: This function generates descriptive statistics of the DataFrame, such as count, mean, standard deviation, minimum, maximum, and quartiles. It gives a quick overview of the distribution of numerical columns.

Example:

import pandas as pd

df = pd.read\_csv('data.csv')

print(df.describe()) # Generate descriptive statistics of the DataFrame

1. ‘shape’: This attribute returns a tuple representing the dimensions of the DataFrame (number of rows, number of columns). It is useful to understand the size of the DataFrame.

import pandas as pd

df = pd.read\_csv('data.csv')

print(df.shape) # Get the dimensions of the DataFrame

1. drop()’: This function allows you to remove rows or columns from the DataFrame based on specified labels. It is handy when you want to eliminate irrelevant or redundant data.

Example:

import pandas as pd

df = pd.read\_csv('data.csv')

df = df.drop(['column1', 'column2'], axis=1) # Drop specified columns

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

In the pandas library, which is commonly used for data manipulation and analysis in Python, both the Series and DataFrame objects are mutable, while the Panel object is not mutable.

A Series is a one-dimensional labeled array that can hold any data type. You can modify its values by assigning new values to specific indexes or by applying various methods to manipulate the data.

A DataFrame is a two-dimensional labeled data structure with columns of potentially different data types. Similar to a Series, you can modify the values within a DataFrame by assigning new values to specific cells, adding or removing columns, or applying various data manipulation operations.

On the other hand, the Panel object, which represents a three-dimensional labeled array, is considered deprecated in recent versions of pandas. Panels were designed to handle data with three dimensions, but they have been mostly replaced by multi-indexing in DataFrame objects. Panels are not mutable in the sense that you cannot modify their values directly. Instead, you would need to create a new Panel with the desired changes.

It's important to note that this information is based on the state of pandas up until September 2021. Since pandas is an actively developed library, it's always a good idea to consult the pandas documentation or check for any updates if you are working with the latest version.

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

To create a DataFrame using multiple Series, you can use the pd.DataFrame() function from the pandas library in Python. A DataFrame is a two-dimensional labeled data structure with columns of potentially different types.

Here's an example to illustrate how you can create a DataFrame using multiple Series:

import pandas as pd

# Create a Series for 'Name'

names = pd.Series(['John', 'Alice', 'Bob'])

# Create a Series for 'Age'

ages = pd.Series([25, 30, 35])

# Create a Series for 'Country'

countries = pd.Series(['USA', 'Canada', 'UK'])

# Create a DataFrame using the Series

df = pd.DataFrame({'Name': names, 'Age': ages, 'Country': countries})

# Display the DataFrame

print(df)

Output:

Name Age Country

0 John 25 USA

1 Alice 30 Canada

2 Bob 35 UK