

Python Pandas Tutorial: A Complete Introduction for Beginners

The *pandas* package is the most important tool at the disposal of Data Scientists and Analysts working in Python today. The powerful machine learning and glamorous visualization tools may get all the attention, but pandas is the backbone of most data projects.

[*pandas*] is derived from the term "**panel data**", an econometrics term for data sets that include observations over multiple time periods for the same individuals. — [Wikipedia](https://en.wikipedia.org/wiki/Pandas_%28software%29)
(https://en.wikipedia.org/wiki/Pandas_%28software%29)

If you're thinking about data science as a career, then it is imperative that one of the first things you do is learn pandas. In this post, we will go over the essential bits of information about pandas, including how to install it, its uses, and how it works with other common Python data analysis packages such as **matplotlib** and **sci-kit learn**.



What's Pandas for?

Pandas has so many uses that it might make sense to list the things it can't do instead of what it can do.

This tool is essentially your data's home. Through pandas, you get acquainted with your data by cleaning, transforming, and analyzing it.

For example, say you want to explore a dataset stored in a CSV on your computer. Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:

- Calculate statistics and answer questions about the data, like
 - What's the average, median, max, or min of each column?
 - Does column A correlate with column B?
 - What does the distribution of data in column C look like?
- Clean the data by doing things like removing missing values and filtering rows or columns by some criteria
- Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more.
- Store the cleaned, transformed data back into a CSV, other file or database

Before you jump into the modeling or the complex visualizations you need to have a good understanding of the nature of your dataset and pandas is the best avenue through which to do that.

How does pandas fit into the data science toolkit?

Not only is the pandas library a central component of the data science toolkit but it is used in conjunction with other libraries in that collection.

Pandas is built on top of the **NumPy** package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in **SciPy**, plotting functions from **Matplotlib**, and machine learning algorithms in **Scikit-learn**.

Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' DataFrames and plots. As a matter of fact, this article was created entirely in a Jupyter Notebook.

When should you start using pandas?

If you do not have any experience coding in Python, then you should stay away from learning pandas until you do. You don't have to be at the level of the software engineer, but you should be adept at the basics, such as lists, tuples, dictionaries, functions, and iterations. Also, I'd also recommend familiarizing yourself with **NumPy** due to the similarities mentioned above.

If you're looking for a good place to learn Python, [Python for Everybody](https://www.learndatasci.com/out/coursera-programming-everybody-getting-started-python/) (<https://www.learndatasci.com/out/coursera-programming-everybody-getting-started-python/>) on Coursera is great (and Free).

Moreover, for those of you looking to do a [data science bootcamp](https://www.learndatasci.com/articles/thinkful-data-science-online-bootcamp-review/) (<https://www.learndatasci.com/articles/thinkful-data-science-online-bootcamp-review/>) or some other accelerated data science education program, it's highly recommended you start learning pandas on your own before you start the program.

Even though accelerated programs teach you pandas, better skills beforehand means you'll be able to maximize time for learning and mastering the more complicated material.

Pandas First Steps

Install and import

Pandas is an easy package to install. Open up your terminal program (for Mac users) or command line (for PC users) and install it using either of the following commands:

```
conda install pandas
```

OR

```
pip install pandas
```

Alternatively, if you're currently viewing this article in a Jupyter notebook you can run this cell:

```
In [1]: !pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\musad\anaconda3\lib\site-packages (0.24.2)
Requirement already satisfied: numpy>=1.12.0 in c:\users\musad\anaconda3\lib\site-packages (from pandas) (1.16.4)
Requirement already satisfied: python-dateutil>=2.5.0 in c:\users\musad\anaconda3\lib\site-packages (from pandas) (2.8.0)
Requirement already satisfied: pytz>=2011k in c:\users\musad\anaconda3\lib\site-packages (from pandas) (2019.1)
Requirement already satisfied: six>=1.5 in c:\users\musad\anaconda3\lib\site-packages (from python-dateutil>=2.5.0->pandas) (1.12.0)
```

The `!` at the beginning runs cells as if they were in a terminal.

To import pandas we usually import it with a shorter name since it's used so much:

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

Now to the basic components of pandas.

Core components of pandas: Series and DataFrames

The primary two components of pandas are the `Series` and `DataFrame`.

A `Series` is essentially a column, and a `DataFrame` is a multi-dimensional table made up of a collection of `Series`.



`DataFrames` and `Series` are quite similar in that many operations that you can do with one you can do with the other, such as filling in null values and calculating the mean.

You'll see how these components work when we start working with data below.

Creating DataFrames from scratch

Creating DataFrames right in Python is good to know and quite useful when testing new methods and functions you find in the pandas docs.

There are *many* ways to create a DataFrame from scratch, but a great option is to just use a simple `dict`.

Let's say we have a fruit stand that sells apples and oranges. We want to have a column for each fruit and a row for each customer purchase. To organize this as a dictionary for pandas we could do something like:

```
In [3]: data = {  
        'apples': [3, 2, 0, 1],  
        'oranges': [0, 3, 7, 2]  
    }
```

And then pass it to the pandas DataFrame constructor:

```
In [4]: purchases = pd.DataFrame(data)  
  
purchases
```

Out[4]:

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

How did that work?

Each *(key, value)* item in `data` corresponds to a *column* in the resulting DataFrame.

The **Index** of this DataFrame was given to us on creation as the numbers 0-3, but we could also create our own when we initialize the DataFrame.

Let's have customer names as our index:

```
In [5]: purchases = pd.DataFrame(data, index=['June', 'Robert', 'Lily', 'David'])  
purchases
```

Out[5]:

	apples	oranges
June	3	0
Robert	2	3
Lily	0	7
David	1	2

So now we could **locate** a customer's order by using their name:

```
In [6]: purchases.loc['June']
```

Out[6]: apples 3
oranges 0
Name: June, dtype: int64

There's more on locating and extracting data from the DataFrame later, but now you should be able to create a DataFrame with any random data to learn on.

Let's move on to some quick methods for creating DataFrames from various other sources.

How to read in data

It's quite simple to load data from various file formats into a DataFrame. In the following examples we'll keep using our apples and oranges data, but this time it's coming from various files.

Reading data from CSVs

With CSV files all you need is a single line to load in the data:

```
In [48]: df = pd.read_csv('purchases.csv')  
df
```

Out[48]:

	Unnamed: 0	apples	oranges
0	June	3	0
1	Robert	2	3
2	Lily	0	7
3	David	1	2

CSVs don't have indexes like our DataFrames, so all we need to do is just designate the `index_col` when reading:

```
In [53]: df = pd.read_csv('purchases.csv', index_col=0)
df
```

Out[53]:

	apples	oranges
June	3	0
Robert	2	3
Lily	0	7
David	1	2

Here we're setting the index to be column zero.

You'll find that most CSVs won't ever have an index column and so usually you don't have to worry about this step.

Reading data from JSON

If you have a JSON file — which is essentially a stored Python `dict` — pandas can read this just as easily:

```
In [55]: df = pd.read_json('purchases.json')
df
```

Out[55]:

	apples	oranges
David	1	2
June	3	0
Lily	0	7
Robert	2	3

Notice this time our index came with us correctly since using JSON allowed indexes to work through nesting. Feel free to open `data_file.json` in a notepad so you can see how it works.

Pandas will try to figure out how to create a DataFrame by analyzing structure of your JSON, and sometimes it doesn't get it right. Often you'll need to set the `orient` keyword argument depending on the structure, so check out [read_json docs \(https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_json.html\)](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_json.html) about that argument to see which orientation you're using.

Reading data from a SQL database

If you're working with data from a SQL database you need to first establish a connection using an appropriate Python library, then pass a query to pandas. Here we'll use SQLite to demonstrate.

First, we need `pysqlite3` installed, so run this command in your terminal:

```
pip install pysqlite3
```

Or run this cell if you're in a notebook:

```
In [ ]: !pip install pysqlite3
```

`sqlite3` is used to create a connection to a database which we can then use to generate a DataFrame through a `SELECT` query.

