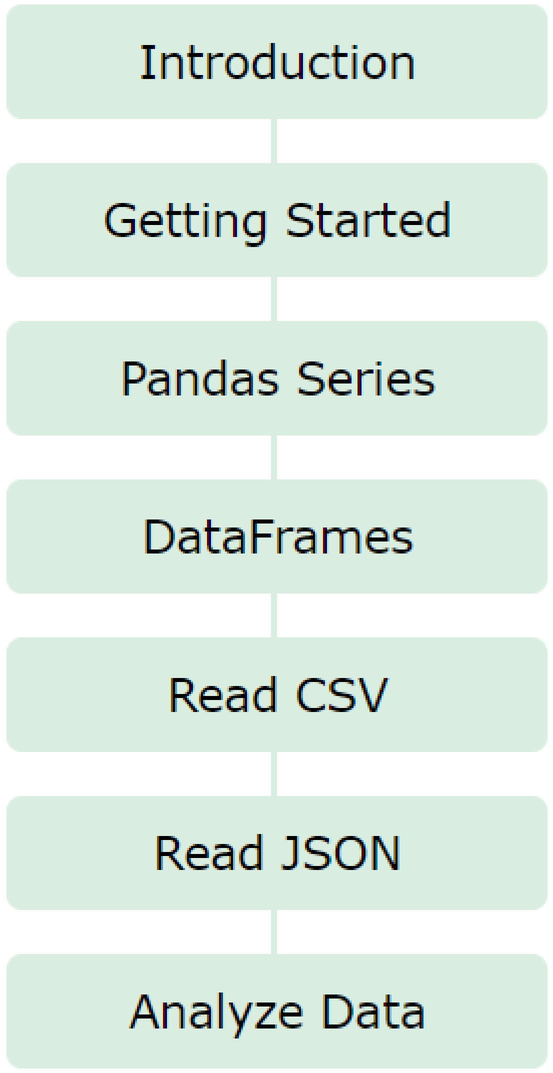
Basic


Introduction
Getting Started
Pandas Series
DataFrames
Read CSV
Read JSON
Analyze Data

Introduction

What is Pandas ?

Pandas is a library specifically for data analysis; it is built using NumPy. It has functions for analyzing, cleaning, exploring, and manipulating data.The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

Why use Pandas?

Data scientists make use of Pandas in Python for its following advantages:

- Easy to handle missing data.
- It uses series for one - dimensional(1D) data structure and DataFrame for multi-dimensional data structure.
- It provides an efficient way to slice the data.
- It provides a flexible way to merge , concatenate , or reshape the data.
- It includes a powerful time series tool to work with.

What Can Pandas Do?

Pandas makes it simple to do many of the time consuming, repetitive tasks associated with working with data, including:

- Data Cleansing
- Data fill
- Data Normalization
- Merges and join
- Data Visualization
- Statistical analysis
- Data Inspection
- Loading and saving data

Getting Started

Installation of Pandas

First, let’s import the Pandas module. We will make an alias of “pandas” as pd because it makes the code a little easy to read and it also avoids any namespace issue.

alias: In Python alias are an alternate name for referring to the same thing.

In [1]:
```
import pandas as pd
```

Now Pandas is imported and ready to use.

In [2]:
```
mydataset = {
    "Books" : ["The Alchemist","The Art of Happiness","The Gift from the Sea"],
    "Ratings" : [4.7,4.7,4.6]
}

data = pd.DataFrame(mydataset)
```

In [3]:
```
print(data)
```

```
                   Books  Ratings
0          The Alchemist      4.7
1   The Art of Happiness      4.7
2  The Gift from the Sea      4.6
```

Checking Pandas Version

In [4]:
```
print(pd.__version__)
```

```
1.0.5
```

Pandas Series

What is a Series?

A series is a one-dimensional data structure. It can have any data structure like integer, float, and string. It is useful when you want to perform computation or return a one-dimensional array. A series, by definition, cannot have multiple columns. For the latter case, please use the data frame structure.

Python Pandas Series has following parameters:

- Data: can be a list, dictionary or scalar value

Example: Create a simple Pandas Series from a list

In [5]:
```
s = pd.Series([1.,2.,3.])
print(s)
```

```
0    1.0
1    2.0
2    3.0
dtype: float64
```

Labels

Note: : In Labels, If nothing else is specified, the values are labeled with their index number. First value has index 0, second value has index 1 etc.

This label can be used to access a specified value.

In [6]:
```
print(s[0])
```

```
1.0
```

Create Labels

With the index argument, you can name your own labels.

In other words, You can add the index with index. It helps to name the rows. The length should be equal to the size of the column

In [7]:
```
s = pd.Series([1.,2.,3.],index=["a","b","c"])
print(s)
```

```
a    1.0
b    2.0
c    3.0
dtype: float64
```

When you have created labels, you can access an item by referring to the label.

In [8]:
```
print(s["b"])
```

```
2.0
```

Key/Value Objects as Series

You can also use a key/value object, like a dictionary, when creating a Series.

Example : Create a simple Pandas Series from a dictionary:

In [9]:
```
bookdata = {
    "item1":"The Alchemist"  ,"item2":"The Art of Happiness","item3":"The Gift from the Sea"
    }

data = pd.Series(bookdata)
```

In [10]:
```
print(data)
```

```
item1             The Alchemist
item2      The Art of Happiness
item3     The Gift from the Sea
dtype: object
```

Note: The keys of the dictionary become the labels.

To select only some of the items in the dictionary, use the index argument and specify only the items you want to include in the Series.

In [11]:
```
bookdata = {
    "item1":"The Alchemist"  ,"item2":"The Art of Happiness","item3":"The Gift from the Sea"
    }

data = pd.Series(bookdata,index=["item1","item2"])
```

In [12]:
```
print(data)
```

```
item1            The Alchemist
item2     The Art of Happiness
dtype: object
```

Pandas DataFrame

A DataFrame in Pandas is a 2-dimensional, labeled data structure which is similar to a SQL Table or a spreadsheet with columns and rows.

Data sets in Pandas are usually multi-dimensional tables, called DataFrames.

Series is like a column, a **DataFrame** is the whole table.

Create a DataFrame from two Series:

In [13]:
```
data = {
    "Classes":["Mathematics","Chemistry","Physics","History","Geography"],
    "Grades" : [90,54,77,22,25]
}
df = pd.DataFrame(data)
```

In [14]:
```
df
```

Out[14]:

	Classes	Grades
0	Mathematics	90
1	Chemistry	54
2	Physics	77
3	History	22
4	Geography	25

Locate Row

As you can see from the result above, the DataFrame is like a table with rows and columns.

Pandas use the **loc** attribute to return one or more specified row(s)

Example : return row 0

In [15]:
```
print(df.loc[0])
```

```
Classes    Mathematics
Grades              90
Name: 0, dtype: object
```

Note: This example return pandas Series

Example : return row 0 and 1

In [16]:
```
print(df.loc[[0,1]])
```

```
       Classes  Grades
0  Mathematics      90
1    Chemistry      54
```

Note: When using [ ], the result is a Pandas DataFrame.

Named Indexes

With the index argument, you can name your own indexes.

Example : Add a list of names to give each row a name:

In [17]:
```
new_df = pd.DataFrame(data,index=["Subject1","Subject2","Subject3","Subject4","Subject5"])
print(new_df)
```

```
              Classes  Grades
Subject1  Mathematics      90
Subject2    Chemistry      54
Subject3      Physics      77
Subject4      History      22
Subject5    Geography      25
```

Locate Named Indexes

Use the named index in the loc attribute to return the specified row(s).

Example : Return "Subject3"

In [18]:
```
print(new_df.loc["Subject3"])
```

```
Classes    Physics
Grades          77
Name: Subject3, dtype: object
```

Load Files Into a DataFrame

If your data sets are stored in a file, Pandas can load them into a DataFrame.

In [19]:
```
df = pd.read_csv("mercedesbenz.csv")
```

In [20]:
```
df
```

Out[20]:

	ID	y	X0	X1	X2	X3	X4	X5	X6	X8	...	X375	X376	X377	X378	X379	X380	X382	X383	X384	X385
0	0	130.81	k	v	at	a	d	u	j	o	...	0	0	1	0	0	0	0	0	0	0
1	6	88.53	k	t	av	e	d	y	l	o	...	1	0	0	0	0	0	0	0	0	0
2	7	76.26	az	w	n	c	d	x	j	x	...	0	0	0	0	0	0	1	0	0	0
3	9	80.62	az	t	n	f	d	x	l	e	...	0	0	0	0	0	0	0	0	0	0
4	13	78.02	az	v	n	f	d	h	d	n	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4204	8405	107.39	ak	s	as	c	d	aa	d	q	...	1	0	0	0	0	0	0	0	0	0
4205	8406	108.77	j	o	t	d	d	aa	h	h	...	0	1	0	0	0	0	0	0	0	0
4206	8412	109.22	ak	v	r	a	d	aa	g	e	...	0	0	1	0	0	0	0	0	0	0
4207	8415	87.48	al	r	e	f	d	aa	l	u	...	0	0	0	0	0	0	0	0	0	0
4208	8417	110.85	z	r	ae	c	d	aa	g	w	...	1	0	0	0	0	0	0	0	0	0

