**Pandas**

Pandas is a powerful and open-source Python library. The Pandas library is used for data manipulation and analysis. Pandas consist of data structures and functions to perform efficient operations on data.

Pandas is well-suited for working with tabular data, such as spreadsheets or SQL tables.

The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

Pandas allows us to analyze big data and make conclusions based on statistical theories.

Pandas can clean messy data sets, and make them readable and relevant.

Relevant data is very important in data science.

**Steps**

1. Open Anaconda Prompt
2. Type ‘jupyter notebook’
3. It will redirect to Home page of Jupyter notebook on browser
4. In Files tab open Desktop folder
5. Create new folder using New button on top right and name it
6. Open the folder and click on Upload button
7. Upload the dataset file (.csv , .xlsv) on which you have to perform tasks
8. Create a Python file by clicking on new button and choosing ‘Python 3’ file
9. On creating, file will open in new tab of the browser
10. Import pandas
11. Store dataset file in a variable

**Perform the following operations on dataset**

Before performing tasks on dataset we need to

1. Import pandas

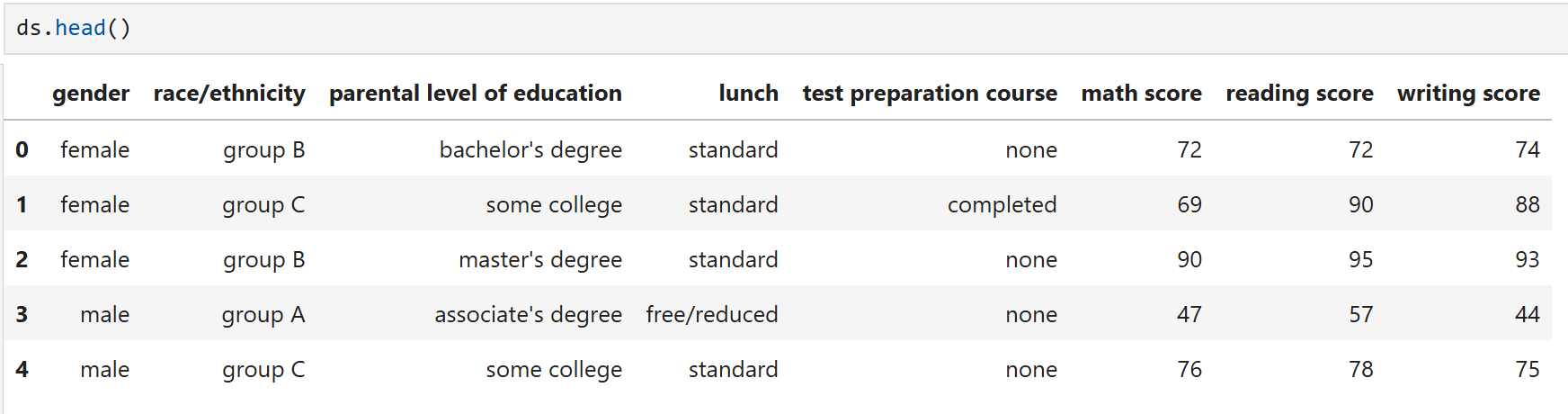
import pandas as pd

1. Store dataset file in a variable

ds = pd.read\_csv("StudentsPerformance.csv")

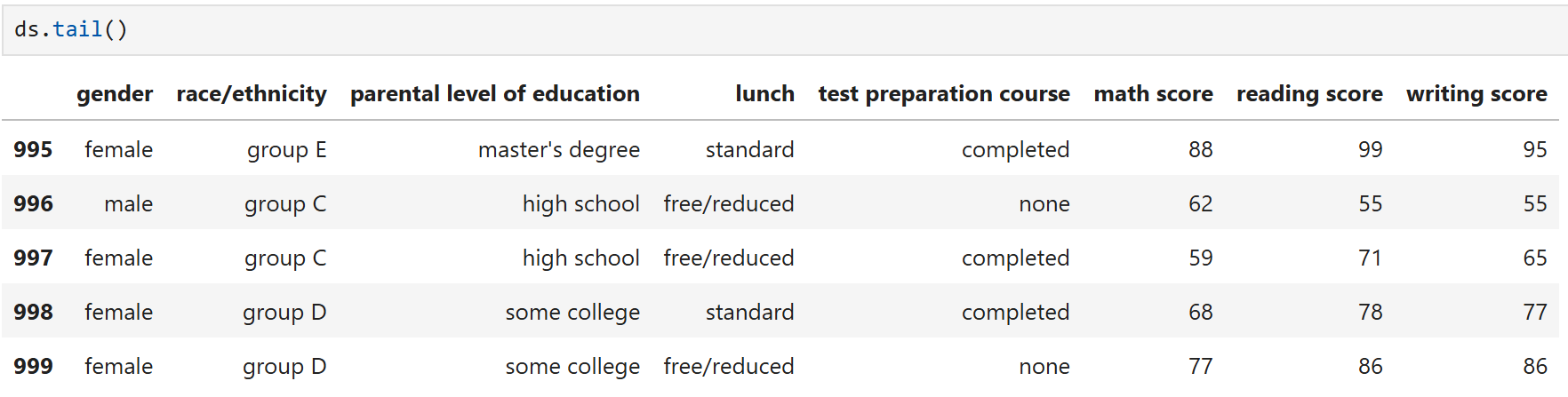
**Q.1. Display first 5 rows of dataset**

**=>** ds.head()



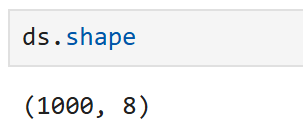
**Q.2. Display the last 5 rows of dataset**

**=>** ds.tail()



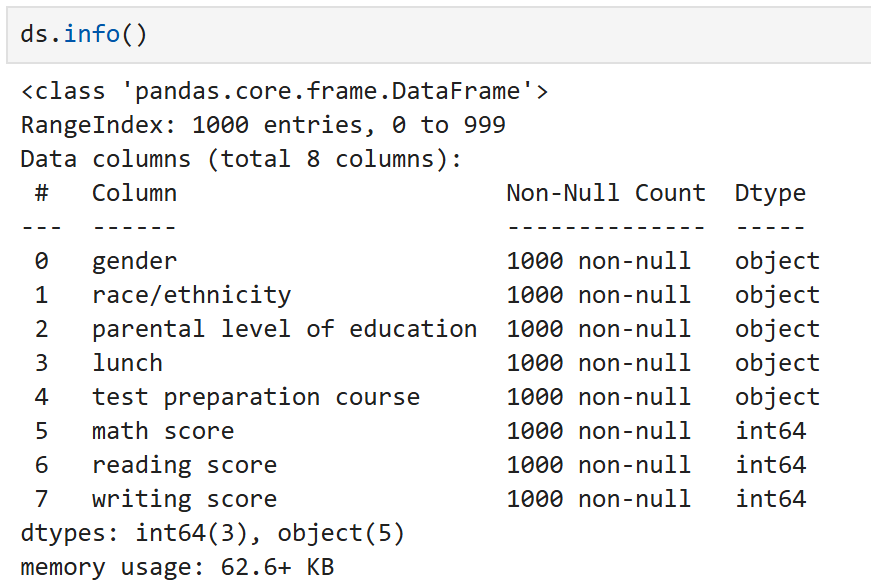
**Q.3. What is the shape of the DataSet (number of rows and columns)?**

**=>** ds.shape



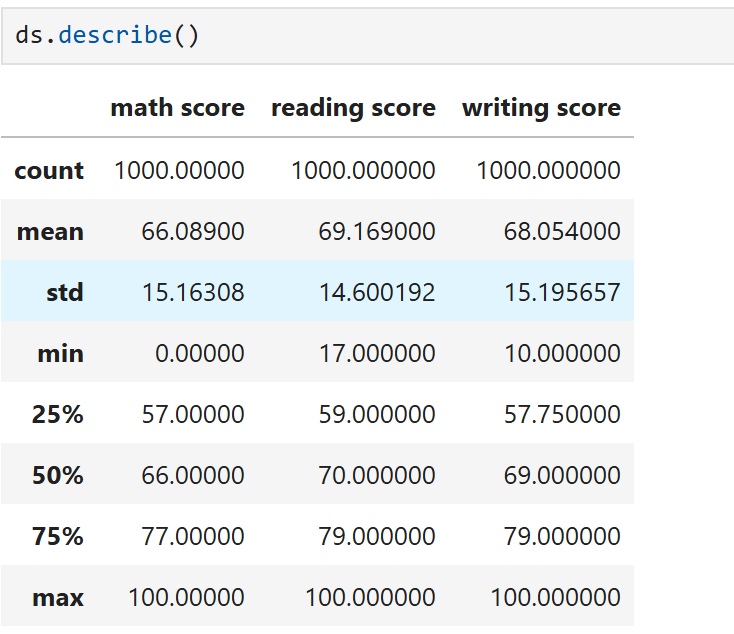
**Q.4. Display summary information about the DataSet, including data types and non-null counts.**

**=>** ds.info()



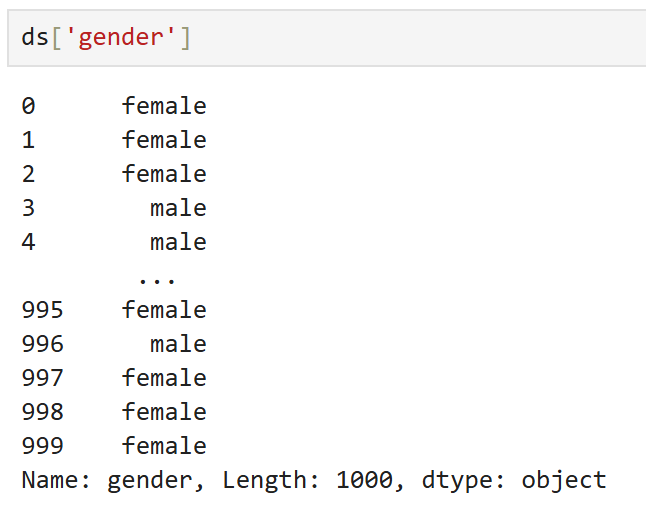
**Q.5. Get descriptive statistics for numerical columns in the DataSet.**

