```
#Create DataFrame from dict using constructor
import pandas as pd
# Create dict object
student_dict = {"name": ["Joe", "Nat", "Harry"], "age": [20, 21, 19], "marks": [85.10, 77.80, 91.54]}
print(student_dict)
# Create DataFrame from dict
student_df = pd.DataFrame(student_dict)
print(student_df, "\n")
student_df
```

```
{'name': ['Joe', 'Nat', 'Harry'], 'age': [20, 21, 19], 'marks': [85.1, 77.8, 9
    name  age  marks
0    Joe   20  85.10
1    Nat   21  77.80
2  Harry   19  91.54
```

	name	age	marks
0	Joe	20	85.10
1	Nat	21	77.80
2	Harry	19	91.54

```
"""                        # Dictinary

# student_df = pd.DataFrame(student_dict)                                      # Create DataFrame from dict
# student_df = pd.DataFrame(student_dict, columns=["name", "marks"])           # from dict with required columns only
# student_df = pd.DataFrame(student_dict, index=["stud1", "stud2", "stud3"])   # from dict with user-defined indexes
# student_df = pd.DataFrame(student_dict, dtype="float64")                     # from dict by changing the column data type
# student_df = pd.DataFrame(student_dict, index=['stud1'])                     # from dict with a single value
# student_df = pd.DataFrame(student_dict.items(), columns=["name", "marks"])   # from dict with key and value name as a colu

                        # List


# fruits_df = pd.DataFrame(fruits_list)                                        # Create DataFrame from list using constructo
# fruits_df = pd.DataFrame(fruits_list, columns=['Fruits'])                    # from list with a customized column name
# fruits_df = pd.DataFrame(fruits_list, index=['Fruit1', 'Fruit2', 'Fruit3'])  # from list with a customized index
# price_df = pd.DataFrame(price_list, dtype='float64')                         # from list by changing data type
# fruits_df = pd.DataFrame(list(zip(fruits_list, price_list )), columns = ['Name', 'Price']) # from multiple lists
   Alternative
    #fruits_dict = {'Name': fruits_list,'Price': price_list}
    # fruits_df = pd.DataFrame(fruits_dict)


                  # head()_and_tail()


# topRows = student_df.head(3)                            # display first 3 rows
# topRows = student_df.head(-2)                           # display rows except bottom 2 rows
# bottomRows = student_df.tail()                          # display the bottom 5 rows
# value = student_df.at[2,"Age"]                          # Select value using row and column labels using DataFrame.at
# student_df.at[2,"Age"] = 50                             # change the value
# value = student_df.iat[1,2]                             # Select value using row and column position using DataFrame.iat

# student_df.iat[1,2]=90. print(student_df.iat[1,2]) # Set specific value in pandas DataFrame


                  # drop_columns


# student_df = student_df.drop(columns='age')                           # drop single column

# print(student_df.columns.values)
  student_df = student_df.drop(columns=['age', 'marks'])                # drop 2 columns at a time
  print(student_df.columns.values)

# student_df = student_df.drop(['age', 'marks'], axis='columns')        # Using drop with axis='columns' or axis=1
# print(student_df.columns.values)
  student_df.drop(columns=['age', 'marks'], inplace=True)               # Drop column in place
  print(student_df.columns.values)


# student_df = student_df.drop(columns='salary', errors='ignore')       # No change in the student_df , # supress error
# student_df = student_df.drop(columns='salary')                        # KeyError: "['salary'] not found in axis", # raise err

# print(student_df.columns.values)
  student_df = student_df.drop(columns=student_df.iloc[:, 1:3])         # Drop range of columns using iloc
  print(student_df.columns.values)
```

```
# print("Before dropping: \n", student_df.columns.values)
  student_df = student_df.drop(columns=student_df.iloc[:, range(2)])   # Dropping the first two columns from a DataFrame.
  print("\nAfter dropping: \n", student_df.columns.values)

# print("Before dropping column: \n", student_df.columns.values)
 student_df = student_df.drop(columns=student_df.loc[:])               # drop column 1 and 2
 print("\nAfter dropping column: \n", student_df.columns.values)



# student_df.pop('age')                                               # drop column
# del student_df['age']                                               # drop column

# student_df = student_df.drop_duplicates()                           # drop duplicate rows
# student_df = student_df.drop_duplicates(keep='last')                # Drop duplicates but keep last
# student_df = student_df.drop_duplicates(keep=False)                 # Drop all duplicates

# student_df = pd.DataFrame(student_dict, index=['a', 'b', 'c', 'd']) # Drop duplicates and reset the index
  student_df = student_df.drop_duplicates(keep=False, ignore_index=True)


                        # Rename


# student_df = student_df.rename(columns={'marks': "percentage"})      # Rename

# print(student_df.columns.values)
  student_df.rename(lambda x: x.strip(), axis='columns', inplace=True) # remove leading & trailing space from column names
  print(student_df.columns.values)

# print(student_df.columns.values)                                      # before rename
  student_df.columns = ['stud_name', 'stud_age', 'stud_marks']          # rename column with list
  print(student_df.columns.values)                                      # after rename

# student_df.set_axis(['new_name', 'new_age', 'new_marks'], axis='columns', inplace=True) # reassign column headers


                          # Python_Pandas_DataFrame_to_Python_dictionary


# studentDf = pd.read_csv("student_data.csv")                           # create dataframe from csv
# studentDict = studentDf.to_dict('list')                               # create dict from dataframe

# studentDict = studentDf.to_dict('series')                             # create dict from dataframe
# studentDict = studentDf.to_dict('index')                              # DataFrame to dict by row index


                        # Set index


# index = pd.Index(['s1', 's2', 's3'])                      # Set index using a list

# student_df = student_df.set_index(['Name', 'Marks'])      # set multi-index
# index = pd.Index(['s1', 's2', 's3'])
  student_df = student_df.set_index([index, 'Name'])        # multi-index using a list and column

# student_df = student_df.set_index('Name', drop=False)     # Set index but keep column

# student_df.set_index('Name', inplace=True)                # set index in place

# cols = list(student_df.columns[[0,2]])                    # set index
  student_df = student_df.set_index(cols)

# student_df = student_df.reset_index(drop=True)            # reset index without new column

# student_df.reset_index(inplace=True)                      # reset index in place

# student_df = student_df.reset_index().rename(columns={'index': 'ID'})     # Reset index and change column name

"""
```