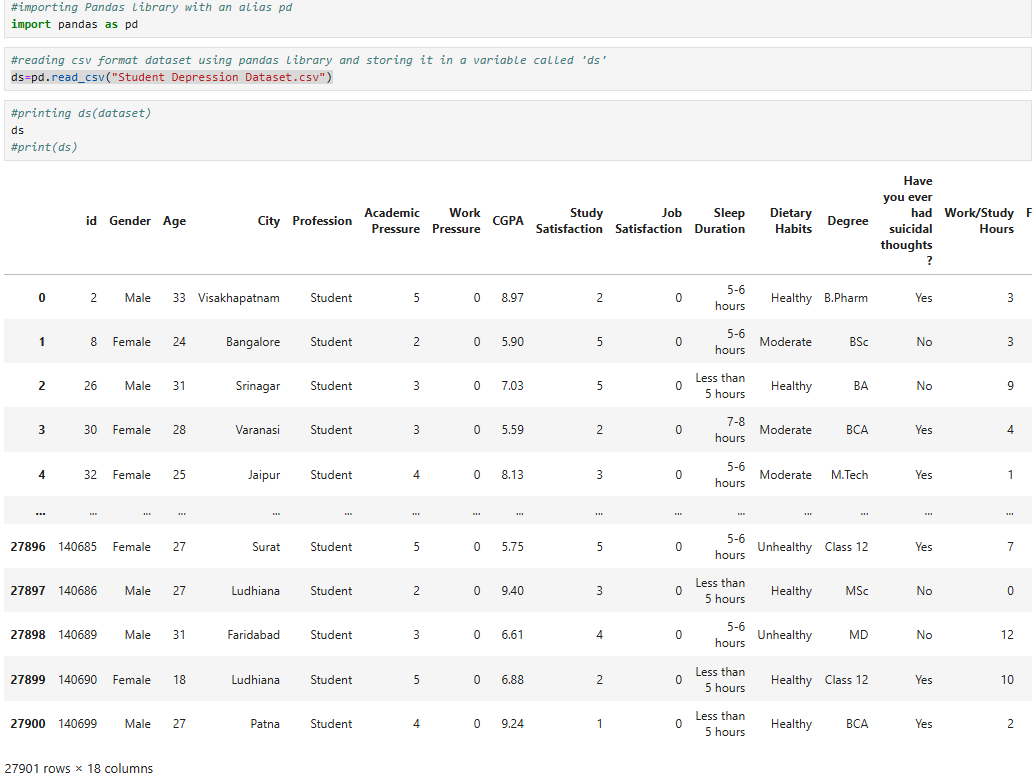
Data Analytics

**Q.1.**

**Ans:** import pandas as pd

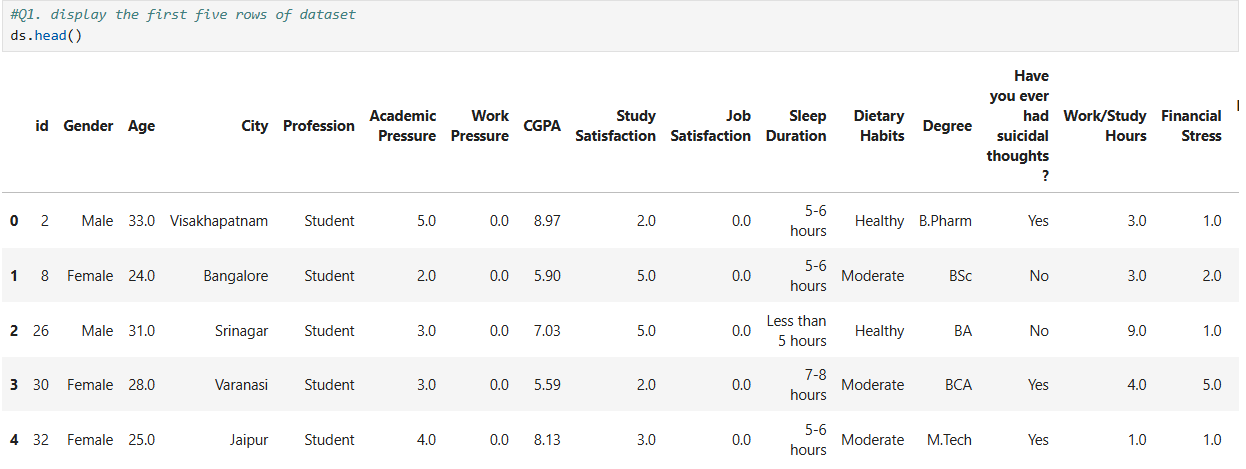
ds=pd.read\_csv("Student Depression Dataset.csv")

print(ds)



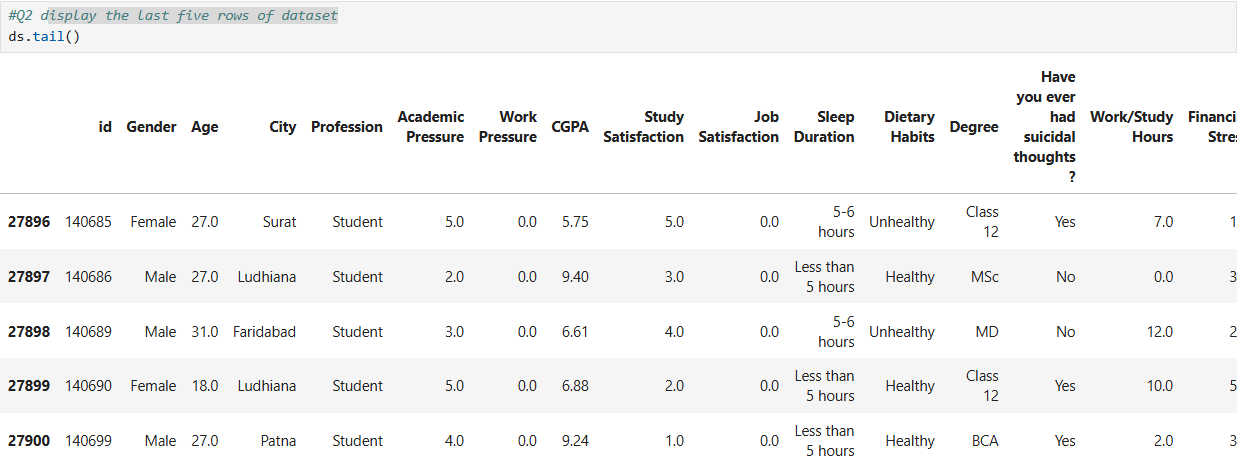
**Q.2. Display the first five rows of dataset**

**Ans:** ds.head()



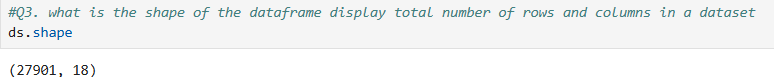
**Q.3. Display the last five rows of dataset**

**Ans:** ds.tail()



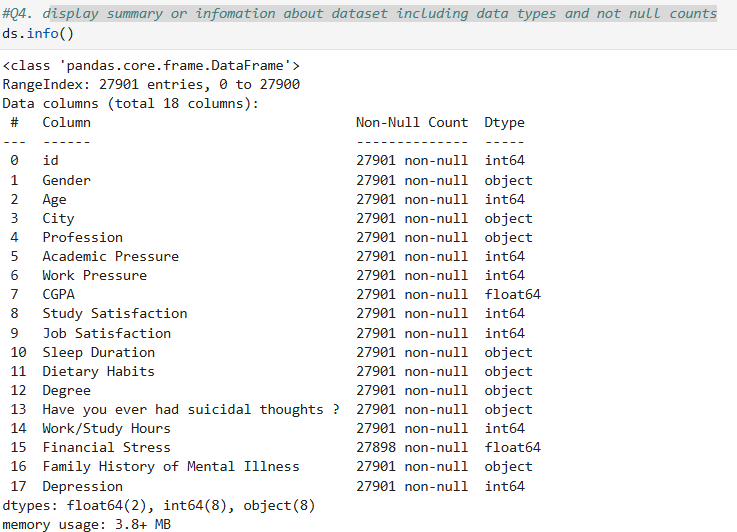
**Q.4. What is the shape of the dataframe? Display total number of rows and columns in a dataset**

**Ans:** ds.shape



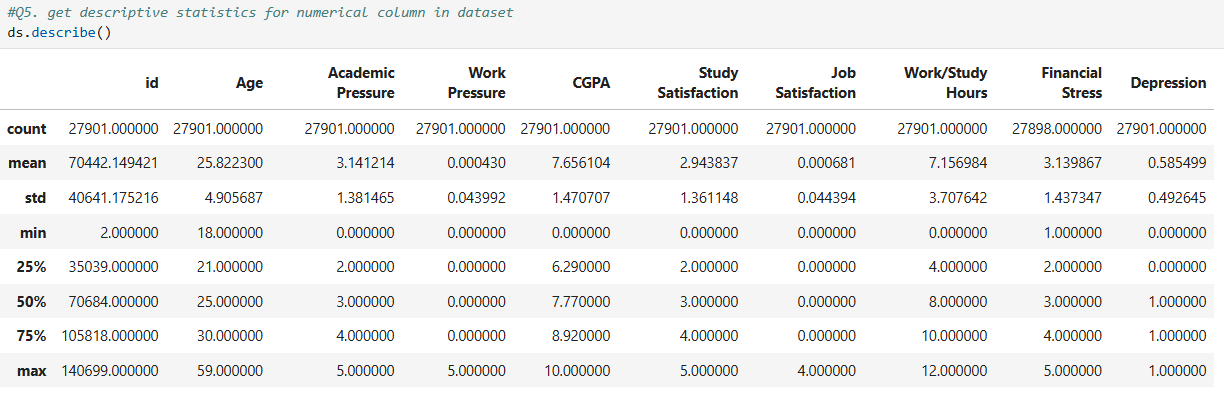
**Q.5. Display summary or infomation about dataset including data types and not null counts**

**Ans:** ds.info()



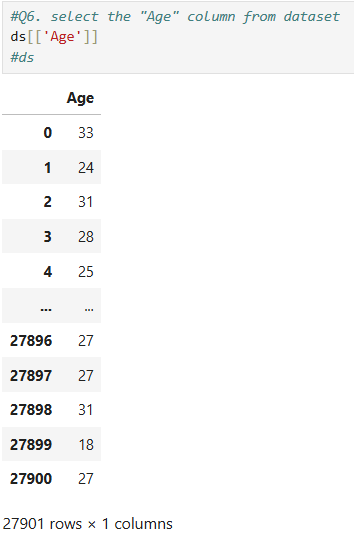
**Q.6. Get descriptive statistics for numerical column in dataset**

**Ans:** ds.describe()



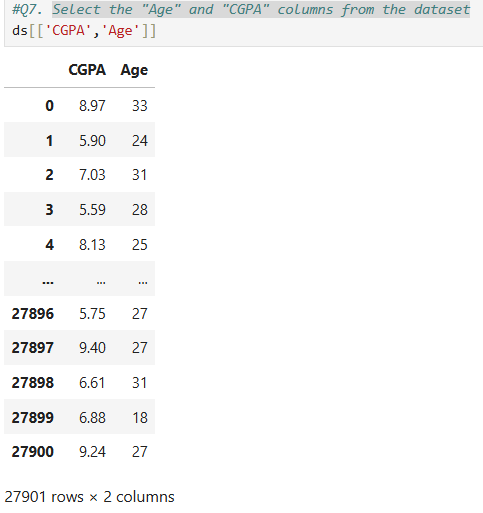
**Q.7. Select the "Age" column from dataset**

**Ans:** ds[['Age']]



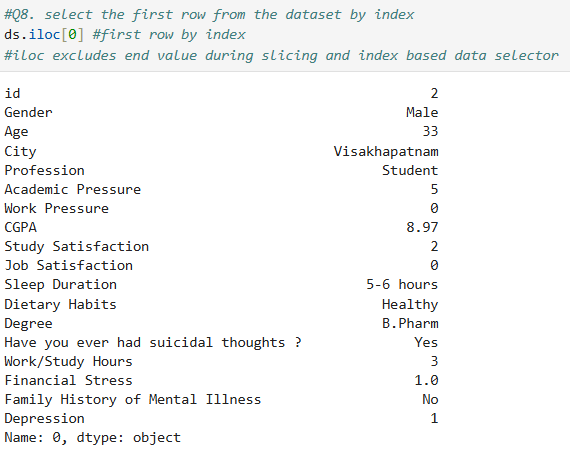
**Q.8. Select the "Age" and "CGPA" columns from the dataset**

**Ans:** [['CGPA','Age']]



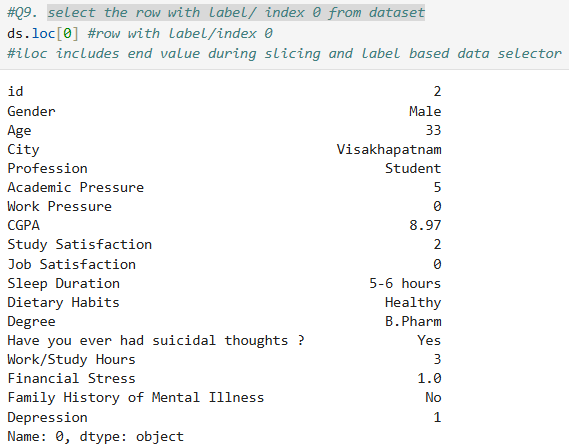
**Q.9. Select the first row from the dataset by index**

**Ans:** ds.iloc[0]



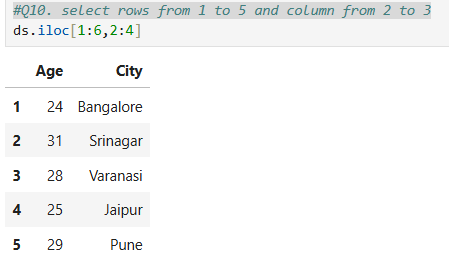
**Q.10. Select the row with label/ index 0 from dataset**

**Ans:** ds.loc[0]



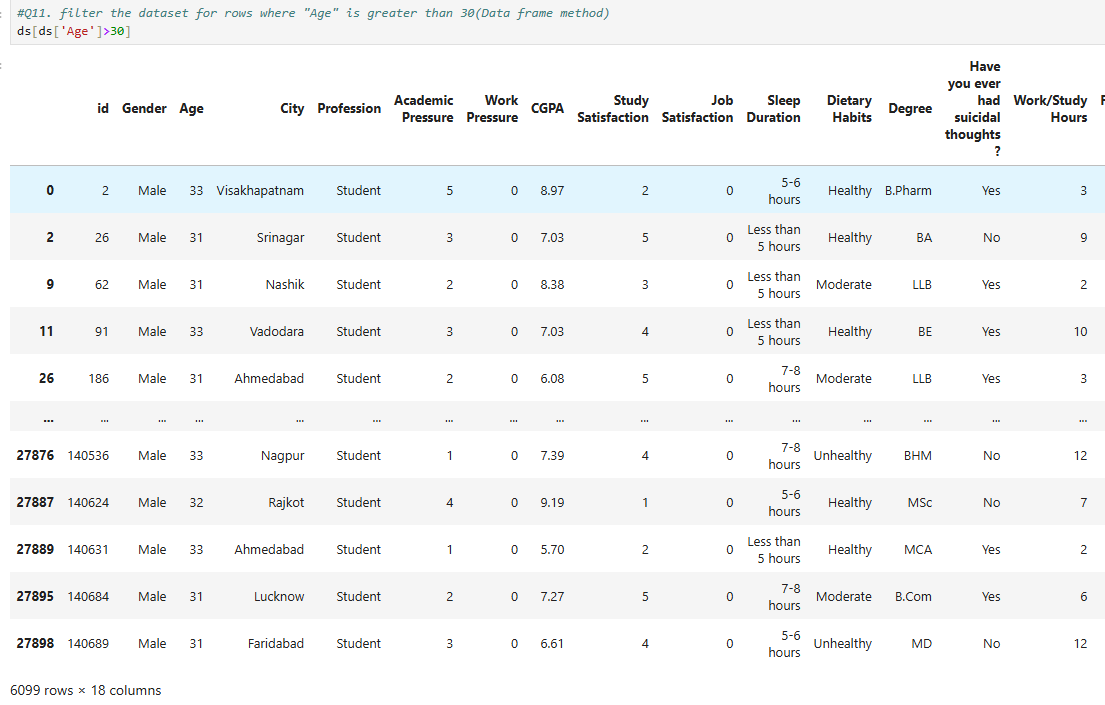
**Q.11. Select rows from 1 to 5 and column from 2 to 3**