**=>** ds.describe()



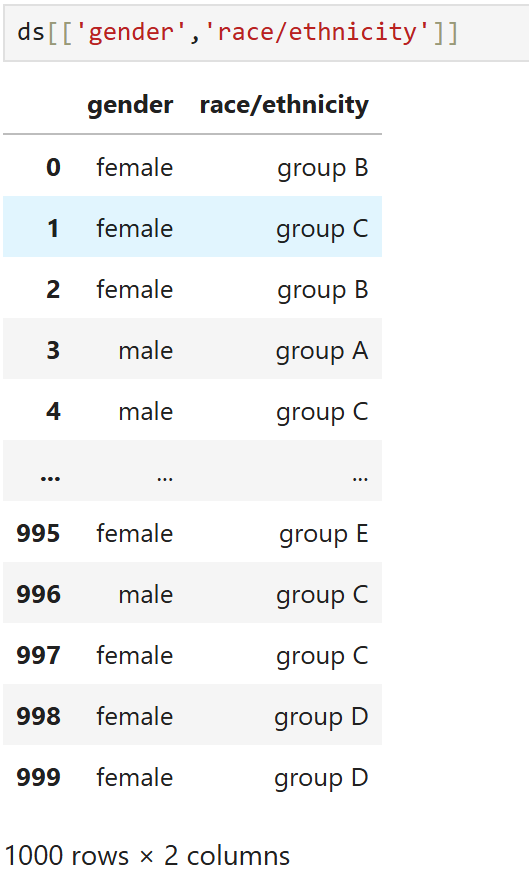
**Q.6. Select the "gender" column from the DataSet.**

**=>** ds['gender']



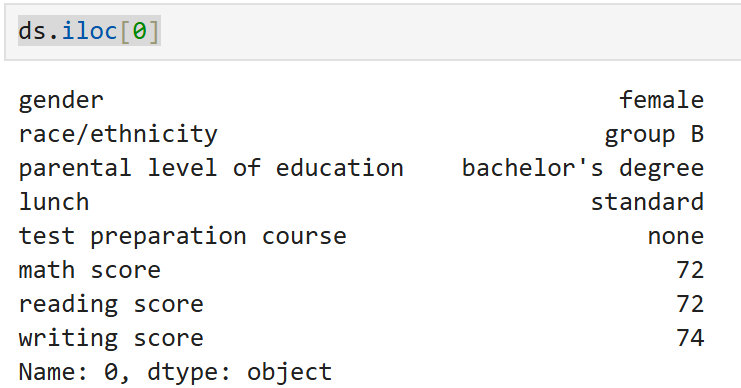
**Q.7. Select the "gender" and "race/ethnicity" columns from the DataSet.**

**=>** ds[['gender','race/ethnicity']]



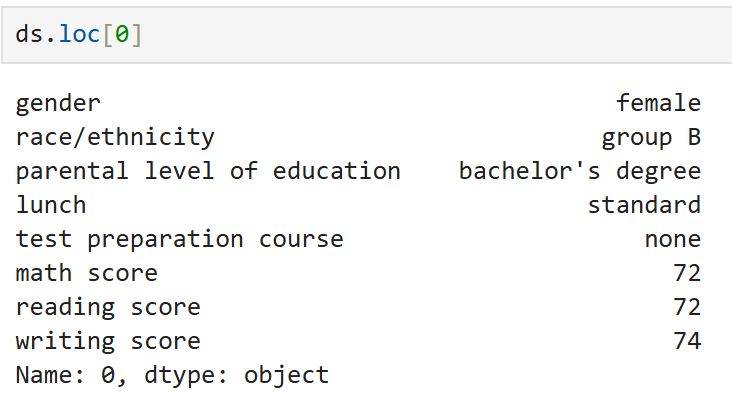
**Q.8. Select the first row from the DataSet by index.**

**=>** ds.iloc[0]



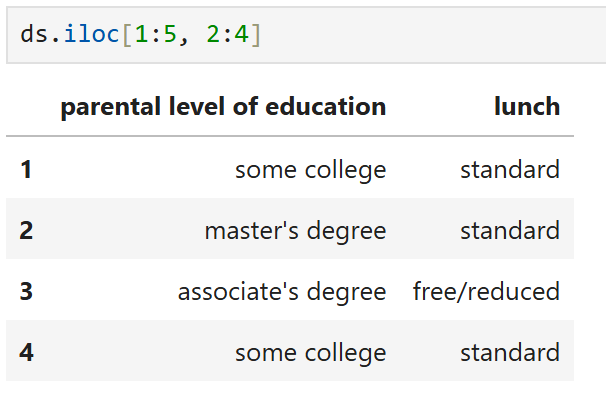
**Q.9. Select the row with label/index 0 from the DataSet.**

**=>** ds.loc[0]



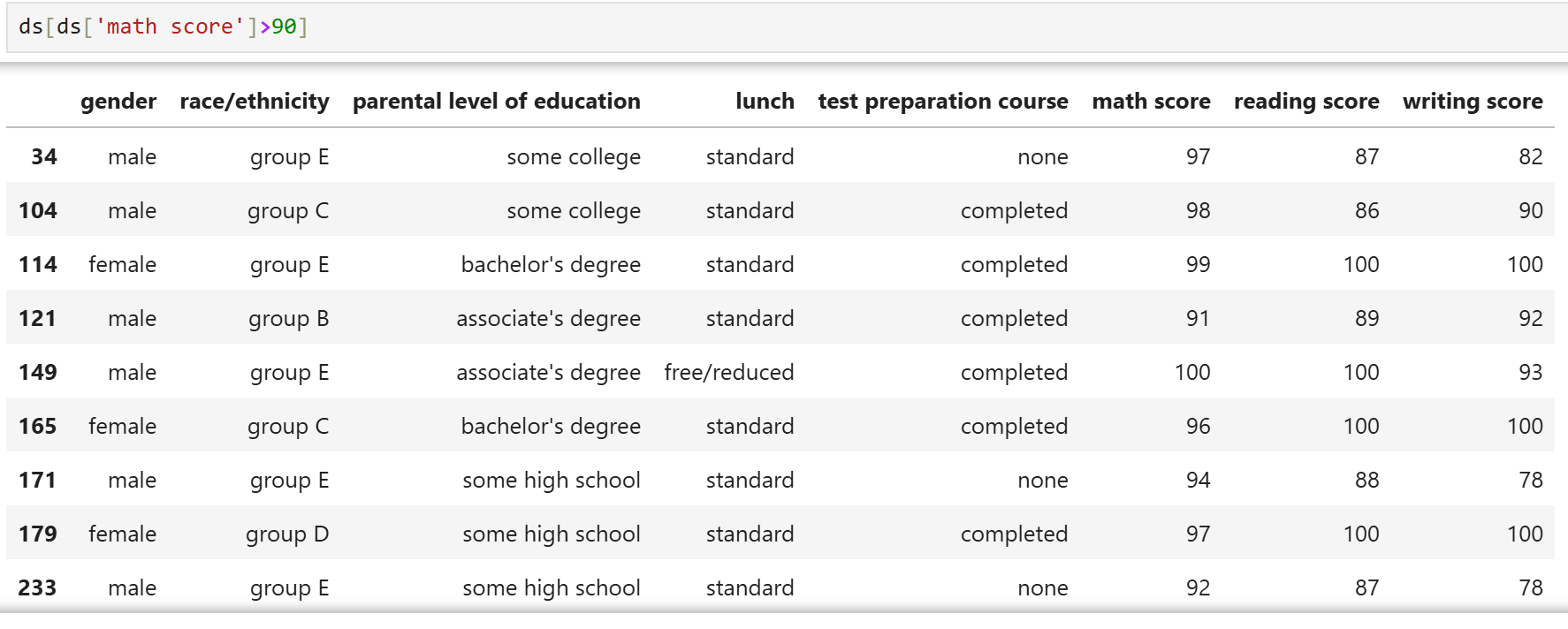
**Q.10. Select rows from 1 to 5 and columns from 2 to 3**

**=>** ds.iloc[1:5, 2:4]



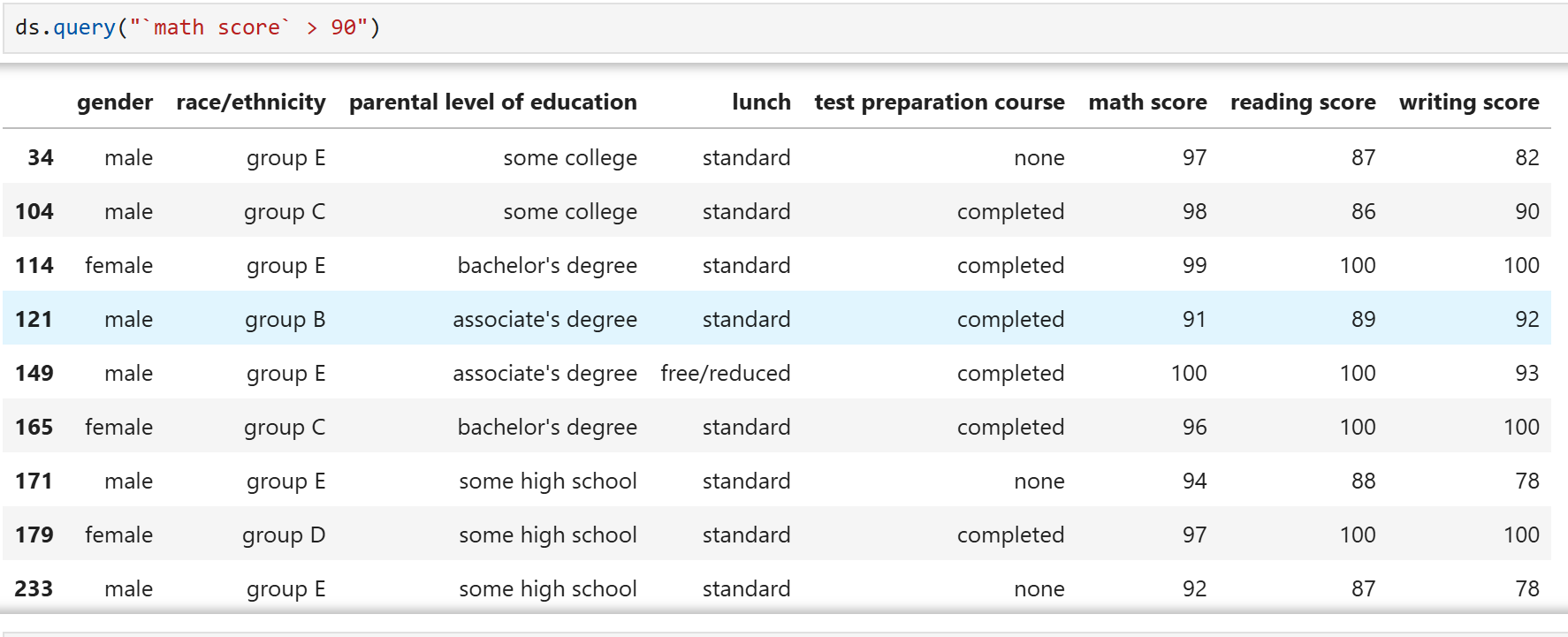
**Q.11. Filter the DataSet for rows where "math score" is greater than 90.**

