Combinando fuentes de datos

En ocasiones dispondremos de conjuntos de datos provenientes de diferentes fuentes y puede ser interesante combinarlos para obtener un nuevo conjunto de datos combinado.

Como puedes suponer, Pandas ofrece funciones para combinar conjuntos de datos. Este cuaderno se basa en esta referencia oficial.

Inicialización del entorno.

Lo primero será importar el paquete Pandas con alias pd. Posteriormente visualizamos la versión usada de Pandas ya que es un dato importante para consultar la documentación. En el momento de editar este notebook la versión de pandas es: 1.4.3

Además para facilitar los ejercicios también importamos el paquete Numpy con el alias np.

In [1]:
```
import pandas as pd
import numpy as np
np.set_printoptions(floatmode='fixed', precision=3)
print('Pandas versión: ', pd.__version__)
```

```
Pandas versión:  1.4.3
```

Concatenación de objetos.

Una forma de combinar dos objetos es concatenándolos. Para concatenar dos objetos Pandas DataFrames) se utiliza la función `pd.concat()`.

Podemos concatenar dos objetos Series, un objeto DataFrame un objeto Series y dos objetos DataFrame. Siempre que haya involucrado un objeto DataFrame, el objeto resultante es un DataFrame. Sólo obtendremos un objeto Series al concatenar dos objetos Series en el axis 0.

Ejercicio 01: Dadas dos objetos Series `s1` y `s2` queremos concatenarlas en una nueva serie `st`.

La salida debería ser algo parecido a lo siguiente:

```
S1:
 1    R
2    G
3    B
dtype: object
S2:
 4    W
5    M
6    Y
dtype: object
```

```
Concatenación:
 1    R
2    G
3    B
4    W
5    M
6    Y
dtype: object
```

In [2]:
```python
s1 = pd.Series(['R', 'G', 'B'])
print('S1:\n', s1)
s2 = pd.Series(['W', 'M', 'Y'])
print('\nS2:\n', s2)

st = None # la serie resultante de la concatenación.

#Pon tu código aquí.
#Sugerencia: utiliza la función concat().

#

print('\nConcatenación:\n', st)
```

```
S1:
 0    R
1    G
2    B
dtype: object

S2:
 0    W
1    M
2    Y
dtype: object

Concatenación:
 None
```

Ejercicio 02: dados dos dataframes `df1` y `df2`, queremos concatenarlos, aumentado las filas, para obtener del dataframe `dft`.

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B

Dataframe 2:
     A   B
3  3A  3B
4  4A  4B

Dataframe resultado:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B
4  4A  4B
```

In [3]:
```python
def gen_DF(Filas, Columnas):
    """Genera un DataFrame para probar."""
    data = {c:[str(i)+str(c) for i in Filas] for c in Columnas}
```

```
    return pd.DataFrame(data, index=list(Filas))

df1 = gen_DF('12', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('12', 'AB')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de concatenar df1 y df2.
#Pon tu código aquí.
#Sugerencia: usa la función concat().

#

print('\nDataframe resultado:\n', dft)
```

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B

Dataframe 2:
     A   B
1  1A  1B
2  2A  2B

Dataframe resultado:
 None
```

Ejercicio 03: dados dos dataframes `df1` y `df2`, queremos concatenarlos, aumentado las columnas, para obtener del dataframe `dft`.

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
Dataframe 1:
     C   D
1  1C  1D
2  2C  2D

Dataframe resultado:
     A   B   C   D
1  1A  1B  1C  1D
2  2A  2B  2C  2D
```

In [4]:

```
df1 = gen_DF('12', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('12', 'AB')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de concatenar df1 y df2.
#Pon tu código aquí.
#Sugerencia: usa la función concat() y da el valor correcto
#a el argumento axis.

#

print('\nDataframe resultado:\n', dft)
```

```
Dataframe 1:
     A   B
```

```
1  1A  1B
2  2A  2B

Dataframe 2:
     A   B
1  1A  1B
2  2A  2B

Dataframe resultado:
 None
```

Ejercicio 04: Dados un dataframe `df` y una serie `s`, queremos concatenarlos, aumentado las columnas, para obtener del dataframe `dft`.

