**PANDAS**

df**.**shape

df.columns

pd**.**set\_option('display.max\_columns', 85)

pd**.**set\_option('display.max\_rows', 85)

# displaying the DataFrame

display(data)

**# Selecting a column**

df['Hobbyist']

**# Selecting multiple columns**

df[['last', 'email']]

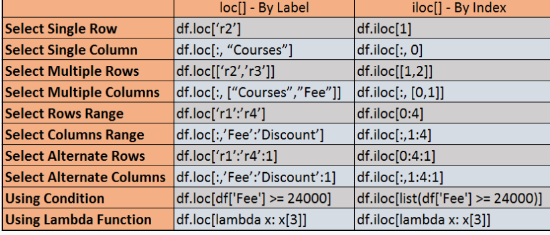
**# Selecting row and column using loc and iloc**

# The main difference between pandas loc[] vs iloc[] is loc gets DataFrame rows & columns by labels/names and iloc[] gets by integer Index/position. For loc[], if the label is not present it gives a key error. For iloc[], if the position is not present it gives an index error.

# Selecting columns from DataFrame results in a new DataFrame containing only specified selected columns from the original DataFrame.

df**.**loc[0:2, 'Hobbyist':'Employment']

df**.**iloc[[0, 1], 2]



# Select Multiple Rows by Label

print(df.loc[['r2','r3']])

# Select Multiple Rows by Index

print(df.iloc[[1,2]])

**# Converting a dictionary to a dataframe**

import pandas as pd

technologies = {

'Courses':["Spark","PySpark","Hadoop","Python","pandas"],

'Fee' :[20000,25000,26000,22000,24000],

'Duration':['30day','40days','35days','40days','60days'],

'Discount':[1000,2300,1200,2500,2000]

}

index\_labels=['r1','r2','r3','r4','r5']

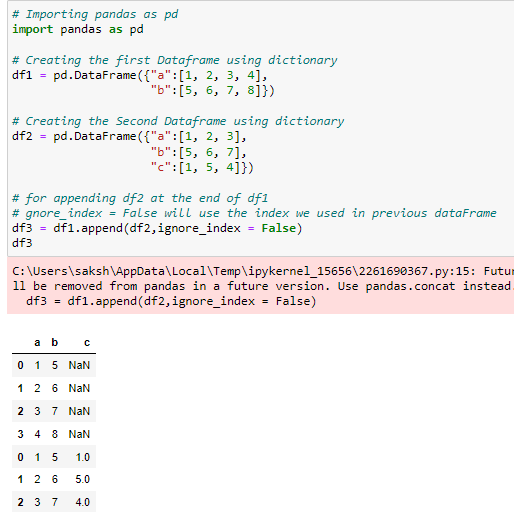
df = pd.DataFrame(technologies,index=index\_labels)

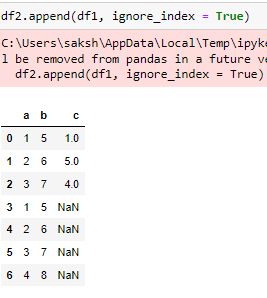
**# Adding a row in Pandas using loc(iloc can’t be used as can’t be extended)**

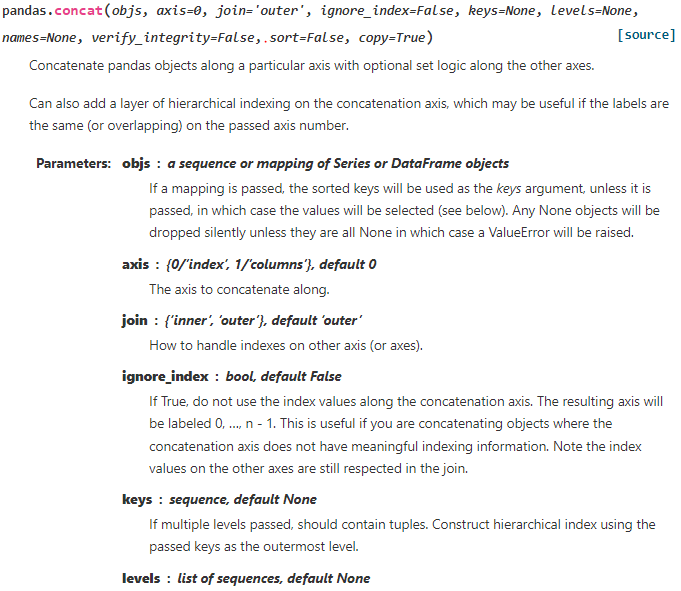
df.loc[len(df.index)] **=** ['Amy', 89, 93]

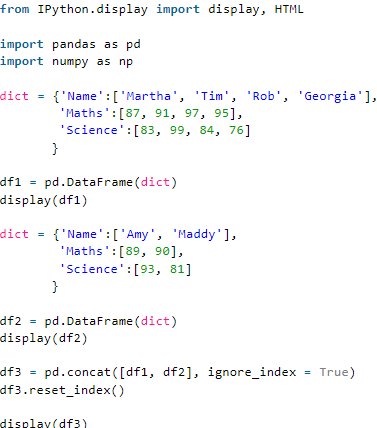
**# Adding a row in Pandas using datframe.append()**

df **=** df.append(df2, ignore\_index **=** True)



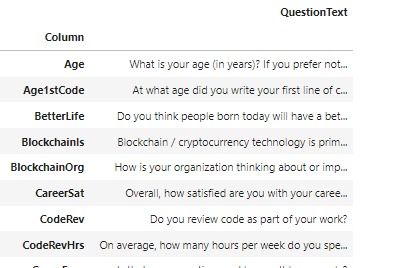


**# Adding the row using df.concat()(It’s recommended than using df.append())**



**# Sorting data frame via using index**

schema\_df**.**sort\_index(inplace**=True**)



**# Sorting data frame via using a particular column**

my\_data.sort\_values(by=["MarksB"],ascending=True)

**# Setting email as measure to choose which row**

df**.**set\_index('email', inplace  
**=True)**

**# Getting values of the column which will be used as a measure to choose which row**

df**.**index

**# Removing the index we choose earlier as a measure of selecting row**

df**.**reset\_index(inplace**=True**)

# Dropping a column from Datafrane

DataframeName.drop(columns=["ColumnName1",”ColumnName2”])

# Dropping a row

df.drop(df.index[2])

# DataFrame to csv

DataframeName.to\_csv(“FileName.csv”)

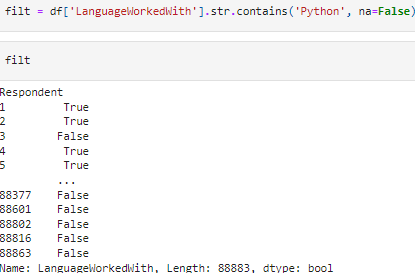
# Mapping in Values

df['Hobbyist'] **=** df['Hobbyist']**.**map({'Yes': **True**, 'No': **False**})

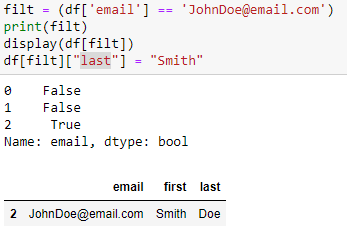
# Changing a rows value

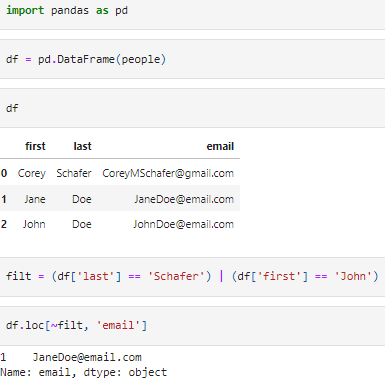
df**.**loc[2] **=** ['John', 'Smith', 'JohnSmith@email.com']

# filtering based on containing a particular string



# filtering based on logical operations





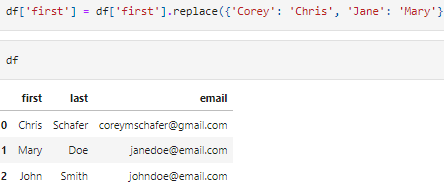
# Renaming columns

df**.**rename(columns**=**{'first\_name': 'first', 'last\_name': 'last'}, inplace**=True**)

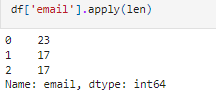
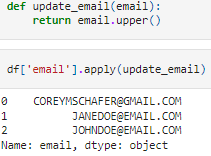
# Changing multiple column values of a particular row

df**.**loc[2, ['last', 'email']] **=** ['Doe', 'JohnDoe@email.com']

# Replacing elements in a particular column

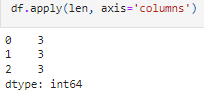


# Appling a particular function on a particular column

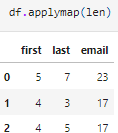
 

# Getting number of elements in that very column

# Appling the function alongside the column



# Appling a particular function to all the elements of the Dataframe



# Changing columns names

