Introducción a la Programación con PYTHON

José Rodriguez

Unidad 13 - Gestión de datos con Pandas

UNIDAD 13 – LIBRERIAS PANDAS

ÍNDICE

1.- Librería Pandas

Pandas es una biblioteca de código abierto de Python que proporciona análisis y manipulación de datos en la programación en Python.

Es una biblioteca muy completa de representación de datos, filtrado y programación estadística. La pieza más importante en pandas es el DataFrame donde almacena y juega con los datos.

Para poder usarla, primero debemos instalarla

>> pip install pandas

En la mayoría de los casos necesitaremos otra librería para poder leer datos desde diferentes fuentes como excell, CSV, etc.

Instalamos la librería xlrd

>> pip install xlrd

Successfully installed xlrd-2.0.1

Puedes leer desde un archivo de Excel usando el método read_excel () de pandas (En realidad se encuentra en xlrd)

Si trabajamos con ficheros xlsx en lugar de xls debemos instalar una nueva librería

>> pip install openpyxl

Successfully built et-xmlfile
Installing collected packages: jdcal, et-xmlfile, openpyxl
Successfully installed et-xmlfile-1.0.1 jdcal-1.4.1 openpyxl-3.0.5

2.- Lectura de ficheros Excell

Ya estamos en condiciones de manejar ficheros excel. Para empezar supongamos la siguiente hoja de datos

	A	B	C
1	Módulo	Profesor	Aula
2	ASO	Alejandro de la Torre	A34
3	ISO	Alejandro de la Torre	A33
4	LM	José Rodriguez	A33
5	SBGD	Francisco Soria	A33
6	SAD	Antonio Torres	A34
7	RAL	Antonio Torres	A33

Se trata de un fichero 'c:/ASIR.xlsx' con una hoja llamada ASIR. Fíjate bien en los nombres, Python es sensible a mayúsculas y debemos escribirlo tal y como aparecen en excell.
El código para leer el fichero es

```
import pandas
print(pandas.read_excel ('c:/ASIR.xlsx','ASIR'))
```

El resultado sería

```
  Módulo          Profesor                 Aula
0   ASO      Alejandro de la Torre  A34
1   ISO      Alejandro de la Torre  A33
2    LM      José Rodriguez                 A33
3  SBGD  Francisco Soria          A33
4   SAD   Antonio Torres          A34
5   RAL   Antonio Torres          A33
```

Este resultado es llamado **DataFrame**. Esa es la unidad básica de pandas con la que vamos a tratar.

El DataFrame es una estructura de 2 dimensiones etiquetada donde podemos almacenar datos de diferentes tipos. DataFrame es similar a una tabla SQL o una hoja de cálculo de Excel.
El resultado obtenido en la lectura del ejemplo anterior lo podemos convertir en un DataFrame

```
import pandas
x=pandas.read_excel ('c:/ASIR.xlsx','ASIR')
df=pandas.DataFrame(x)
print(len(df))
```

Ahora, df es un array del cual podemos obtener su tamaño con la propiedad len(). Para movernos por este array usamos la propiedad loc[indice]. Por ejemplo

```
print(df.loc[1])
```

nos devolverá la fila segunda (con índice 1) en el siguiente formato

Módulo ISO
Profesor Alejandro de la Torre
Aula A33

Se trata de un array bidimensional. La primera dimensión es la fila de la hoja excell y la segunda dimensión es la columna. Por ejemplo

```
a= (df.loc[0][0])
print(a)
```

Nos devolverá la primera columna de la primera fila, es decir, ASO. Esta nomenclatura nos permite procesar los datos de una hoja de cálculo con bastante flexibilidad.
En el siguiente ejemplo mostramos los nombres de los profesores que imparten los módulos recorriendo con un bucle todas las filas de la tabla y mostrando sólo la columna correspondiente al profesor.

```
l=len(df)
i=0
while i<l:
    a= (df.loc[i][1])
    print(a)
    i=i+1
```

El resultado sería

Alejandro de la Torre
Alejandro de la Torre
José Rodriguez
Francisco Soria
Antonio Torres
Antonio Torres

En muchas ocasiones los campos pueden aparecer repetidos en algunas de sus columnas. Por ejemplo, en nuestro caso hay dos profesores que imparten varios módulos y por tanto aparecen dos veces. La función duplicates() devuelve true o false según que la columna que estemos procesando aparezca en varios registros.

```
a= (df.duplicated(['Profesor']))
print(a)
```

El resultado será

0 False
1 True
2 False
3 False
4 False
5 True

Si queremos obtener una lista de los profesores que imparten módulos queda muy mal que aparezcan estas repeticiones. Para ello utilizamos el método drop_duplicates() que elimina estas repeticiones

```
a= (df.drop_duplicates(['Profesor']))
print(a)
```

quedando

	Módulo	Profesor	Aula
0	ASO	Alejandro de la Torre	A34
2	LM	José Rodriguez	A33
3	SBGD	Francisco Soria	A33
4	SAD	Antonio Torres	A34

Si te fijas, al borrar algunos registros, el índice da cada registro se mantiene con lo cual no podríamos recorrer el array con un bucle ya que, por ejemplo, el índice 1 ya no está.

también, podemos contarlos

```
a= (df.drop_duplicates(['Profesor']))
l=len(a)
print('hay ', l, ' Profesors')
```

Para borrar una fila completa se utiliza la función drop()

```
a=df.drop([1])
print(a)
```

El resultado es ahora

	Módulo	Profesor	Aula
0	ASO	Alejandro de la Torre	A34
2	LM	José Rodriguez	A33
3	SBGD	Francisco Soria	A33
4	SAD	Antonio Torres	A34
5	RAL	Antonio Torres	A33

Como puedes comprobar, el índice 1 ya no está.

También puedes eliminar un rango de filas de la siguiente forma:

>>> df.drop(df.index[[0, 1]])

Esto eliminará las filas del índice 0 a 1.

Aplicado a nuestro ejemplo, el siguiente código

```
a=df.drop(df.index[[0, 1]])
print(a)
```

borra las filas 0 y 1 quedando como resultado

```
  Módulo        Profesor Aula
2    LM     José Rodriguez    A33
3   SBGD  Francisco Soria  A33
4   SAD     Antonio Torres    A34
5   RAL     Antonio Torres    A33
```

Supongamos ahora que la hoja excell tiene una columnas más con el número de alumnos matriculados en cada módulo

	A	B	C	D
1	Módulo	Profesor	Aula	Alumnos
2	ASO	Alejandro de la Torre	A34	22
3	ISO	Alejandro de la Torre	A33	21
4	LM	José Rodriguez	A33	20
5	SBGD	Francisco Soria	A33	23
6	SAD	Antonio Torres	A34	21
7	RAL	Antonio Torres	A33	22

Podemos utilizar la función sum() para sumar la columna de alumnos matriculados

```
a=df['Alumnos'].sum()
print('Hay ', a,'Alumnos matriculados')
```

y tendremos como resultado

```
Hay  129 Alumnos matriculados
```

Resulta evidente que esta operación la podríamos haber hecho directamente en Excell colocando la función sumatorio.

Aula	Alumnos
A34	22
A33	21
A33	20
A33	23
A34	21
A33	22
	129

La función count() cuenta el número de registros

```
a=df['Módulo'].count()
print('Hay ', a,'Módulos')
```

En este ejemplo contamos el número de módulos que se imparten. Si intentásemos aplicar este código al campo profesor daría un error ya que hay profesores que repiten módulo

```
a=df['Profesor'].count()
print('Hay ', a,'Profesores')
```

El resultado sería Hay 6 profesores.

Para contar el número de profesores deberíamos hacer

```
a=df['Profesor'].unique()
print('Hay ', len(a),'Profesores')
```

y el resultado sería correcto

Hay 4 Profesores

3.- Creando DataFrame

En los ejemplos anteriores hemos creado objetos DataFrame a partir de ficheros Excell. La librería 'pandas' nos ofrece la posibilidad de crear objetos de este tipo desde el propio script a partir de una lista.

```
import pandas
alumnos = [ ('Antonio Perez', 18, '30456756Y' ) , ('Maribel Luna', 25,
'34567564K') , ('Nuria Sanchez', 18, '30456123J' ) ]
#Creamos un objeto DataFrame
df = pandas.DataFrame(alumnos, columns = ['Nombre' , 'edad', 'DNI'],
index=['0', '1', '2'])
print(df)
```

la función tiene tres parámetros. El primer parámetro es la lista que contiene los datos, el segundo es la cabecera y el tercero son los índices que llevarán cada uno de los registros.

El resultado sería

```
      Nombre  edad       DNI
0  Antonio Perez    18  30456756Y
1   Maribel Luna    25  34567564K
2  Nuria Sanchez    18  30456123J
```

Una vez creado el DF podemos insertar nuevas filas como si se tratara de un array

```
import pandas
alumnos = [ ('Antonio Perez', 18, '30456756Y' ) , ('Maribel Luna', 25,
'34567564K') , ('Nuria Sanchez', 18, '30456123J' ) ]
#Creamos un objeto DataFrame
df = pandas.DataFrame(alumnos, columns = ['Nombre' , 'edad', 'DNI'],
index=['0', '1', '2'])
df.loc[3] = [ 'Antonio Perez', 18, '30456756Y'  ]
print(df)
```

Hemos insertado un nuevo registro definiendo directamente el índice que ocupa con df.loc[3] y asignándole valores en el orden que habíamos definido previamente.