So first we'll make a connection to a SQLite database file:

```
In [56]: import sqlite3

con = sqlite3.connect("database.db")
```

Note: If you have data in PostgreSQL, MySQL, or some other SQL server, you'll need to obtain the right Python library to make a connection. For example, `psycopg2` ([link \(http://initd.org/psycopg/download/\)](http://initd.org/psycopg/download/)) is a commonly used library for making connections to PostgreSQL. Furthermore, you would make a connection to a database URI instead of a file like we did here with SQLite. For a great course on SQL check out [The Complete SQL Bootcamp \(https://learndatasci.com/out/udemy-the-complete-sql-bootcamp/\)](https://learndatasci.com/out/udemy-the-complete-sql-bootcamp/) on Udemy

In this SQLite database we have a table called *purchases*, and our index is in a column called "index".

By passing a `SELECT` query and our `con`, we can read from the *purchases* table:

```
In [62]: df = pd.read_sql_query("SELECT * FROM purchases", con)
df
```

Out[62]:

	index	apples	oranges
0	June	3	0
1	Robert	2	3
2	Lily	0	7
3	David	1	2

Just like with CSVs, we could pass `index_col='index'` , but we can also set an index after-the-fact:

```
In [64]: df = df.set_index('index')
df
```

Out[64]:

	apples	oranges
index		
June	3	0
Robert	2	3
Lily	0	7
David	1	2

In fact, we could use `set_index()` on *any* DataFrame using *any* column at *any* time. Indexing Series and DataFrames is a very common task, and the different ways of doing it is worth remembering.

Converting back to a CSV, JSON, or SQL

So after extensive work on cleaning your data, you're now ready to save it as a file of your choice. Similar to the ways we read in data, pandas provides intuitive commands to save it:

```
In [ ]: df.to_csv('new_purchases.csv')
df.to_json('new_purchases.json')
df.to_sql('new_purchases', con)
```


When we save JSON and CSV files, all we have to input into those functions is our desired filename with the appropriate file extension. With SQL, we're not creating a new file but instead inserting a new table into the database using our `conn` variable from before.

Let's move on to importing some real-world data and detailing a few of the operations you'll be using a lot.

Most important DataFrame operations

DataFrames possess hundreds of methods and other operations that are crucial to any analysis. As a beginner, you should know the operations that perform simple transformations of your data and those that provide fundamental statistical analysis.

```
In [7]: movies_df = pd.read_csv("IMDB-Movie-Data.csv", index_col="Title")
```

We're loading this dataset from a CSV and designating the movie titles to be our index.

Viewing your data

The first thing to do when opening a new dataset is print out a few rows to keep as a visual reference. We accomplish this with `.head()` :

In [8]: `movies_df.head()`

Out[8]:

	Rank	Genre	Description	Director	Actors	Year	(
Title							
Guardians of the Galaxy	1	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	
Split	3	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	
Sing	4	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	
Suicide Squad	5	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	

`.head()` outputs the **first** five rows of your DataFrame by default, but we could also pass a number as well: `movies_df.head(10)` would output the top ten rows, for example.

To see the **last** five rows use `.tail()`. `tail()` also accepts a number, and in this case we printing the bottom two rows.:

In [9]: `movies_df.tail(2)`

Out[9]:

	Rank	Genre	Description	Director	Actors	Year	Runtime (Minutes)
Title							
Search Party	999	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	2014	93
Nine Lives	1000	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87

Typically when we load in a dataset, we like to view the first five or so rows to see what's under the hood. Here we can see the names of each column, the index, and examples of values in each row.

You'll notice that the index in our DataFrame is the *Title* column, which you can tell by how the word *Title* is slightly lower than the rest of the columns.

Getting info about your data

`.info()` should be one of the very first commands you run after loading your data:

In [10]: `movies_df.info()`

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, Guardians of the Galaxy to Nine Lives
Data columns (total 11 columns):
Rank                1000 non-null int64
Genre               1000 non-null object
Description          1000 non-null object
Director            1000 non-null object
Actors              1000 non-null object
Year               1000 non-null int64
Runtime (Minutes)   1000 non-null int64
Rating              1000 non-null float64
Votes               1000 non-null int64
Revenue (Millions)  872 non-null float64
Metascore           936 non-null float64
dtypes: float64(3), int64(4), object(4)
memory usage: 93.8+ KB
```

`.info()` provides the essential details about your dataset, such as the number of rows and columns, the number of non-null values, what type of data is in each column, and how much memory your DataFrame is using.

Notice in our movies dataset we have some obvious missing values in the `Revenue` and `Metascore` columns. We'll look at how to handle those in a bit.

Seeing the datatype quickly is actually quite useful. Imagine you just imported some JSON and the integers were recorded as strings. You go to do some arithmetic and find an "unsupported operand" Exception because you can't do math with strings. Calling `.info()` will quickly point out that your column you thought was all integers are actually string objects.

Another fast and useful attribute is `.shape`, which outputs just a tuple of (rows, columns):

```
In [11]: movies_df.shape
```

```
Out[11]: (1000, 11)
```

Note that `.shape` has no parentheses and is a simple tuple of format (rows, columns). So we have **1000 rows** and **11 columns** in our movies DataFrame.

You'll be going to `.shape` a lot when cleaning and transforming data. For example, you might filter some rows based on some criteria and then want to know quickly how many rows were removed.

Handling duplicates

This dataset does not have duplicate rows, but it is always important to verify you aren't aggregating duplicate rows.

To demonstrate, let's simply just double up our movies DataFrame by appending it to itself:

```
In [12]: temp_df = movies_df.append(movies_df)
temp_df.shape
```

```
Out[12]: (2000, 11)
```

Using `append()` will return a copy without affecting the original DataFrame. We are capturing this copy in `temp` so we aren't working with the real data.

Notice call `.shape` quickly proves our DataFrame rows have doubled.

Now we can try dropping duplicates:

```
In [13]: temp_df = temp_df.drop_duplicates()

temp_df.shape
```

Out[13]: (1000, 11)

Just like `append()`, the `drop_duplicates()` method will also return a copy of your DataFrame, but this time with duplicates removed. Calling `.shape` confirms we're back to the 1000 rows of our original dataset.

It's a little verbose to keep assigning DataFrames to the same variable like in this example. For this reason, pandas has the `inplace` keyword argument on many of its methods. Using `inplace=True` will modify the DataFrame object in place:

```
In [14]: temp_df.drop_duplicates(inplace=True)
```

Now our `temp_df` will have the transformed data automatically.

Another important argument for `drop_duplicates()` is `keep`, which has three possible options:

- `first` : (default) Drop duplicates except for the first occurrence.
- `last` : Drop duplicates except for the last occurrence.
- `False` : Drop all duplicates.

Since we didn't define the `keep` argument in the previous example it was defaulted to `first`. This means that if two rows are the same pandas will drop the second row and keep the first row. Using `last` has the opposite effect: the first row is dropped.

`keep`, on the other hand, will drop all duplicates. If two rows are the same then both will be dropped. Watch what happens to `temp_df`:

```
In [15]: temp_df = movies_df.append(movies_df) # make a new copy

temp_df.drop_duplicates(inplace=True, keep=False)

temp_df.shape
```

Out[15]: (0, 11)

Since all rows were duplicates, `keep=False` dropped them all resulting in zero rows being left over. If you're wondering why you would want to do this, one reason is that it allows you to locate all duplicates in your dataset. When conditional selections are shown below you'll see how to do that.

Column cleanup

Many times datasets will have verbose column names with symbols, upper and lowercase words, spaces, and typos. To make selecting data by column name easier we can spend a little time cleaning up their names.

Here's how to print the column names of our dataset:

```
In [16]: movies_df.columns

Out[16]: Index(['Rank', 'Genre', 'Description', 'Director', 'Actors', 'Year',
               'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
               'Metascore'],
              dtype='object')
```

Not only does `.columns` come in handy if you want to rename columns by allowing for simple copy and paste, it's also useful if you need to understand why you are receiving a `Key Error` when selecting data by column.

We can use the `.rename()` method to rename certain or all columns via a `dict`. We don't want parentheses, so let's rename those:

```
In [17]: movies_df.rename(columns={
          'Runtime (Minutes)': 'Runtime',
          'Revenue (Millions)': 'Revenue_millions'
        }, inplace=True)

movies_df.columns

Out[17]: Index(['Rank', 'Genre', 'Description', 'Director', 'Actors', 'Year', 'Runtime',
               'Rating', 'Votes', 'Revenue_millions', 'Metascore'],
              dtype='object')
```

Excellent. But what if we want to lowercase all names? Instead of using `.rename()` we could also set a list of names to the columns like so:

```
In [18]: movies_df.columns = ['rank', 'genre', 'description', 'director', 'actors', 'year',
                              'runtime', 'rating', 'votes', 'revenue_millions', 'metascore']

movies_df.columns

Out[18]: Index(['rank', 'genre', 'description', 'director', 'actors', 'year', 'runtime',
               'rating', 'votes', 'revenue_millions', 'metascore'],
              dtype='object')
```

But that's too much work. Instead of just renaming each column manually we can do a list comprehension:

```
In [19]: movies_df.columns = [col.lower() for col in movies_df]

movies_df.columns

Out[19]: Index(['rank', 'genre', 'description', 'director', 'actors', 'year', 'runtime',
               'rating', 'votes', 'revenue_millions', 'metascore'],
              dtype='object')
```

list (and dict) comprehensions come in handy a lot when working with pandas and data in general.

It's a good idea to lowercase, remove special characters, and replace spaces with underscores if you'll be working with a dataset for some time.

How to work with missing values

When exploring data, you'll most likely encounter missing or null values, which are essentially placeholders for non-existent values. Most commonly you'll see Python's `None` or NumPy's `np.nan`, each of which are handled differently in some situations.

There are two options in dealing with nulls:

1. Get rid of rows or columns with nulls
2. Replace nulls with non-null values, a technique known as **imputation**

Let's calculate the total number of nulls in each column of our dataset. The first step is to check which cells in our DataFrame are null:

```
In [20]: movies_df.isnull()
```


Out[20]:

	rank	genre	description	director	actors	year	runtime	rating	votes	revenue
Title										
Guardians of the Galaxy	False	False	False	False	False	False	False	False	False	False
Prometheus	False	False	False	False	False	False	False	False	False	False
Split	False	False	False	False	False	False	False	False	False	False
Sing	False	False	False	False	False	False	False	False	False	False
Suicide Squad	False	False	False	False	False	False	False	False	False	False
The Great Wall	False	False	False	False	False	False	False	False	False	False
La La Land	False	False	False	False	False	False	False	False	False	False
Mindhorn	False	False	False	False	False	False	False	False	False	False
The Lost City of Z	False	False	False	False	False	False	False	False	False	False
Passengers	False	False	False	False	False	False	False	False	False	False
Fantastic Beasts and Where to Find Them	False	False	False	False	False	False	False	False	False	False
Hidden Figures	False	False	False	False	False	False	False	False	False	False
Rogue One	False	False	False	False	False	False	False	False	False	False
Moana	False	False	False	False	False	False	False	False	False	False
Colossal	False	False	False	False	False	False	False	False	False	False
The Secret Life of Pets	False	False	False	False	False	False	False	False	False	False
Hacksaw Ridge	False	False	False	False	False	False	False	False	False	False
Jason Bourne	False	False	False	False	False	False	False	False	False	False
Lion	False	False	False	False	False	False	False	False	False	False
Arrival	False	False	False	False	False	False	False	False	False	False
Gold	False	False	False	False	False	False	False	False	False	False
Manchester by the Sea	False	False	False	False	False	False	False	False	False	False
Hounds of Love	False	False	False	False	False	False	False	False	False	False
Trolls	False	False	False	False	False	False	False	False	False	False
Independence Day: Resurgence	False	False	False	False	False	False	False	False	False	False
Paris pieds nus	False	False	False	False	False	False	False	False	False	False

	rank	genre	description	director	actors	year	runtime	rating	votes	revenue
Title										
Bahubali: The Beginning	False	False	False	False	False	False	False	False	False	False
Dead Awake	False	False	False	False	False	False	False	False	False	False
Bad Moms	False	False	False	False	False	False	False	False	False	False
Assassin's Creed	False	False	False	False	False	False	False	False	False	False
...
Texas Chainsaw 3D	False	False	False	False	False	False	False	False	False	False
Disturbia	False	False	False	False	False	False	False	False	False	False
Rock of Ages	False	False	False	False	False	False	False	False	False	False
Scream 4	False	False	False	False	False	False	False	False	False	False
Queen of Katwe	False	False	False	False	False	False	False	False	False	False
My Big Fat Greek Wedding 2	False	False	False	False	False	False	False	False	False	False
Dark Places	False	False	False	False	False	False	False	False	False	False
Amateur Night	False	False	False	False	False	False	False	False	False	False
It's Only the End of the World	False	False	False	False	False	False	False	False	False	False
The Skin I Live In	False	False	False	False	False	False	False	False	False	False
Miracles from Heaven	False	False	False	False	False	False	False	False	False	False
Annie	False	False	False	False	False	False	False	False	False	False
Across the Universe	False	False	False	False	False	False	False	False	False	False
Let's Be Cops	False	False	False	False	False	False	False	False	False	False
Max	False	False	False	False	False	False	False	False	False	False
Your Highness	False	False	False	False	False	False	False	False	False	False
Final Destination 5	False	False	False	False	False	False	False	False	False	False
Endless Love	False	False	False	False	False	False	False	False	False	False
Martyrs	False	False	False	False	False	False	False	False	False	False
Selma	False	False	False	False	False	False	False	False	False	False
Underworld: Rise of the Lycans	False	False	False	False	False	False	False	False	False	False

	rank	genre	description	director	actors	year	runtime	rating	votes	revenue
Title										
Taare Zameen Par	False	False	False	False	False	False	False	False	False	False
Take Me Home Tonight	False	False	False	False	False	False	False	False	False	False
Resident Evil: Afterlife	False	False	False	False	False	False	False	False	False	False
Project X	False	False	False	False	False	False	False	False	False	False
Secret in Their Eyes	False	False	False	False	False	False	False	False	False	False
Hostel: Part II	False	False	False	False	False	False	False	False	False	False
Step Up 2: The Streets	False	False	False	False	False	False	False	False	False	False
Search Party	False	False	False	False	False	False	False	False	False	False
Nine Lives	False	False	False	False	False	False	False	False	False	False

1000 rows × 11 columns



Notice `isnull()` returns a DataFrame where each cell is either True or False depending on that cell's null status.

To count the number of nulls in each column we use an aggregate function for summing:

