PandasAssignment

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

Ans. To import a CSV file and put the contents in to a Pandas dataframe we use **read_csv(). Read_csv()** takes many arguments but by default you just need to provide the path of the file.

```
import pandas as pd
data = pd.read_csv("weather_data.csv")
```

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

Ans. To check the data type of a column in Pandas we use the “dtype” attribute. The attribute returns a series with a data type of each columns

```
import pandas as pd
data = pd.read_csv("weather_data.csv")
print(data.dtypes)
```

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

Ans. padas.DataFrame.loc is a function used to select rows from Pandas Dataframe based on the condition provided.

Syntax: *df.loc[df[‘cname’] ‘condition’]*
Parameters:
df: *represents data frame*
cname: *represents column name*
condition: *represents condition on which rows has to be selected*

```
from pandas import DataFrame

# Creating a data frame
cart = {'Product': ['Mobile', 'AC', 'Laptop', 'TV',
'Football'],
        'Type': ['Electronic', 'HomeAppliances',
'Electronic',
                 'HomeAppliances', 'Sports'],
        'Price': [10000, 35000, 50000, 30000, 799]
        }

df = DataFrame(cart, columns=['Product', 'Type',
'Price'])
```

```
# Print original data frame
print("Original data frame:\n")
print(df)
print("\n")
selected_Data = df.loc[df["Price"] > 1000]
print(selected_Data)

print("\n")
selected_Data = df.loc[df["Type"] == "Electronic"]
print(selected_Data)
```

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

Ans. Using rename() function

```
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']}

df = pd.DataFrame(rankings)
print(df.columns)

#rename test column to Test
```

```
df.rename(columns={'test':'Test'},inplace=True)
print(df.columns)
```

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

Ans. Using drop()

```
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']}

df = pd.DataFrame(rankings)
print(df.columns)

# remove a specific single column
df.drop(["t20"],axis=1,inplace=True)
print(df.columns)

# remove column based on column index
df.drop(df.columns[[1]],axis=1,inplace=True)
print(df.columns)
```

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

Ans. using unique()

```
import pandas as pd
data = {
```

```
    'A':['A1', 'A2', 'A3', 'A4', 'A5'],
    'B':['B1', 'B2', 'B3', 'B4', 'B4'],
    'C':['C1', 'C2', 'C3', 'C3', 'C3'],
    'D':['D1', 'D2', 'D2', 'D2', 'D2'],
    'E':['E1', 'E1', 'E1', 'E1', 'E1'] }
df = pd.DataFrame(data)
print(df["E"].unique())

# get count of unique values
print(df["D"].nunique(dropna=True))
```

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

Ans. To get the count of missing values in each column of a dataframe, we can use pandas isnull() and sum() function together.

Syntax: df.isnull().sum() --- this will give us the total number of missing values in each column

Df.isnull().sum().sum() ---- this will give us the total number of missing values in the dataframe

```
import pandas as pd

df =
pd.read_csv("https://gist.githubusercontent.com/fyy
ying/4aa5b471860321d7b47fd881898162b7/raw/6907bb3a3
8bfbb6fccf3a8b1edfb90e39714d14f/titanic_dataset.csv
")
print(df.isnull().sum())
```

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

Ans. The fillna() method allow us to replace empty cells with a value.

```
import pandas as pd
df =
pd.read_csv("https://gist.githubusercontent.com/fyy
ying/4aa5b471860321d7b47fd881898162b7/raw/6907bb3a3
8bfbb6fccf3a8b1edfb90e39714d14f/titanic_dataset.csv
")
```

```
df.fillna(130,inplace=True)
print(df.isnull().sum())
```

Q9. How do you concatenate two Pandas DataFrames?

Ans. A concatenation of two or more data frames can be done using padas.concat() method. We can concat two or more data frames either along rows(axis=0) or along columns(axis=1)

```
import pandas as pd
import numpy as np

df1 = pd.DataFrame(np.random.randint(25,
size=(4,4)), index=["1","2","3","4"],
                   columns=["A","B","C","D"])
df2 =
pd.DataFrame(np.random.randint(25,size=(4,4)),index
=["1","2","3","4"],
                   columns=["A","B","C","D"])
vertical_concat = pd.concat([df1,df2],axis=0)
print(vertical_concat)

horizontal_concat = pd.concat([df1,df2],axis=1)
print(horizontal_concat)
```

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

Ans. Pandas provide a single function, merge() as entry point for all the standard database join operations between dataframe objects.

```
import pandas as pd
left = pd.DataFrame({'Key': ['K0', 'K1', 'K2',
'K3'],
                     'A': ['A0', 'A1', 'A2', 'A3'],
                     'B': ['B0', 'B1', 'B2',
'B3']})
```

```
right = pd.DataFrame({'Key': ['K0', 'K1', 'K2',
'K3'],
                      'C': ['C0', 'C1', 'C2',
'C3'],
                      'D': ['D0', 'D1', 'D2',
'D3']})

# Merging the dataframes
print(pd.merge(left,right,how="inner",on="Key"))
```

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

Ans. We can use groupby() method.

```
import pandas as pd
data1 = {

"Name":['Jai','Anuj','Jai','Prince','Gaurav','Anuj'
,'Prince','Abhi'],
    "Age":[27,24,22,32,33,36,27,32],

"Address":["Nagpur","Kanpur","Allahbad","Kanpur","J
aunpur","Kasi","Allahbad","Alligarh"],

"Qualification":["MSC","MA","MCA","PHD","BTECH","BC
om","MSc","MA"]
}

df = pd.DataFrame(data1)
print(df.groupby(["Name"]).mean())
```

Q12. How do you pivot a Pandas DataFrame?

Ans. Using df.pivot() method. This method returns a reshaped dataframe organized by given index/column values. This function uses unique values from specified index/columns to form axes of the resulting dataframe.

```
import pandas as pd
df = pd.DataFrame({'foo': ['one', 'one', 'one',
'two', 'two',
                           'two'],
                   'bar': ['A', 'B', 'C', 'A', 'B',
'C'],
                   'baz': [1, 2, 3, 4, 5, 6],
                   'zoo': ['x', 'y', 'z', 'q', 'w',
't']})
print(df.pivot(index='foo',columns='bar',values=['b
az','zoo']))
```

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

Ans. Using dataframe.astype() method

```
import pandas as pd
df = pd.DataFrame({'No': [1, 2, 3],
                   'Name': ['Geeks', 'for',
                            'Geeks'],
                   'id': [111, 222,
                          333]})
print(df.dtypes)

# changing 'id' datatype to string
df = df.astype({'id':str})
print(df.dtypes)
```

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

Ans. To sort the rows of a Dataframe by a column use pandas.Dataframe.sort_values() method with argument by=column_name.

```
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]}
df = pd.DataFrame(data)
print(df.sort_values(by='algebra'))
```

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

Ans. We can use pandas dataframe copy() function to create a copy of a dataframe. It creates a deep copy by default.

```
# create dataframe df's copy

df.copy(deep=True)
```

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

Ans. Boolean indexing is an effective way to filter a pandas dataframe based on multiple conditions. But remember to use parenthesis to group conditions together and use operator &, | and ~ for performing logical operations on series.

```
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]}
df = pd.DataFrame(data)
df1 = df[(df['physics'] > 70) &(df['physics'] <
80)]
print(df1)
```

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

Ans. Using mean() method.

```
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]}
df = pd.DataFrame(data)
print(df['physics'].mean())
```

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

Ans. To calculate the standard deviation, use std() method of Pandas.

```
import pandas as pd
df = pd.DataFrame(
   {
      "Car": ['BMW', 'Lexus', 'Audi', 'Tesla',
'Bentley', 'Jaguar'],
      "Units": [100, 150, 110, 80, 110, 90]
   }
)

print(df['Units'].std())
```

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

Ans. We can use the .corr() method to get correlation between two columns in Pandas.

```
import pandas as pd
df = pd.DataFrame(
   {
      "x": [5, 2, 7, 0],
      "y": [4, 7, 5, 1],
      "z": [9, 3, 5, 1]
   }
)

col1, col2 = "x", "y"
corr = df[col1].corr(df[col2])
print(corr)
```

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

Ans. Using loc() method.

```
# selecting cars with brand 'Maruti' and Mileage >
25

display(data.loc[(data.Brand == 'Maruti') & amp

                 (data.Mileage & gt

                  25)])
```

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

Ans. using iloc() function

```
# selecting 0th, 2th, 4th, and 7th index rows
display(data.iloc[[0, 2, 4, 7]])



# selecting rows from 1 to 4 and columns from 2 to
4

display(data.iloc[1: 5, 2: 5])
```

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

Ans. To sort the rows of a Dataframe by a column use pandas.Dataframe.sort_values() method with argument by=column_name.

```
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]}
df = pd.DataFrame(data)
print(df.sort_values(by='algebra'))
```

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

```
import pandas as pd
df = pd.DataFrame(
   {
      "x": [5, 2, 7, 0],
      "y": [4, 7, 5, 1],
```

```
        "z": [9, 3, 5, 1]
    }
)
df['z1'] = df['x']+df['y']
print(df)
```

Q24. How do you remove duplicates from a DataFrame?

Ans. To remove all duplicates use drop_duplicates() method.

```
df.drop_duplicates(inplace = True)
```

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

Ans. The main difference is loc[] function is a label based data selecting method which means that we have to pass name of the row/column which we want to select . And iloc[] function is an indexed-based selecting method which means we have to pass an integer index in the method to select a specific row/column.

Loc method includes the last element of the range passed in it, unlike iloc(). Loc() can accept the Boolean data unlike iloc().