2 Python Pandas-II

Fastrack Revision

- In previous chapter, we have learnt to work with series object and how to create DataFrames in many ways. Recalling that series is a 1D Data Structure in Pandas whereas DataFrame is a 2D data structure. In this chapter, we shall talk about DataFrames, basic operations of DataFrame, descriptive statistics, pivoting, handling missing data, combining/merging etc.

Series

	Apples
0	3
1	5
2	4
3	6

Series

	Oranges
0	1
1	2
2	5
3	4

DataFrame

	Apples	Oranges
0	3	1
1	5	2
2	4	5
3	6	4

- **Descriptive Statistics:** Descriptive Statistics are used to summarise the given data. They refer to the methods which are used to get some basic idea about the data. There are various descriptive statistical methods that can be applied to a DataFrame. These are max, min, count, sum, mean, median, mode, quartiles and variance.
- **Calculating Maximum Values:** DataFrame.max() is used to calculate the maximum values from the DataFrame, regardless of its data types. By default, the max() method finds the maximum value of each column.

  Syntax: <DataFrame>.max(axis = None, skipna =None, numeric_only = None)
- **Calculating Minimum Values:** DataFrame.min() is used to display the minimum values from the DataFrame, regardless of the data types. It shows the minimum value of each column or row.

  Syntax: <DataFrame>.min(axis = None, skipna =None, numeric_only = None)
- **Calculating Sum of Values:** DataFrame.sum() will display the sum of the values from the DataFrame regardless of its datatype.

  Syntax: <DataFrame>.sum(axis = None, skipna =None, numeric_only = None, min_count = 0)
- **Calculating Number of Values:** DataFrame.count() will display the total number of values for each column or row of a DataFrame.

  Syntax: <DataFrame>.count(axis = 0, numeric_only = False)
- **Calculating Mean:** DataFrame.mean() will display the mean (average) of the values of each column of a DataFrame. It is only applicable for numeric values.

  Syntax: <DataFrame>.mean(axis = None, skipna =None, numeric_only = None)
- **Calculating Median:** DataFrame.Median() will display the middle value of the data. This function will display the median of the values of each column of a DataFrame. It is only applicable for numeric values.

  Syntax: <DataFrame>.median(axis = None, skipna =None, numeric_only = None)
- **Calculating Mode:** DateFrame.mode() will display the mode. The mode is defined as the value that appears the most number of times in a data. This function will display the mode of each column or row of the DataFrame.

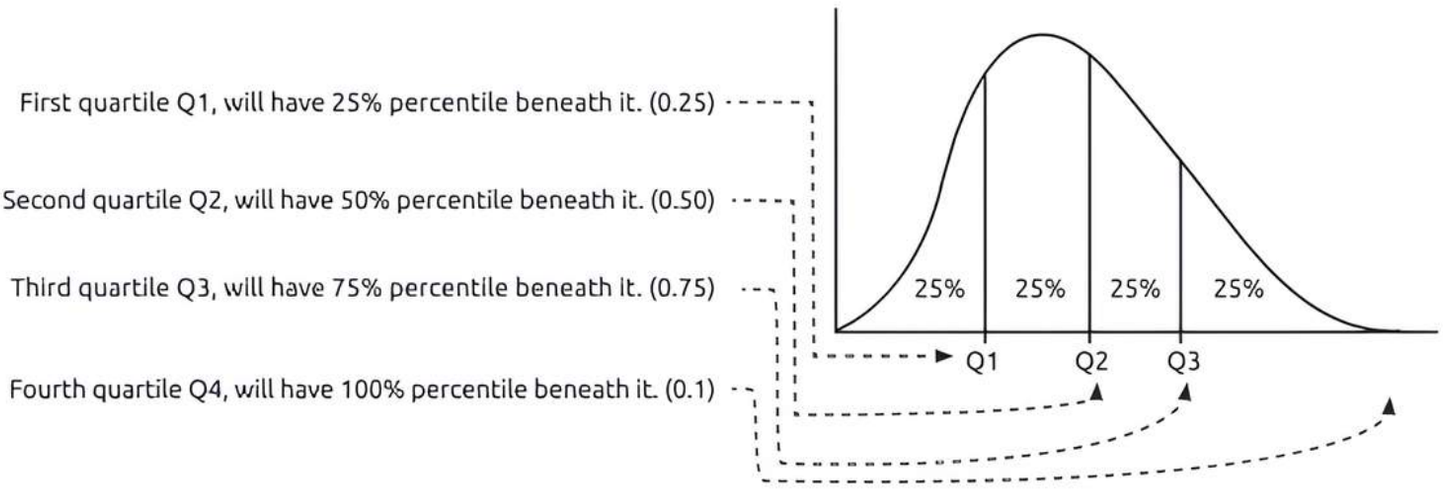
  Syntax: <DataFrame>.mode(axis = 0, numeric_only = False)
- **Calculating Variance:** DataFrame.var() is used to display the variance. It is the average of squared differences from the mean.

  Syntax: <DataFrame>.var(axis = None, skipna = None, numeric_only = None)
- **Calculating Standard Deviation:** DataFrame.std() returns the standard deviation of the values. Standard deviation is calculated as the square root of the variance.

  Syntax: <DataFrame>.std(axis = None, skipna = None, numeric_only = None)
- **DataFrame.describe()** function displays the descriptive statistical values in a single command. These values help us describe a set of data in a DataFrame.

  Syntax: <DF>.describe( )
- **Calculating Quartile:** DataFrame.quartile() is used to get the quartiles. It will output the quartile of each column or row of the DataFrame in four parts *i.e.* the first quartile is 25% (parameter q = .25), the second quartile is 50% (Median), the third quartile is 75% (parameter q = .75). By default, it will display the second quartile (median) of all numeric values.
- **Quartile:** It refers to the division of the total distribution of given data into a four equal proportions with each containing one-fourth of the total population.
- **Quantiles:** It is a process of dividing the total distribution of given data into a given number of equal proportions.

  Syntax: <DataFrame>.quantile(q = 0.5, axis = 0, numeric_only = True)



- **Data Aggregations:** Aggregation means to transform the dataset and produce a single numeric value from an array. Aggregation can be applied to one or more columns together. Aggregate functions are max(),min(), sum(), count(), std(), var().
- **Various Aggregation Functions:**

S. No.	Aggregation	Description
1.	count()	Total number of items
2.	sum()	Sum of all items
3.	mean(), median()	Mean and median
4.	min(), max()	Minimum and maximum
5.	std(), var()	Standard deviation and variance
6.	mad()	Mean absolute deviation

- **mad() Function:** It is used to calculate the mean absolute deviation of the values for the requested axis. The Mean Absolute Deviation (MAD) of a set of data is the average distance between each data value and the mean.

  Syntax: <DataFrame>.mad(axis = None, skipna = None)

- **Sorting a DataFrame:** Sorting refers to the arrangement of data elements in a specified order, which can either be ascending or descending. Pandas provide sort_values() function to sort the data values of a DataFrame.

  Syntax: <DataFrame>.sort_values(by, axis=0, ascending= True, inplace=False, kind ='quicksort', na_position='last')

- **GROUP BY Functions:** In pandas, DataFrame.GROUP BY() function is used to split the data into groups based on some criteria. Pandas objects like a DataFrame can be split on any of their axes. The GROUP BY function works based on a split-apply-combine strategy which is shown below using a 3-step process:

  Step 1: Split the data into groups by creating a GROUP BY object from the original DataFrame.

  Step 2: Apply the required function.

  Step 3: Combine the results to form a new DataFrame.

- **Reshaping Data:** Reshaping data refers to the process of changing the shape of the dataset to make it suitable for some analysis problems. For reshaping data, two basic functions are available in Pandas, pivot and pivot_table.
- **Pivoting:** It is a technique to rearrange the data from rows and columns by possibly rotating rows and columns or by aggregating data from multiple sources, in a report form.
- **Pivot:** The pivot( ) function is used to reshape and create a new DataFrame from the original one.
- **Pivot Table:** It works like a pivot function, but aggregates the values from rows with duplicate entries for the specified columns. The default aggregate function is mean.

  Syntax: pandas.pivot_table(<DataFrame>, values=None, index=None, columns=None, aggfunc='mean')

  Or

  Syntax: <DataFrame>.pivot_table(values=None, index=None, columns=None, aggfunc='mean')

- **Iterating over a DataFrame:** There are many ways using which we can iterate over a DataFrame. Most common are **<DFobject>.itemrows()** and **<DFobject>.iteritems()**.
  - **<DFobject>.itemrows():** This method views a DataFrame in the form of horizontal subsets *i.e.*, row-wise.
  - **<DFobject>.iteritems():** This method views a DataFrame in the form of vertical subsets *i.e.*, column-wise.
- **Binary Operations in a DataFrame:**
  - **Addition Binary Operation:** We can perform add binary operation on two DataFrame objects using either + (add) operator or using add( ) as per syntax: **<DF1>.add(<DF2>)** which means **<DF1>+<DF2>** or by using radd( ) *i.e.*, reverse add as per syntax: **<DF1>.radd(<DF2>)** which means **<DF2>+<DF1>**
  - **Subtraction Binary Operation:** We can perform subtract binary operation on two DataFrame objects using either – (minus) operator or using sub( ) as per syntax:

<DF1>.sub(<DF2>) which means <DF1> – <DF2> or by using rsub() *i.e.*, reverse subtract as per syntax: <DF1>.rsub(<DF2>) which means <DF2> – <DF1>

- **Multiplication Binary Operation:** We can perform multiplication binary operation on two DataFrame objects using either × operator or using mul( ) as per syntax : <DF>.mul(<DF>)
- **Division Binary Operation:** We can perform division binary operation on two DataFrame objects using either/(division) operator or using div( ) as per syntax : <DF>.div(<DF>)

- **Handling Missing Values:** Missing values are the values that cannot contribute to any computation or we can say that missing values are the values that carry no computational significance. A missing value is denoted by NaN.
- Missing values create a lot of problems during data analysis and have to be handled properly. The two most common strategies for handling missing values are:
  - Drop the object having missing values.
  - Fill or estimate the missing value.
- **Detecting/Filtering Missing Data:** Pandas provide a function isnull() to check whether any value is missing or not in the DataFrame. This function checks all attributes and returns true in case that attribute has missing values, otherwise returns False. It can be used as: **<PandaObject>.isnull( )**
- **Dropping Missing Values:** Missing values can be handled by either dropping the entire row having missing value or replacing it with appropriate value. Dropping will remove the entire row (object) having the missing value(s).
- **Dropna( ) Function:** To drop missing values we can use dropna( ) in following three ways:
  - **<PandaObject>.dropna( ):** This will drop all the rows that have NaN values in them, even row with a single NaN value in it.
  - **<DF>.dropna(how ='all'):** It will drop only those rows that have all NaN values *i.e.*, no value is non – null in those rows.
  - **<DF>.dropna(axis = 1):** It will drop columns that have any NaN values in them.
- **Filling Missing Values:** Missing values can be filled by using estimations or approximations for example, a value just before (or after) the missing value, average/minimum/maximum of the values of that attribute, etc. For this purpose, we can use fillna( ) in following ways:
  - **<PandaObject>.fillna(<n>):** This will fill all NaN values in a Pandas object with the given <n> value.
  - **Using dictionary with fillna( ) to specify fill values for each column separately:** The syntax to use for this format of fillna( ) is: **<DF>.fillna(<dictionary having fill values for columns>**

Practice Exercise

Multiple Choice Questions

Q 1. To iterate over horizontal subsets of DataFrame, ... function may be used.
a. iterate() b. iterrows()
c. itercols() d. iteritems()

Q 2. To iterate over vertical subsets of a DataFrame, ... function may be used.
a. iterate() b. iterrows()
c. itercols() d. iteritems()

Q 3. To add two DataFrames' values, ... function may be used.
a. plus b. add
c. radd d. either b. or c.

Q 4. To subtract the values of two DataFrames, ... function may be used.
a. sub b. difference
c. rsub d. either a. or c.

Q 5. To divide the values of two DataFrames, ... function may be used.
a. divide b. div
c. rdiv d. either b. or c.

Q 6. To skip NaN values in a calculation, you can specify ..... attribute.
a. NaN b. NA
c. skipna d. All of these

Q 7. Which of the following is not a valid function that can be used with DataFrames?
a. count() b. sum() c. length() d. mad()

Q 8. A technique, which when performed on a DataFrame, rearranges the data from rows and columns in a report form, is called .....
a. summarising b. reporting
c. grouping d. pivoting

Q 9. The technique that divides total distribution of data into a given number of equal proportions is called a .....
a. quartile b. tercile
c. median d. quantile

Q 10. ..... divides the total distribution in four equal parts.
a. Quartile b. Tercile
c. Median d. Quantile

Q 11. To divide total distribution of given data in two equal parts, ... function is used.
a. median() b. quartile()
c. quantile() d. All of these

Q 12. To divide total distribution of given data in four equal parts, ... function is used.
a. median() b. quartile()
c. quantile() d. All of these

Q 13. To divide total distribution of given data in eight equal parts, ... function is used.
a. median() b. quartile()
c. quantile() d. All of these

Q 14. The technique to summarise given data by transferring rows to columns is called .....
a. transfer b. transpose
c. pivoting d. swapping

Q 15. The function to perform pivoting with DataFrames having unique values is .....
a. pivot(unique = True)
b. pivot()
c. pivot_table(unique = True)
d. pivot_table()

Q 16. The function to perform pivoting with DataFrames having duplicate values is .....
a. pivot(unique = True)
b. pivot()
c. pivot_table(unique = True)
d. pivot_table()

Q 17. The function to create histograms for all numeric columns of a DataFrame is .....
a. histogram()
b. hist(numeric_only = True)
c. hist()
d. All of the above

Q 18. In Python Pandas, while performing mathematical operations on series, index matching is implemented and all missing values are filled in with ..... by default. *[CBSE SQP 2023-24]*
a. Null b. Blank
c. NaN d. Zero

Q 19. Which function calculates descriptive statistical details for a DataFrame?
a. info() b. describe()
c. show() d. list()

Q 20. To calculate cumulative sum of a column of a DataFrame, you may use ..... function.
a. sum() b. sum(cumulative =True)
c. cumsum() d. None of these

Q 21. The function to get the index of maximum value in a column of DataFrame is .....
a. max() b. index()
c. idxmax() d. maxidx()

Q 22. To get top 5 rows of a DataFrame, you may use ..... function.
a. head() b. head(5)
c. top() d. Either a. or b.

Q 23. To get bottom 3 rows of a DataFrame, you may use ..... function.
a. tail() b. tail(3)
c. bottom() d. bottom(3)

Q 24. Function ..... can be used to drop missing values.
a. fillna() b. isnull()
c. dropna() d. delna()

Q 25. Which of the following methods of combining two DataFrames is a patching method?
a. concat() b. merge()
c. join() d. None of these

Fill in the Blanks Type Questions

Q 26. Using ..... function you can calculate octiles.

Q 27. Using ..... method, you can check if there are any missing values in DataFrame.

Q 28. To concatenate two DataFrames, you can use ..... method.

Q 29. The ..... function combines two DataFrames such that two rows with some common value are merged together in the final result.

Q 30. ..... function sorts on the basis of the values of a DataFrame.

Q 31. ..... function sorts on the basis of index of a DataFrame.

Assertion & Reason Type Questions

Directions (Q. Nos. 32-39): *In the questions given below, there are two statements marked as Assertion (A) and Reason (R). Read the statements and choose the correct option.*

a. Both Assertion (A) and Reason (R) are true and Reason (R) is the correct explanation of Assertion (A).
b. Both Assertion (A) and Reason (R) are true, but Reason (R) is not the correct explanation of Assertion (A).
c. Assertion (A) is true, but Reason (R) is false.
d. Assertion (A) is false, but Reason (R) is true.

Q 32. **Assertion (A):** Descriptive Statistics are used to summarise the given data.
Reason (R): They refer to the methods which are used to get some basic idea about the data.

Q 33. **Assertion (A):** DataFrame.max() is used to calculate the maximum values from the DataFrame, regardless of its data types.
Reason (R): There are various calculating maximum values methods that can be applied to a DataFrame. These are max, min, count, sum, mean, median, mode, quartiles and variance.

Q 34. **Assertion (A):** DataFrame.min() is used to display the minimum values from the DataFrame, regardless of the data types.
Reason (R): It shows the minimum value of each column or row.

Q 35. Assertion (A): DataFrame.count() will display the total number of values for each column or row of a DataFrame.
Reason (R): DataFrame.sum() will display the sum of the values from the DataFrame regardless of its datatype.

Q 36. Assertion (A): DataFrame.describe() function displays the descriptive statistical values in a single command. These values help us describe a set of data in a DataFrame.
Reason (R): DataFrame.mean() will display the half of the average of the values of each column of a DataFrame. It is only applicable for numeric values.

Q 37. Assertion (A): Quantiles is a process of multiplying the total distribution of given data into a given number of equal proportions.
Reason (R): DataFrame.var() is used to display the variance. It is the average of squared differences from the mean.

Q 38. Assertion (A): DataFrame.quantile() is used to get the quartiles.
Reason (R): It will output the quartile of each column or row of the DataFrame in four parts.

Q 39. Assertion (A): Quartile refers to the division of the total distribution of given data into a five equal proportions with each containing one-fifth of the total population.
Reason (R): <DFobject>.itemrows() method views a DataFrame in the form of horizontal subsets i.e., row-wise.

Answers

1. (b)	2. (d)	3. (d)	4. (d)	5. (d)
6. (c)	7. (c)	8. (d)	9. (d)	10. (a)
11. (c)	12. (c)	13. (c)	14. (c)	15. (b)
16. (d)	17. (c)	18. (c)	19. (b)	20. (c)
21. (c)	22. (d)	23. (b)	24. (c)	25. (c)

26. quantile() 27. isnull()
28. concat() 29. merge()
30. sort_values() 31. sort_index

32. (a)	33. (c)	34. (a)	35. (b)	36. (c)
37. (d)	38. (a)	39. (d)		

Case Study Based Questions

Case Study 1

Descriptive Statistics with Pandas: Python Pandas is a widely used data science library and it offers many useful functions. Among many other functions, the functions offered by Pandas also include many useful statistical functions. Descriptive Statistics are used to summarise the given data. They refer to the methods which are used to get some basic idea about the data. There are various descriptive statistical methods that can be applied to a DataFrame. These are max, min, count, sum, mean, median, mode, quartiles and variance.

Q 1. What does the axis parameter includes?
a. 0 (zero) b. 1 (one)
c. Both (a) and (b) d. None of these

Q 2. What does the numeric_only parameter includes?
a. Float b. Int
c. Boolean d. All of these

Q 3. What is the correct syntax for mode() function?
a. <DataFrame>.std(axis = None, skipna =None, numeric_only = None)
b. <DataFrame>.mode(axis = 0, numeric_only = False)
c. <DataFrame>.count(axis = 0, numeric_only = False)
d. None of the above

Q 4. Function ..... returns the computed mean (average) from a set of values.
a. var() b. mode()
c. median() d. mean()

Q 5. What does the skipna parameter excludes?
a. Null Values b. NA values
c. NaN values d. All of these

Answers

1. (c) 2. (d) 3. (b) 4. (d) 5. (d)

Case Study 2

Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A DataFrame is a two-dimensional data structure, *i.e.*, data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the data, rows and columns. In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file and Excel file. Pandas DataFrame can be created from the lists, dictionary and from a list of dictionary, etc.

Q 1. The following code create a DataFrame named 'D1' with ..... columns.
```
import pandas as pd
D1 = pd.DataFrame ( [1, 2, 3] )
```
a. 1 b. 2
c. 3 d. 4

Q 2. We can create DataFrame from .....
a. numpy arrays b. list of dictionaries
c. dictionary of lists d. All of these

Q 3. Which of the following is used to give user defined column index in DataFrame?

a. index b. column
c. columns d. colindex

Q 4. The following code create a DataFrame named 'D1' with ..... columns.

```
import pandas as pd
LoD = [ {'a': 10, 'b': 20}, {'a': 5, 'b': 10, 'c':20} ]
D1 = pd.DataFrame (LoD)
```

a. 1 b. 2
c. 3 d. 4

Q 5. The following code create a DataFrame named 'D1' with ..... rows.

```
import pandas as pd
LoD = [ {'a': 10, 'b': 20}, {'a': 5, 'b': 10, 'c':20} ]
D1 = pd.DataFrame (LoD)
```

a. 0 b. 1
c. 2 d. 3

Answers

1. (a) **2.** (d) **3.** (c) **4.** (c) **5.** (b)

Case Study 3

Pandas DataFrame is a widely used data structure which works with a two-dimensional array with labeled axes (rows and columns). DataFrame is defined as a standard way to store data that has two different indexes, i.e., row index and column index. It consists of the following properties: The columns can be heterogeneous types like int, bool and so on. It can be seen as a dictionary of series structure where both the rows and columns are indexed. It is denoted as "columns" in case of columns and "index" in case of rows.

Parameter and Description:

- **Data:** It consists of different forms like ndarray, series, map, constants, lists and array.
- **Index:** The Default np.arrange(n) index is used for the row labels if no index is passed.
- **Columns:** The default syntax is np.arange(n) for the column labels. It shows only true if no index is passed.

Q 1. When we create DataFrame from List of Dictionaries, then dictionary keys will become .....

a. column labels b. row labels
c. Both (a) and (b) d. None of these

Q 2. When we create DataFrame from List of Dictionaries, then number of columns in DataFrame is equal to the .....

a. maximum number of keys in first dictionary of the list
b. maximum number of different keys in all dictionaries of the list
c. maximum number of dictionaries in the list
d. None of the above

Q 3. When we create DataFrame from List of Dictionaries, then number of rows in DataFrame is equal to the .....

a. maximum number of keys in first dictionary of the list
b. maximum number of keys in any dictionaries of the list
c. number of dictionaries in the list
d. None of the above

Q 4. In given code DataFrame 'D1' has ..... rows and ..... columns.

```
import pandas as pd
LoD = [ {'a': 10, 'b': 20}, {'a': 5, 'b': 10, 'c':20}, {'a': 7, 'd': 10, 'e': 20}
D1 = pd.DataFrame (LoD)
```

a. 3, 3 b. 3, 4
c. 3, 5 d. None of these

Q 5. When we create DataFrame from Dictionary of List then keys becomes the .....

a. row labels b. column labels
c. Both (a) and (b) d. None of these

Answers

1. (a) **2.** (b) **3.** (c) **4.** (c) **5.** (b)

Case Study 4

Pandas is a popular Python package for data science and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The DataFrame is one of these structures. Those who are familiar with R know the DataFrame as a way to store data in rectangular grids that can easily be overviewed. Each row of these grids corresponds to measurements or values of an instance, while each column is a vector containing data for a specific variable. This means that a DataFrame's rows do not need to contain, but can contain, the same type of values: they can be numeric, character, logical, etc. Now, DataFrames in Python are very similar: they come with the Pandas library and they are defined as two-dimensional labeled data structures with columns of potentially different types.

Q 1. Name some valid function that can be used with DataFrames?

Q 2. Define pivoting.

Q 3. In which condition quantile function is used?

Q 4. Define pivot_table().

Q 5. Which function calculates descriptive statistical details for a DataFrame?

Answers

1. Some valid function that can be used with DataFrames are count(), sum() and mad().
2. A technique, which when performed on a DataFrame, rearranges the data from rows and columns in a report form, is called pivoting.
3. To divide total distribution of given data in two equal parts, quantile function is used.
4. The function to perform pivoting with DataFrames having duplicate values is pivot_table().
5. describe()

Case Study 5

Descriptive Statistics in Python: Python Descriptive Statistics process describe the basic features of data in a study. It delivers summaries on the sample and the measures and does not use the data to learn about the population it represents. Under descriptive statistics, fall two sets of properties-central tendency and dispersion. Python Central tendency characterises one central value for the entire distribution. Measures under this include mean, median and mode. Python Dispersion is the term for a practice that characterises how apart the members of the distribution are from the center and from each other. Variance/Standard Deviation is one such measure of variability.

Q 1. Write the correct syntax for calculating Standard Deviation.

Q 2. Write the correct syntax for calculating Number of Values.

Q 3. Write the correct syntax for calculating Sum of values.

Q 4. Write the correct syntax for calculating Variance.

Q 5. Write the correct syntax for calculating Median.

Answers

1. Syntax: <DataFrame>.std(axis=None, skipna = None, numeric_only = None)
2. Syntax: <DataFrame>.count(axis=0, numeric_only = False)
3. Syntax: <DataFrame>.sum(axis=None, skipna =None, numeric_only = None, min_count = 0)
4. Syntax: <DataFrame>.var(axis=None, skipna =None, numeric_only = None)
5. Syntax: <DataFrame>.median(axis=None, skipna =None, numeric_only = None)

Very Short Answer Type Questions

Q 1. Name the functions that give you maximum and minimum values in a DataFrame.

Ans. max() and min() function.

Q 2. Name the functions that give you the index of maximum and minimum values in a DataFrame.

Ans. idmax() and idmin()

Q 3. Write appropriate functions to perform the following on a DataFrame:
(i) Calculate the sum (ii) Count the values

Ans. (i) df.sum() (ii) df.count()

Q 4. Write appropriate functions to perform the following on a DataFrame:
(i) Calculate the average
(ii) Calculate the median

Ans. (i) df.mean() (ii) df.median()

Q 5. Answer the following questions:
(i) Write appropriate functions to calculate the variance on a DataFrame.
(ii) Name some functions that perform descriptive statistics on a DataFrame.

Ans. (i) df.var() (ii) describe()

Q 6. What does info( ) and describe( ) do?

Ans. The describe() is used to view some basic statistical details. The info() function is used to print a summary of a DataFrame. This method show information about a DataFrame.

Q 7. Is there one function that calculates much of descriptive statistics values? Name it.

Ans. Yes, describe( ) function.

Q 8. What happens if mode() returns multiple values for a column but other columns have a single mode?

Ans. The other columns have some NaN values.

Q 9. What is quartile and quantile?

Ans. **Quartile:** It refers to the division of the total distribution of given data into four equal proportions with each containing one-fourth of the total population.

Quantile: It is a process of dividing the total distribution of given data into a given number of equal proportions.

Q 10. Name the function that lets you calculate different types of quantiles.

Ans. quantile()

Q 11. What is pivoting? Name the two functions that you can use for pivoting.

Ans. Pivoting means summarising the data in a way to make understanding of descriptive data easier.
Functions are: - pivot(), pivot_table()

Q 12. What is the basic difference between pivot() and pivot_table( )?

Ans. **pivot ():** It is used for pivoting the DataFrame without applying aggregation. It doesn't contain same values or columns/index.

pivot_table (): It will pivot the DataFrame by applying aggregation on it. And it will work with duplicate values or columns/Index.

Q 13. Name the function to iterate over a DataFrame horizontally.

Ans. iterrows()

Q 14. Name the function to iterate over a DataFrame vertically.

Ans. iteritems()

Q 15. Name the functions you can use to iterate over DataFrames.

Ans. iterrows() and iteritems()

Q 16. What is the basic difference between iterrows() and iteritems()?

Ans. **iteritems():** It iterates over vertical subsets in the form of (col – index, series) pairs.

iterrows(): It iterates over horizontal subsets in the form of (row – index, series) pairs.

Q 17. Write equivalent expressions for the given functions:

(i) A.add(B) **(ii) B.add(A)**

Ans. (i) A+B (ii) B+A

Q 18. Write equivalent expressions for the given functions:

(i) A.sub(B) **(ii) B.sub(A)**

Ans. (i) A-B (ii) B-A

Q 19. Write equivalent expressions for the given functions:

(i) A.rsub(B) **(ii) B.mul(A)**

Ans. (i) B-A (ii) B*A

Q 20. Is the result of sub() and rsub() the same? Why/ Why not?

Ans. No, because A.sub(B) is subtraction of 'B' from 'A' while A.rsub(B) is subtraction of 'A' from 'B'.

COMMON ERROR

There may be confusion regarding the two functions as they both are used for subtraction but there is a slight difference.

Q 21. What are missing values?

Ans. Missing values are the values that cannot contribute to any computation. Missing values means NaN values.

Q 22. Why estimation is an important concept in data analysis?

Ans. Estimation will surely change the datasets. In all cases, data analysis results will not be actual results but will be a good approximation of actual results.

Q 23. Explain the importance of reshaping of data with an example.

Ans. Reshaping of data refers to the process of changing the shape of the datasets to make it suitable for some analysis problems.

Q 24. Name the functions you can use for filling missing data.

Ans. fillna()

Q 25. Name some functions used to handle missing data in DataFrames.

Ans. fillna(), dropna() and isnull()

Short Answer Type-I Questions

Q 1. What is descriptive statistics? Name the functions commonly used for calculating this.

Ans. A descriptive statistic is a summary statistic that quantitatively describes or summarises features of a collection of information.

Commonly used functions for descriptive statistics are: count(), sum(), mean(), max(), min(), std(), quartiles, etc.

Q 2. Write appropriate functions to perform the following on a DataFrame?

(i) Calculate the standard deviation

(ii) Calculate the variance

(iii) Calculate the maximum value

Ans. (i) df.std() (ii) df.var()

(iii) df.max() (iv) df.mode()

Q 3. Write a program to print a DataFrame, df one row at a time.

Ans.
```
import pandas as pd
dict = {'Name' : ["Ram", "Pam", "Sam"],
        'Marks' : [ 70, 95, 80]}
df = pd. DataFrame (dict, index = ['Rno.1', Rno.2', 'Rno.3'])
for i, j in df. interrows ():
    print (j)
    print ("_____")
```

Q 4. Write a program to print a DataFrame, df one column at a time.

Ans.
```
import pandas as pd
dict = {'Name' : ["Ram", "Pam", "Sam"],
        'Marks' : [70, 95, 80]}
df = pd. DataFrame(dict, Index = ['Rno. 1', 'Rno. 2', 'Rno. 3'])
for I, j In df. iteritems():
    print(j)
    print ("_____")
```

Q 5. Write a program to print a DataFrame one column at a time and print only first three columns.

Ans.
```
import pandas as pd
import numpy as z
dict = { 'name' : [1, 2, 3] }
df = pd.DataFrame(dict)
for i , j in df.iteritems() :
    print( j )
```

Q 6. Write a program to print a DataFrame one row at a time and print only first five rows.

Ans.
```
import pandas as pd
import numpy as z
dict = { 'name' : [1,2,3] }
df = pd.DataFrame(dict)
for I , j In df.Iterrows():
print( j )
```

Q 7. Write a program that performs count, sum, max and min functions:

(i) On rows **(ii) On columns**

Ans. (i) On rows

```
import pandas as pd
import numpy as z
dict = { 'name' : [1, 2, 3] }
df = pd.DataFrame(dict)
df.min() #min
df.max() #max
df.count() #count
df.sum() #sum
```

(ii) On columns

```
df.min(axis = 1) #min
df.max(axis = 1) #max
df.count(axis = 1) #count
df.sum(axis = 1) #sum
```

Q 8. Take a DataFrame of your choice. Write a program to calculate count of value only in a selective column.

Ans.

```
import pandas as pd
import numpy as z
dict = { 'name' : [1, 2, 3], 'name2' : [11, 22, 44] }
df = pd.DataFrame(dict)
print( df[ 'name' ].count())
```

Q 9. A DataFrame fdf stores data about passengers, flights and years. First few rows of the DataFrame are shown below:

	Year	Month	Passengers
0	2009	January	112
1	2009	February	118
2	2009	March	132
3	2009	April	129
4	2009	May	121

Using above DataFrame, write commands for the following:

(i) Compute total passengers per year.

(ii) Compute average passengers per month.

Ans. (i) fdf.pivot_table(index = 'year', values = 'passengers', aggfunc = 'sum')

(ii) fdf.pivot_table(index = 'month', values = 'passengers', aggfunc = 'mean')

Q 10. The info() and describe() are said to be inspection functions. What do they do?

Ans. describe() is used to view some basic statistical details like percentile, mean, std, etc. of a DataFrame or a series of numeric values.

info() Print a concise summary of a DataFrame. This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.

Q 11. Write the purpose of Data aggregation.

Ans. Aggregation means to transform the dataset and produce a single numeric value from an array. Aggregation can be applied to one or more columns together. Aggregate functions are max().min(), sum(), count(), std(), var().

Q 12. What is pivoting? Which functions of Pandas support pivoting?

Ans. Data pivoting is a summarising technique to rearrange the columns and rows in a report so as to view data from different perspectives.

Pandas library makes available two functions for pivoting-the pivot() and pivot_table() functions.

Q 13. What is the difference between pivot() and pivot_table() functions?

Ans. The pivot() and pivot_table() both perform data pivoting a data set. But with pivot(), if there are multiple entries for a column's value for the same values for index(row), it leads to an error.

The pivot_table() pivots the data by aggregating it thus it can work with duplicate entries.

Q 14. What are binary operations? Name the functions that let you perform binary operations on a DataFrame.

Ans. Binary operations means operations requiring two values to perform and these values are picked elementwise.

Functions are:

(i) + or add() (ii) - or sub() or rsub()

(iii) * or mul() (iv) / or div() or rdiv()

Q 15. Write equivalent expressions for the given functions:

(i) A.rdiv(B) **(ii) B.div(A)**

(iii) B.rdiv(A) **(iv) A.div(B)**

Ans. (i) B/A (ii) B/A

(iii) A/B (iv) A/B

Q 16. Given two DataFrames df3 and df4 as shown below:

>>>df3

	A	B	C
0	100	200	300
1	400	500	600

>>>df4

	A	B
0	1000	2000
1	4000	5000
2	7000	8000

>>>df3 + df4

	A	B	C
0	1100.0	2200.0	NaN
1	4400.0	5500.0	NaN
2	NaN	NaN	NaN

Both these DataFrames store integer values but when they are added as df3 + df4, the values in the resultant object automatically change to floating point (as shown on above right) contrary to the fact the two integers when added will result into integer only. Can you specify the reason?

Ans. The reason behind the conversion to floating point type is that the two DataFrames have different indexes and columns. For the non-matching row indexes and columns, Python will add NaN values to corresponding value from another DataFrame.

Python stores NaN values in a non-integer suitable data type. Thus, the moment NaN is added or present in any column, the datatype of the entire column is changed. Thus, all the values are represented as floating point value because of the presence of NaN values in their column.

Q 17. Given a DataFrame namely wdf as shown below:

(i) Write command to compute sum of every column of the DataFrame.

(ii) Write command to compute mean of column Rainfall.

(iii) Write command to compute sum of every row of the DataFrame.

(iv) Write command to compute average of all the columns for last 10 rows only.

	minTemp	maxTemp	Rainfall	Evaporation
0	1	8.0	24.3	0.0
1	2	14.0	26.9	3.6
2	3	13.7	23.4	3.6
3	4	13.3	15.5	39.8
4	5	7.6	16.1	2.8
5	6	6.2	16.9	0.0
6	7	6.1	18.2	0.2
7	8	8.3	17.0	0.0
8	9	8.8	19.5	0.0
9	10	8.4	22.8	16.2
10	11	9.1	25.2	0.0
11	12	8.5	27.3	0.2
12	13	10.1	27.9	0.0
13	14	12.1	30.9	0.0
14	15	10.1	31.2	0.0
15	16	12.4	32.1	0.0
16	17	13.8	31.2	0.0
17	18	11.7	30.0	1.2
18	19	12.4	32.3	0.6
19	20	15.6	33.4	0.0
20	21	15.3	33.4	0.0

Ans. (i) wdf. sum() (ii) wdf['Rainfall']. mean ()
(iii) wdf. sum(axis = 1) (iv) wdf.loc[11 : ,]. mean ()

Q 18. What is missing data? Why is it considered a problem?

Ans. Missing data means when no information is provided for one or more items or for a whole unit. Missing data can also refer to as NA (Not Available) values in Pandas. Pandas puts NaN in place of missing data in DataFrames.

Missing data is a very big problem in real life scenario. This is because, the presence of NaN hampers calculations or NaN cannot be used in calculations and in fact, it makes the whole calculation result as NaN.

Q 19. What are missing values? What are the strategies to handle them?

Ans. If a value corresponding to a column is not present, it is considered to be a missing value. A missing value is denoted by NaN.

The two most common strategies for handling missing values explained in this section are:

(i) drop the object having missing values

(ii) fill or estimate the missing value

Q 20. Why does Python change the datatype of a column as soon as it stores an empty value (NaN) even though it has all other values stored as integer?

Ans. The reason behind it is that whenever any column of a DataFrame has a NaN value in it, then its type is automatic changed to floating point because in python integer types cannot store NaN values.

Short Answer Type-II Questions

Q 1. Define the following terms: Median, Standard Deviation and Variance with their syntax.

Ans. **Median:** It will display the middle value of the data. DataFrame.Median() function will display the median of the values of each column of a DataFrame. It is only applicable for numeric values.

Syntax: <DataFrame>.median(axis = None, skipna =None, numeric_only = None)

Standard Deviation: It returns the standard deviation of the values. Standard deviation is calculated as the square root of the variance. DataFrame.std() function is used to calculate standard deviation.

Syntax: <DataFrame>.std(axis = None, skipna =None, numeric_only = None)

Variance: It is used to display the variance. It is the average of squared differences from the mean. DataFrame.var() function is used to calculate variance.

Syntax: <DataFrame>.var(axis = None, skipna =None, numeric _only = None)

COMMON ERROR

Students do not write proper syntax for all the terms. Revise them properly to avoid any error.

Q 2. Given a DataFrame df1 as shown below:

City	MaxTemp	MinTemp	RainFall
Delhi	40	32	24.1
Bengaluru	31	25	36.2
Chennai	35	27	40.8
Mumbai	29	21	35.2
Kolkata	39	23	41.8

(i) Write command to compute sum of every column of the DataFrame.

(ii) Write command to compute mean of column Rainfall.

(iii) Write command to compute median of the MaxTemp Column. *[CBSE SQP 2019-20]*

Ans. (i) df1.sum()

(ii) df1['RainFall'].mean()

(iii) df1.loc[:,'MaxTemp'].median()

Q 3. Consider the following DataFrame and answer the questions given below:

```
import pandas as pd
df = pd.DataFrame ({"Quarter1" : [2000, 4000, 5000, 4400, 10000],
                    "Quarter2" : [5800, 2500, 5400, 3000, 2900],
                    "Quarter3" : [20000, 16000, 7000, 3600, 8200],
                    "Quarter4" : [1400, 3700, 1700, 2000, 6000]})
```

(i) Write the code to find mean value from above DataFrame df over the index and column axis.

(ii) Use sum() function to find the sum of all the values over the Index axis.

(iii) Find the median of the DataFrame df.

[CBSE SQP 2019-20]

Ans. (i)
```
print(df.mean(axis = 1))
print(df.mean(axis= 0))
```
(ii)
```
print (df.sum(axis = 1))
```
(iii)
```
print (df.median())
```

Q 4. Write the Python statement for the following question on the basis of given datasets:

	Name	Degree	Score
0	Aparna	MBA	90.0
1	Pankaj	BCA	NaN
2	Ram	M.Tech	80.0
3	Ramesh	MBA	98.0
4	Naveen	NaN	97.0
5	Krrishnav	BCA	78.0
6	Bhawna	MBA	89.0

(i) To display the name and degree wise average marks of each student.

(ii) To count the number of students in MBA.

(iii) To print the mode of marks of BCA.

Ans. (i) print ('degree wise average marks of each student.')
```
for y in d:
print (f' average score in {y} is')
print (df1 [df1 ['Degree'] == y]. mean())
print()
print()
```
(ii) len (df1[df1 ['Degree'] == 'mba'])

(iii) df1[df1 ['Degree'] == 'bca']. mode()

Q 5. Assuming the given table: Product. Write the python code for the following:

Item	Company	Rupees	USD
TV	LG	12000	700
TV	VIDEOCON	10000	650
TV	LG	15000	800
AC	SONY	14000	750

(i) To create the DataFrame for the above table.

(ii) To add the new rows in the DataFrame.

(iii) To display the sum of all products.

Ans. (i)
```
import mysql.connector as a
db=a.connect(user='root',passwd
='0000000000',
host='localhost',database='hindustan')
import pandas as pd
df=pd.read_sql(f'select * from product',db)
print(df)
```
(ii) df.loc[4] = ['corona_vaccine','hindustan',74.1]

(iii) df.Rupees.sum()

Q 6. Given the two DataFrames as:

In [126]: dfc1
Out[126]:
0 1
0 2 a
1 3 b
2 4 c

In [130]: dfc2
Out[130]:
0 1 2
0 2 a 4
2 p q r

Why are following statements giving errors?

(i) print(dfc1 + dfc2)

(ii) print(dfc1. sub(dfc2))

(iii) print(dfc1 * dfc2)

Ans. (i) String and number cannot be added.

(ii) String and number cannot be subtracted.

(iii) String and number cannot be multiplied.

Q 7. What is a quartile? How is it related to quantile? How do you generate it in Pandas?

Ans. Quartiles: Q1, Q2 and Q3 are three points that divide a distribution into 4 equal parts containing 25% percentile each of the entire distribution. The 4-quantiles are called ***quartiles***. A quantile refers to an equal share in an equally divided distribution for example, median quantile divides a distribution into 2 equal parts and each equal share is 50% quantile. **Quartile**, on the other hand, refers to when a distribution is divided into four quantiles each containing 25% percentile.

In Pandas, we generate these with function **quantile()**.

Q 8. Consider the following code that creates two DataFrames:

```
ore1 = pd.DataFrame(np.array([[20,35,25,20], [11,28,32, 29]]), columns = ['iron', 'magnesium', 'copper', 'silver'])
ore2 = pd. DataFrame(np.array([[14, 34, 26, 26], [33, 19, 25, 23]]), columns = ['iron', 'magnesium', 'gold', 'silver'])
```

What will be the output produced by the following code fragments:

(i)
```
print(ore1 + ore2)
ore3 = ore1.radd(ore2)
print(ore)
```

(ii)
```
print (ore1 – ore2)
ore3 = ore1.rsub(ore2)
print(ore3)
```

Ans. (i)
```
print(ore1+ore2)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	34	69	46
1	NaN	NaN	44	47	52

```
ore3=ore1.radd(ore2)
print(ore3)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	34	69	46
1	NaN	NaN	44	47	52

(ii)
```
print(ore1 – ore2)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	6	1	-6
1	NaN	NaN	-22	9	6

```
ore3=ore1.rsub(ore2)
print(ore3)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	-6	-1	6
1	NaN	NaN	22	-9	-6

Q 9. Consider the following code that creates two DataFrames:

```
ore1 = pd.DataFrame(np.array([[20,35,25,20], [11,28,32, 29]]), columns = ['iron', 'magnesium', 'copper', 'silver'])
ore2 = pd. DataFrame(np.array([[14, 34, 26, 26], [33, 19, 25, 23]]), columns = ['iron', 'magnesium', 'gold', 'silver'])
```

What will be the output produced by the following code fragments:

(i)
```
print (ore1 * ore2)
ore3 = ore1.mul(ore2)
print(ore3)
```

(ii)
```
print (ore1/ore2)
ore3 = ore1.rdiv(ore2)
print(ore3)
```

Ans. (i)
```
print(ore1*ore2)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	280	1190	520
1	NaN	NaN	363	532	667

```
ore3=ore1.mul(ore2)
print(ore3)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	280	1190	520
1	NaN	NaN	363	532	667

(ii)
```
print(ore1/ore2)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	1.428571	1.029412	0.769231
1	NaN	NaN	0.333333	1.473684	1.260870

```
ore3=ore1.rdiv(ore2)
print(ore3)
```

	copper	gold	iron	magnesium	silver
0	NaN	NaN	0.7	0.971429	1.300000
1	NaN	NaN	3.0	0.678571	0.793103

Q 10. Given two identical DataFrames Sales16 and Sales17. But Sales17 has some values missing. Write code so that Sales17 fills its missing values from corresponding entries of Sales16.

Ans.
```
import pandas as pd
import numpy as np
sales16 = pd.DataFrame( data = [531. 35. 27. 93. \
          30. 40. 5. 81. 51. 14. 59. 96]. columns = \
          ( 'closingPrice' ) )
sales17 = pd.DataFrame( data = [531. 35. 27. 93. \
          np.nan. 40. 5. np.nan. 51. 14. 59. 96 ]. \
          columns = \
          ( 'closingPrice' ) )
print( sales17.fillna(sales16) )
```

Q 11. Given a DataFrame df:

	Name	Sex	Position	City	Age	Projects	Budget
0	Rabia	F	Manager	Bangalore	30	13	48
1	Evan	M	Programmer	New Delhi	27	17	13
2	Jia	F	Manager	Chennai	32	16	32
3	Lalit	M	Manager	Mumbai	40	20	21
4	Jaspreet	M	Programmer	Chennai	28	21	17
5	Suji	F	Programmer	Bangalore	32	14	10

Write a program to print only the Name, Age and Position for all rows.

Ans.
```
import pandas as pd
import numpy as np
: # given df created or loaded
for i, row in df. interrows():
    print(row['Name']. '\t'. row['Age']. '\t'. row ['Position'])
```

Output:

Rabia	30	Manager
Evan	27	Programmer
Jia	32	Manager
Lalit	40	Manager
Jaspreet	28	Programmer
Suji	32	Programmer

Q 12. Consider DataFrames pdf and pdf2 as shown below:

pdf

	Sean	Shadab	Sia	Simran
L1	295.0	600.0	505	NaN
L2	397.0	NaN	600	397.0
L3	NaN	800.0	700	500.0

pdf2

	Sean	Shadab	Sia	Simran
L1	295	600.0	400	200.0
L2	397	200.0	600	397.0
L3	400	NaN	500	NaN

Write a program to fill the missing values in pdf with corresponding values.

Ans.
```
import pandas as pd
import numpy as np
:  # given pdf, pdf2 created or loaded
print("Original pdf")
print(pdf)
print ("DataFrame pdf after filling values from DataFrame pdf2")
Print(pdf.fillna(pdf2))
```

Output:

original pdf

	Sean	Shadab	Sia	Simran
L1	295.0	600.0	505	NaN
L2	397.0	NaN	600	397.0
L3	NaN	800.0	700	500.0

DataFrame pdf after filling values from DataFrame pdf2

	Sean	Shadab	Sia	Simran
L1	295.0	600.0	505	200.0
L2	397.0	200.0	600	397.0
L3	400.0	800.0	700	500.0

Long Answer Type Questions

Q 1. Answer the following questions:

Given a DataFrame mks1 as shown below:

Sub	A	B	C	D
Acct	99	94	92	99
Eco	94	94	92	97
Eng	95	89	91	89
IP	94	87	99	94
Math	97	100	99	99

(i) Consider the DataFrame mks1 given above. Write a program to calculate mode (maximum repeated value) for each subject.

(ii) Consider the DataFrame mks1 given above. Write a program to calculate median value for each section.

(iii) Consider the DataFrame mks1 given above. Write a program to calculate median value for each subject.

(iv) Consider the DataFrame mks1 given above. Write a program to calculate average value (mean) for each section.

(v) Consider the DataFrame mks1 given above. Write a program to calculate average value (mean) for each subject.

Ans. (i)
```
import pandas as pd
import numpy as np
:  # given mks1 created or loaded
print ("Mode value for each subject")
print(mks1.mode(axis = 1))
```

Output:

Mode value for each subject

Acct	99
Eco	94
Eng	89
IP	94
Math	99

dtype: int 64

(ii)
```
import pandas as pd
import numpy as np
:  # given mks1 created or loaded
print ("Median value for each section")
print(mks1.median())
```

Output:

Median value for each section

A	95.0
B	94.0
C	92.0
D	97.0

dtype:float64

(iii)
```
import pandas as pd
import numpy as np
:  # given mks1 created or loaded
print ("Median value for each subject")
print(mks1.median (axis = 1))
```

Output:

Median value for each subject

Acct	96.5
Eco	94.0

```
Eng     90.0
IP      94.0
Math    99.0
dtype: float64
```

(iv)
```
import pandas as pd
import numpy as np
:   # given mks1 created or loaded
print ("Average (Mean) marks for each section")
print(mks1.mean())
```

Output:

Average (Mean) marks for each section

```
A    95.8
B    92.8
C    94.6
D    95.6
dtype: float64
```

(v)
```
import pandas as pd
import numpy as np
:   # given mks1 created or loaded
print ("Average (Mean) marks for each subject")
print(mks1.mean(axis = 1))
```

Output:

Average (mean) marks for each subject

```
Acct    96.00
Eco     94.25
Eng     91.00
IP      93.50
Math    98.75
dtype : float64
```

Q 2. Write the Python statement for the following question on the basis of given datasets:

	Name	Degree	Score
0	Aparna	MBA	90.0
1	Pankaj	BCA	NaN
2	Ram	M.Tech	80.0
3	Ramesh	MBA	98.0
4	Naveen	NaN	97.0
5	Krrishnav	BCA	78.0
6	Bhawna	MBA	89.0

(i) To create the above DataFrame.

(ii) To print the Degree and maximum marks in each stream.

(iii) To fill the NaN with 76.

Ans. (i)
```
d={'mba':'bca':'mtech'}
dic={'Name':['aparna':'pankaj':'ram':'ramesh'],'Degree':
d+{'mba'},'score':[90,np.NaN,80,98]}
df1=pd.DataFrame(dic)
print(df1)
```

(ii)
```
print(df['Degree'])
#code for maximum marks in each stream.
import numpy as np
import pandas as pd
d={'mba':'bca':'mtech'}
dic={'Name':['aparna':'pankaj':'ram':'ramesh'],'Degree':
d+{'mba'},'score':[90,np.NaN,80,98]}
df1=pd.DataFrame(dic)
print(df1)
print()
for y in d:
    print(f'max score in {y} is ')
    print(df1[df1['Degree']==y].max())
    print()
    print()
```

(iii)
```
df1.fillna(76)
```

Knowledge BOOSTER

The fillna ( ) method replace the NULL values with a specified value. Sometimes, incorrect syntax is written for this method.

Q 3. Assuming the given table: Product, write the Python code for the following:

Item	Company	Rupees	USD
TV	LG	12000	700
TV	VIDEOCON	10000	650
TV	LG	15000	800
AC	SONY	14000	750

(i) To sort the data according to the Rupees and transfer the data to MySQL.

(ii) To transfer the new DataFrame into the MySQL with new values.

(iii) To display the median of the USD of Sony products.

Ans. (i)
```
print(df.sort_values(by={'rupees'}))
from sqlalchemy import create_engine
engine=create_engine('mysql+pymysql://
root:0000000000@localhost/hindustan')
conn=engine.connect()
df.to_sql('Product_table',conn,index=False)
#data transferred to sql
```

(ii)
```
d={'item':['tv'],'Company':['lg'],'rupees':[17000]}
df2=pd.DataFrame(d)
df.append(df2) # now df have new values
from sqlalchemy import create_engine
engine=create_engine('mysql+pymysql://
root:0000000000@localhost/hindustan')
conn=engine.connect()
df.to_sql('new_Product_table',conn,index=False)
#data transferred to sql
```

(iii)
```
df[df['Company']=='Sony']['USD'].median()
```

Q 4. Write a program to iterate over a DataFrame containing names and marks, which then calculates grades as per marks (as per guidelines below) and adds them to grade column.

Marks > = 90 grade A+; Marks 50-60 grade C;
Marks 70–90 grade A; Marks 40-50 grade D;
Marks 60-70 grade B; Marks < 40 grade F

Ans.

```
import pandas as pd
import numpy as np
names = pd. Series (['Rohan', 'Misha', 'Mike', 'Simran'])
marks = pd. Series ([76.0, 56.0, 91.0, 67.0])
Stud = {'Name' : names, 'Marks' : marks}
df1 = pd. DataFrame(Stud. columns = ['Name', 'Marks'])
df1 ['Grade'] = np. NaN
Print ("Initial values in DataFrame")
print(df1)
for (col, colSeries) in df1. iteritems():
    length = len(colSeries)              # number of entries in colSeries
    if col == 'Marks':
      lstMrks = []                       # initialise empty list
      for row in range (length):
        mrks = colSeries [row]
        if mrks >= 90:
          1st Mrks. append('A+')         # grade appended to list 1stMrks
        elif mrks >= 70:
          1stMrks. append ('A')          # grade appended to list 1stMrks
        elif mrks >= 60:
          1stMrks. append('B')           # grade appended to list 1stMrks
        elif mrks >= 50:
          1stMrks. append('C')           #grade appended to list 1stMrks
        elif mrks >= 40:
          elifMrks. append('D')          #grade appended to list 1stMrks
        else :
          1stMrks. append('F'))          #grade appended to list 1stMrks
df1 ['Grade'] = 1stMrks
Print ("nDataFrame after calculating grades")
print(df1)
```

Output:

Initial values in DataFrame

	Name	Marks	Grade
0	Rohan	76.0	NaN
1	Misha	56.0	NaN
2	Mike	91.0	NaN
3	Simran	67.0	NaN

DataFrame after calculating grades

	Name	Marks	Grade
0	Rohan	76.0	A
1	Misha	56.0	C
2	Mike	91.0	A+
3	Simran	67.0	B

Chapter Test

Multiple Choice Questions

Q 1. Which of the following is the correct syntax to select or access columns from the DataFrame using column names?

a. df(col1,col2,....,coln) b. df[[col1,col2,....,coln]]
c. df{col1,col2,...,coln] d. df{col1,col2,...,:coln}

Q 2. Ms. Kavitha wants to print a single column from the DataFrame, which of the following is correct syntax for her?

a. df(col) b. df[col]
c. df<col> d. df{df:col}

Q 3. The following code create a DataFrame named 'D1' with ..... columns.

```
import pandas as pd
LoD = [{'a': 10, 'b': 20}, {'a': 5, 'b': 10, 'c': 20}]
D1 = pd.DataFrame(LoD)
```

a. 1 b. 2
c. 3 d. 4

Q 4. The following code create a DataFrame named 'D1' with ..... rows.

```
import pandas as pd
LoD = [{'a': 10, 'b': 20}, {'a': 5, 'b': 10, 'c': 20}]
D1 = pd.DataFrame(LoD)
```

a. 0 b. 1
c. 2 d. 3

Q 5. In given code DataFrame 'D1' has ..... rows and ..... columns.

```
import pandas as pd
LoD = [ {'a': 10, 'b': 20}, {'a': 5, 'b': 10, 'c':20}, {'a': 7, 'd': 10, 'e': 20} ]
D1 = pd.DataFrame (LoD)
```

a. 3, 3 b. 3, 4
c. 3, 5 d. None of these

Fill in the Blanks

Q 6. ..... and ..... functions help you to iterate over a DataFrame.

Q 7. To add two DataFrames, you may use functions ..... *or* ..... .

Q 8. To concatenate two string columns of a DataFrame, ... operator is used.

Assertion & Reason Type Questions

Directions (Q. Nos. 9-11): *In the questions given below, there are two statements marked as Assertion (A) and Reason (R). Read the statements and choose the correct option.*

a. Both Assertion (A) and Reason (R) are true and Reason (R) is the correct explanation of Assertion (A).
b. Both Assertion (A) and Reason (R) are true, but Reason (R) is not the correct explanation of Assertion (A).
c. Assertion (A) is true, but Reason (R) is false.
d. Assertion (A) is false, but Reason (R) is true.

Q 9. **Assertion (A): mad() function is used to calculate the mean absolute deviation of the values for the requested axis.**
Reason (R): The Mean Absolute Deviation (MAD) of a set of data is the average distance between each data value and the mean.

Q 10. **Assertion (A): In pandas, DataFrame.GROUP BY() function is used to split the data into groups based on some criteria. Pandas objects like a DataFrame can be split on any of their axes.**
Reason (R): Reshaping data refers to the process of changing the shape of the dataset to make it suitable for some analysis problems.

Q 11. **Assertion (A): Missing values can be handled by either dropping the entire column having missing value or replacing it with appropriate value.**
Reason (R): <PandaObject>.dropna() will drop all the rows that have NaN values in them, even row with a single NaN value in it.

Case Study Based Questions

Q 12. **Combining DataFrames:** While working, we may encounter situations where we have access to two or more similar DataFrames containing more or less similar data. To make a decision based on those multiple DataFrames having similar data, you would want a way that combines the two DataFrames in some efficient way so that missing data in one DataFrame is picked from another similar DataFrame or dissimilar entries from two DataFrames are combined to form one DataFrame.

(i) **What are various ways to combine multiple DataFrame?**
a. merge() b. join()
c. concat() d. All of these

(ii) **The concat() can concatenate two DataFrames along .....**
a. rows b. columns
c. Both a. and b. d. None of these

(iii) **The concat() method is useful if the two DataFrames have ..... structures.**
a. similar
b. dissimilar
c. Both a. and b.
d. None of these

(iv) **The join() method used by DataFrames basically creates a new DataFrame from two DataFrames by joining their .....**
a. columns b. rows
c. index d. table

(v) **The DataFrame that calls the join() is the first DataFrame known as .....**
a. Second DataFrame
b. Other DataFrame
c. Calling DataFrame
d. Index DataFrame

Q 13. **Working with Missing Data in Pandas:** Missing Data can occur when no information is provided for one or more items or for a whole unit. Missing Data is a very big problem in a real-life scenarios. Missing Data can also refer to as NA (Not Available) values in Pandas. In DataFrame sometimes many datasets simply arrive with missing data, either because it exists and was not collected or it never existed. For Example, suppose different users being surveyed may choose not to share their income, some users may choose not to share the address in this way many datasets went missing.

In Pandas missing data is represented by two values:

- **None:** None is a Python singleton object that is often used for missing data in Python code.
- **NaN:** NaN (an acronym for Not a Number), is a special floating-point value recognised by all systems that use the standard IEEE floating-point representation.

(i) **Define isnull() function.**
(ii) **What do you mean by dropping missing values?**
(iii) **What is the meaning of the syntax <DF>. dropna(axis = 1)?**
(iv) **What is the meaning of the syntax <PandaObject>.fillna(<n>)?**
(v) **What do you mean by the term pivot()?**

Very Short Answer Type Questions

Q 14. Name some functions used to join or combine DataFrames.

Q 15. Name two functions that can produce result similar to SQL joins.

Q 16. Name and explain one cumulative functions provided by Pandas.

Q 17. Define concat( ) function.

Short Answer Type-I Questions

Q 18. What is the difference between cumulative sum and sum? How do you perform the two on DataFrame?

Q 19. The head() and tail() extract rows or columns from a DataFrame. Explain.

Short Answer Type-II Questions

Q 20. A DataFrame Stu stores details like 'Name', 'Class', 'Subject_Id' for 10 students and another DataFrame Marks stores details like 'Subject_Id' and 'Avgmarks'. Write code so that two DataFrames merge data on the basis of common Subject id.

Q 21. A DataFrame Result stores the details as (Rollno., marks). Write a program to join it with a DataFrame Stu storing details as (Name, Class, Subject). Make sure that all the rows from both the DataFrames are part of the resultant DataFrame.

Long Answer Type Questions

Q 22. In an online contest, two player teams' points in 4 rounds are stored in two DataFrames as shown below:

	Team 1's points df1			Team 2's points df2	
	p1	p2		p1	p2
1	700	490	1	1100	1400
2	975	460	2	1275	1260
3	970	570	3	1270	1500
4	900	590	4	1400	1190

Answer the following questions based on the given two DataFrames.

(i) Write a program to calculate total points earned by both the teams in each round.

(ii) Consider the points earned by two teams, display how much point difference Team2 has with Team1.

(iii) Given two DataFrames storing points of a 2-player teams in four rounds, write a program to calculate average points obtained by each player in each round.

Q 23. Answer the following questions:

Given a DataFrame dfmks as shown below:

Sub	A	B	C	D
Acct	99	94.0	92	97.0
Eco	90	94.0	92	97.0
Eng	95	89.0	91	89.0
IP	94	NaN	99	95.5
Math	97	100.0	99	NaN

(i) Consider the DataFrame dfmks given above. Write a program to print maximum marks scored in each subject across all sections.

(ii) Which statement would you change in the above program so that it considers only non-Nan values for calculation purpose?

(iii) Consider the DataFrame dfmks given above. Write a program to print the maximum marks scored in a section, across all subjects.