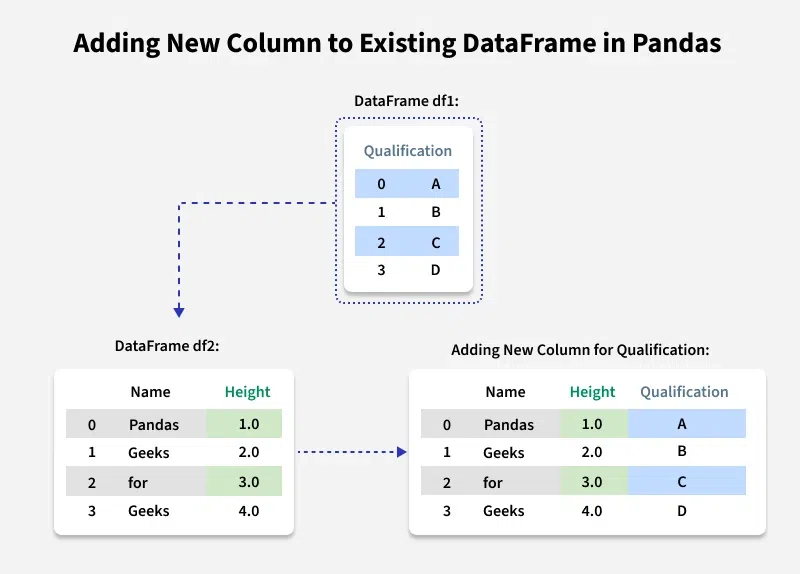
How to create new columns existing dataframe

Adding a new column to a DataFrame in Pandas is a simple and common operation when working with data in Python. You can quickly create new columns by directly assigning values to them. 

## By Declaring a New List as column

Let's first create a dataframe

import pandas as pd

# Define a dictionary containing Students data

Data2 = {'Name': ['Pandas', 'Geeks', 'for', 'Geeks'],

'Height': [1, 2, 3, 4],

'Qualification': ['A', 'B', 'C', 'D']}

# Convert the dictionary into DataFrame

df = pd.DataFrame(data2)

We can simply add new column to the DataFrame with the specified list values. Note that the length of your list should match the length of index column, else it will show an error.

# Declare a list that is to be converted into a column

address = ['NewYork', 'Chicago', 'Boston', 'Miami']

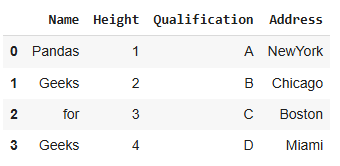
# Using 'Address' as the column name

# and equating it to the list

df['Address'] = address

display(df)

Output



Adding New Column to Existing DataFrame in Pandas

**Adding a new column to existing DataFrame in Pandas in Python**

**Steps**

1. Create DataFrame using a dictionary.

2. Create a list containing new column data. Make sure that the length of the list matches the length of the data which is already present in the data frame.

3. Add the list to the DataFrame like dictionary element.

print('create a new column to the existing column in dataframe ')

print('create a dictionary')

import pandas as pd

data3={

    'Name':['Pandas','Loc','Lio','Geeks'],

    'Height':[22,12,23,11],

    'Qualification':['Mba','Ca',"data A",'Doc'],

    'price':[200,300,400,500]

}

print('convert dictionary into dataframe')

print('before adding new column')

df=pd.DataFrame(data3)

display(df)

print('\n')

print('create a new column in existing dataframe')

df['price change']=df['price']\*40

df

output

Output

create a new column to the existing dataframe

create a dictionary

convert dictionary into dataframe

before adding new column

|  | **Name** | **Height** | **Qualification** | **price** |
| --- | --- | --- | --- | --- |
| 0 | Pandas | 22 | Mba | 200 |
| 1 | Loc | 12 | Ca | 300 |
| 2 | Lio | 23 | data A | 400 |
| 3 | Geeks | 11 | Doc | 500 |

create a new column in existing dataframe

|  | **Name** | **Height** | **Qualification** | **price** | **price change** |
| --- | --- | --- | --- | --- | --- |
| 0 | Pandas | 22 | Mba | 200 | 8000 |
| 1 | Loc | 12 | Ca | 300 | 12000 |
| 2 | Lio | 23 | data A | 400 | 16000 |
| 3 | Geeks | 11 | Doc | 500 | 20000 |

## DataFrame.insert()

There is a built-in method called insert() to add a new column.

### steps

1. Create DataFrame using dictionary.

2. Create a list containing new column data. Make sure that the length of the list matches the length of the data which is already present in the data frame.

3. Insert the data into the DataFrame using DataFrame.insert(index, column\_name, data) method.

#import pandas

import pandas as pd

# craete dataframe

data4={

    'Name': ['Hafeez', 'Aslan', 'Kareem'],

   'Age': [19, 18, 15],

   'Profession': ['Pythoneer', 'Programmer', 'Student']

}

dataframe=pd.DataFrame(data4)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*before add new column\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print(dataframe)

print('\n')

print('craete a new list')

address=['jalandhar','amritsar','pune']

# now use address column in existing dataframe gives the index number where we want to insert

dataframe.insert(2,'person address',address)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*after add new column\*\*\*\*\*\*\*\*\*\*\*\*')

print(dataframe)

Output

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*before add new column\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Name Age Profession

0 Hafeez 19 Pythoneer

1 Aslan 18 Programmer

2 Kareem 15 Student

craete a new list

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*after add new column\*\*\*\*\*\*\*\*\*\*\*\*

Name Age person address Profession

0 Hafeez 19 jalandhar Pythoneer

1 Aslan 18 amritsar Programmer

2 Kareem 15 pune Student

## DataFrame.assign()

This method takes one argument i.e., a list of data and adds it to the data frame as a column at the end.

### steps

1. Create DataFrame using a dictionary.

2. Create a list containing new column data. Make sure that the length of the list matches the length of the data which is already present in the data frame.

3. Insert the data into the DataFrame using DataFrame.assign(column\_name = data) method. It returns a new data frame. So, we have to store it.

4. Print the new data frame.

#import pandas

import pandas as pd

# craete dataframe

data4={

    'Name': ['Hafeez', 'Aslan', 'Kareem','sunita'],

   'Age': [19, 18, 15,32],

   'Profession': ['Pythoneer', 'Programmer', 'Student','wife']

}

dataframe=pd.DataFrame(data4)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*before add new column\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print(dataframe)

print('\n')

print('craete a new list')

address=['jalandhar','amritsar','pune','pape']

# now use assigning address column in existing dataframe

new\_dataframe=dataframe.assign(PeopleAddress=address)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*after add new column\*\*\*\*\*\*\*\*\*\*\*\*')

print(new\_dataframe)

output

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*before add new column\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Name Age Profession

0 Hafeez 19 Pythoneer

1 Aslan 18 Programmer

2 Kareem 15 Student

3 sunita 32 wife

craete a new list

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*after add new column\*\*\*\*\*\*\*\*\*\*\*\*

Name Age Profession peopleAddress

0 Hafeez 19 Pythoneer jalandhar

1 Aslan 18 Programmer amritsar

2 Kareem 15 Student pune

3 sunita 32 wife pape

## Add Multiple Columns in Dataframe at once (By using Dictionary Method)

We can use a [Python dictionary](https://www.geeksforgeeks.org/python/python-dictionary/) to add a new column in pandas DataFrame. This method is recommended when you want to add multiple columns at once or if you have columns in a dictionary format.

# import pandas

import pandas as pd

# Define dictionary defined as studen data

student={

    'Name':['Pandas', 'Geeks','For','david'],

    'cals':['2nd','3rd','4th','5th'],

    'rollno':[100,200,300,202]

}

# Convert the dictinory in the data frame

df=pd.DataFrame(student)

# now create a ned dictionary name as address with the key and address value

address={

    'Pandas':'NewYork','Geeks':'Delhi','For':'Denmark','david':'Chicago'

}

# Now add address column by mapping the 'Name' column

# add address dictinot=ry in the existing dataframe

df['Address']=df['Name'].map(address)

print(df)

output

Name cals rollno Address

0 Pandas 2nd 100 NewYork

1 Geeks 3rd 200 Delhi

2 For 4th 300 Denmark

3 david 5th 202 Chicago

Add Multiple Columns in Dataframe at once (By using Dictionary Method)

We can use a Python dictionary to add a new column in pandas DataFrame. This method is recommended when you want to add multiple columns at once or if you have columns in a dictionary format.

#add two new column in the existing dataframe

new\_column={

    'Age':[22,33,23,44],

    'Qualification':['MBA','CA','Doctor','Nurse']

}

# now unpack or add these two column in the existing dataframe

df=df.assign(\*\*new\_column)

print(df)

output

Name cals rollno Address Age Qualification

0 Pandas 2nd 100 NewYork 22 MBA

1 Geeks 3rd 200 Delhi 33 CA

2 For 4th 300 Denmark 23 Doctor

3 david 5th 202 Chicago 44 Nurse

By using Dataframe.loc()

Using .loc[], you can add a new column directly or modify values based on conditions, or when adding new columns based on specific row selections.

# import pandas

import pandas as pd

#create dictinory as company data

company={

    'Name':['arya','dario','cola','fanta'],

    'no\_of\_employee':[22,22,44,100],

    'Average\_salary':[3000,2000,3200,2000]

}

# convert dictinory in data frame

df=pd.DataFrame(company)

print('-----------before apply condition data shows-------------')

print(df)

#using .loc() function add new column category

df.loc()[df['Average\_salary']>2500,'Category']='Good'

df.loc()[df['Average\_salary']<2500,'Category']='less average'

print('----------after column addition-----------')

print(df)

output

-----------before apply condition data shows-------------

Name no\_of\_employee Average\_salary

0 arya 22 3000

1 dario 22 2000

2 cola 44 3200

3 fanta 100 2000

----------after column addition-----------

Name no\_of\_employee Average\_salary Category

0 arya 22 3000 Good

1 dario 22 2000 less average

2 cola 44 3200 Good

3 fanta 100 2000 less average

Python Pandas Dataframe Renaming

A Pandas Dataframe is a 2-dimensional data structure that displays data in tables with rows and columns.

Pandas Dataframe by using:

The rename() function.

A List.

The set\_axis() function.

How to Rename a Column in Pandas Using the rename() Function

Renaming a Pandas Dataframe using the rename() function.

we created a Python dictionary which we used to store the firstname and lastname of students.

We then converted the dictionary to a Dataframe by passing it as a parameter to the Pandas Dataframe object: pd.DataFrame(students).

We can do that using the rename() function.

Syntax

df.rename(columns={"OLD\_COLUMN\_VALUE": "NEW\_COLUMN\_VALUE"})

change the column names (firstname and lastname) in the table from lowercase to uppercase (FIRSTNAME and LASTNAME).

we specified that the columns firstname and lastname should be renamed to FIRSTNAME and LASTNAME, respectively:

df.rename(columns={"firstname": "FIRSTNAME", "lastname": "LASTNAME"}, inplace=True)

we added the inplace=True parameter. This helps in persisting the new changes in the Dataframe. Inplace true means the changes permanently changes in the dataframe.

# import pandas

import pandas as pd

# create dictinory

student1={

    'stu\_first\_name':['sonia','anita','rahul'],

    'stu\_last\_name':['sharma','goel','verma']

    }

# convert dictinory in dataframe

df=pd.DataFrame(student1)

#print dataframe

print('before ramname data display')

print(df)

#print('rename column name in the dataframe in display')

print('\n')

df.rename(columns={'stu\_first\_name':'First\_name', 'stu\_last\_name':'Last\_name'},inplace=True)

print('after use rename option')

print(df)

out put

before ramname data display

stu\_first\_name stu\_last\_name

0 sonia sharma

1 anita goel

2 rahul verma

after use rename option

First\_name Last\_name

0 sonia sharma

1 anita goel

2 rahul verma

Index(['First\_name', 'Last\_name'], dtype='object')

Rename a Column in Pandas Using a List

We can access the column names of a Dataframe using df.columns.

We can print out the column names with the code

print(df.columns)

print('rename the columns with the df.column command for crete a list then rename the columns names')

# import pandas

import pandas as pd

# creete dictinory

student2={

    'Name':['Amit','Sumit','Anu'],

    'class':['4th','5th','2nd']

}

# convert dictinory in data frame

df=pd.DataFrame(student2)

print('display dataframe')

print(df)

#create a list and then rename the existing columns in the dataframe

print('\n')

print('after rename the columns name')

df.columns=['student\_name','student\_class']

print(df)

output

rename the columns with the df.column command for crete a list then rename the columns names

display dataframe

Name class

0 Amit 4th

1 Sumit 5th

2 Anu 2nd

after rename the columns name

student\_name student\_class

0 Amit 4th

1 Sumit 5th

2 Anu 2nd

Rename a Column in Pandas Using the set\_axis() Function

The syntax for renaming a column with the set\_axis() function looks like this:

df.set\_axis([NEW\_COLUMN\_NAME,...], axis="columns")

the inplace=True parameter might raise a warning because it's deprecated for the set\_axis() function and will be replaced in the future.

print('rename the columns with use of set\_axis() function')

# import pandas

import pandas as pd

# creete dictinory

student3={

    'Name':['Amit','Sumit','Anu'],

    'class':['4th','5th','2nd']

}

# convert dictinory in data frame

df=pd.DataFrame(student3)

print('display dataframe')

print(df)

# rename the existing columns in the dataframe with use of set\_axis() function

print('\n')

print('after rename the columns name')

df.set\_axis(['student\_name','student\_class'], axis='columns')

print(df)

ouput

rename the columns with use of set\_axis() function

display dataframe

Name class

0 Amit 4th

1 Sumit 5th

2 Anu 2nd

after rename the columns name

Name class

0 Amit 4th

1 Sumit 5th

2 Anu 2nd

REMEMBER points

Create a new column by assigning the output to the DataFrame with a new column name in between the [].

Operations are element-wise, no need to loop over rows.

Use rename with a dictionary or function to rename row labels or column names.

Rename could be done by list withuse of df.column() function

Rename can be possible by set\_axis() fumctions