Pràctica 5: el paquet `pandas`

12 de desembre de 2020

1 El paquet `pandas`.

La biblioteca de Python **pandas** (**pan**el **da**ta) aporta funcions per explorar, netejar i transformar dades. Les seves principals estructures són les `Series` i els `DataFrames`:

- Les `Series` són estructures 1-dimensionals que contenen dades de qualsevol tipus: enters, números en punt flotant, cadenes, ...).
- Els `DataFrames` són estructures 2-dimensionals on les columnes estan etiquetades. Es pot pensar com una família de `Series` organitzades per columnes.

1.1 Exemple amb `Series`:

Considerem el nombre d'estudiants de les universitats públiques catalanes al curs 2018/19 (font: Idescat).

[1]:
```
import pandas as pd
univ=pd.Series([44442, 30926, 22397, 15801, 13597,  8755, 11946],
               index=['UB','UAB','UPC','UPF','UdG','UdL','URV'])
```

[2]:
```
print(univ)
```

```
UB     44442
UAB    30926
UPC    22397
UPF    15801
UdG    13597
UdL     8755
URV    11946
dtype: int64
```

[3]:
```
print(type(univ))
```

```
<class 'pandas.core.series.Series'>
```

[4]:
```
print(univ.index)
```

```
Index(['UB', 'UAB', 'UPC', 'UPF', 'UdG', 'UdL', 'URV'], dtype='object')
```

[5]:
```
print(type(univ.index))
```

```
<class 'pandas.core.indexes.base.Index'>
```

[6]:
```
print(univ.shape)
```

```
(7,)
```

[7]:
```
print(univ.shape[0])
```

```
7
```

[8]:
```
print(univ['UAB']) # Retorna un número
```

```
30926
```

[9]:
```
print(univ[['UB','UAB','UPC','UPF']]) # Retorna una Serie
```

```
UB     44442
UAB    30926
UPC    22397
UPF    15801
dtype: int64
```

Podem aprofitar aquesta sintaxis per a definir una nova sèrie a partir d'aquesta: considerem les universitats de la província de Barcelona

[10]:
```
univB=univ[['UB','UAB','UPC','UPF']]
print(univB)
```

```
UB     44442
UAB    30926
UPC    22397
UPF    15801
dtype: int64
```

`univB` és una còpia independent de `univ`. Ho podem comprovar modificant un dels valors a `univB` i comprovant que no es modifica a `univ`:

[11]:
```
univB['UAB']=31000
```

[12]:
```
print(univ)
```

```
UB     44442
UAB    30926
UPC    22397
UPF    15801
UdG    13597
UdL     8755
```

```
URV    11946
dtype: int64
```

[13]:
```
print(univB)
```

```
UB     44442
UAB    31000
UPC    22397
UPF    15801
dtype: int64
```

També accepta la sintaxis dels `array` per accedir a diferents posiscions:

[14]:
```
print(univ[1])
```

```
30926
```

[15]:
```
print(univ[1:2])
```

```
UAB    30926
dtype: int64
```

1.2 Exemple amb `DataFrame`:

Definim ara les dades sobre els estudiants de les universitats públiques separades per gènere. Primer ho entrem en format diccionari i llavors ho convertim a `DataFrame`.

[16]:
```
import pandas as pd
univ=pd.DataFrame({'Dones':[27523, 18529,  58091,  8900,  7682,  5077,  7022],
                   'Homes':[16919, 12397, 16506,  6901,  5775,  3678,  4923]},
                   index=['UB','UAB','UPC','UPF','UdG','Udl','URV'])
```

[17]:
```
print(univ)
```

```
     Dones  Homes
UB   27523  16919
UAB  18529  12397
UPC  58091  16506
UPF   8900   6901
UdG   7682   5775
Udl   5077   3678
URV   7022   4923
```

Podem afegir una columna amb la suma:

[18]:
```
univ['Total']=univ['Dones']+univ['Homes']
```

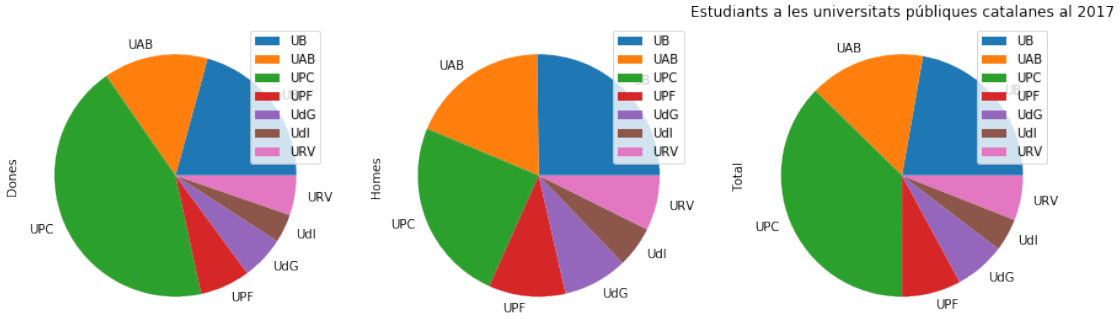
[19]:
```
print(univ)
```

```
      Dones  Homes  Total
UB    27523  16919  44442
UAB   18529  12397  30926
UPC   58091  16506  74597
UPF    8900   6901  15801
UdG    7682   5775  13457
Udl    5077   3678   8755
URV    7022   4923  11945
```

I fins i tot podem fer un gràfic per cada columna:

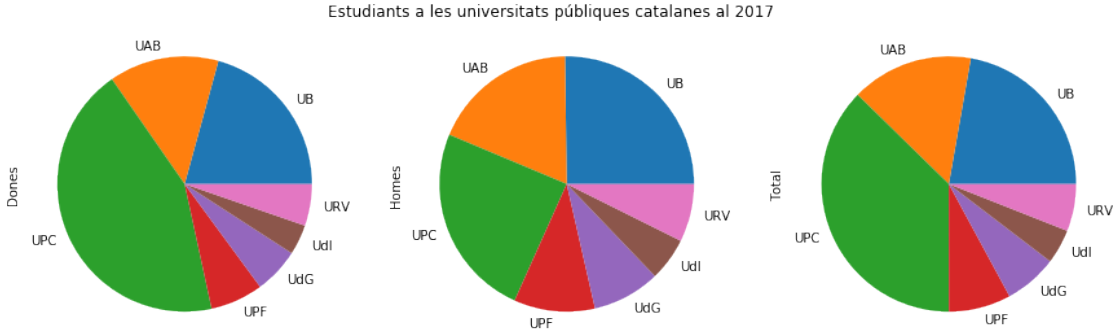
[20]:
```
import matplotlib.pyplot as plt
```

[21]:
```
univ.plot.pie(subplots=True,figsize=(15,5))
plt.title("Estudiants a les universitats públiques catalanes al 2017")
plt.show()
```



Treiem la llegenda (en aquest cas no fa falta) i centrem el títol:

[22]:
```
pie=univ.plot.pie(subplots=True,figsize=(15,5),legend=False)
plt.title("Estudiants a les universitats públiques catalanes al 2017",x=-.75)
plt.show()
```



1.3 Exemple

A aquest exemple utilitzarem dades de la web **Open Data BCN**, que podeu trobar a l'adreça http://opendata-ajuntament.barcelona.cat.
Com a exemple descarregarem els *Naixements de la població per sexe per barris de la ciutat de Barcelona* corresponents a l'any *2019*.
http://opendata-ajuntament.barcelona.cat/data/ca/dataset/est-demo-naixements-sexe
La descàrrega es pot fer amb el navegador d'internet que estigueu utilitzant i el primer que hem de fer és mirar en quin format estan les dades.
Mirant les primeres línies:

```
"Any","Codi_Districte","Nom_Districte","Codi_Barri","Nom_Barri","Sexe","Nombre"
2019,1,"Ciutat Vella",1,"el Raval","Nens",210
2019,1,"Ciutat Vella",2,"el Barri Gòtic","Nens",64
```

Veiem que les columnes estan separades per comes i quan hi ha un text està entre cometes. Guardem a la variable `url` l'adreça d'internet, que s'ha d'escriure en una sola línia:

[23]:
```
url='https://opendata-ajuntament.barcelona.cat/data/dataset/
 ↪237dc137-c306-40e8-bfb2-62d073df3d7d/resource/
 ↪83a0195f-ce6f-4bcc-9ffb-b863ab5de468/download/2019_naixements_sexe.csv'
```

[24]:
```
naixements=pd.read_csv(url,sep=',')
```

[25]:
```
print(naixements)
```

```
      Any  Codi_Districte Nom_Districte  Codi_Barri  \
0    2019               1  Ciutat Vella           1
1    2019               1  Ciutat Vella           2
2    2019               1  Ciutat Vella           3
3    2019               1  Ciutat Vella           4
4    2019               2      Eixample           5
..    ...             ...           ...         ...
141  2019              10    Sant Martí          69
142  2019              10    Sant Martí          70
143  2019              10    Sant Martí          71
144  2019              10    Sant Martí          72
145  2019              10    Sant Martí          73

                                      Nom_Barri   Sexe  Nombre
0                                      el Raval  Nenes     210
1                                  el Barri Gòtic  Nenes      64
2                                  la Barceloneta  Nenes      36
3          Sant Pere, Santa Caterina i la Ribera  Nenes      60
4                                   el Fort Pienc  Nenes     101
..                                            ...    ...     ...
141  Diagonal Mar i el Front Marítim del Poblenou   Nens      79
142                         el Besòs i el Maresme   Nens     111
143                     Provençals del Poblenou   Nens      98
```

```
144                      Sant Martí de Provençals   Nens      95
145                           la Verneda i la Pau   Nens      95

[146 rows x 7 columns]
```

