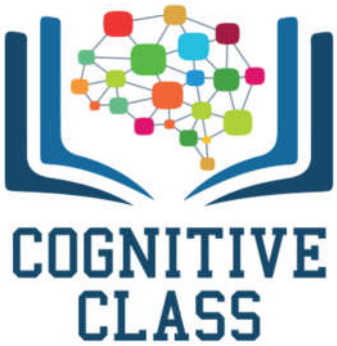
(https://cocl.us/PY0101EN_edx_add_top)



(https://cognitiveclass.ai/)

Introduction to Pandas Python

Welcome! This notebook will teach you about using `Pandas` in the Python Programming Language. By the end of this lab, you'll know how to use `Pandas` package to view and access data.

Table of Contents

Estimated time needed: **15 min**

About the Dataset

The table has one row for each album and several columns

- **artist**: Name of the artist
- **album**: Name of the album
- **released_year**: Year the album was released
- **length_min_sec**: Length of the album (hours,minutes,seconds)
- **genre**: Genre of the album
- **music_recording_sales_millions**: Music recording sales (millions in USD) on [SONG://DATABASE] (http://www.song-database.com/)
- **claimed_sales_millions**: Album's claimed sales (millions in USD) on [SONG://DATABASE] (http://www.song-database.com/)
- **date_released**: Date on which the album was released
- **soundtrack**: Indicates if the album is the movie soundtrack (Y) or (N)
- **rating_of_friends**: Indicates the rating from your friends from 1 to 10

You can see the dataset here:

<table font-size:xx-small style="width:25%"> Artist Album Released Length Genre Music recording sales (millions) Claimed sales (millions) Released Soundtrack Rating (friends) Michael Jackson Thriller 1982 00:42:19 Pop, rock, R&B 46 65 30-Nov-82 10.0 AC/DC Back in Black 1980 00:42:11 Hard rock 26.1 50 25-Jul-80 8.5 Pink Floyd The Dark Side of the Moon 1973 00:42:49 Progressive rock 24.2 45 01-Mar-73 9.5 Whitney Houston The Bodyguard 1992 00:57:44 Soundtrack/R&B, soul, pop 26.1 50 25-Jul-80 Y 7.0 Meat Loaf Bat Out of Hell 1977 00:46:33 Hard rock, progressive rock 20.6 43 21-Oct-77 7.0 Eagles Their Greatest Hits (1971-1975) 1976 00:43:08 Rock, soft rock, folk rock 32.2 42 17-Feb-76 9.5 Bee Gees Saturday Night Fever 1977 1:15:54 Disco 20.6 40 15-Nov-77 Y 9.0 Fleetwood Mac Rumours 1977 00:40:01 Soft rock 27.9 40 04-Feb-77 9.5 </table>

Introduction of `Pandas`

In [8]:

```
# Dependency needed to install file

!pip install xlrd
!pip install openpyxl
```

```
Requirement already satisfied: xlrd in /opt/conda/envs/Python-3.8-main/lib/python3.8/site-packages (2.0.1)
Collecting openpyxl
  Downloading openpyxl-3.0.7-py2.py3-none-any.whl (243 kB)
     |████████████████████████████████| 243 kB 12.0 MB/s eta 0:00:01
Collecting et-xmlfile
  Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Installing collected packages: et-xmlfile, openpyxl
Successfully installed et-xmlfile-1.1.0 openpyxl-3.0.7
```

In [9]:

```
# Import required library

import pandas as pd
```

After the import command, we now have access to a large number of pre-built classes and functions. This assumes the library is installed; in our lab environment all the necessary libraries are installed. One way pandas allows you to work with data is a dataframe. Let's go through the process to go from a comma separated values (**.csv**) file to a dataframe. This variable `csv_path` stores the path of the **.csv**, that is used as an argument to the `read_csv` function. The result is stored in the object `df` , this is a common short form used for a variable referring to a Pandas dataframe.

In [10]:

```
# Read data from CSV file

csv_path = 'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/PY0101EN/Chapter%204/Datasets/TopSellingAlbums.csv'
df = pd.read_csv(csv_path)
```

We can use the method `head()` to examine the first five rows of a dataframe:

In [11]:

```
# Print first five rows of the dataframe

df.head()
```

Out[11]:

	Artist	Album	Released	Length	Genre	Music Recording Sales (millions)	Claimed Sales (millions)	Released.1	Soundtra
0	Michael Jackson	Thriller	1982	0:42:19	pop, rock, R&B	46.0	65	30-Nov-82	Na
1	AC/DC	Back in Black	1980	0:42:11	hard rock	26.1	50	25-Jul-80	Na
2	Pink Floyd	The Dark Side of the Moon	1973	0:42:49	progressive rock	24.2	45	01-Mar-73	Na
3	Whitney Houston	The Bodyguard	1992	0:57:44	R&B, soul, pop	27.4	44	17-Nov-92	
4	Meat Loaf	Bat Out of Hell	1977	0:46:33	hard rock, progressive rock	20.6	43	21-Oct-77	Na

We use the path of the excel file and the function `read_excel` . The result is a data frame as before:

In [13]:

```
# Read data from Excel File and print the first five rows

xlsx_path = 'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/PY0101EN/Chapter%204/Datasets/TopSellingAlbums.xlsx'

df = pd.read_excel(xlsx_path)
df.head()
```

Out[13]:

	Artist	Album	Released	Length	Genre	Music Recording Sales (millions)	Claimed Sales (millions)	Released.1	Soundtr
0	Michael Jackson	Thriller	1982	00:42:19	pop, rock, R&B	46.0	65	1982-11-30	N
1	AC/DC	Back in Black	1980	00:42:11	hard rock	26.1	50	1980-07-25	N
2	Pink Floyd	The Dark Side of the Moon	1973	00:42:49	progressive rock	24.2	45	1973-03-01	N
3	Whitney Houston	The Bodyguard	1992	00:57:44	R&B, soul, pop	27.4	44	1992-11-17	
4	Meat Loaf	Bat Out of Hell	1977	00:46:33	hard rock, progressive rock	20.6	43	1977-10-21	N

We can access the column **Length** and assign it a new dataframe **x**:

In [14]:

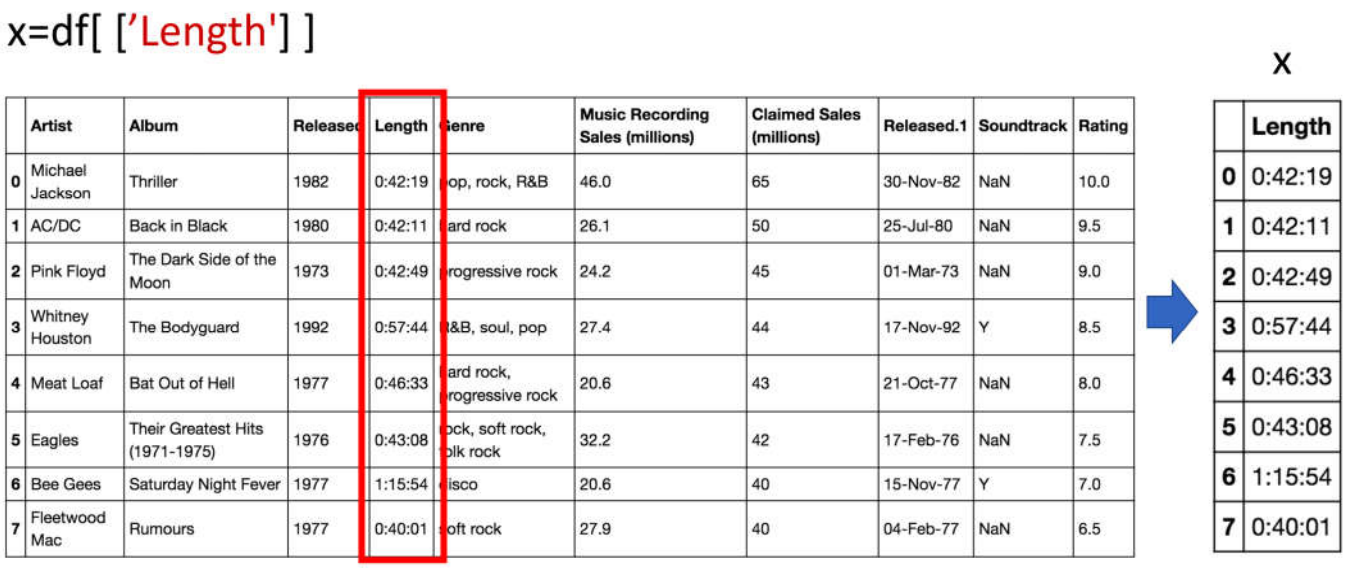
```
# Access to the column Length

x = df[['Length']]
x
```

Out[14]:

	Length
0	00:42:19
1	00:42:11
2	00:42:49
3	00:57:44
4	00:46:33
5	00:43:08
6	01:15:54
7	00:40:01

The process is shown in the figure:



	Artist	Album	Released	Length	Genre	Music Recording Sales (millions)	Claimed Sales (millions)	Released.1	Soundtrack	Rating
0	Michael Jackson	Thriller	1982	0:42:19	pop, rock, R&B	46.0	65	30-Nov-82	NaN	10.0
1	AC/DC	Back in Black	1980	0:42:11	hard rock	26.1	50	25-Jul-80	NaN	9.5
2	Pink Floyd	The Dark Side of the Moon	1973	0:42:49	progressive rock	24.2	45	01-Mar-73	NaN	9.0
3	Whitney Houston	The Bodyguard	1992	0:57:44	R&B, soul, pop	27.4	44	17-Nov-92	Y	8.5
4	Meat Loaf	Bat Out of Hell	1977	0:46:33	hard rock, progressive rock	20.6	43	21-Oct-77	NaN	8.0
5	Eagles	Their Greatest Hits (1971-1975)	1976	0:43:08	rock, soft rock, folk rock	32.2	42	17-Feb-76	NaN	7.5
6	Bee Gees	Saturday Night Fever	1977	1:15:54	disco	20.6	40	15-Nov-77	Y	7.0
7	Fleetwood Mac	Rumours	1977	0:40:01	soft rock	27.9	40	04-Feb-77	NaN	6.5

	Length
0	0:42:19
1	0:42:11
2	0:42:49
3	0:57:44
4	0:46:33
5	0:43:08
6	1:15:54
7	0:40:01

Viewing Data and Accessing Data

You can also get a column as a series. You can think of a Pandas series as a 1-D dataframe. Just use one bracket:

In [15]:

```
# Get the column as a series

x = df['Length']
x
```

Out[15]:

```
0    00:42:19
1    00:42:11
2    00:42:49
3    00:57:44
4    00:46:33
5    00:43:08
6    01:15:54
7    00:40:01
Name: Length, dtype: object
```

You can also get a column as a dataframe. For example, we can assign the column **Artist**:

In [16]:

```
# Get the column as a dataframe

x = type(df[['Artist']])
x
```

Out[16]:

```
pandas.core.frame.DataFrame
```

You can do the same thing for multiple columns; we just put the dataframe name, in this case, `df` , and the name of the multiple column headers enclosed in double brackets. The result is a new dataframe comprised of the specified columns:

In [17]:

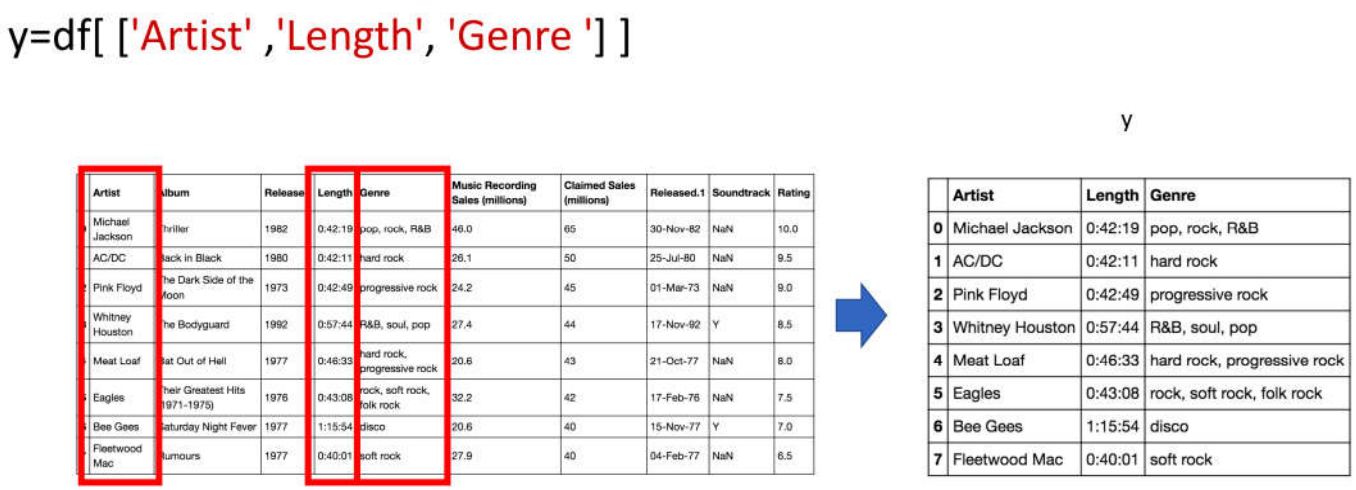
```
# Access to multiple columns

y = df[['Artist','Length','Genre']]
y
```

Out[17]:

	Artist	Length	Genre
0	Michael Jackson	00:42:19	pop, rock, R&B
1	AC/DC	00:42:11	hard rock
2	Pink Floyd	00:42:49	progressive rock
3	Whitney Houston	00:57:44	R&B, soul, pop
4	Meat Loaf	00:46:33	hard rock, progressive rock
5	Eagles	00:43:08	rock, soft rock, folk rock
6	Bee Gees	01:15:54	disco
7	Fleetwood Mac	00:40:01	soft rock

The process is shown in the figure:



One way to access unique elements is the `iloc` method, where you can access the 1st row and the 1st column as follows:

In [18]:

```
# Access the value on the first row and the first column

df.iloc[0, 0]
```

Out[18]:

```
'Michael Jackson'
```

You can access the 2nd row and the 1st column as follows:

In [19]:

```
# Access the value on the second row and the first column

df.iloc[1,0]
```

Out[19]:

```
'AC/DC'
```

You can access the 1st row and the 3rd column as follows:

In [20]:

```
# Access the value on the first row and the third column

df.iloc[0,2]
```

Out[20]:

```
1982
```

You can access the column using the name as well, the following are the same as above:

In [21]:

```
# Access the column using the name

df.loc[0, 'Artist']
```

Out[21]:

```
'Michael Jackson'
```

In [22]:

```
# Access the column using the name

df.loc[1, 'Artist']
```

Out[22]:

```
'AC/DC'
```

In [23]:

```
# Access the column using the name

df.loc[0, 'Released']
```

Out[23]:

```
1982
```

In [24]:

```
# Access the column using the name

df.loc[1, 'Released']
```

Out[24]:

```
1980
```

You can perform slicing using both the index and the name of the column:

In [25]:

```
# Slicing the dataframe

df.iloc[0:2, 0:3]
```

Out[25]:

	Artist	Album	Released
0	Michael Jackson	Thriller	1982
1	AC/DC	Back in Black	1980

In [26]:

```
# Slicing the dataframe using name

df.loc[0:2, 'Artist':'Released']
```

Out[26]:

	Artist	Album	Released
0	Michael Jackson	Thriller	1982
1	AC/DC	Back in Black	1980
2	Pink Floyd	The Dark Side of the Moon	1973

Quiz on DataFrame

Use a variable `q` to store the column **Rating** as a dataframe

In [27]:

```
# Write your code below and press Shift+Enter to execute
q = df[['Rating']]
q
```

Out[27]:

	Rating
0	10.0
1	9.5
2	9.0
3	8.5
4	8.0
5	7.5
6	7.0
7	6.5

Double-click **here** for the solution.

Assign the variable `q` to the dataframe that is made up of the column **Released** and **Artist**:

In [28]:

```
# Write your code below and press Shift+Enter to execute
q = df[['Released', 'Artist']]
q
```

Out[28]:

	Released	Artist
0	1982	Michael Jackson
1	1980	AC/DC
2	1973	Pink Floyd
3	1992	Whitney Houston
4	1977	Meat Loaf
5	1976	Eagles
6	1977	Bee Gees
7	1977	Fleetwood Mac

Double-click **here** for the solution.

Access the 2nd row and the 3rd column of `df` :

In [29]:

```
# Write your code below and press Shift+Enter to execute
df.iloc[2,3]
```

Out[29]:

```
datetime.time(0, 42, 49)
```

Double-click **here** for the solution.

The last exercise!

Congratulations, you have completed your first lesson and hands-on lab in Python. However, there is one more thing you need to do. The Data Science community encourages sharing work. The best way to share and showcase your work is to share it on GitHub. By sharing your notebook on GitHub you are not only building your reputation with fellow data scientists, but you can also show it off when applying for a job. Even though this was your first piece of work, it is never too early to start building good habits. So, please read and follow this article (https://cognitiveclass.ai/blog/data-scientists-stand-out-by-sharing-your-notebooks/) to learn how to share your work.

Get IBM Watson Studio free of charge!

(https://cocl.us/PY0101EN_edx_add_bbottom)

About the Authors:

Joseph Santarcangelo (https://www.linkedin.com/in/joseph-s-50398b136/) is a Data Scientist at IBM, and holds a PhD in Electrical Engineering. His research focused on using Machine Learning, Signal Processing, and Computer Vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Other contributors: Mavis Zhou (www.linkedin.com/in/jiahui-mavis-zhou-a4537814a)

Copyright © 2018 IBM Developer Skills Network. This notebook and its source code are released under the terms of the MIT License (https://cognitiveclass.ai/mit-license/).