

---

# PREDICT 420

Atef Bader, PhD

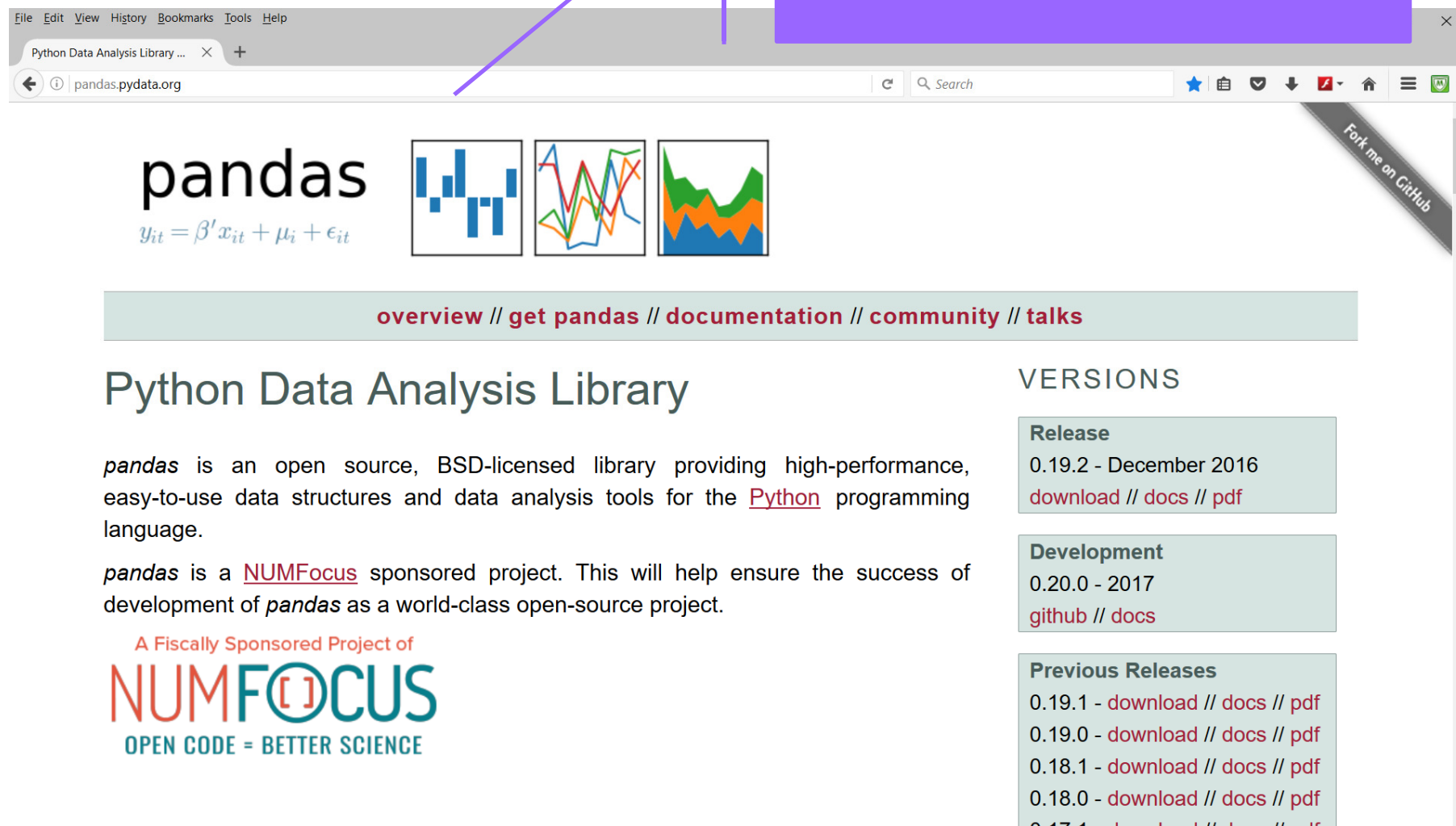
# Agenda

---

- Pandas Data Analysis Library
- A-MUST readings for DataFrame
- Exercise #2 Walkthrough & Deliverable

# Pandas – Data Analysis Library

Pandas is a library of packages for Data Analysis in Python



The screenshot shows the pandas website in a web browser. The browser's address bar displays 'pandas.pydata.org'. The website header features the 'pandas' logo with the equation  $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$  below it, followed by three small charts: a bar chart, a line chart, and a stacked area chart. A navigation bar contains links: 'overview // get pandas // documentation // community // talks'. The main heading is 'Python Data Analysis Library'. The introductory text states: 'pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.' Below this, it mentions 'pandas is a NUMFocus sponsored project. This will help ensure the success of development of pandas as a world-class open-source project.' The NUMFocus logo is shown with the tagline 'OPEN CODE = BETTER SCIENCE'. On the right, a 'VERSIONS' section lists: 'Release 0.19.2 - December 2016' with links for 'download // docs // pdf'; 'Development 0.20.0 - 2017' with links for 'github // docs'; and 'Previous Releases' listing versions 0.19.1, 0.19.0, 0.18.1, 0.18.0, and 0.17.1, each with links for 'download // docs // pdf'.

File Edit View History Bookmarks Tools Help

Python Data Analysis Library ... X +

← pandas.pydata.org Search

**pandas**  
 $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

overview // get pandas // documentation // community // talks

## Python Data Analysis Library

*pandas* is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the [Python](#) programming language.

*pandas* is a [NUMFocus](#) sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source project.

A Fiscally Sponsored Project of

**NUMFOCUS**  
OPEN CODE = BETTER SCIENCE

### VERSIONS

**Release**  
0.19.2 - December 2016  
[download](#) // [docs](#) // [pdf](#)

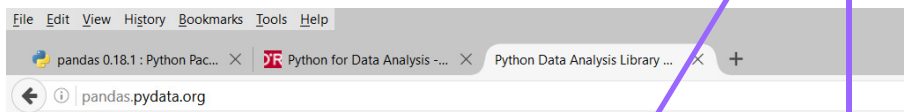
**Development**  
0.20.0 - 2017  
[github](#) // [docs](#)

**Previous Releases**  
0.19.1 - [download](#) // [docs](#) // [pdf](#)  
0.19.0 - [download](#) // [docs](#) // [pdf](#)  
0.18.1 - [download](#) // [docs](#) // [pdf](#)  
0.18.0 - [download](#) // [docs](#) // [pdf](#)  
0.17.1 - [download](#) // [docs](#) // [pdf](#)

# Pandas – Data Analysis Library

Here is the main theme behind pandas package.

That is, to do data munging, preparation, modeling, and analysis without the need to switch to R language



## What problem does *pandas* solve?

Python has long been great for data munging and preparation, but less so for data analysis and modeling. *pandas* helps fill this gap, enabling you to carry out your entire data analysis workflow in Python without having to switch to a more domain specific language like R.

Combined with the excellent [IPython](#) toolkit and other libraries, the environment for doing data analysis in Python excels in performance, productivity, and the ability to collaborate.

*pandas* does not implement significant modeling functionality outside of linear and panel regression; for this, look to [statsmodels](#) and [scikit-learn](#). More work is still needed to make Python a first class statistical modeling environment, but we are well on our way toward that goal.

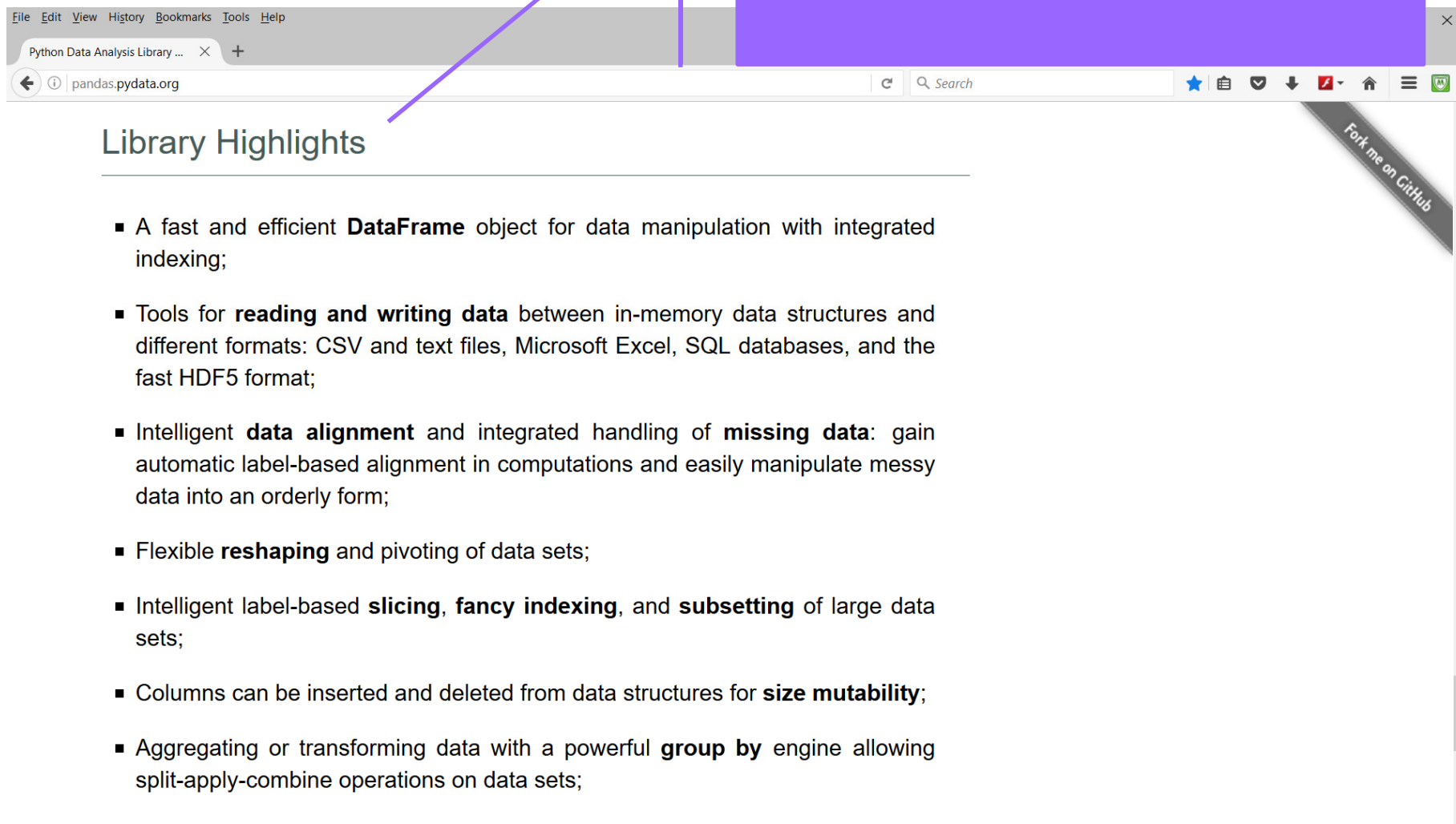
## What do our users have to say?

Roni Israelov, PhD  
Portfolio Manager

AOR | CAPITAL  
MANAGEMENT

# Pandas – Data Analysis Library

All you need to work on for a given dataset:  
slice/dice/index/groupby/CSV/SQL/NoSQL

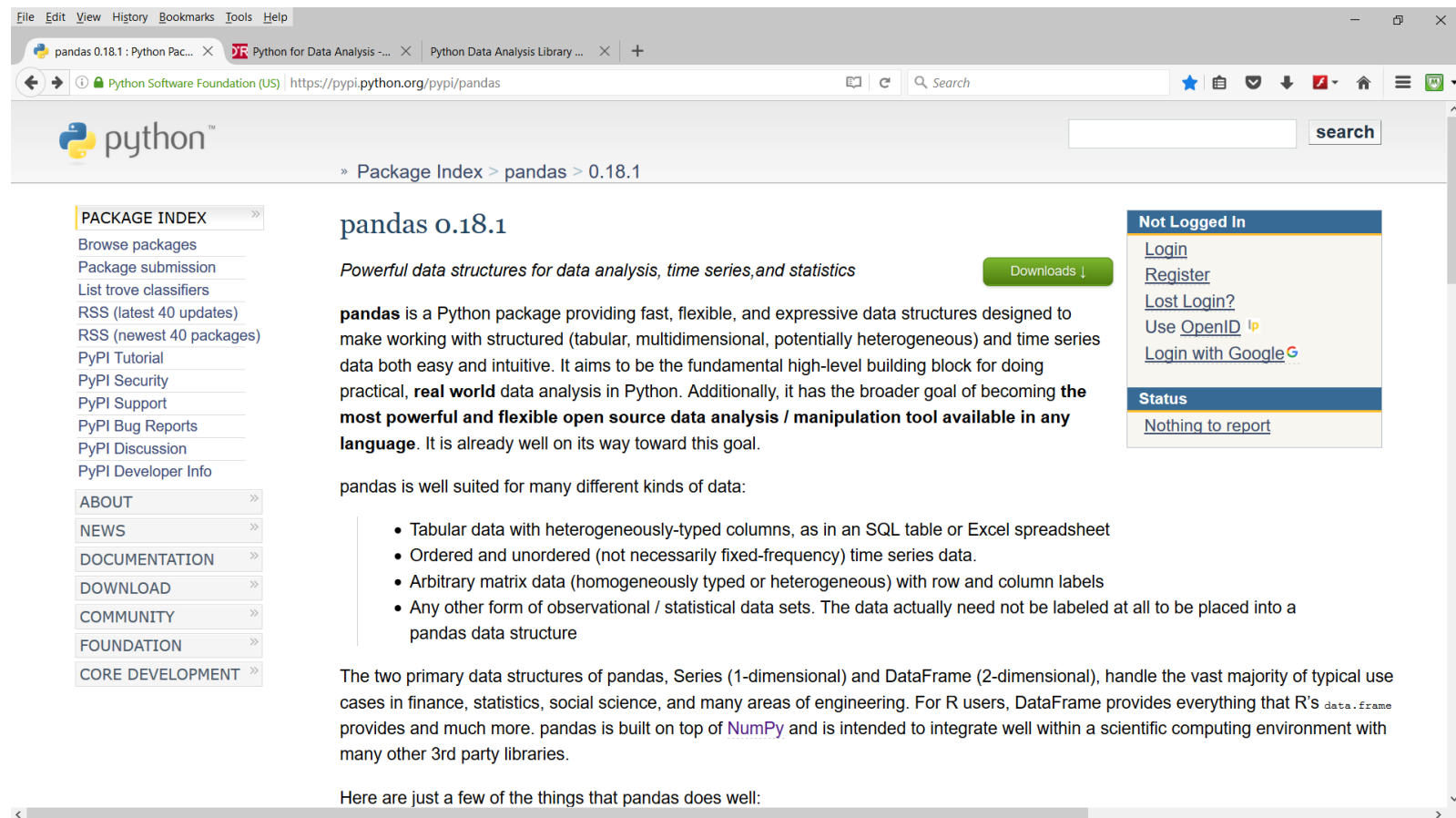


The screenshot shows the pandas.pydata.org website. A purple callout box with white text is positioned in the upper right, listing operations: slice/dice/index/groupby/CSV/SQL/NoSQL. A purple line originates from the bottom left of this box and points to the 'Library Highlights' section on the website. The website's browser window shows the URL 'pandas.pydata.org' and a search bar. The 'Library Highlights' section contains a list of features.

## Library Highlights

- A fast and efficient **DataFrame** object for data manipulation with integrated indexing;
- Tools for **reading and writing data** between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format;
- Intelligent **data alignment** and integrated handling of **missing data**: gain automatic label-based alignment in computations and easily manipulate messy data into an orderly form;
- Flexible **reshaping** and pivoting of data sets;
- Intelligent label-based **slicing, fancy indexing**, and **subsetting** of large data sets;
- Columns can be inserted and deleted from data structures for **size mutability**;
- Aggregating or transforming data with a powerful **group by** engine allowing split-apply-combine operations on data sets;

# DataFrame – Required Reading



The screenshot shows the PyPI page for pandas 0.18.1. The browser address bar shows the URL <https://pypi.python.org/pypi/pandas>. The page features a sidebar with navigation links, a main content area with the package title and description, and a right sidebar with login and status information.

**PACKAGE INDEX** »

- [Browse packages](#)
- [Package submission](#)
- [List trove classifiers](#)
- [RSS \(latest 40 updates\)](#)
- [RSS \(newest 40 packages\)](#)
- [PyPI Tutorial](#)
- [PyPI Security](#)
- [PyPI Support](#)
- [PyPI Bug Reports](#)
- [PyPI Discussion](#)
- [PyPI Developer Info](#)

**ABOUT** »

**NEWS** »

**DOCUMENTATION** »

**DOWNLOAD** »

**COMMUNITY** »

**FOUNDATION** »

**CORE DEVELOPMENT** »

## pandas 0.18.1

*Powerful data structures for data analysis, time series, and statistics*

[Downloads ↓](#)

**pandas** is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series 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 actually need not be labeled at all to be placed into a pandas data structure

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.

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

**Not Logged In**

- [Login](#)
- [Register](#)
- [Lost Login?](#)
- [Use OpenID](#) 
- [Login with Google](#) 

**Status**

- [Nothing to report](#)


# DataFrame – Required Reading

File Edit View History Bookmarks Tools Help

pandas 0.18.1 : Python Pac... Python for Data Analysis ~... Python Data Analysis Library ... +

shop.oreilly.com/product/0636920023784.do Search

**Search Inside and Read** Python for Data Analysis  
Data Wrangling with Pandas, NumPy, and IPython  
By [Wes McKinney](#)  
Publisher: O'Reilly Media  
Final Release Date: October 2012  
Pages: 466  
★ ★ ★ ★ ★ 3.9  
[Read 27 Reviews](#) | [Write a Review](#)

  
O'REILLY™ Wes McKinney  
[Larger Cover](#) [Full description](#)

*Python for Data Analysis* is concerned with the nuts and bolts of manipulating, processing, cleaning, and crunching data in Python. It is also a practical, modern introduction to scientific computing in Python, tailored for data-intensive applications. This is a book about the parts of the Python language and libraries you'll...

**Buying Options**

**Immediate Access - Go Digital** what's this?  
Ebook: **\$33.99** [Add to Cart](#)  
Formats: DAISY, ePub, Mobi, PDF

Print & Ebook: **\$43.99** [Add to Cart](#)

Print: **\$39.99**  [Add to Cart](#)

**Safari Books Online** - Read now >

**Essential Links**

- [Download Example Code](#)
- [Register Your Book](#)
- [View/Submit Errata](#)
- [Media Praise](#)
- [Ask a Question](#)
- [Bulk Discounts & Licensing](#)

**Table of Contents** | **Product Details** | **About the Author** | **Colophon**

**Chapter 1 : Preliminaries**

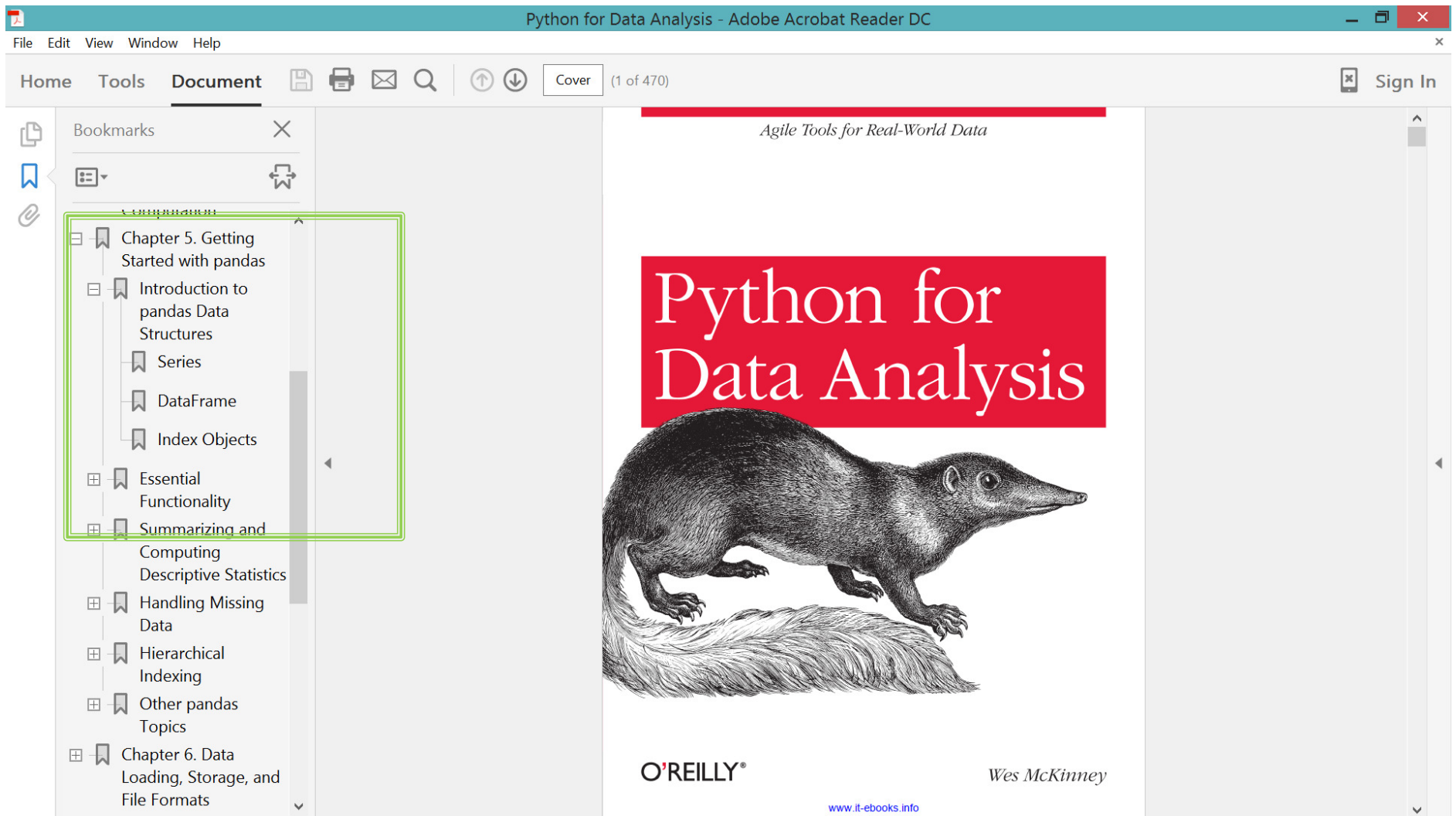
- What Is This Book About?
- Why Python for Data Analysis?
- Essential Python Libraries
- Installation and Setup
- Community and Conferences
- Navigating This Book
- Acknowledgements

**Chapter 2 : Introductory Examples**

- 1.usa.gov data from bit.ly
- MovieLens 1M Data Set

**Recommended for You**

# DataFrame – Required Reading





# DataFrame – API

The screenshot shows the pandas.DataFrame API documentation page. The browser window has a teal header with menu items: File, Edit, View, History, Bookmarks, Tools, Help. The address bar shows the URL: pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html. Below the address bar is a green navigation bar with the text "pandas 0.16.2 documentation » API Reference »" and links for "previous | next | modules | index". On the left is a "Table Of Contents" sidebar with links like "What's New", "Installation", "Contributing to pandas", "Frequently Asked Questions (FAQ)", "Package overview", "10 Minutes to pandas", "Tutorials", "Cookbook", "Intro to Data Structures", "Essential Basic", "Functionality", "Working with Text Data", "Options and Settings", "Indexing and Selecting Data", "MultiIndex / Advanced", "Indexing", "Computational tools", "Working with missing data", "Group By: split-apply-combine", "Merge, join, and concatenate", and "Reshaping and Pivot Tables". The main content area has a blue header "pandas.DataFrame" and a class definition: `class pandas.DataFrame(data=None, index=None, columns=None, dtype=None, copy=False)`. Below this is a description: "Two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). Arithmetic operations align on both row and column labels. Can be thought of as a dict-like container for Series objects. The primary pandas data structure". To the right of the description is a "Parameters:" section with details for `data`, `index`, and `columns`.

File Edit View History Bookmarks Tools Help

pandas.DataFrame — pandas 0... x +

pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html

Search

pandas 0.16.2 documentation » API Reference » previous | next | modules | index

## Table Of Contents

- What's New
- Installation
- Contributing to pandas
- Frequently Asked Questions (FAQ)
- Package overview
- 10 Minutes to pandas
- Tutorials
- Cookbook
- Intro to Data Structures
- Essential Basic
- Functionality
- Working with Text Data
- Options and Settings
- Indexing and Selecting Data
- MultiIndex / Advanced
- Indexing
- Computational tools
- Working with missing data
- Group By: split-apply-combine
- Merge, join, and concatenate
- Reshaping and Pivot Tables