Volem fer una taula nova amb les dades que ens interessen: * Com que l'any és constant, no cal que hi sigui. * Dels districtes i barris ja està bé tenir el nom i el codi. Tot i això veiem que en aquesta taula el que fa d'índex és el `Codi_Barri` (no hi ha el mateix codi a dues línies diferents, tret per dir el nombre de nens i nenes). * Enlloc de les columnes `Sexe` i `Nombre` volem que hi hagi `Nens` i `Nenes` amb el nombre que correspongui a cada columna.

[26]:
```
naix=pd.DataFrame.copy(naixements)
del naix['Any']
```

Mirem quins valors diferents pren la variable `Sexe` (en realitat, tant sols ens interessa quants valors diferents hi ha):

[27]:
```
sexes=naix['Sexe'].unique()
print(sexes)
print(len(sexes))
```

```
['Nenes' 'Nens']
2
```

El codi següent crea primer, per cada valor de la variable `Sexe` una taula, llavors esborra la columna que ja no aporta res, modifica el nom de la columna `Nombre` pel que correspon i barreja les taules resultants en una nova taula, que és la que volem.

[28]:
```
naix0=naix[naix['Sexe']==sexes[0]]
naix1=naix[naix['Sexe']==sexes[1]]
del naix0['Sexe']
del naix1['Sexe']
naix0=naix0.rename(columns={'Nombre':sexes[0]})
naix1=naix1.rename(columns={'Nombre':sexes[1]})
# Esborrem les columnes d'una taula que ja són a l'altra, excepte l'índex.
naix1=naix1[['Codi_Barri',sexes[1]]]
naixDEF=pd.merge(naix0,naix1,on='Codi_Barri')
print(naixDEF)
```

```
    Codi_Districte Nom_Districte  Codi_Barri  \
0                1  Ciutat Vella           1
1                1  Ciutat Vella           2
2                1  Ciutat Vella           3
3                1  Ciutat Vella           4
4                2      Eixample           5
..             ...           ...         ...
68              10    Sant Martí          69
69              10    Sant Martí          70
70              10    Sant Martí          71
```

```
71              10    Sant Martí          72
72              10    Sant Martí          73
```

	Nom_Barri	Nenes	Nens
0	el Raval	210	222
1	el Barri Gòtic	64	60
2	la Barceloneta	36	48
3	Sant Pere, Santa Caterina i la Ribera	60	83
4	el Fort Pienc	101	104
..	...	...	...
68	Diagonal Mar i el Front Marítim del Poblenou	85	79
69	el Besòs i el Maresme	127	111
70	Provençals del Poblenou	92	98
71	Sant Martí de Provençals	89	95
72	la Verneda i la Pau	82	95

```
[73 rows x 6 columns]
```

Finalment, escrivim el resultat a un fitxer per a poder-lo utilitzar quan vulguem:

[29]:
```
naixDEF.to_csv('Naixements2019.csv',index=False,quotechar='"')
```

1.3.1 Exercici

Podeu veure que a l'exemple anterior hem hagut de repetir el mateix per a cada valor de la variable `Sexe`. Afortunadament, en aquest cas, tant sols hi havia dos valors diferents.
L'exercici consisteix en modificar el codi per a que es faci el mateix i serveixi per a qualsevol nombre de casos diferents.
Una dificultat apareixerà del fet de necessitar un nombre no fixat de variables que hauran de tenir noms diferents. Això ho podeu fer amb la instrucció `exec`, de la que hi ha un exemple a continuació:

[30]:
```
a='hola' # a guarda el nom de la nova variable
# a aquesta nova variable volem escriure-hi la cadena 'valor nou'
exec(a + "= 'valor nou'")
print(hola)
```

```
valor nou
```

1.3.2 Exercici

Aprofiteu la millora que heu fet al codi anterior per a estudiar les dades del 2019 a:
http://opendata-ajuntament.barcelona.cat/data/ca/dataset/est-demo-altes-omissio-edat-quinquenal
Fent l'anàleg que hem fer per cada `Sexe`, però ara per `Edats quinquenals` (haurà de sortir una columna per `0-4 anys`, una altra per `5-9 anys`, ...