```
Dataframe:
     A   B   C
1  1A  1B  1C
2  2A  2B  2C
3  3A  3B  3C

Serie:
 1    1D
2    2D
3    3D
Name: D, dtype: object

Dataframe resultado:
     A   B   C   D
1  1A  1B  1C  1D
2  2A  2B  2C  2D
3  3A  3B  3C  3D
```

In [5]:

```
df = gen_DF('123', 'ABC')
print('Dataframe:\n',df)

s = pd.Series(['1D', '2D', '3D'], index=list('123'), name='D')
print('\nSerie:\n', s)

dft = None #Dataframe resultado de concatenar df y s.
#Pon tu código aquí.
#Sugerencia: usa la función concat() y el atributo axis.

#

print('\nDataframe resultado:\n', dft)
```

```
Dataframe:
     A   B   C
1  1A  1B  1C
2  2A  2B  2C
3  3A  3B  3C

Serie:
 1    1D
2    2D
3    3D
Name: D, dtype: object

Dataframe resultado:
 None
```

Verificando duplicados.

Puede ocurrir que al unir dos objetos, existan valores de índice repetidos. Podemos tratar este problema de varias formas:

- Duplicar los valores duplicados en el resultado. Esto es lo que hemos estado realizando hasta este punto en los ejemplos anteriores.
- Detectar valores duplicados en los índices y abortar la operación. Para ello utilizamos el argumento `verify_integrity=True`. Activando esta opción, si hay valores duplicados se genera una excepción `ValueError`.
- Ignorar el índice con duplicados y generar un nuevo índice sin duplicados. Para ello utilizamos el argumento `ignore_index=True`.
- Generar en el resultado un multiindex a partir de los índices con duplicados. Indicaremos los nombres de los niveles con el argumento `keys`.

Ejercicio 05: Dados dos dataframes `df1` y `df2`, queremos concatenarlos, aumentado la filas, para obtener del dataframe `dft`. Se requiere detectar posibles valores de índices duplicados y en ese caso se abortará la concatenación y mostrar un mensaje de error.

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Concatenación con duplicados:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B
1  1A  1B
2  2A  2B
3  3A  3B

Error al concatenar df1 y df2.  Hay valores de índice duplicados.

Resultado final:
 None
```

In [6]:

```python
df1 = gen_DF('123', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('345', 'AB')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de concatenar df1 y df2.

#Pon tu código aquí.
#Sugerencia: usa un bloque try/except al usar función concat().
#   Activa la comprobación de duplicados y captura la excepción
```

```
#   ValueError para mostrar un mensaje de error.

#
```

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     A   B
3  3A  3B
4  4A  4B
5  5A  5B
```

Ejercicio 06: Dados dos dataframes `df1` y `df2`, queremos concatenarlos, aumentado la filas, para obtener del dataframe `dft`. Como es posible que haya valores de índice duplicados, se requiere que la operación genere un índice nuevo.

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Concatenación
     A   B
0  1A  1B
1  2A  2B
2  3A  3B
3  1A  1B
4  2A  2B
5  3A  3B
```

In [7]:
```
df1 = gen_DF('123', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('123', 'AB')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de concatenar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función concat() indicando que se ignore
#   los índices origen y se genere uno nuevo.

#

print('\nConcatenación\n', dft)
```

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B
```

```
Dataframe 2:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Concatenación
 None
```

Ejercicio 07: dados dos dataframes `df1` y `df2`, queremos concatenarlos, aumentado la filas, para obtener el dataframe `dft`. Como es posible que haya valores de índice duplicados, se requiere que la operación genere un multi índice con un nuevo nivel con las etiquetas `'df1'`, `'df2'`.

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Concatenación
         A   B
df1 1  1A  1B
    2  2A  2B
    3  3A  3B
df2 1  1A  1B
    2  2A  2B
    3  3A  3B
```

