# Data Manipulation with Pandas

# 使用Pandas处理数据

> In the previous chapter, we dove into detail on NumPy and its `ndarray` object, which provides efficient storage and manipulation of dense typed arrays in Python. Here we'll build on this knowledge by looking in detail at the data structures provided by the Pandas library. Pandas is a newer package built on top of NumPy, and provides an efficient implementation of a `DataFrame`. `DataFrame`s are essentially multidimensional arrays with attached row and column labels, and often with heterogeneous types and/or missing data. As well as offering a convenient storage interface for labeled data, Pandas implements a number of powerful data operations familiar to users of both database frameworks and spreadsheet programs.

在上一章中，我们深入介绍了NumPy和它的`ndarray`对象，它被用来在Python存储和操作非稀疏的数组数据。以此为基础，本章将要详细介绍Pandas库为我们提供数据结构。Pandas是一个在NumPy的基础上创建的第三方库，它提供了对于`DataFrame`对象的有效支持。`DataFrame`是一个多维的数组，其行和列都有标签，通常列之间都含有不同种类的数据类型或者有缺失的数据。除了提供了对于标签数据存储的支持之外，Pandas还实现了数量众多的数据操作方法，这些方法无论对于数据库的用户还是对于工作表单用户而言都非常熟悉。

> As we saw, NumPy's `ndarray` data structure provides essential features for the type of clean, well-organized data typically seen in numerical computing tasks. While it serves this purpose very well, its limitations become clear when we need more flexibility (e.g., attaching labels to data, working with missing data, etc.) and when attempting operations that do not map well to element-wise broadcasting (e.g., groupings, pivots, etc.), each of which is an important piece of analyzing the less structured data available in many forms in the world around us. Pandas, and in particular its `Series` and `DataFrame` objects, builds on the NumPy array structure and provides efficient access to these sorts of "data munging" tasks that occupy much of a data scientist's time.

正如我们前面看到的，NumPy的`ndarray`数据结构能为数值计算任务所需要的数据提供必不可少的功能。虽然`ndarray`的功能已经很强大，但是当我们需要更多的灵活性的时候，它的缺陷就体现了出来（例如，为数据提供标签，处理缺失的数据等）。而且如果当需要对数据进行超过广播能处理范畴的操作时（例如分组，数据透视等），NumPy就无能为力了。而上述提到的这些能力对于我们处理真实世界中产生的非严格格式化数据来说是非常重要的。Pandas，或者更具体的来说，它的`Series`和`DataFrame`对象，在NumPy的基础上提供了上述操作，让数据科学家能从花很多时间的这种乏味的数据处理工作中解脱出来。

> In this chapter, we will focus on the mechanics of using `Series`, `DataFrame`, and related structures effectively. We will use examples drawn from real datasets where appropriate, but these examples are not necessarily the focus.

我们在本章中会聚焦于了解`Series`、`DataFrame`和相关结构的机制上。例子中使用了真实的数据集进行说明，以方便理解，但是并不需要特别关注例子数据本身。

## Installing and Using Pandas

## 安装和使用Pandas

> Installation of Pandas on your system requires NumPy to be installed, and if building the library from source, requires the appropriate tools to compile the C and Cython sources on which Pandas is built. Details on this installation can be found in the Pandas documentation. If you followed the advice outlined in the Preface and used the Anaconda stack, you already have Pandas installed.

在你的系统上安装Pandas必要先安装NumPy，如果选择从源码进行安装，还需要能够编译C和Cython的工具，因为Pandas源码是使用这两种语言编写的。详细的安装文档可以访问Pandas在线文档。如果你是依照序言中的方法使用Anaconda安装的环境，那么Pandas已经安装好了。

> Once Pandas is installed, you can import it and check the version:

安装后，你可以载入包并检查版本信息，验证安装是否成功：

```
In [1]: import pandas
        pandas.__version__
Out[1]: '0.24.2'
```

> Just as we generally import NumPy under the alias `np`, we will import Pandas under the alias `pd`:

就像我们管理将NumPy载入并命名为`np`一样，我们也惯例将Pandas载入并命名为`pd`：

```
In [2]: import pandas as pd
```

> This import convention will be used throughout the remainder of this book.

这个惯例会贯穿本书后续所有内容。

## Reminder about Built-In Documentation

## 內建帮助及文档的提醒

> As you read through this chapter, don't forget that IPython gives you the ability to quickly explore the contents of a package (by using the tab-completion feature) as well as the documentation of various functions (using the `?` character). (Refer back to Help and Documentation in IPython if you need a refresher on this.)

当你阅读本章的时候，不要忘记了IPython提供了快速查看对象内容（使用tab自动补全）和帮助文档（使用`?`语句）的工具。（参见IPython的帮助和文档）

> For example, to display all the contents of the pandas namespace, you can type

例如，要查看pandas命名空间中的所有内容，你可以输入

```
In [3]: pd.<TAB>
```

> And to display Pandas's built-in documentation, you can use this:

要列示Pandas的内建文件，你可以输入

```
In [4]: pd?
```

> More detailed documentation, along with tutorials and other resources, can be found at http://pandas.pydata.org/.

更详细的文档，包括教程和其他资源，可以访问http://pandas.pydata.org/。