```
In [21]: movies_df.isnull().sum()
```

```
Out[21]: rank                0
genre                0
description          0
director            0
actors              0
year                0
runtime             0
rating              0
votes               0
revenue_millions    128
metascore           64
dtype: int64
```

`.isnull()` just by itself isn't very useful, and is usually used in conjunction with other methods, like `sum()`.

We can see now that our data has **128** missing values for `revenue_millions` and **64** missing values for `metascore`.

Removing null values

Data Scientists and Analysts regularly face the dilemma of dropping or imputing null values, and is a decision that requires intimate knowledge of your data and its context. Overall, removing null data is only suggested if you have a small amount of missing data.

Remove nulls is pretty simple:

In [22]: `movies_df.dropna()`

Out[22]:

	rank	genre	description	director	actors
Title					
Guardians of the Galaxy	1	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...
Split	3	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...
Sing	4	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...
Suicide Squad	5	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...
The Great Wall	6	Action,Adventure,Fantasy	European mercenaries searching for black powde...	Yimou Zhang	Matt Damon, Tian Jing, Willem Dafoe, Andy Lau
La La Land	7	Comedy,Drama,Music	A jazz pianist falls for an aspiring actress i...	Damien Chazelle	Ryan Gosling, Emma Stone, Rosemarie DeWitt, J....
The Lost City of Z	9	Action,Adventure,Biography	A true-life drama, centering on British explor...	James Gray	Charlie Hunnam, Robert Pattinson, Sienna Mille...
Passengers	10	Adventure,Drama,Romance	A spacecraft traveling to a distant colony pla...	Morten Tyldum	Jennifer Lawrence, Chris Pratt, Michael Sheen,...
Fantastic Beasts and Where to Find Them	11	Adventure,Family,Fantasy	The adventures of writer Newt Scamander in New...	David Yates	Eddie Redmayne, Katherine Waterston, Alison Su...
Hidden Figures	12	Biography,Drama,History	The story of a team of female African-American...	Theodore Melfi	Taraji P. Henson, Octavia Spencer, Janelle Mon...

	rank	genre	description	director	actors
Title					
Rogue One	13	Action,Adventure,Sci-Fi	The Rebel Alliance makes a risky move to steal...	Gareth Edwards	Felicity Jones, Diego Luna, Alan Tudyk, Donnie...
Moana	14	Animation,Adventure,Comedy	In Ancient Polynesia, when a terrible curse in...	Ron Clements	Auli'i Cravalho, Dwayne Johnson, Rachel House,...
Colossal	15	Action,Comedy,Drama	Gloria is an out-of-work party girl forced to ...	Nacho Vigalondo	Anne Hathaway, Jason Sudeikis, Austin Stowell,...
The Secret Life of Pets	16	Animation,Adventure,Comedy	The quiet life of a terrier named Max is upend...	Chris Renaud	Louis C.K., Eric Stonestreet, Kevin Hart, Lake...
Hacksaw Ridge	17	Biography,Drama,History	WWII American Army Medic Desmond T. Doss, who ...	Mel Gibson	Andrew Garfield, Sam Worthington, Luke Bracey,...
Jason Bourne	18	Action,Thriller	The CIA's most dangerous former operative is d...	Paul Greengrass	Matt Damon, Tommy Lee Jones, Alicia Vikander,V...
Lion	19	Biography,Drama	A five-year-old Indian boy gets lost on the st...	Garth Davis	Dev Patel, Nicole Kidman, Rooney Mara, Sunny P...
Arrival	20	Drama,Mystery,Sci-Fi	When twelve mysterious spacecraft appear aroun...	Denis Villeneuve	Amy Adams, Jeremy Renner, Forest Whitaker,Mich...
Gold	21	Adventure,Drama,Thriller	Kenny Wells, a prospector desperate for a luck...	Stephen Gaghan	Matthew McConaughey, Edgar Ramírez, Bryce Dall...
Manchester by the Sea	22	Drama	A depressed uncle is asked to take care of his...	Kenneth Lonergan	Casey Affleck, Michelle Williams, Kyle Chandle...
Trolls	24	Animation,Adventure,Comedy	After the Bergens invade Troll Village, Poppy,...	Walt Dohrn	Anna Kendrick, Justin Timberlake,Zoey Deschan...
Independence Day: Resurgence	25	Action,Adventure,Sci-Fi	Two decades after the first Independence Day i...	Roland Emmerich	Liam Hemsworth, Jeff Goldblum, Bill Pullman,Ma...


	rank	genre	description	director	actors
Title					
Bad Moms	29	Comedy	When three overworked and under-appreciated mo...	Jon Lucas	Mila Kunis, Kathryn Hahn, Kristen Bell, Christi...
Assassin's Creed	30	Action,Adventure,Drama	When Callum Lynch explores the memories of his...	Justin Kurzel	Michael Fassbender, Marion Cotillard, Jeremy I...
Why Him?	31	Comedy	A holiday gathering threatens to go off the ra...	John Hamburg	Zoey Deutch, James Franco, Tangie Ambrose, Cedr...
Nocturnal Animals	32	Drama,Thriller	A wealthy art gallery owner is haunted by her ...	Tom Ford	Amy Adams, Jake Gyllenhaal, Michael Shannon, A...
X-Men: Apocalypse	33	Action,Adventure,Sci-Fi	After the re-emergence of the world's first mu...	Bryan Singer	James McAvoy, Michael Fassbender, Jennifer Law...
Deadpool	34	Action,Adventure,Comedy	A fast-talking mercenary with a morbid sense o...	Tim Miller	Ryan Reynolds, Morena Baccarin, T.J. Miller, E...
Resident Evil: The Final Chapter	35	Action,Horror,Sci-Fi	Alice returns to where the nightmare began: Th...	Paul W.S. Anderson	Milla Jovovich, Iain Glen, Ali Larter, Shawn R...
...
That Awkward Moment	956	Comedy,Romance	Three best friends find themselves where we've...	Tom Gormican	Zac Efron, Michael B. Jordan, Miles Teller, Im...
Legion	957	Action,Fantasy,Horror	When a group of strangers at a dusty roadside ...	Scott Stewart	Paul Bettany, Dennis Quaid, Charles S. Dutton,...
End of Watch	958	Crime,Drama,Thriller	Shot documentary-style, this film follows the ...	David Ayer	Jake Gyllenhaal, Michael Peña, Anna Kendrick, ...
3 Days to Kill	959	Action,Drama,Thriller	A dying CIA agent trying to reconnect with his...	McG	Kevin Costner, Hailee Steinfeld, Connie Nielse...
Lucky Number Slevin	960	Crime,Drama,Mystery	A case of mistaken identity lands Slevin into ...	Paul McGuigan	Josh Hartnett, Ben Kingsley, Morgan Freeman, L...

	rank	genre	description	director	actors
Title					
Trance	961	Crime,Drama,Mystery	An art auctioneer who has become mixed up with...	Danny Boyle	James McAvoy, Rosario Dawson, Vincent Cassel,D...
Into the Forest	962	Drama,Sci-Fi,Thriller	After a massive power outage, two sisters lear...	Patricia Rozema	Ellen Page, Evan Rachel Wood, Max Minghella, Ca...
The Other Boleyn Girl	963	Biography,Drama,History	Two sisters contend for the affection of King ...	Justin Chadwick	Natalie Portman, Scarlett Johansson, Eric Bana...
I Spit on Your Grave	964	Crime,Horror,Thriller	A writer who is brutalized during her cabin re...	Steven R. Monroe	Sarah Butler, Jeff Branson, Andrew Howard,Dani...
Texas Chainsaw 3D	971	Horror,Thriller	A young woman travels to Texas to collect an i...	John Luessenhop	Alexandra Daddario, Tania Raymonde, Scott East...
Rock of Ages	973	Comedy,Drama,Musical	A small town girl and a city boy meet on the S...	Adam Shankman	Julianne Hough, Diego Boneta, Tom Cruise, Alec...
Scream 4	974	Horror,Mystery	Ten years have passed, and Sidney Prescott, wh...	Wes Craven	Neve Campbell, Courteney Cox, David Arquette, ...
Queen of Katwe	975	Biography,Drama,Sport	A Ugandan girl sees her world rapidly change a...	Mira Nair	Madina Nalwanga, David Oyelowo, Lupita Nyong'o...
My Big Fat Greek Wedding 2	976	Comedy,Family,Romance	A Portokalos family secret brings the beloved ...	Kirk Jones	Nia Vardalos, John Corbett, Michael Constantin...
The Skin I Live In	980	Drama,Thriller	A brilliant plastic surgeon, haunted by past t...	Pedro Almodóvar	Antonio Banderas, Elena Anaya, Jan Cornet,Mari...
Miracles from Heaven	981	Biography,Drama,Family	A young girl suffering from a rare digestive d...	Patricia Riggen	Jennifer Garner, Kylie Rogers, Martin Henderso...
Annie	982	Comedy,Drama,Family	A foster kid, who lives with her mean foster m...	Will Gluck	Quvenzhané Wallis, Cameron Diaz, Jamie Foxx, R...

	rank	genre	description	director	actors
Title					
Across the Universe	983	Drama,Fantasy,Musical	The music of the Beatles and the Vietnam War f...	Julie Taymor	Evan Rachel Wood, Jim Sturgess, Joe Anderson, ...
Let's Be Cops	984	Comedy	Two struggling pals dress as police officers f...	Luke Greenfield	Jake Johnson, Damon Wayans Jr., Rob Riggle, Ni...
Max	985	Adventure,Family	A Malinois dog that helped American Marines in...	Boaz Yakin	Thomas Haden Church, Josh Wiggins, Luke Kleint...
Your Highness	986	Adventure,Comedy,Fantasy	When Prince Fabious's bride is kidnapped, he g...	David Gordon Green	Danny McBride, Natalie Portman, James Franco, ...
Final Destination 5	987	Horror,Thriller	Survivors of a suspension-bridge collapse lear...	Steven Quale	Nicholas D'Agosto, Emma Bell, Arlen Escarpeta,...
Endless Love	988	Drama,Romance	The story of a privileged girl and a charismat...	Shana Feste	Gabriella Wilde, Alex Pettyfer, Bruce Greenwoo...
Underworld: Rise of the Lycans	991	Action,Adventure,Fantasy	An origins story centered on the centuries-old...	Patrick Tatopoulos	Rhona Mitra, Michael Sheen, Bill Nighy, Steven...
Taare Zameen Par	992	Drama,Family,Music	An eight-year-old boy is thought to be a lazy ...	Aamir Khan	Darsheel Safary, Aamir Khan, Tanay Chheda, Sac...
Resident Evil: Afterlife	994	Action,Adventure,Horror	While still out to destroy the evil Umbrella C...	Paul W.S. Anderson	Milla Jovovich, Ali Larter, Wentworth Miller,K...
Project X	995	Comedy	3 high school seniors throw a birthday party t...	Nima Nourizadeh	Thomas Mann, Oliver Cooper, Jonathan Daniel Br...
Hostel: Part II	997	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...

	rank	genre	description	director	actors
Title					
Step Up 2: The Streets	998	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...
Nine Lives	1000	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...

838 rows × 11 columns



This operation will delete any **row** with at least a single null value, but it will return a new DataFrame without altering the original one. You could specify `inplace=True` in this method as well.

So in the case of our dataset, this operation would remove 128 rows where `revenue_millions` is null and 64 rows where `metascore` is null. This obviously seems like a waste since there's perfectly good data in the other columns of those dropped rows. That's why we'll look at imputation next.

Other than just dropping rows, you can also drop columns with null values by setting `axis=1` :

```
In [23]: movies_df.dropna(axis=1)
```

Out[23]:

	rank	genre	description	director	actors
Title					
Guardians of the Galaxy	1	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...
Split	3	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...
Sing	4	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...
Suicide Squad	5	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...
The Great Wall	6	Action,Adventure,Fantasy	European mercenaries searching for black powde...	Yimou Zhang	Matt Damon, Tian Jing, Willem Dafoe, Andy Lau
La La Land	7	Comedy,Drama,Music	A jazz pianist falls for an aspiring actress i...	Damien Chazelle	Ryan Gosling, Emma Stone, Rosemarie DeWitt, J....
Mindhorn	8	Comedy	A has-been actor best known for playing the ti...	Sean Foley	Essie Davis, Andrea Riseborough, Julian Barrat...
The Lost City of Z	9	Action,Adventure,Biography	A true-life drama, centering on British explor...	James Gray	Charlie Hunnam, Robert Pattinson, Sienna Mille...
Passengers	10	Adventure,Drama,Romance	A spacecraft traveling to a distant colony pla...	Morten Tyldum	Jennifer Lawrence, Chris Pratt, Michael Sheen,...
Fantastic Beasts and Where to Find Them	11	Adventure,Family,Fantasy	The adventures of writer Newt Scamander in New...	David Yates	Eddie Redmayne, Katherine Waterston, Alison Su...

	rank	genre	description	director	actors
Title					
Hidden Figures	12	Biography,Drama,History	The story of a team of female African-American...	Theodore Melfi	Taraji P. Henson, Octavia Spencer, Janelle Mon...
Rogue One	13	Action,Adventure,Sci-Fi	The Rebel Alliance makes a risky move to steal...	Gareth Edwards	Felicity Jones, Diego Luna, Alan Tudyk, Donnie...
Moana	14	Animation,Adventure,Comedy	In Ancient Polynesia, when a terrible curse in...	Ron Clements	Auli'i Cravalho, Dwayne Johnson, Rachel House,...
Colossal	15	Action,Comedy,Drama	Gloria is an out-of-work party girl forced to ...	Nacho Vigalondo	Anne Hathaway, Jason Sudeikis, Austin Stowell,...
The Secret Life of Pets	16	Animation,Adventure,Comedy	The quiet life of a terrier named Max is upend...	Chris Renaud	Louis C.K., Eric Stonestreet, Kevin Hart, Lake...
Hacksaw Ridge	17	Biography,Drama,History	WWII American Army Medic Desmond T. Doss, who ...	Mel Gibson	Andrew Garfield, Sam Worthington, Luke Bracey,...
Jason Bourne	18	Action,Thriller	The CIA's most dangerous former operative is d...	Paul Greengrass	Matt Damon, Tommy Lee Jones, Alicia Vikander,V...
Lion	19	Biography,Drama	A five-year-old Indian boy gets lost on the st...	Garth Davis	Dev Patel, Nicole Kidman, Rooney Mara, Sunny P...
Arrival	20	Drama,Mystery,Sci-Fi	When twelve mysterious spacecraft appear aroun...	Denis Villeneuve	Amy Adams, Jeremy Renner, Forest Whitaker,Mich...
Gold	21	Adventure,Drama,Thriller	Kenny Wells, a prospector desperate for a luck...	Stephen Gaghan	Matthew McConaughey, Edgar Ramírez, Bryce Dall...
Manchester by the Sea	22	Drama	A depressed uncle is asked to take care of his...	Kenneth Lonergan	Casey Affleck, Michelle Williams, Kyle Chandle...


	rank	genre	description	director	actors
Title					
Hounds of Love	23	Crime,Drama,Horror	A cold-blooded predatory couple while cruising...	Ben Young	Emma Booth, Ashleigh Cummings, Stephen Curry,S...
Trolls	24	Animation,Adventure,Comedy	After the Bergens invade Troll Village, Poppy,...	Walt Dohrn	Anna Kendrick, Justin Timberlake,Zoey Deschan...
Independence Day: Resurgence	25	Action,Adventure,Sci-Fi	Two decades after the first Independence Day i...	Roland Emmerich	Liam Hemsworth, Jeff Goldblum, Bill Pullman,Ma...
Paris pieds nus	26	Comedy	Fiona visits Paris for the first time to assis...	Dominique Abel	Fiona Gordon, Dominique Abel,Emmanuelle Riva, ...
Bahubali: The Beginning	27	Action,Adventure,Drama	In ancient India, an adventurous and daring ma...	S.S. Rajamouli	Prabhas, Rana Daggubati, Anushka Shetty,Tamann...
Dead Awake	28	Horror,Thriller	A young woman must save herself and her friend...	Phillip Guzman	Jocelin Donahue, Jesse Bradford, Jesse Borrego...
Bad Moms	29	Comedy	When three overworked and under-appreciated mo...	Jon Lucas	Mila Kunis, Kathryn Hahn, Kristen Bell,Christi...
Assassin's Creed	30	Action,Adventure,Drama	When Callum Lynch explores the memories of his...	Justin Kurzel	Michael Fassbender, Marion Cotillard, Jeremy I...
...
Texas Chainsaw 3D	971	Horror,Thriller	A young woman travels to Texas to collect an i...	John Luessenhop	Alexandra Daddario, Tania Raymonde, Scott East...
Disturbia	972	Drama,Mystery,Thriller	A teen living under house arrest becomes convi...	D.J. Caruso	Shia LaBeouf, David Morse, Carrie-Anne Moss, S...
Rock of Ages	973	Comedy,Drama,Musical	A small town girl and a city boy meet on the S...	Adam Shankman	Julianne Hough, Diego Boneta, Tom Cruise, Alec...

	rank	genre	description	director	actors
Title					
Scream 4	974	Horror,Mystery	Ten years have passed, and Sidney Prescott, wh...	Wes Craven	Neve Campbell, Courteney Cox, David Arquette, ...
Queen of Katwe	975	Biography,Drama,Sport	A Ugandan girl sees her world rapidly change a...	Mira Nair	Madina Nalwanga, David Oyelowo, Lupita Nyong'o...
My Big Fat Greek Wedding 2	976	Comedy,Family,Romance	A Portokalos family secret brings the beloved ...	Kirk Jones	Nia Vardalos, John Corbett, Michael Constantin...
Dark Places	977	Drama,Mystery,Thriller	Libby Day was only eight years old when her fa...	Gilles Paquet-Brenner	Charlize Theron, Nicholas Hoult, Christina Hen...
Amateur Night	978	Comedy	Guy Carter is an award-winning graduate studen...	Lisa Addario	Jason Biggs, Janet Montgomery,Ashley Tisdale, ...
It's Only the End of the World	979	Drama	Louis (Gaspard Ulliel), a terminally ill write...	Xavier Dolan	Nathalie Baye, Vincent Cassel, Marion Cotillar...
The Skin I Live In	980	Drama,Thriller	A brilliant plastic surgeon, haunted by past t...	Pedro Almodóvar	Antonio Banderas, Elena Anaya, Jan Cornet,Mari...
Miracles from Heaven	981	Biography,Drama,Family	A young girl suffering from a rare digestive d...	Patricia Riggen	Jennifer Garner, Kylie Rogers, Martin Henderso...
Annie	982	Comedy,Drama,Family	A foster kid, who lives with her mean foster m...	Will Gluck	Quvenzhané Wallis, Cameron Diaz, Jamie Foxx, R...
Across the Universe	983	Drama,Fantasy,Musical	The music of the Beatles and the Vietnam War f...	Julie Taymor	Evan Rachel Wood, Jim Sturgess, Joe Anderson, ...
Let's Be Cops	984	Comedy	Two struggling pals dress as police officers f...	Luke Greenfield	Jake Johnson, Damon Wayans Jr., Rob Riggle, Ni...
Max	985	Adventure,Family	A Malinois dog that helped American Marines in...	Boaz Yakin	Thomas Haden Church, Josh Wiggins, Luke Kleint...

	rank	genre	description	director	actors
Title					
Your Highness	986	Adventure,Comedy,Fantasy	When Prince Fabious's bride is kidnapped, he g...	David Gordon Green	Danny McBride, Natalie Portman, James Franco, ...
Final Destination 5	987	Horror,Thriller	Survivors of a suspension-bridge collapse lear...	Steven Quale	Nicholas D'Agosto, Emma Bell, Arlen Escarpeta,...
Endless Love	988	Drama,Romance	The story of a privileged girl and a charismat...	Shana Feste	Gabriella Wilde, Alex Pettyfer, Bruce Greenwood...
Martyrs	989	Horror	A young woman's quest for revenge against the ...	Pascal Laugier	Morjana Alaoui, Mylène Jampanoï, Catherine Bég...
Selma	990	Biography,Drama,History	A chronicle of Martin Luther King's campaign t...	Ava DuVernay	David Oyelowo, Carmen Ejogo, Tim Roth, Lorrain...
Underworld: Rise of the Lycans	991	Action,Adventure,Fantasy	An origins story centered on the centuries-old...	Patrick Tatopoulos	Rhona Mitra, Michael Sheen, Bill Nighy, Steven...
Taare Zameen Par	992	Drama,Family,Music	An eight-year-old boy is thought to be a lazy ...	Aamir Khan	Darsheel Safary, Aamir Khan, Tanay Chheda, Sac...
Take Me Home Tonight	993	Comedy,Drama,Romance	Four years after graduation, an awkward high s...	Michael Dowse	Topher Grace, Anna Faris, Dan Fogler, Teresa P...
Resident Evil: Afterlife	994	Action,Adventure,Horror	While still out to destroy the evil Umbrella C...	Paul W.S. Anderson	Milla Jovovich, Ali Larter, Wentworth Miller,K...
Project X	995	Comedy	3 high school seniors throw a birthday party t...	Nima Nourizadeh	Thomas Mann, Oliver Cooper, Jonathan Daniel Br...
Secret in Their Eyes	996	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...
Hostel: Part II	997	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...

	rank	genre	description	director	actors
Title					
Step Up 2: The Streets	998	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...
Search Party	999	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...
Nine Lives	1000	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...

1000 rows × 9 columns



In our dataset, this operation would drop the `revenue_millions` and `metascore` columns.

Intuition side note: What's with this `axis=1` parameter?

It's not immediately obvious where `axis` comes from and why you need it to be 1 for it to affect columns. To see why, just look at the `.shape` output:

```
In [24]: movies_df.shape
```

```
Out[24]: (1000, 11)
```

As we learned above, this is a tuple that represents the shape of the DataFrame, i.e. 1000 rows and 11 columns. Note that the *rows* are at index zero of this tuple and *columns* are at **index one** of this tuple. This is why `axis=1` affects columns. This comes from NumPy, and is a great example of why learning NumPy is worth your time.

Imputation

Imputation is a conventional feature engineering technique used to keep valuable data that have null values.

There may be instances where dropping every row with a null value removes too big a chunk from your dataset, so instead we can impute that null with another value, usually the **mean** or the **median** of that column.

Let's look at imputing the missing values in the `revenue_millions` column. First we'll extract that column into its own variable:

```
In [25]: revenue = movies_df['revenue_millions']
```

Using square brackets is the general way we select columns in a DataFrame.

If you remember back to when we created DataFrames from scratch, the keys of the `dict` ended up as column names. Now when we select columns of a DataFrame, we use brackets just like if we were accessing a Python dictionary.

`revenue` now contains a Series:

```
In [26]: revenue.head()

Out[26]: Title
Guardians of the Galaxy    333.13
Prometheus                 126.46
Split                    138.12
Sing                     270.32
Suicide Squad             325.02
Name: revenue_millions, dtype: float64
```

Slightly different formatting than a DataFrame, but we still have our `Title` index.

We'll impute the missing values of revenue using the mean. Here's the mean value:

```
In [27]: revenue_mean = revenue.mean()

revenue_mean
```

```
Out[27]: 82.95637614678897
```

With the mean, let's fill the nulls using `fillna()` :

```
In [28]: revenue.fillna(revenue_mean, inplace=True)
```

We have now replaced all nulls in `revenue` with the mean of the column. Notice that by using `inplace=True` we have actually affected the original `movies_df` :

```
In [29]: movies_df.isnull().sum()
```

```
Out[29]: rank                0
genre                    0
description              0
director                 0
actors                  0
year                    0
runtime                 0
rating                  0
votes                   0
revenue_millions        0
metascore                64
dtype: int64
```

Imputing an entire column with the same value like this is a basic example. It would be a better idea to try a more granular imputation by Genre or Director.

For example, you would find the mean of the revenue generated in each genre individually and impute the nulls in each genre with that genre's mean.

Let's now look at more ways to examine and understand the dataset.

Understanding your variables

Using `describe()` on an entire DataFrame we can get a summary of the distribution of continuous variables:

```
In [30]: movies_df.describe()
```

```
Out[30]:
```

	rank	year	runtime	rating	votes	revenue_millions	me
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	1000.000000	936
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	58
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	96.412043	17
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	11
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	17.442500	47
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	60.375000	59
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	99.177500	72
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	100

Understanding which numbers are continuous also comes in handy when thinking about the type of plot to use to represent your data visually.

`.describe()` can also be used on a categorical variable to get the count of rows, unique count of categories, top category, and freq of top category:

```
In [31]: movies_df['genre'].describe()

Out[31]: count                1000
         unique                207
         top      Action,Adventure,Sci-Fi
         freq                50
         Name: genre, dtype: object
```

This tells us that the genre column has 207 unique values, the top value is Action/Adventure/Sci-Fi, which shows up 50 times (freq).

`.value_counts()` can tell us the frequency of all values in a column:

```
In [32]: movies_df['genre'].value_counts().head(10)

Out[32]: Action,Adventure,Sci-Fi    50
         Drama                    48
         Comedy,Drama,Romance      35
         Comedy                    32
         Drama,Romance              31
         Comedy,Drama              27
         Animation,Adventure,Comedy 27
         Action,Adventure,Fantasy   27
         Comedy,Romance            26
         Crime,Drama,Thriller       24
         Name: genre, dtype: int64
```

Relationships between continuous variables

By using the correlation method `.corr()` we can generate the relationship between each continuous variable:

In [33]: `movies_df.corr()`

Out[33]:

	rank	year	runtime	rating	votes	revenue_millions	metasc
rank	1.000000	-0.261605	-0.221739	-0.219555	-0.283876	-0.252996	-0.191
year	-0.261605	1.000000	-0.164900	-0.211219	-0.411904	-0.117562	-0.079
runtime	-0.221739	-0.164900	1.000000	0.392214	0.407062	0.247834	0.211
rating	-0.219555	-0.211219	0.392214	1.000000	0.511537	0.189527	0.631
votes	-0.283876	-0.411904	0.407062	0.511537	1.000000	0.607941	0.325
revenue_millions	-0.252996	-0.117562	0.247834	0.189527	0.607941	1.000000	0.133
metascore	-0.191869	-0.079305	0.211978	0.631897	0.325684	0.133328	1.000

Correlation tables are a numerical representation of the bivariate relationships in the dataset.

Positive numbers indicate a positive correlation — one goes up the other goes up — and negative numbers represent an inverse correlation — one goes up the other goes down. 1.0 indicates a perfect correlation.

So looking in the first row, first column we see `rank` has a perfect correlation with itself, which is obvious. On the other hand, the correlation between `votes` and `revenue_millions` is 0.6. A little more interesting.

Examining bivariate relationships comes in handy when you have an outcome or dependent variable in mind and would like to see the features most correlated to the increase or decrease of the outcome. You can visually represent bivariate relationships with scatterplots (seen below in the plotting section).

For a deeper look into data summarizations check out [Essential Statistics for Data Science](https://www.learndatasci.com/tutorials/data-science-statistics-using-python/) (<https://www.learndatasci.com/tutorials/data-science-statistics-using-python/>).

Let's now look more at manipulating DataFrames.

DataFrame slicing, selecting, extracting

Up until now we've focused on some basic summaries of our data. We've learned about simple column extraction using single brackets, and we imputed null values in a column using `fillna()`. Below are the other methods of slicing, selecting, and extracting you'll need to use constantly.

It's important to note that, although many methods are the same, DataFrames and Series have different attributes, so you'll need be sure to know which type you are working with or else you will receive attribute errors.

Let's look at working with columns first.

By column

You already saw how to extract a column using square brackets like this:

```
In [34]: genre_col = movies_df['genre']

type(genre_col)
```

Out[34]: pandas.core.series.Series

This will return a *Series*. To extract a column as a *DataFrame*, you need to pass a list of column names. In our case that's just a single column:

```
In [35]: genre_col = movies_df[['genre']]

type(genre_col)
```

Out[35]: pandas.core.frame.DataFrame

Since it's just a list, adding another column name is easy:

```
In [36]: subset = movies_df[['genre', 'rating']]

subset.head()
```

Out[36]:

	genre	rating
Title		
Guardians of the Galaxy	Action,Adventure,Sci-Fi	8.1
Prometheus	Adventure,Mystery,Sci-Fi	7.0
Split	Horror,Thriller	7.3
Sing	Animation,Comedy,Family	7.2
Suicide Squad	Action,Adventure,Fantasy	6.2

Now we'll look at getting data by rows.

By rows

For rows, we have two options:

- `.loc` - **l**ocates by name
- `.iloc` - **l**ocates by numerical index

Remember that we are still indexed by movie Title, so to use `.loc` we give it the Title of a movie:

```
In [37]: prom = movies_df.loc["Prometheus"]
```

```
prom
```

```
Out[37]: rank                2
genre                Adventure,Mystery,Sci-Fi
description    Following clues to the origin of mankind, a te...
director                Ridley Scott
actors    Noomi Rapace, Logan Marshall-Green, Michael Fa...
year                2012
runtime                124
rating                7
votes                485820
revenue_millions    126.46
metascore                65
Name: Prometheus, dtype: object
```

On the other hand, with `iloc` we give it the numerical index of Prometheus:

```
In [38]: prom = movies_df.iloc[1]
```

`loc` and `iloc` can be thought of as similar to Python `list` slicing. To show this even further, let's select multiple rows.

How would you do it with a list? In Python, just slice with brackets like `example_list[1:4]`. It works the same way in pandas:


```
In [39]: movie_subset = movies_df.loc['Prometheus':'Sing']

movie_subset = movies_df.iloc[1:4]

movie_subset
```

Out[39]:

	rank	genre	description	director	actors	year	ru
Title							
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	
Split	3	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	
Sing	4	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	

One important distinction between using `.loc` and `.iloc` to select multiple rows is that `.loc` includes the movie *Sing* in the result, but when using `.iloc` we're getting rows 1:4 but the movie at index 4 (*Suicide Squad*) is not included.

Slicing with `.iloc` follows the same rules as slicing with lists, the object at the index at the end is not included.

Conditional selections

We've gone over how to select columns and rows, but what if we want to make a conditional selection?

For example, what if we want to filter our movies DataFrame to show only films directed by Ridley Scott or films with a rating greater than or equal to 8.0?

To do that, we take a column from the DataFrame and apply a Boolean condition to it. Here's an example of a Boolean condition:

```
In [40]: condition = (movies_df['director'] == "Ridley Scott")
condition.head()
```

```
Out[40]: Title
Guardians of the Galaxy    False
Prometheus                 True
Split                     False
Sing                      False
Suicide Squad              False
Name: director, dtype: bool
```

Similar to `isnull()`, this returns a Series of True and False values: True for films directed by Ridley Scott and False for ones not directed by him.

We want to filter out all movies not directed by Ridley Scott, in other words, we don't want the False films. To return the rows where that condition is True we have to pass this operation into the DataFrame:

```
In [41]: movies_df[movies_df['director'] == "Ridley Scott"].head()
```

```
Out[41]:
```

	rank	genre	description	director	actors	year	runtime	rating
Title								
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.
The Martian	103	Adventure,Drama,Sci-Fi	An astronaut becomes stranded on Mars after hi...	Ridley Scott	Matt Damon, Jessica Chastain, Kristen Wiig, Ka...	2015	144	8.
Robin Hood	388	Action,Adventure,Drama	In 12th century England, Robin and his band of...	Ridley Scott	Russell Crowe, Cate Blanchett, Matthew Macfady...	2010	140	6.
American Gangster	471	Biography,Crime,Drama	In 1970s America, a detective works to bring d...	Ridley Scott	Denzel Washington, Russell Crowe, Chiwetel Eji...	2007	157	7.
Exodus: Gods and Kings	517	Action,Adventure,Drama	The defiant leader Moses rises up against the ...	Ridley Scott	Christian Bale, Joel Edgerton, Ben Kingsley, S...	2014	150	6.

You can get used to looking at these conditionals by reading it like:

Select movies_df where movies_df director equals Ridley Scott

Let's look at conditional selections using numerical values by filtering the DataFrame by ratings:

In [42]: `movies_df[movies_df['rating'] >= 8.6].head(3)`

Out[42]:

	rank	genre	description	director	actors	year	runtime	rat
Title								
Interstellar	37	Adventure,Drama,Sci-Fi	A team of explorers travel through a wormhole ...	Christopher Nolan	Matthew McConaughey, Anne Hathaway, Jessica Ch...	2014	169	
The Dark Knight	55	Action,Crime,Drama	When the menace known as the Joker wreaks havo...	Christopher Nolan	Christian Bale, Heath Ledger, Aaron Eckhart,Mi...	2008	152	
Inception	81	Action,Adventure,Sci-Fi	A thief, who steals corporate secrets through ...	Christopher Nolan	Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen...	2010	148	

We can make some richer conditionals by using logical operators | for "or" and & for "and".

Let's filter the the DataFrame to show only movies by Christopher Nolan OR Ridley Scott:

```
In [43]: movies_df[(movies_df['director'] == 'Christopher Nolan') | (movies_df['director'] == 'Ridley Scott')].head()
```

Out[43]:

	rank	genre	description	director	actors	year	runtime	
Title								
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	
Interstellar	37	Adventure,Drama,Sci-Fi	A team of explorers travel through a wormhole ...	Christopher Nolan	Matthew McConaughey, Anne Hathaway, Jessica Ch...	2014	169	
The Dark Knight	55	Action,Crime,Drama	When the menace known as the Joker wreaks havo...	Christopher Nolan	Christian Bale, Heath Ledger, Aaron Eckhart,Mi...	2008	152	
The Prestige	65	Drama,Mystery,Sci-Fi	Two stage magicians engage in competitive one-...	Christopher Nolan	Christian Bale, Hugh Jackman, Scarlett Johanss...	2006	130	
Inception	81	Action,Adventure,Sci-Fi	A thief, who steals corporate secrets through ...	Christopher Nolan	Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen...	2010	148	

We need to make sure to group evaluations with parentheses so Python knows how to evaluate the conditional.

Using the `isin()` method we could make this more concise though:

```
In [44]: movies_df[movies_df['director'].isin(['Christopher Nolan', 'Ridley Scott'])].head()
```

Out[44]:

	rank	genre	description	director	actors	year	runtime	
Title								
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	
Interstellar	37	Adventure,Drama,Sci-Fi	A team of explorers travel through a wormhole ...	Christopher Nolan	Matthew McConaughey, Anne Hathaway, Jessica Ch...	2014	169	
The Dark Knight	55	Action,Crime,Drama	When the menace known as the Joker wreaks havo...	Christopher Nolan	Christian Bale, Heath Ledger, Aaron Eckhart,Mi...	2008	152	
The Prestige	65	Drama,Mystery,Sci-Fi	Two stage magicians engage in competitive one-...	Christopher Nolan	Christian Bale, Hugh Jackman, Scarlett Johanss...	2006	130	
Inception	81	Action,Adventure,Sci-Fi	A thief, who steals corporate secrets through ...	Christopher Nolan	Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen...	2010	148	

Let's say we want all movies that were released between 2005 and 2010, have a rating above 8.0, but made below the 25th percentile in revenue.

Here's how we could do all of that:

