

In [8]:

```
#importamos las librerías necesarias
import pandas as pd
import glob
import os

#Recuperamos todos los ficheros que tenemos guardados de enfermedades

path = 'datasets enfermedades/' #librería donde tengas los datasets creados anterior
mente
fmask = os.path.join(path, '*.csv')

# Primero especificamos un patrón del archivo y lo pasamos como parámetro en la función
glob
csv_files = glob.glob(fmask)

# Mostrar el archivo csv_files, el cual es una lista de nombres
print(csv_files)

['datasets enfermedades\\AER_for_2017-Zika-virus-disease.csv', 'datasets e
nfermedades\\chikungunya-annual-epidemiological-report-2018.csv', 'dataset
s enfermedades\\chikungunya-virus-disease-annual-epidemiological-report-20
17.csv', 'datasets enfermedades\\dengue-annual-epidemiological-report-201
7.csv', 'datasets enfermedades\\dengue-annual-epidemiological-report-2018.
csv', 'datasets enfermedades\\Zika-annual-epidemiological-report-2018.cs
v']
```

In [9]:

```
#Función para cambiar las columnas de nombre, ya que muchas son iguales y para especificar enfermedad en la columna
#Así conoceremos a qué enfermedad pertenece cada columna
def Changed_Column(table, w_filename):

    #Evaluamos las 3 enfermedades que tenemos. Según la enfermedad y como todas tienen la columna 'Reported', renombramos
    #dicha columna con el nombre de la enfermedad que toca.
    if w_filename.count('Zika') > 0:
        table.rename(columns=lambda x: x.replace('Reported', 'Zika'), inplace=True)
    elif w_filename.count('dengue') > 0:
        table.rename(columns=lambda x: x.replace('Reported', 'Dengue'), inplace=True)
    #Renombramos 'Confirmed cases' y la ponemos igual que en el caso del Chikunbun
    ya a 'Confirmed' a secas
        table.rename(columns=lambda x: x.replace('Confirmed cases', 'Confirmed'), inplace=True)
    elif w_filename.count('chikungunya') > 0:
        table.rename(columns=lambda x: x.replace('Reported', 'Chikungunya'), inplace=True)

    #No todos los datasets tienen 'Rate' por tanto si con tienen este en particular es que tiene varios.
    #Eliminamos todos los Rate. Evaluamos el año porque tenemos a partir del año 4 años atrás, por tanto
    #los ficheros de 2018 empiezan en 2014 y los de 2017 en 2013
    if 'Rate_2016' in table.columns:
        if w_filename.count('2018'):
            # Nos guardamos el número anterior donde se encuentra 'Rate_2014', para poder eliminarlo
            num = table.columns.get_loc('Rate_2014') - 1

            # Nos guardamos el número anterior donde se encuentra 'ASR_2018', justo es donde tendremos 'Rate_2018'
            max = table.columns.get_loc('ASR_2018') - 1

            #Recorreremos cada dos ya que tenemos Reported y Rate del mismo año.
            #empezamos por Rate_2018 y vamos hacia atrás para eliminar las últimas columnas primero (así no tendremos
            #problemas con número de columna, ya que si eliminamos una primera (la 2 por ejemplo), la numeración cambia)
            for i in range(max, num, -2):
                del table[table.columns[i]]

        elif w_filename.count('2017'):
            ## Nos guardamos el número anterior donde se encuentra 'Rate_2013', para poder eliminarlo
            num = table.columns.get_loc('Rate_2013') - 1

            # Nos guardamos el número anterior donde se encuentra 'Confirmed', justo es donde tendremos 'Rate_2017'
            max = table.columns.get_loc('Confirmed') - 1

            #Recorreremos cada dos ya que tenemos Reported y Rate del mismo año.
            #empezamos por Rate_2017 y vamos hacia atrás para eliminar las últimas columnas primero (así no tendremos
            #problemas con número de columna, ya que si eliminamos una primera (la 2 por ejemplo), la numeración cambia)
            for i in range(max, num, -2):
                del table[table.columns[i]]
```

```

#Eliminamos la columna 'Confirmed' quien la tenga
if 'Confirmed' in table.columns:
    del table[table.columns[table.columns.get_loc('Confirmed')]]

#Eliminamos la columna 'ASR_2018' quien la tenga
if 'ASR_2018' in table.columns:
    del table[table.columns[table.columns.get_loc('ASR_2018')]]

#retornamos la tabla
return table

```

In [10]:

```

# Escribimos un loop que irá a través de cada uno de los nombres de archivo

for i, filename in enumerate(csv_files):
    if i == 0:
        #La primera vez nos guardamos el dataset entero. Lo pasamos por la función c
        reada por si ha de eliminar alguna
        #columna
        data = Changed_Column(pd.read_csv(filename, sep='\t', index_col=0), filename
e)

    else:
        #La segunda vez pasamos el dataset por función creada para que elimine colu
        mnas no necesarias
        df2 = Changed_Column(pd.read_csv(filename, sep='\t', index_col=0), filename
        )

        #unimos los datasets
        data = pd.concat([data, df2], axis=1)

#Mostramos cabecera del dataset resultante de todos
data.head()

```

Out[10]:

	Country	Zika_2015	Zika_2016	Zika_2017	Country	Chikungunya_2014	Chikungunya_2015
0	Austria	1	41	8	Austria	.	.
1	Belgium	1	120	42	Belgium	74	44
2	Bulgaria	.	.	.	Bulgaria	.	.
3	Croatia	.	.	0	Croatia	0	0
4	Cyprus	.	.	.	Cyprus	.	.

5 rows × 33 columns



In [11]:

```
#Vemos que hay columnas repetidas y de las repetidas hay una que no tiene datos en la última fila que corresponde a EU/EEA
```

```
data[['Zika_2015', 'Chikungunya_2014', 'Dengue_2016', 'Country', 'Dengue_2013']]
```

Out[11]:

	Zika_2015	Zika_2015	Chikungunya_2014	Chikungunya_2014	Dengue_2016	Dengue_2016
0	1	1	.	.	116	116
1	1	1	74	74	114	114
2
3	.	.	0	0	2	2
4
5	.	.	3	3	0	0
6
7	.	.	0	0	9	9
8	1	1	4	4	66	66
9	.	.	550	550	297	297
10	.	.	162	162	956	956
11	.	.	1	1	2	2
12	.	.	2	2	24	24
13	0	0
14	1	1	1	1	18	18
15	.	.	39	39	106	106
16	0	0	0	0	9	9
17
18	.	.	0	0	4	4
19	.	.	0	0	1	1
20	.	.	0	0	1	1
21	11	11	33	33	6	6
22	64	64
23	.	.	0	0	41	41
24	13	13
25	.	.	0	0	8	8
26	.	.	0	0	4	4
27	.	.	0	0	6	6
28	10	10	272	272	261	261
29	1	1	19	19	225	225
30	3	3	301	301	468	468
31	NaN	29	1461	1461	NaN	2823



In [13]:

```
#Esto pasa porque como tenemos 4 años atrás tenemos datos duplicados, es decir, en 2017  
tenemo Dengue_2016 pero en 2018 también  
#pues al tratar 4 años atrás en cada fichero tenemos años duplicados.  
#Por este motivo y para evitar columnas repetidas creamos una función que eliminará la  
columna repetida y se quedará con la que  
#tenga el mayor número de datos informados (observar que en Zinka_2015 una columna no t  
iene datos para EU/EEA y otra sí,  
#nos interesa quedarnos con la que tiene dato)  
  
#Creamos función borrado de duplicados: introducimos las dos tablas que queremos unir  
def Delete_duplicated(table1, table2):  
  
    #Para cada columna de la segunda tabla (que es la más pequeña) la comparamos con to  
das las columnas de la tabla1  
    for i2 in table2.columns:  
        for i1 in table1.columns:  
            #si los elementos que comparamos son iguales  
            if i1 == i2:  
                #miramos longitud de filas: eliminamos columna de la tabla que tenga la  
longitud más pequeña  
                if table1.shape[0] < table2.shape[0]:  
                    del table1[table1.columns[table1.columns.get_loc(i1)]]  
  
                elif table2.shape[0] < table1.shape[0]:  
                    del table2[table2.columns[table2.columns.get_loc(i1)]]  
  
                else:  
                    #en caso de ser iguales eliminamos los datos de la primera tabla  
                    del table1[table1.columns[table1.columns.get_loc(i1)]]  
  
    #devolvemos las tablas  
    return table1, table2
```

In [14]:

```

#Volvemos a realizar el proceso pero esta vez llamando a la nueva función creada

# Escribimos un loop que irá a través de cada uno de los nombres de archivo a través de
globbing y el resultado final será la lista dataframes

for i, filename in enumerate(csv_files):
    if not filename.count('desastres') > 0:
        if i == 0:
            data = Changed_Column(pd.read_csv(filename, sep='\t', index_col=0), filename)

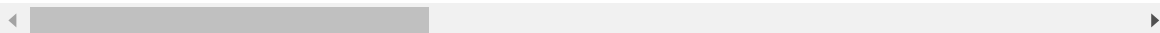
        else:
            df2 = Changed_Column(pd.read_csv(filename, sep='\t', index_col=0), filename)
            data, df2 = Delete_duplicated(data, df2)
            data = pd.concat([data, df2], axis=1)

#Para chequear que todo está bien, mostramos la list_data por consola
data.head()

```

Out[14]:

	Chikungunya_2018	Chikungunya_2013	Chikungunya_2014	Chikungunya_2015	Chikungunya
0	1	0	.	.	
1	3	7	74	44	
2	
3	0	0	0	0	
4	



In [15]:

#Vemos que ahora no tenemos datos duplicados

data[['Zika_2015', 'Chikungunya_2014', 'Dengue_2016', 'Country', 'Dengue_2013']]

Out[15]:

	Zika_2015	Chikungunya_2014	Dengue_2016	Country	Dengue_2013
0	1	.	116	Austria	89
1	1	74	114	Belgium	139
2	.	.	.	Bulgaria	.
3	.	0	2	Croatia	3
4	.	.	.	Cyprus	.
5	.	3	0	Czech Republic	0
6	.	.	.	Denmark	.
7	.	0	9	Estonia	0
8	1	4	66	Finland	80
9	.	550	297	France	271
10	.	162	958	Germany	877
11	.	1	2	Greece	1
12	.	2	24	Hungary	10
13	.	.	0	Iceland	0
14	1	1	18	Ireland	15
15	.	39	106	Italy	142
16	0	0	9	Latvia	7
17	.	.	.	Liechtenstein	.
18	.	0	4	Lithuania	1
19	.	0	1	Luxembourg	0
20	.	0	1	Malta	0
21	11	33	6	Netherlands	.
22	.	.	64	Norway	57
23	.	0	41	Poland	13
24	.	.	13	Portugal	.
25	.	0	8	Romania	6
26	.	0	4	Slovakia	4
27	.	0	6	Slovenia	8
28	10	272	261	Spain	0
29	1	19	225	Sweden	220
30	3	301	468	United Kingdom	571
31	29	1461	2823	EU/EEA	NaN

In [16]:

```
#Reordenamos por nombre las columnas
data.reindex(sorted(data.columns), axis=1).head()

#Ponemos la primera columna Country
country = data['Country']
data.drop(labels=['Country'], axis=1,inplace = True)
data.insert(0, 'Country', country)

#muestra dataset final
data.head()
```

Out[16]:

	Country	Chikungunya_2018	Chikungunya_2013	Chikungunya_2014	Chikungunya_2015	Cl
0	Austria	1	0	.	.	
1	Belgium	3	7	74	44	
2	Bulgaria	
3	Croatia	0	0	0	0	
4	Cyprus	

In [18]:

```
# Lo guardamos en csv

filename = 'enfermedades.csv'
data.to_csv(filename, sep='\t', encoding='utf-8')
print('Guardado csv: ', filename)

print('\n\n Ya puedes analizar diferentes enfermedades\n\n --\t DISFRUTA  ---')
```

Guardado csv: enfermedades.csv

Ya puedes analizar diferentes enfermedades

-- DISFRUTA ---

In []: