Use describe() Function in Pandas (With Examples)

Authored by

Mohammed looti

November 3, 2025

The **describe()** function is a foundational method within the Pandas library, designed to quickly generate **descriptive statistics** for a given DataFrame. This powerful utility provides a rapid summary of the central tendency, dispersion, and shape of the dataset's distribution, making it the essential first step in any data exploration process.

At its most basic level, invoking the **describe()** function requires only the DataFrame object itself. It automatically calculates key metrics, offering an immediate snapshot of the underlying data quality and characteristics.

df.describe()

To fully illustrate the practical application of this function, we will utilize a sample Pandas DataFrame containing fictional team performance metrics:

import pandas as pd

#create DataFrame
df = pd.DataFrame({'team': ,
'points': ,
'assists': ,
'rebounds': })

#view DataFrame
df

team points assists rebounds
0 A 25 5 11
1 A 12 7 8
2 B 15 7 10
3 B 14 9 6
4 B 19 12 6
5 C 23 9 5
6 C 25 9 9
7 C 29 4 12

Example 1: Analyzing Only Numeric Columns (Default Behavior)

By default, the **describe()** function is intelligently designed to focus exclusively on **numeric columns** (i.e., those with integer or floating-point data types) within a Pandas DataFrame. This behavior ensures that the output metrics are statistically meaningful for quantitative data,

automatically excluding text or categorical columns.

When run without any arguments, the function provides key metrics such as the mean, standard deviation, minimum, and maximum values. These results are crucial for understanding the range and variability of your quantitative features, forming the backbone of quick exploratory data analysis.

#generate descriptive statistics for all numeric columns
df.describe()

points assists rebounds
count 8.000000 8.00000 8.000000
mean 20.250000 7.75000 8.375000
std 6.158618 2.54951 2.559994
min 12.000000 4.00000 5.000000
25% 14.750000 6.50000 6.000000
50% 21.000000 8.00000 8.500000
75% 25.000000 9.00000 10.250000
max 29.000000 12.00000 12.000000

The resulting table presents a comprehensive set of **descriptive statistics** for 'points', 'assists', and 'rebounds'. Specifically, it includes the total `count` of non-null observations, the `mean` (average), the measure of spread (`std`, or standard deviation), and the complete five-number summary (`min`, 25th percentile, `50%`/median, 75th percentile, and `max`).

A critical consideration when analyzing data is the presence of **missing values**. Pandas inherently handles data cleanliness; if null values (NaN) exist in any of the analyzed columns, the function will automatically exclude these values from the calculation of the descriptive statistics, ensuring that all reported figures are accurate based only on the available, non-missing data points.

Example 2: Generating Statistics for All Column Types

In many analytical scenarios, a comprehensive summary of the entire dataset is necessary, including both quantitative and qualitative variables. To achieve this level of detail, we must override the default behavior of the **describe()** function by explicitly using the `include='all'` argument. This parameter instructs the function to evaluate statistics for every column present in the DataFrame, regardless of whether it is numeric, object (string), or categorical.

Implementing `include='all'` results in a heterogeneous output matrix where the reported metrics adapt based on the data type of the column being analyzed. Numeric columns will display statistical measures like mean and standard deviation, while non-numeric columns will display

relevant categorical summaries.

#generate descriptive statistics for all columns
df.describe(include='all')

team points assists rebounds
count 8 8.000000 8.00000 8.000000
unique 3 NaN NaN NaN
top B NaN NaN NaN
freq 3 NaN NaN NaN
mean NaN 20.250000 7.75000 8.375000
std NaN 6.158618 2.54951 2.559994
min NaN 12.000000 4.00000 5.000000
25% NaN 14.750000 6.50000 6.000000
50% NaN 21.000000 8.00000 8.500000
75% NaN 25.000000 9.00000 10.250000
max NaN 29.000000 12.00000 12.000000

The expanded output successfully analyzes the categorical 'team' column alongside the numeric data. For the non-numeric data, **describe()** provides essential metrics such as the number of `unique` categories, the most frequently occurring category (`top`), and its corresponding `freq` (frequency). Conversely, for metrics that are nonsensical for a given data type (e.g., calculating the mean of team names), the function intelligently reports `NaN`, ensuring appropriate categorization of summary statistics.

Example 3: Describing Specific Columns or Subsets

During targeted data exploration, analysts frequently need to focus on the distribution characteristics of only one variable rather than summarizing the entire DataFrame. The **describe()** method can be applied directly to a specific column, which is technically represented as a Pandas Series after indexing the DataFrame using bracket notation (e.g., `df`). This provides a focused, single-variable statistical summary.

Applying the function to an individual Pandas Series, such as 'points', returns the standard eight descriptive statistics (count, mean, standard deviation, and the five-number summary) specific to that variable, which is highly effective for isolating and understanding its distribution.

#calculate descriptive statistics for 'points' column only
df.describe()

count 8.000000

mean 20.250000
std 6.158618
min 12.000000
25% 14.750000
50% 21.000000
75% 25.000000
max 29.000000
Name: points, dtype: float64

Furthermore, the method allows for calculating **descriptive statistics** for a defined subset of multiple columns simultaneously. This is achieved by passing a list of column names (e.g.,) to the DataFrame index before calling **describe()**. This capability streamlines the process of analyzing related variables without needing to create a temporary subset.

#calculate descriptive statistics for 'points' and 'assists' columns only
df].describe()

points assists
count 8.000000 8.00000
mean 20.250000 7.75000
std 6.158618 2.54951
min 12.000000 4.00000
25% 14.750000 6.50000
50% 21.000000 8.00000
75% 25.000000 9.00000
max 29.000000 12.00000

Conclusion: Leveraging the Power of describe()

The **describe()** function is an indispensable, high-efficiency tool for the initial stages of exploratory data analysis (EDA). By providing immediate, relevant insights into the central tendency, dispersion, and overall structure of your data, it enables analysts to quickly assess data quality, identify potential outliers, and understand variable distributions before moving on to more complex modeling.

For users seeking advanced functionality, such as specifying custom percentiles (beyond the default 25%, 50%, and 75%) or tailoring the inclusion/exclusion of specific data types, it is highly recommended to consult the official Pandas documentation for a detailed review of all available parameters.

Additional Resources for Pandas Functionality

The following tutorials explain how to perform other common data manipulation and analysis functions using the Pandas library: