Guided Lab - 343.1.2

Creating a Pandas DataFrame

# 

# Lab Objective:

In this lab, we will demonstrate how to create a Panda Dataframe.

# Learning Objective:

By the end of this lab, you will be able to create and declare the Panda Dataframe.

**Instructions:**

You can start by importing pandas along with [NumPy](https://numpy.org/devdocs/), which you will use throughout the following examples:

import numpy as np

import pandas as pd

That’s it. Now you’re ready to create some DataFrames.

# Example 1: Creating a Pandas DataFrame With Dictionaries

We can create a Pandas DataFrame with a Python dictionary:

| d = {'x': [1, 2, 3], 'y': np.array([2, 4, 8]), 'z': 100}  pd.DataFrame(d) |
| --- |

**Result:**

x y z

0 1 2 100

1 2 4 100

2 3 8 100

* The keys of the dictionary are the DataFrame’s column labels, and the dictionary values are the data values in the corresponding DataFrame columns. The values can be contained in a [tuple](https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences), [list](https://docs.python.org/3/tutorial/datastructures.html#more-on-lists), one-dimensional [NumPy array](https://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.html), [Pandas Series object](https://pandas.pydata.org/pandas-docs/stable/reference/series.html), or one of several other data types. You can also provide a single value that will be copied along the entire column.
* It’s possible to control the order of the columns with the **columns** parameter and row labels with **index** as shown in the below example:

| pd.DataFrame(d, index=[100, 200, 300], columns=['z', 'y', 'x']) |
| --- |

**Result:**

z y x

100 100 2 1

200 100 4 2

300 100 8 3

As you can see, you have specified the row labels 100, 200, and 300. You also forced the order of columns: z, y, x.

# Example 2: Creating a Pandas DataFrame With Lists

Another way to create a Pandas DataFrame is to use a list of dictionaries:

| l = [{'x': 1, 'y': 2, 'z': 100},  {'x': 2, 'y': 4, 'z': 100},  {'x': 3, 'y': 8, 'z': 100}]  pd.DataFrame(l) |
| --- |

**Result:**

x y z

0 1 2 100

1 2 4 100

2 3 8 100

Again, the dictionary keys are the column labels, and the dictionary values are the data values in the DataFrame.

You can also use a **nested list**, or a **list of lists**, as the data values. If you do, then it is wise to explicitly specify the labels of columns, rows, or both when you create the DataFrame.

| l = [[1, 2, 100],  [2, 4, 100],  [3, 8, 100]]  pd.DataFrame(l, columns=['x', 'y', 'z']) |
| --- |

**Result:**

x y z

0 1 2 100

1 2 4 100

2 3 8 100

That is how you can use a **nested list** to create a Pandas DataFrame. You can also use a list of tuples in the same way. To do so, just replace the nested lists in the example above with tuples.

# Example 3: Creating a pandas DataFrame With NumPy Arrays

You can pass a two-dimensional NumPy array to the DataFrame constructor the same way you do with a list:

| arr = np.array([[1, 2, 100],[2, 4, 100],[3, 8, 100]])  df = pd.DataFrame(arr, columns=['x', 'y', 'z'])  df |
| --- |

**Result:**

x y z

0 1 2 100

1 2 4 100

2 3 8 100

Although this example looks almost the same as the nested list implementation above, it has one advantage. You can specify the optional parameter **copy**.

When a **copy** is set to **False** (its default setting), the data from the NumPy array is not copied. This means that the original data from the array is assigned to the Pandas DataFrame. If you modify the array, your DataFrame will change too:

| arr[0, 0] = 1000  df |
| --- |

**Result:**

x y z

0 1000 2 100

1 2 4 100

2 3 8 100

As you can see, when you change the first item of arr, you also modify df.

Note: Not copying data values can save you a significant amount of time and processing power when working with large datasets.

If this behavior is not what you want, you should specify **copy=True** in the **DataFrame** constructor. That way, **df** will be created with a copy of the values from **arr** instead of the actual values.

**For Canvas Official Team**

| **Instructions for Canvas Assignment Creation** |
| --- |
| **Assignment Name: GLAB - 343.1.2 - Creating a pandas DataFrame**  **Points:** **100**  **Assignment Group: Module 343 - Data Analytics with Python**  **Display Grade As: Complete/Incomplete**  **Do not count this assignment towards the final grade: Checked**  **Submission Types: File Upload**  **Everything else is the default.** |