In [8]:
```
df1 = gen_DF('123', 'AB')
print('Dataframe 1:\n', df1)

df2 = gen_DF('123', 'AB')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de concatenar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función concat() indicando que se genere
#   un MultiIndex con un nuevo nivel con etiquetas 'df1' y 'df2'. Usa
#   el argumento keys para esto.

#

print('\nConcatenación\n', dft)
```

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     A   B
1  1A  1B
2  2A  2B
```

```
3  3A  3B

Concatenación
 None
```

Ordenando los índices.

Cuando concatenamos dos objetos, por defecto, las etiquetas del índice que no se usa para concatenar quedan en el mismo orden en que están orignialmente. Podemos utilizar el argumento `sort` para cambiar este comportamiento.

Ejercicio 08: dados dos dataframes `df1` y `df2`, queremos concatenarlos, aumentado la filas, para obtener el dataframe `dft`. Como es posible que haya valores de índice duplicados, se requiere que la operación genere un multi índice con un nuevo nivel con los valores `'df1'`, `'df2'`. Además se requiere que las etiquetas de columna queden ordenadas.

```
Dataframe 1:
     C   A   B
1  1C  1A  1B
2  2C  2A  2B
3  3C  3A  3B

Dataframe 2:
     B   C   A
1  1B  1C  1A
2  2B  2C  2A
3  3B  3C  3A

Concatenación (sin ordenar):
         C   A   B
df1 1  1C  1A  1B
    2  2C  2A  2B
    3  3C  3A  3B
df2 1  1C  1A  1B
    2  2C  2A  2B
    3  3C  3A  3B

Concatenación (ordenando):
         A   B   C
df1 1  1A  1B  1C
    2  2A  2B  2C
    3  3A  3B  3C
df2 1  1A  1B  1C
    2  2A  2B  2C
    3  3A  3B  3C
```

In [9]:

```
df1 = gen_DF('123', 'CAB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('123', 'BCA')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de concatenar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función concat() indicando que se genere
#   un MultiIndex con un nuevo nivel con etiquetas 'df1' y 'df2'. Usa
```

```
#   el argumento keys para esto.

#
print('\nConcatenación (sin ordenar):\n', dft)

dft = None #Dataframe resultado de concatenar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función concat() indicando que se genere
#   un MultiIndex con un nuevo nivel con etiquetas 'df1' y 'df2'. Usa
#   el argumento keys para esto.
#Sugerencia: Indica que se ordene el indice de columnas
#   con el argumento sort.

#

print('\nConcatenación (ordenando):\n', dft)
```

```
Dataframe 1:
     C   A   B
1  1C  1A  1B
2  2C  2A  2B
3  3C  3A  3B

Dataframe 2:
     B   C   A
1  1B  1C  1A
2  2B  2C  2A
3  3B  3C  3A

Concatenación (sin ordenar):
 None

Concatenación (ordenando):
 None
```

Gestión de valores perdidos.

Al producirse la concatenación puede ocurrir que, para él índice que no se usa para concatenar, haya etiquetas de índice diferentes en los objetos a concatenar. Esto provocará que en el objeto resultante de la concatenación aparezcan valores perdidos (`NAN`). Este es el comportamiento por defecto indicado con el argumento `join='outer'` (método de unión externa).

De forma alternativa, podemos indicar que para realizar la concatenación, la unión sea realizada usando sólo las etiquetas que coinciden (intersección) en las fuentes de datos. Este comportamiento se indica con el argumento `join='inner'` (método de unión interna).

Ejercicio 09: Dados dos dataframes `df1` y `df2` , queremos concatenarlos, aumentado la filas, para obtener el dataframe `dft` . Queremos comparar los resultados obtenidos con los métodos de unión externa (`dft1`) y e interna (`dft2`).

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     B   C
```

```
   4  4B  4C
   5  5B  5C
   6  6B  6C

   Concatenación con unión externa:
         A   B    C
   1   1A  1B  NaN
   2   2A  2B  NaN
   3   3A  3B  NaN
   4  NaN  4B   4C
   5  NaN  5B   5C
   6  NaN  6B   6C

   Concatenación con unión interna:
        B
   1  1B
   2  2B
   3  3B
   4  4B
   5  5B
   6  6B
```

In [10]:

```python
df1 = gen_DF('123', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('456', 'BC')
print('\nDataframe 2:\n',df2)

dft1 = None
#Pon tu código aquí.
#Sugerencia: utiliza la función concat() indicando unión externa
#   con el argumento join.

#

dft2 = None
#Pon tu código aquí.
#Sugerencia: utiliza la función concat() indicando unión interna
#   con el argumento join

#

print('\nConcatenación con unión externa:\n', dft1)
print('\nConcatenación con unión interna:\n', dft2)
```

```
Dataframe 1:
     A   B
1  1A  1B
2  2A  2B
3  3A  3B

Dataframe 2:
     B   C
4  4B  4C
5  5B  5C
6  6B  6C

Concatenación con unión externa:
 None

Concatenación con unión interna:
 None
```

Combinar dos objetos usando un campo clave para mezclar.

Pandas ofrece la función `merge()` para combinar dos fuentes utilizando un campo clave. La forma de proceder es similar a las operaciones de combinación usadas en Bases de Datos Relacionales usando el lenguaje `SQL`.

Combinaciones 1-a-1.

Este tipo de combinación ocurre cuando el campo clave usado en ambos objetos a combinarlos tiene valores únicos.

Ejemplo 08: Dados dos dataframes `df1` y `df2` se quiere obtener un dataframe `dft` combinándolos. Se utilizará la columna 'Key' como clave de la combinación.

```
   Dataframe 1:
      Key   A   B
   1   1  1A  1B
   2   2  2A  2B
   3   3  3A  3B

   Dataframe 2:
      Key   C   D
   4   1  4C  4D
   5   2  5C  5D
   6   3  6C  6D

   Resultado de la mezcla 1-a-1:
      Key   A   B   C   D
   0   1  1A  1B  4C  4D
   1   2  2A  2B  5C  5D
   2   3  3A  3B  6C  6D
```

In [11]:
```
def gen_DF(Keys, Filas, Columnas):
    """Genera un DataFrame para probar."""
    d={}
    d['Key']=list(Keys)
    for c in Columnas:
        d[c] = [str(i)+c for i in Filas]
    return pd.DataFrame(d, index=list(Filas))

df1 = gen_DF('123', '123', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('123', '456', 'CD')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de mezclar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función merge(). Utiliza el argumento 'on'
#   para indicar la columna usada como clave.

#

print('\nResultado de la mezcla 1-a-1:\n', dft)
```

```
Dataframe 1:
```

```
   Key   A   B
1   1  1A  1B
2   2  2A  2B
3   3  3A  3B

Dataframe 2:
   Key   C   D
4   1  4C  4D
5   2  5C  5D
6   3  6C  6D

Resultado de la mezcla 1-a-1:
 None
```

Observa que para el resultado se ha generado un nuevo índice de filas.

Combinaciones 1-a-n.

En este caso, en uno de los objetos a combinar, el campo clave tiene valores repetidos. El resultado de la combinación preservará los valores duplicados.

Ejemplo 09: Dados dos dataframes `df1` y `df2` se quiere obtener un dataframe `dft` combinándolos. Se utilizará la columna 'Key' como clave de la combinación.

```
   Dataframe 1:
      Key   A   B
   1   0  1A  1B
   2   1  2A  2B
   3   2  3A  3B
   4   3  4A  4B

   Dataframe 2:
      Key   C   D
   1   0  1C  1D
   2   3  2C  2D
   3   1  3C  3D
   4   2  4C  4D
   5   0  5C  5D

   Resultado de la mezcla 1-n:
      Key   A   B   C   D
   0   0  1A  1B  1C  1D
   1   0  1A  1B  5C  5D
   2   1  2A  2B  3C  3D
   3   2  3A  3B  4C  4D
   4   3  4A  4B  2C  2D
```

In [12]:
```python
df1 = gen_DF('0123','1234', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('03120','12345', 'CD')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de mezclar df1 y df2.
#Pon tu código aquí.
#Sugerencia: utiliza la función merge() indicando la columna 'Key'
#   como clave de combinación usando el argumento on.
```

```
#

print('\nResultado de la mezcla 1-n:\n', dft)
```

```
Dataframe 1:
   Key   A   B
1   0  1A  1B
2   1  2A  2B
3   2  3A  3B
4   3  4A  4B

Dataframe 2:
   Key   C   D
1   0  1C  1D
2   3  2C  2D
3   1  3C  3D
4   2  4C  4D
5   0  5C  5D

Resultado de la mezcla 1-n:
 None
```

Combinaciones n-a-m.

Este caso es el más genérico y ocurre cuando el campo clave tiene valores repetidos en ambos objetos. El resultado será el producto cartesiano de ambas columnas.

Ejemplo 10: Dados dos dataframes `df1` y `df2` se quiere obtener un dataframe `dft` combinándolos. Se utilizará la columna 'Key' como clave de la combinación.

```
Dataframe 1:
   Key   A   B
1   0  1A  1B
2   0  2A  2B
3   1  3A  3B

Dataframe 2:
   Key   C   D
1   0  1C  1D
2   1  2C  2D
3   1  3C  3D
4   0  4C  4D

Resultado de la mezcla n-m:
   Key   A   B   C   D
0   0  1A  1B  1C  1D
1   0  1A  1B  4C  4D
2   0  2A  2B  1C  1D
3   0  2A  2B  4C  4D
4   1  3A  3B  2C  2D
5   1  3A  3B  3C  3D
```

In [13]:
```
df1 = gen_DF('001','123', 'AB')
print('Dataframe 1:\n',df1)

df2 = gen_DF('0110', '1234', 'CD')
print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de mezclar df1 y df2.
```

```
#Pon tu código aquí.
#Sugerencia: utiliza la función merge() indicando la columna 'Key'
#   como clave de combinación usando el argumento on.

#

print('\nResultado de la mezcla n-m:\n', dft)
```

```
Dataframe 1:
   Key   A   B
1   0  1A  1B
2   0  2A  2B
3   1  3A  3B

Dataframe 2:
   Key   C   D
1   0  1C  1D
2   1  2C  2D
3   1  3C  3D
4   0  4C  4D

Resultado de la mezcla n-m:
 None
```

Otras opciones de merge.

Usar dos columnas distintas como campo clave.

Hasta ahora en los ejemplos vistos, ambos objetos han usado el mismo nombre para la columna usada como clave de combinación. Sin embargo es una situación muy común que el nombre de la columna sea distinto en cada dataframe. Este caso usaremos los parámetros `left_on` y `right_on` para indicar la columna del primer objeto y del segundo objeto respectivamente.

En el dataframe resultante de la combinación tendrá las dos columnas indicadas por lo suele ser comun usar el método `Dataframe.drop()` para eliminar una de las dos ya que los valores son redundantes.

Ejercicio 11: Dados dos dataframes `df1` y `df2` se requiere combinarlos usando las columnas `Nombre` y `Alumno` respectivamente como clave de combinación. Eliminar la columna `'Alumno'` del resultado.

```
Dataframe 1:
     Nombre           Email
0     Juan     juan@tu.com
1  Antonio  antonio@tu.com
2     Luis     luis@tu.com
3    David    david@tu.com

Dataframe 2:
     Alumno  Nota
0     Juan   6.5
1  Antonio   8.9
2     Luis   5.5
3     Pepe   6.3

Resultado de la mezcla:
     Nombre           Email  Nota
0     Juan     juan@tu.com   6.5
1  Antonio  antonio@tu.com   8.9
2     Luis     luis@tu.com   5.5
```

In [14]:
```
df1 = pd.DataFrame({'Nombre': ['Juan', 'Antonio', 'Luis', 'David'],
    'Email': ['juan@tu.com', 'antonio@tu.com', 'luis@tu.com', 'david@tu.com']})

print('Dataframe 1:\n',df1)

df2 = pd.DataFrame({'Alumno': ['Luis', 'Juan', 'David', 'Antonio'],
    'Nota': [6.5, 8.9, 5.5, 6.3]})

print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de mezclar df1 y df2.
#Pon tu código aquí.
#Sugerencia: utiliza la función merge() indicando la columna 'Nombre'
#  como clave izquierda y la columna 'Alumno' como clave derecha.
#  Utiliza el método .drop() en el dataframe resultante para
#  eliminar la columna 'Alumno'.

#

print('\nResultado de la mezcla:\n', dft)
```

```
Dataframe 1:
     Nombre           Email
0     Juan     juan@tu.com
1  Antonio  antonio@tu.com
2     Luis     luis@tu.com
3    David    david@tu.com

Dataframe 2:
     Alumno  Nota
0     Luis   6.5
1     Juan   8.9
2    David   5.5
3  Antonio   6.3

Resultado de la mezcla:
 None
```

Usar los índices como campo clave.

También es posible usar los valores del índice de filas como clave de combinación. Para ello utilizamos los parámetros `left_index=True` o `right_index=True` según el lado donde queremos usar el índice.

Ejercicio 12: Dados dos dataframes `df1` y `df2` se requiere combinarlos usando el índice de filas como clave de combinación.

```
Dataframe 1:
                   Email
Nombre
Juan         juan@tu.com
Antonio   antonio@tu.com
Luis         luis@tu.com
David       david@tu.com

Dataframe 2:
          Nota
Alumno
Luis       6.5
Juan       8.9
David      5.5
Antonio    6.3
```

```
Resultado de la mezcla:
                    Email  Nota
Juan          juan@tu.com   8.9
Antonio  antonio@tu.com   6.3
Luis          luis@tu.com   6.5
David       david@tu.com   5.5
```

In [15]:

```python
df1 = pd.DataFrame({'Nombre': ['Juan', 'Antonio', 'Luis', 'David'],
    'Email': ['juan@tu.com', 'antonio@tu.com', 'luis@tu.com', 'david@tu.com']})
df1 = df1.set_index('Nombre')
print('Dataframe 1:\n',df1)

df2 = pd.DataFrame({'Alumno': ['Luis', 'Juan', 'David', 'Antonio'],
    'Nota': [6.5, 8.9, 5.5, 6.3]})
df2 = df2.set_index('Alumno')
print('\nDataframe 2:\n',df2)


dft = None #Dataframe resultado de mezclar df1 y df2.
#Pon tu código aquí.
#Sugerencia: utiliza la función merge() indicando que se
#utilicen los índices de fila tanto en la derecha como en la
#izquierda.

#

print('\nResultado de la mezcla:\n', dft)
```

```
Dataframe 1:
                    Email
Nombre
Juan          juan@tu.com
Antonio  antonio@tu.com
Luis          luis@tu.com
David       david@tu.com

Dataframe 2:
          Nota
Alumno
Luis       6.5
Juan       8.9
David      5.5
Antonio    6.3

Resultado de la mezcla:
 None
```

Especificando el tipo de combinación.

Hasta el momento la combinación de los dos objetos se realiza obteniendo una unión interna (*inner*), esto quiere decir utilizar la intersección de los campos claves, por lo que, si un valor clave no existe en alguno de las columnas, los datos asociados no aparecerán en la combinación. Este es el comportamiento por defecto (`how='inner'`).

Además podemos indicar con el parámetro `how` otros comportamientos:

- El método `how='outer'` utiliza la unión de los valores clave.
- El método `how='left'` utiliza los valores clave sólo del objeto izquierdo.
- El método `how='right'` utiliza los valores clave sólo del objeto derecho.
- El método `how='cross'` utiliza el producto cartesiano de los valores clave del objeto izquierdo y derecho.

Con estos métodos se utilizará el valor `NAN` en dataframe resultante para aquelos valores de columna para los que no existan datos.

Ejercicio 13: Dados dos dataframes `df1` y `df2` se requiere combinarlos usando las columnas `Nombre` y `Alumno` respectivamente como clave de combinación. Se ha detectado que pueden existir alumnos con nota que aún no han sido registrados, por lo que se requiere que apararezcan todos los datos completando con NAN los valores incompletos.

```
Dataframe 1:
     Nombre           Email
0     Juan     juan@tu.com
1  Antonio  antonio@tu.com
2     Luis     luis@tu.com
3    David    david@tu.com

Dataframe 2:
     Alumno  Nota
0     Luis   6.5
1     Juan   8.9
2    Mauro   5.5
3  Antonio   6.3

Resultado de la mezcla:
     Nombre           Email   Alumno  Nota
0     Juan     juan@tu.com     Juan   8.9
1  Antonio  antonio@tu.com  Antonio   6.3
2     Luis     luis@tu.com     Luis   6.5
3    David    david@tu.com      NaN   NaN
4      NaN             NaN    Mauro   5.5
```

In [16]:
```
df1 = pd.DataFrame({'Nombre': ['Juan', 'Antonio', 'Luis', 'David'],
    'Email': ['juan@tu.com', 'antonio@tu.com', 'luis@tu.com', 'david@tu.com']})

print('Dataframe 1:\n',df1)

df2 = pd.DataFrame({'Alumno': ['Luis', 'Juan', 'Mauro', 'Antonio'],
    'Nota': [6.5, 8.9, 5.5, 6.3]})

print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de mezclar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función merge() indicando la columna 'Nombre'
#  como clave izquierda y la columna 'Alumno' como clave derecha.
#  Utiliza el método 'outer' para que no se descarten datos.

#

print('\nResultado de la mezcla:\n', dft)
```

```
Dataframe 1:
     Nombre           Email
0     Juan     juan@tu.com
1  Antonio  antonio@tu.com
2     Luis     luis@tu.com
3    David    david@tu.com

Dataframe 2:
     Alumno  Nota
```

```
0     Luis   6.5
1     Juan   8.9
2    Mauro   5.5
3  Antonio   6.3

Resultado de la mezcla:
 None
```

Ejercicio 14: Dados dos dataframes `df1` y `df2` se requiere combinarlos usando las columnas `Nombre` y `Alumno` respectivamente como clave de combinación. Se ha detectado que pueden existir alumnos con nota que aún no han sido registrados, por lo que se requiere sólo utilizar los valores de clave de alumnos registrados (dataframe izquierdo `df1`).

```
Dataframe 1:
     Nombre           Email
0     Juan     juan@tu.com
1  Antonio  antonio@tu.com
2     Luis     luis@tu.com
3    David    david@tu.com

Dataframe 2:
     Alumno  Nota
0     Luis   6.5
1     Juan   8.9
2    Mauro   5.5
3  Antonio   6.3

Resultado de la mezcla:
     Nombre           Email   Alumno  Nota
0     Juan     juan@tu.com     Juan   8.9
1  Antonio  antonio@tu.com  Antonio   6.3
2     Luis     luis@tu.com     Luis   6.5
3    David    david@tu.com      NaN   NaN
```

In [17]:
```python
df1 = pd.DataFrame({'Nombre': ['Juan', 'Antonio', 'Luis', 'David'],
    'Email': ['juan@tu.com', 'antonio@tu.com', 'luis@tu.com', 'david@tu.com']})

print('Dataframe 1:\n',df1)

df2 = pd.DataFrame({'Alumno': ['Luis', 'Juan', 'Mauro', 'Antonio'],
    'Nota': [6.5, 8.9, 5.5, 6.3]})

print('\nDataframe 2:\n',df2)

dft = None #Dataframe resultado de mezclar df1 y df2.

#Pon tu código aquí.
#Sugerencia: utiliza la función merge() indicando la columna 'Nombre'
#   como clave izquierda y la columna 'Alumno' como clave derecha.
#   Utiliza el método 'left' para forzar a utilizar sólo las claves
#   del dataframe df1.

#

print('\nResultado de la mezcla:\n', dft)
```

```
Dataframe 1:
     Nombre           Email
0     Juan     juan@tu.com
1  Antonio  antonio@tu.com
2     Luis     luis@tu.com
```

```
3    David    david@tu.com

Dataframe 2:
     Alumno  Nota
0     Luis   6.5
1     Juan   8.9
2    Mauro   5.5
3  Antonio   6.3

Resultado de la mezcla:
 None
```

In [ ]: