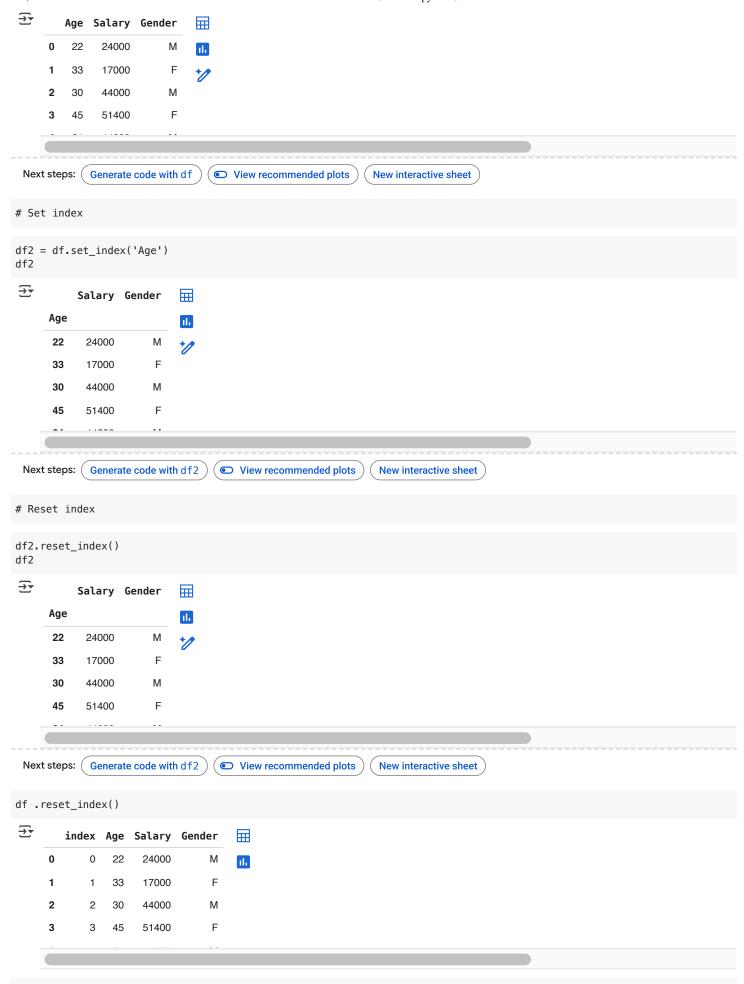
```
import numpy as np
import pandas as pd
# Pandas Series
# Pandas Series can be created using list
l = [10, 20, 30]
s = pd.Series(l)
print (s)
₹
    0
         10
          20
    2
         30
    dtype: int64
type(s)
₹
      pandas.core.series.Series
      def __init__(data=None, index=None, dtype: Dtype | None=None, name=None, copy: bool |
      None=None, fastpath: bool | lib.NoDefault=lib.no_default) -> None
      /usr/local/lib/python3.11/dist-packages/pandas/core/series.py
      One-dimensional ndarray with axis labels (including time series).
      Labels need not be unique but must be a hashable type. The object
      supports both integer- and label-based indexing and provides a host of
s.shape
→ (3,)
l = [10, 20, 30]
s = pd.Series(l,index=['a','b','c'])
print (s)
\overline{\Rightarrow}
          10
    b
         20
         30
    dtype: int64
# creating Series using dictionary
d = {'a':10,'b':20,'c':30}
s = pd.Series(d)
print (s)
          10
    b
         20
         30
    dtype: int64
# Pandas dataFrame
d = {'a':[1,2,3],'b':[4,5,6],'c':[7,8,9]}
df = pd.DataFrame(d)
print (df)
```

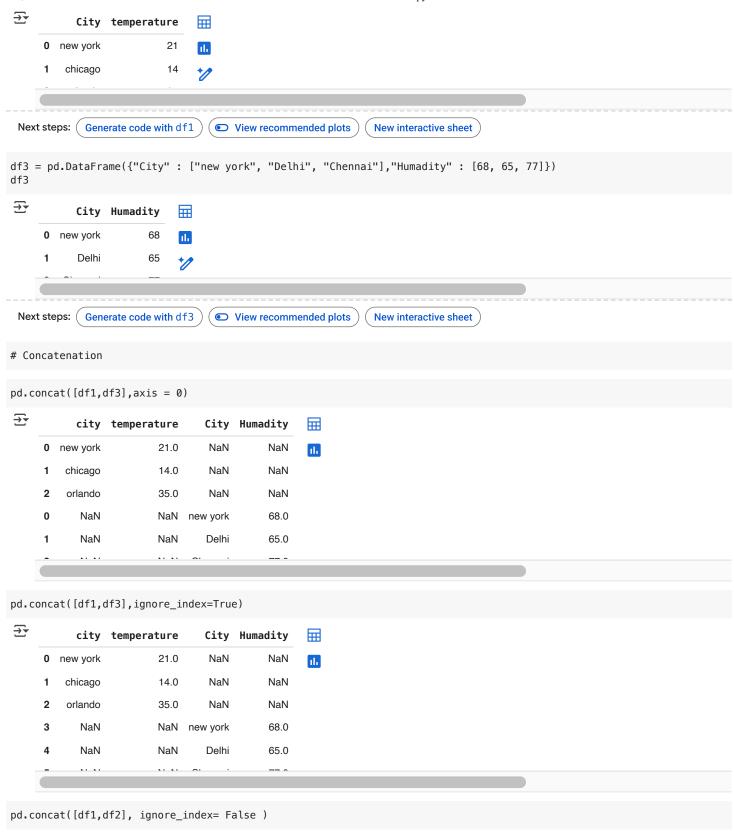
df

```
4/18/25, 3:57 PM
                                                                Untitled8.ipynb - Colab
          1
              4
        1 2 5 8
        2 3 6 9
    type(d)
    → dict
    type(df)
    ₹
          pandas.core.frame.DataFrame
          def __init__(data=None, index: Axes | None=None, columns: Axes | None=None, dtype: Dtype |
          None=None, copy: bool | None=None) -> None
          /usr/local/lib/python3.11/dist-packages/pandas/core/frame.py
          Two-dimensional, size-mutable, potentially heterogeneous tabular data.
          Data structure also contains labeled axes (rows and columns).
          Arithmetic operations align on both row and column labels. Can be
    df.shape
    → (3, 3)
    # Creating DataFrame using Nested List
    data = [[1,2,3],[4,5,6],[7,8,9]]
    df = pd.DataFrame(data,columns=['a','b','c'])
    df
    ₹
            a b c
         0 1 2 3
           4 5 6
     Next steps: (
                Generate code with df
                                     View recommended plots
                                                                New interactive sheet
    # Create DataFrame using 2D array
    arr_2d = np.array([[1,2,3],[4,5,6]])
    df2 = pd.DataFrame(arr_2d,columns=['a','b','c'])
    df2
    \overline{2}
            a b c
         0 1 2 3
         1 4 5 6
     Next steps: (
                Generate code with df2
                                      View recommended plots
                                                                 New interactive sheet
    # Pandas
```

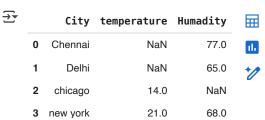
df = pd.DataFrame({'Age':[22,33,30,45,24],'Salary':[24000,17000, 44000, 51400,44600],'Gender': ["M", "F", "M", "F", "



```
# Drop a single row
df1 = df.drop(1)
df1
→
        Age Salary Gender
     0
         22
               24000
     2
         30
               44000
                           M
               51400
                           F
     3
         45
 Next steps: (
             Generate code with df1
                                    View recommended plots
                                                                 New interactive sheet
df3 = df.drop(columns = ["Age"])
df3
\overline{\pm}
        Salary Gender
                          \blacksquare
          24000
                           16
          17000
                      F
     1
          44000
                      F
     3
          51400
 Next steps: (
             Generate code with df3
                                    View recommended plots
                                                                 New interactive sheet
# sorting values in a dataframe
df.sort_values(by = 'Salary',ascending = True)
₹
        Age Salary Gender
                                1
         33
               17000
                           F
                                ıl.
         22
               24000
                           Μ
     2
               44000
         30
                           M
         24
               44600
                           Μ
df.sort_values(by = 'Salary',ascending = False)
₹
        Age Salary Gender
                                \blacksquare
     3
         45
               51400
                                th
               44600
     4
         24
                           Μ
         30
               44000
     2
                           M
               24000
# Combining Dataframes
df1 = pd.DataFrame({"City":["new york","chicago","orlando"],"temperature":[21,14,35]})
df1
```







$https://colab.research.google.com/drive/1eVMV3kDeMIaGg524PTq0gdvW0eJGjUtD\#scrollTo=SCw-jNHpbVz_\&uniqifier=1\&printMode=truewardscript{0.00000000000000000000000000000000000$