Pandas Series & DataFrame Methods

Series

Description: One-dimensional labeled array capable of holding any data type. Similar to a column in an Excel spreadsheet.

Syntax: pd.Series(data, index=None)

Example: pd.Series([10, 20, 30], index=['a', 'b', 'c'])
Creates a Series with values 10,20,30 and custom indices a,b,c

dtypes

Description: Returns the data types of each column in a DataFrame. Helps identify if columns are integers, floats, objects, etc.

Syntax: df.dtypes

Example: df.dtypes
Returns: age int64, name object, salary float64

dtype

Description: Returns the data type of a single Series object.

Syntax: series.dtype

Example: pd.Series([1,2,3]).dtype
Returns: int64

size

Description: Returns the total number of elements in the Series or DataFrame.

Syntax: df.size

Example: df.size
For a 3x3 DataFrame returns 9

shape

Description: Returns a tuple representing the dimensions (rows, columns) of the DataFrame.

Syntax: df.shape

Example: df.shape
For 100 rows and 5 columns returns (100, 5)

values

Description: Returns the data from the Series or DataFrame as a NumPy array.

Syntax: df.values

Example: df.values
Returns: array([[1,2,3], [4,5,6]])

mean

Description: Returns the average (mean) value of the data. For DataFrames, calculates mean for each numeric column.

Syntax: df.mean(axis=0, skipna=True)

Example: df['Score'].mean()
Returns: 85.5 (average of all scores)

max

Description: Returns the maximum value from the data.

Syntax: df.max()

Example: df['Temperature'].max()
Returns: 98.6 (highest temperature)

min

Description: Returns the minimum value from the data.

Syntax: df.min()

Example: df['Price'].min()
Returns: 10.99 (lowest price)

std

Description: Returns the standard deviation, measuring data spread around the mean.

Syntax: df.std()

Example: df['Scores'].std()
Returns: 15.3 (shows how spread out scores are)

value_counts

Description: Counts the frequency of each unique value in a Series. Useful for categorical data.

Syntax: series.value_counts(normalize=False)

Example: df['City'].value_counts()
Returns: New York 5, Los Angeles 3, Chicago 2

DataFrame

Description: Two-dimensional labeled data structure with columns of potentially different types.

Syntax: pd.DataFrame(data, index=None, columns=None)

Example: pd.DataFrame({'Name': ['Alice', 'Bob'], 'Age': [25, 30]})
Creates DataFrame with Name and Age columns

read_csv

Description: Reads a comma-separated values (CSV) file into a DataFrame.

Syntax: pd.read_csv('file.csv', sep=',', nrows=None)

Example: pd.read_csv('sales.csv', nrows=100)
Reads first 100 rows from sales.csv

read_excel

Description: Reads an Excel file (.xlsx, .xls) into a DataFrame.

Syntax: pd.read_excel('file.xlsx', sheet_name=0)

Example: pd.read_excel('data.xlsx', sheet_name='Sheet1')
Reads Sheet1 from Excel file

read_json

Description: Reads a JSON (JavaScript Object Notation) file into a DataFrame.

Syntax: pd.read_json('file.json')

Example: pd.read_json('employees.json')
Reads JSON file with employee data

head

Description: Returns the first n rows of the DataFrame. Useful for quick data inspection.

Syntax: df.head(n=5)

Example: df.head(3)
Returns first 3 rows of the DataFrame

tail

Description: Returns the last n rows of the DataFrame.

Syntax: df.tail(n=5)

Example: df.tail(3)
Returns last 3 rows of the DataFrame

sample

Description: Returns random rows from the DataFrame for sampling.

Syntax: df.sample(n=5, frac=None)

Example: df.sample(n=2)
Returns 2 randomly selected rows

ndim

Description: Returns the number of dimensions (1 for Series, 2 for DataFrame).

Syntax: df.ndim

Example: df.ndim
Returns: 2 for DataFrame, 1 for Series

columns

Description: Returns the column labels of the DataFrame.

Syntax: df.columns

Example: df.columns
Returns: Index(['Name', 'Age', 'City'], dtype='object')

info

Description: Prints a concise summary of the DataFrame including data types and non-null counts.

Syntax: df.info()

Example: df.info()
Shows column names, data types, and memory usage

describe

Description: Generates descriptive statistics (count, mean, std, min, quartiles, max) for numeric columns.

Syntax: df.describe(include='all')

Example: df.describe()
Shows statistical summary of all numeric columns

select_dtypes

Description: Selects columns from DataFrame based on their data types.

Syntax: df.select_dtypes(include=['int'])

Example: df.select_dtypes(include=['float64'])
Returns all float columns

iloc

Description: Integer-location based indexing for selecting data by position.

Syntax: df.iloc[row, col]

Example: df.iloc[0, 1]
Selects first row, second column value

iat

Description: Fast integer scalar access for accessing a single value by position.

Syntax: df.iat[row, col]

Example: df.iat[2, 3]
Quickly accesses value at row 3, column 4

isnull

Description: Detects missing values (NaN, None) in the DataFrame. Returns boolean mask.

Syntax: df.isnull()

Example: df.isnull().sum()
Counts missing values in each column

notnull

Description: Detects non-missing values. Opposite of isnull().

Syntax: df.notnull()

Example: df[df['Age'].notnull()]
Returns rows where Age is not null

dropna

Description: Removes missing values (rows or columns containing NaN).

Syntax: df.dropna(axis=0, inplace=False)

Example: df.dropna()
Removes all rows with any missing values

fillna

Description: Fills missing values with specified value or method.

Syntax: df.fillna(value, method=None)

Example: df.fillna(0)
Replaces all NaN values with 0

duplicated

Description: Returns boolean Series indicating duplicate rows.

Syntax: df.duplicated()

Example: df.duplicated()
Identifies duplicate rows in DataFrame

drop_duplicates

Description: Removes duplicate rows from the DataFrame.

Syntax: df.drop_duplicates()

Example: df.drop_duplicates(keep='first')
Removes duplicates keeping first occurrence

sort_values

Description: Sorts DataFrame by one or more columns.

Syntax: df.sort_values(by='col', ascending=True)

Example: df.sort_values(by='Salary', ascending=False)
Sorts by Salary descending

sort_index

Description: Sorts DataFrame by index labels.

Syntax: df.sort_index()

Example: df.sort_index(ascending=False)
Sorts index in descending order

reset_index

Description: Resets the index of the DataFrame, converting index to a column.

Syntax: df.reset_index(drop=False)

Example: df.reset_index(drop=True)
Resets index and drops the old index

corr

Description: Computes pairwise correlation of columns. Shows relationship between variables.

Syntax: df.corr(method='pearson')

Example: df.corr()
Shows correlation matrix between numeric columns

unique

Description: Returns unique values in a Series. Order of appearance is preserved.

Syntax: series.unique()

Example: df['Category'].unique()
Returns array of unique categories

nunique

Description: Counts the number of distinct/unique elements.

Syntax: series.nunique()

Example: df['City'].nunique()
Returns number of unique cities (e.g., 5)

apply

Description: Applies a function along an axis of the DataFrame.

Syntax: df.apply(func, axis=0)

Example: df['Age'].apply(lambda x: x*2)
Doubles all ages

to_csv

Description: Writes DataFrame to a comma-separated values (CSV) file.

Syntax: df.to_csv('file.csv', index=False)

Example: df.to_csv('output.csv', index=False)
Saves DataFrame without row indices

to_excel

Description: Writes DataFrame to an Excel file.

Syntax: df.to_excel('file.xlsx', index=False)

Example: df.to_excel('report.xlsx', sheet_name='Data')
Saves to Excel with custom sheet name

to_json

Description: Writes DataFrame to a JSON file.

Syntax: df.to_json('file.json')

Example: df.to_json('data.json', orient='records')
Saves as JSON with record orientation

mode

Description: Returns the mode(s) (most frequent values) of each column.

Syntax: df.mode()

Example: df['Color'].mode()
Returns most frequent color (e.g., 'Blue')

sum

Description: Returns the sum of values along the requested axis.

Syntax: df.sum()

Example: df['Sales'].sum()
Returns total sales sum

to_datetime

Description: Converts argument to datetime format.

Syntax: pd.to_datetime(col, errors='coerce')

Example: pd.to_datetime(df['Date'])
Converts date strings to datetime objects

concat

Description: Concatenates pandas objects along a particular axis.

Syntax: pd.concat([df1, df2], axis=0)

Example: pd.concat([df1, df2], ignore_index=True)
Stacks DataFrames vertically

merge

Description: Merges DataFrame objects by performing a database-style join.

Syntax: pd.merge(df1, df2, on='id', how='inner')

Example: pd.merge(employees, depts, on='dept_id')
Joins employee and department data