**=>** ds[ds['math score']>90]



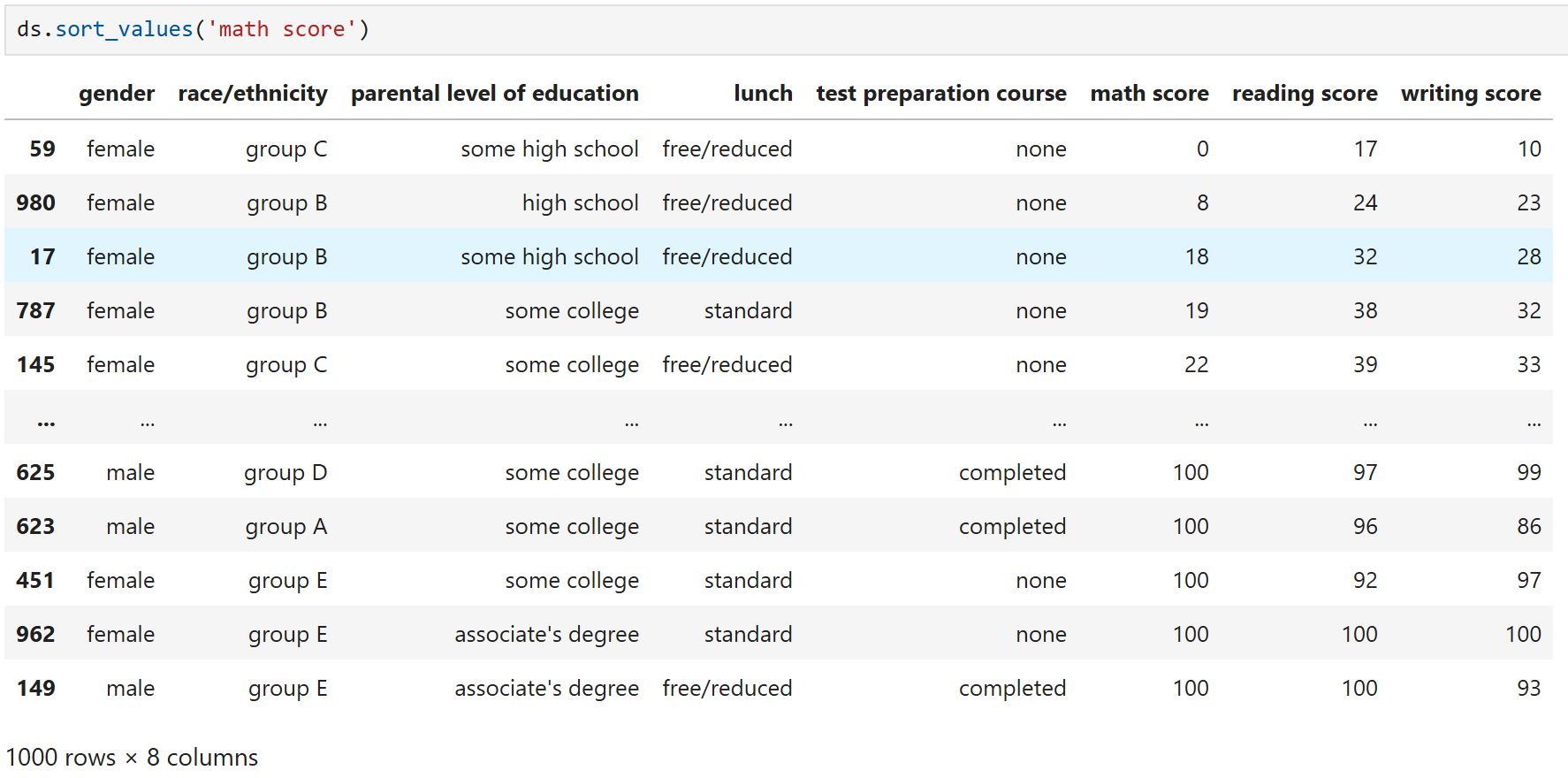
**Q.12. Use the query method to filter rows where "math score" is greater than 90.**

**=>** ds.query("`math score` > 90")

****

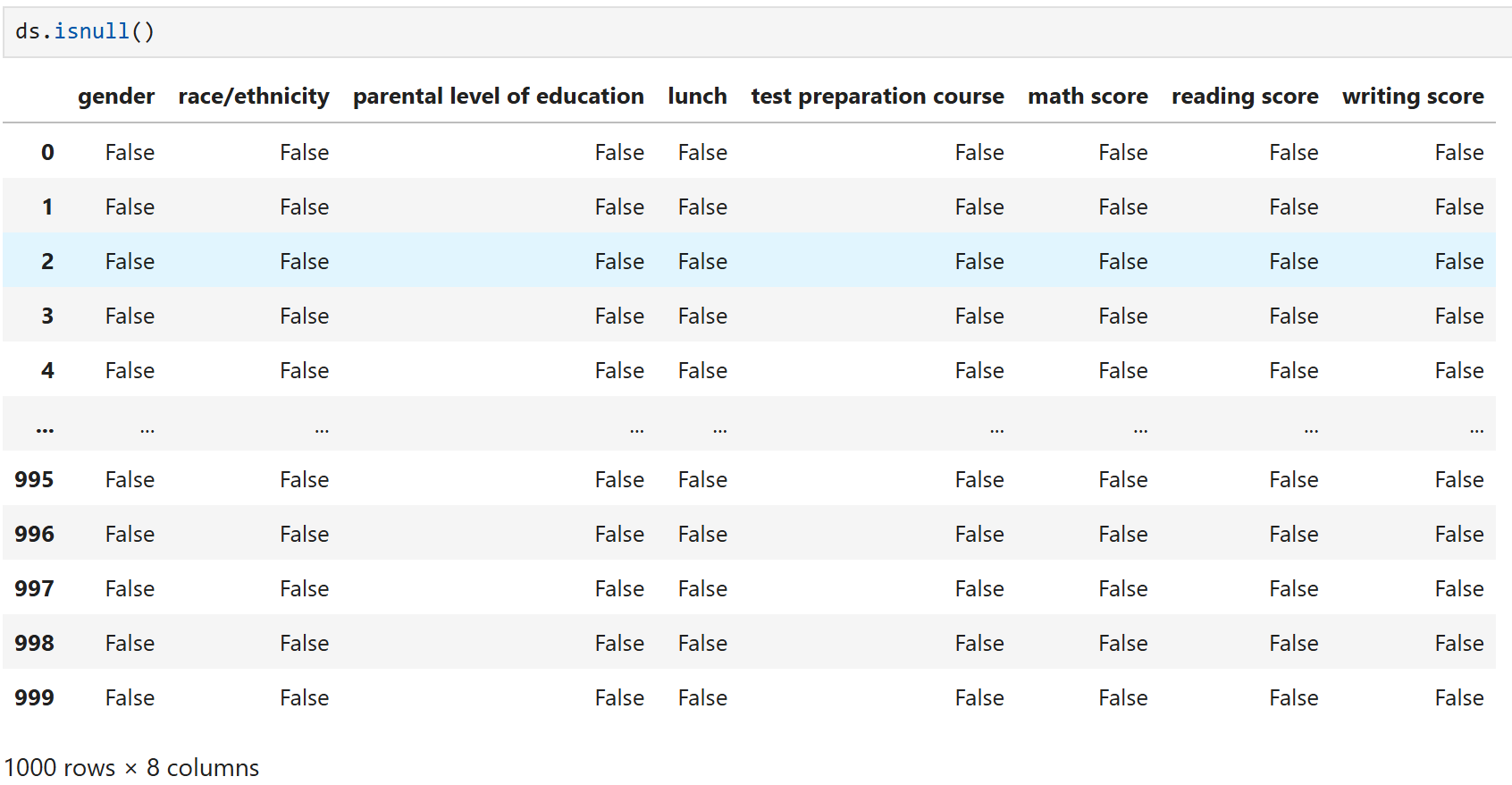
**Q.13. Sort the DataSet by the "math score" column in ascending order.**

**=>** ds.sort\_values('math score')

****

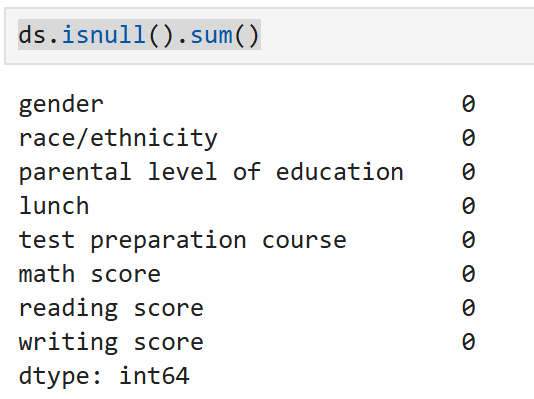
**Q.14. Check if there are any missing values in the DataSet.**

**=>** ds.isnull()

****

**Q.15. Count the number of missing values in each column.**

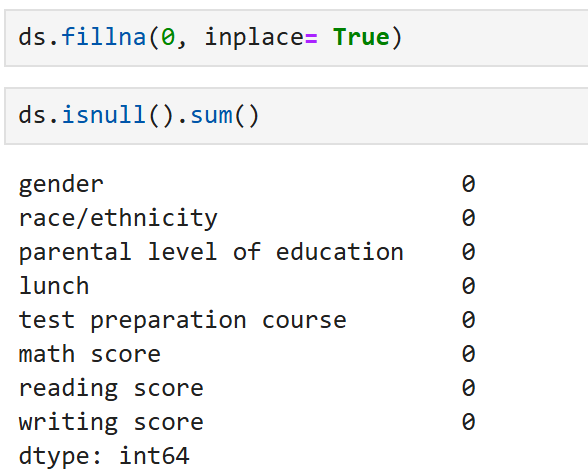
**=>** ds.isnull().sum()

****

**Q.16. Fill missing values with 0.**

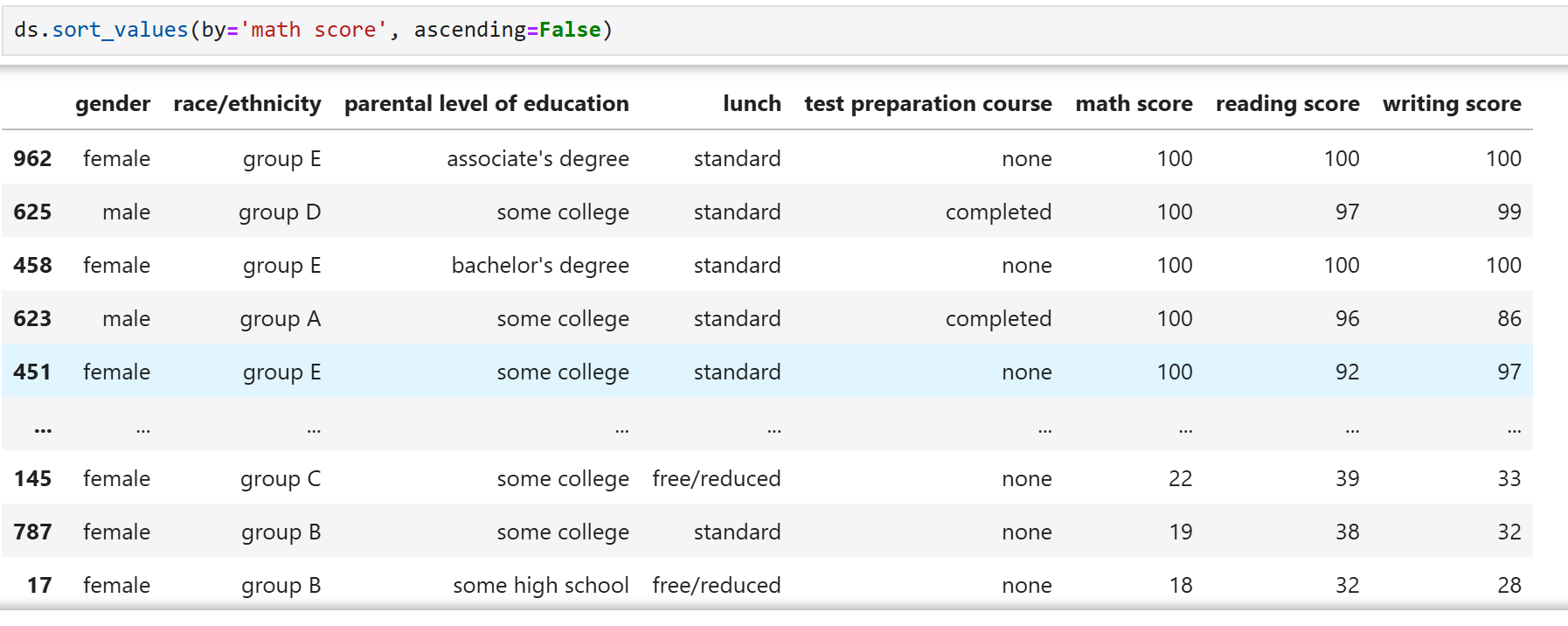
**=>** ds.fillna(0, inplace= True)

ds.isnull().sum()

****

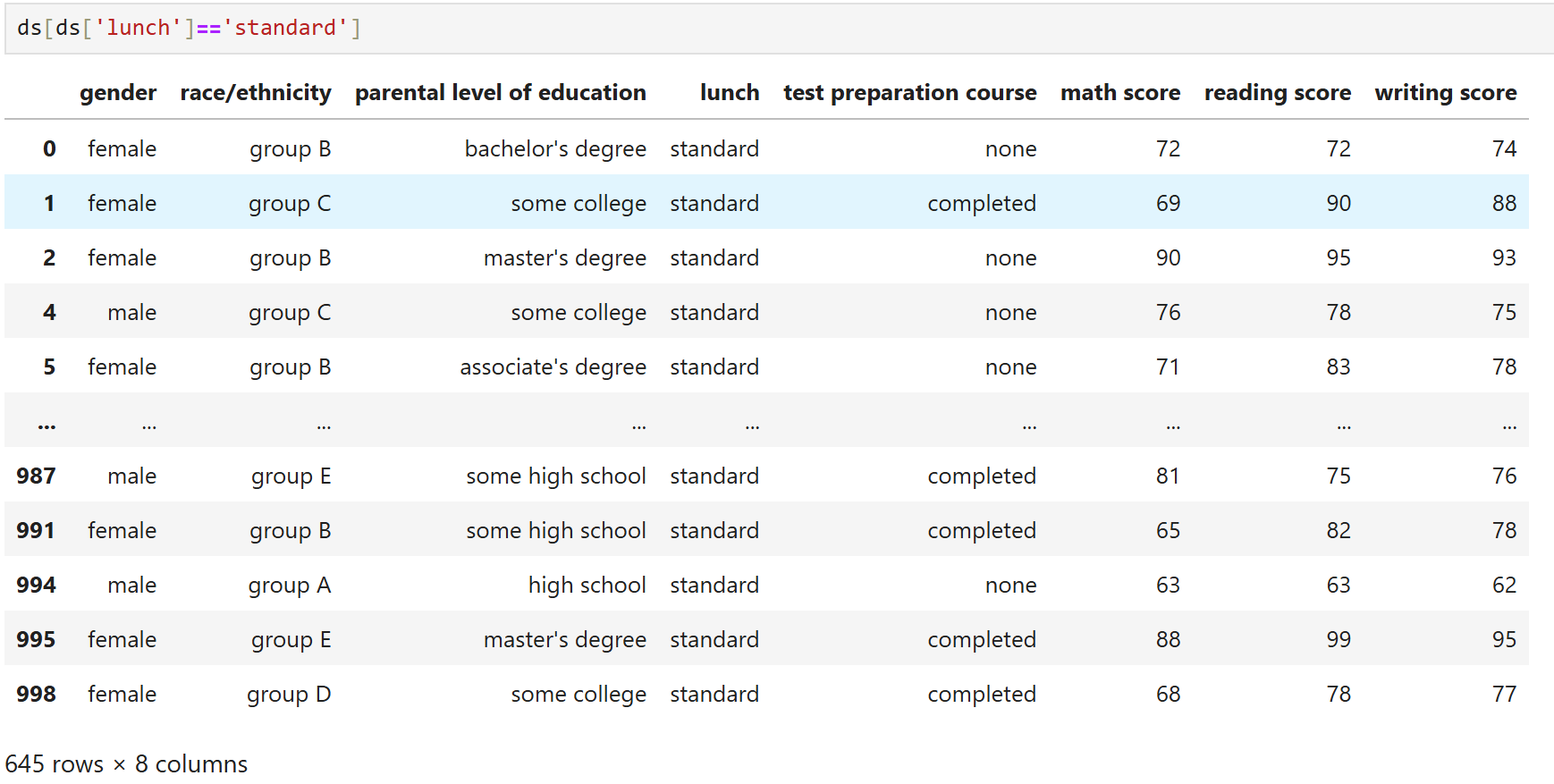
**Q.17. Sort the dataSet based on a specific column.**

**=>** ds.sort\_values(by='math score', ascending=False)

****

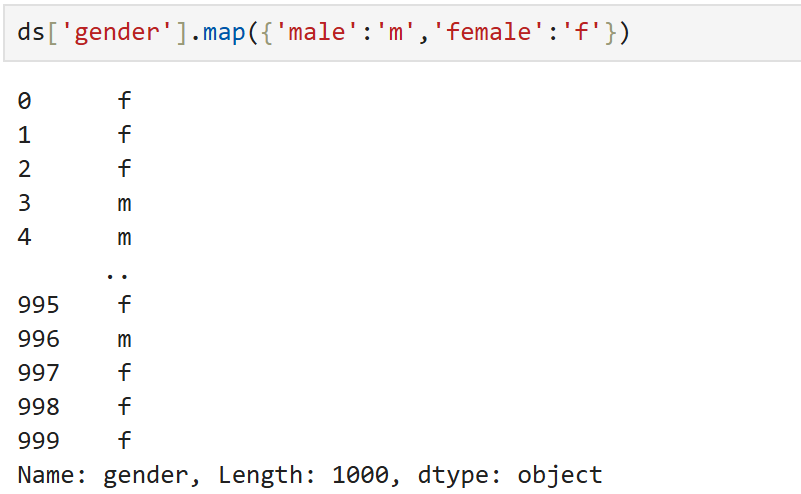
**Q.18. Filter rows where the lunch is of standard type.**

**=>** ds[ds['lunch']=='standard']

****

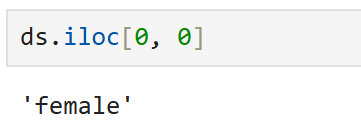
**Q.19. Convert 'male'/'female' responses to 'm'/'f'**

**=>** ds['gender'].map({'male':'m','female':'f'})

****

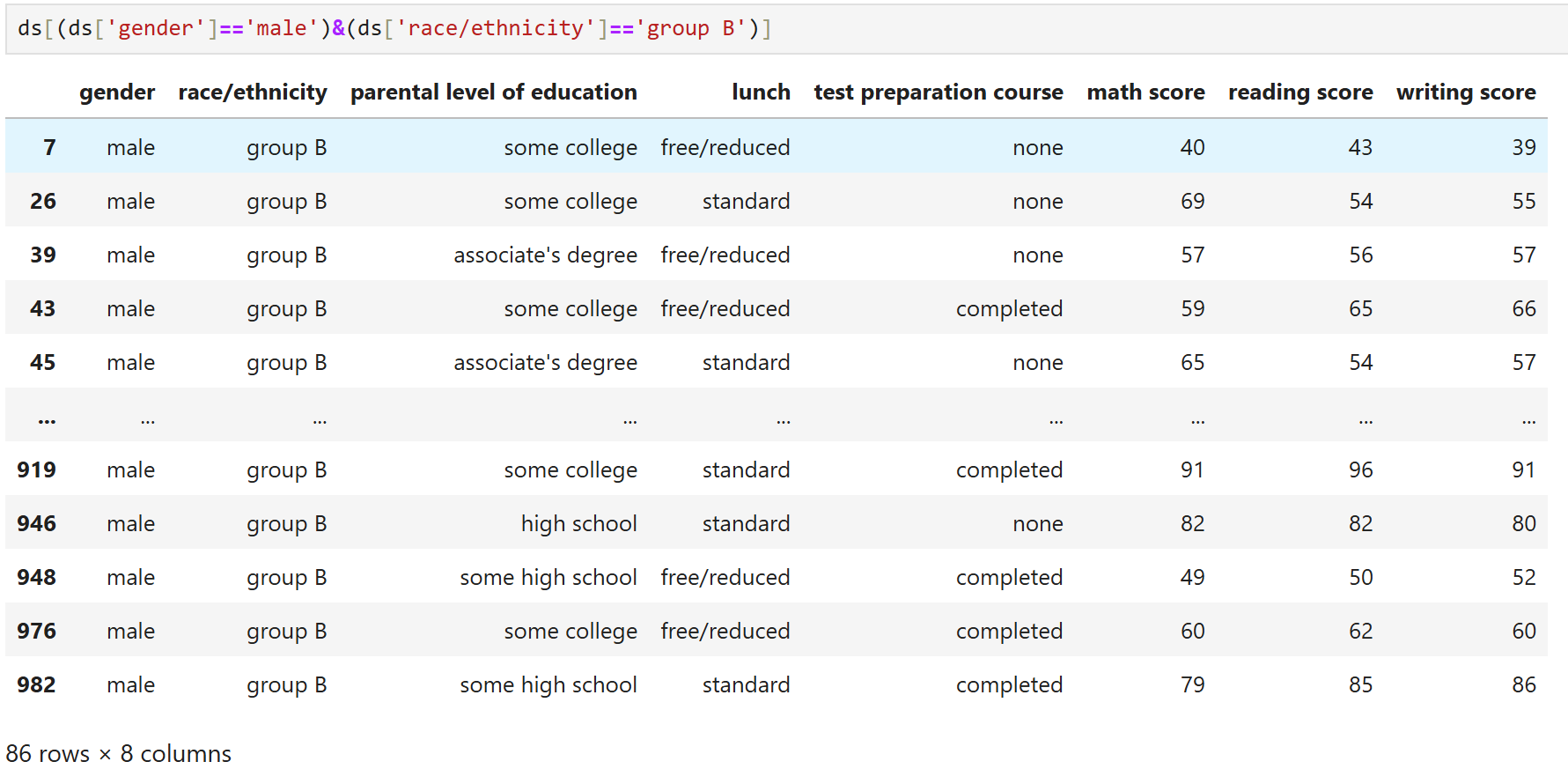
**Q.20. Selects the element in the first row and first column**

**=> ds.iloc[0, 0]**

****

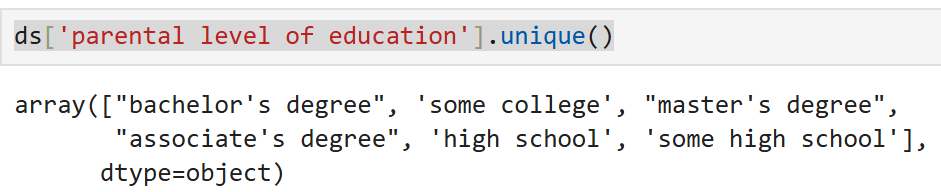
**Q.21. Find all rows where gender is "male" and race/ethnicity is “group B”**

**=>** ds[(ds['gender']=='male')&(ds['race/ethnicity']=='group B')]

****

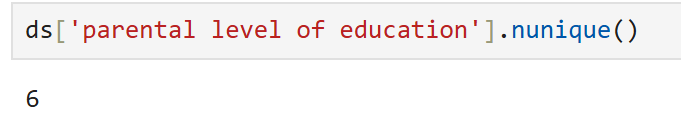
**Q.22. Find unique values in the 'parental level of education' column.**

**=>** ds['parental level of education'].unique()

****

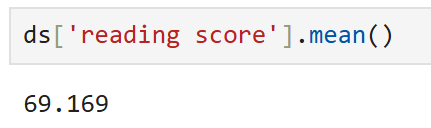
**Q.23. Count unique values in the 'parental level of education' column.**

**=>** ds['parental level of education'].nunique()

****

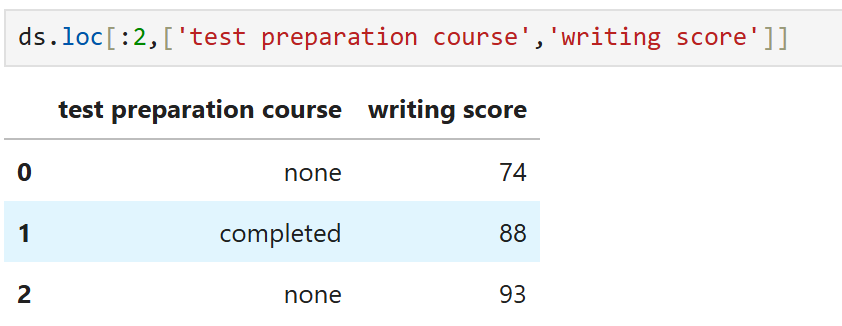
**Q.24. Calculate the average reading score.**

**=>** ds['reading score'].mean()

****

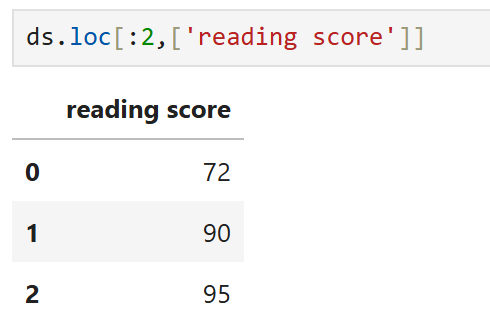
**Q.25. Select the first 3 rows of the 'test preparation course' and 'writing score' columns.**

**=>** ds.loc[:2,['test preparation course','writing score']]

****

**Q.26. Select the first 3 rows of the 'reading score'.**

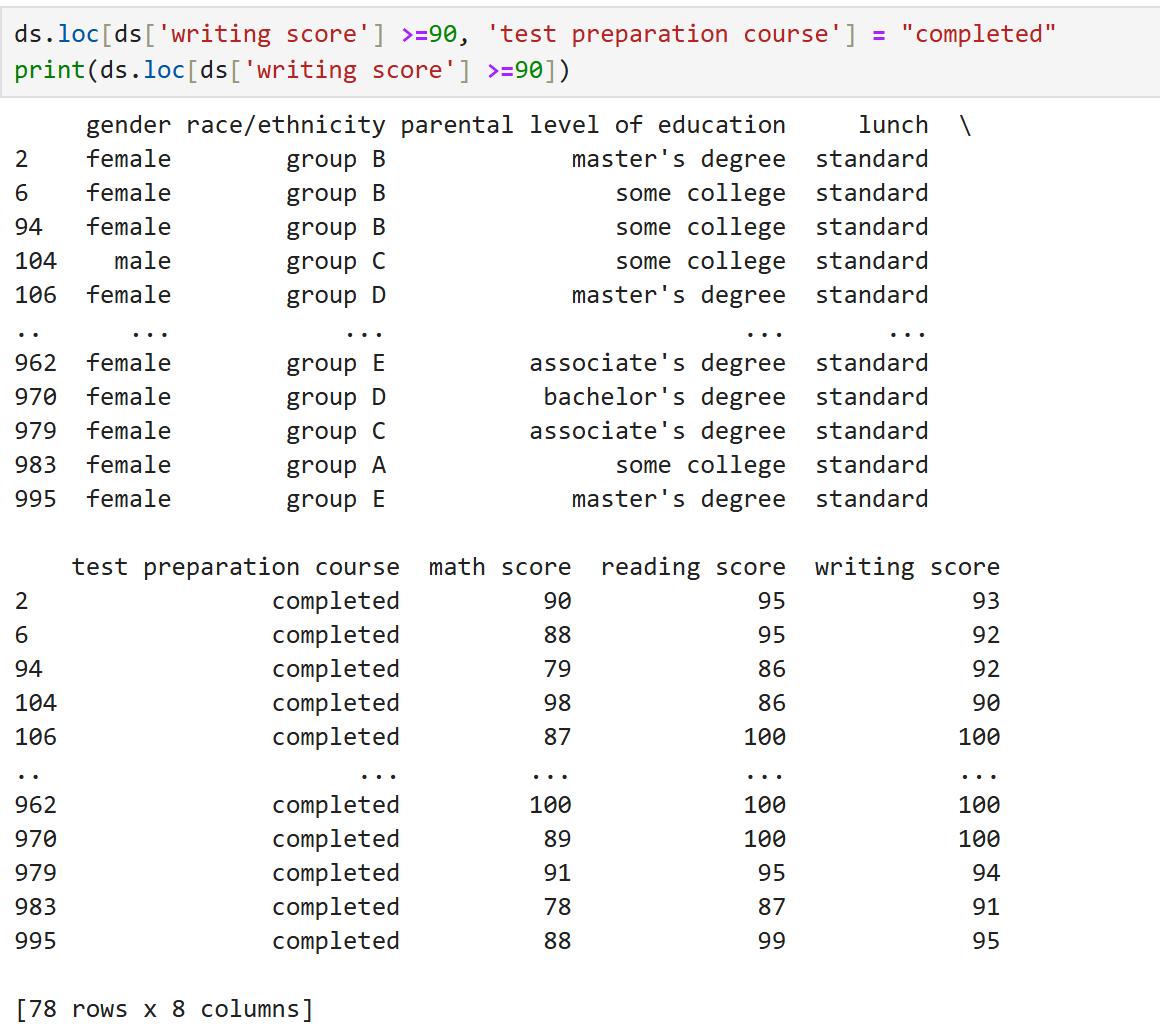
**=>** ds.loc[:2,['reading score']]

****

**Q.27. Change ‘test preparation course’ for the student with ‘writing score’ >=90 to "completed”.**

**=>** ds.loc[ds['writing score'] >=90, 'test preparation course'] = "completed"

print(ds.loc[ds['writing score'] >=90])

****

**Q.28. Check if 'race/ethnicity' == ‘group F’ exists**

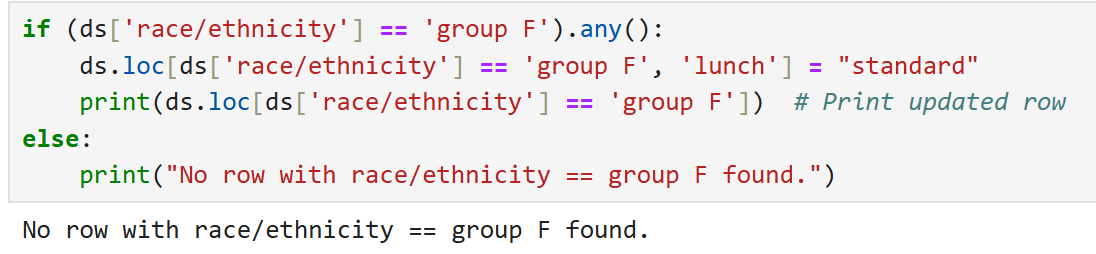
**=>** if (ds['race/ethnicity'] == 'group F').any():

ds.loc[ds['race/ethnicity'] == 'group F', 'lunch'] = "standard"

print(ds.loc[ds['race/ethnicity'] == 'group F'])

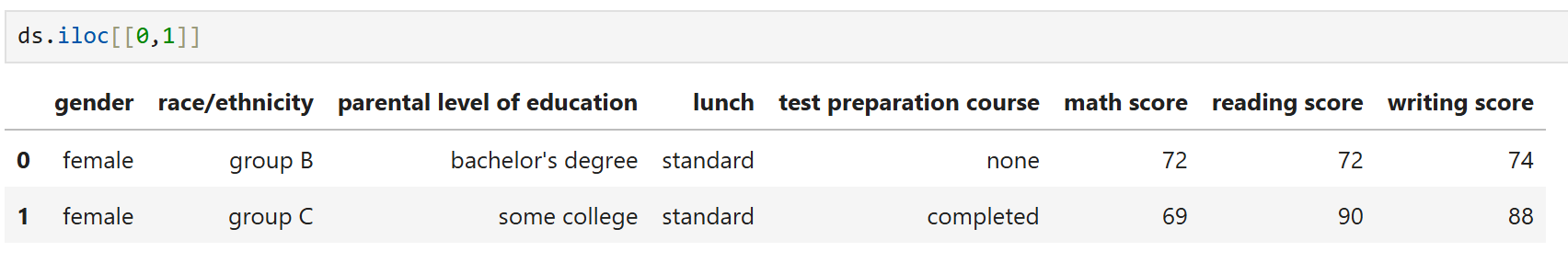
else:

print("No row with race/ethnicity == group F found.")

****

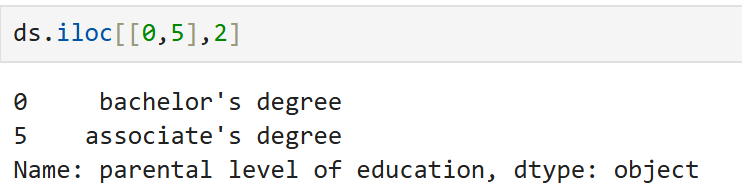
**Q.29.**

**=>** ds.iloc[[0,1]]

****

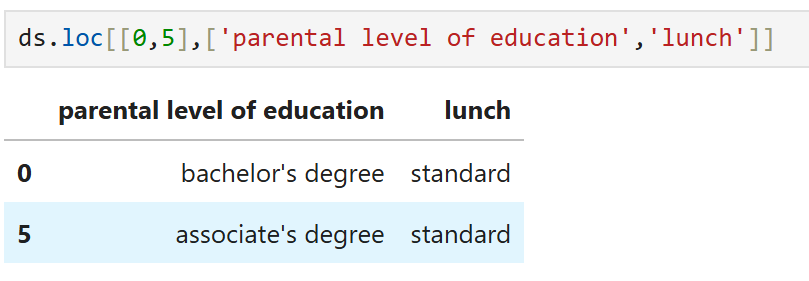
**Q.30.**

**=>** ds.iloc[[0,5],2]

****

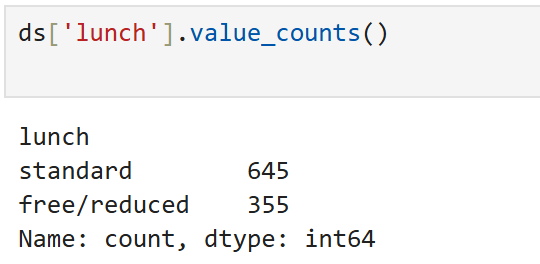
**Q.31.**

**=>** ds.loc[[0,5],['parental level of education','lunch']]

****

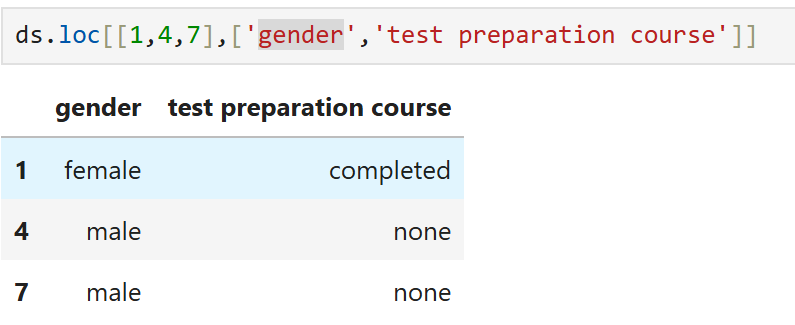
**Q.32.**

**=>** ds['lunch'].value\_counts()

****

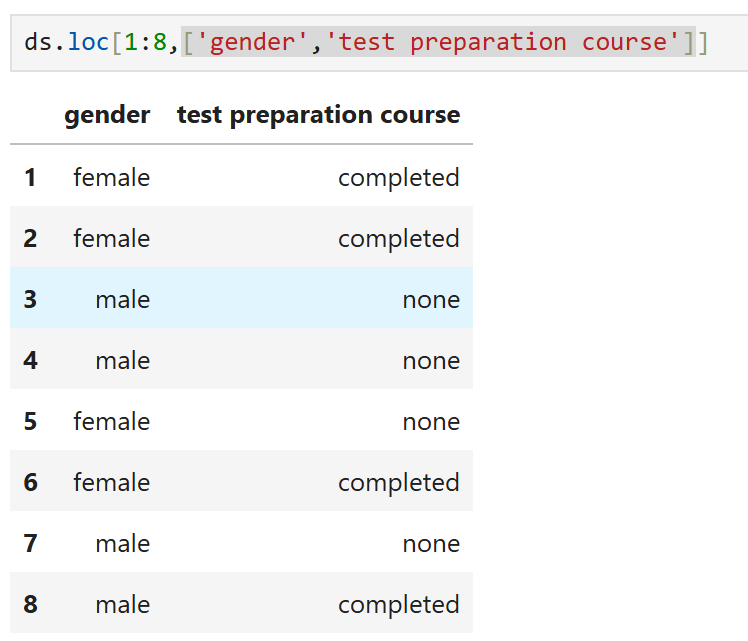
**Q.33.**

**=>** ds.loc[[1,4,7],['gender','test preparation course']]

****

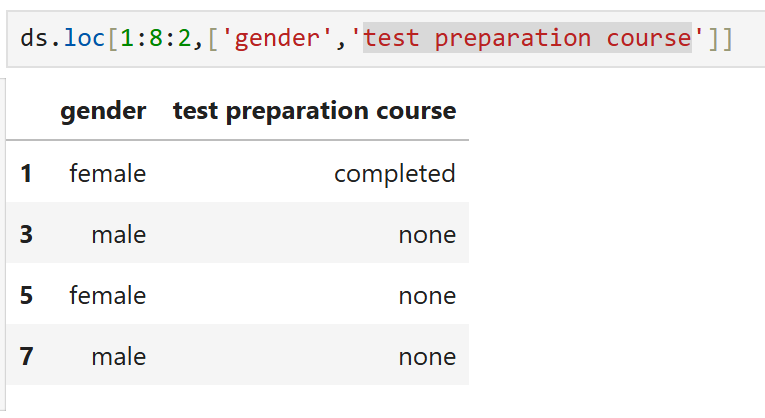
**Q.34.**

**=>** ds.loc[1:8,['gender','test preparation course']]

****

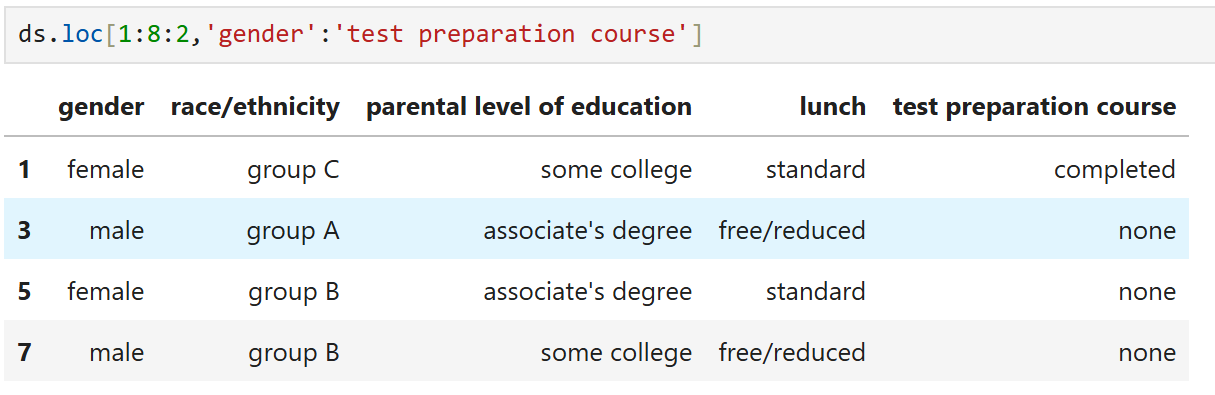
**Q.35.**

**=>** ds.loc[1:8:2,['gender','test preparation course']]

****

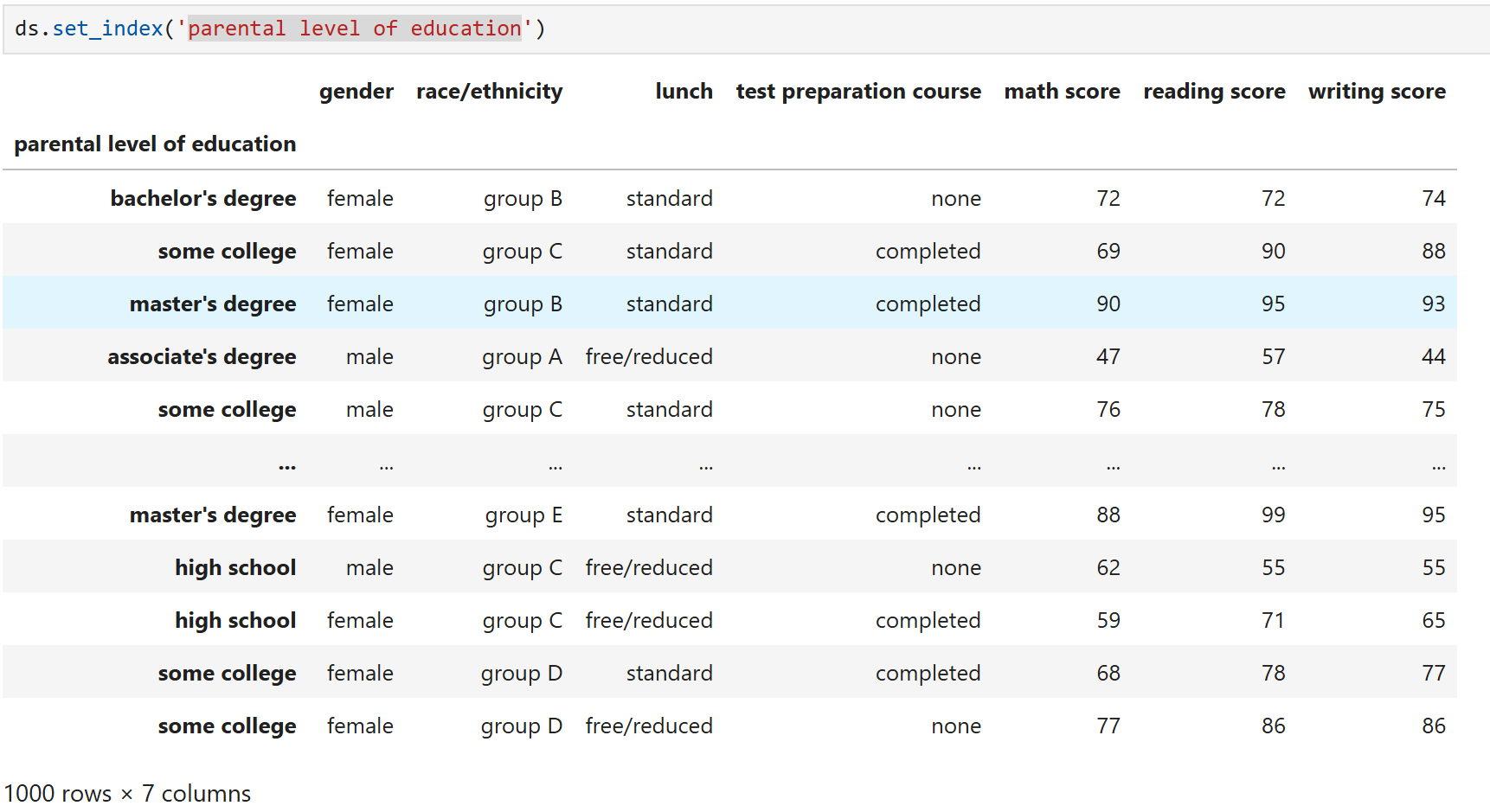
**Q.36.**

**=>** ds.loc[1:8:2,'gender':'test preparation course']

****

**Q.37.**

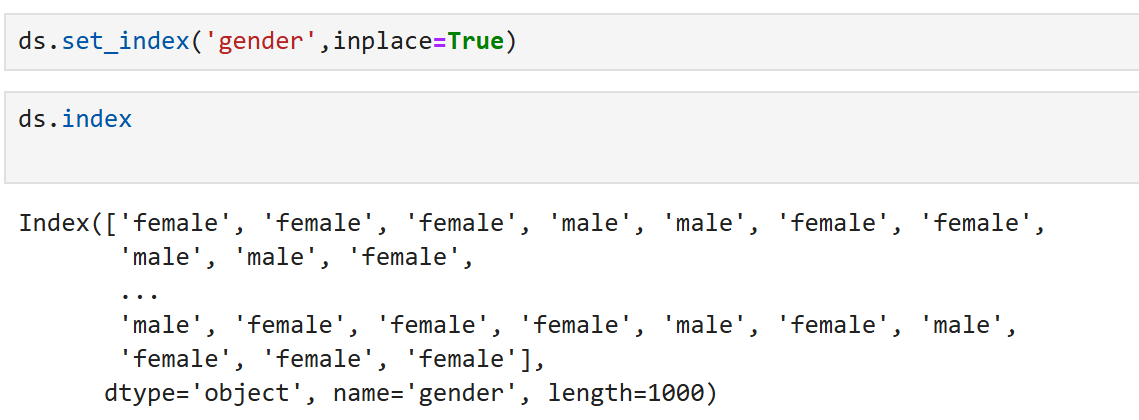
**=>** ds.set\_index('parental level of education')

****

**Q.38.**

**=>** ds.set\_index('gender',inplace=True)

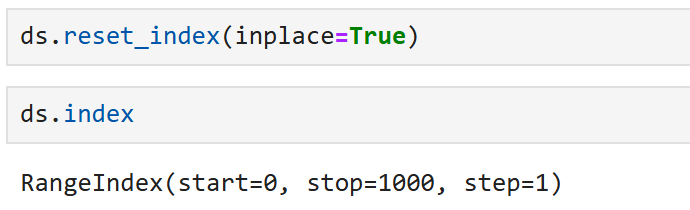
ds.index

****

**Q.39.**

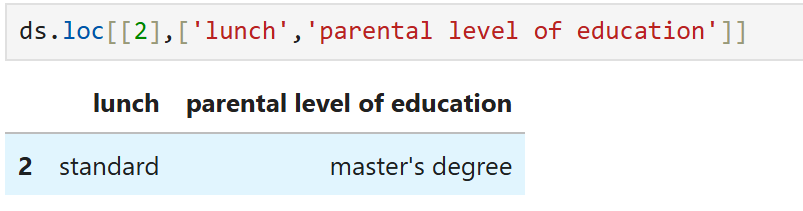
**=>** ds.reset\_index(inplace=True)

ds.index

****

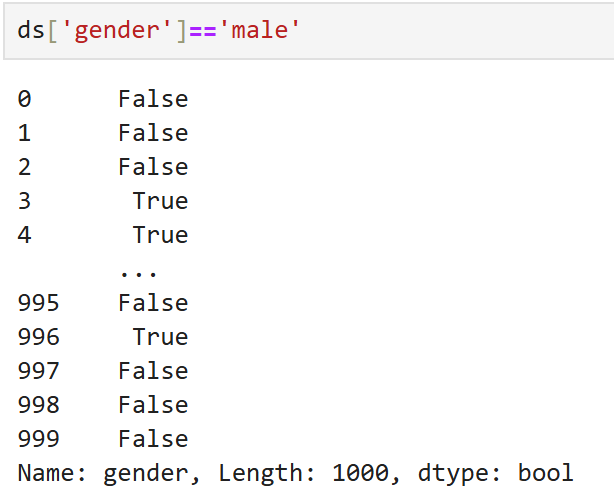
**Q.40.**

**=>** ds.loc[[2],['lunch','parental level of education']]

****

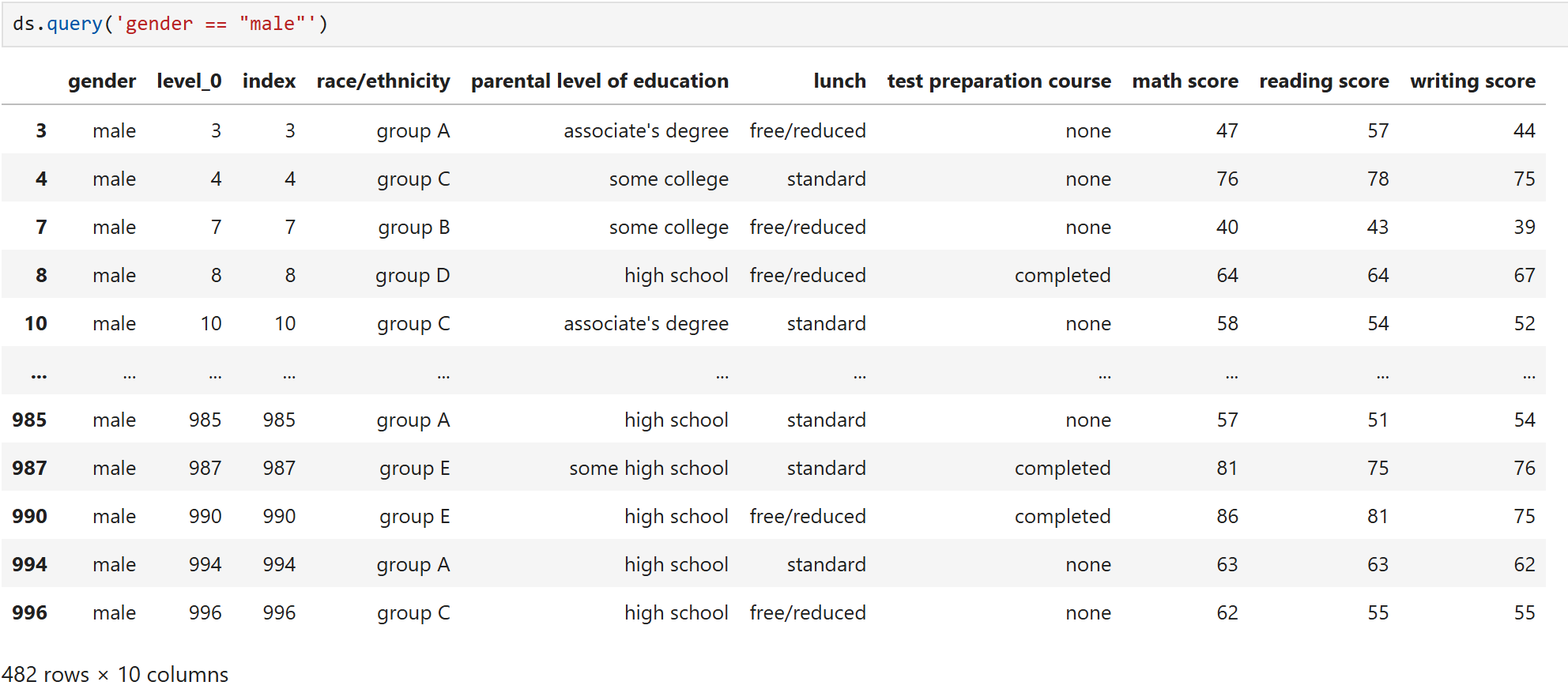
**Q.41.**

**=>** ds['gender']=='male'

****

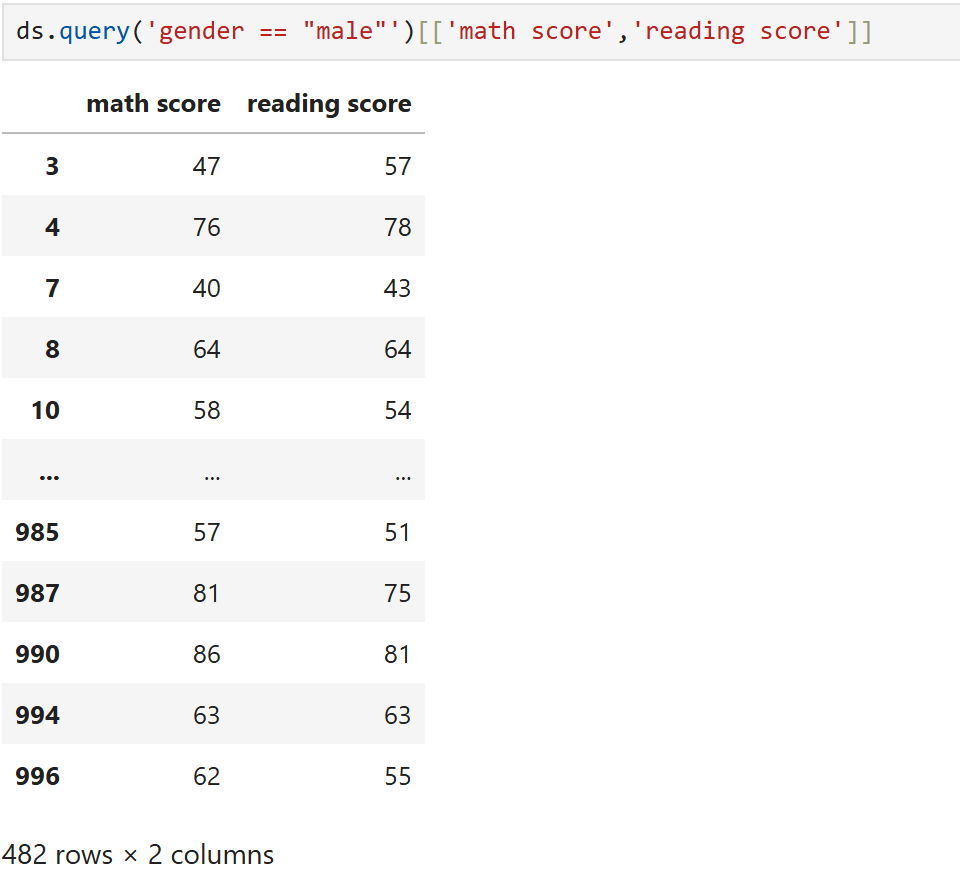
**Q.42.**

**=>** ds.query('gender == "male"')

****

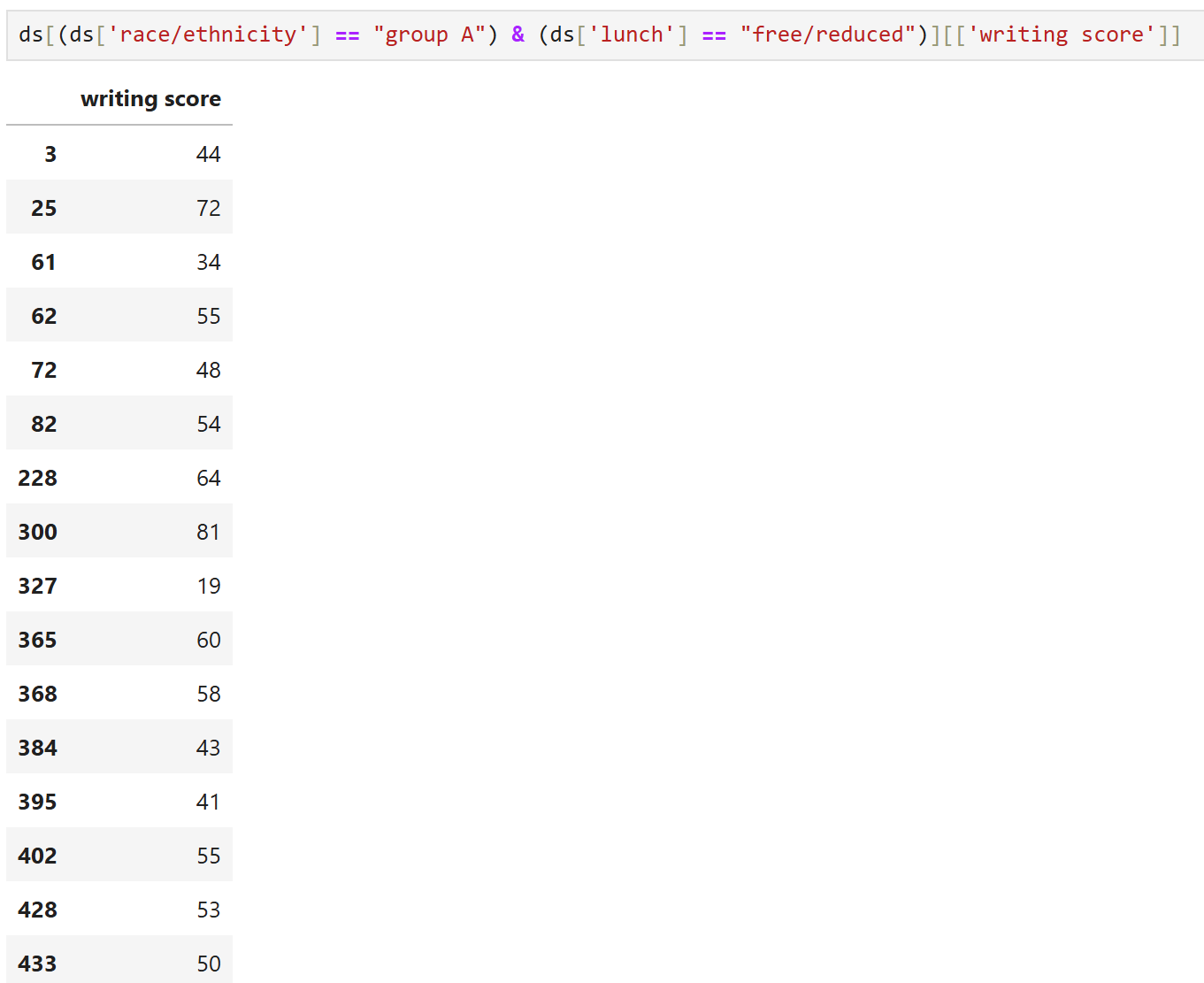
**Q.43.**

**=>** ds.query('gender == "male"')[['math score','reading score']]

****

**Q.44.**

**=>** ds[(ds['race/ethnicity'] == "group A") & (ds['lunch'] == "free/reduced")][['writing score']]

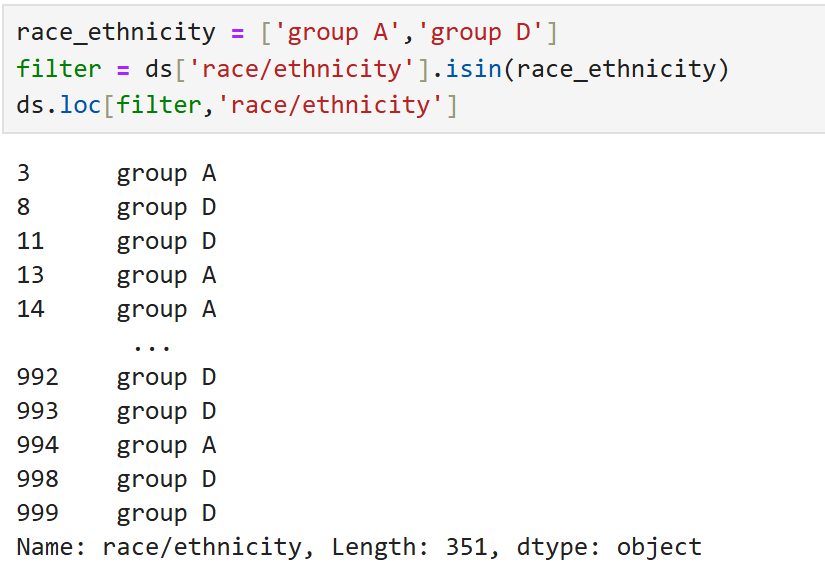
****

**Q.45.**

**=>** race\_ethnicity = ['group A','group D']

filter = ds['race/ethnicity'].isin(race\_ethnicity)

ds.loc[filter,'race/ethnicity']

****

**Q.46.**

**=>** m= ds['parental level of education'].str.contains("associate's degree",na=False)

ds.loc[m,'parental level of education']