Ya hemos visto previamente como borrar un registro

```
a=df.drop([1])
```

4.- Escritura en ficheros Excell

Para escribir un objeto DataFrame en una hoja Excell utilizamos la función to_excel(nombre fichero, nombre hoja) que está incluida en la librería openpyxl que ya instalamos al principio.

```
import pandas
y=pandas.read_excel ('c:/ASIR1.xlsx','ASIR')
df=pandas.DataFrame(y)
b=df['Profesor'].unique()
a=pandas.DataFrame(b)
a.to_excel('c:/nuevo1.xlsx','hoja2')
```

En este ejemplo, hemos leído el fichero 'c:/ASIR1.xlsx' con la hoja ASIR. Hemos creado un array llamado b con una columna solamente en la que aparecen los profesores no repetidos. Este array lo hemos convertido en otro objeto DataFrame llamado 'a' y, por último, lo hemos pasado a un fichero excell.

	A	B
1		**0**
2	**0**	Alejandro de la Torre
3	**1**	José Rodriguez
4	**2**	Francisco Soria
5	**3**	Antonio Torres

A su vez, para poder funcionar, nuestra librería “pandas” va a necesitar hacer uso de otro módulo de nombre “**xlrd**“. Por lo que tendremos que instalarlo también:

Una vez que tenemos importadas las librerías necesarias, podemos empezar a trabajar con archivos excel. Nuestra primera practica va a consistir en la lectura y visualización del contenido de un archivo excel (de extensión “**.xlsx**“) al que hemos dado el nombre de “**telefonos**” y que contiene la siguiente información:

Como se puede ver hemos creado un archivo en el que tenemos dos columnas: “**NOMBRES**” en la que tenemos almacenados los nombres de nuestros amigos, y la columna “**TELEFONOS**” en la que aparecen sus respectivos números telefónicos.

Para imprimir esta información con python, lo primero que haremos será importar la librería “**pandas**” con el nombre abreviado de “**pd**“:

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
>>> #LECTURA DE ARCHIVOS "EXCEL" CON "pandas".
>>>
>>> #IMPORTAMOS "pandas".
>>> import pandas as pd
>>>
```

Tras ello, mediante una variable, a la que hemos bautizado como “**archivo_excel**“, definimos el archivo (de extensión “**.xlsx**” en este caso) al que queremos acceder (en nuestro caso **“telefonos.xlsx**“) el cual, a su vez, tendrá que estar ubicado dentro de nuestra carpeta python:

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
>>> #LECTURA DE ARCHIVOS "EXCEL" CON "pandas".
>>>
>>> #IMPORTAMOS "pandas".
>>> import pandas as pd
>>>
>>> #ESPECIFICAMOS EL ARCHIVO EXCEL A REPRODUCIR.
>>> archivo_excel="telefonos.xlsx"
>>>
```

Una vez que tenemos identificado el nombre de nuestro archivo “**.xlsx**“, el siguiente paso será el proceder a su lectura, introduciendo el siguiente código:

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
>>> #LECTURA DE ARCHIVOS "EXCEL" CON "pandas".
>>>
>>> #IMPORTAMOS "pandas".
>>> import pandas as pd
>>>
>>> #ESPECIFICAMOS EL ARCHIVO EXCEL A REPRODUCIR.
>>> archivo_excel="telefonos.xlsx"
>>>
>>> #LEEMOS EL ARCHIVO "archivo_excel".
>>> datos=pd.read_excel(archivo_excel)
>>>
```

Así, hemos creado una variable (“datos”) que será igual al contenido leído mediante la función “**pd.read_excel()**” que tomará como argumento el nombre del archivo que queremos leer (que como recordaremos, se encuentra almacenado en la variable “**archivo_excel**“).

Si, hecho esto, ejecutamos y usamos **“print**” para ver la información del archivo **excel**, obtenemos el siguiente output:

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Inte
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
>>> #LECTURA DE ARCHIVOS "EXCEL" CON "pandas".
>>>
>>> #IMPORTAMOS "pandas".
>>> import pandas as pd
>>>
>>> #ESPECIFICAMOS EL ARCHIVO EXCEL A REPRODUCIR.
>>> archivo_excel="telefonos.xlsx"
>>>
>>> #LEEMOS EL ARCHIVO "archivo_excel".
>>> datos=pd.read_excel(archivo_excel)
>>>
>>> #IMPRIMIMOS INFORMACIÓN.
>>> print(datos)
     NOMBRE   TELEFONOS
0    Antonio      235678
1      Borja      345678
2      Carla      982366
3     Felipe      887334
4       Jose      232424
5      Lucia      443737
>>>
```

Como se puede ver, hemos procedido a la lectura de la tabla de nombres y números telefónicos que tenemos en nuestro archivo excel “**telefonos.xlsx**“.

No obstante, en ocasiones, es posible que eventualmente, deseemos que nuestro programa realice cambios en los datos leídos por el procedimiento anterior. Así, en nuestro ejemplo, imaginemos que queremos cambiar, en la columna de nombres, el nombre de “Jose”, por el de “Miguel”:

Para este caso, empezaremos importando “pandas” y definiendo el archivo al que queremos acceder, tal y como hacíamos en el ejemplo anterior.

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
>>> #IMPORTAMOS "pandas".
>>> import pandas as pd
>>>
>>> #DEFINIMOS EL ARCHIVO EXCEL A LEER.
>>> archivo_excel="telefonos.xlsx"
>>>
```