Proyecto 1 Data mining

Realizado por:

Nombres

In [1]:
```
!pip install plotly
```

```
Collecting plotly
  Using cached plotly-4.14.3-py2.py3-none-any.whl (13.2 MB)
Requirement already satisfied (use --upgrade to upgrade): plotly from http
s://files.pythonhosted.org/packages/1f/f6/bd3c17c8003b6641df1228e80e1acac97ed
8402635e46c2571f8e1ef63af/plotly-4.14.3-py2.py3-none-any.whl#sha256=d68fc15fc
b49f88db27ab3e0c87110943e65fee02a47f33a8590f541b3042461 in c:\users\angel\ana
conda3\lib\site-packages
Requirement already satisfied: retrying>=1.3.3 in c:\users\angel\anaconda3\li
b\site-packages (from plotly) (1.3.3)
Requirement already satisfied: six in c:\users\angel\anaconda3\lib\site-packa
ges (from plotly) (1.14.0)
```

In [2]:
```
!pip install pyreadstat
```

```
Requirement already satisfied: pyreadstat in c:\users\angel\anaconda3\lib\sit
e-packages (1.1.0)
Requirement already satisfied: pandas>0.24.0 in c:\users\angel\anaconda3\lib
\site-packages (from pyreadstat) (1.0.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\angel\anaconda3\lib\s
ite-packages (from pandas>0.24.0->pyreadstat) (2019.3)
Requirement already satisfied: numpy>=1.13.3 in c:\users\angel\anaconda3\lib
\site-packages (from pandas>0.24.0->pyreadstat) (1.18.1)
Requirement already satisfied: python-dateutil>=2.6.1 in c:\users\angel\anaco
nda3\lib\site-packages (from pandas>0.24.0->pyreadstat) (2.8.1)
Requirement already satisfied: six>=1.5 in c:\users\angel\anaconda3\lib\site-
packages (from python-dateutil>=2.6.1->pandas>0.24.0->pyreadstat) (1.14.0)
```

In [3]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import pyreadstat
import numpy as np
import plotly.express as px
```

In [4]:
```
a = range(2009,2020)
b = "Data/defunciones"
lista = []
nombres = ['Depreg', 'Mupreg', 'Mesreg', 'Añoreg', 'Depocu', 'Mupocu', 'Areag'
,
       'Sexo', 'Diaocu', 'Mesocu', 'Añoocu', 'Edadif', 'Perdif', 'Getdif',
       'Ecidif', 'Ocudif', 'Dnadif', 'Mnadif', 'Nacdif', 'Dredif', 'Mredif',
       'Caudef', 'Asist', 'Ocur', 'Cerdef', 'year']
palabras = ['mupreg', 'mupocu','añoocu', 'Escodif', 'mnadif i in columnas:if',
'Pnadif', 'Predif', 'Puedif', 'Ciuodif',
       'caudef.descrip']
```

In [5]:
```
for k in a:
    path = b + str(k) + ".sav"
    df, meta = pyreadstat.read_sav(path)
    columnas = df.columns
    columnas = list(columnas)
    for i in columnas:
        if i in palabras:
            for j in nombres:

                if i.lower() == j.lower():
                    df = df.rename(columns={i: j})
    df["year"] = k
    lista.append(df)
```

In [6]:
```
defunciones = pd.concat(lista)
defunciones.shape
```

Out[6]: (857750, 33)

In [7]:
```
defunciones.head()
```

Out[7]:

	Depreg	Mupreg	Mesreg	Añoreg	Depocu	Mupocu	Areag	Sexo	Diaocu	Mesocu	...	Ocu
0	5.0	0505	1.0	9.0	5.0	0505	9.0	1.0	2.0	1.0	...	3.(
1	1.0	0101	9.0	9.0	1.0	0101	9.0	1.0	27.0	9.0	...	3.(
2	22.0	2206	9.0	9.0	22.0	2206	9.0	2.0	23.0	8.0	...	3.(
3	2.0	0201	12.0	9.0	2.0	0201	9.0	1.0	5.0	12.0	...	3.(
4	1.0	0101	5.0	9.0	1.0	0101	9.0	2.0	7.0	5.0	...	3.(

5 rows × 33 columns

In [8]:
```
defunciones = defunciones[defunciones.columns[:-7]]
defunciones.head()
```

Out[8]:

	Depreg	Mupreg	Mesreg	Añoreg	Depocu	Mupocu	Areag	Sexo	Diaocu	Mesocu	...	Dna
0	5.0	0505	1.0	9.0	5.0	0505	9.0	1.0	2.0	1.0	...	1
1	1.0	0101	9.0	9.0	1.0	0101	9.0	1.0	27.0	9.0	...	1
2	22.0	2206	9.0	9.0	22.0	2206	9.0	2.0	23.0	8.0	...	1
3	2.0	0201	12.0	9.0	2.0	0201	9.0	1.0	5.0	12.0	...	1
4	1.0	0101	5.0	9.0	1.0	0101	9.0	2.0	7.0	5.0	...	1

5 rows × 26 columns

Observamos que tenemos algunas columnas con data perdida

In [9]:
```
defunciones.isna().any()
```

Out[9]:
```
Depreg    False
Mupreg    False
Mesreg    False
Añoreg    False
Depocu    False
Mupocu    False
Areag      True
Sexo      False
Diaocu    False
Mesocu    False
Añoocu     True
Edadif    False
Perdif    False
Getdif     True
Ecidif    False
Ocudif     True
Dnadif    False
Mnadif     True
Nacdif    False
Dredif    False
Mredif    False
Caudef    False
Asist     False
Ocur      False
Cerdef    False
year      False
dtype: bool
```

Decidí llenar los vacios con 0 Puesto que si la columna es categórica entonces no habría problema porque a la hora de contar no se categorizaría al elemento

In [10]:
```
data = defunciones.fillna(0)
data.head()
```

Out[10]:

	Depreg	Mupreg	Mesreg	Añoreg	Depocu	Mupocu	Areag	Sexo	Diaocu	Mesocu	...	Dna
0	5.0	0505	1.0	9.0	5.0	0505	9.0	1.0	2.0	1.0	...	1
1	1.0	0101	9.0	9.0	1.0	0101	9.0	1.0	27.0	9.0	...	1
2	22.0	2206	9.0	9.0	22.0	2206	9.0	2.0	23.0	8.0	...	1
3	2.0	0201	12.0	9.0	2.0	0201	9.0	1.0	5.0	12.0	...	1
4	1.0	0101	5.0	9.0	1.0	0101	9.0	2.0	7.0	5.0	...	1

5 rows × 26 columns

In [11]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 857750 entries, 0 to 85599
Data columns (total 26 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Depreg  857750 non-null  float64
 1   Mupreg  857750 non-null  object 
 2   Mesreg  857750 non-null  float64
 3   Añoreg  857750 non-null  float64
 4   Depocu  857750 non-null  float64
 5   Mupocu  857750 non-null  object 
 6   Areag   857750 non-null  float64
 7   Sexo    857750 non-null  float64
 8   Diaocu  857750 non-null  float64
 9   Mesocu  857750 non-null  float64
 10  Añoocu  857750 non-null  float64
 11  Edadif  857750 non-null  float64
 12  Perdif  857750 non-null  float64
 13  Getdif  857750 non-null  float64
 14  Ecidif  857750 non-null  float64
 15  Ocudif  857750 non-null  object 
 16  Dnadif  857750 non-null  float64
 17  Mnadif  857750 non-null  object 
 18  Nacdif  857750 non-null  float64
 19  Dredif  857750 non-null  float64
 20  Mredif  857750 non-null  object 
 21  Caudef  857750 non-null  object 
 22  Asist   857750 non-null  float64
 23  Ocur    857750 non-null  float64
 24  Cerdef  857750 non-null  float64
 25  year    857750 non-null  int64  
dtypes: float64(19), int64(1), object(6)
memory usage: 176.7+ MB
```

In [12]:
```
data.describe()
```

Out[12]:

	Depreg	Mesreg	Añoreg	Depocu	Areag	Sex
count	857750.000000	857750.000000	857750.000000	857750.000000	857750.000000	857750.0000(
mean	8.655697	6.458961	1847.015834	8.625140	1.282894	1.4370'
std	6.708185	3.447899	555.147829	6.687205	1.218407	0.4960'
min	1.000000	1.000000	9.000000	1.000000	0.000000	1.0000(
25%	1.000000	3.000000	2011.000000	1.000000	1.000000	1.0000(
50%	9.000000	6.000000	2014.000000	9.000000	1.000000	1.0000(
75%	14.000000	9.000000	2017.000000	14.000000	2.000000	2.0000(
max	22.000000	12.000000	2020.000000	22.000000	9.000000	2.0000(

In [13]:
```
dic = pd.read_excel("Data/diccionario.xlsx",sheet_name="Defunciones",skiprows
= [0])
dic["Valor"].unique()
```

Out[13]:
```
array(['Departamento de registro', nan, 'Municipio de registro',
       'Mes de registro', 'Año de registro', 'Departamento de ocurrencia',
       'Municipio de ocurrencia', 'Área geográfica de ocurrencia',
       'Sexo del difunto(a)', 'Día de ocurrencia', 'Mes de ocurrencia',
       'Edad del difunto(a)', 'Periodo de edad del difunto(a)',
       'Grupo étnico del difunto(a)', 'Estado civil del difunto(a)',
       'Ocupación del difunto(a)',
       'Departamento de nacimiento del difunto(a)',
       'Municipio de nacimiento del difunto(a)',
       'Nacionalidad del difunto(a)',
       'Departamento de residencia del difunto(a)',
       'Municipio de residencia del difunto(a)', 'Causa de defuncion',
       'Asistencia recibida', 'Sitio de ocurrencia', 'Quien certifica'],
      dtype=object)
```

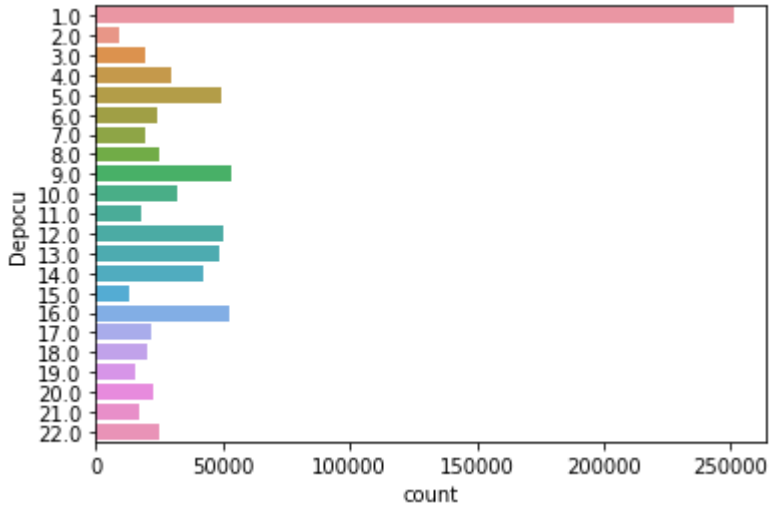
Note que los departamentos que tienen más fallecidos son Guatemala, Quetzaltenango, Altaverapaz y San Marcos

In [14]:
```
a = data["Depocu"].value_counts()
a[:5]
```

Out[14]:
```
1.0     251663
9.0      52964
16.0     52625
12.0     49991
5.0      49089
Name: Depocu, dtype: int64
```

In [15]:
```
sns.countplot(y=data["Depocu"])
```

Out[15]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce96268848>
```



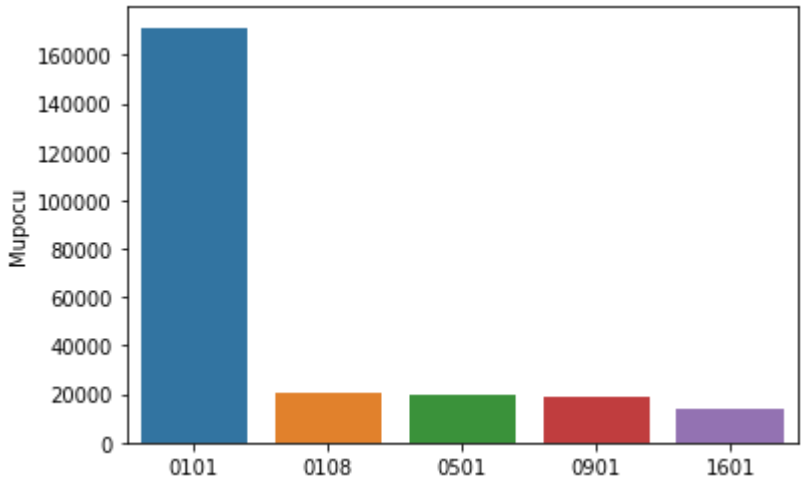
Por otro lado los municipios donde más gente fallece es Guatemala,Quetzaltenango, Escuintla y Mazatenango

In [16]:
```
a = data["Mupocu"].value_counts()
a[:5]
```

Out[16]:
```
0101    171274
0108     20342
0501     19974
0901     19017
1601     13971
Name: Mupocu, dtype: int64
```

In [17]:
```
b = a[:5]
sns.barplot(b.index,b)
```

Out[17]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce97369c08>
```



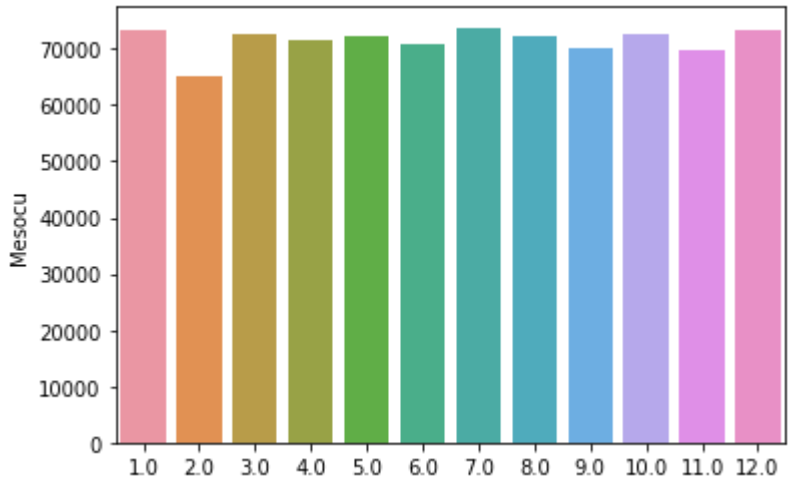
Los meses donde la gente fallecio mas fuel mes de Enero, Julio y Mayo. Aunque no hay mucha diferencia con el resto. Yo no la consideraría una variable para un modelo puesto que apenas hay alguna diferencia.

In [18]:
```
a = data["Mesocu"].value_counts()
a[:5]
```

Out[18]:
```
7.0     73796
12.0    73310
1.0     73209
10.0    72716
3.0     72521
5.0     72443
Name: Mesocu, dtype: int64
```

In [19]:
```
sns.barplot(a.index,a)
```

Out[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce973c9188>
```



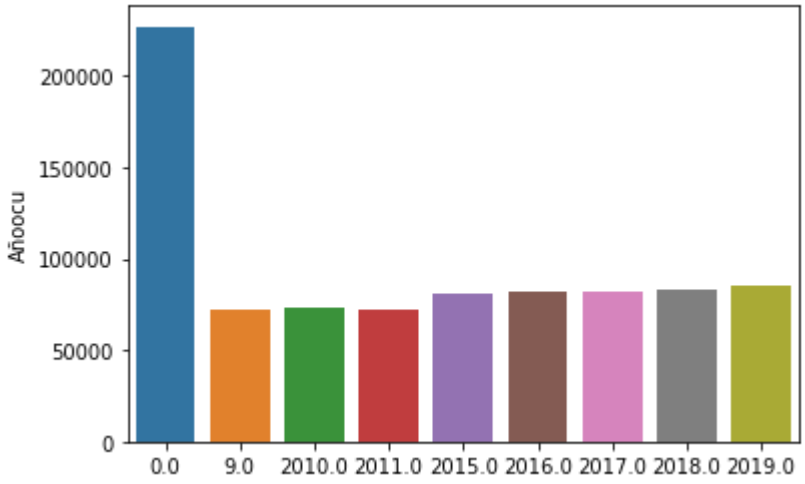
Si ignoramos la categoría donde se llenaron los datos no disponibles vemos que ha ido en aumento el número de defunciones en Guatemala, en este periodo han aumentado más de 10000 muertes

In [20]:
```
a = data["Añoocu"].value_counts()
a
```

Out[20]:
```
0.0       227103
2019.0     85600
2018.0     83071
2016.0     82565
2017.0     81726
2015.0     80876
2010.0     72748
2011.0     72354
9.0        71707
Name: Añoocu, dtype: int64
```

In [21]:
```
sns.barplot(a.index,a)
```

Out[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce97439c08>
```



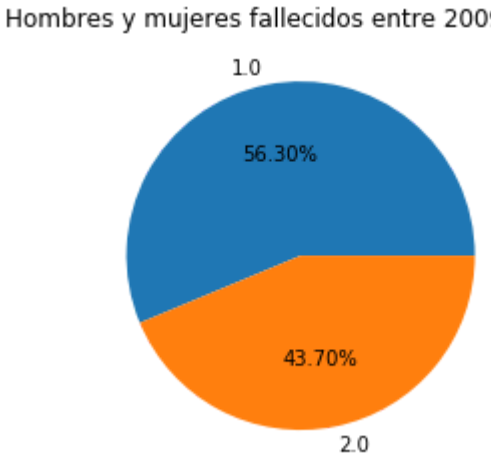
Claramente vemos como es que los hombres murieron más que las mujeres, sin embargo la proporción es bastante similar

In [22]:
```
a = data["Sexo"].value_counts()
a
```

Out[22]:
```
1.0    482899
2.0    374851
Name: Sexo, dtype: int64
```

In [23]:
```
plt.pie(a,labels=a.index,autopct='%1.2f%%')
plt.title("Hombres y mujeres fallecidos entre 2009 a 2019")
```

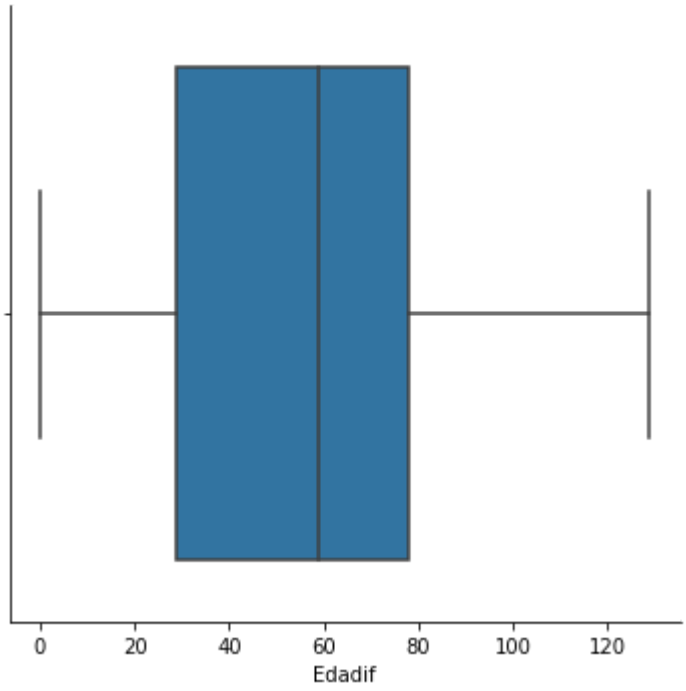
Out[23]:
```
Text(0.5, 1.0, 'Hombres y mujeres fallecidos entre 2009 a 2019')
```



Vemos que ha medida que las personas van aumentando de edad también aumenta la probabilidad de fallecer puesto que el histogram tiene una asimetría hacia la izquierda. Sin embargo hasta la izquierda tenemos un gran número de defunciones que son los niños menores de un año

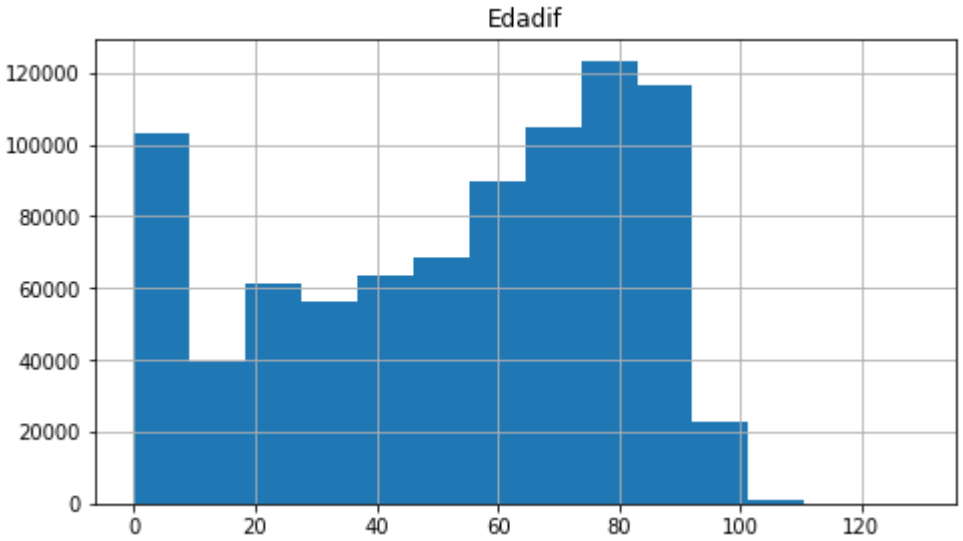
In [24]:
```
sns.catplot(x = "Edadif",
            data = data,
            kind = "box",
            sym = "", # to ommit the points that are so far
            )
data.Edadif.quantile([0.25,0.5,0.75])
```

Out[24]:
```
0.25    29.0
0.50    59.0
0.75    78.0
Name: Edadif, dtype: float64
```



In [25]:
```
from matplotlib import rcParams
rcParams['figure.figsize'] = 7.7,4.27
# Los datos atipicos que removimos aca fueron porque hubo un mal ingreso de da
tos porque el dato era de 999
data = data[data["Edadif"] < 200]
data.hist('Edadif',bins=14)
```

Out[25]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000001CE974F3A88
>]],
      dtype=object)
```



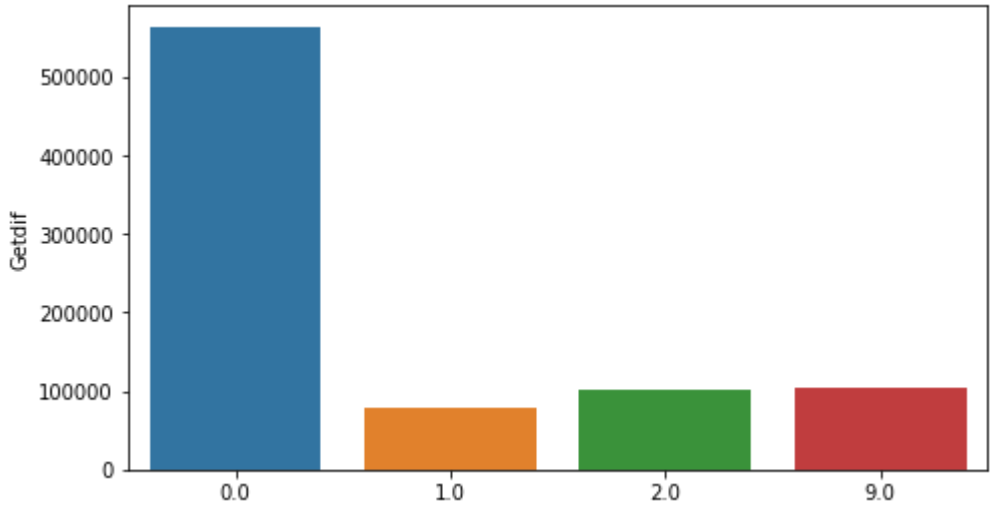
Vemos que las personas que no son del grupo índigena fallecieron más que las personas que si pertenecen a este grupo, pero esta variable no la tomaria en cuenta debido a la poca cantidad de datos

In [26]:
```
a = data["Getdif"].value_counts()
a
```

Out[26]:
```
0.0    563973
9.0    103849
2.0    102304
1.0     79802
Name: Getdif, dtype: int64
```

In [27]:
```
sns.barplot(a.index,a)
```

Out[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce96ab4a48>
```



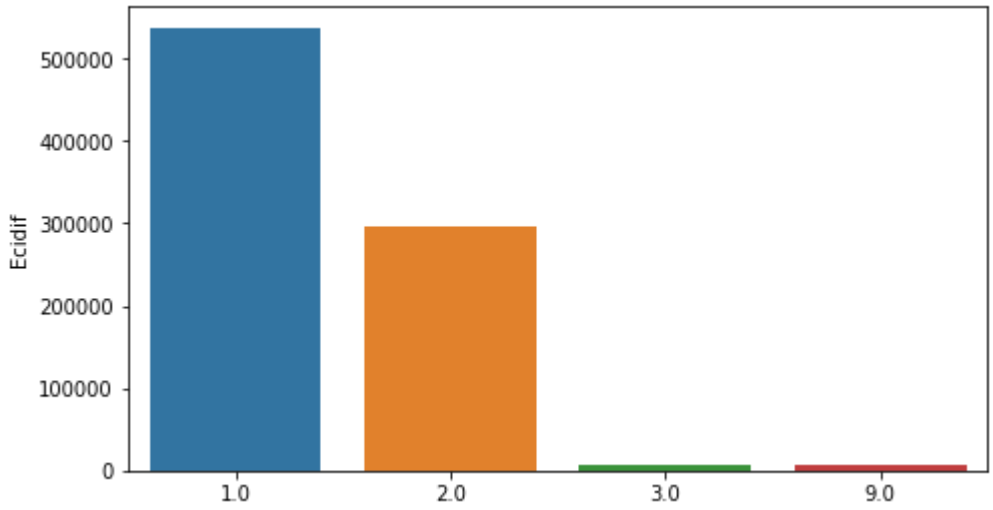
Vemos que las personas que están solteras fueron las más fallecieron a lo largo de este período

In [28]:
```
a = data['Ecidif'].value_counts()
a
```

Out[28]:
```
1.0    537545
2.0    296752
3.0      8022
9.0      7609
Name: Ecidif, dtype: int64
```

In [29]:
```
sns.barplot(a.index,a)
```

Out[29]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce96acd848>
```



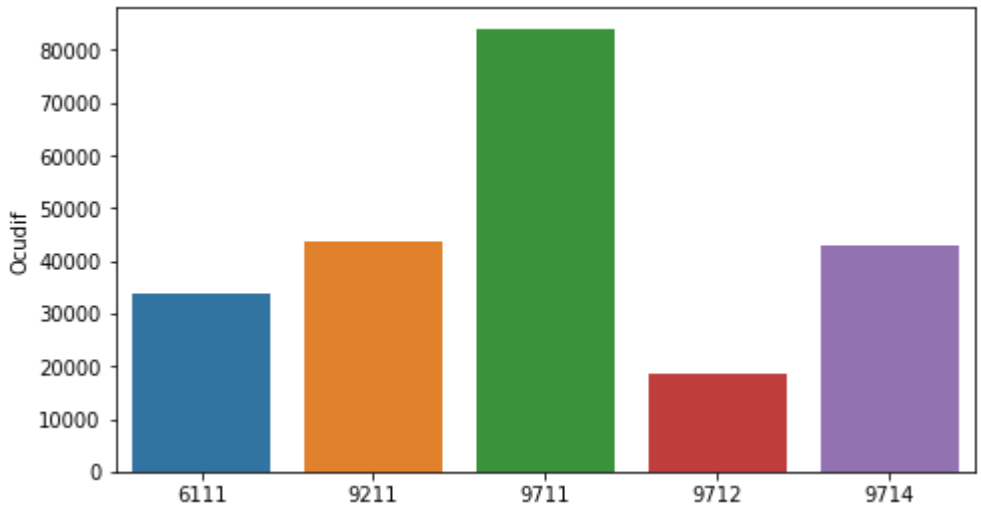
Las ocupaciones de las personas que más fallecieron fueron las de trabajos domésticos no remunerados, peones de explotaciones agrícolas, agricultores y trabajadores calificados de cultivos extensivos, Estudiante

In [30]:
```
a = data['Ocudif'].value_counts()
a[1:6]
```

Out[30]:
```
9711    84080
9211    43616
9714    42799
6111    33872
9712    18390
Name: Ocudif, dtype: int64
```

In [31]:
```
b = a[1:6]
sns.barplot(b.index,b)
```

Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce96ab2848>
```

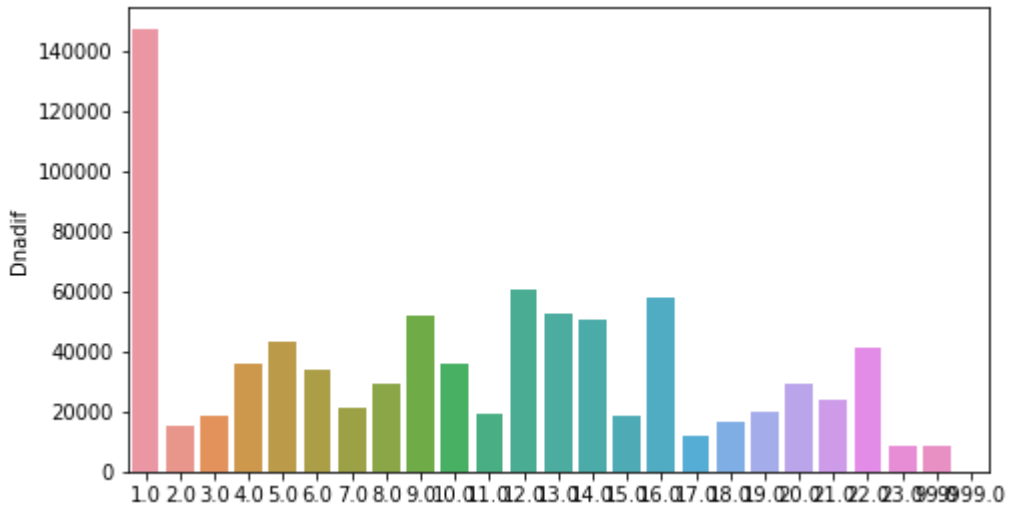


Los departamentos donde nacieron la mayoria de los difuntos son Guatemala, San Marcos, Alta Verapaz y Huehuetenango

In [32]:
```
a = data['Dnadif'].value_counts()
a[:9]
```

Out[32]:
```
1.0     147173
12.0     60714
16.0     58013
13.0     52303
9.0      51606
Name: Dnadif, dtype: int64
```

In [33]:
```
g = sns.barplot(a.index,a)
```

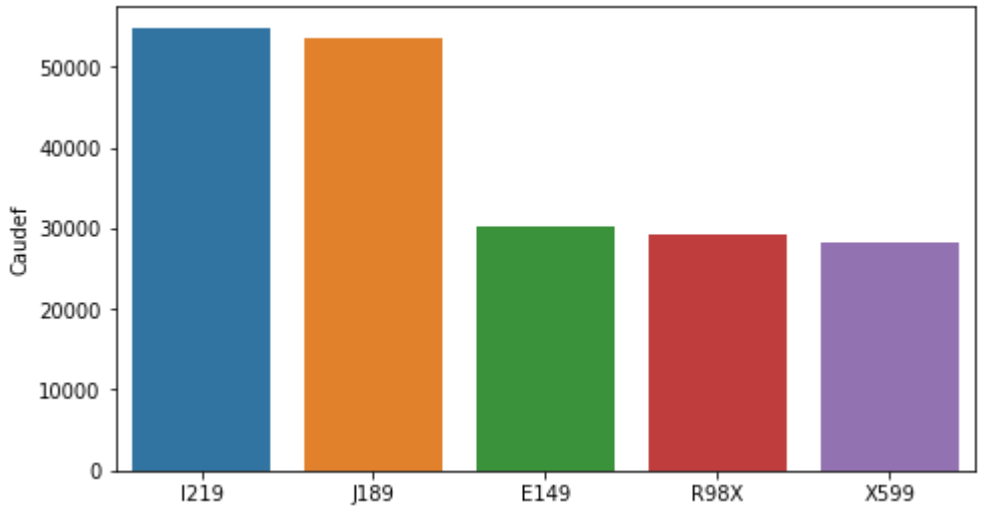


Las causas principales de fallecimiento fueron: Infarto agudo del miocardio, sin otra especificación, Neumonía, no especificada, Diabetes mellitus no especificada, sin mención de complicación, Muerte sin asistencia, Exposición a factores no especificados, causando otras lesiones y las no especificadas.

In [34]:
```
a = data['Caudef'].value_counts()
a[:5]
```

Out[34]:
```
I219    54824
J189    53489
E149    30287
R98X    29133
X599    28292
Name: Caudef, dtype: int64
```

In [35]:
```
b = a[:5]
g = sns.barplot(b.index,b)
```

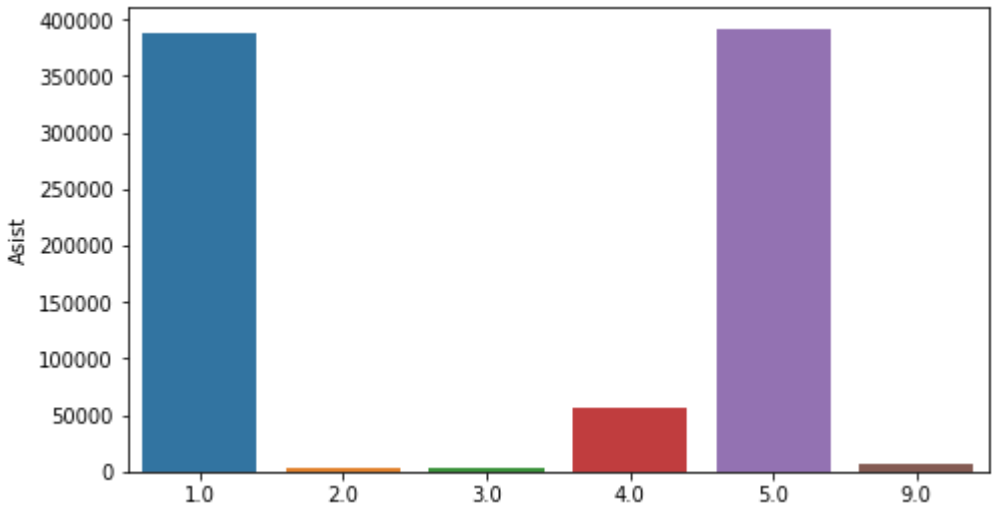


La gran mayoría no recibio ninguna asistencia, aunque su proporción es bastante parecida con los que recibieron atención médica, muy por atrás encontramos la asistencia empírica y los otros tipos de asistencia.

In [36]:
```
a = data['Asist'].value_counts()
a
```

Out[36]:
```
5.0    391831
1.0    388586
4.0     57096
9.0      6600
2.0      3114
3.0      2701
Name: Asist, dtype: int64
```

In [37]:
```
g = sns.barplot(a.index,a)
```



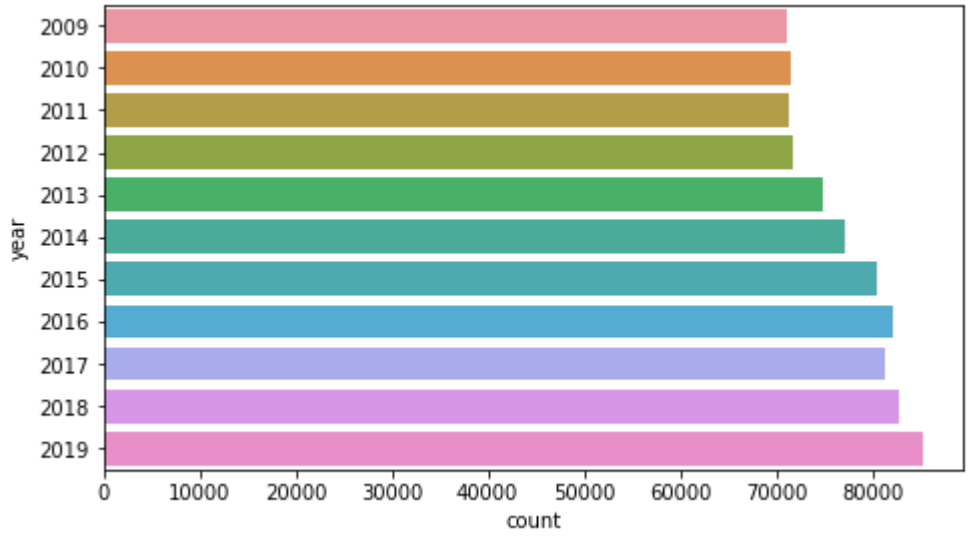
In [38]:
```
data['Asist']
```

Out[38]:
```
0        1.0
1        4.0
2        4.0
3        4.0
4        4.0
        ...
85593    5.0
85594    1.0
85595    1.0
85596    1.0
85597    1.0
Name: Asist, Length: 849928, dtype: float64
```

Podemos observar que a lo largo de los años la cantidad de defunciones ha ido en aumento, podemos observar que en este periodo de 10 años han aumentado las defunciones aproximadamente

In [39]:
```
sns.countplot(y=data["year"])
```

Out[39]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ce967ef7c8>
```



Cruzando variables

En la siguiente tabla cruzamos el departamento donde se registro el difunto y el departamento de residia el difunto, encontramos por ejemplo que son más de 50000 personas las que vivían en el departamento de Guatemala pero sin embargo fallecieron fuera de este departamento

In [40]:
```
pd_crosstab = pd.crosstab(data["Depreg"], data["Dredif"], margins=True)
pd_crosstab
```

Out[40]:

Dredif	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	...	16.0	
Depreg													
1.0	191450	2374	2641	2584	5607	3239	865	609	1696	2157	...	2009	1
2.0	217	8187	2	5	19	7	3	2	6	7	...	42	
3.0	808	12	14403	569	230	54	19	16	21	70	...	11	
4.0	309	4	275	26253	59	18	38	23	34	24	...	8	
5.0	1083	37	128	194	39215	632	61	37	94	1257	...	65	
6.0	761	13	15	22	165	18846	8	15	8	13	...	14	
7.0	106	5	11	47	18	3	16580	59	20	81	...	1	
8.0	96	1	5	6	10	8	57	22372	219	22	...	7	
9.0	309	7	9	25	98	8	305	1457	42135	443	...	13	
10.0	223	4	15	38	218	17	616	27	201	26784	...	6	
11.0	101	3	4	9	55	9	3	13	341	265	...	9	
12.0	292	3	5	9	31	7	8	25	212	23	...	5	
13.0	104	3	4	6	16	3	11	27	101	19	...	3	
14.0	261	2	13	79	57	19	30	114	29	32	...	181	
15.0	83	26	0	0	8	4	0	4	2	0	...	48	
16.0	102	14	2	4	12	3	2	7	5	7	...	46856	
17.0	102	11	8	8	26	20	2	8	9	8	...	180	18
18.0	142	28	7	7	14	8	3	6	7	10	...	71	
19.0	166	359	6	2	9	5	2	2	5	10	...	21	
20.0	146	23	7	4	16	9	0	5	4	6	...	21	
21.0	182	65	2	7	15	48	1	4	1	4	...	8	
22.0	336	10	8	8	77	194	0	12	5	13	...	8	
All	197379	11191	17570	29886	45975	23161	18614	24844	45155	31255	...	49587	21

23 rows × 25 columns

In [41]:
```
# Creamos una columna clasificando a los distintos grupos de edad en categoria
s
criteria = [data['Edadif'].between(0, 1),data['Edadif'].between(1.1, 10),data[
'Edadif'].between(10.1, 20),data['Edadif'].between(21.1, 35),
            data['Edadif'].between(35.1, 50),data['Edadif'].between(51.1, 65),
data['Edadif'].between(65.1, 100)]
values = ["Menos de un año", "Niño", "Adolescente","Joven","Adulto","Adulto 2"
,"Edad avanzada"]

data['Edadrange'] = np.select(criteria, values, 0)
```

Ahora cruzamos los grupos de edad y el sexo donde vemos que las mujeres de edad avanzada fueron las personas que más fallecieron en este período de tiempo seguidas de los hombres.

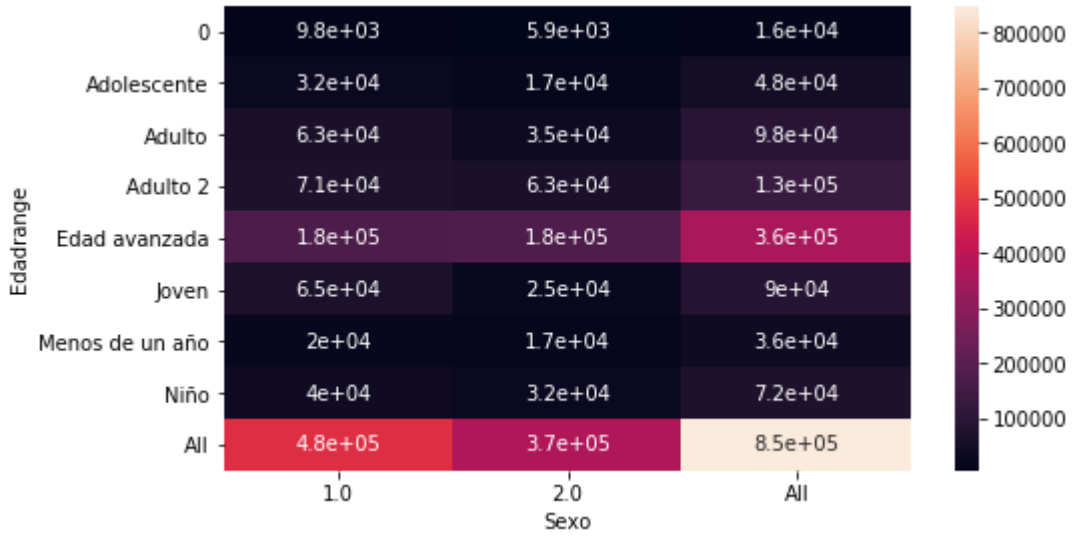
In [42]:
```
pd_crosstab = pd.crosstab(data["Edadrange"], data["Sexo"], margins=True)
pd_crosstab
```

Out[42]:

Sexo / Edadrange	1.0	2.0	All
0	9790	5897	15687
Adolescente	31679	16656	48335
Adulto	63046	34801	97847
Adulto 2	71267	63001	134268
Edad avanzada	177182	178611	355793
Joven	65483	24563	90046
Menos de un año	19878	16570	36448
Niño	39632	31872	71504
All	477957	371971	849928

In [43]:
```
sns.heatmap(pd_crosstab,annot=True)
```

Out[43]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1cef28a00c8>
```



Los hombres solteros fueron las personas que más fallecieron en este período seguidos de las mujeres solteras

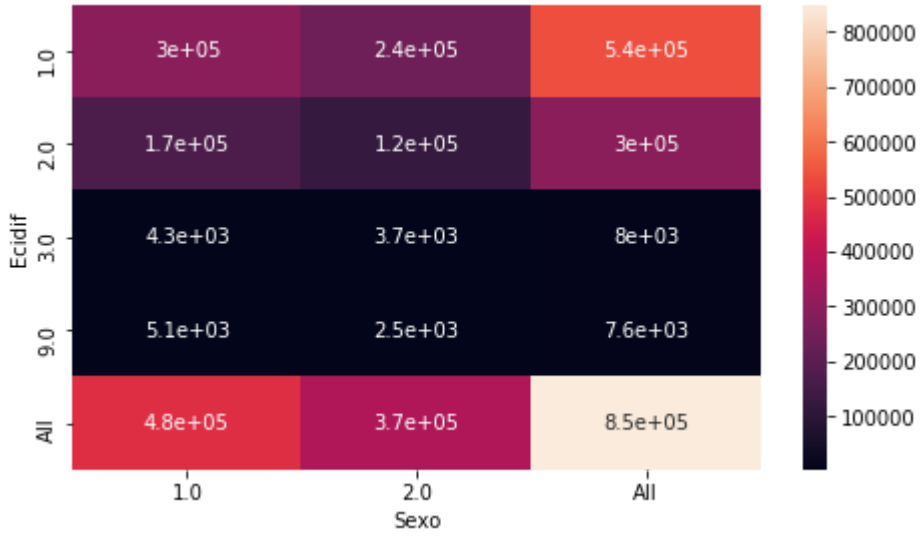
In [44]:
```
pd_crosstab = pd.crosstab(data["Ecidif"], data["Sexo"], margins=True)
pd_crosstab
```

Out[44]:

Sexo	1.0	2.0	All
Ecidif			
1.0	295860	241685	537545
2.0	172696	124056	296752
3.0	4309	3713	8022
9.0	5092	2517	7609
All	477957	371971	849928

In [45]:
```
sns.heatmap(pd_crosstab,annot=True)
```

Out[45]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1ceae453148>
```



En la siguiente tabla comparamos la causa de defunción con el departamento donde sucede y vemos que en Guatemala es donde más gente fallece por ataques al corazón, seguido de Quetzaltenango y Alta Verapaz.

Con la Neumonía tenemos a Alta Verapaz como el departamento donde más gente muere por esto, esto puede ser debido a sus temperaturas frías y sus altas temperaturas, seguido de Guatemala y Totonicapán.

In [46]:
```
pd_crosstab = pd.crosstab(data["Caudef"], data["Dredif"], margins=True)
pd_crosstab.sort_values("All",ascending=False).head(6)
```

Out[46]:

Dredif	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	...	16.0	
Caudef													
All	197379	11191	17570	29886	45975	23161	18614	24844	45155	31255	...	49587	21
I219	13043	1594	1052	1638	3479	1965	655	785	2280	1070	...	2819	1
J189	5450	552	723	2612	1688	871	2375	3009	2139	763	...	6333	
E149	9026	492	776	1118	1964	765	568	971	2033	1188	...	1039	
R98X	1756	64	367	463	2704	1048	546	1021	1100	6525	...	262	
X599	8287	305	752	1066	1667	698	604	558	1436	1252	...	1267	

6 rows × 25 columns

A continuación hacemos una tabla del rango de edad y la causa de muerte, donde vemos que las personas mayores fallecieron a causa de enfermedades del corazón, seguida de la diabetes y muerte sin asistencia.

El segundo grupo más afectado son los adultos que se encuentran entre los 50 y 65 años de edad

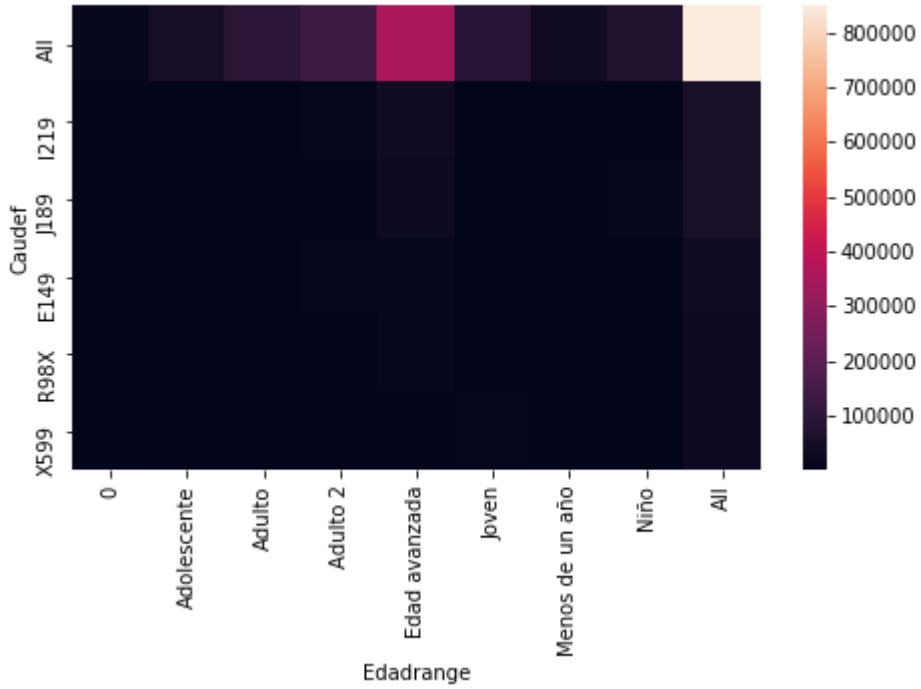
In [47]:
```
pd_crosstab = pd.crosstab(data["Caudef"], data["Edadrange"], margins=True)
pd_crosstab.sort_values("All",ascending=False).head(6)
```

Out[47]:

Edadrange	0	Adolescente	Adulto	Adulto 2	Edad avanzada	Joven	Menos de un año	Niño	All
Caudef									
All	15687	48335	97847	134268	355793	90046	36448	71504	849928
I219	747	371	4160	9111	38625	1794	4	12	54824
J189	494	1369	2554	4741	26607	1997	5465	10262	53489
E149	415	70	3341	9340	16452	655	6	8	30287
R98X	462	912	3069	4894	15555	2084	787	1370	29133
X599	970	3351	5646	3978	4655	8048	282	1362	28292

In [48]:
```
a = pd_crosstab.sort_values("All",ascending=False).head(6)
sns.heatmap(a)
```

Out[48]: <matplotlib.axes._subplots.AxesSubplot at 0x1ce96283048>



Aplicando Kmeans

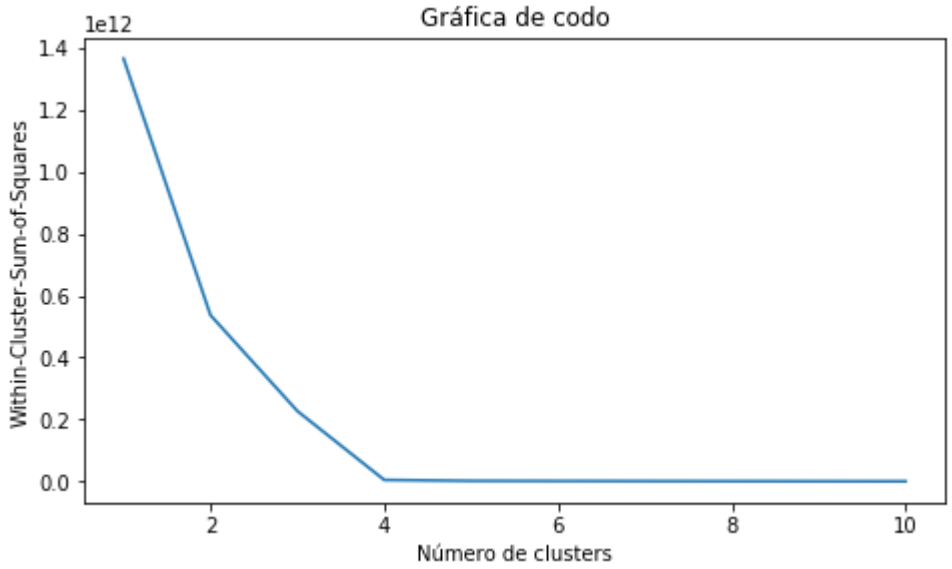
In [49]:
```
from sklearn.cluster import KMeans
data = data.select_dtypes(exclude=['object'])
```

In [50]:
```
lista2 = []
for i in range(1,11):
    kmeans = KMeans(n_clusters = i, max_iter = 300)
    kmeans.fit(data)
    lista2.append(kmeans.inertia_)
```

Al hacer la gráfica de codo vemos que el número optimo de clusters para este conjunto de datos es de 4 puesto que es el que menor error cuadrado tiene y práctiamente tiene la misma precisión con más clusters

In [51]:
```
plt.plot(range(1,11),lista2)
plt.title("Gráfica de codo")
plt.xlabel("Número de clusters")
plt.ylabel("Within-Cluster-Sum-of-Squares")
```

Out[51]:
```
Text(0, 0.5, 'Within-Cluster-Sum-of-Squares')
```



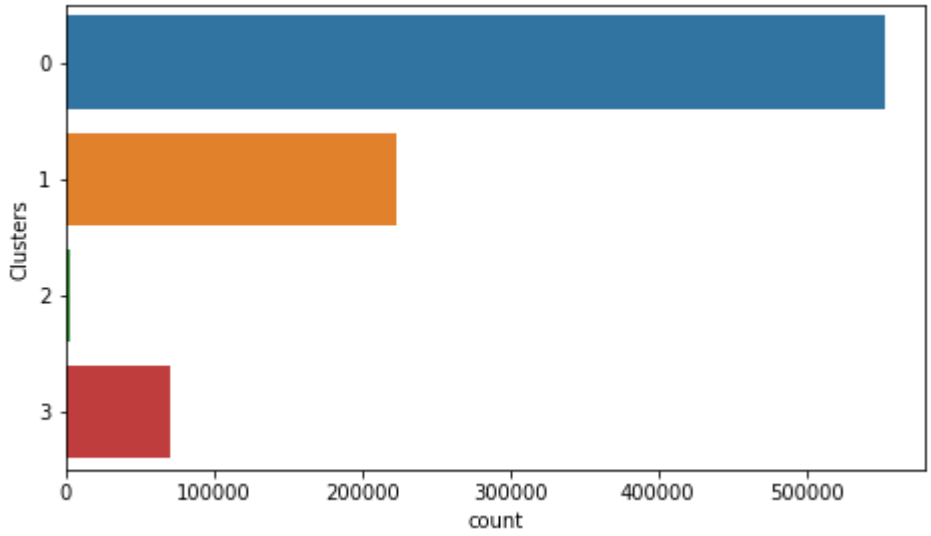
In [52]:
```
# Create a KMeans instance with 3 clusters: model
model = KMeans(n_clusters = 4)

# Fit model to points
model.fit(data)
```

Out[52]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=4, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

Creado ya el modelo creamos una nueva columna para ver a donde pertenece cada individuo

In [53]:
```
data['Clusters'] = model.labels_
sns.countplot(y=data["Clusters"])
```

Out[53]: <matplotlib.axes._subplots.AxesSubplot at 0x1ce96a392c8>



Note que el sexo no es un factor tan importante en los primeros 3 clusters, solamente en el cluster 3 donde la mayoria son hombres

In [54]:
```
pd_crosstab = pd.crosstab(data["Clusters"], data["Sexo"], margins=True,normali
ze='index')
pd_crosstab
```

Out[54]:

Sexo	1.0	2.0
Clusters		
0	0.559277	0.440723
1	0.562266	0.437734
2	0.773534	0.226466
3	0.576605	0.423395
All	0.562350	0.437650

En todos los clusters se mantiene casi la misma proporcion en cuanto al departamento donde fallecieron menos en el cluster 3

In [55]:
```
pd_crosstab = pd.crosstab(data["Clusters"], data["Depocu"], margins=True,norma
lize='index')
pd_crosstab
```

Out[55]:

Depocu	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	
Clusters									
0	0.292326	0.010861	0.022453	0.034946	0.057532	0.028348	0.022594	0.028822	0.061
1	0.291107	0.010598	0.021784	0.035304	0.057093	0.028441	0.023364	0.029330	0.060
2	0.762707	0.006917	0.013233	0.003609	0.029474	0.008722	0.001504	0.000902	0.025
3	0.299848	0.011265	0.020830	0.034229	0.053205	0.025844	0.023161	0.029089	0.062
All	0.294477	0.010810	0.022105	0.034857	0.056945	0.028086	0.022761	0.028868	0.061

5 rows × 22 columns

Notemos que los que pertenecen al cluster 2 y 1 tienen una mayor proporcion de personas con edad avanzada que el cluster 0

In [56]:
```
# Creamos una columna clasificando a los distintos grupos de edad en categoria
s
criteria = [data['Edadif'].between(0, 1),data['Edadif'].between(1.1, 10),data[
'Edadif'].between(10.1, 20),data['Edadif'].between(21.1, 35),
            data['Edadif'].between(35.1, 50),data['Edadif'].between(51.1, 65),
data['Edadif'].between(65.1, 100)]
values = ["Menos de un año", "Niño", "Adolescente","Joven","Adulto","Adulto 2"
,"Edad avanzada"]

data['Edadrange'] = np.select(criteria, values, 0)

pd_crosstab = pd.crosstab(data["Clusters"], data["Edadrange"], margins=True,no
rmalize='index')
pd_crosstab
```

Out[56]:

Edadrange	0	Adolescente	Adulto	Adulto 2	Edad avanzada	Joven	Menos de un año	Niño
Clusters								
0	0.018506	0.055323	0.114666	0.159832	0.425896	0.102074	0.042359	0.081344
1	0.018225	0.057650	0.113625	0.157893	0.418966	0.106941	0.041472	0.085229
2	0.009323	0.084812	0.233383	0.130226	0.125714	0.258346	0.086917	0.071278
3	0.019229	0.065130	0.117844	0.145121	0.374684	0.125765	0.049315	0.102913
All	0.018457	0.056870	0.115124	0.157976	0.418615	0.105945	0.042884	0.084129

In [ ]:
```
X = data2.iloc[:,:].values
from sklearn.metrics import silhouette_score
```

In [ ]:
```
grupos = [KMeans(n_clusters = i, max_iter = 300).fit(X) for i in range (1,11)]
```

In [ ]:
```
scores = [silhouette_score(X, model.labels_)
            for model in kmeans_per_k[1:]]
scores
```