4209 rows × 378 columns

Pandas Read CSV

In Pandas, csv files are read as complete datasets. You do not have to explicitly open and close the dataset. All of the dataset records are assembled into a Dataframe. If your dataset has column headers in the first record then these can be used as the Dataframe column names. You can explicitly state this in the parameters to the call, but pandas is usually able to infer that there ia a header row and use it automatically.

Read CSV Files

In [21]:
```
import pandas as pd

df = pd.read_csv('sales.csv',encoding= 'unicode_escape')


```

In [22]:

```
df
```

Out[22]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	STATUS	QTR_ID	MONTH_ID	YEAR_ID	...
0	10107	30	95.70	2.0	2871.00	2/24/2003 0:00	Shipped	1	2.0	2003.0	...
1	10121	34	81.35	5.0	2765.90	05-07-2003 00:00	Shipped	2	5.0	2003.0	...
2	10134	41	94.74	2.0	3884.34	07-01-2003 00:00	Shipped	3	7.0	2003.0	...
3	10145	45	83.26	6.0	3746.70	8/25/2003 0:00	Shipped	3	8.0	2003.0	...
4	10159	49	100.00	14.0	5205.27	10-10-2003 00:00	Shipped	4	10.0	2003.0	...
...	...	...	...	...	...	...	...	...	...	...	...
2818	10350	20	100.00	15.0	2244.40	12-02-2004 00:00	Shipped	4	12.0	2004.0	...
2819	10373	29	100.00	1.0	3978.51	1/31/2005 0:00	Shipped	1	1.0	2005.0	...
2820	10386	43	100.00	4.0	5417.57	03-01-2005 00:00	Resolved	1	3.0	2005.0	...
2821	10397	34	62.24	1.0	2116.16	3/28/2005 0:00	Shipped	1	3.0	2005.0	...
2822	10414	47	65.52	9.0	3079.44	05-06-2005 00:00	On Hold	2	5.0	2005.0	...

2823 rows × 25 columns

By default, when you print a DataFrame, you will only get the first 5 rows, and the last 5 rows:

Pandas Read JSON

Read JSON

Big data sets are often stored, or extracted as JSON.

JSON is plain text, but has the format of an object, and is well known in the world of programming, including Pandas.

In our examples we will be using a JSON file called 'data.json'.

In [23]:
```
df = pd.read_json("data.json")
```

In [24]:
```
df
```

Out[24]:

	Country	Year	Area (square kilometres)	Total population	Population density, pers. per sq. km	Population aged 0-14, male	Population aged 0-14, female	Population aged 15-64, male	Population aged 15-64, female	Population aged 64+, male	...	Employment in industry and energy (ISIC Rev. 4 B-E), share of total employment	Employmen i constructio (ISIC Rev. F), share c tota employmen
0	Albania	2000	28748.0	3401198.0	118.31077	564851.0	529820.0	1029446.0	1086946.0	82588.0	...	NaN	Nal
1	Albania	2001	28748.0	3073734.0	106.91992	460732.0	436652.0	963612.0	980800.0	108254.0	...	NaN	Nal
2	Albania	2002	28748.0	3093465.0	107.60627	452373.0	427711.0	977294.0	995773.0	112546.0	...	NaN	Nal
3	Albania	2003	28748.0	3111162.0	108.22186	442939.0	417968.0	991075.0	1010693.0	116714.0	...	NaN	Nal
4	Albania	2004	28748.0	3127264.0	108.78197	433178.0	407988.0	1004398.0	1024845.0	120801.0	...	NaN	Nal
...	...	...	...	...	...	...	...	...	...	...	...	...	.
879	Uzbekistan	2012	447400.0	29774448.0	66.54995	4342195.0	4101477.0	10046617.0	10103648.0	516791.0	...	NaN	Nal
880	Uzbekistan	2013	447400.0	30243172.0	67.59761	4384094.0	4133279.0	10238024.0	10286713.0	525998.0	...	NaN	Nal
881	Uzbekistan	2014	447400.0	30757669.0	68.74758	4465526.0	4201156.0	10410214.0	10451289.0	539106.0	...	NaN	Nal
882	Uzbekistan	2015	447400.0	31298929.0	69.95737	4567154.0	4286838.0	10566923.0	10599761.0	561491.0	...	NaN	Nal
883	Uzbekistan	2016	447400.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	Nal

884 rows × 79 columns

Tip: use to_string() to print the entire DataFrame.

Dictionary as JSON

JSON = Python Dictionary

JSON objects have the same format as Python dictionaries.

If your JSON code is not in a file, but in a Python Dictionary, you can load it into a DataFrame directly:

In [25]:
```
new_data = {
  "Duration":{
    "0":60,
    "1":60,
    "2":60,
    "3":45,
    "4":45,
    "5":60
  },
  "Pulse":{
    "0":110,
    "1":117,
    "2":103,
    "3":109,
    "4":117,
    "5":102
  },
  "Maxpulse":{
    "0":130,
    "1":145,
    "2":135,
    "3":175,
    "4":148,
    "5":127
  },
  "Calories":{
    "0":409,
    "1":479,
    "2":340,
    "3":282,
    "4":406,
    "5":300
  }
}

df = pd.DataFrame(new_data)
```

In [26]:
```
df
```

Out[26]:

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409
1	60	117	145	479
2	60	103	135	340
3	45	109	175	282
4	45	117	148	406
5	60	102	127	300

Analyzing DataFrames

I will read the data from the .csv file and perform the following basic operations on movies data

1. Read data
2. View the data
3. Understand some basic information about the data
4. Data Selection – Indexing and Slicing data
5. Data Selection – Based on Conditional filtering
6. Groupby operations
7. Sorting operation
8. Dealing with missing values
9. Dropping columns and null values
10. Apply( ) functions

1. Read data

Load data from CSV file.

In [27]:
```
data = pd.read_csv("IMDB-Movie-Data.csv")
```

In [28]:
```
# Read data with specified explicit index.
# We will use this later in our analysis
data_indexed = pd.read_csv('IMDB-Movie-Data.csv', index_col="Title")
```

In [29]:
```python
data_indexed
```

Out[29]:

	Rank	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
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	121	8.1	757074	333.13	76.0
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	7.0	485820	126.46	65.0
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	117	7.3	157606	138.12	62.0
Sing	4	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	108	7.2	60545	270.32	59.0
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	123	6.2	393727	325.02	40.0
...	...	...	...	...	...	...	...	...	...	...	...
Secret in Their Eyes	996	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	2015	111	6.2	27585	NaN	45.0
Hostel: Part II	997	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	94	5.5	73152	17.54	46.0

	Rank	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
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,...	2008	98	6.2	70699	58.01	50.0
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	5.6	4881	NaN	22.0
Nine Lives	1000	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87	5.3	12435	19.64	11.0

1000 rows × 11 columns

2. View data

One of the most used method for getting a quick overview of the DataFrame, is the head() method.

The head() method returns the headers and a specified number of rows, starting from the top.

In [30]:
```python
data.head()
```

Out[30]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	121	8.1	757074	333.13	76.0
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.0	485820	126.46	65.0
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	117	7.3	157606	138.12	62.0
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	108	7.2	60545	270.32	59.0
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	123	6.2	393727	325.02	40.0

- Returns the top 5 rows in the dataset by default
- It can also take the number of rows to be viewed as a parameter

There is also a tail() method for viewing the last rows of the DataFrame.

The tail() method returns the headers and a specified number of rows, starting from the bottom.

In [31]:
```
data.tail()
```

Out[31]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
995	996	Secret in Their Eyes	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	2015	111	6.2	27585	NaN	45.0
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	94	5.5	73152	17.54	46.0
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...	2008	98	6.2	70699	58.01	50.0
998	999	Search Party	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	2014	93	5.6	4881	NaN	22.0
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87	5.3	12435	19.64	11.0

3. Understand basic information about the data

The DataFrames object has a method called info(), that gives you more information about the data set.

In [32]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Rank                1000 non-null   int64  
 1   Title               1000 non-null   object 
 2   Genre               1000 non-null   object 
 3   Description         1000 non-null   object 
 4   Director            1000 non-null   object 
 5   Actors              1000 non-null   object 
 6   Year                1000 non-null   int64  
 7   Runtime (Minutes)   1000 non-null   int64  
 8   Rating              1000 non-null   float64
 9   Votes               1000 non-null   int64  
 10  Revenue (Millions)  872 non-null    float64
 11  Metascore           936 non-null    float64
dtypes: float64(3), int64(4), object(5)
memory usage: 93.9+ KB
```

Result Explained

- **shape** can be used to get the shape of dataframe
- **columns** gives us the list of columns in the dataframe

In [33]:
```
data.shape
```

Out[33]:
```
(1000, 12)
```

This function tells us that there are 1000 rows and 12 columns in the dataset

- **describe( )** method gives the basic statistical summaries of all numerical attributes in the dataframe.

In [34]:
```python
data.describe()
```

Out[34]:

	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	872.000000	936.000000
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	58.985043
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	103.253540	17.194757
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	11.000000
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	13.270000	47.000000
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	47.985000	59.500000
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	113.715000	72.000000
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	100.000000

Some insights from the description table

- The min and max values in ‘Year’ depict the minimum and maximum release years. We can see that the dataset contains movies from 2006 to 2016.
- The average rating for the movies in this dataset is about 6.7 and the minimum rating is 1.9 and the maximum rating is 9.0
- The maximum revenue earned by a movie is 936.6 million

4. Data Selection – Indexing and Slicing

Extract data using columns

Extracting data from a dataframe is similar to Series. Here the column label is used to extract data from the columns.

Let’s quickly extract ‘Genre’ data from the dataframe

In [35]:
```python
genre = data["Genre"]
```

This operation will retrieve all the data from the ‘Genre’ column as **Series**. If we want to retrieve this data as a dataframe, then indexing must be done using double **square brackets** as below:

In [36]:
```
# Extract data as dataframe

data[["Genre"]]
```

Out[36]:

	Genre
0	Action,Adventure,Sci-Fi
1	Adventure,Mystery,Sci-Fi
2	Horror,Thriller
3	Animation,Comedy,Family
4	Action,Adventure,Fantasy
...	...
995	Crime,Drama,Mystery
996	Horror
997	Drama,Music,Romance
998	Adventure,Comedy
999	Comedy,Family,Fantasy

1000 rows × 1 columns

If we want to extract multiple columns from the data, simply add the column names to the list.

In [37]:
```
cols = data[['Title','Genre','Actors','Director','Rating']]

```

In [38]:
```
cols
```

Out[38]:

	Title	Genre	Actors	Director	Rating
0	Guardians of the Galaxy	Action,Adventure,Sci-Fi	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	James Gunn	8.1
1	Prometheus	Adventure,Mystery,Sci-Fi	Noomi Rapace, Logan Marshall-Green, Michael Fa...	Ridley Scott	7.0
2	Split	Horror,Thriller	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	M. Night Shyamalan	7.3
3	Sing	Animation,Comedy,Family	Matthew McConaughey,Reese Witherspoon, Seth Ma...	Christophe Lourdelet	7.2
4	Suicide Squad	Action,Adventure,Fantasy	Will Smith, Jared Leto, Margot Robbie, Viola D...	David Ayer	6.2
...	...	...	...	...	...
995	Secret in Their Eyes	Crime,Drama,Mystery	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	Billy Ray	6.2
996	Hostel: Part II	Horror	Lauren German, Heather Matarazzo, Bijou Philli...	Eli Roth	5.5
997	Step Up 2: The Streets	Drama,Music,Romance	Robert Hoffman, Briana Evigan, Cassie Ventura,...	Jon M. Chu	6.2
998	Search Party	Adventure,Comedy	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	Scot Armstrong	5.6
999	Nine Lives	Comedy,Family,Fantasy	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	Barry Sonnenfeld	5.3

1000 rows × 5 columns

loc and iloc are two functions that can be used to slice data from specific row indexes.

loc – locates the rows by name

loc performs slicing based explicit index. It takes string indexes to retrieve data from specified rows **iloc** – locates the rows by integer index

iloc performs slicing based on Python’s default numerical index. In the beginning, when we read the data, we created a dataframe with **‘Title’** as the string index.

We will use the **loc** function to index and slice that dataframe using the specified ‘Title’.

In [39]:
```
data_indexed.loc[["Captain America: Civil War"]][["Genre","Actors","Director","Rating","Revenue (Millions)"]]
```

Out[39]:

Title	Genre	Actors	Director	Rating	Revenue (Millions)
Captain America: Civil War	Action,Adventure,Sci-Fi	Chris Evans, Robert Downey Jr.,Scarlett Johans...	Anthony Russo	7.9	408.08

Here, iloc is used to slice data using integer indexes.

In [40]:
```
data_indexed.iloc[15:20][["Director","Rating","Revenue (Millions)"]]
```

Out[40]:

Title	Director	Rating	Revenue (Millions)
The Secret Life of Pets	Chris Renaud	6.6	368.31
Hacksaw Ridge	Mel Gibson	8.2	67.12
Jason Bourne	Paul Greengrass	6.7	162.16
Lion	Garth Davis	8.1	51.69
Arrival	Denis Villeneuve	8.0	100.50

5. Data Selection – Based on Conditional Filtering

Pandas also enable retrieving data from dataframe based on conditional filters.

What if we want to pick only movies that are released from 2010 to 2016, have a rating of less than 6.0 but topped in terms of revenue?

In [55]:

```
data[((data['Year'] >= 2010) & (data['Year'] <= 2016))
      & (data['Rating'] < 6.0)
      & (data['Revenue (Millions)'] > data['Revenue (Millions)'].quantile(0.95))]
```

Out[55]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
941	942	The Twilight Saga: Eclipse	Adventure,Drama,Fantasy	As a string of mysterious killings grips Seatt...	David Slade	Kristen Stewart, Robert Pattinson, Taylor Laut...	2010	124	4.9	192740	300.52	58.0

‘The Twilight Saga: Breaking Dawn – Part 2′ and ‘The Twilight Saga: Eclipse’ are the movies that topped in the box office, despite having lower ratings.

6. Groupby

Pandas groupby is used for grouping the data according to the categories and apply a function to the categories. It also helps to aggregate data efficiently.

Pandas dataframe.groupby() function is used to split the data into groups based on some criteria. pandas objects can be split on any of their axes. The abstract definition of grouping is to provide a mapping of labels to group names.

In [59]:
```
data.groupby("Director")[["Rating"]].mean().head()
```

Out[59]:

Director	Rating
Aamir Khan	8.5
Abdellatif Kechiche	7.8
Adam Leon	6.5
Adam McKay	7.0
Adam Shankman	6.3

7. Sorting Operation

sort_values( ) method is used to perform sorting operation on a column or a list of multiple columns

In the above example, where we have listed the average rating for each 'Director', if we want to sort them from highly rated to lowest, we can perform the sorting operation.

In [60]:
```
data.groupby("Director")[["Rating"]].mean().sort_values(["Rating"],ascending=False).head()
```

Out[60]:

Director	Rating
Nitesh Tiwari	8.80
Christopher Nolan	8.68
Makoto Shinkai	8.60
Olivier Nakache	8.60
Florian Henckel von Donnersmarck	8.50

8. Dealing with missing values

Pandas has isnull( ) for detecting null values in a dataframe.

In [61]:
```
# to check null values

data.isnull().sum()
```

Out[61]:
```
Rank                    0
Title                   0
Genre                   0
Description             0
Director                0
Actors                  0
Year                    0
Runtime (Minutes)       0
Rating                  0
Votes                   0
Revenue (Millions)    128
Metascore              64
dtype: int64
```

Here we know that ‘Revenue (Millions)’ and ‘Metascore’ are two columns where there are null values.

As we have seen null values in data, we can either choose to drop those or impute these values

9. Dropping columns and null values

Dropping columns/rows is yet another operation that is most important for data analysis.

drop( ) function can be used to drop rows or columns based on condition

In [62]:
```
data.drop('Metascore', axis=1)
```

Out[62]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	121	8.1	757074	333.13
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.0	485820	126.46
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	117	7.3	157606	138.12
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	108	7.2	60545	270.32
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	123	6.2	393727	325.02
...	...	...	...	...	...	...	...	...	...	...	...
995	996	Secret in Their Eyes	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	2015	111	6.2	27585	NaN
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	94	5.5	73152	17.54
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...	2008	98	6.2	70699	58.01
998	999	Search Party	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	2014	93	5.6	4881	NaN
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87	5.3	12435	19.64

```
1000 rows × 11 columns
```

Using the above code, the ‘Metascore’ column is dropped completely from data. Here axis= 1 specifies that column is to be dropped. These changes will not take place in actual data unless we specify inplace=True as a parameter in the drop( ) function.

We can also drop rows/ columns with null values by using dropna( ) function.

In [64]:
```
## Drops all rows containing missing data

data.dropna()
```

Out[64]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	121	8.1	757074	333.13	76.0
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.0	485820	126.46	65.0
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	117	7.3	157606	138.12	62.0
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	108	7.2	60545	270.32	59.0
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	123	6.2	393727	325.02	40.0
...	...	...	...	...	...	...	...	...	...	...	...	...
993	994	Resident Evil: Afterlife	Action,Adventure,Horror	While still out to destroy the evil Umbrella C...	Paul W.S. Anderson	Milla Jovovich, Ali Larter, Wentworth Miller,K...	2010	97	5.9	140900	60.13	37.0
994	995	Project X	Comedy	3 high school seniors throw a birthday party t...	Nima Nourizadeh	Thomas Mann, Oliver Cooper, Jonathan Daniel Br...	2012	88	6.7	164088	54.72	48.0

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	94	5.5	73152	17.54	46.0
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...	2008	98	6.2	70699	58.01	50.0
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87	5.3	12435	19.64	11.0

838 rows × 12 columns

In [65]:

```
# Drop all columns containing missing data
data.dropna(axis=1)
```

Out[65]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	121	8.1	757074
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.0	485820
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016	117	7.3	157606
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016	108	7.2	60545
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016	123	6.2	393727
...	...	...	...	...	...	...	...	...	...	...
995	996	Secret in Their Eyes	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	2015	111	6.2	27585
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	94	5.5	73152
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...	2008	98	6.2	70699
998	999	Search Party	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	2014	93	5.6	4881
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87	5.3	12435

1000 rows × 10 columns

In our movies data, we know that there are some records where the Revenue is null. We can impute these null values with mean Revenue (Millions).

fillna( ) –> function used to fill null values with specified values

In [68]:
```
rev_mean = data_indexed["Revenue (Millions)"].mean()
print(rev_mean)
```

```
82.95637614678897
```

In [69]:
```
# We can fill the null values with this mean revenue
data_indexed['Revenue (Millions)'].fillna(rev_mean, inplace=True)
```

Now, if we check the dataframe, there won't be any null values in the Revenue column

In [71]:
```
data_indexed.isnull().sum()
```

Out[71]:
```
Rank                   0
Genre                  0
Description            0
Director               0
Actors                 0
Year                   0
Runtime (Minutes)      0
Rating                 0
Votes                  0
Revenue (Millions)     0
Metascore             64
dtype: int64
```

10. Apply( ) function

The apply( ) function comes in handy when we want to apply any function to the dataset. It returns a value after passing each row of the dataframe to some function. The function can be built-in or user-defined.

For example, if we want to classify the movies based on their ratings, we can define a function to do so and then apply the function to the dataframe as shown below.

I will write a function that will classify movies into groups based on rating.

In [72]:
```
# classify movies based on ratings

def rating_group(rating):
    if rating>=7.5:
        return "Good"
    if rating>=6.0:
        return "Average"
    else:
        return "Bad"
```

Now, I will apply this function to our actual dataframe and the ‘Rating_category’ will be computed for each row.

In [73]:
```
# creating a new variable in the dataset to hold the rating category

data["Rating_category"] = data["Rating"].apply(rating_group)
```

Here is the resultant data after applying the rating_group( ) function.

In [74]:
```
data[['Title','Director','Rating','Rating_category']].head(10)
```

Out[74]:
```
                 Title             Director  Rating  Rating_category
0  Guardians of the Galaxy         James Gunn     8.1             Good
1               Prometheus       Ridley Scott     7.0          Average
2                    Split  M. Night Shyamalan     7.3          Average
3                     Sing Christophe Lourdelet    7.2          Average
4            Suicide Squad         David Ayer     6.2          Average
5           The Great Wall        Yimou Zhang     6.1          Average
6               La La Land    Damien Chazelle     8.3             Good
7                 Mindhorn         Sean Foley     6.4          Average
8        The Lost City of Z         James Gray     7.1          Average
9               Passengers      Morten Tyldum     7.0          Average
```