**Ans:** ds.iloc[1:6,2:4]



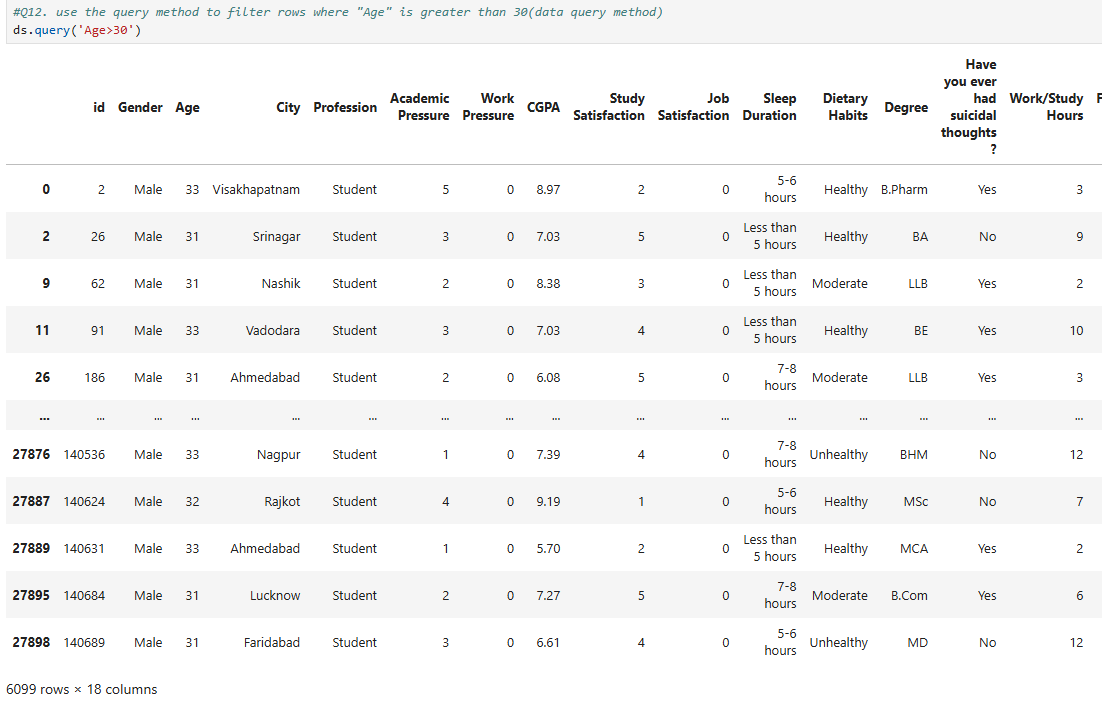
**Q.12. Filter the dataset for rows where "Age" is greater than 30(Data frame method)**

**Ans:** ds[ds['Age']>30]



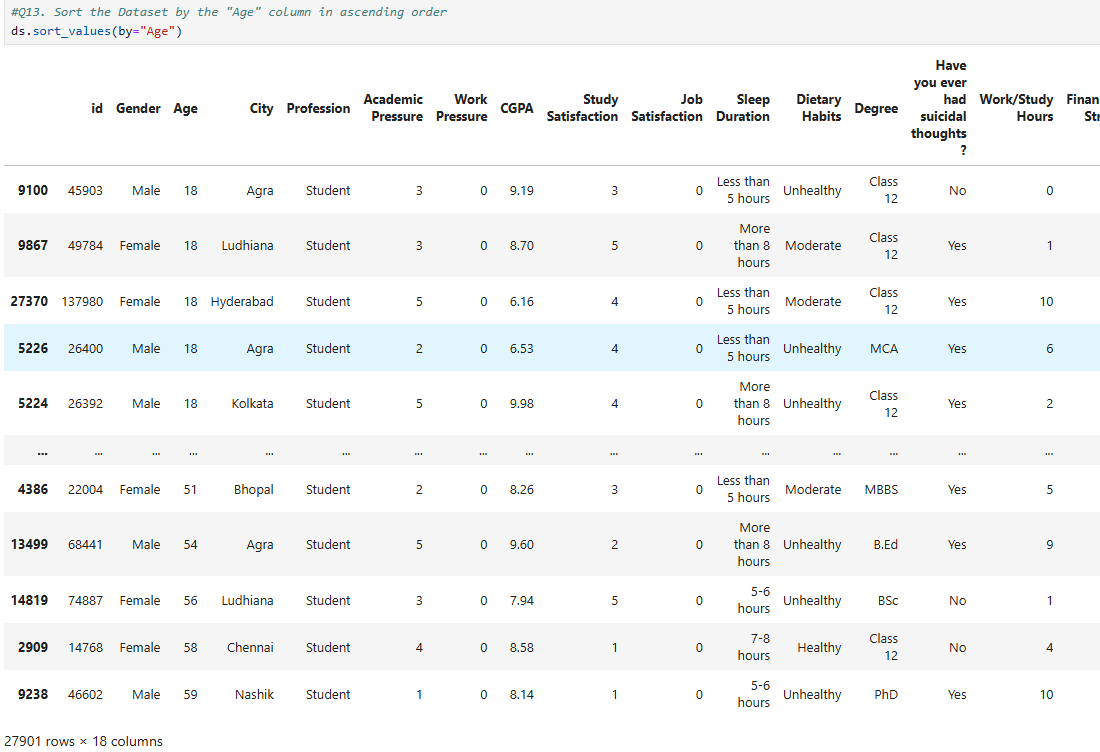
**Q.13. Use the query method to filter rows where "Age" is greater than 30(data query method)**

**Ans:** d ds.query('Age>30')



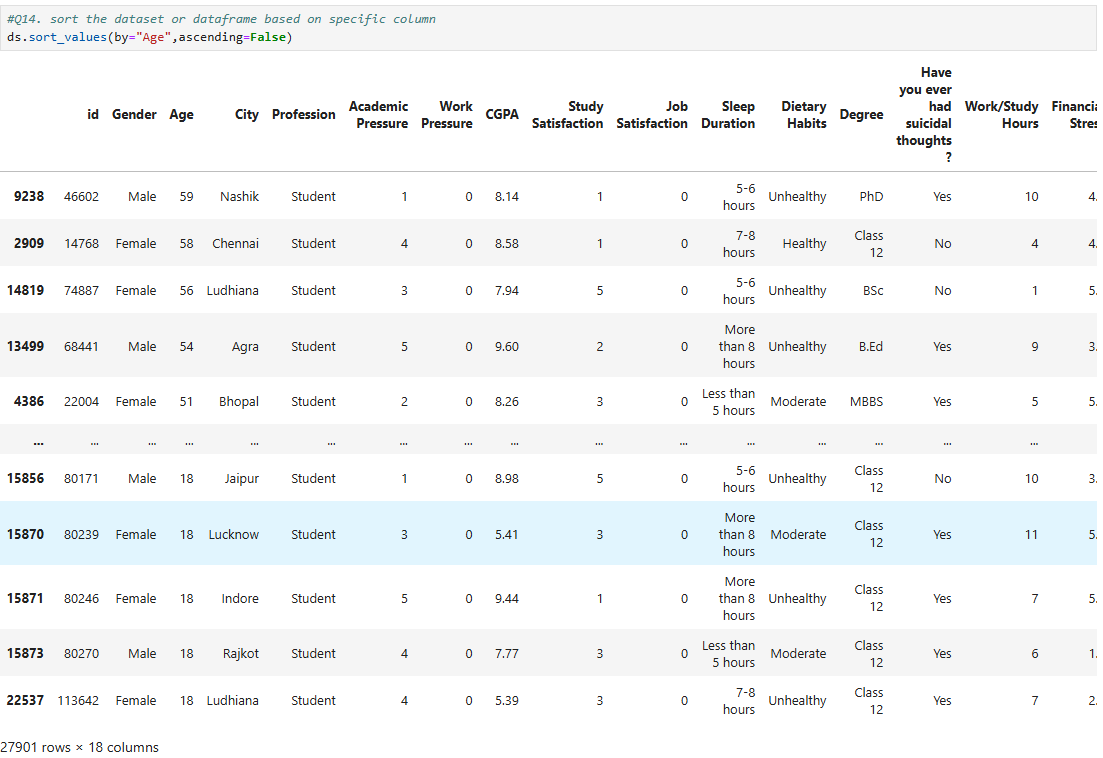
**Q.14. Sort the Dataset by the "Age" column in ascending order**

**Ans:** ds.sort\_values(by="Age")



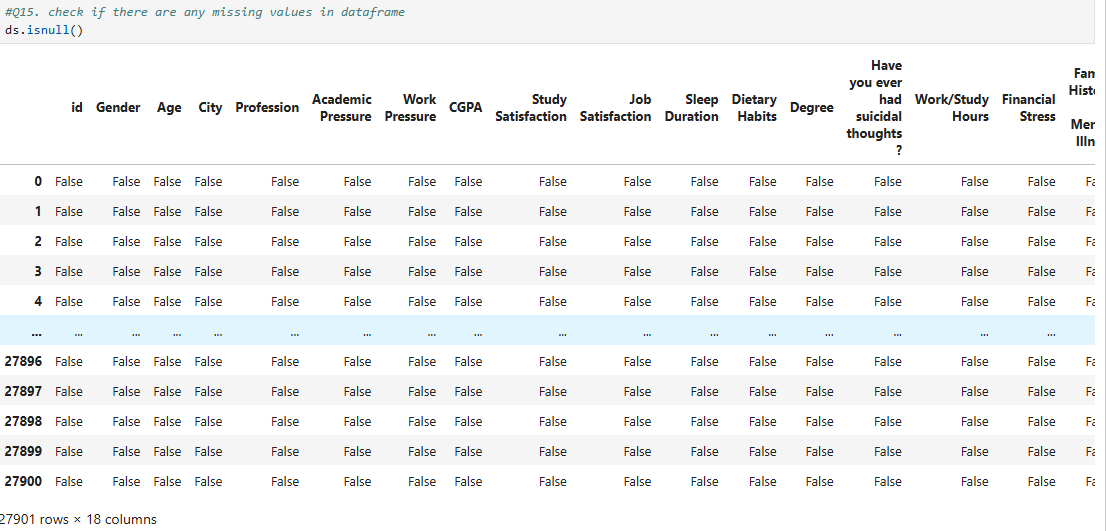
**Q.15. Sort the dataset or dataframe based on specific column**

**Ans:** ds.sort\_values(by="Age",ascending=False)



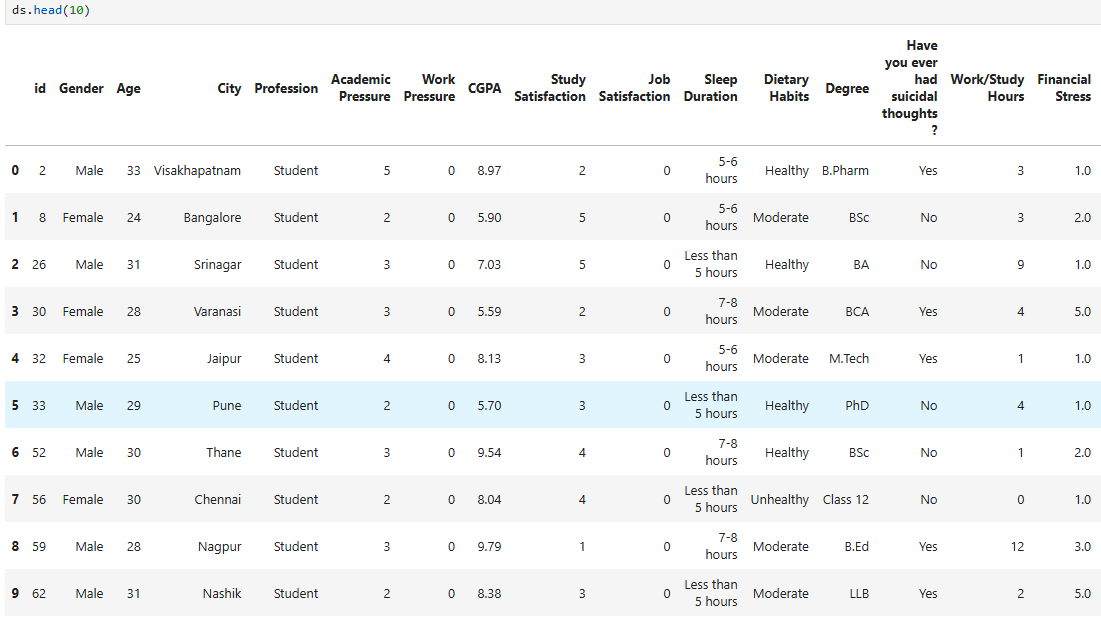
**Q.16. Check if there are any missing values in dataframe**

**Ans:** ds.isnull()



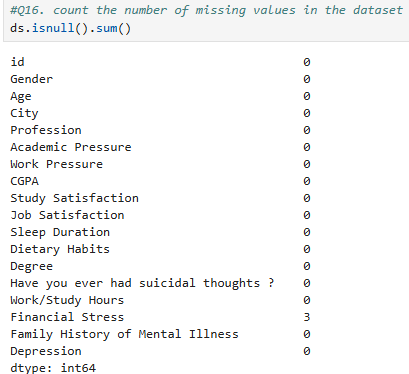
**Q.17.**

**Ans:** ds.head(10)



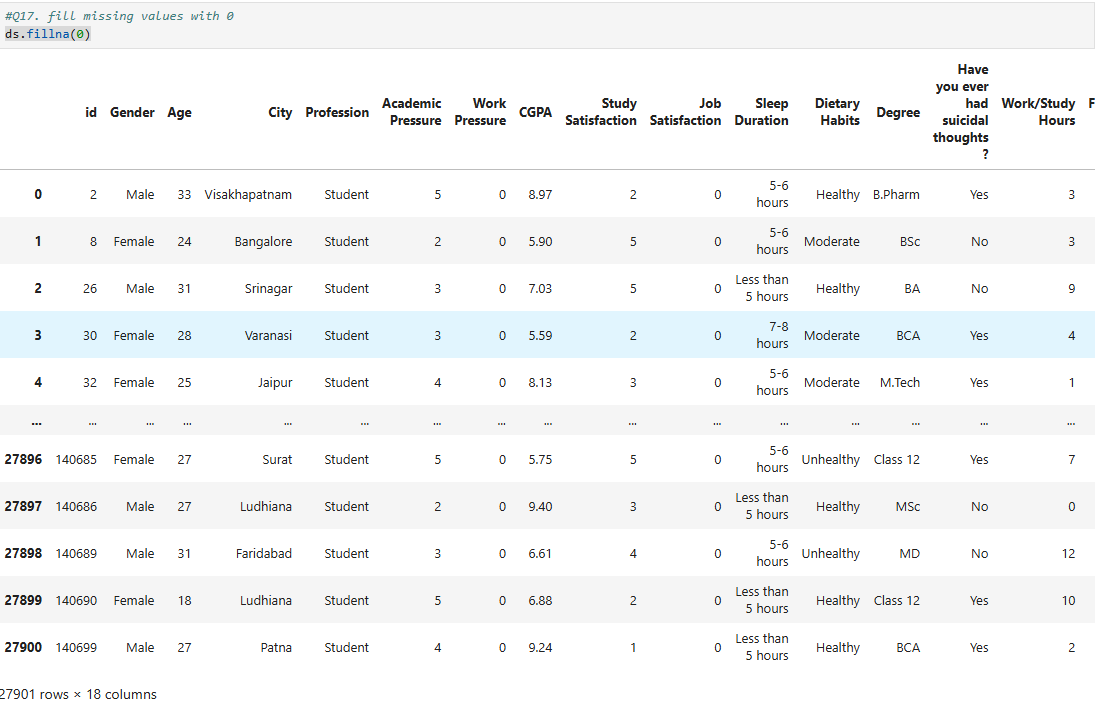
**Q.18. Count the number of missing values in the dataset**

**Ans:** ds.isnull().sum()



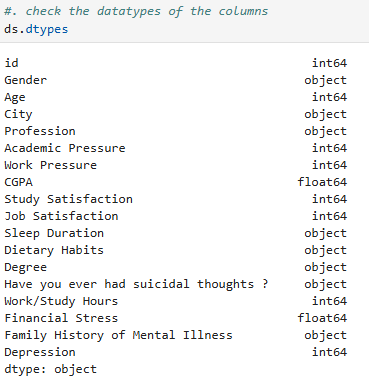
**Q.19. Fill missing values with 0**

**Ans:** ds.fillna(0)

****

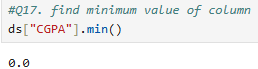
**Q.20. Check the datatypes of the columns**

**Ans:** ds.dtypes



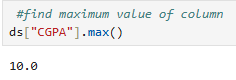
**Q.21. Find minimum value of column**

**Ans:** ds["CGPA"].min()



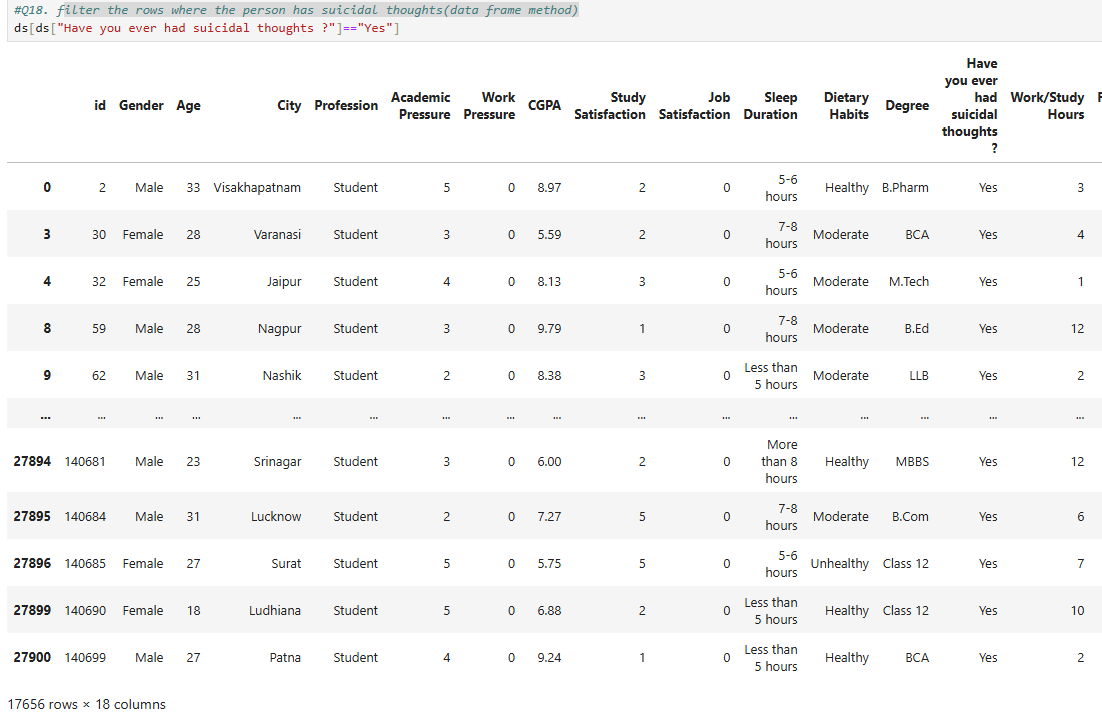
**Q.22. Find maximum value of column**

**Ans:** ds["CGPA"].max()



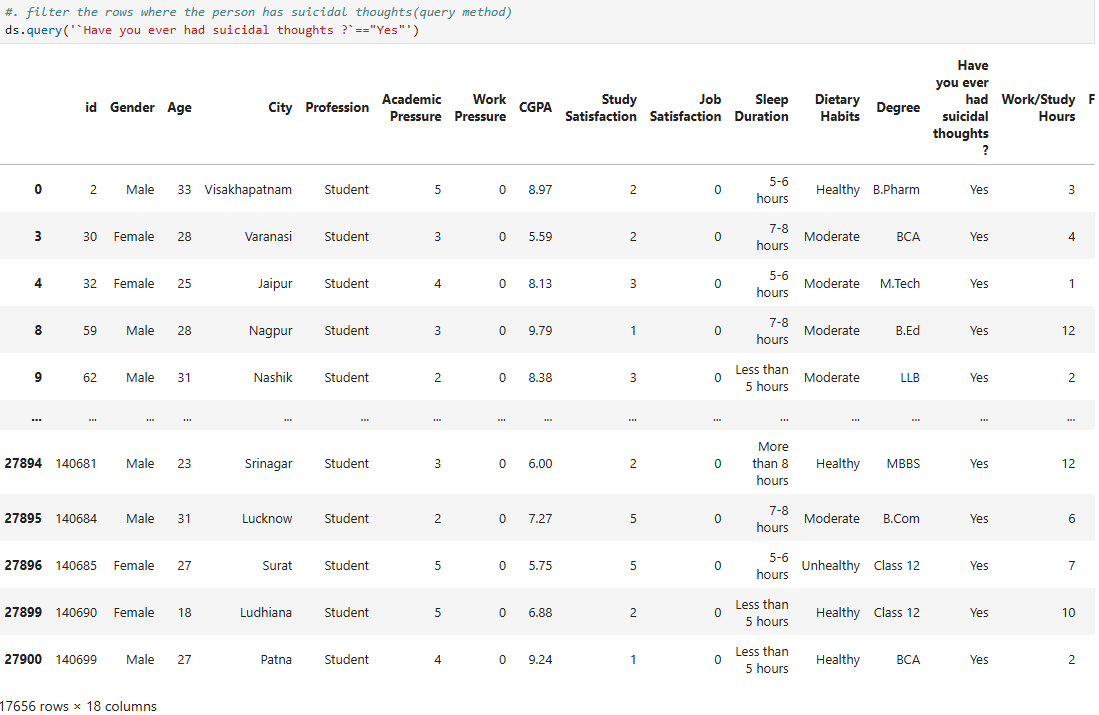
**Q.23. Filter the rows where the person has suicidal thoughts(data frame method)**

**Ans:** ds[ds["Have you ever had suicidal thoughts ?"]=="Yes"]



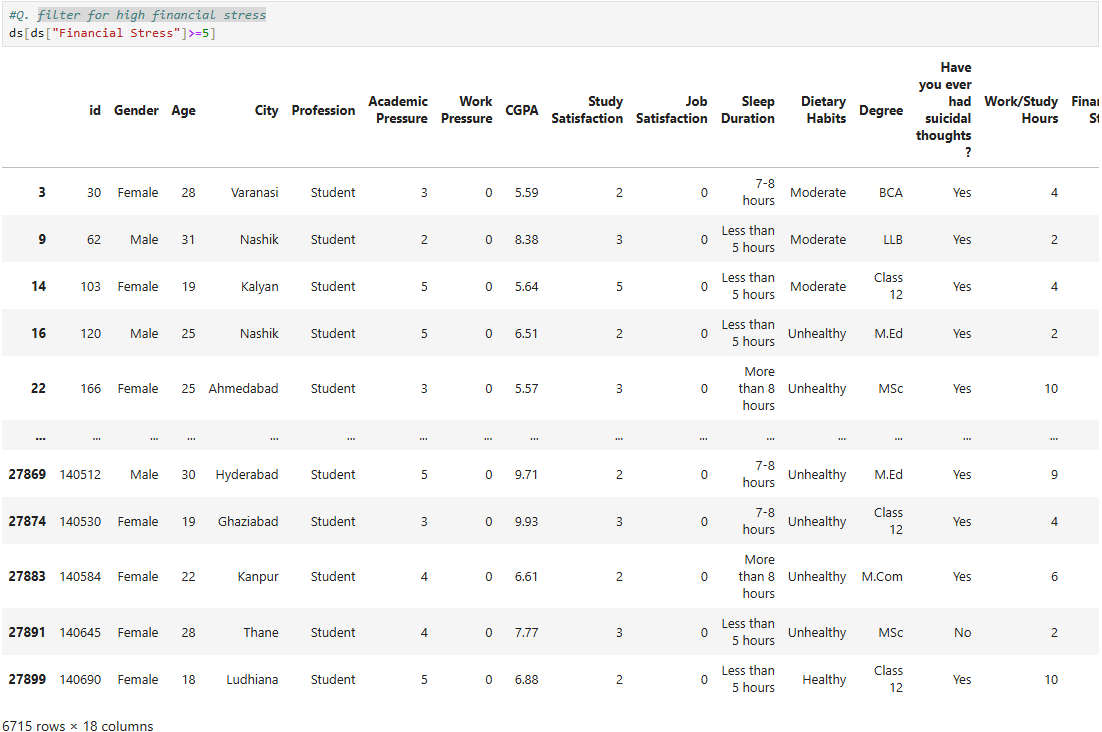
**Q.24. Filter the rows where the person has suicidal thoughts(query method)**

**Ans:** ds.query('`Have you ever had suicidal thoughts ?`=="Yes"')



**Q.25. Filter for high financial stress**

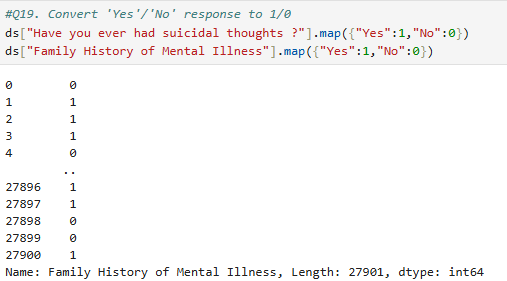
**Ans:** ds[ds["Financial Stress"]>=5]



**Q.26. Convert 'Yes'/'No' response to 1/0**

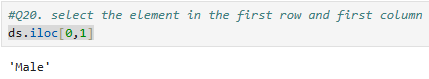
**Ans:** ds["Have you ever had suicidal thoughts ?"].map({"Yes":1,"No":0})

ds["Family History of Mental Illness"].map({"Yes":1,"No":0})



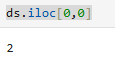
**Q.27. Select the element in the first row and first column**

**Ans:** ds.iloc[0,1]



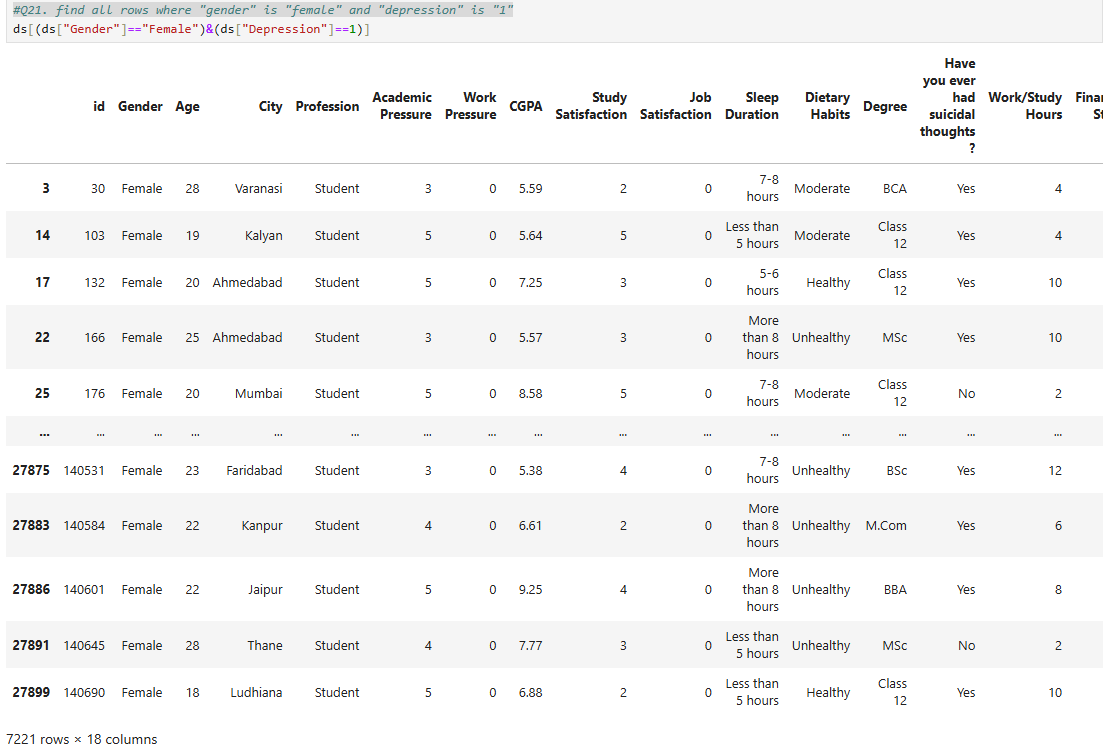
**Q.28.**

**Ans:** ds.iloc[0,0]



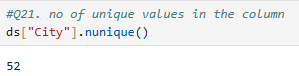
**Q.29. Find all rows where "gender" is "female" and "depression" is "1"**

**Ans:** ds[(ds["Gender"]=="Female")&(ds["Depression"]==1)]



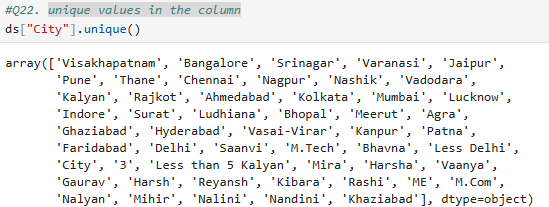
**Q.30. No. of unique values in the column**

**Ans:** ds["City"].nunique()



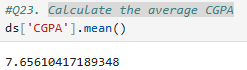
**Q.31. Unique values in the column**

**Ans:** ds["City"].unique()



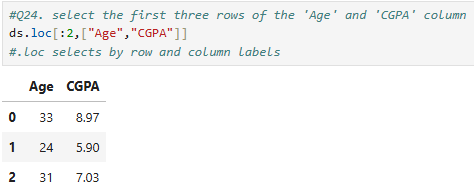
**Q.32. Calculate the average CGPA**

**Ans:** ds['CGPA'].mean()



**Q.33. Select the first three rows of the 'Age' and 'CGPA' column**

**Ans:** d.s.loc[:2,["Age","CGPA"]]



**Q.34. Change the sleep duration of the person with id=1 to 8 to 9 hour**

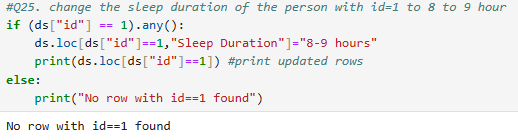
**Ans:** if (ds["id"] == 1).any():

ds.loc[ds["id"]==1,"Sleep Duration"]="8-9 hours"

print(ds.loc[ds["id"]==1]) #print updated rows

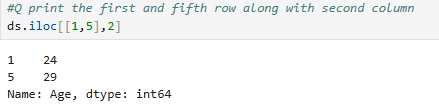
else:

print("No row with id==1 found")



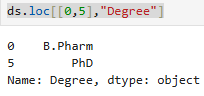
**Q.35. Print the first and fifth row along with second column**

**Ans:** ds.iloc[[1,5],2]



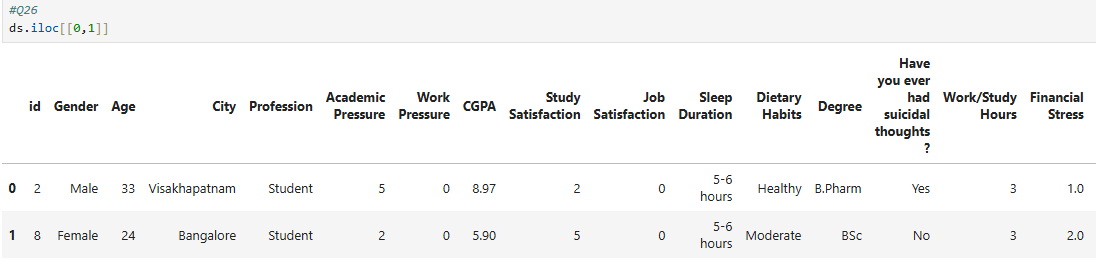
**Q.36.**

**Ans:** ds.loc[[0,5],"Degree"]



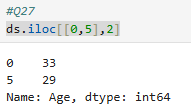
**Q.37.**

**Ans:** ds.iloc[[0,1]]



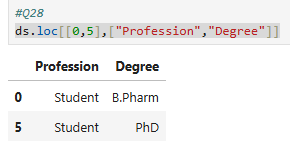
**Q.38.**

**Ans:** ds.iloc[[0,5],2]



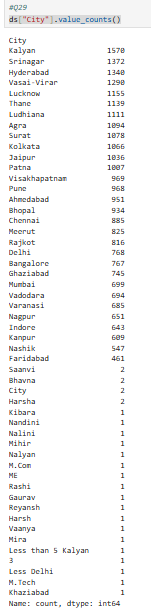
**Q.39.**

**Ans:** ds.loc[[0,5],["Profession","Degree"]]



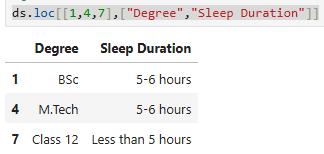
**Q.40.**

**Ans:** ds["City"].value\_counts()



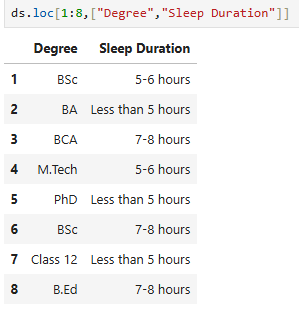
**Q.41.**

**Ans:** ds.loc[[1,4,7],["Degree","Sleep Duration"]]



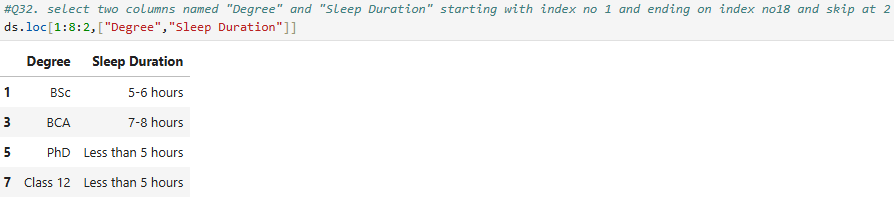
**Q.42.**

**Ans:** ds.loc[1:8,["Degree","Sleep Duration"]]



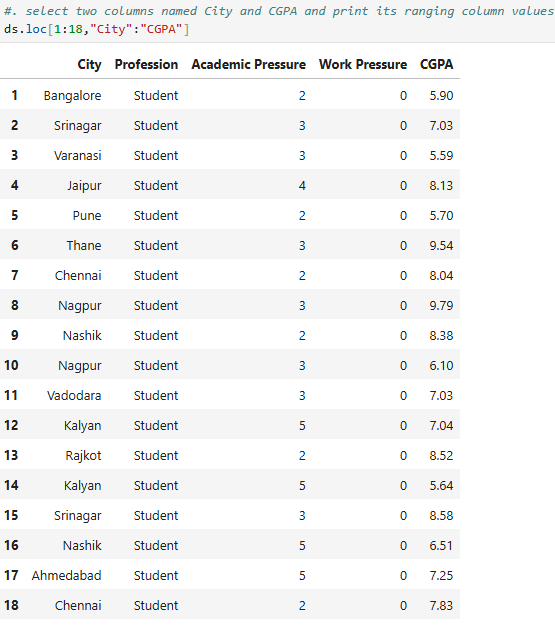
**Q.43. Select two columns named "Degree" and "Sleep Duration" starting with index no 1 and ending on index no18 and skip at 2**

**Ans:** ds.loc[1:8:2,["Degree","Sleep Duration"]]



**Q.44. Select two columns named City and CGPA and print its ranging column values**

**Ans:** ds.loc[1:18,"City":"CGPA"]



**Q.45. Write a code to filter a dataframe or dataset and select rows where the city column is kalyan,jaypur or delhi and then select only financial stress column from the filtered rows.**

**Ans:** ds[(ds["City"]=="Kalyan")|(ds["City"]=="Jaipur")|(ds["City"]=="Delhi")][["Financial Stress"]]



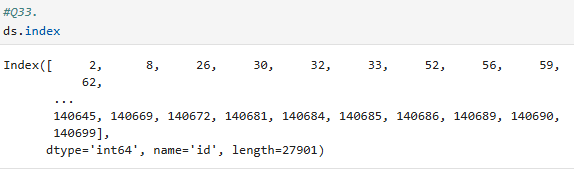
**Q.46.**

**Ans:** ds.set\_index('id',inplace=True)



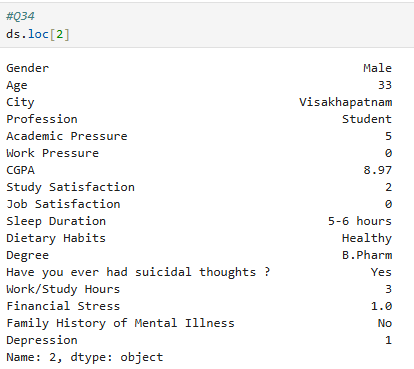
**Q.47.**

**Ans:** ds.index



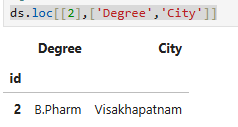
**Q.48.**

**Ans:** s.loc[2]



**Q.49.**

**Ans:** ds.loc[[2],['Degree','City']]

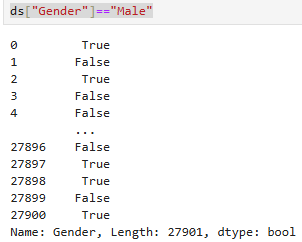


**Q.50.**

**Ans:** ds.reset\_index(inplace=True)

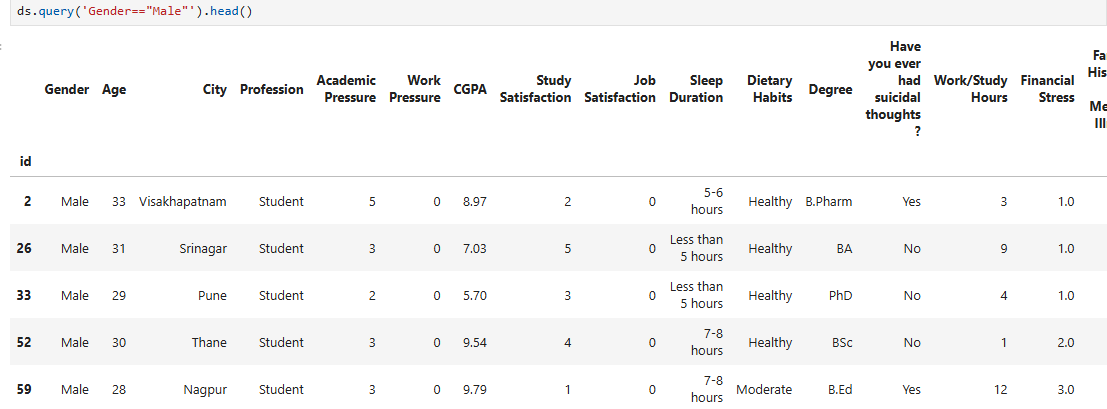
**Q.51.**

**Ans:** ds["Gender"]=="Male"



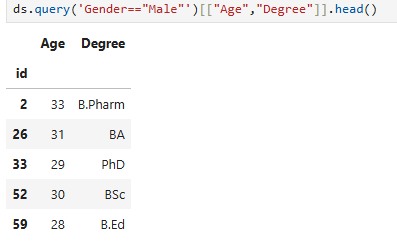
**Q.52.**

**Ans:** ds.query('Gender=="Male"').head()



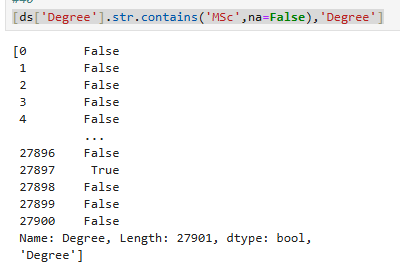
**Q.53.**

**Ans:** ds.query('Gender=="Male"')[["Age","Degree"]].head()



**Q.54.**

**Ans:** [ds['Degree'].str.contains('MSc',na=False),'Degree']



**Q.55. Sample data for Indian students(Dictionary)**

**Ans:** data={

"Name":["Aarav","Isha","Vivaan","Priya","Reyansh"],

"Age":[20,21,22,23,24],

"Grade":["A","B","A","C","B"],

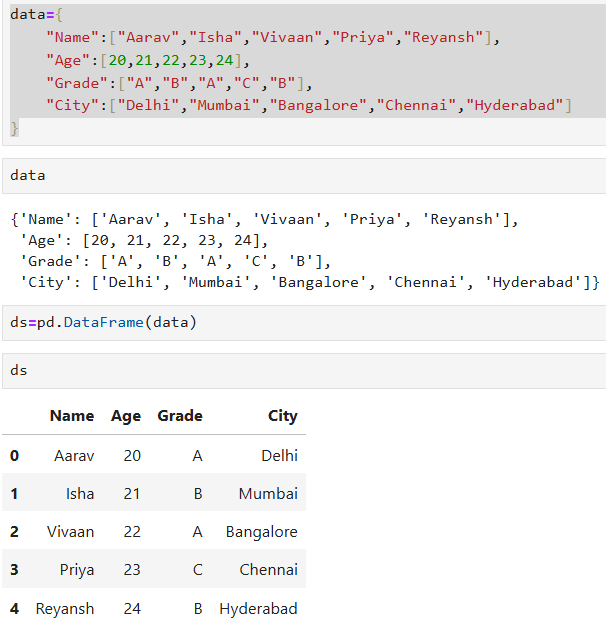
"City":["Delhi","Mumbai","Bangalore","Chennai","Hyderabad"]

}

data

ds=pd.DataFrame(data)

ds



**Q.56. Create dataframe using zip() to combine the lists into a dictionary**

**Ans:** names=["Aarav","Isha","Vivaan","Priya","Reyansh"];

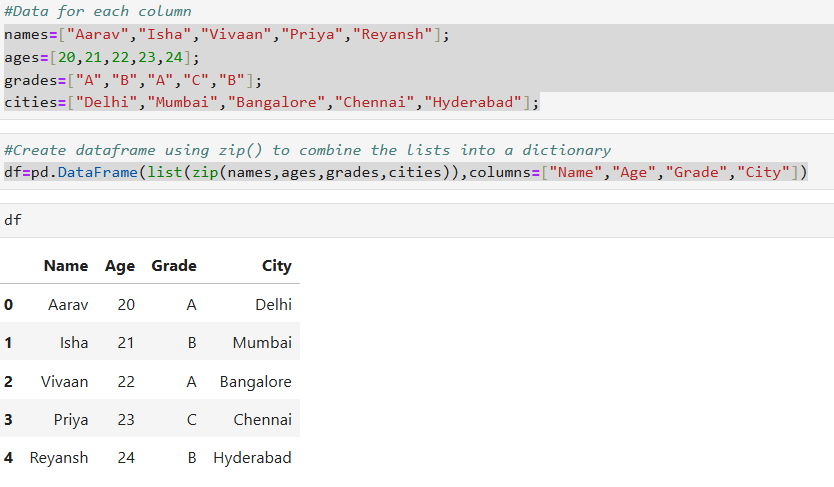
ages=[20,21,22,23,24];

grades=["A","B","A","C","B"];

cities=["Delhi","Mumbai","Bangalore","Chennai","Hyderabad"];

df=pd.DataFrame(list(zip(names,ages,grades,cities)),columns=["Name","Age","Grade","City"])

df



**Q.57. Using numpy array method**

**Ans:** import numpy as np

names=np.array(["Aarav","Isha","Vivaan","Priya","Reyansh"]);

ages=np.array([20,21,22,23,24]);

grades=np.array(["A","B","A","C","B"]);

cities=np.array(["Delhi","Mumbai","Bangalore","Chennai","Hyderabad"]);

df=pd.DataFrame({

"Name":names,

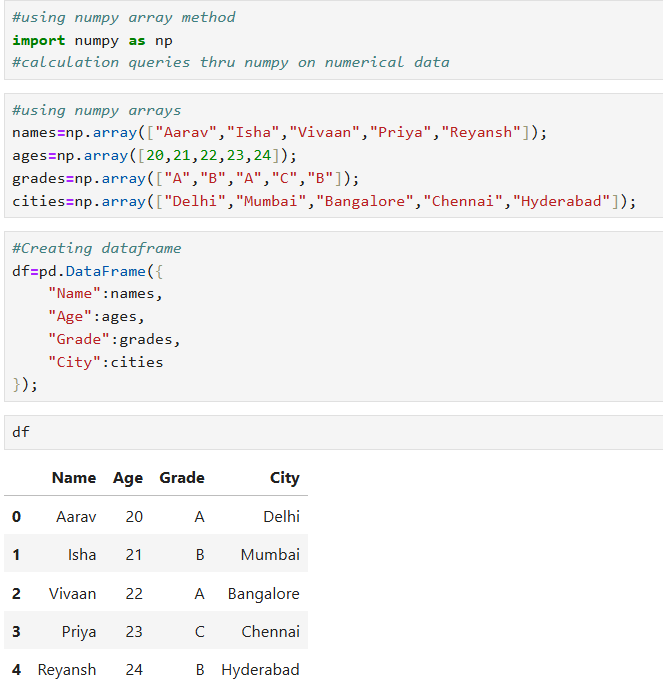
"Age":ages,

"Grade":grades,

"City":cities

});

df

****

**Q.58. Dictionary of data**

**Ans:** data={

"Name":["Aarav","Isha","Vivaan","Priya","Reyansh"],

"Age":[20,21,22,23,24],

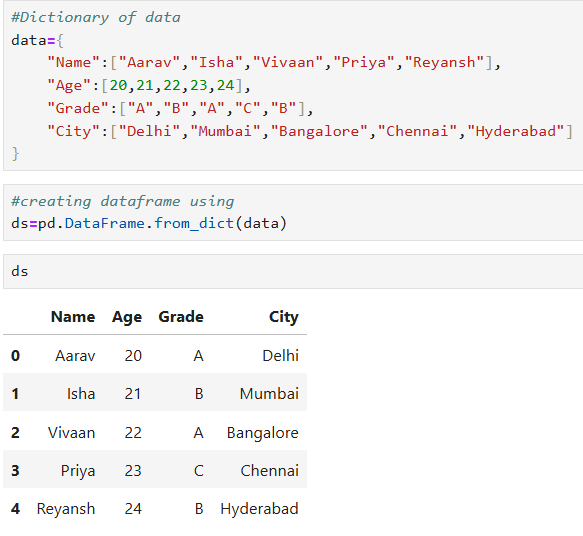
"Grade":["A","B","A","C","B"],

"City":["Delhi","Mumbai","Bangalore","Chennai","Hyderabad"]

}

ds=pd.DataFrame.from\_dict(data)

ds



**Q.59. Using list from dictionary**

**Ans:** data=[

{"Name":"Aarav","Age":20,"Grade":"A","City":"Delhi"},

{"Name":"Isha","Age":21,"Grade":"B","City":"Mumbai"},

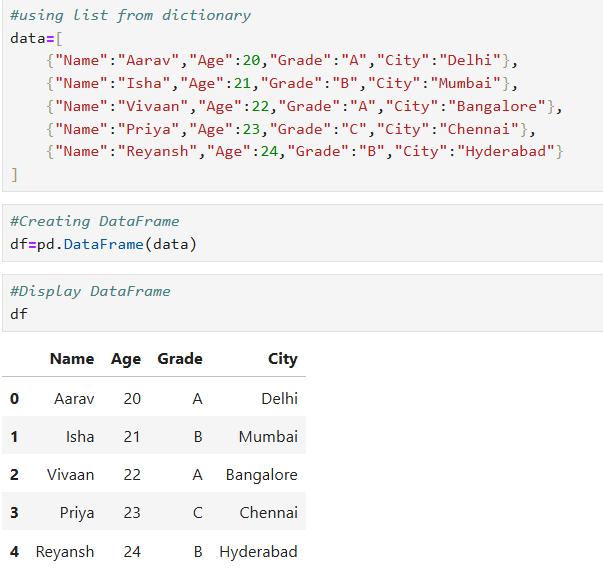
{"Name":"Vivaan","Age":22,"Grade":"A","City":"Bangalore"},

{"Name":"Priya","Age":23,"Grade":"C","City":"Chennai"},

{"Name":"Reyansh","Age":24,"Grade":"B","City":"Hyderabad"}

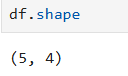
df=pd.DataFrame(data)

df



**Q.60. What is the shape of the table?**

**Ans:** df.shape

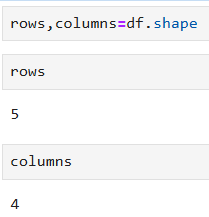


**Q.61. What is the shape of rows and columns?**

**Ans:** rows,columns=df.shape

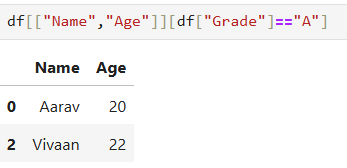
rows

columns



**Q.62. Select “Name” and “Age” Columns with Grades = “A”**

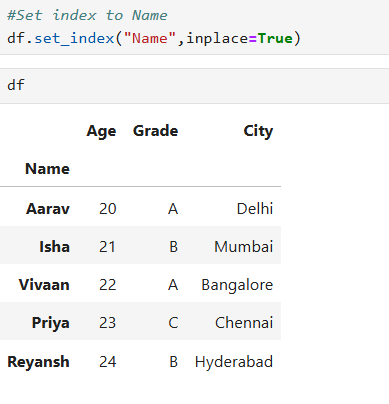
**Ans:** df[["Name","Age"]][df["Grade"]=="A"]



**Q.63. Set index to Name**

**Ans:** df.set\_index("Name",inplace=True)

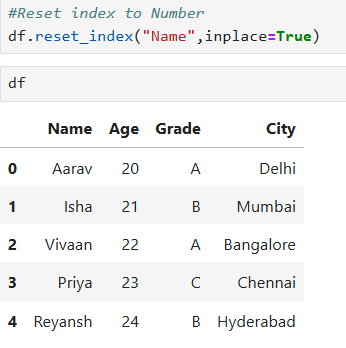
df



**Q.64. Reset index to Number**

**Ans:** df.reset\_index("Name",inplace=True)

df



**Q.65.**

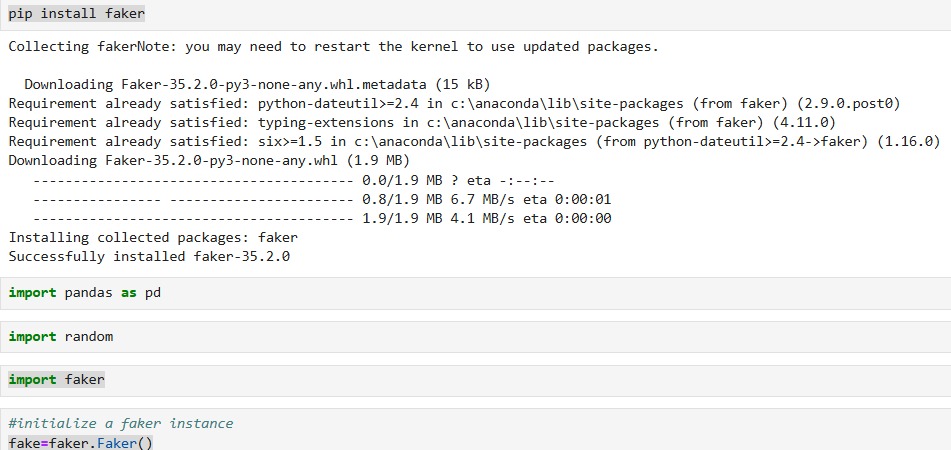
**Ans:** pip install faker

import pandas as pd

Import random

import faker

fake=faker.Faker()



**Q.66. sample data Indian first and last names(including Muslim names)**

**Ans:** data={

"First Name": ["Aarav","Isha","Mohammad","Zoya","ayaan","Priya","Reeta"],

"Last Name": ["Sharma","Patel","Khan","Shaikh","Ansari","Verma","Sharif"],

"Email ID": [fake.email() for \_ in range(7)], #Random email using faker

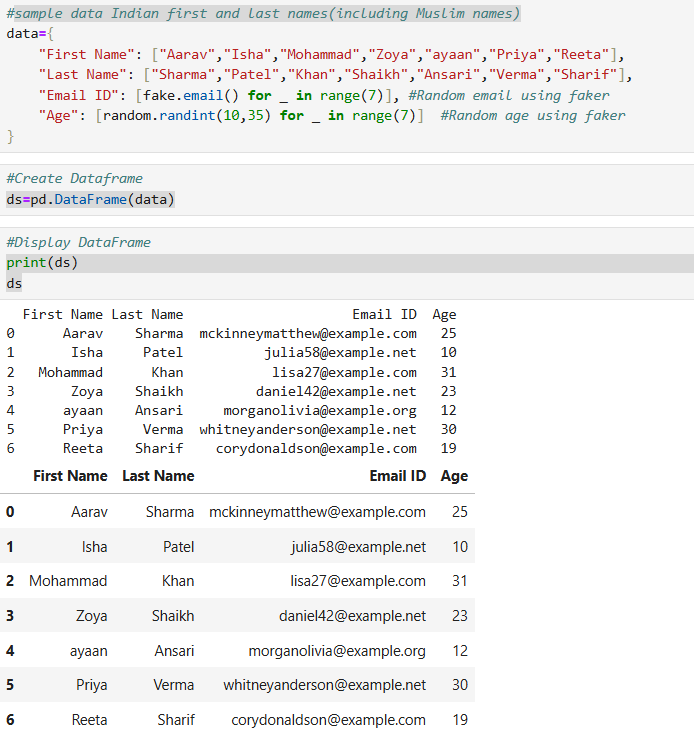
"Age": [random.randint(10,35) for \_ in range(7)] #Random age using faker

}

ds=pd.DataFrame(data)

print(ds)

ds



**Q.67. Generating random values**

**Ans:** print(fake.name())

print(fake.address())

print(fake.email())

print(fake.phone\_number())



**Q.68.**

**Ans:** data=[]

for \_ in range(7):

record={

"Name":fake.name(),

"Address":fake.address(),

"Email":fake.email(),

"Date of Birth":fake.date\_of\_birth(),

"Phone Number":fake.phone\_number(),

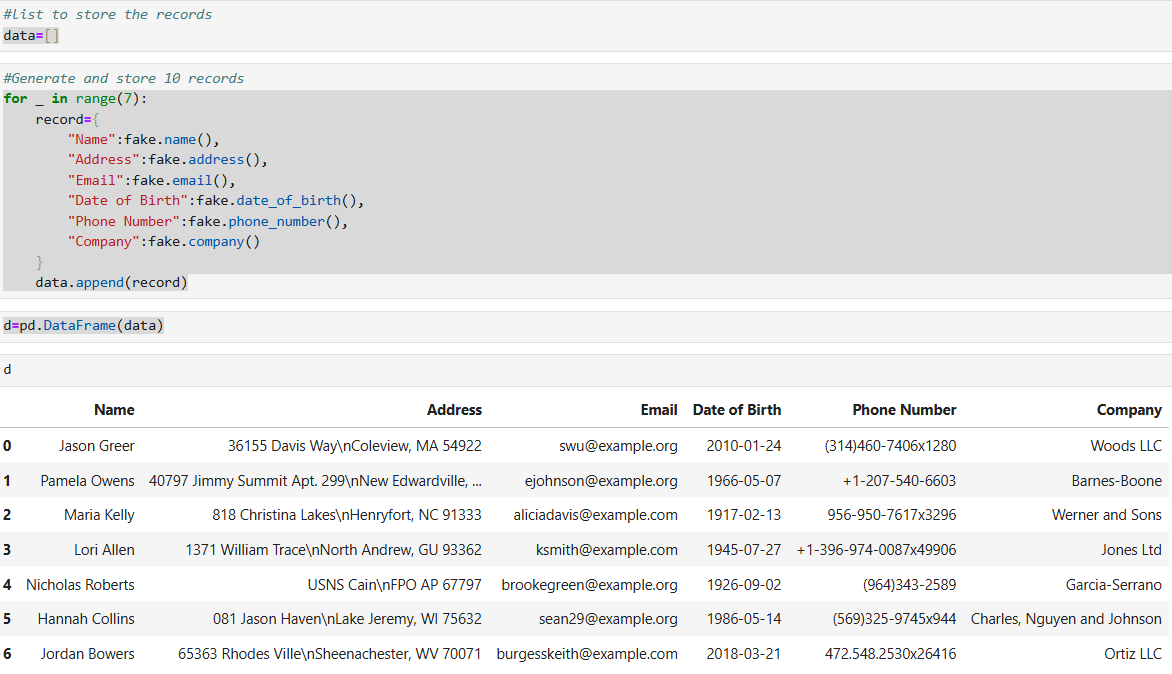
"Company":fake.company()

}

data.append(record)

d=pd.DataFrame(data)

d



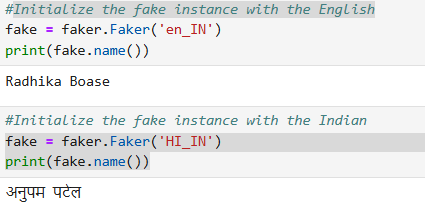
**Q.69. Initialize the fake instance with the English and Hindi**

**Ans:** fake = faker.Faker('en\_IN')

print(fake.name())

fake = faker.Faker('HI\_IN')

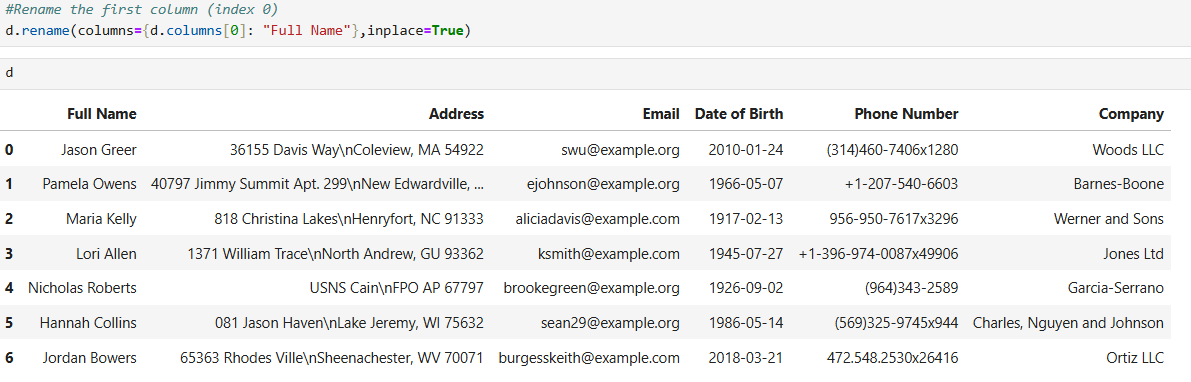
print(fake.name())



**Q.70. Rename the first column (index 0)**

**Ans:** d.rename(columns={d.columns[0]: "Full Name"},inplace=True)

d



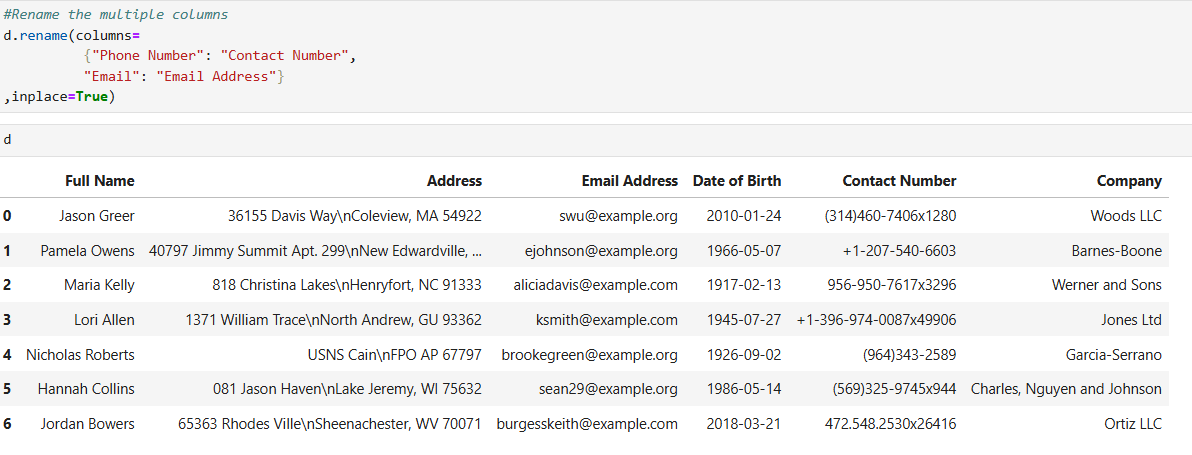
**Q.71. Rename the multiple columns**

**Ans:** d.rename(columns=

{"Phone Number": "Contact Number",

"Email": "Email Address"}

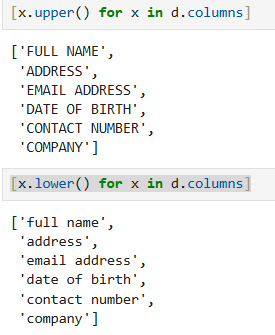
,inplace=True)



**Q.72.**

**Ans:** [x.upper() for x in d.columns]

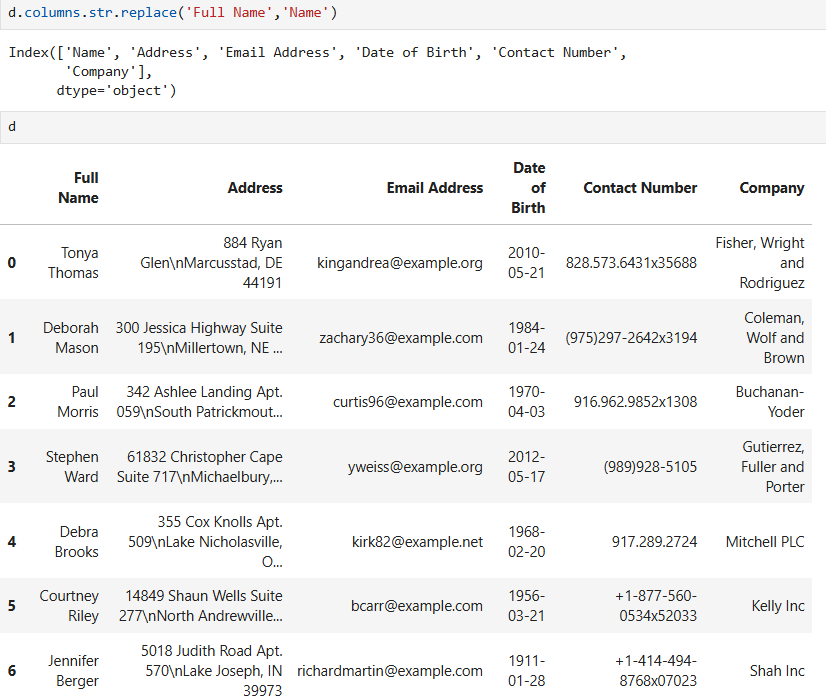
[x.lower() for x in d.columns]



**Q.73.**

**Ans:** d.columns.str.replace('Full Name','Name')

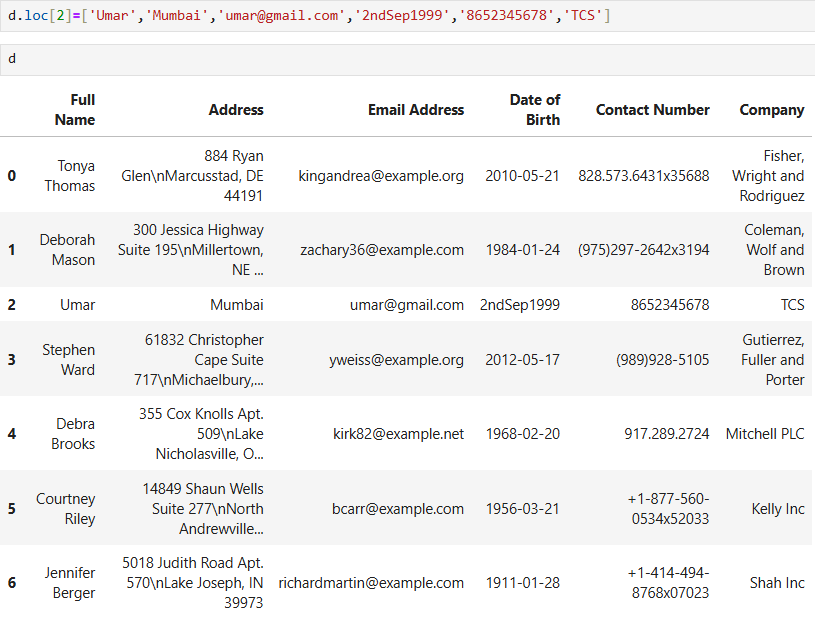
d



**Q.74.**

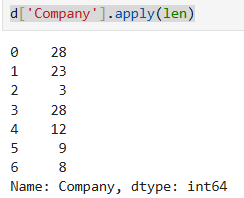
**Ans:** d.loc[2]=['Umar','Mumbai','umar@gmail.com','2ndSep1999','8652345678','TCS']

d



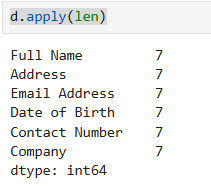
**Q.75.**

**Ans:** d['Company'].apply(len)



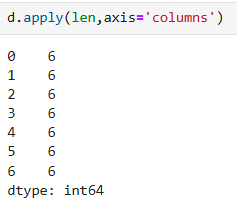
**Q.76.**

**Ans:** d.apply(len)



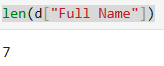
**Q.77.**

**Ans:** d.apply(len,axis='columns')

****

**Q.78.**

**Ans:** len(d["Full Name"])

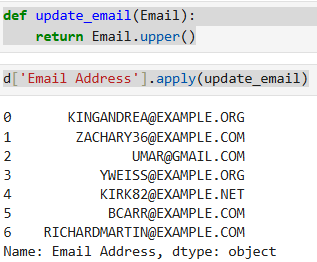
****

**Q.79.**

**Ans:** def update\_email(Email):

return Email.upper()

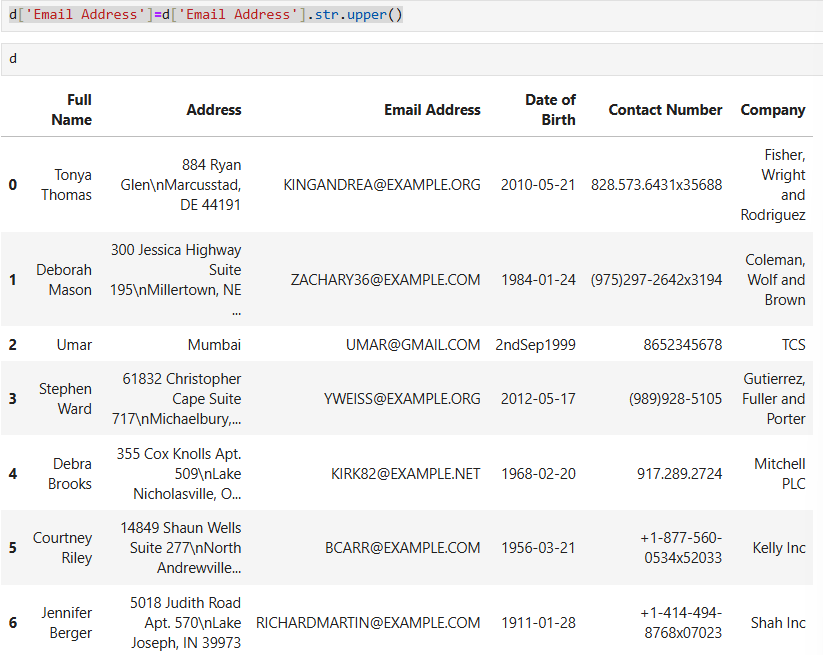
d['Email Address'].apply(update\_email)



**Q.80.**

**Ans:** d['Email Address']=d['Email Address'].str.upper()

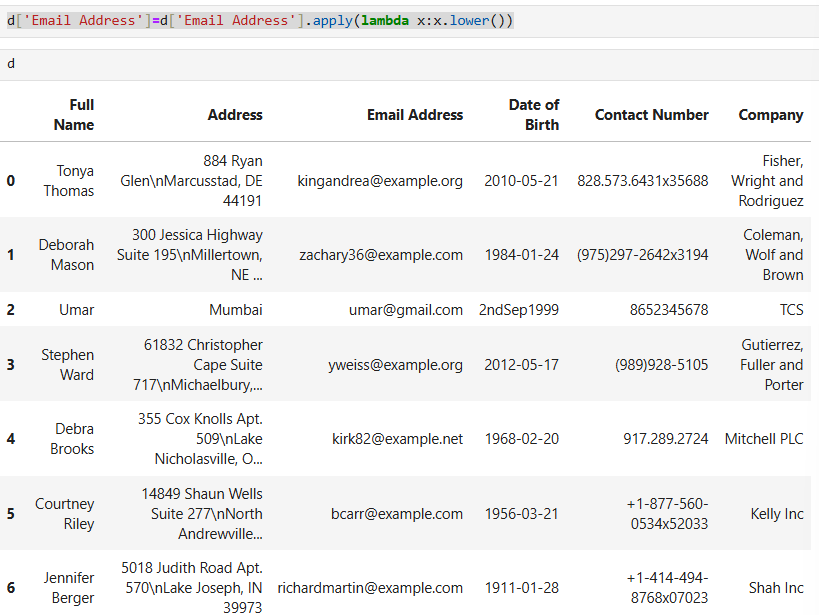
d



**Q.81.**

**Ans:** d['Email Address']=d['Email Address'].apply(lambda x:x.lower())

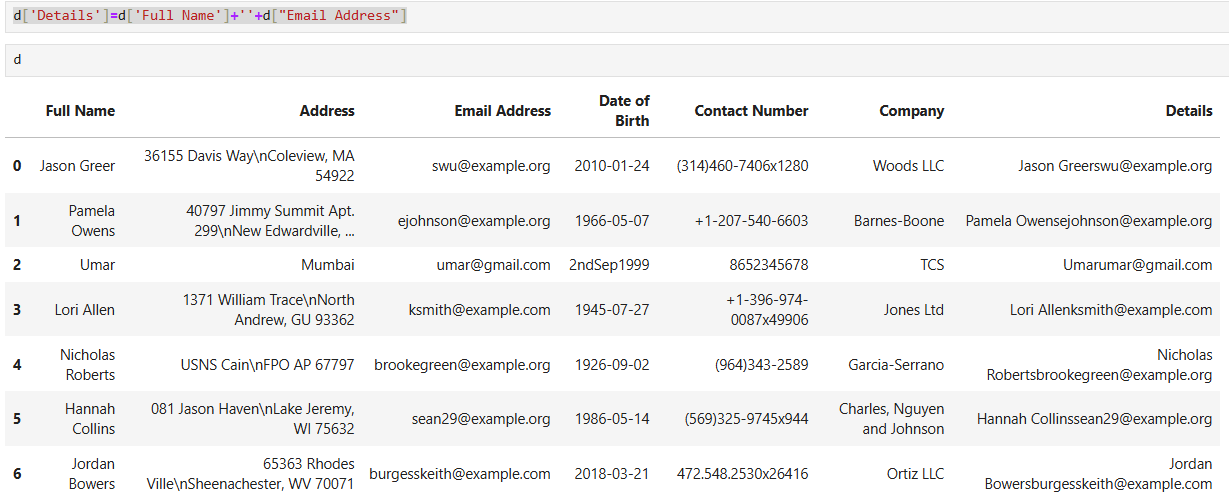
d

****

**Q.82.**

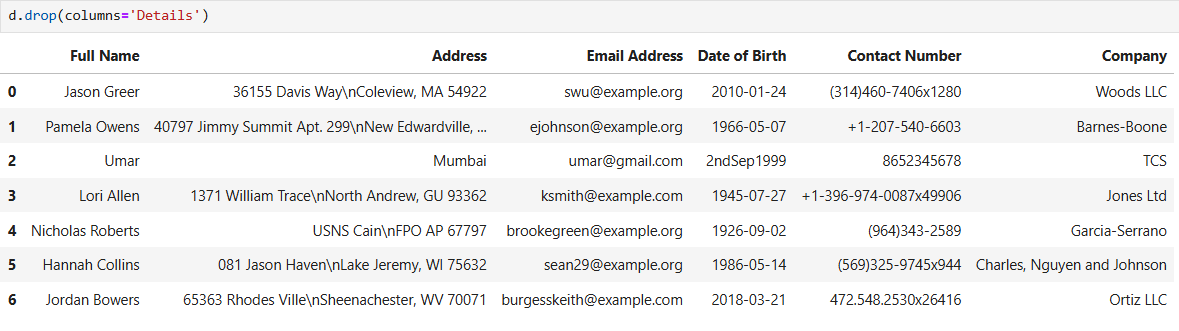
**Ans:** d['Details']=d['Full Name']+''+d["Email Address"]

d

****

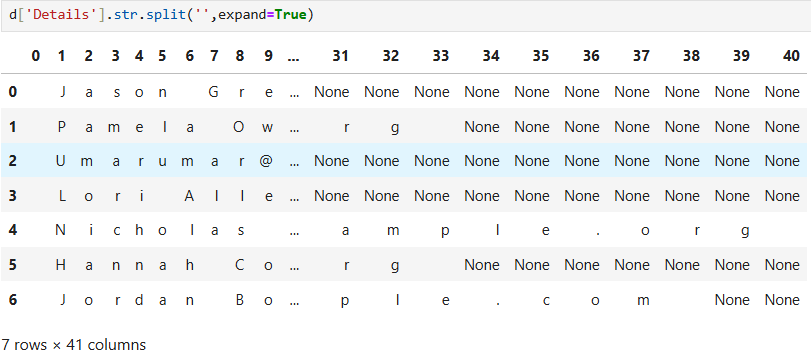
**Q.83.**

**Ans:** d.drop(columns='Details')



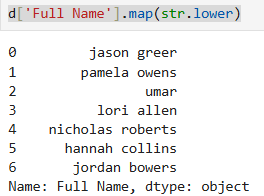
**Q.84.**

**Ans:** d['Details'].str.split('',expand=True)



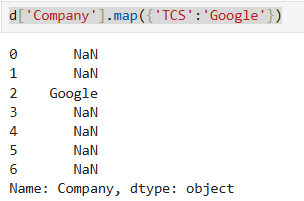
**Q.85.**

**Ans:** d['Full Name'].map(str.lower)



**Q.86.**

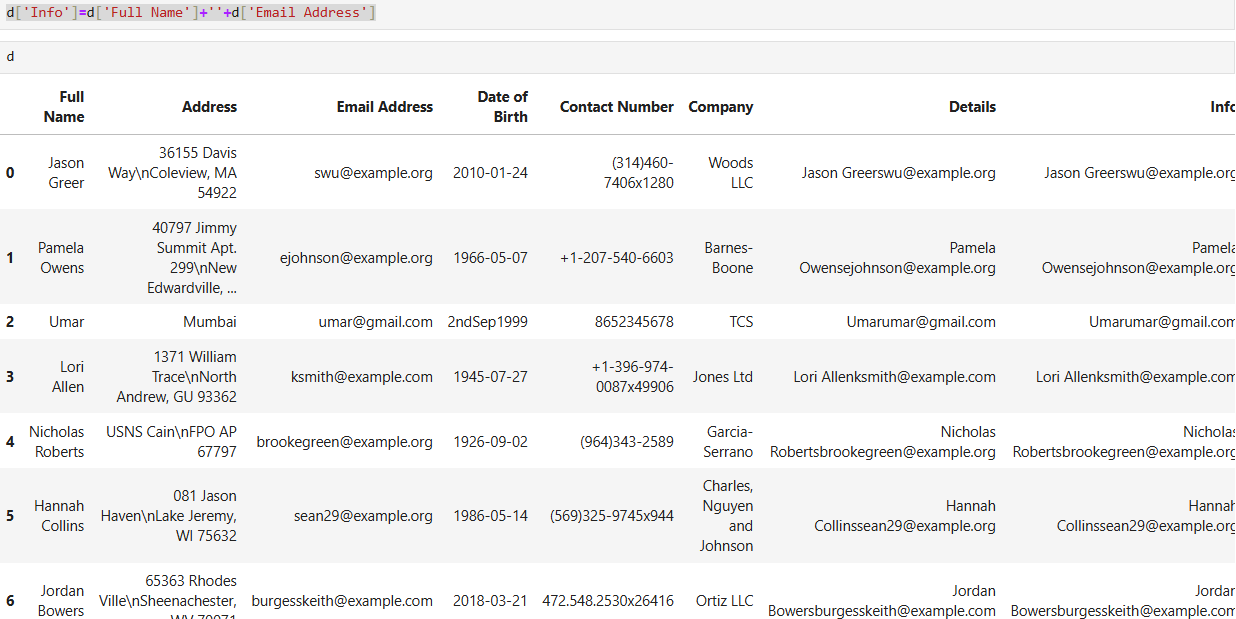
**Ans:** d['Company'].map({'TCS':'Google'})



**Q.87.**

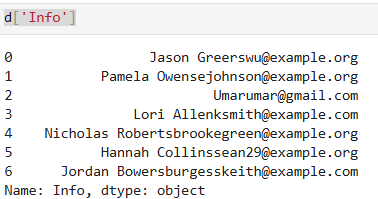
**Ans:** d['Info']=d['Full Name']+''+d['Email Address']

d



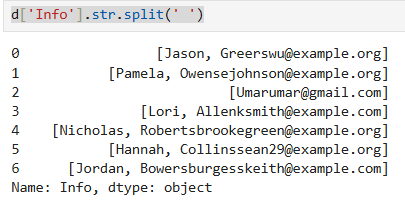
**Q.88.**

**Ans:** d['Info']



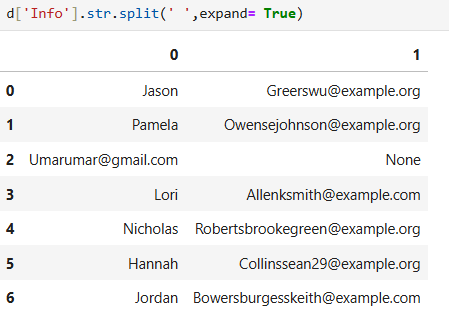
**Q.89.**

**Ans:** d['Info'].str.split(' ')



**Q.90.**

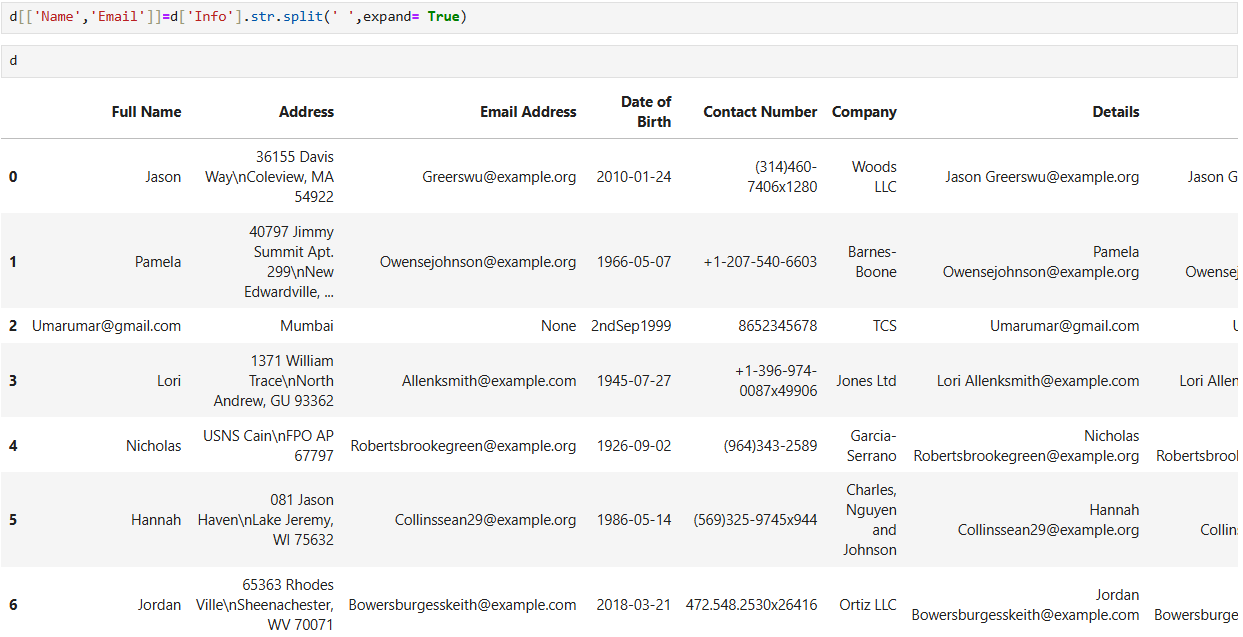
**Ans:** d['Info'].str.split(' ',expand= True)



**Q.91.**

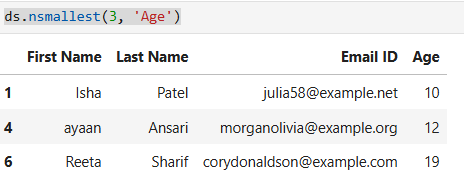
**Ans:** d[['Name','Email']]=d['Info'].str.split(' ',expand= True)

d



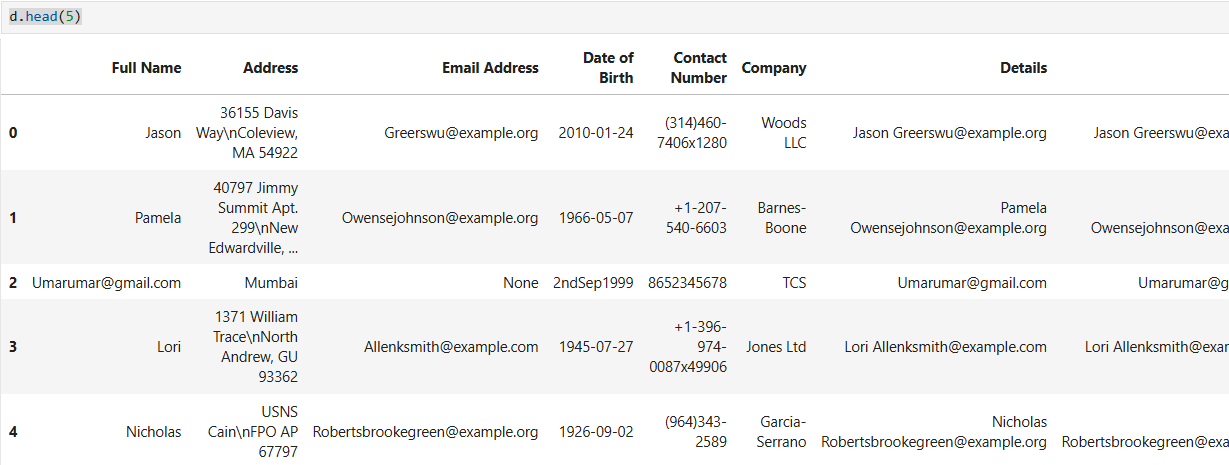
**Q.92.**

**Ans:** ds.nsmallest(3, 'Age')



**Q.93.**

**Ans:** d.head(5)

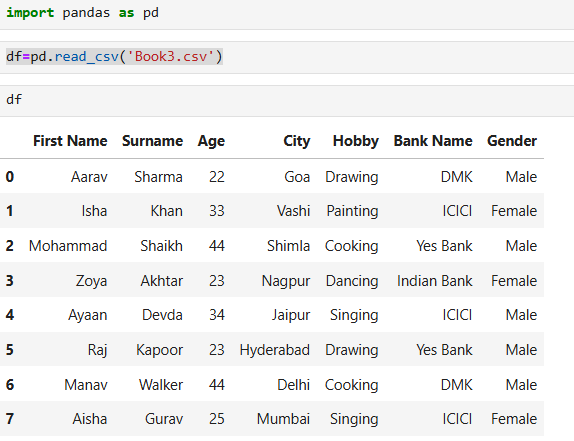


**Q.94.**

**Ans:** import pandas as pd

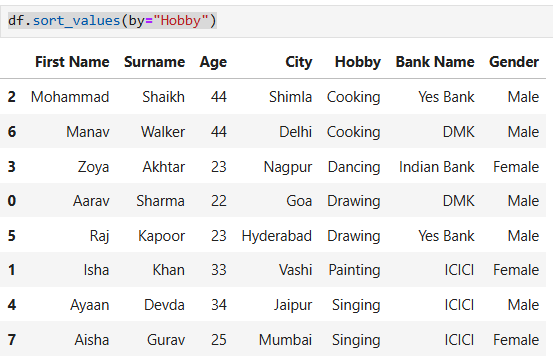
df=pd.read\_csv('Book3.csv')

df



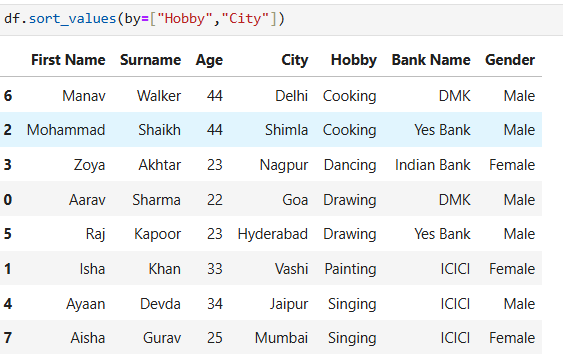
**Q.95.**

**Ans:** df.sort\_values(by="Hobby")



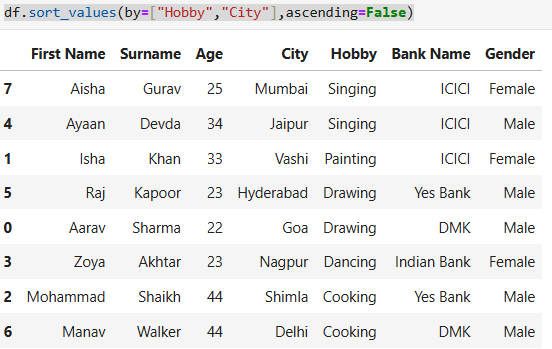
**Q.96.**

**Ans:** df.sort\_values(by=["Hobby","City"])



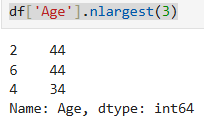
**Q.97.**

**Ans:** df.sort\_values(by=["Hobby","City"],ascending=False)



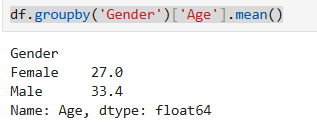
**Q.98.**

**Ans:** df['Age'].nlargest(3)



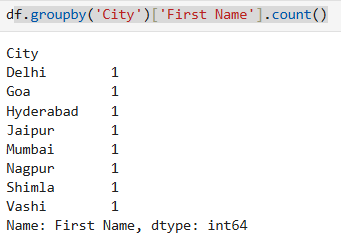
**Q.99.**

**Ans:** df.groupby('Gender')['Age'].mean()



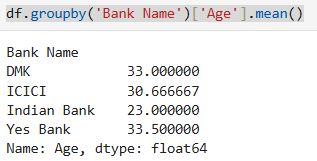
**Q.100.**

**Ans:** f.groupby('City')['First Name'].count()



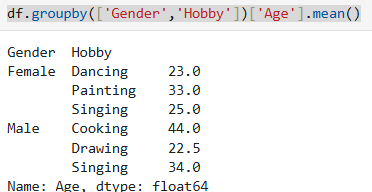
**Q.101.**

**Ans:** df.groupby('Bank Name')['Age'].mean()



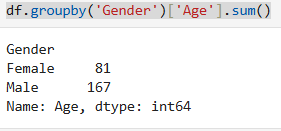
**Q.102.**

**Ans:** df.groupby(['Gender','Hobby'])['Age'].mean()



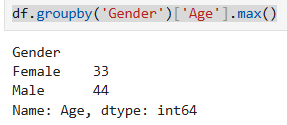
**Q.103.**

**Ans:** df.groupby('Gender')['Age'].sum()



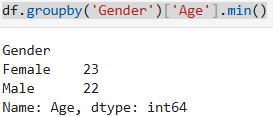
**Q.104.**

**Ans:** df.groupby('Gender')['Age'].max()



**Q.105.**

**Ans:** df.groupby('Gender')['Age'].min()



**Q.106.**

**Ans:** df.groupby('Gender').agg(

{

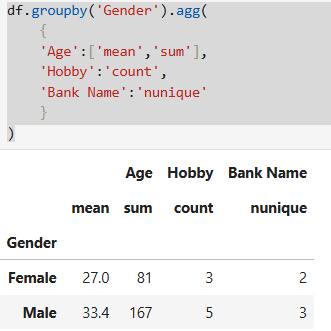
'Age':['mean','sum'],

'Hobby':'count',

'Bank Name':'nunique'

}

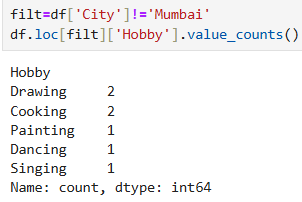
)



**Q.107.**

**Ans:** filt=df['City']!='Mumbai'

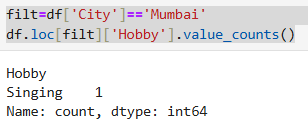
df.loc[filt]['Hobby'].value\_counts()



**Q.108.**

**Ans:** filt=df['City']=='Mumbai'

df.loc[filt]['Hobby'].value\_counts()



**Q.109.**

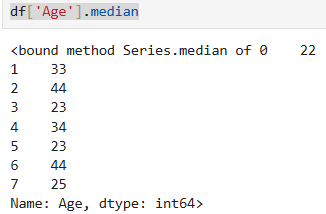
**Ans:** filt=df['Age']<24

df.loc[filt]['City'].str.contains('Mumbai').sum()



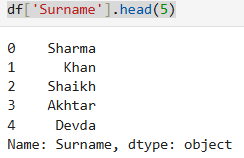
**Q.110.**

**Ans:** df['Age'].median



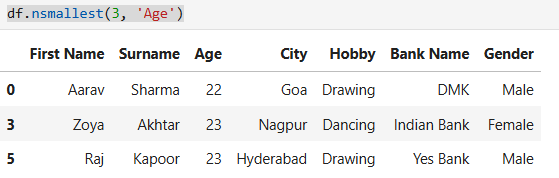
**Q.111.**

**Ans:** df['Surname'].head(5)



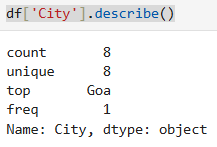
**Q.112.**

**Ans:** df.nsmallest(3, 'Age')



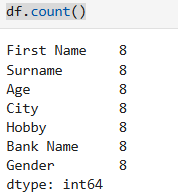
**Q.113.**

**Ans:** df['City'].describe()



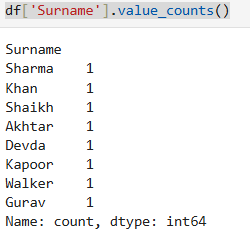
**Q.114.**

**Ans:** df.count()



**Q.115.**

**Ans:** df['Surname'].value\_counts()



**Q.116.**

**Ans:** import numpy as np

people = {

'first': ['Corey', 'Jane', 'John', 'Chris', np.nan, None, 'NA'],

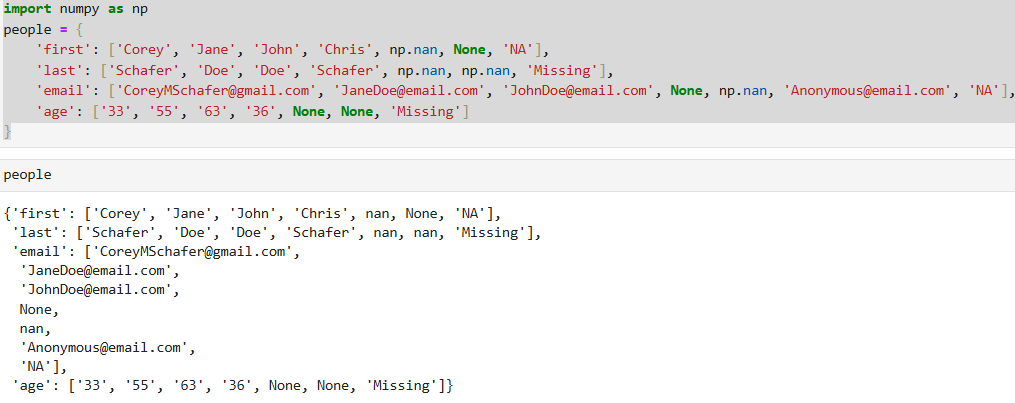
'last': ['Schafer', 'Doe', 'Doe', 'Schafer', np.nan, np.nan, 'Missing'],

'email': ['CoreyMSchafer@gmail.com', 'JaneDoe@email.com', 'JohnDoe@email.com', None, np.nan, 'Anonymous@email.com', 'NA'],

'age': ['33', '55', '63', '36', None, None, 'Missing']

}

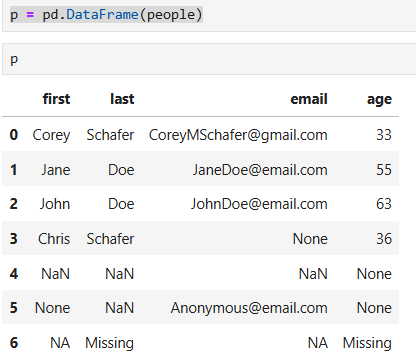
people



**Q.117.**

**Ans:** p = pd.DataFrame(people)

p

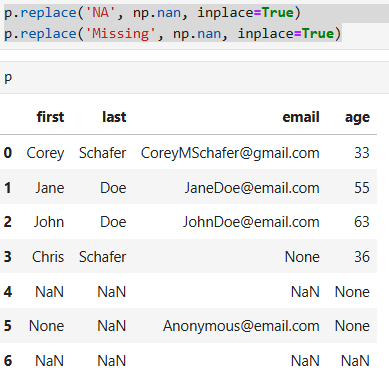


**Q.118.**

**Ans:** p.replace('NA', np.nan, inplace=True)

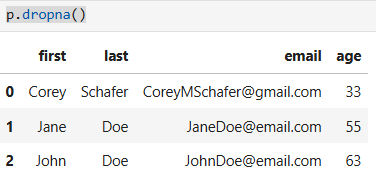
p.replace('Missing', np.nan, inplace=True)

p



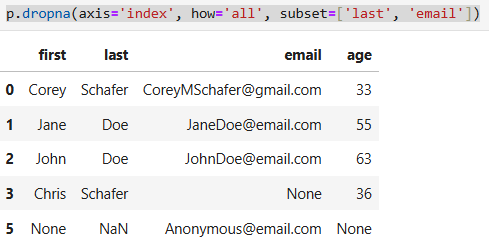
**Q.119.**

**Ans:** p.dropna()

S

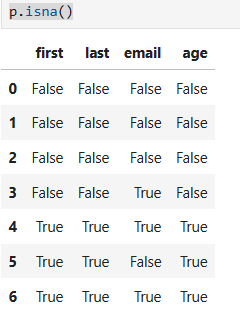
**Q.120.**

**Ans:** p.dropna(axis='index', how='all', subset=['last', 'email'])



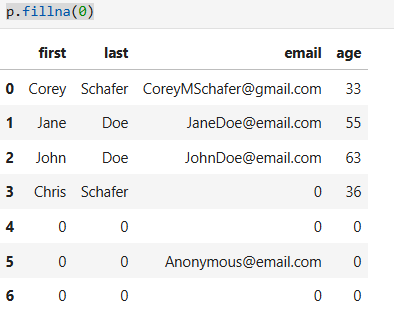
**Q.121.**

**Ans:** p.isna()



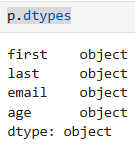
**Q.122.**

**Ans:** p.fillna(0)



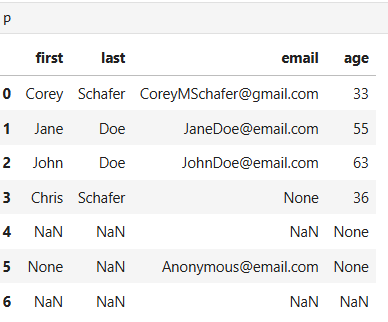
**Q.123.**

**Ans:** p.dtypes



**Q.124.**

**Ans:** p

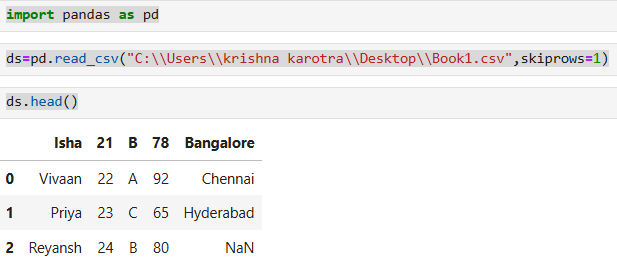


**Q.125.**

**Ans:** import pandas as pd

ds=pd.read\_csv("C:\\Users\\krishna karotra\\Desktop\\Book1.csv",skiprows=1)

ds.head()



**Q.126.**

**Ans:** import pandas as pd

import matplotlib.pyplot as plt

x=[10,12]

y=[2,8]

plt.plot(x,y,marker='\*',color='blue',linestyle='-',linewidth='5', markersize=15)

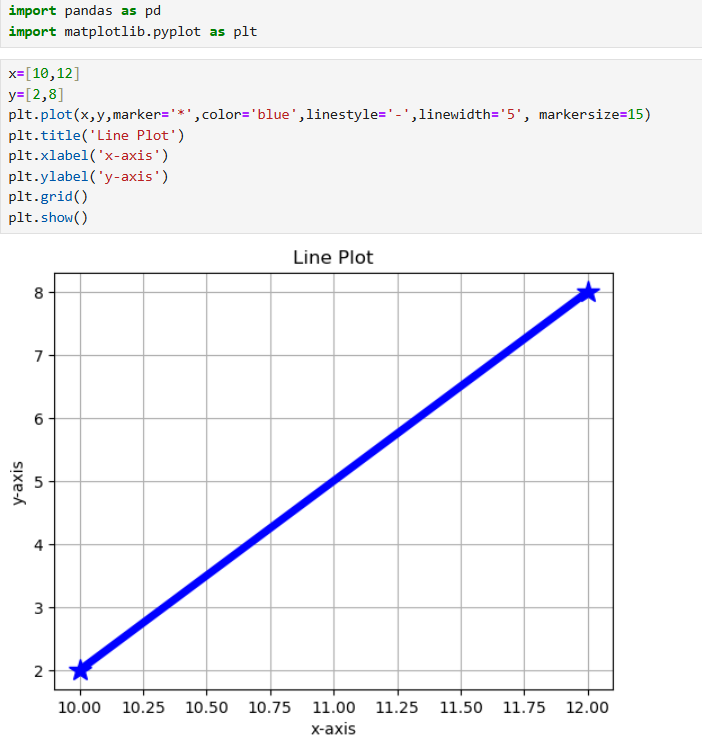
plt.title('Line Plot')

plt.xlabel('x-axis')

plt.ylabel('y-axis')

plt.grid()

plt.show()



**Q.127.**

**Ans:** x=[10,12]

y=[2,8]

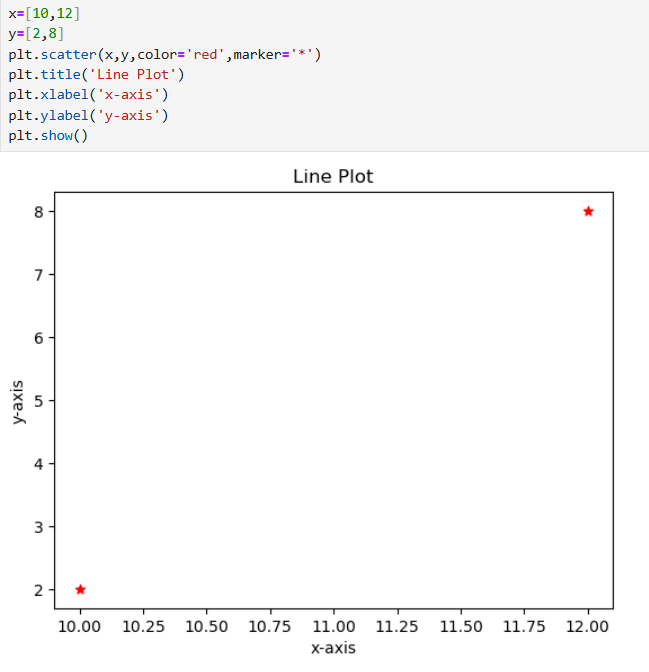
plt.scatter(x,y,color='red',marker='\*')

plt.title('Line Plot')

plt.xlabel('x-axis')

plt.ylabel('y-axis')

plt.show()



**Q.128. Bar Graph**

**Ans:** Name=['Divya','Laxmi']

Age=[25,20]

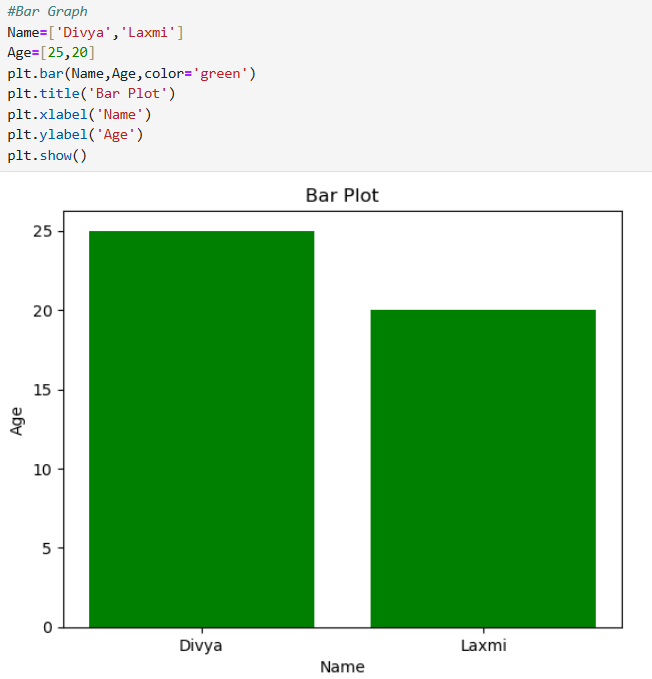
plt.bar(Name,Age,color='green')

plt.title('Bar Plot')

plt.xlabel('Name')

plt.ylabel('Age')

plt.show()

****

**Q.129. Histogram**

**Ans:** data=[2,3,4,7,8]

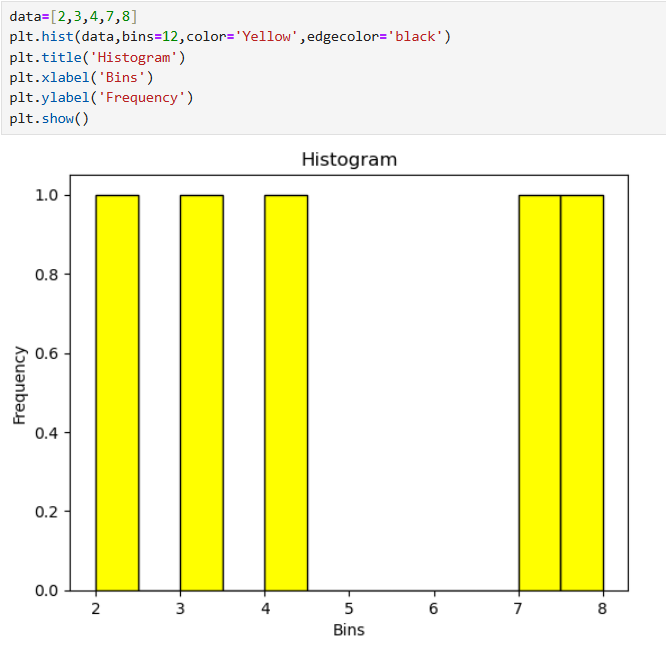
plt.hist(data,bins=12,color='Yellow',edgecolor='black')

plt.title('Histogram')

plt.xlabel('Bins')

plt.ylabel('Frequency')

plt.show()

****

**Q.130.**

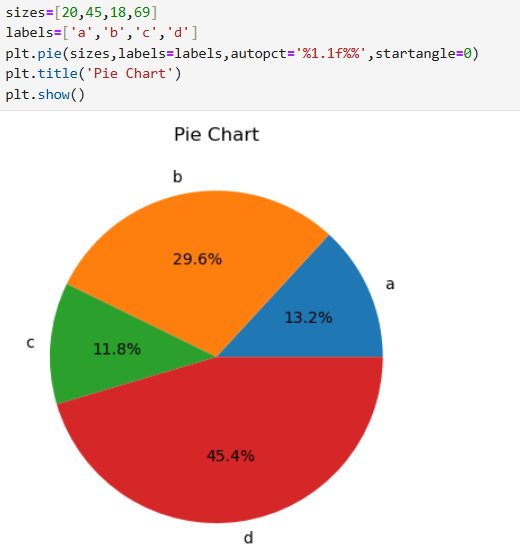
**Ans:** sizes=[20,45,18,69]

labels=['a','b','c','d']

plt.pie(sizes,labels=labels,autopct='%1.1f%%',startangle=0)

plt.title('Pie Chart')

plt.show()

****