😎 Indexación y Slicing en Pandas

Acceso a datos de series con posición e índices

- Si se inserta un **':'** (dos puntos) delante de él, se extraerán todos los elementos de ese índice en adelante.
- Si se utilizan dos parámetros (con **':'** entre ellos), se recuperan los elementos entre los dos índices (sin incluir el índice de detención)

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

s = pd.Series([1,2,3,4,5,6,7,8],index = ['a','b','c','d','e','f','g','h'])
s
```

Out[1]:
```
a    1
b    2
c    3
d    4
e    5
f    6
g    7
h    8
dtype: int64
```

In [2]:
```
print(f"Recupera el valor de la posición 2:\n {s[2]}")
print(f"Recupera los primeros tres elementos de la serie:\n{s[:3]}")
print(f"Recupera los últimos tres elementos de la serie:\n{s[-3:]}")
print(f"Recupera desde la posicion 1 hasta la 3. También llamado Corte por \
      'índice entero implícito':\n{s[1:4]}")
print(f"Recupera un solo elemento utilizando el valor de la etiqueta de índice:\n{s['a']}")
print(f"Recupera múltiples elementos usando una lista de valores de \
      etiquetas de índice. También llamada \
'Indexación elegante':\n{s[['a','c','f']]}")
print(f"Recupera los valores entre los índices indicados. También llamado \
'Corte por índice explícito':\n{s['b':'f']}")
```

```
Recupera el valor de la posición 2:
 3
Recupera los primeros tres elementos de la serie:
a    1
b    2
c    3
dtype: int64
Recupera los últimos tres elementos de la serie:
f    6
g    7
h    8
dtype: int64
Recupera desde la posicion 1 hasta la 3. También llamado Corte por       'índice entero implícito':
b    2
c    3
d    4
dtype: int64
Recupera un solo elemento utilizando el valor de la etiqueta de índice:
1
Recupera múltiples elementos usando una lista de valores de       etiquetas de índice. También llamada 'Indexación elegante':
a    1
c    3
f    6
dtype: int64
Recupera los valores entre los índices indicados. También llamado 'Corte por índice explícito':
b    2
c    3
d    4
e    5
f    6
dtype: int64
```

- **Cuando se corta con un índice explícito, s['b':'d'], el índice final se incluye en el segmento.**
- **Cuando se corta con índice implícito, s[1:4], el valor final del índice se excluye del segmento.**

In [3]:
```
print(f"Recupera los valores que cumplen con la condición. También llamado \
'Enmascaramiento':\n{s[(s > 3) & (s < 7)]}")
```

```
Recupera los valores que cumplen con la condición. También llamado 'Enmascaramiento':
d    4
e    5
f    6
dtype: int64
```

Indexadores: loc e iloc

Atención!: Estas convenciones de segmentación e indexación pueden ser confusas. Por ejemplo:

- si la serie tiene un índice entero explícito, una operación de indexación como s[1] usará los índices **explícitos**,
- mientras que una operación de corte como s[1:3] usará el índice **implícito**.

In [4]:
```
s = pd.Series(['a', 'b', 'c'], index=[1, 3, 5])
s
```

Out[4]:
```
1    a
3    b
5    c
dtype: object
```

In [5]:
```
print(f"Índice explícito al indexar:\n{s[1]}")
print(f"Índice implícito al cortar:\n{s[1:3]}")
```

```
Índice explícito al indexar:
a
Índice implícito al cortar:
3    b
5    c
dtype: object
```

```
Debido a esta posible confusión en el caso de los índices de enteros, Pandas
proporciona atributos que exponen una interfaz de corte particular a los datos en la Serie:
```

- El atributo **loc** que permite la indexación y el corte que siempre hace referencia al índice **explícito**:

In [6]:
```
s.loc[1]
```

Out[6]:
```
'a'
```

In [7]:
```
s.loc[1:3]
```

Out[7]:
```
1    a
3    b
dtype: object
```

- El atributo **iloc** permite la indexación y el corte que siempre hace referencia al índice **implícito**:

In [8]:
```
s.iloc[1]
```

Out[8]:
```
'b'
```

In [9]:
```
s.iloc[1:3]
```

Out[9]:
```
3    b
5    c
dtype: object
```

Acceso a datos de dataframe con posición e índices

La indexación se refiere a las columnas, el corte (slicing) se refiere a las filas:

In [10]:
```
poblacion_dict = {'Buenos Aires':8332521,'Córdoba':7448193,'Mendoza':1965112,
                  'Neuquén':1955607,'Santa Fé':1281353}
s1 = pd.Series(poblacion_dict)
area_dict = {'Buenos Aires':423967, 'Córdoba':695662, 'Mendoza':141297,
                   'Neuquén':170312, 'Santa Fé':149995}
s2 = pd.Series(area_dict)
provincias = pd.DataFrame({'población': s1,'área': s2})
provincias
```

Out[10]:

	población	área
Buenos Aires	8332521	423967
Córdoba	7448193	695662
Mendoza	1965112	141297
Neuquén	1955607	170312
Santa Fé	1281353	149995

Pasar un solo "índice" a un DataFrame accede a una columna:

In [11]:
```
provincias['área']
```

Out[11]:
```
Buenos Aires    423967
Córdoba         695662
Mendoza         141297
Neuquén         170312
Santa Fé        149995
Name: área, dtype: int64
```

Los segmentos también pueden referirse a filas por número en lugar de por índice:

In [12]:
```
provincias['Córdoba':'Neuquén']
```

Out[12]:

	población	área
Córdoba	7448193	695662
Mendoza	1965112	141297
Neuquén	1955607	170312

In [13]:
```
provincias[1:4]
```

Out[13]:

	población	área
Córdoba	7448193	695662
Mendoza	1965112	141297
Neuquén	1955607	170312

loc: *selección de fila pasando la etiqueta*

Pasar un solo índice accede a una fila:

In [14]:
```
provincias.loc['Mendoza']
```

Out[14]:
```
población    1965112
área          141297
Name: Mendoza, dtype: int64
```

El resultado es una serie con etiquetas con el nombres de columna del DataFrame. Y el nombre de la serie es la etiqueta con la que se recupera:

In [15]:
```
type(provincias.loc['Mendoza'])
```

Out[15]:
```
pandas.core.series.Series
```

Recuperamos los datos con el atributo de nombres de columna que son cadenas:

In [16]:
```
provincias.área
```

Out[16]:
```
Buenos Aires    423967
Córdoba         695662
Mendoza         141297
Neuquén         170312
Santa Fé        149995
Name: área, dtype: int64
```

Con el indexador loc podemos combinar el enmascaramiento y la indexación elegante:

In [17]:
```
provincias.loc[:'Mendoza', :'población']
```

Out[17]:

	población
Buenos Aires	8332521
Córdoba	7448193
Mendoza	1965112

iloc: *selección de fila pasando la ubicación entera:*

In [18]:
```
d = {'uno' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
     'dos' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd']),
    'tres' : pd.Series([4, 5, 6, 7], index=['a', 'b', 'c', 'd']),
    'cuatro' : pd.Series([5, 8, 7], index=['a', 'c', 'd'])}
df = pd.DataFrame(d)
df
```

Out[18]:

	uno	dos	tres	cuatro
a	1.0	1	4	5.0
b	2.0	2	5	NaN
c	3.0	3	6	8.0
d	NaN	4	7	7.0

In [19]:
```
df.iloc[2]
```

Out[19]:
```
uno       3.0
dos       3.0
tres      6.0
cuatro    8.0
Name: c, dtype: float64
```

*Se pueden seleccionar varias filas con el operador **':'***

In [20]:
```
df.iloc[2:4]
```

Out[20]:

	uno	dos	tres	cuatro
c	3.0	3	6	8.0
d	NaN	4	7	7.0

In [21]:
```
provincias.iloc[:3, :2]
```

Out[21]:

	población	área
Buenos Aires	8332521	423967
Córdoba	7448193	695662
Mendoza	1965112	141297

Formato fila columna. Se pueden seleccionar modificar valores:

In [22]:
```
df.iloc[1,2] = 90
df
```

Out[22]:

	uno	dos	tres	cuatro
a	1.0	1	4	5.0
b	2.0	2	90	NaN
c	3.0	3	6	8.0
d	NaN	4	7	7.0

Selección de datos de Pandas

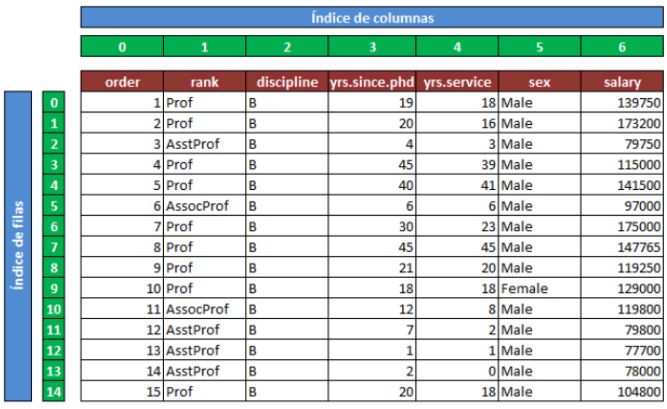
- Seleccionar datos por posición (.iloc)
- Seleccionar datos por etiqueta o por una declaración condicional (.loc)

Para verificar la estructura que devuelve la selección (Series o Dataframe) asignar a una variable la selección y posteriormente aplicar type().

In [23]:
```
import pandas as pd
datos = pd.read_csv('archs/salarios.csv')
datos.head()
```

Out[23]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	20	16	Male	173200
2	3	AsstProf	B	4	3	Male	79750
3	4	Prof	B	45	39	Male	115000
4	5	Prof	B	40	41	Male	141500



Seleccionar una fila

In [24]:
```
datos.head(6)
```

Out[24]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	20	16	Male	173200
2	3	AsstProf	B	4	3	Male	79750
3	4	Prof	B	45	39	Male	115000
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000

Slicing en Pandas con iloc

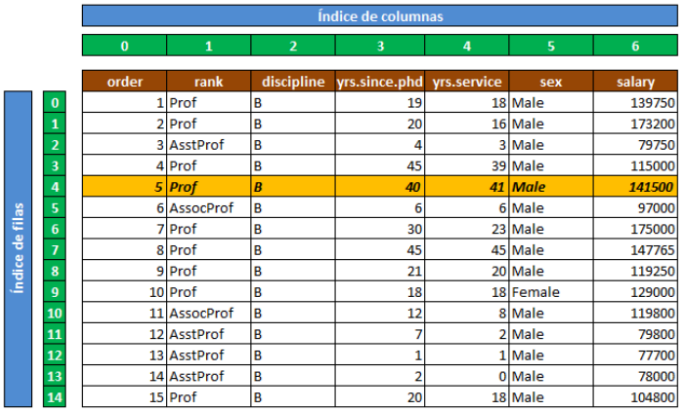
```
In [25]: 1 datos.iloc[4]
```

```
Out[25]: order              5
         rank            Prof
         discipline         B
         yrs.since.phd     40
         yrs.service       41
         sex             Male
         salary          141500
         Name: 4, dtype: object
```

```
In [26]: 1 type(datos.iloc[4])
```

```
Out[26]: pandas.core.series.Series
```

- En esta selección pandas devuelve una Serie




```
In [27]: 1 datos.iloc[-1]
```

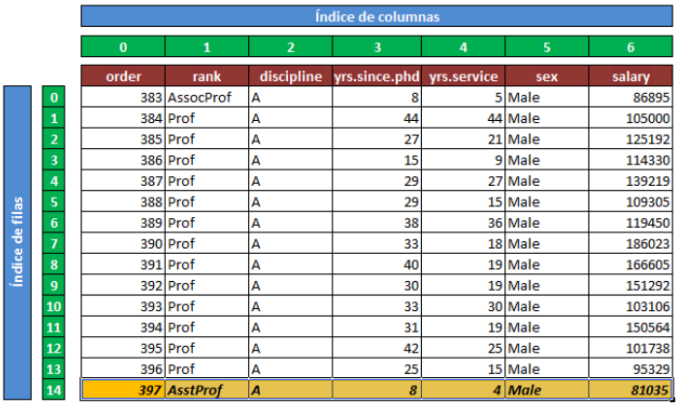
```
Out[27]: order              397
         rank           AsstProf
         discipline            A
         yrs.since.phd         8
         yrs.service           4
         sex                Male
         salary            81035
         Name: 396, dtype: object
```

Podemos ingresar una lista, con un solo índice entero, cuando usamos iloc. Esto indexará una fila, pero la salida será diferente en comparación con el ejemplo anterior:

```
In [28]: 1 datos.iloc[[-1]]
```

```
Out[28]:
```

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
396	397	AsstProf	A	8	4	Male	81035



Seleccionar una celda específica

```
In [29]: 1 datos.iloc[9,5]
```

```
Out[29]: 'Female'
```

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	20	16	Male	173200
2	3	AsstProf	B	4	3	Male	79750
3	4	Prof	B	45	39	Male	115000
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

Seleccionar múltiples filas

In [30]:
```
1 datos.iloc[[7, 2, 0]]
```

Out[30]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
7	8	Prof	B	45	45	Male	147765
2	3	AsstProf	B	4	3	Male	79750
0	1	Prof	B	19	18	Male	139750

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	*1*	*Prof*	*B*	*19*	*18*	*Male*	*139750*
1	2	Prof	B	20	16	Male	173200
2	*3*	*AsstProf*	*B*	*4*	*3*	*Male*	*79750*
3	4	Prof	B	45	39	Male	115000
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000
6	7	Prof	B	30	23	Male	175000
7	*8*	*Prof*	*B*	*45*	*45*	*Male*	*147765*
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

- En esta selección pandas devuelve un Dataframe

Seleccionar parte de los datos de una fila

In [31]:
```
1 datos.iloc[3, [1, 2, 3]]
```

Out[31]:
```
rank             Prof
discipline          B
yrs.since.phd      45
Name: 3, dtype: object
```

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	20	16	Male	173200
2	3	AsstProf	B	4	3	Male	79750
3	*4*	*Prof*	*B*	*45*	39	Male	115000
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

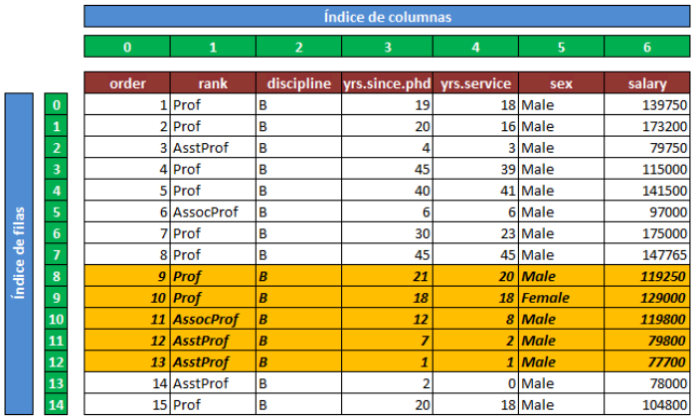
- En esta selección pandas devuelve una Serie

Seleccionar rango de filas y todas las columnas

In [32]:
```
1 datos.iloc[8:13]
```

Out[32]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700

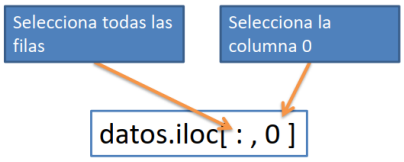


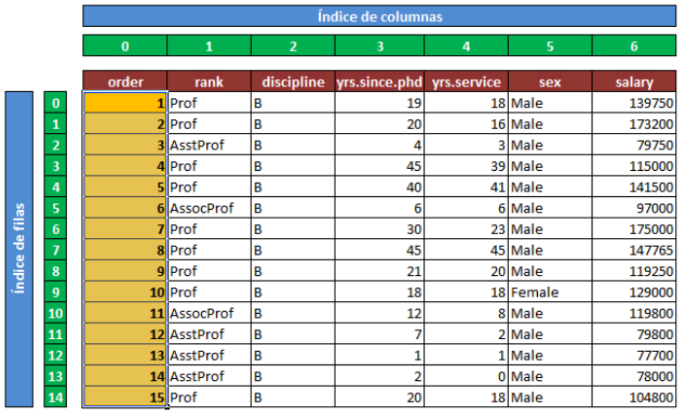
- Al seleccionar varias columnas o varias filas , las filas / columnas seleccionadas se ejecutarán desde el primer número hasta uno menos del segundo valor, por ejemplo, [1: 5] será 1, 2, 3, 4.

Seleccionar columnas

In [33]:
```
1 datos.iloc[:, 0]
```

Out[33]:
```
0        1
1        2
2        3
3        4
4        5
      ...
392    393
393    394
394    395
395    396
396    397
Name: order, Length: 397, dtype: int64
```





In [34]:
```
1 datos.iloc[:, -1]
```

Out[34]:
```
0      139750
1      173200
2       79750
3      115000
4      141500
        ...
392    103106
393    150564
394    101738
395     95329
396     81035
Name: salary, Length: 397, dtype: int64
```

Índice de filas	Índice de columnas						
	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	*139750*
1	2	Prof	B	20	16	Male	*173200*
2	3	AsstProf	B	4	3	Male	*79750*
3	4	Prof	B	45	39	Male	*115000*
4	5	Prof	B	40	41	Male	*141500*
5	6	AssocProf	B	6	6	Male	*97000*
6	7	Prof	B	30	23	Male	*175000*
7	8	Prof	B	45	45	Male	*147765*
8	9	Prof	B	21	20	Male	*119250*
9	10	Prof	B	18	18	Female	*129000*
10	11	AssocProf	B	12	8	Male	*119800*
11	12	AsstProf	B	7	2	Male	*79800*
12	13	AsstProf	B	1	1	Male	*77700*
13	14	AsstProf	B	2	0	Male	*78000*
14	15	Prof	B	20	18	Male	*104800*

Seleccionar parte de filas y una columna

In [35]:
```
1 datos.iloc[1:5, 3]
```

Out[35]:
```
1    20
2     4
3    45
4    40
Name: yrs.since.phd, dtype: int64
```

Índice de filas	Índice de columnas						
	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	*20*	16	Male	173200
2	3	AsstProf	B	*4*	3	Male	79750
3	4	Prof	B	*45*	39	Male	115000
4	5	Prof	B	*40*	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

- En esta selección pandas devuelve una Serie

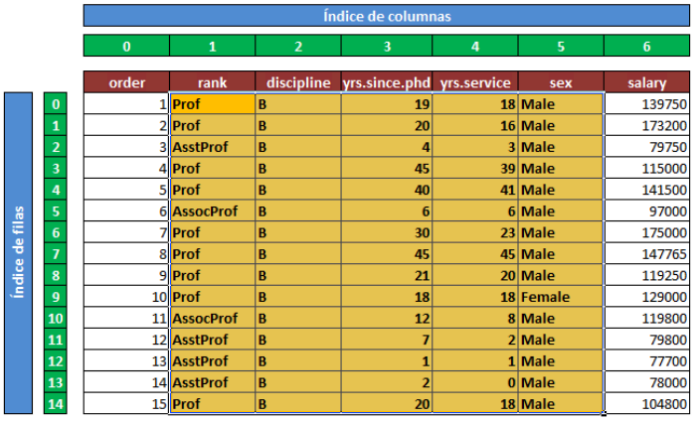
Seleccionar todas las filas y un rango de columnas

In [36]:
```
1 datos.iloc[ : , 1: 6]
```

Out[36]:

	rank	discipline	yrs.since.phd	yrs.service	sex
0	Prof	B	19	18	Male
1	Prof	B	20	16	Male
2	AsstProf	B	4	3	Male
3	Prof	B	45	39	Male
4	Prof	B	40	41	Male
...	...	...	...	...	...
392	Prof	A	33	30	Male
393	Prof	A	31	19	Male
394	Prof	A	42	25	Male
395	Prof	A	25	15	Male
396	AsstProf	A	8	4	Male

397 rows × 5 columns



La tabla sigue...

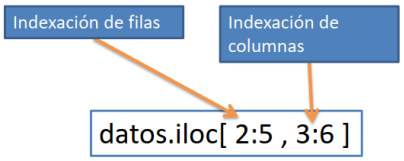
- En esta selección pandas devuelve un Dataframe

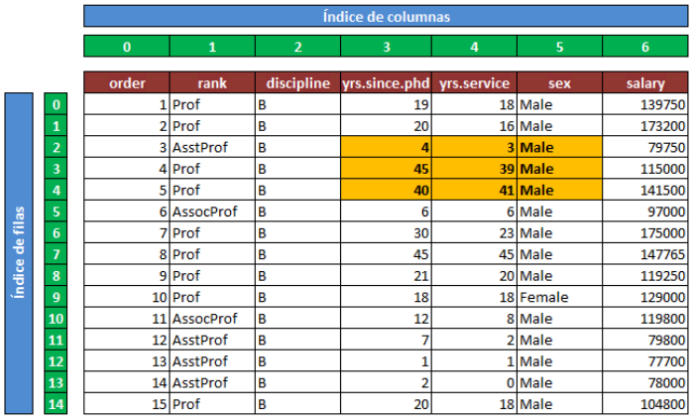
Seleccionar subconjuntos de celdas

In [37]:
```
1 datos.iloc[2:5, 3:6]
```

Out[37]:

	yrs.since.phd	yrs.service	sex
2	4	3	Male
3	45	39	Male
4	40	41	Male





Slicing en Pandas con loc

In [38]:
```
1 import pandas as pd
2 datos = pd.read_csv('archs/salarios.csv')
3 datos.loc[3]
```

Out[38]:
```
order              4
rank            Prof
discipline         B
yrs.since.phd     45
yrs.service       39
sex             Male
salary        115000
Name: 3, dtype: object
```

In [39]:
```
1 datos.loc[[3]]
```

Out[39]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
3	4	Prof	B	45	39	Male	115000

In [40]:
```
1 type(datos.loc[[3]])
```

Out[40]:
```
pandas.core.frame.DataFrame
```

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	20	16	Male	173200
2	3	AsstProf	B	4	3	Male	79750
3	***4***	***Prof***	***B***	***45***	***39***	***Male***	***115000***
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

Seleccionar subconjunto

In [41]:
```
1 datos.loc[1:5]
```

Out[41]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
1	2	Prof	B	20	16	Male	173200
2	3	AsstProf	B	4	3	Male	79750
3	4	Prof	B	45	39	Male	115000
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	***2***	***Prof***	***B***	***20***	***16***	***Male***	***173200***
2	***3***	***AsstProf***	***B***	***4***	***3***	***Male***	***79750***
3	***4***	***Prof***	***B***	***45***	***39***	***Male***	***115000***
4	***5***	***Prof***	***B***	***40***	***41***	***Male***	***141500***
5	***6***	***AssocProf***	***B***	***6***	***6***	***Male***	***97000***
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

Seleccionar filas alternadas

In [42]:
```
1 datos.loc[[1, 3, 7, 10, 13]]
```

Out[42]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
1	2	Prof	B	20	16	Male	173200
3	4	Prof	B	45	39	Male	115000
7	8	Prof	B	45	45	Male	147765
10	11	AssocProf	B	12	8	Male	119800
13	14	AsstProf	B	2	0	Male	78000

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	***2***	***Prof***	***B***	***20***	***16***	***Male***	***173200***
2	3	AsstProf	B	4	3	Male	79750
3	***4***	***Prof***	***B***	***45***	***39***	***Male***	***115000***
4	5	Prof	B	40	41	Male	141500
5	6	AssocProf	B	6	6	Male	97000
6	7	Prof	B	30	23	Male	175000
7	***8***	***Prof***	***B***	***45***	***45***	***Male***	***147765***
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	***11***	***AssocProf***	***B***	***12***	***8***	***Male***	***119800***
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	***14***	***AsstProf***	***B***	***2***	***0***	***Male***	***78000***
14	15	Prof	B	20	18	Male	104800

Seleccionar con nombres de columnas

In [43]:
```
1 datos.loc[:, 'rank']
```

Out[43]:
```
0          Prof
1          Prof
2      AsstProf
3          Prof
4          Prof
          ...
392        Prof
393        Prof
394        Prof
395        Prof
396    AsstProf
Name: rank, Length: 397, dtype: object
```

Índice de columnas

Índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	***Prof***	B	19	18	Male	139750
1	2	***Prof***	B	20	16	Male	173200
2	3	***AsstProf***	B	4	3	Male	79750
3	4	***Prof***	B	45	39	Male	115000
4	5	***Prof***	B	40	41	Male	141500
5	6	***AssocProf***	B	6	6	Male	97000
6	7	***Prof***	B	30	23	Male	175000
7	8	***Prof***	B	45	45	Male	147765
8	9	***Prof***	B	21	20	Male	119250
9	10	***Prof***	B	18	18	Female	129000
10	11	***AssocProf***	B	12	8	Male	119800
11	12	***AsstProf***	B	7	2	Male	79800
12	13	***AsstProf***	B	1	1	Male	77700
13	14	***AsstProf***	B	2	0	Male	78000
14	15	***Prof***	B	20	18	Male	104800

La tabla sigue...

In [44]:
```
1 datos.loc[1:5, ['rank', 'yrs.service']]
```

Out[44]:

	rank	yrs.service
1	Prof	16
2	AsstProf	3
3	Prof	39
4	Prof	41
5	AssocProf	6

Índice de columnas

índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	*Prof*	B	20	*16*	Male	173200
2	3	*AsstProf*	B	4	*3*	Male	79750
3	4	*Prof*	B	45	*39*	Male	115000
4	5	*Prof*	B	40	*41*	Male	141500
5	6	*AssocProf*	B	6	*6*	Male	97000
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

In [45]:
```
1 datos.loc[1:5, 'rank':'yrs.service']
```

Out[45]:

	rank	discipline	yrs.since.phd	yrs.service
1	Prof	B	20	16
2	AsstProf	B	4	3
3	Prof	B	45	39
4	Prof	B	40	41
5	AssocProf	B	6	6

Índice de columnas

índice de filas	0	1	2	3	4	5	6
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	*Prof*	*B*	*20*	*16*	Male	173200
2	3	*AsstProf*	*B*	*4*	*3*	Male	79750
3	4	*Prof*	*B*	*45*	*39*	Male	115000
4	5	*Prof*	*B*	*40*	*41*	Male	141500
5	6	*AssocProf*	*B*	*6*	*6*	Male	97000
6	7	Prof	B	30	23	Male	175000
7	8	Prof	B	45	45	Male	147765
8	9	Prof	B	21	20	Male	119250
9	10	Prof	B	18	18	Female	129000
10	11	AssocProf	B	12	8	Male	119800
11	12	AsstProf	B	7	2	Male	79800
12	13	AsstProf	B	1	1	Male	77700
13	14	AsstProf	B	2	0	Male	78000
14	15	Prof	B	20	18	Male	104800

Seleccionar con el nombre de una columna un dato determinado

In [46]:
```
1 datos.loc[datos['rank'] == 'Prof']
```

Out[46]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0	1	Prof	B	19	18	Male	139750
1	2	Prof	B	20	16	Male	173200
3	4	Prof	B	45	39	Male	115000
4	5	Prof	B	40	41	Male	141500
6	7	Prof	B	30	23	Male	175000
...	...	...	...	...	...	...	...
391	392	Prof	A	30	19	Male	151292
392	393	Prof	A	33	30	Male	103106
393	394	Prof	A	31	19	Male	150564
394	395	Prof	A	42	25	Male	101738
395	396	Prof	A	25	15	Male	95329

266 rows × 7 columns

Índice de filas		Índice de columnas						
		0	1	2	3	4	5	6
		order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
0		1	Prof	B	19	18	Male	139750
1		2	Prof	B	20	16	Male	173200
2		3	AsstProf	B	4	3	Male	79750
3		4	Prof	B	45	39	Male	115000
4		5	Prof	B	40	41	Male	141500
5		6	AssocProf	B	6	6	Male	97000
6		7	Prof	B	30	23	Male	175000
7		8	Prof	B	45	45	Male	147765
8		9	Prof	B	21	20	Male	119250
9		10	Prof	B	18	18	Female	129000
10		11	AssocProf	B	12	8	Male	119800
11		12	AsstProf	B	7	2	Male	79800
12		13	AsstProf	B	1	1	Male	77700
13		14	AsstProf	B	2	0	Male	78000
14		15	Prof	B	20	18	Male	104800

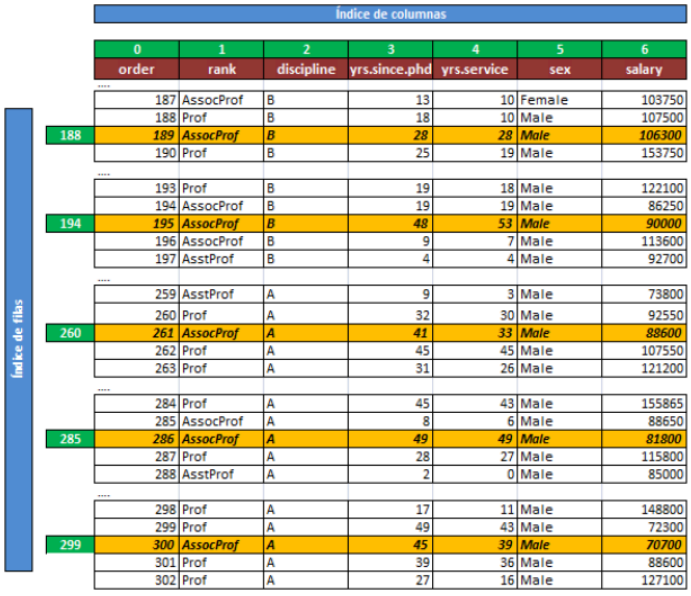
La tabla sigue...

Seleccionar filas usando múltiples condiciones

In [47]:
```
1 datos.loc[(datos['yrs.service'] > 25) & (datos['rank'] == 'AssocProf')]
```

Out[47]:

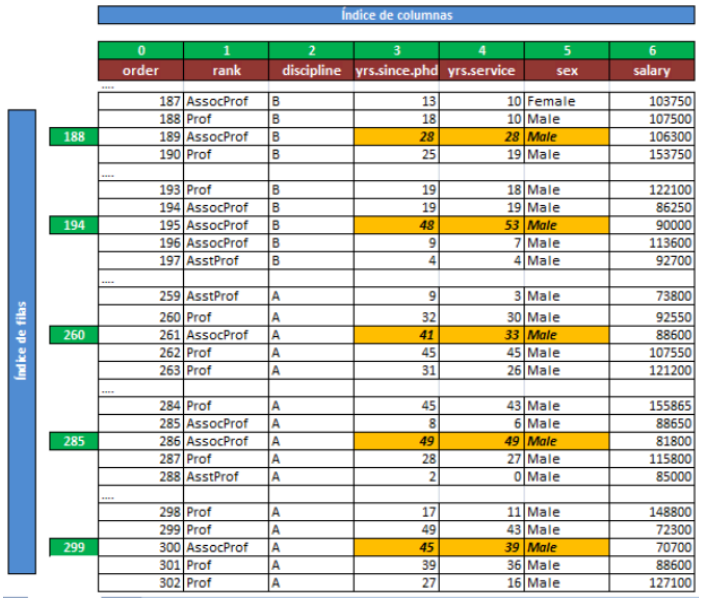
	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary
188	189	AssocProf	B	28	28	Male	106300
194	195	AssocProf	B	48	53	Male	90000
260	261	AssocProf	A	41	33	Male	88600
285	286	AssocProf	A	49	49	Male	81800
299	300	AssocProf	A	45	39	Male	70700



In [48]:
```
1 datos.loc[(datos['yrs.service'] > 25) & (datos['rank'] == 'AssocProf'), 'yrs.since.phd' : 'sex']
```

Out[48]:

	yrs.since.phd	yrs.service	sex
188	28	28	Male
194	48	53	Male
260	41	33	Male
285	49	49	Male
299	45	39	Male



Agregando columna y dato según una condición

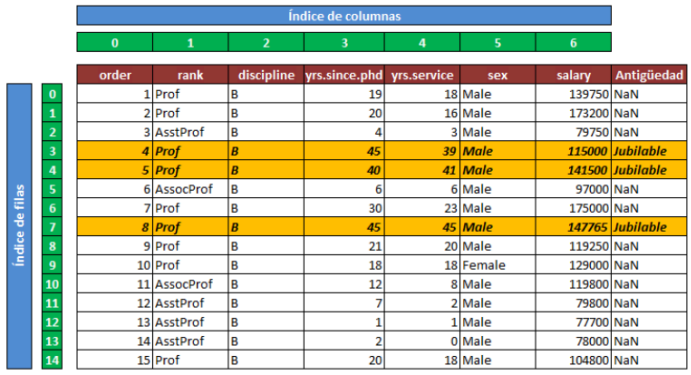
In [49]:
```
datos.loc[datos['yrs.service'] > 25, 'Antigüedad'] = 'Jubilable'
```

In [50]:
```
datos.loc[datos['yrs.service'] > 25, 'yrs.since.phd':'Antigüedad']
datos
```

Out[50]:

	order	rank	discipline	yrs.since.phd	yrs.service	sex	salary	Antigüedad
0	1	Prof	B	19	18	Male	139750	NaN
1	2	Prof	B	20	16	Male	173200	NaN
2	3	AsstProf	B	4	3	Male	79750	NaN
3	4	Prof	B	45	39	Male	115000	Jubilable
4	5	Prof	B	40	41	Male	141500	Jubilable
...	...	...	...	...	...	...	...	...
392	393	Prof	A	33	30	Male	103106	Jubilable
393	394	Prof	A	31	19	Male	150564	NaN
394	395	Prof	A	42	25	Male	101738	NaN
395	396	Prof	A	25	15	Male	95329	NaN
396	397	AsstProf	A	8	4	Male	81035	NaN

397 rows × 8 columns



La tabla sigue...