**Isnull() :-**

The isnull() function in Pandas is used to detect missing values in a DataFrame or Series. It returns a boolean mask indicating whether each value is null (True) or not null (False).

Common uses of isnull():

1. Identify missing values: df.isnull().sum() shows the count of missing values in each column.

2. Filter missing values: df[df['column\_name'].isnull()] selects rows with missing values in a specific column.

3. Drop missing values: df.dropna() removes rows with missing values; df.dropna(how='all') removes rows with all values missing.

4. Fill missing values: df.fillna(value) replaces missing values with a specified value.

**Isnull.sum() :-**

1. isnull(): Detects missing values in a DataFrame or Series, returning a boolean mask (True for missing values, False otherwise).

2. sum(): Sums up the True values (missing values) in the boolean mask, returning the count of missing values.

In essence, isnull().sum() calculates the total number of missing values in each column of a DataFrame.

Example:

import pandas as pd

data = {'Name': ['John', 'Anna', None, 'Peter'],

'Age': [28, 24, 35, None],

'City': ['NYC', None, 'LA', 'Chicago']}

df = pd.DataFrame(data)

print(df.isnull().sum())

**shape() :-**

The shape() function in Pandas returns the dimensions of a DataFrame or Series, represented as a tuple:

(number of rows, number of columns)

For a DataFrame:

import pandas as pd

data = {'Name': ['John', 'Anna', 'Peter'],

'Age': [28, 24, 35]}

df = pd.DataFrame(data)

print(df.shape)

Output:

(3, 2)

**Max() :-**

The max() function in Pandas returns the maximum value in a DataFrame or Series.

For a DataFrame:

- df.max(): Returns a Series with the maximum value in each column.

- df.max(axis=1): Returns a Series with the maximum value in each row.

For a Series:

- series.max(): Returns the maximum value in the Series.

Use cases for max():

1. Find the highest value: Identify the maximum value in a column or row.

2. Data validation: Check if a value exceeds a maximum allowed value.

3. Filtering: Select rows or columns with values below or above a maximum threshold.

4. Aggregation: Calculate the maximum value in a group of data.

**Describe() :-**

The describe() function in Pandas provides a summary of the central tendency, dispersion, and shape of a dataset's distribution.

For a DataFrame or Series, describe() returns a summary of:

1. Count: Number of non-null values.

2. Mean: Average value.

3. Std: Standard deviation.

4. Min: Minimum value.

**Use cases for describe():**

1. Quick data overview: Get a sense of the data's distribution and central tendency.

2. Data quality check: Identify missing values, outliers, or data entry errors.

3. Exploratory data analysis: Understand the data's shape, central tendency, and variability.

4. Feature engineering: Select relevant features for modeling based on summary statistics.

**Min() :-**

The min() function in Pandas returns the minimum value in a DataFrame or Series.

For a DataFrame:

- df.min(): Returns a Series with the minimum value in each column.

- df.min(axis=1): Returns a Series with the minimum value in each row.

For a Series:

- series.min(): Returns the minimum value in the Series.

Use cases for min():

1. Find the lowest value: Identify the minimum value in a column or row.

2. Data validation: Check if a value is below a minimum allowed value.

3. Filtering: Select rows or columns with values above or below a minimum threshold.

4. Aggregation: Calculate the minimum value in a group of data.

**Shape() :-**

The shape() function in Pandas returns the dimensions of a DataFrame or Series as a tuple:

(number of rows, number of columns)

Use cases for shape():

1. Quickly check the size of a dataset: Verify the number of rows and columns.

2. Data validation: Ensure the expected number of rows and columns.

3. Looping and indexing: Use the shape to iterate over rows and columns.

4. Reshaping data: Use the shape to transform data from wide to long format or vice versa.

5. Memory optimization: Check the shape to optimize memory usage.

**Corr() :-**

The corr() function in Pandas calculates the correlation between columns in a DataFrame.

Use cases for corr():

1. Identify relationships: Detect positive or negative correlations between variables.

2. Feature selection: Select relevant features for modeling based on correlation.

3. Data exploration: Understand the underlying structure of the data.

4. Predictive modeling: Use correlation to inform feature engineering and model selection.

Example:

import pandas as pd

data = {'A': [1, 2, 3, 4, 5],

'B': [2, 3, 5, 7, 11],

'C': [1, 4, 9, 16, 25]}

df = pd.DataFrame(data)

print(df.corr())