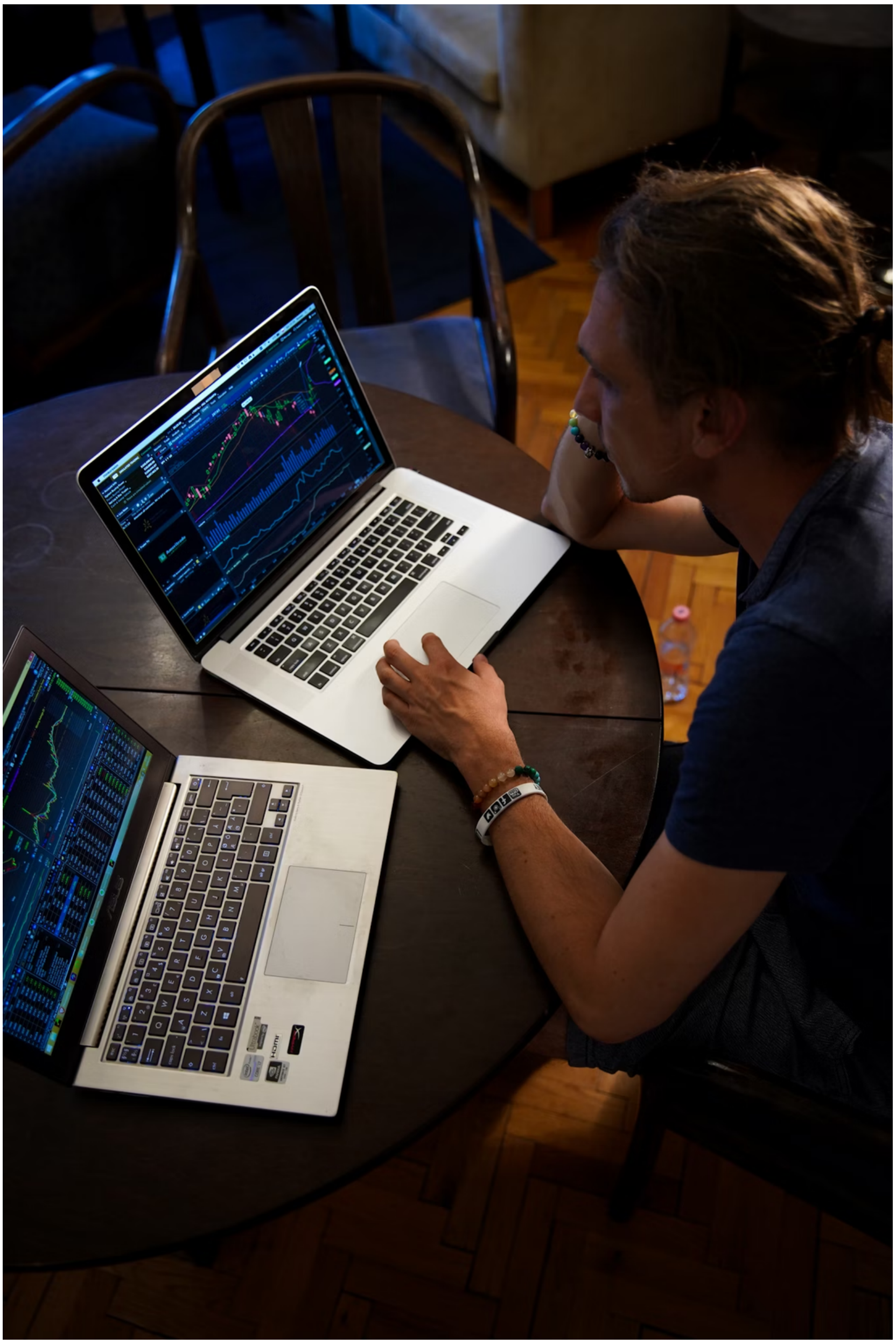
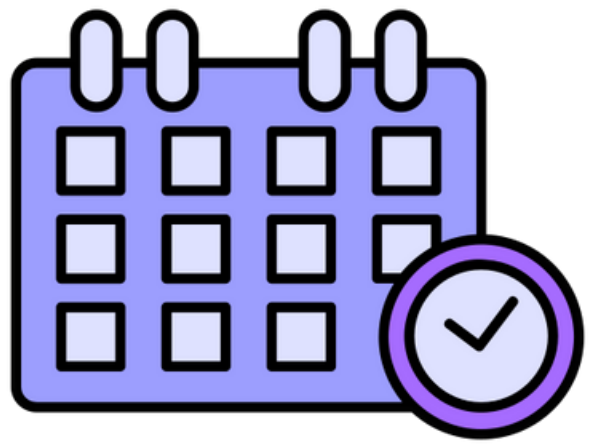
Lección 2: Limpieza y preparación de datos



Planteamiento de la sesión

Tiempo de ejecución: 4 horas



Una de las actividades más comunes a la hora de estar trabajando con datasets, es aplicar algunas limpiezas sobre los datos para minimizar la cantidad de datos que no aportan valor y darle un peso más relevante a datos que van a sumar valor.

A continuación se presentan algunas estrategias para la limpieza y preparación de DataFrames.

Materiales:

- pandas - Aprende Python
- La librería Pandas | Aprende con Alf

Eliminar filas con NaN, Nulos o datos desconocidos

El método dropna(subset=columns) elimina la fila completa de cada uno de los registros en donde, en la columna indicada, se encuentren valores nulos o no válidos.

Si no se le especifica una columna particular, va a buscar los valores en todo el DataFrame y a eliminar todas las filas que contengan algún valor nulo.

```
import pandas as pd
df = pd.read_csv(
'https://raw.githubusercontent.com/asalber/manual-python/master/datos/colesterol.csv')

print(df.dropna(subset='colesterol'))
```

También es posible darle manejo a los valores nulos o desconocidos rellenando con algún valor sustituto, como 0 por ejemplo.

```
import pandas as pd
df = pd.read_csv(
'https://raw.githubusercontent.com/asalber/manual-python/master/datos/colesterol.csv')
print(df.fillna(0))
```

¿Qué hace el método df.interpolate() con los valores nulos o desconocidos?

Agrupar y dividir un DataFrame

Es posible que sea necesario agrupar los datos de un DataFrame de acuerdo con valores de una columna o de varias, algunos ejemplos puede ser agrupar usuarios por paises, rangos de edades o sexo.

```
import pandas as pd
df = pd.read_csv(
'https://raw.githubusercontent.com/asalber/manual-python/master/datos/colesterol.csv')

print(df.groupby('sexo').groups)
```

El método retorna un diccionario en donde las claves son los valores disponibles en la columna seleccionada y los valores, son listas con los índices de cada elemento que coincide con su respectiva clave.

O con get_group() para obtener un DataFrame con los valores que cumplen con la agrupación.

```
import pandas as pd
df = pd.read_csv(
'https://raw.githubusercontent.com/asalber/manual-python/master/datos/colesterol.csv')

print(df.groupby('sexo').get_group('H'))
print(df.groupby('sexo').get_group('M'))
```

Reestructurar un DataFrame

Los DataFrames pueden estructurarse de dos formas, utilizando el formato ancho o el formato largo.

Formato ancho

Nombre	Economía	Matemáticas	Programación
Carmen	5.0	3.5	9.0
Luis	6.5	7.0	4.0
María	8.0	8.5	6.5

Formato largo

Nombre	Asignatura	Nota
Carmen	Economía	5.0
Luis	Economía	6.5
María	Economía	8.0
Carmen	Matemáticas	3.5
Luis	Matemáticas	7.0
María	Matemáticas	8.5
Carmen	Programación	9.0
Luis	Programación	4.0
María	Programación	6.5

Fuente: Aprende con alf, DataFrames, consultado en febrero de 2024

El paso entre uno u otro, depende de la necesidad del análisis que se esté trabajando, puede que un formato permite graficar los datos de mejor manera que otro o filtrar o unir.

Para realizar la conversión a formato largo se utiliza el método .melt().

#df.melt(id_vars=id-columnas, value_vars=columnas, var_name=nombre-columnas, var_value=nombre-valores)

```
import pandas as pd
datos = {'nombre':['María', 'Luis', 'Carmen'],
'edad':[18, 22, 20],
'Matemáticas':[8.5, 7, 3.5],
'Economía':[8, 6.5, 5],
'Programación':[6.5, 4, 9]}
df = pd.DataFrame(datos)

print(df)

df1 = df.melt(id_vars=['nombre', 'edad'], var_name='asignatura',
value_name='nota')
print(df1)
```

El método funciona así, todas las columnas que se encuentran en la lista ‘columnas’ de value_vars, se reestructuran en dos nuevas columnas ‘nombre-columnas’ y ‘nombre-valores’, dichas nuevas columnas contienen los nombres de las columnas originales y sus valores.

Las columnas en la lista ‘id-columnas’ se mantienen sin reestructurar, si esta lista ‘columnas’ no se pasa al método, entonces reestructura todas las columnas menos las indicadas en la lista ‘id-columnas’.

Para convertir un DataFrame a formato ancho se utiliza el método .pivot().

El método funciona así, recibe un index al que se le pasa una lista con los nombres del índice del nuevo DataFrame, recibe un atributo columns que crea tantas columnas nuevas como valores diferentes contenga la columna que se le asigna, los nombres de cada columna nueva, corresponde con cada valor diferente que se encuentre en la columna indicada y sus valores se toman de la columna que se le pasa a un tercer parámetro de la función llamado values.

Partiendo del ejemplo anterior donde se cuenta con un DataFrame en formato largo, para pasarlo a uno en formato ancho es necesario:

```
print(df1.pivot(index=['nombre', 'edad'], columns=['asignatura'],
values='nota'))
```

Combinar DataFrames

Varios Dataframes pueden ser unidos en un nuevo DataFrame y para esta acción se cuenta con dos posibilidades, realizar la combinación por concatenación o por mezcla.

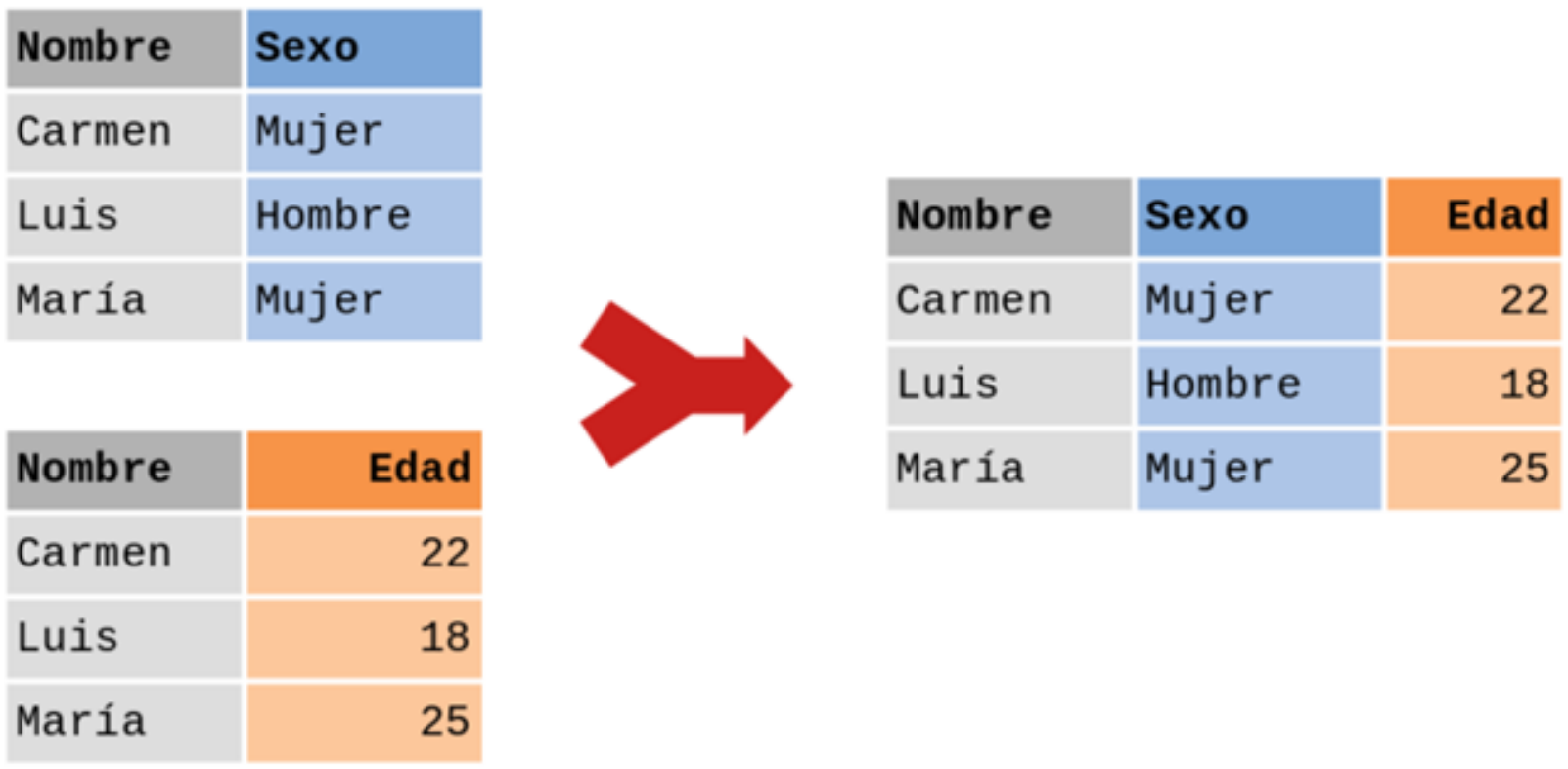
Concatenación

Cuando se cuente con dos o más DataFrames que tengan el mismo índice de columnas es posible realizar una concatenación de filas, lo cual consiste en agregar un grupo de filas a continuación de las existentes.

Fuente: Concatenación por filas, aprendeconalf, consultado en febrero de 2024, disponible en: https://aprendeconalf.es/docencia/python/manual/img/pandas-concatenacion-filas.png_

```
#por filas

import pandas as pd

df1 = pd.DataFrame({"Nombre":["Carmen", "Luis"],
"Sexo":["Mujer", "Hombre"], "Edad":[22, 18]}).set_index("Nombre")

df2 = pd.DataFrame({"Nombre":["María", "Pedro"],
"Sexo":["Mujer", "Hombre"], "Edad":[25, 30]}).set_index("Nombre")

df = pd.concat([df1, df2])

print(df)
```

Si lo que se busca es hacer una concatenación por columnas, ahora es necesario que los DataFrames que se combinen tengan el mismo índice de filas.



Fuente: Concatenación por columnas, aprendeconalf, consultado en febrero de 2024, disponible en: https://aprendeconalf.es/docencia/python/manual/img/pandas-concatenacion-columnas.png_

```
#por columnas

import pandas as pd
df1 = pd.DataFrame({"Nombre":["Carmen", "Luis", "María"],
"Sexo":["Mujer", "Hombre", "Mujer"]}).set_index("Nombre")

print(df1)

df2 = pd.DataFrame({"Nombre":["Carmen", "Luis", "María"],
 "Edad":[22, 18, 25]}).set_index("Nombre")

print(df2)

df = pd.concat([df1, df2], axis = 1)

print(df)
```

Para realizar dicha funcionalidad se utiliza el método .concat(), el método .concat() recibe dos argumentos, el primero es el listado de los DataFrames a unir y el segundo es el axis, en donde por defecto o 0 indica concatenación por filas y 1 indica que será por columnas.

Mezcla de DataFrames

Es posible mezclar las filas de dos DataFrames que contiene información en común, en una, varias columnas o índices conocidas como clave. El método para aplicar la mezcla es .merge() y tiene 4 formas de cómo hacer el merge.

Fuente: Fusión de DataFrames, aprendepython.es, consultado en febrero de 2024, disponible en: https://aprendepython.es/pypi/datascience/pandas/#uniendo-dataframes

df.merge(df1, df2, on = clave, how = tipo): Devuelve el DataFrame que resulta de mezclar el DataFrame df2 con el DataFrame df1, usando como claves las columnas de la lista clave y siguiendo el método de mezcla indicado por tipo.

```
import pandas as pd

df1 = pd.DataFrame({"Nombre":["Carmen", "Luis", "María"], "Sexo":["Mujer",
"Hombre", "Mujer"]})

print(df1)
print("\n\n");

df2 = pd.DataFrame({"Nombre":["María", "Pedro", "Luis"], "Edad":[25, 30, 18]})
print(df2)
print("\n\n");



print(pd.merge(df1, df2, on="Nombre"))
print("\n\n");
print(pd.merge(df1, df2, on="Nombre", how="outer"))
print("\n\n");
print(pd.merge(df1, df2, on="Nombre", how="left"))
print("\n\n");
print(pd.merge(df1, df2, on="Nombre", how="right"))
```

inner

El DataFrame resultante solo contiene las filas cuyos valores en la clave están en los dos DataFrames. Es equivalente a la intersección de conjuntos.

outer

El DataFrame resultante contiene todas las filas de los dos DataFrames. Si una fila de un DataFrame no puede emparejarse con otra los mismos valores en la clave en el otro DataFrame, la fila se añade igualmente al DataFrame resultante rellenando las columnas del otro DataFrame con el valor NaN. Es equivalente a la unión de conjuntos

left

El DataFrame resultante contiene todas las filas del primer DataFrame y descarta las filas del segundo DataFrame que no pueden emparejarse con alguna fila del primer DataFrame a través de la clave.

right

El DataFrame resultante contiene todas las filas del segundo DataFrame y descarta las filas del primer DataFrame que no pueden emparejarse con alguna fila del segundo DataFrame a través de la clave.

¿Qué hace la siguiente línea?

```
print(pd.merge(df1, df2, how='cross'))
```