```
In [45]: movies_df[
    ((movies_df['year'] >= 2005) & (movies_df['year'] <= 2010))
    & (movies_df['rating'] > 8.0)
    & (movies_df['revenue_millions'] < movies_df['revenue_millions'].quantile(
0.25))
]
```

Out[45]:

	rank	genre	description	director	actors	year	runtime
Title							
3 Idiots	431	Comedy,Drama	Two friends are searching for their long lost ...	Rajkumar Hirani	Aamir Khan, Madhavan, Mona Singh, Sharman Joshi	2009	170
The Lives of Others	477	Drama,Thriller	In 1984 East Berlin, an agent of the secret po...	Florian Henckel von Donnersmarck	Ulrich Mühe, Martina Gedeck, Sebastian Koch, Ul...	2006	137
Incendies	714	Drama,Mystery,War	Twins journey to the Middle East to discover t...	Denis Villeneuve	Lubna Azabal, Mélissa Désormeaux-Poulin, Maxim...	2010	131
Taare Zameen Par	992	Drama,Family,Music	An eight-year-old boy is thought to be a lazy ...	Aamir Khan	Darsheel Safary, Aamir Khan, Tanay Chheda, Sac...	2007	165

If you recall up when we used `.describe()` the 25th percentile for revenue was about 17.4, and we can access this value directly by using the `quantile()` method with a float of 0.25.

So here we have only four movies that match that criteria.

Brief Plotting

Another great thing about pandas is that it integrates with Matplotlib, so you get the ability to plot directly off DataFrames and Series. To get started we need to import Matplotlib (`pip install matplotlib`):

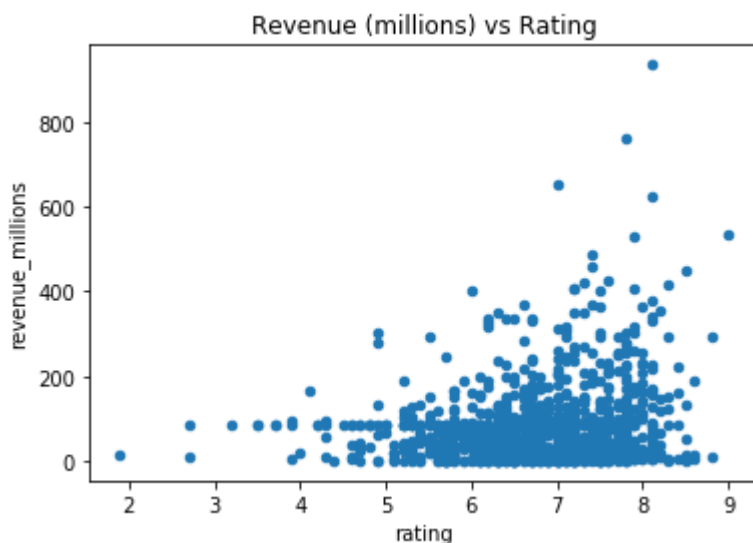
```
In [46]: import matplotlib.pyplot as plt
plt.rcParams.update({'font.size': 20, 'figure.figsize': (10, 8)}) # set font a
nd plot size to be larger
```

Now we can begin. There won't be a lot of coverage on plotting, but it should be enough to explore you're data easily.

Side note: For categorical variables utilize Bar Charts* and Boxplots. For continuous variables utilize Histograms, Scatterplots, Line graphs, and Boxplots.

Let's plot the relationship between ratings and revenue. All we need to do is call `.plot()` on `movies_df` with some info about how to construct the plot:

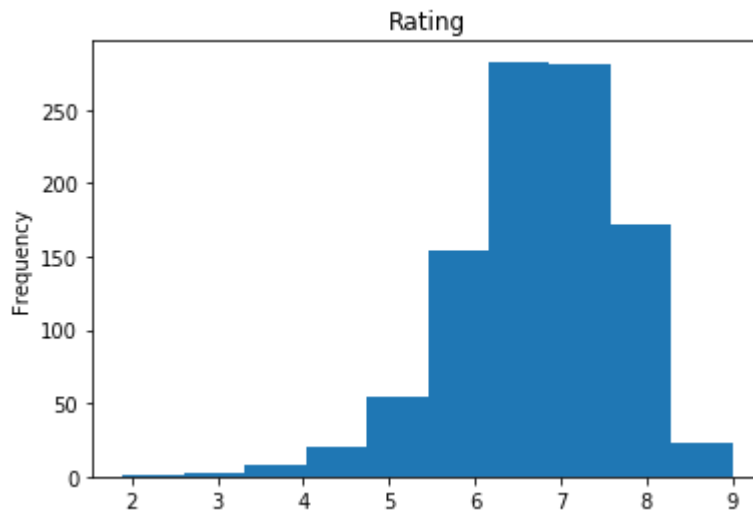
```
In [47]: movies_df.plot(kind='scatter', x='rating', y='revenue_millions', title='Revenue (millions) vs Rating');
```



What's with the semicolon? It's not a syntax error, just a way to hide the `<matplotlib.axes._subplots.AxesSubplot at 0x26613b5cc18>` output when plotting in Jupyter notebooks.

If we want to plot a simple Histogram based on a single column, we can call plot on a column:

```
In [48]: movies_df['rating'].plot(kind='hist', title='Rating');
```



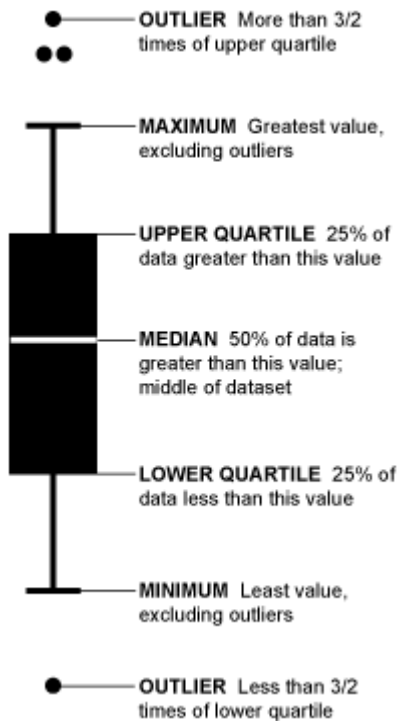
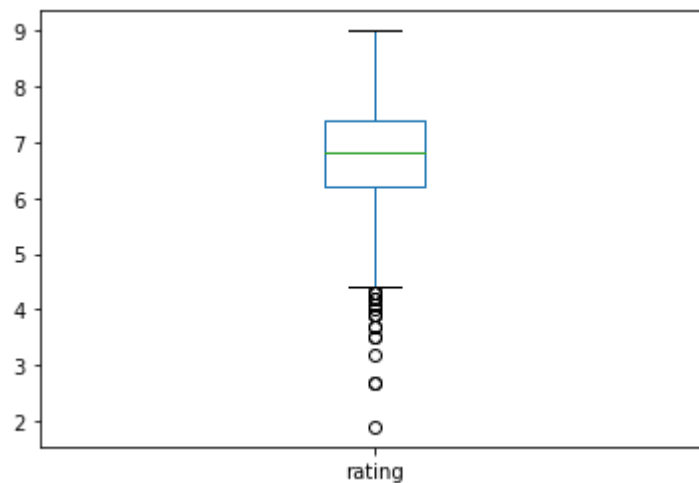
Do you remember the `.describe()` example at the beginning of this tutorial? Well, there's a graphical representation of the interquartile range, called the Boxplot. Let's recall what `describe()` gives us on the ratings column:

```
In [49]: movies_df['rating'].describe()
```

```
Out[49]: count    1000.000000
         mean      6.723200
         std       0.945429
         min       1.900000
         25%       6.200000
         50%       6.800000
         75%       7.400000
         max       9.000000
         Name: rating, dtype: float64
```

Using a Boxplot we can visualize this data:


```
In [50]: movies_df['rating'].plot(kind="box");
```



Source: *Flowing Data*

By combining categorical and continuous data, we can create a Boxplot of revenue that is grouped by the Rating Category we created above: