**Q1. How do you load a CSV file into a Pandas DataFrame?**

**Ans –**

We can load csv file into panda by using below command.

df = pd.read\_csv('https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv')

df

**Q2. How do you check the data type of a column in a Pandas DataFrame?**

**Ans** – To check data type of a column in Panda using **df.dtypes**

**Q3. How do you select rows from a Pandas DataFrame based on a condition?**

**Ans –**

# importing pandas

import pandas as pd

record = {

'Name': ['Ankit', 'Amit', 'Aishwarya', 'Priyanka', 'Priya', 'Shaurya' ],

'Age': [21, 19, 20, 18, 17, 21],

'Stream': ['Math', 'Commerce', 'Science', 'Math', 'Math', 'Science'],

'Percentage': [88, 92, 95, 70, 65, 78] }

# create a dataframe

dataframe = pd.DataFrame(record, columns = ['Name', 'Age', 'Stream', 'Percentage'])

print("Given Dataframe :\n", dataframe)

# selecting rows based on condition

rslt\_df = dataframe[dataframe['Percentage'] > 80]

print('\nResult dataframe :\n', rslt\_df)

**Q4. How do you rename columns in a Pandas DataFrame?**

**Ans –**

# Import pandas package

import pandas as pd

# Define a dictionary containing ICC rankings

rankings = {'test': ['India', 'South Africa', 'England',

                            'New Zealand', 'Australia'],

            'odi': ['England', 'India', 'New Zealand',

                            'South Africa', 'Pakistan'],

            't20': ['Pakistan', 'India', 'Australia',

                            'England', 'New Zealand']}

# Convert the dictionary into DataFrame

rankings\_pd = pd.DataFrame(rankings)

# Before renaming the columns

rankings\_pd

rankings\_pd.rename(columns = {'test':'TEST'}, inplace = True)

# After renaming the columns

rankings\_pd

**Q5. How do you drop columns in a Pandas DataFrame?**

**Ans –**

We can drop a column using drop() function. Let’s understand with above example. We want to drop “TEST” column.

**df.drop(‘TEST’) 🡪** This will throw axis error because by default axis value is 0 and this for row. For columns axis value is 1. So we need to mention axis = 1.

**df.drop(‘TEST’, axis = 1) 🡪** It will delete the column temporary. It means that when we again run **rankings\_pd.** It will again show output as TEST column.

**df.drop(‘TEST’, axis = 1, inplace = 1) 🡪** It will delete the TEST column permanently.

**Q6. How do you find the unique values in a column of a Pandas DataFrame?**

**Ans** – Using unique() function, we can get the unique value of a column.

df[‘Pclass’].unique()

**Q7. How do you find the number of missing values in each column of a Pandas DataFrame?**

**Ans** – We can find the missing value using isnull() and sum of all null value in a column using isnull().sum().

df.isnull()

df.isnull().sum()

**Q8. How do you fill missing values in a Pandas DataFrame with a specific value?**

**Ans –**

fillna() method is used to fill the missing value with specified value.

**Q9. How do you concatenate two Pandas DataFrames?**

**Ans –**

# importing pandas as pd

import pandas as pd

# define value in platce of data1 and import it as DataFrame

df1 = pd.DataFrame(data1)

# define value in platce of data2 and import it as DataFrame

df2 = pd.DataFrame(data2)

# Concatenate using concat() function and pass the value of df1 and df2 as argument

res = pd.concat([df1 , df2])

# display result

res

If we want to merge row then default axis value is 0 and if we want to merge column then we need to mention axis value is 1.

**Q10. How do you merge two Pandas DataFrames on a specific column?**

**Ans** –

We can merge the values using merge() function.

# importing modules

import pandas as pd

# creating a dataframe

df1 = pd.DataFrame({'Name':['Raju', 'Rani', 'Geeta', 'Sita', 'Sohit'],

'Marks':[80, 90, 75, 88, 59]})

# creating another dataframe with different data

df2 = pd.DataFrame({'Name':['Raju', 'Divya', 'Geeta', 'Sita'],

'Grade':['A', 'A', 'B', 'A'],

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

'Gender':['Male', 'Female', 'Female', 'Female']})

# display df1

display(df1)

# display df2

display(df2)

# applying merge (merging Grade and Rank from df2 to df1)

df1.merge(df2[['Name', 'Grade', 'Rank']])

**Q11. How do you group data in a Pandas DataFrame by a specific column and apply an aggregation function?**

**Ans –** Using groupby() and agg()

Import pandas as pd

data1 = {"Name": ["Virat", "Virat", "Virat", "Virat", "Virat", "Virat", "Steve", "Steve", "Steve", "Steve", "Steve", "Steve"],

"Run": [70, 71, 62, 63, 64, 65, 56, 57, 58, 59, 60, 61],

"Team": ["India", "India", "India", "India", "India", "India", "Australia", "Australia", "Australia", "Australia", "Australia", "Australia"]}

df = pd.DataFrame(data1)

df

grouped\_data1 = df.groupby('Team').agg({'Age' : ['mean' , 'max' , 'min']})

grouped\_data1

**Q12. How do you pivot a Pandas DataFrame?**

**Ans –**

Using Pivot function.

df.pivot(argument)

**Q13. How do you change the data type of a column in a Pandas DataFrame?**

Ans –

#importing panda as pd

import pandas as pd

# Input data as dataframe

df = pd.DataFrame({

'A': [1, 2, 3, 4, 5],

'B': ['a', 'b', 'c', 'd', 'e'],

'C': [1.1, 1.0, 1.3, 2, 5]})

# Display data type for the A, B and C column

print(df.dtypes)

# Output

A int64

B object

C float64

dtype: object

# Changing data types to object

df = df.astype(str)

# Output

print(df.dtypes)

A object

B object

C object

dtype: object

**Q14. How do you sort a Pandas DataFrame by a specific column?**

Ans – We can sort data by using sort\_values() function.

# Importing Pandas as pd

import pandas as pd

# Creating Data

age\_list = [['Afghanistan', 1952, 8425333, 'Asia'],

['Australia', 1957, 9712569, 'Oceania'],

['Brazil', 1962, 76039390, 'Americas'],

['China', 1957, 637408000, 'Asia'],

['France', 1957, 44310863, 'Europe'],

['India', 1952, 3.72e+08, 'Asia'],

['United States', 1957, 171984000, 'Americas']]

# Importing that data as data frame and adding columns

df = pd.DataFrame(age\_list, columns=['Country' , 'Year' , 'Population' , 'Continent'])

# Print those data as data frame

df

# Sorting data by Country. By default it is in ascending order

df.sort\_values(by=['Country'])

# Output - Sorted by country in ascending order

Country Year Population Continent

0 Afghanistan 1952 8425333.0 Asia

1 Australia 1957 9712569.0 Oceania

2 Brazil 1962 76039390.0 Americas

3 China 1957 637408000.0 Asia

4 France 1957 44310863.0 Europe

5 India 1952 372000000.0 Asia

6 United States 1957 171984000.0 Americas

# Sorting data by Country by descending order

df.sort\_values(by=['Country'] , ascending = False)

# Output - Sorted by country in descending order

Country Year Population Continent

6 United States 1957 171984000.0 Americas

5 India 1952 372000000.0 Asia

4 France 1957 44310863.0 Europe

3 China 1957 637408000.0 Asia

2 Brazil 1962 76039390.0 Americas

1 Australia 1957 9712569.0 Oceania

0 Afghanistan 1952 8425333.0 Asia

**Q15. How do you create a copy of a Pandas DataFrame?**

**Ans –**

# Importing panda as pd

import pandas as pd

# Creating data

data = {

"name": ["Sally", "Mary", "John"],

"qualified": [True, False, False]

}

# Transform those data into Data Frames

df = pd.DataFrame(data)

#Make a copy: # By default deep is true. Deep value true means. Changes made in original data will not affect the copy

newdf = df.copy()

# print the copy

print(newdf)

# If deep value is False, it means changes made in original data file will affect copy.

**Q16. How do you filter rows of a Pandas DataFrame by multiple conditions?**

Ans-

# Importing panda as pd

import pandas as pd

# Creating data

data = {

"name": ["Sally", "Mary", "John"],

"qualified": [True, False, False]

}

# Transform those data into Data Frames

df = pd.DataFrame(data)

# Filter data frame using multiple condition

df.loc[(df['Year']>=1957) & (df['Population']< 637408000.0)]

Output –

**Q17. How do you calculate the mean of a column in a Pandas DataFrame?**

**Ans –**

# Importing Pandas as pd

import pandas as pd

# Creating data and transforming data into DataFrame

df = pd.DataFrame({"A":[12, 4, 5, 44, 1],

"B":[5, 2, 54, 3, 2],

"C":[20, 16, 7, 3, 8],

"D":[14, 3, 17, 2, 6]})

#Display Output

df

# Output

A B C D

0 12 5 20 14

1 4 2 16 3

2 5 54 7 17

3 44 3 3 2

4 1 2 8 6

# Calculating mean column wise

df.mean(axis = 0)

# Output

A 13.2

B 13.2

C 10.8

D 8.4

dtype: float64

# Calculating mean row wise and skip the Null value if any

df.mean(axis = 1, skipna = True)

#Output

0 12.75

1 6.25

2 20.75

3 13.00

4 4.25

dtype: float64

**Q18. How do you calculate the standard deviation of a column in a Pandas DataFrame?**

Ans –

# Importing Pandas as pd

import pandas as pd

# Creating data and transforming data into DataFrame

df = pd.DataFrame({"A":[12, 4, 5, 44, 1],

"B":[5, 2, 54, 3, 2],

"C":[20, 16, 7, 3, 8],

"D":[14, 3, 17, 2, 6]})

#Display Output

df

# Output

A B C D

0 12 5 20 14

1 4 2 16 3

2 5 54 7 17

3 44 3 3 2

4 1 2 8 6

# Calculate Standard deviation column wise

df.std(axis = 0)

# Output

A 17.683325

B 22.840753

C 6.978539

D 6.730527

dtype: float64

# Calculating standard deviation row wise and skip the Null value if any

df.std(axis = 1, skipna = True)

# Output

0 12.75

1 6.25

2 20.75

3 13.00

4 4.25

dtype: float64

**Q19. How do you calculate the correlation between two columns in a Pandas DataFrame?**

**Ans –**

# import pandas module

import pandas as pd

# create dataframe with 3 columns

data = pd.DataFrame({

"column1": [12, 23, 45, 67],

"column2": [67, 54, 32, 1],

"column3": [34, 23, 56, 23]

}

)

# display dataframe

print(data)

# correlation between column 1 and column2

print(data['column1'].corr(data['column2']))

# correlation between column 2 and column3

print(data['column2'].corr(data['column3']))

# correlation between column 1 and column3

print(data['column1'].corr(data['column3']))

# Output

column1 column2 column3

0 12 67 34

1 23 54 23

2 45 32 56

3 67 1 23

-0.9970476685163736

0.07346999975265099

0.0

# get correlation between element wise

print(data.corr())

Output -

column1 column2 column3

column1 1.000000 -0.997048 0.00000

column2 -0.997048 1.000000 0.07347

column3 0.000000 0.073470 1.00000

**Q20. How do you select specific columns in a DataFrame using their labels?**

**Ans –**

# import pandas

import pandas as pd

# List of Tuples

employees = [('Stuti', 28, 'Varanasi', 20000),

('Saumya', 32, 'Delhi', 25000),

('Aaditya', 25, 'Mumbai', 40000),

('Saumya', 32, 'Delhi', 35000),

('Saumya', 32, 'Delhi', 30000),

('Saumya', 32, 'Mumbai', 20000),

('Aaditya', 40, 'Dehradun', 24000),

('Seema', 32, 'Delhi', 70000)

]

# Create a DataFrame object from list

df = pd.DataFrame(employees,

columns =['Name', 'Age',

'City', 'Salary'])

# Show the dataframe

df

# Using the operator []

# to select a column

result = df["City"]

# Show the dataframe

result

**Q21. How do you select specific rows in a DataFrame using their indexes?**

**Ans-**

# import pandas

import pandas as pd

# List of Tuples

employees = [('Stuti', 28, 'Varanasi', 20000),

('Saumya', 32, 'Delhi', 25000),

('Aaditya', 25, 'Mumbai', 40000),

('Saumya', 32, 'Delhi', 35000),

('Saumya', 32, 'Delhi', 30000),

('Saumya', 32, 'Mumbai', 20000),

('Aaditya', 40, 'Dehradun', 24000),

('Seema', 32, 'Delhi', 70000)

]

# Create a DataFrame object from list

df = pd.DataFrame(employees,

columns =['Name', 'Age',

'City', 'Salary'])

# Show the dataframe

df

# Set 'Name' column as index

# on a Dataframe

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

# Using the operator .loc[]

# to select single row

result = df.loc["Stuti"]

# Show the data

result

**Q22. How do you sort a DataFrame by a specific column?**

**Ans –**

import pandas as pd

data = {'name': ['Somu', 'Kiku', 'Amol', 'Lini'],

'physics': [68, 74, 77, 78],

'chemistry': [84, 56, 73, 69],

'algebra': [78, 88, 82, 87]}

#create dataframe

df\_marks = pd.DataFrame(data)

#sort dataframe

sorted\_df = df\_marks.sort\_values(by='algebra')

print(sorted\_df)

**Q23. How do you create a new column in a DataFrame based on the values of another column?**

**Ans –**

df.[‘new\_Col’] = df[‘Survived’] + df[‘Pclass’]

**Q24. How do you remove duplicates from a DataFrame?**

**Ans –**

# Importing Pandas as pd

import pandas as pd

# Creating data

data = {

"A": ["TeamA", "TeamB", "TeamB", "TeamC", "TeamA"],

"B": [50, 40, 40, 30, 50],

"C": [True, False, False, False, True]

}

# Transforming data into data frames

df = pd.DataFrame(data)

# Print after removing duplicate value

display(df.drop\_duplicates())

# Output

A B C

0 TeamA 50 True

1 TeamB 40 False

3 TeamC 30 False

**Q25. What is the difference between .loc and .iloc in Pandas?**

**Ans –**

The[loc() function](https://www.geeksforgeeks.org/python-pandas-dataframe-loc/) is label based data selecting method which means that we have to pass the name of the row or column which we want to select. This method includes the last element of the range passed in it, unlike iloc(). loc() can accept the boolean data unlike iloc().

# importing the module

import pandas as pd

# creating a sample dataframe

data = pd.DataFrame({'Brand': ['Maruti', 'Hyundai', 'Tata',

'Mahindra', 'Maruti', 'Hyundai',

'Renault', 'Tata', 'Maruti'],

'Year': [2012, 2014, 2011, 2015, 2012,

2016, 2014, 2018, 2019],

'Kms Driven': [50000, 30000, 60000,

25000, 10000, 46000,

31000, 15000, 12000],

'City': ['Gurgaon', 'Delhi', 'Mumbai',

'Delhi', 'Mumbai', 'Delhi',

'Mumbai', 'Chennai', 'Ghaziabad'],

'Mileage': [28, 27, 25, 26, 28,

29, 24, 21, 24]})

# displaying the DataFrame

display(data)

# selecting range of rows from 2 to 5

display(data.loc[2: 5])

The[iloc() function](https://www.geeksforgeeks.org/python-extracting-rows-using-pandas-iloc/) is an indexed-based selecting method which means that we have to pass an integer index in the method to select a specific row/column. This method does not include the last element of the range passed in it unlike loc(). iloc() does not accept the boolean data unlike loc().

# selecting 0th, 2th, 4th, and 7th index rows

display(data.iloc[[0, 2, 4, 7]])