

[🏠](#) > [Getting started](#) > [Package overview](#)

Package overview

pandas is a [Python](#) package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real-world** data analysis in Python. Additionally, it has the broader goal of becoming **the most powerful and flexible open source data analysis/manipulation tool available in any language**. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data need not be labeled at all



The two primary data structures of pandas, [Series](#) (1-dimensional) and [DataFrame](#) (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, [DataFrame](#) provides everything that R's [data.frame](#) provides and much more. pandas is built on top of [NumPy](#) and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

[Back to top](#)

Here are just a few of the things that pandas does well:

- Easy handling of **missing data** (represented as NaN) in floating point as well as non-floating point data

[Skip to main content](#)

- Size mutability: columns can be **inserted and deleted** from DataFrame and higher dimensional objects
- Automatic and explicit **data alignment**: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let `Series`, `DataFrame`, etc. automatically align the data for you in computations
- Powerful, flexible **group by** functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data
- Make it **easy to convert** ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects
- Intelligent label-based **slicing**, **fancy indexing**, and **subsetting** of large data sets
- Intuitive **merging** and **joining** data sets
- Flexible **reshaping** and pivoting of data sets
- **Hierarchical** labeling of axes (possible to have multiple labels per tick)
- Robust IO tools for loading data from **flat files** (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast **HDF5 format**
- **Time series**-specific functionality: date range generation and frequency conversion, moving window statistics, date shifting, and lagging.

Many of these principles are here to address the shortcomings frequently experienced using other languages / scientific research environments. For data scientists, working with data is typically divided into multiple stages: munging and cleaning data, analyzing / modeling it, then organizing the results of the analysis into a form suitable for plotting or tabular display. pandas is the ideal tool for all of these tasks.

Some other notes

- pandas is **fast**. Many of the low-level algorithmic bits have been extensively tweaked in [Cython](#) code. However, as with anything else generalization usually sacrifices performance. So if you focus on one feature for your application you may be able to create a faster specialized tool.
- pandas is a dependency of [statsmodels](#), making it an important part of the statistical computing ecosystem in Python.
- pandas has been used extensively in production in financial applications.

[Skip to main content](#)

Data structures

Dimensions	Name	Description
1	Series	1D labeled homogeneously-typed array
2	DataFrame	General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed column

Why more than one data structure?

The best way to think about the pandas data structures is as flexible containers for lower dimensional data. For example, `DataFrame` is a container for `Series`, and `Series` is a container for scalars. We would like to be able to insert and remove objects from these containers in a dictionary-like fashion.

Also, we would like sensible default behaviors for the common API functions which take into account the typical orientation of time series and cross-sectional data sets. When using the N-dimensional array (`ndarrays`) to store 2- and 3-dimensional data, a burden is placed on the user to consider the orientation of the data set when writing functions; axes are considered more or less equivalent (except when C- or Fortran-contiguity matters for performance). In pandas, the axes are intended to lend more semantic meaning to the data; i.e., for a particular data set, there is likely to be a “right” way to orient the data. The goal, then, is to reduce the amount of mental effort required to code up data transformations in downstream functions.

For example, with tabular data (`DataFrame`) it is more semantically helpful to think of the **index** (the rows) and the **columns** rather than axis 0 and axis 1. Iterating through the columns of the `DataFrame` thus results in more readable code:

```
for col in df.columns:
    series = df[col]
    # do something with series
```

Mutability and copying of data

[Skip to main content](#)

All pandas data structures are value-mutable (the values they contain can be altered) but not always size-mutable. The length of a Series cannot be changed, but, for example, columns can be inserted into a DataFrame. However, the vast majority of methods produce new objects and leave the input data untouched. In general we like to **favor immutability** where sensible.

Getting support

The first stop for pandas issues and ideas is the [GitHub Issue Tracker](#). If you have a general question, pandas community experts can answer through [Stack Overflow](#).

Community

pandas is actively supported today by a community of like-minded individuals around the world who contribute their valuable time and energy to help make open source pandas possible. Thank to [all of our contributors](#).

If you're interested in contributing, please visit the [contributing guide](#).

pandas is a [NumFOCUS](#) sponsored project. This will help ensure the success of the development of pandas as a world-class open-source project and makes it possible to [donate](#) to the project.

Project governance

The governance process that pandas project has used informally since its inception in 2008 is formalized in [Project Governance documents](#). The documents clarify how decisions are made and how the various elements of our community interact, including the relationship between open source collaborative development and work that may be funded by for-profit or non-profit entities.

Wes McKinney is the Benevolent Dictator for Life (BDFL).

Development team

The list of the Core Team members and more detailed information can be found on the [pandas website](#).

[Skip to main content](#)

Institutional partners

The information about current institutional partners can be found on [pandas website page](#).

License

BSD 3-Clause License

Copyright (c) 2008–2011, AQR Capital Management, LLC, Lambda Foundry, Inc. and Py
All rights reserved.

Copyright (c) 2011–2023, Open source contributors.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this
list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.
- * Neither the name of the copyright holder nor the names of its
contributors may be used to endorse or promote products derived from
this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

< Previous
[Installation](#)

Next >
[Getting started tutorials](#)

© 2024, pandas via [NumFOCUS, Inc.](#) Hosted by
[OVHcloud](#).

Built with the [PyData Sphinx Theme](#)

0.14.4.

Created using [Sphinx](#) 7.2.6.