## pandas.DataFrame

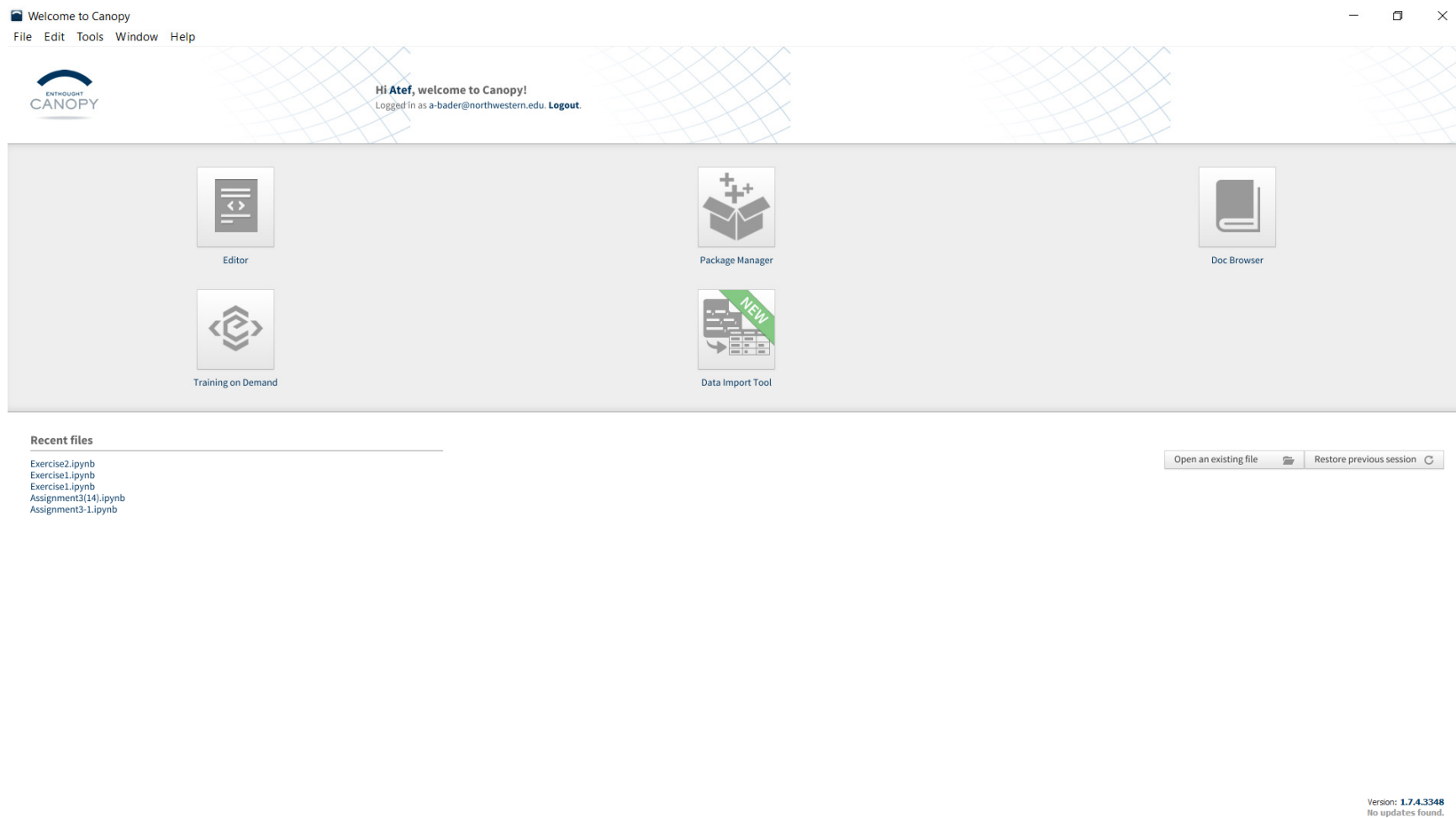
`class pandas.DataFrame(data=None, index=None, columns=None, dtype=None, copy=False)`

Two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). Arithmetic operations align on both row and column labels. Can be thought of as a dict-like container for Series objects. The primary pandas data structure

**Parameters:**


- data** : *numpy ndarray (structured or homogeneous), dict, or DataFrame*  
Dict can contain Series, arrays, constants, or list-like objects
- index** : *Index or array-like*  
Index to use for resulting frame. Will default to `np.arange(n)` if no indexing information part of input data and no index provided
- columns** : *Index or array-like*  
Column labels to use for resulting frame. Will default to `np.arange(n)` if no column labels are provided

# Package Manager



# Package Manager – Installed Packages

Package Manager - Canopy

 **Package Manager**  
Install, update or remove your Python packages


Refresh

 • Logged in as **a-bader@northwestern.edu**








Installed **107**


Available 525

Updates **42**

History 

Settings

Package Name	Installed Version
 matplotlib	1.5.1-8
mccabe	0.3.1-1
memory_profiler	0.41-1
mistune	0.7.1-1
mkl	2017.0.1-1
 mkl_service	1.0-4
mpmath	0.19-1
 nbconvert	4.2.0-1
 nbformat	4.0.1-5
nose	1.3.7-2
 notebook	4.2.1-2
 numpy	1.10.4-4
 pandas	0.19.0-1
path.py	8.1.1-3
pep8	1.7.0-1
pexpect	2.4-1
pickleshare	0.5-1

**pandas** Update available! 

data analysis library

Installed: 0.19.0-1

Available on store: (enthought/free) 

0.19.0-2


Uninstall

Upgrade to v0.19.0-2

pandas is a python package providing convenient data structures for time series, cross-sectional, or any other form of "labeled" data, with tools for building statistical and econometric models.

# Package Manager – Available Packages

Package Manager - Canopy

 **Package Manager**  
Install, update or remove your Python packages

[Refresh](#) • Logged in as a-bader@northwestern.edu

Installed107

Available525

Updates42

History

Settings

Package Name	Latest Available Version
7z	9.20-1
affine	2.0.0.post1-1
agw	0.9.1-1
alabaster	0.7.9-1
amqp	1.4.5-1
aniso8601	0.92-1
ansi2html	1.1.1-1
anyjson	0.3.3-4
apipkg	1.4-1
✓ appinst	2.1.5-1
✓ apptools	4.4.0-4
arch	3.0-18
argcomplete	1.0.0-1
argparse	1.4.0-1
astropy	1.3-1
✓ atom	0.3.10-1

**anyjson**

No summary available.

Installed:

Available on store:  
(enthought/free)

Currently not installed.

0.3.3-4

Install v0.3.3-4

No description available.

# Exercise #2

File Edit View History Bookmarks Tools Help

exercise2

localhost:8890/notebooks/bader/nu/420/spring 2016/sync sessions/sync session 2/exerciseppractice2/exercise2

Jupyter exercise2 Last Checkpoint: 04/05/2015 (autosaved)

File Edit View Insert Cell Kernel Help Python 2

Week #2 - Python Practice

Some Ins and Outs with Python

There are many different kinds of data to be managed and analyzed today, and there are many ways to do it using Python. Being able to manage and modify data isn't useful unless you can also get data into Python, and save your results from it. Beginning with this session we're going to review techniques for input and output from Python starting with the simplest file formats and some basic Python tools. We'll also take a first look at the pandas package. Pandas has become very popular amongst "Pythonic" data scientists and is being used at the largest of the big data firms. In the sessions that follow we'll consider more complicated file types and data "munging" tools and techniques.

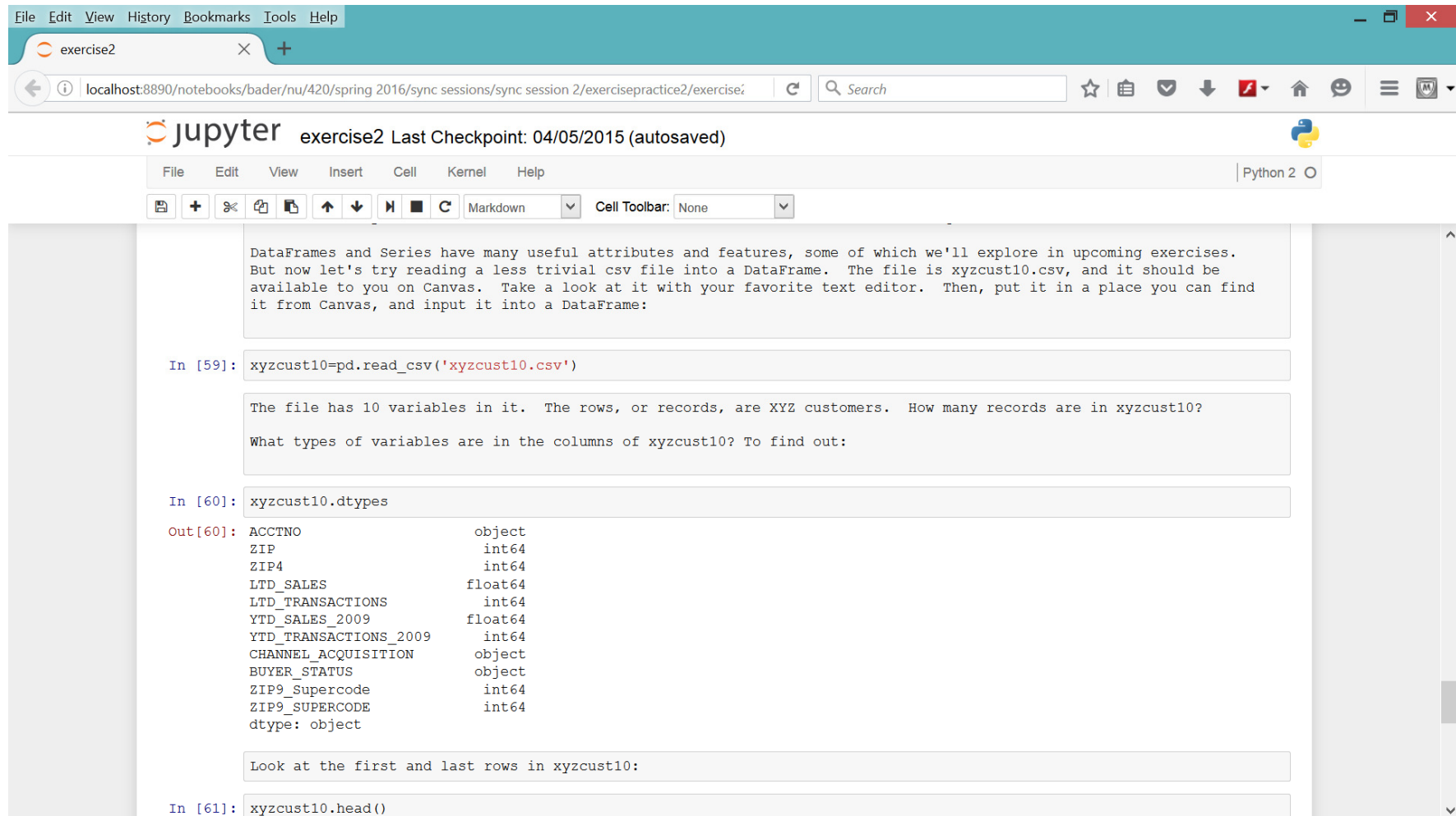
Time (and the term) is a'wasting, so lets crank up your Canopy environment and get to work with some examples. In what follows it's assumed that you are using the Canopy "Editor," which includes a version of IPython, and so the command prompts will look like In '[n]:' without the single quotes. What "n" is depends on where you are in the current session. As I'm sure you realize by now since you have most likely use Canopy or IPython before, for many "Ins" in IPython there is an "Out" of some kind which is a result of the "In" that precedes it.

So, let's start with flat files. A flat file is just a file that's, well, flat. It's typically a string of characters that may include end of line markers like a newline or carriage return code. Let's write a simple flat file out to disk by entering the following command:

```
In [32]: outfile = open('myflatfile.txt','w') # open to write to a text file
```

Note: "In [1]:" represents the command prompt in your IPython session. Depending on what you've been doing in a session, the digit or digits you see in it will vary. But you knew that, right?

# Exercise #2



The screenshot displays a Jupyter Notebook titled "exercise2" in a web browser. The browser's address bar shows the URL: `localhost:8890/notebooks/bader/nu/420/spring 2016/sync sessions/sync session 2/exercisepactice2/exercise/`. The Jupyter interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for file operations and cell execution. The notebook content consists of several cells:

- A text cell explaining that DataFrames and Series have many useful attributes and features, and that the task is to read a CSV file named `xyzcust10.csv` into a DataFrame.
- A code cell (In [59]) containing the command: `xyzcust10=pd.read_csv('xyzcust10.csv')`.
- A text cell asking for the number of records and variable types in the DataFrame.
- A code cell (In [60]) containing the command: `xyzcust10.dtypes`.
- An output cell (Out [60]) showing the data types for each column in the DataFrame:

Column Name	Data Type
ACCTNO	object
ZIP	int64
ZIP4	int64
LTD_SALES	float64
LTD_TRANSACTIONS	int64
YTD_SALES_2009	float64
YTD_TRANSACTIONS_2009	int64
CHANNEL_ACQUISITION	object
BUYER_STATUS	object
ZIP9_Supercode	int64
ZIP9_SUPERCODE	int64
dtype:	object

- A text cell asking to look at the first and last rows of the DataFrame.
- A code cell (In [61]) containing the command: `xyzcust10.head()`.

# Exercise #2

File Edit View History Bookmarks Tools Help

exercise2

localhost:8890/notebooks/bader/nu/420/spring 2016/sync sessions/sync session 2/exercisepactice2/exercise2

jupyter exercise2 Last Checkpoint: 04/05/2015 (autosaved)

Python 2

File Edit View Insert Cell Kernel Help

Cell Toolbar: None

```
CHANNEL_ACQUISITION    object
BUYER_STATUS            object
ZIP9_Supercode          int64
ZIP9_SUPERCODE          int64
dtype: object
```

Look at the first and last rows in xyzcust10:

```
In [61]: xyzcust10.head()
```

```
Out[61]:
```

	ACCTNO	ZIP	ZIP4	LTD_SALES	LTD_TRANSACTIONS	YTD_SALES_2009	YTD_TRANSACTIONS_2009	CHANNEL_ACQUISITION	BUYER_STATUS
0	WDQQLDQL	60084	5016	90	1	0	0	IB	INAC
1	WQWAYHYLA	60091	1750	4227	9	1263	3	RT	ACTI
2	GSHAPLHAW	60067	900	420	3	129	1	RT	ACTI
3	PGGYDYWAD	60068	3838	6552	6	0	0	RT	INAC
4	LWPSGPLLS	60090	3932	189	3	72	1	RT	ACTI

```
In [62]: xyzcust10.tail()
```

```
Out[62]:
```

	ACCTNO	ZIP	ZIP4	LTD_SALES	LTD_TRANSACTIONS	YTD_SALES_2009	YTD_TRANSACTIONS_2009	CHANNEL_ACQUISITION	BUYER_STATUS
30466	SYDQYLSWH	60098	3951	2736	9	96	1	RT	ACTI
30467	SAPDQHQLP	60098	9681	2412	8	108	1	RT	ACTI
30468	SASYAPDSY	60098	0	429	1	0	0	RT	ACTI
30469	PWQPDWHA	60098	7927	651	1	0	0	RT	ACTI
30470	SOQHDYHWH	60098	4160	4527	16	672	2	RT	ACTI

## What you need to submit for Exercise #2 on Canvas?

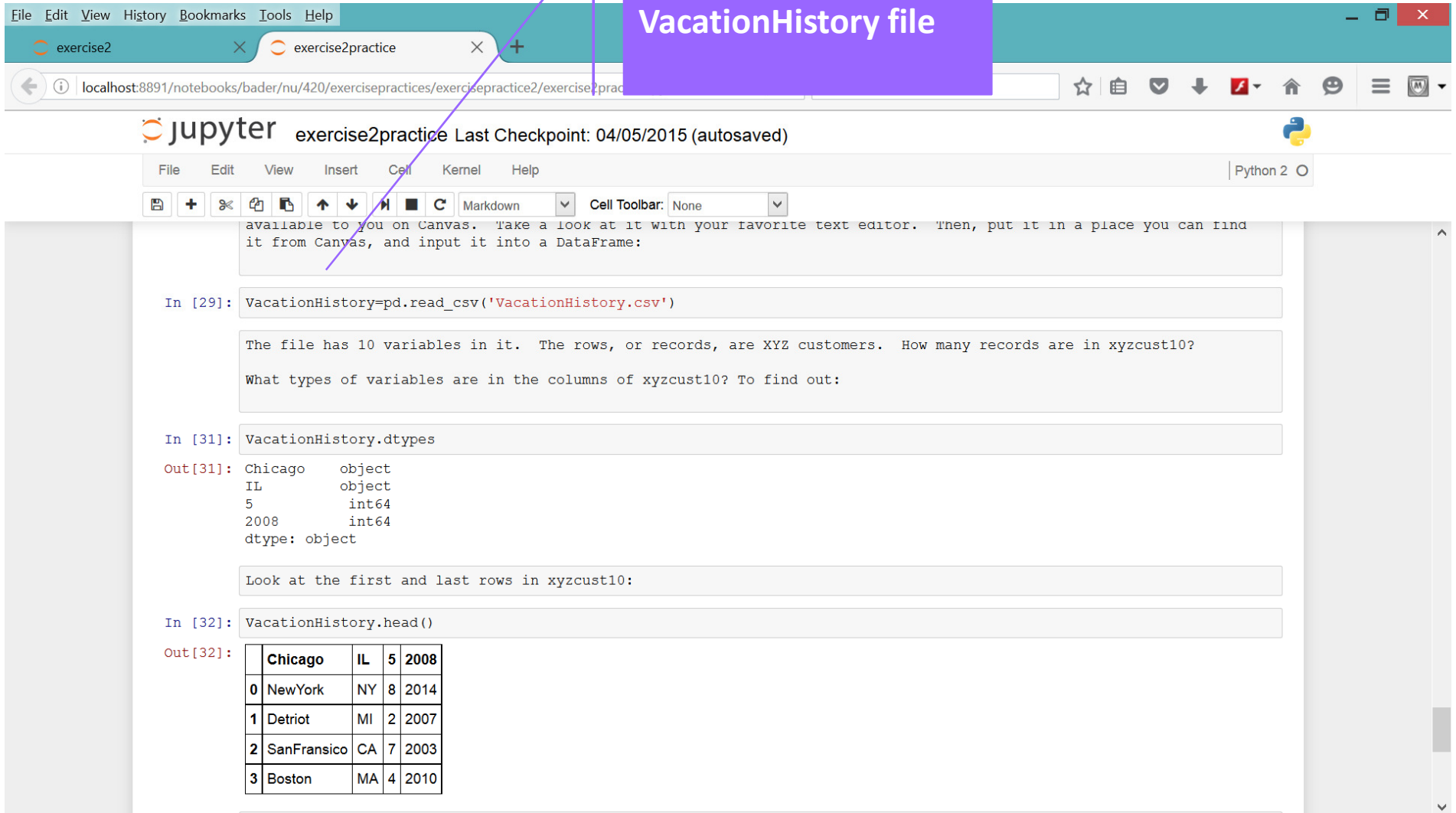
---

1. Change the Ipython Notebook script to use the VacationHistory.csv file instead of xyzcust10.csv file
2. Run the IPython Notebook Script Again
3. Save your updated IPython Notebook Script along with the OUTPUT for the cells
4. Submit your updated IPython Notebook Script on Canvas



# Exercise #2

Add your  
VacationHistory file

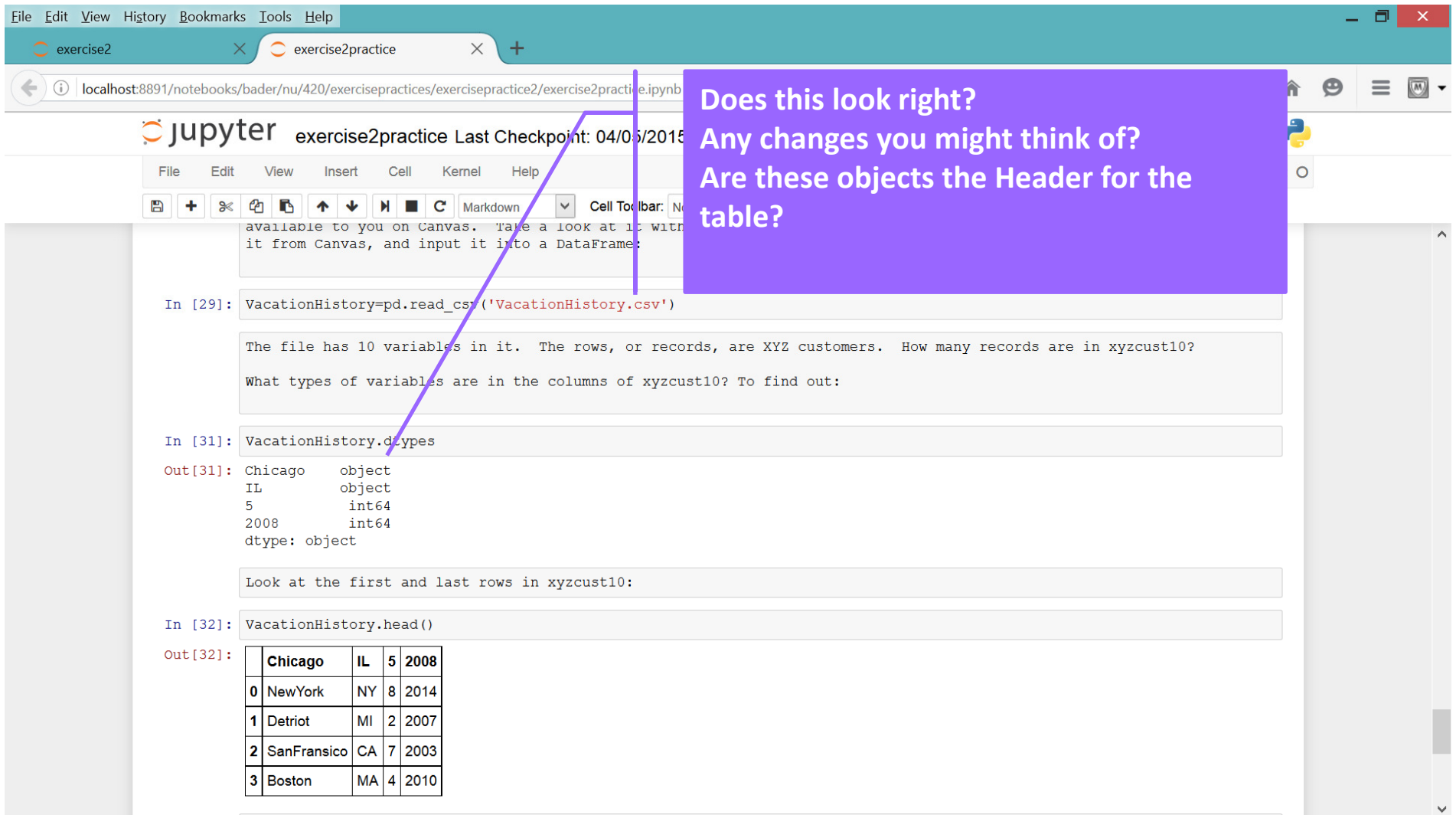


The screenshot shows a Jupyter Notebook interface in a web browser. The browser's address bar shows the URL: `localhost:8891/notebooks/bader/nu/420/exercisepactices/exercisepactice2/exercise2prac`. The Jupyter Notebook title bar indicates the file is named `exercise2practice` and shows the last checkpoint as `04/05/2015 (autosaved)`. The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for saving, adding cells, and running code. The notebook content consists of several cells:

- A text cell containing instructions: "available to you on Canvas. Take a look at it with your favorite text editor. Then, put it in a place you can find it from Canvas, and input it into a DataFrame:".
- A code cell (In [29]) with the command: `VacationHistory=pd.read_csv('VacationHistory.csv')`.
- A text cell asking: "The file has 10 variables in it. The rows, or records, are XYZ customers. How many records are in xyzcust10? What types of variables are in the columns of xyzcust10? To find out:".
- A code cell (In [31]) with the command: `VacationHistory.dtypes`.
- An output cell (Out [31]) showing the data types for the first four columns: `Chicago object`, `IL object`, `5 int64`, and `2008 int64`, with a summary `dtype: object`.
- A text cell asking: "Look at the first and last rows in xyzcust10:".
- A code cell (In [32]) with the command: `VacationHistory.head()`.
- An output cell (Out [32]) displaying a table of the first four rows of the data.

	Chicago	IL	5	2008
0	NewYork	NY	8	2014
1	Detriot	MI	2	2007
2	SanFransico	CA	7	2003
3	Boston	MA	4	2010

# Exercise #2



The screenshot shows a Jupyter Notebook interface with a browser window at the top. The notebook has two tabs: 'exercise2' and 'exercise2practice'. The active tab is 'exercise2practice', showing a Jupyter Notebook with a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar. The notebook content includes a text cell with instructions, two code cells, and their outputs.

**Code Cell 1:**

```
In [29]: VacationHistory=pd.read_csv('VacationHistory.csv')
```

**Output 1:**

The file has 10 variables in it. The rows, or records, are XYZ customers. How many records are in xyzcust10?  
What types of variables are in the columns of xyzcust10? To find out:

**Code Cell 2:**

```
In [31]: VacationHistory.dtypes
```

**Output 2:**

```
Out[31]: Chicago    object
IL              object
5              int64
2008            int64
dtype: object
```

**Text Cell:**

Look at the first and last rows in xyzcust10:

**Code Cell 3:**

```
In [32]: VacationHistory.head()
```

**Output 3:**

	Chicago	IL	5	2008
0	NewYork	NY	8	2014
1	Detriot	MI	2	2007
2	SanFransico	CA	7	2003
3	Boston	MA	4	2010

A purple callout box with white text is overlaid on the right side of the notebook, containing the following questions:

- Does this look right?
- Any changes you might think of?
- Are these objects the Header for the table?

# Exercise #2

The screenshot shows a Jupyter Notebook window with the following content:

File Edit View History Bookmarks Tools Help

localhost:8891/notebooks/bader/nu/420/exercisepactices/exercisepactice2/exercise2practice.ipynb

jupyter exercise2practice Last Checkpoint: 04/05/2015 (autosaved)

File Edit View Insert Cell Kernel Help Python 2

Markdown Cell Toolbar: None

available to you on Canvas. Take a look at it with your favorite text editor. Then, put it in a place you can find it from Canvas, and input it into a DataFrame:

```
In [29]: VacationHistory=pd.read_csv('VacationHisto
```

The file has 10 variables in it. The rows  
What types of variables are in the columns

```
In [31]: VacationHistory.dtypes
```

```
Out[31]: Chicago    object  
         IL        object  
         5         int64  
         2008      int64  
         dtype: object
```

Look at the first and last rows in xyzcust10:

```
In [32]: VacationHistory.head()
```

```
Out[32]:
```

	Chicago	IL	5	2008
0	NewYork	NY	8	2014
1	Detriot	MI	2	2007
2	SanFransico	CA	7	2003
3	Boston	MA	4	2010

Does the header of the table look right?  
Any changes you might think of?  
Is this the right table header?