Universidad de San Carlos de Guatemala

Facultad de Ciencias de la Ingeniería

Seminario de Sistemas 2

Ing. Marlon Orellana

Aux. Erick Villatoro

PROYECTO - FASE 02

Análisis EDA en Python

Rodas Sánchez, Carlos Eduardo Leonel - 3350 46142 0901

Quetzaltenango, 20 de diciembre de 2023

DESCRIPCIÓN DE LOS DATOS USADOS

NOMBRE COLUMNA	DESCRIPCIÓN	VALORES POSIBLES
departamento	Indica el nombre del departamento	Valores alfabéticos
codigo_departamento	Indica el codigo del departamento	Valores numéricos
municipio	Indica el nombre del municipio	Valores alfabéticos
codigo_municipio	Indica el codigo del municipio	Valores numéricos
poblacion	Indica la cantidad de poblacion	Valores numéricos
fecha	Indica la fecha de toma de datos	Valores de tipo fecha (aaaa/MM/dd)
numero_casos	Indica el numero de casos positivos	Valores numéricos
Country_code	Indica el codigo del pais	Valores alfanuméricos
Country	Indica el nombre del pais	Valores alfabéticos
WHO_region	Indica el alias del pais	Valores alfabéticos
New_cases	Indica el numero de casos nuevos	Valores numéricos
Cumulative_cases	Indica el numero de casos acumulados	Valores numéricos
New_deaths	Indica el numero de muertes nuevas	Valores numéricos
Cumulative_deaths	Indica el numero de muertes acumuladas	Valores numéricos

- Utilizando el método “dataframe.shape” nos brinda lo siguiente:

  Despliega el número de variables y número de observaciones

- Utilizando el método “dataframe.dtypes” nos brinda lo siguiente:

  Despliega el tipo de dato de cada una de las variables. El subconjunto del dataframe formado por las columnas de los tipos especificados.

---------DESCRIPCION DE LOS DATOS---------

```
[4] df.shape  # Despliega el numero de variables y numero de observaciones
```

```
(5070, 15)
```

```
df.dtypes  #Instruccion Despliega el tipo de dato de cada una de las variables
```

```
departamento           object
codigo_departamento     int64
municipio              object
codigo_municipio        int64
poblacion               int64
fecha                  object
numero_casos            int64
Date_reported          object
Country_code           object
Country                object
WHO_region             object
New_cases               int64
Cumulative_cases        int64
New_deaths              int64
Cumulative_deaths       int64
dtype: object
```

- Utilizando el método “dataframe.count()” nos brinda lo siguiente:

  Cuenta los datos no nulos en cada columna del Dataset

- Utilizando el método “dataframe.isna().sum()” nos brinda lo siguiente:

  Determina los valores nulos por cada columna del dataframe.

- Utilizando el método “dataframe.isna().sum().sum()” nos brinda lo siguiente:

  Determina los valores nulos de todo el dataframe.

```
df.count()  #Cuenta los datos no nulos en cada columna del Dataset
```

```
departamento           5070
codigo_departamento    5070
municipio              5070
codigo_municipio       5070
poblacion              5070
fecha                  5070
numero_casos           5070
Date_reported          5070
Country_code           5070
Country                5070
WHO_region             5070
New_cases              5070
Cumulative_cases       5070
New_deaths             5070
Cumulative_deaths      5070
dtype: int64
```

```
[7] df.isna().sum().sum() #Determinacion de valores nulos en todo el dataset
```

```
0
```

```
df.isna().sum() #Determinacion de valores nulos por columna
```

```
departamento           0
codigo_departamento    0
municipio              0
codigo_municipio       0
poblacion              0
fecha                  0
numero_casos           0
Date_reported          0
Country_code           0
Country                0
WHO_region             0
New_cases              0
Cumulative_cases       0
New_deaths             0
```

Estadística descriptiva de los datos - EDA

- Utilizando el método “dataframe.describe(include=”all”)” nos brinda los siguientes valores:

  - ➔ Count: Proporciona el número total de datos descrita por cada columna del dataframe.
  - ➔ Unique: Indica la cantidad de datos posibles para cada columna del dataframe.
  - ➔ Top: Indica el valor mas utilizado de cada columna del dataframe.
  - ➔ Freq: Indica la frecuencia del valor utilizado de cada columna del dataframe.
  - ➔ Mean: Calcula la media de valores para cada columna del dataFrame.
  - ➔ Std: Calcula la desviación estándar de cada columna numérica del dataframe.
  - ➔ Min: Obtiene el mínimo de los valores de cada columna numérica del dataframe.
  - ➔ 25%: Indica el valor del percentil del 25% por cada columna numérica del dataframe.
  - ➔ 50%: Indica el valor del percentil del 50% por cada columna numérica del dataframe.
  - ➔ 75%: Indica el valor del percentil del 75% por cada columna numérica del dataframe.
  - ➔ Max: Obtiene el máximo de los valores de cada columna numérica del dataframe.

```
df.describe()
```

	codigo_departamento	codigo_municipio	poblacion	numero_casos	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
count	5070.000000	5070.000000	5.070000e+03	5070.000000	5070.000000	5070.000000	5070.000000	5070.000000
mean	40.671598	1118.653846	4.982360e+04	0.043393	651.400000	141515.133333	23.200000	4948.000000
std	542.549500	585.766997	8.139402e+04	0.329533	365.865697	2949.721297	23.300939	115.288428
min	1.000000	0.000000	0.000000e+00	0.000000	80.000000	138012.000000	0.000000	4813.000000
25%	6.000000	613.000000	1.739300e+04	0.000000	181.000000	138475.000000	6.000000	4833.000000
50%	12.000000	1204.500000	3.086200e+04	0.000000	783.000000	141074.000000	24.000000	4928.000000
75%	15.000000	1506.000000	5.852600e+04	0.000000	982.000000	143976.000000	34.000000	5025.000000
max	9999.000000	2217.000000	1.205668e+06	7.000000	1063.000000	146937.000000	92.000000	5151.000000

```
df.describe(include='all')
```

	departamento	codigo_departamento	municipio	codigo_municipio	poblacion	fecha	numero_casos	Date_reported	Country_code
count	5070	5070.000000	5070	5070.000000	5.070000e+03	5070	5070.000000	5070	5070
unique	23	NaN	333	NaN	NaN	15	NaN	15	1
top	HUEHUETENANGO	NaN	SAN PEDRO SACATEPEQUEZ	NaN	NaN	2021-01-01	NaN	2021-01-01	GT
freq	495	NaN	30	NaN	NaN	338	NaN	338	5070
mean	NaN	40.671598	NaN	1118.653846	4.982360e+04	NaN	0.043393	NaN	NaN
std	NaN	542.549500	NaN	585.766997	8.139402e+04	NaN	0.329533	NaN	NaN
min	NaN	1.000000	NaN	0.000000	0.000000e+00	NaN	0.000000	NaN	NaN
25%	NaN	6.000000	NaN	613.000000	1.739300e+04	NaN	0.000000	NaN	NaN
50%	NaN	12.000000	NaN	1204.500000	3.086200e+04	NaN	0.000000	NaN	NaN
75%	NaN	15.000000	NaN	1506.000000	5.852600e+04	NaN	0.000000	NaN	NaN
max	NaN	9999.000000	NaN	2217.000000	1.205668e+06	NaN	7.000000	NaN	NaN

	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
count	5070	5070	5070.000000	5070.000000	5070.000000	5070.000000
unique	1	1	NaN	NaN	NaN	NaN
top	Guatemala	AMRO	NaN	NaN	NaN	NaN
freq	5070	5070	NaN	NaN	NaN	NaN
mean	NaN	NaN	651.400000	141515.133333	23.200000	4948.000000
std	NaN	NaN	365.865697	2949.721297	23.300939	115.288428
min	NaN	NaN	80.000000	138012.000000	0.000000	4813.000000
25%	NaN	NaN	181.000000	138475.000000	6.000000	4833.000000
50%	NaN	NaN	783.000000	141074.000000	24.000000	4928.000000
75%	NaN	NaN	982.000000	143976.000000	34.000000	5025.000000
max	NaN	NaN	1063.000000	146937.000000	92.000000	5151.000000

- Utilizando el método “dataframe.info()” nos brinda lo siguiente:

  Muestra toda la info del DataSet

```
df.info()  #Muestra toda la info del DataSet
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5070 entries, 0 to 5069
Data columns (total 15 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   departamento         5070 non-null   object
 1   codigo_departamento  5070 non-null   int64 
 2   municipio            5070 non-null   object
 3   codigo_municipio     5070 non-null   int64 
 4   poblacion            5070 non-null   int64 
 5   fecha                5070 non-null   object
 6   numero_casos         5070 non-null   int64 
 7   Date_reported        5070 non-null   object
 8   Country_code         5070 non-null   object
 9   Country              5070 non-null   object
 10  WHO_region           5070 non-null   object
 11  New_cases            5070 non-null   int64 
 12  Cumulative_cases     5070 non-null   int64 
 13  New_deaths           5070 non-null   int64 
 14  Cumulative_deaths    5070 non-null   int64 
dtypes: int64(8), object(7)
memory usage: 594.3+ KB
```

- Utilizando el método “dataframe.duplicated().” nos brinda lo siguiente:

  Manejo o búsqueda de valores duplicados. En este archivo no existe ningún valor duplicado.

```
df.duplicated()   #Manejo o busqueda de valores duplicados
```

```
0       False
1       False
2       False
3       False
4       False
        ...
5065    False
5066    False
5067    False
5068    False
5069    False
Length: 5070, dtype: bool
```

EDA - MONOVARIABLE - DATOS CUANTITATIVOS

Manejo de Outliers

Un outlier es un elemento/objeto de datos que se desvía significativamente del resto de los objetos (llamados normales). Pueden deberse a errores de medición o de ejecución. El análisis para la detección de valores atípicos se conoce como minería de valores atípicos.

- Utilizando el método “dataframe.quantile.” nos brinda lo siguiente:

  Es una función que retorna el valor del cuantil dado sobre el eje solicitado.

- Utilizando el método “dataframe.clip().” nos brinda lo siguiente:

  Se utiliza para recortar valores en un umbral de entrada especificado. Podemos usar esta función para poner un límite inferior y un límite superior a los valores que puede tener cualquier celda en el marco de datos.

- Utilizando el método “dataframe.plot().” nos brinda lo siguiente:

  Es un método proporciona un conjunto de estilos de trama a través del argumento de palabra clave amable para crear tramas de apariencia decente.

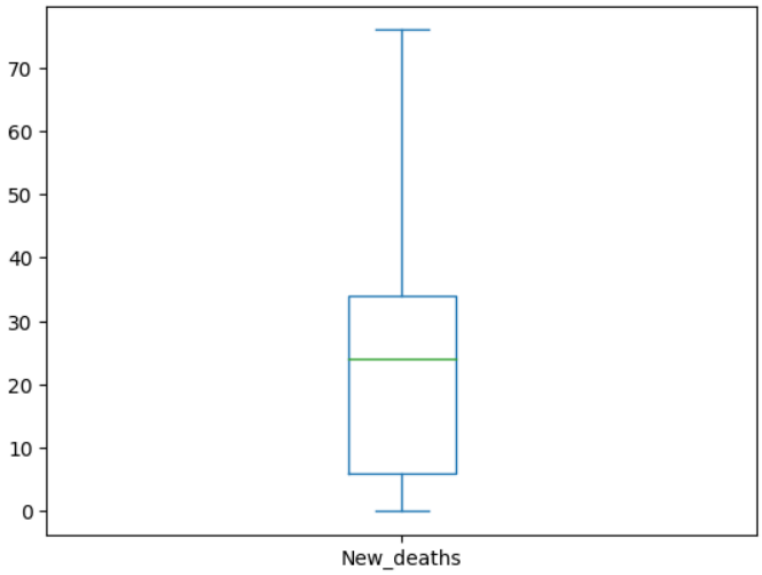
➔ **Outlier de la columna “NEW DEATHS”**

```
---EDA - MONOVARIABLE----------

---------DATOS CUANTITATIVOS---------

------VARIABLE DE ESTUDIO: "CANTIDAD DE NUEVAS MUERTES"------
```

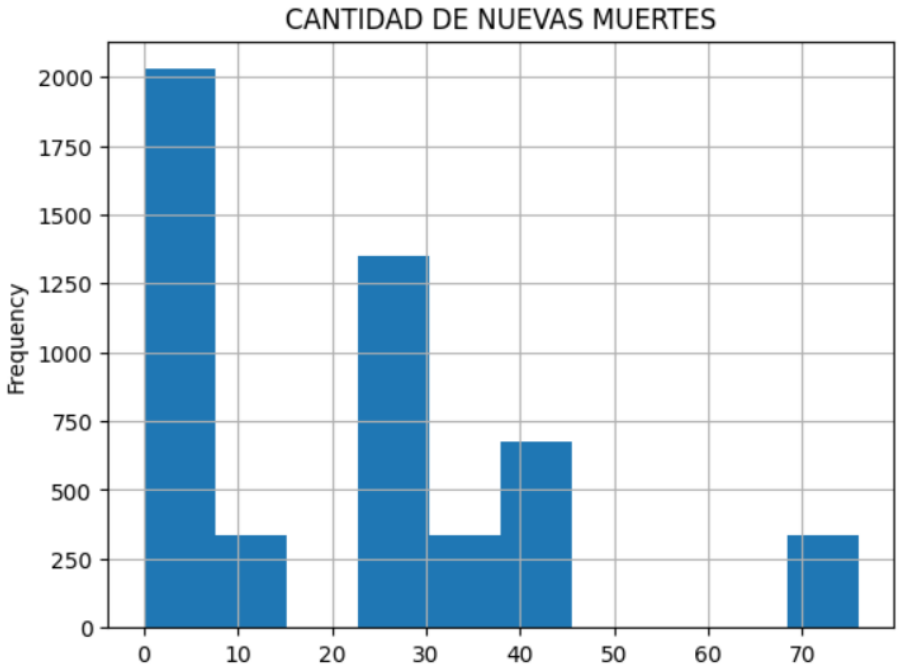
```
[14] #Calculo de percentiles
     p0=df.New_deaths.min()
     p100=df.New_deaths.max()
     q1=df.New_deaths.quantile(0.25)
     q2=df.New_deaths.quantile(0.5)
     q3=df.New_deaths.quantile(0.75)
     iqr=q3-q1
     #Calculo lc y uc
     lc= q1-1.5*iqr
     uc=q3 +1.5*iqr
     print( "p0 = " , p0 ,", p100 = " , p100 ,", q1 = " , q1,", q2 = " , q2,", q3 = " ,

     p0 =  0 , p100 =  92 , q1 =  6.0 , q2 =  24.0 , q3 =  34.0 , iqr =  28.0 , lc =  -36.0 , uc =  76.0
```

```
15] df.New_deaths.clip(upper=uc,inplace=True)   # Funcion para normalizar Outliers.
    df.New_deaths.plot(kind='box') #Outlier Nuevas muertes
```

```
<Axes: >
```




```
df.New_deaths.plot(kind='hist',grid=True)
plt.title("CANTIDAD DE NUEVAS MUERTES")
plt.show()
```



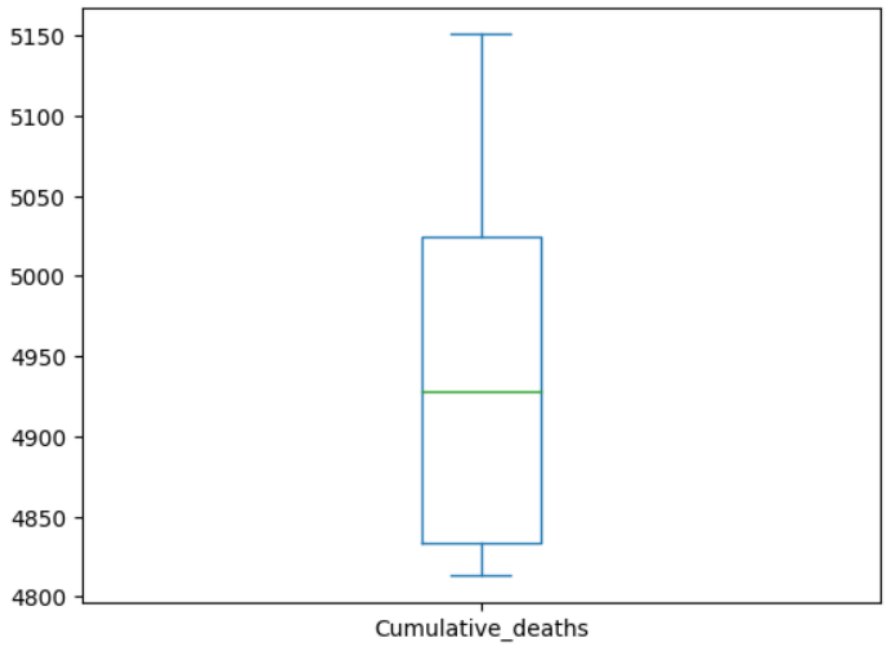
➔ **Outlier de la columna “CUMULATIVE DEATHS”**

```
#Calculo de percentiles
p0=df.Cumulative_deaths.min()
p100=df.Cumulative_deaths.max()
q1=df.Cumulative_deaths.quantile(0.25)
q2=df.Cumulative_deaths.quantile(0.5)
q3=df.Cumulative_deaths.quantile(0.75)
iqr=q3-q1
#Calculo lc y uc
lc= q1-1.5*iqr
uc=q3 +1.5*iqr
print( "p0 = " , p0 ,", p100 = " , p100 ,", q1 = " , q1,", q2 = " , q2,", q3 = " , q3 ,", iqr = "
```

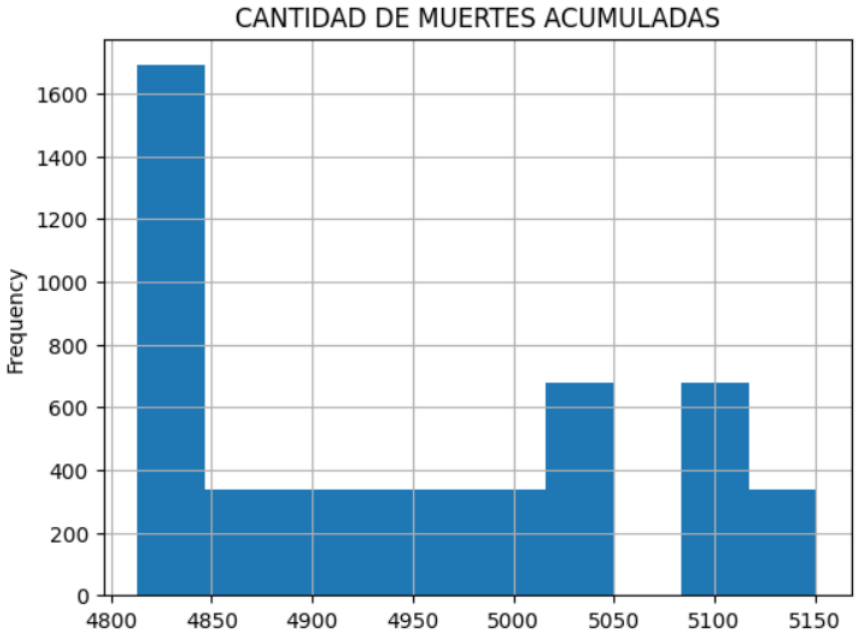
```
p0 =  4813 , p100 =  5151 , q1 =  4833.0 , q2 =  4928.0 , q3 =  5025.0 , iqr =  192.0 , lc =  4545.0 , uc =  5313.0
```

```
df.Cumulative_deaths.clip(upper=uc,inplace=True)   # Funcion para normalizar Outliers.
df.Cumulative_deaths.plot(kind='box') #Outlier Muertes acumuladas
```

```
<Axes: >
```



```
df.Cumulative_deaths.plot(kind='hist',grid=True)
plt.title("CANTIDAD DE MUERTES ACUMULADAS")
plt.show()
```



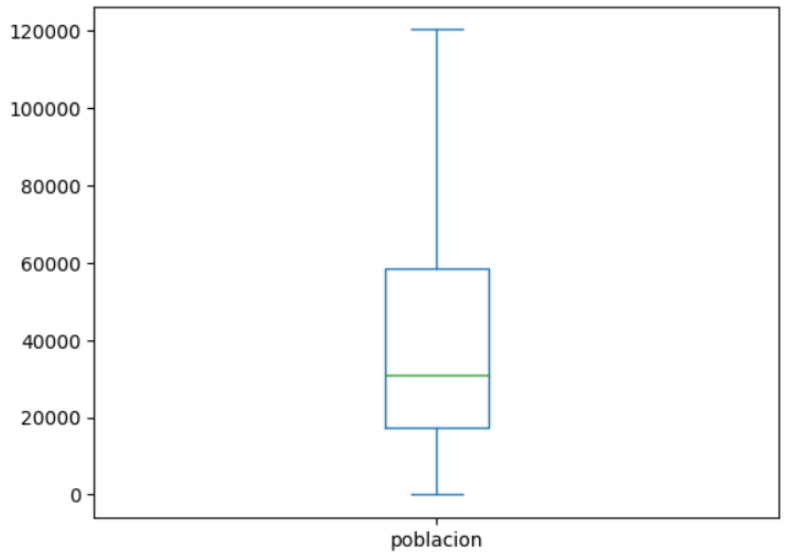
➔ **Outlier de la columna “POBLACION”**

```
#Calculo de percentiles
p0=df.poblacion.min()
p100=df.poblacion.max()
q1=df.poblacion.quantile(0.25)
q2=df.poblacion.quantile(0.5)
q3=df.poblacion.quantile(0.75)
iqr=q3-q1
#Calculo lc y uc
lc= q1-1.5*iqr
uc=q3 +1.5*iqr
print( "p0 = " , p0 ,", p100 = " , p100 ,", q1 = " , q1,", q2 = " , q2,", q3 = " , q3 ,", iqr = " , iqr
```

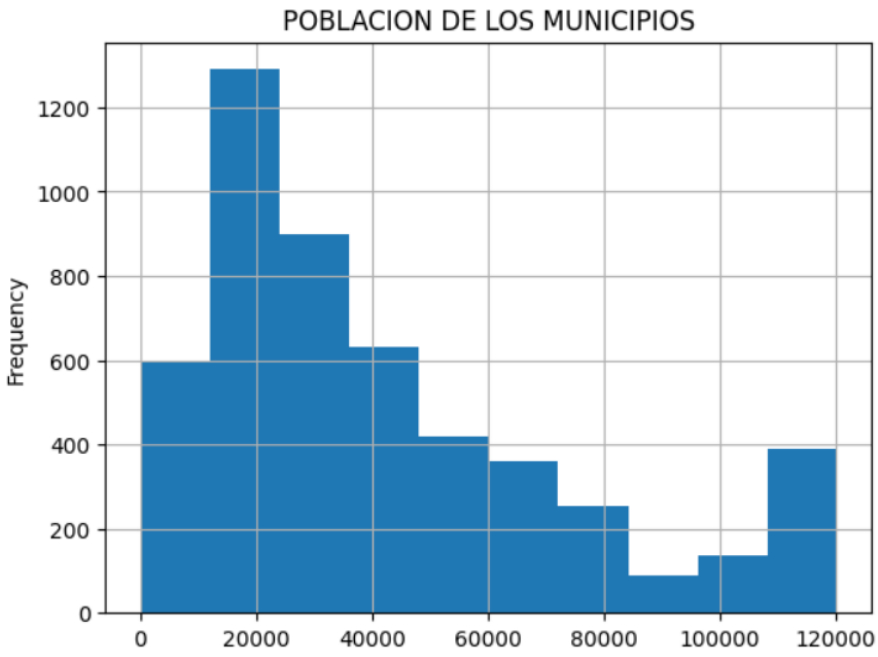
```
p0 =  0 , p100 =  1205668 , q1 =  17393.0 , q2 =  30862.0 , q3 =  58526.0 , iqr =  41133.0 , lc =  -44306.5 , uc =  120225.5
```

```
df.poblacion.clip(upper=uc,inplace=True)   # Funcion para normalizar Outliers.
df.poblacion.plot(kind='box') #Outlier Poblacion de municipios
```

```
<Axes: >
```



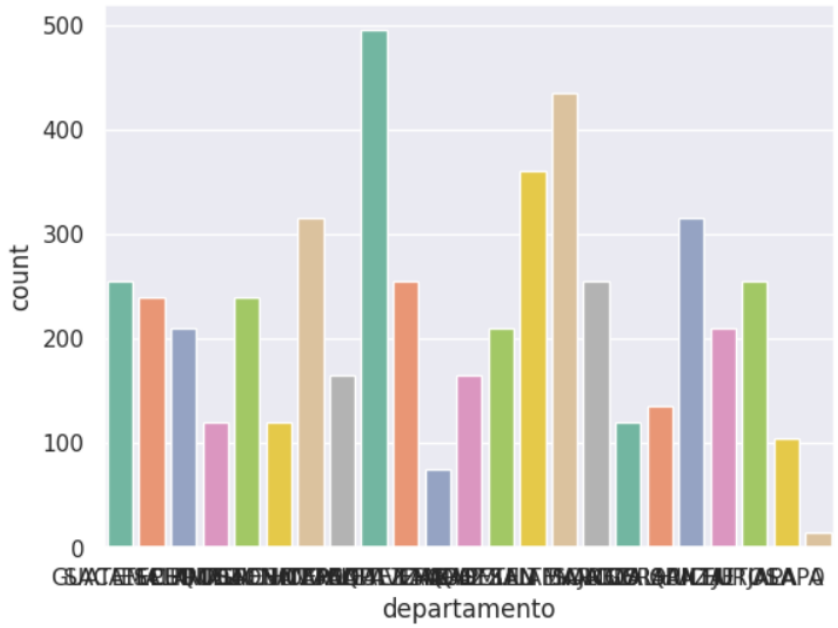

```
df.poblacion.plot(kind='hist',grid=True)
plt.title("POBLACION DE LOS MUNICIPIOS")
plt.show()
```

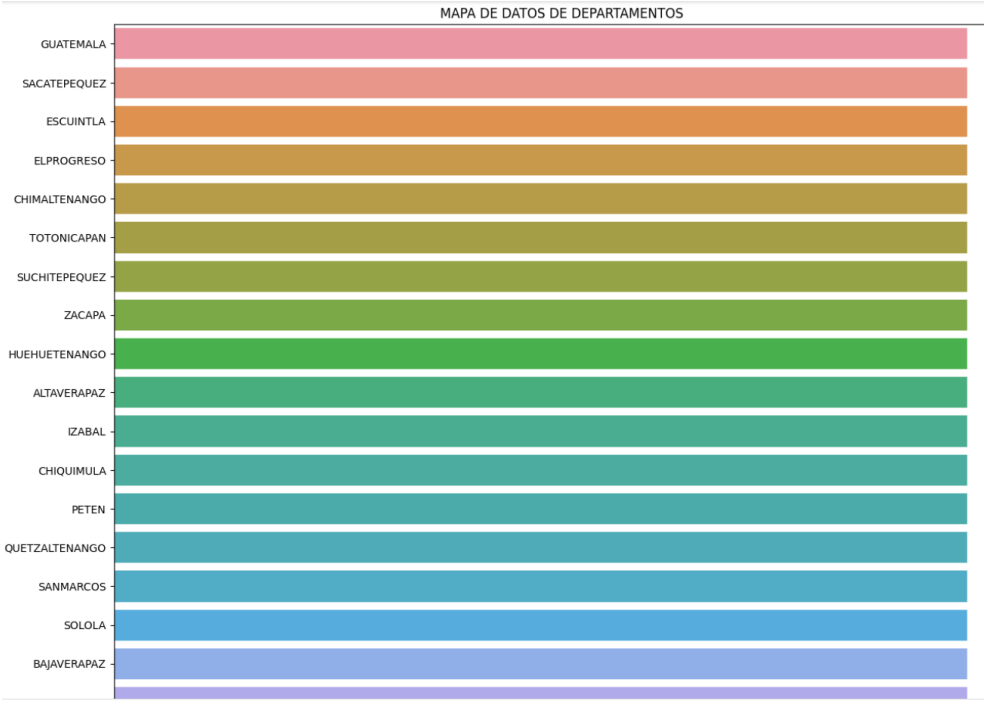


EDA - MONOVARIABLE - DATOS CUALITATIVOS

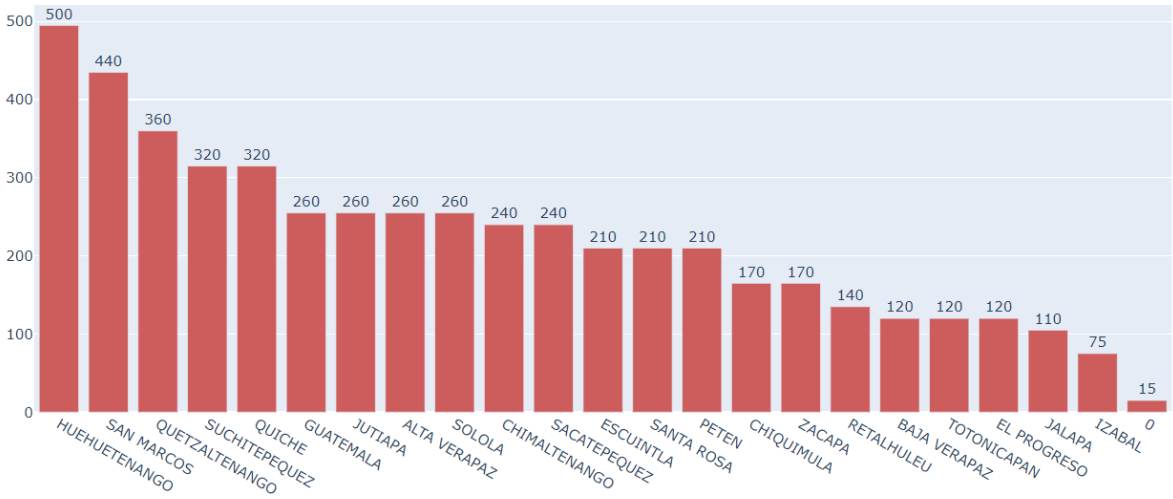
➔ **Conteo de registros de la variable “DEPARTAMENTO”**

```
sns.set(style="darkgrid")
ax = sns.countplot(x="departamento", data=df, palette="Set2")
```

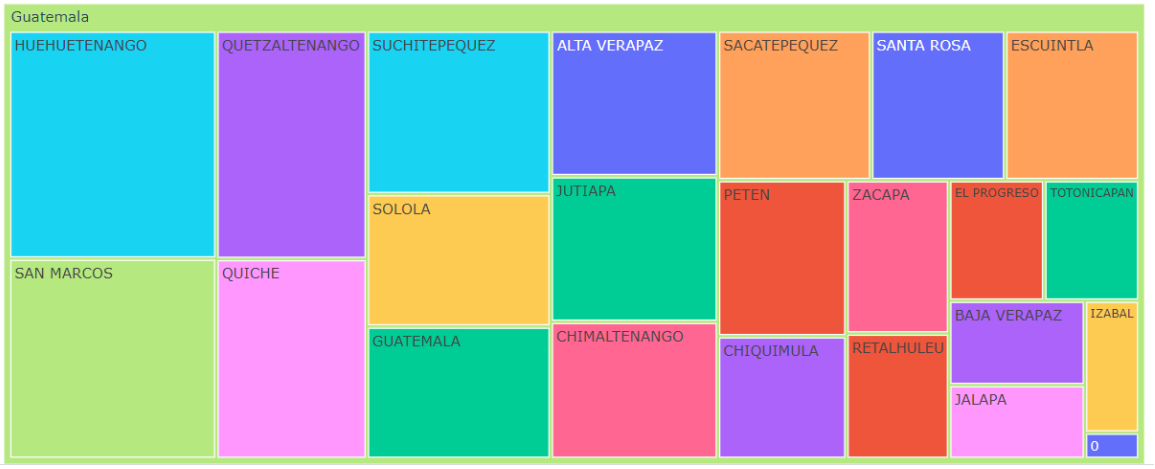




```
import plotly.graph_objects as go
topdirs=pd.value_counts(df['departamento'])
fig = go.Figure([go.Bar(x=topdirs.index, y=topdirs.values , text=topdirs.values,marker_color='indianred')])
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.show()
```

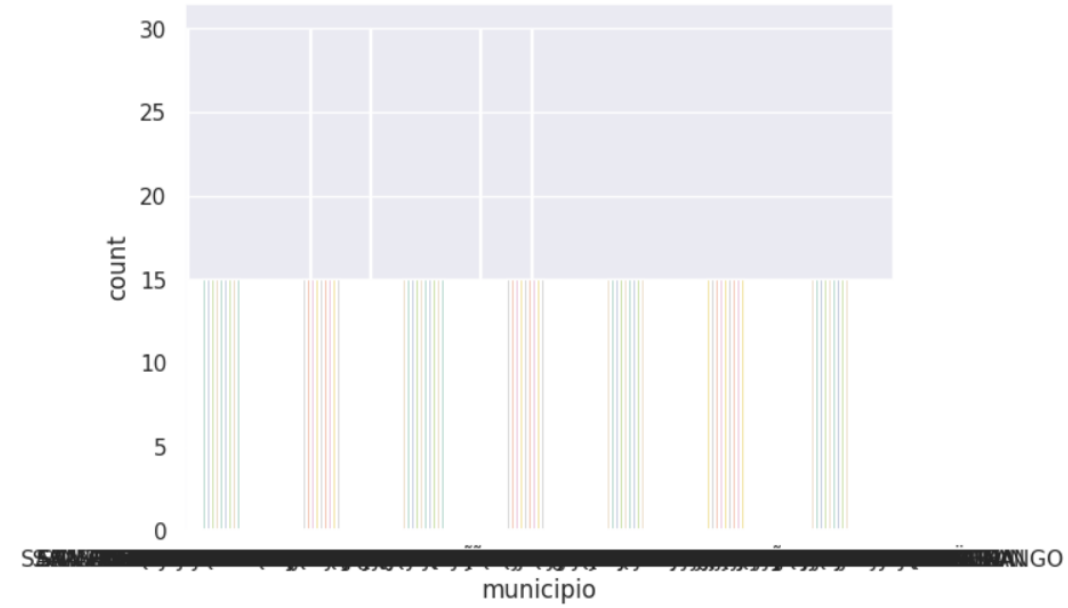


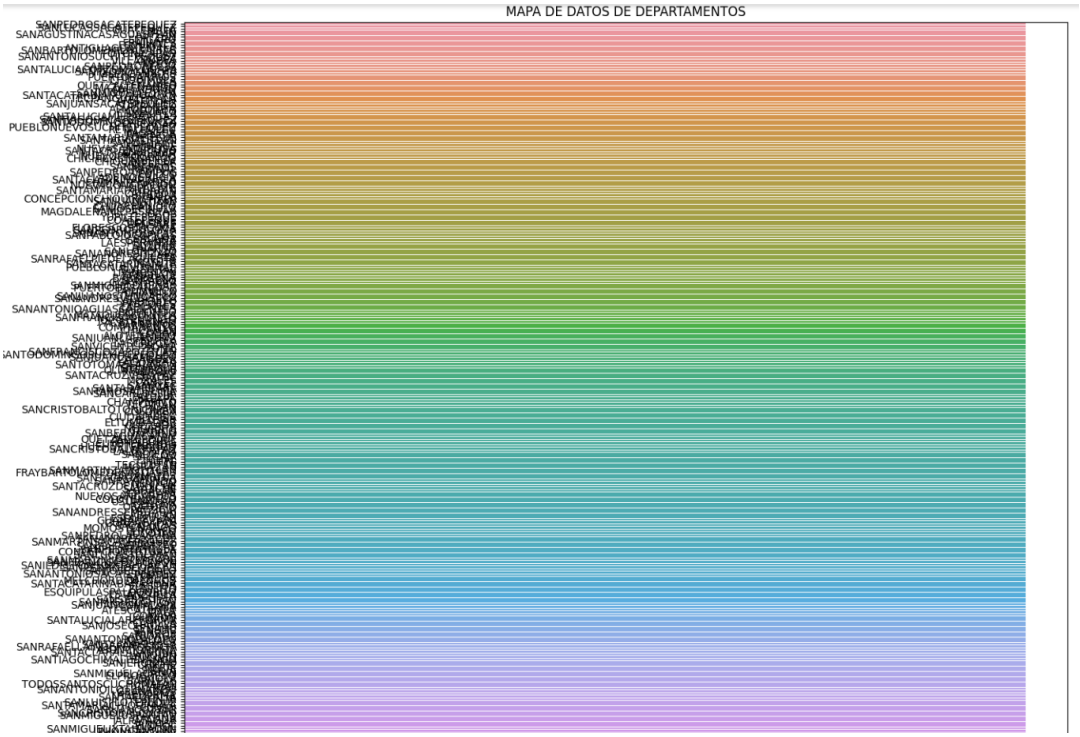

```
netflix_fr=df[df['Country']=='Guatemala']
nannef=netflix_fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['Country','departamento'],
                 color='departamento', hover_data=['departamento'],color_continuous_scale='Purples')
fig.show()
```



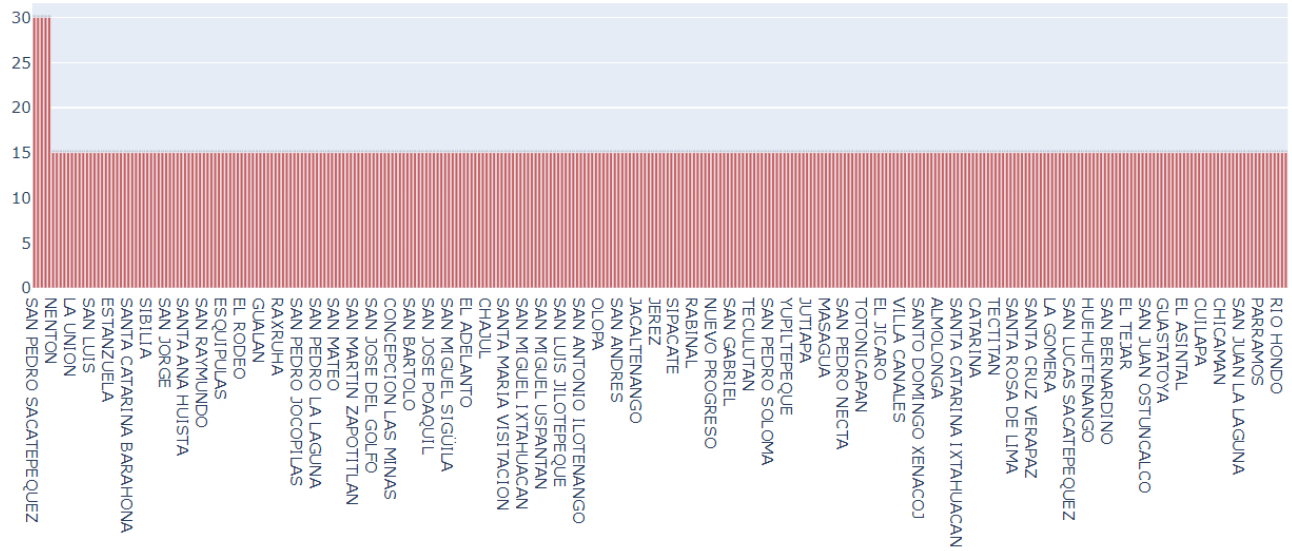
➔ **Conteo de registros de la variable “MUNICIPIO”**

```
sns.set(style="darkgrid")
ax = sns.countplot(x="municipio", data=df, palette="Set2")
```





```
import plotly.graph_objects as go
top=pd.value_counts(df['municipio'])
fig = go.Figure([go.Bar(x=top.index, y=top.values , text=top.values,marker_color='indianred')])
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.show()
```

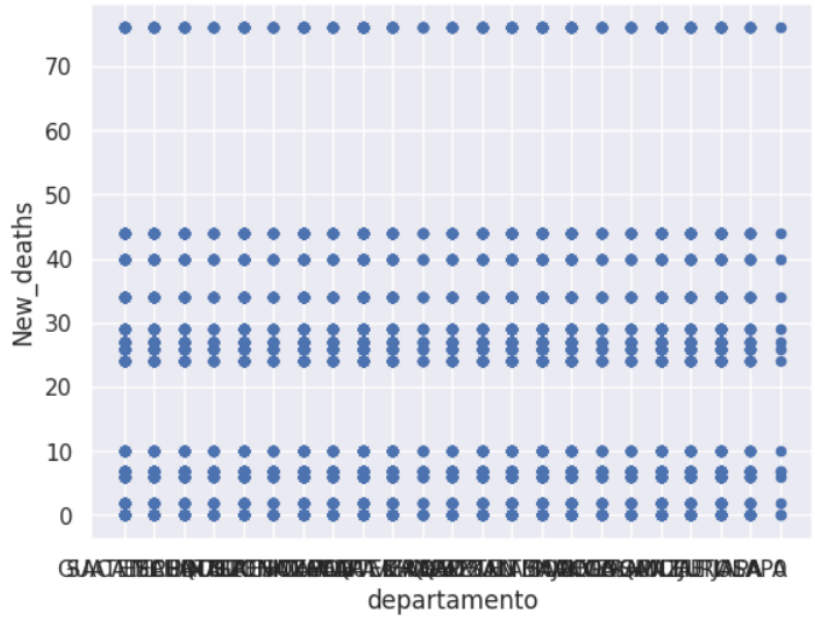



```
fr=df[df['departamento']=='QUETZALTENANGO']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['departamento','municipio'],
                 color='municipio', hover_data=['municipio','poblacion'],color_continuous_scale='Purples')
fig.show()
```

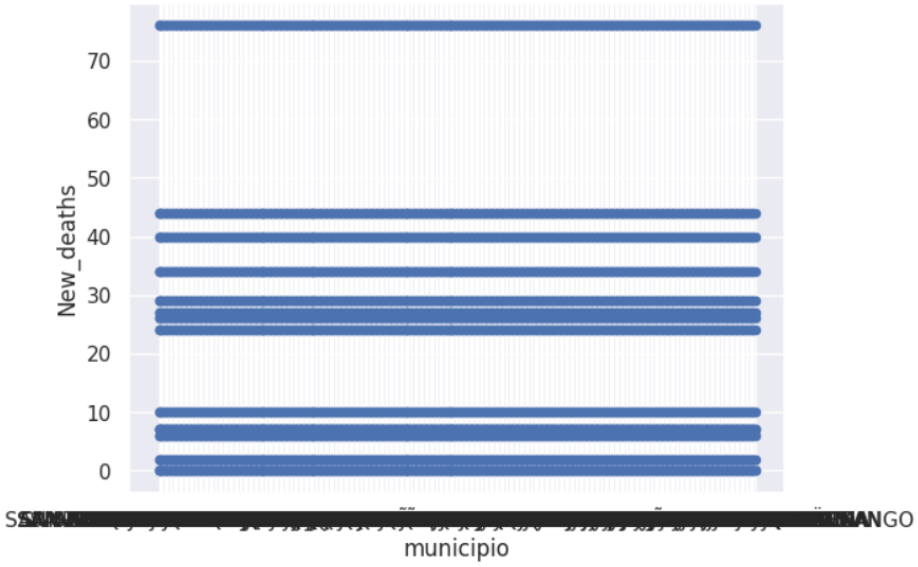
EDA - MULTIVARIABLE - DATOS CUANTITATIVOS

➔ **Graficas de dispersión de la variable “NEW DEATHS”**

```
df.plot(x='departamento', y='New_deaths',kind='scatter')
plt.show()
```

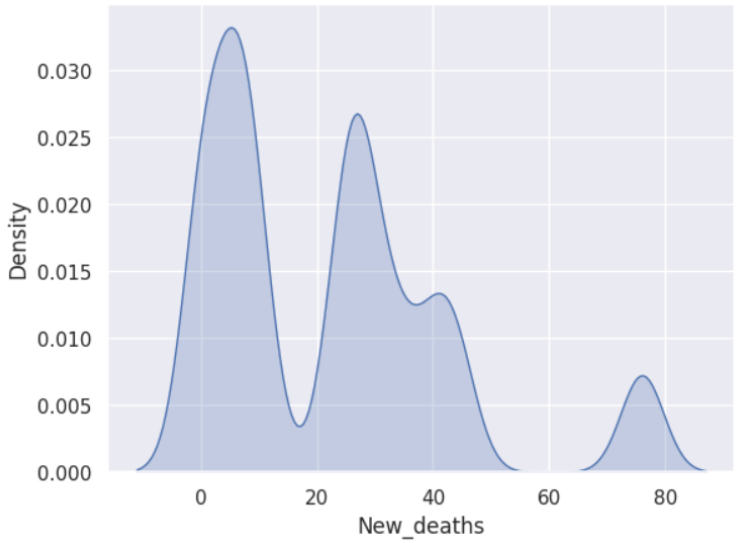



```
df.plot(x='municipio', y='New_deaths',kind='scatter')
plt.show()
```



```
sns.set(style="darkgrid")
sns.kdeplot(data=df['New_deaths'], shade=True)
```

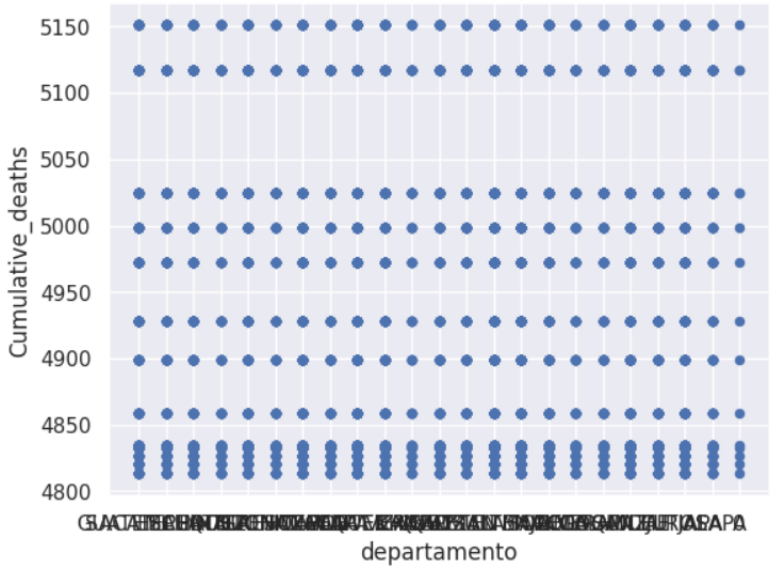
```
<ipython-input-79-8635d3e369d0>:2: FutureWarning:


`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.


<Axes: xlabel='New_deaths', ylabel='Density'>
```



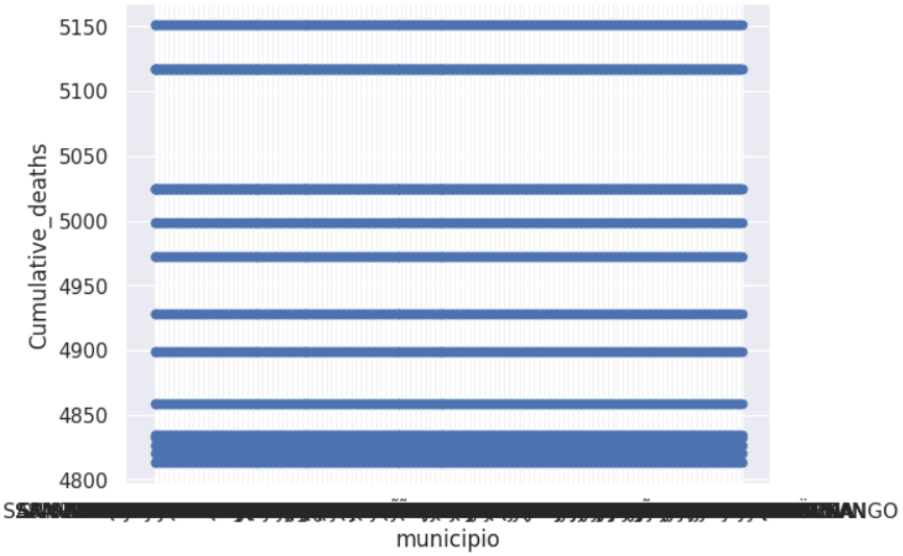
➔ Graficas de dispersión de la variable “CUMULATIVE DEATHS”

-VARIABLE DE ESTUDIO: "CANTIDAD DE MUERTES ACUMULADAS"------

```
df.plot(x='departamento', y='Cumulative_deaths',kind='scatter')
plt.show()
```

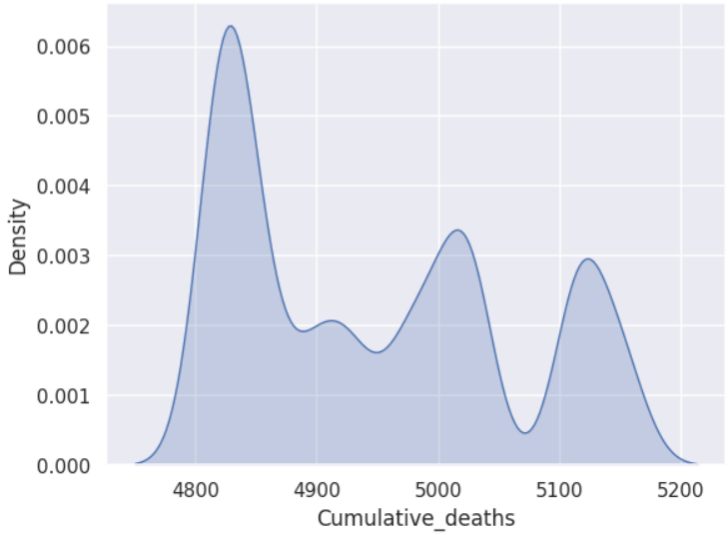


```
df.plot(x='municipio', y='Cumulative_deaths',kind='scatter')
plt.show()
```




```
sns.set(style="darkgrid")
sns.kdeplot(data=df['Cumulative_deaths'], shade=True)
```

```
<ipython-input-81-23029cc136b3>:2: FutureWarning:


`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.


<Axes: xlabel='Cumulative_deaths', ylabel='Density'>
```

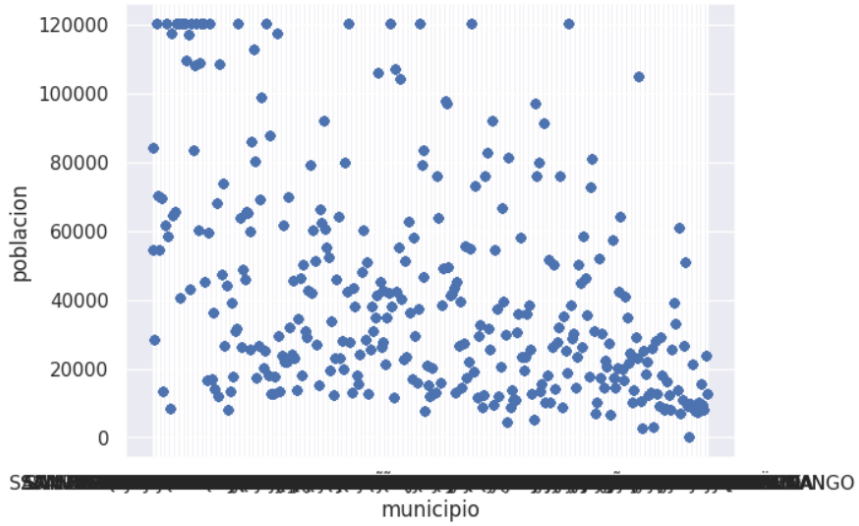


➔ **Graficas de dispersión de la variable “POBLACION POR MUNICIPIO”**

VARIABLE DE ESTUDIO: "POBLACION DE LOS MUNICIPIOS"------

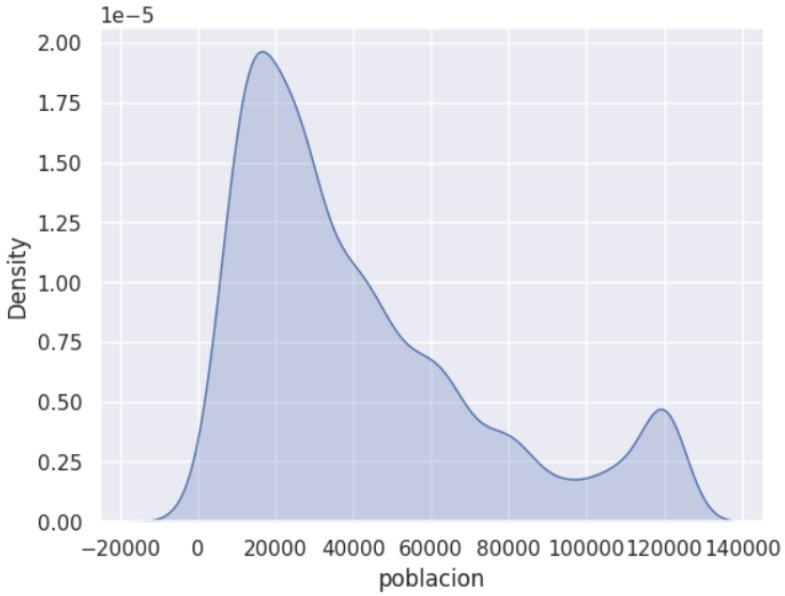
```
df.plot(x='departamento', y='poblacion',kind='scatter')
plt.show()
```


```
df.plot(x='municipio', y='poblacion',kind='scatter')
plt.show()
```



```
sns.set(style="darkgrid")
sns.kdeplot(data=df['poblacion'], shade=True)
```

```
<ipython-input-86-3e4ed6810d0a>:2: FutureWarning:


`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.


<Axes: xlabel='poblacion', ylabel='Density'>
```



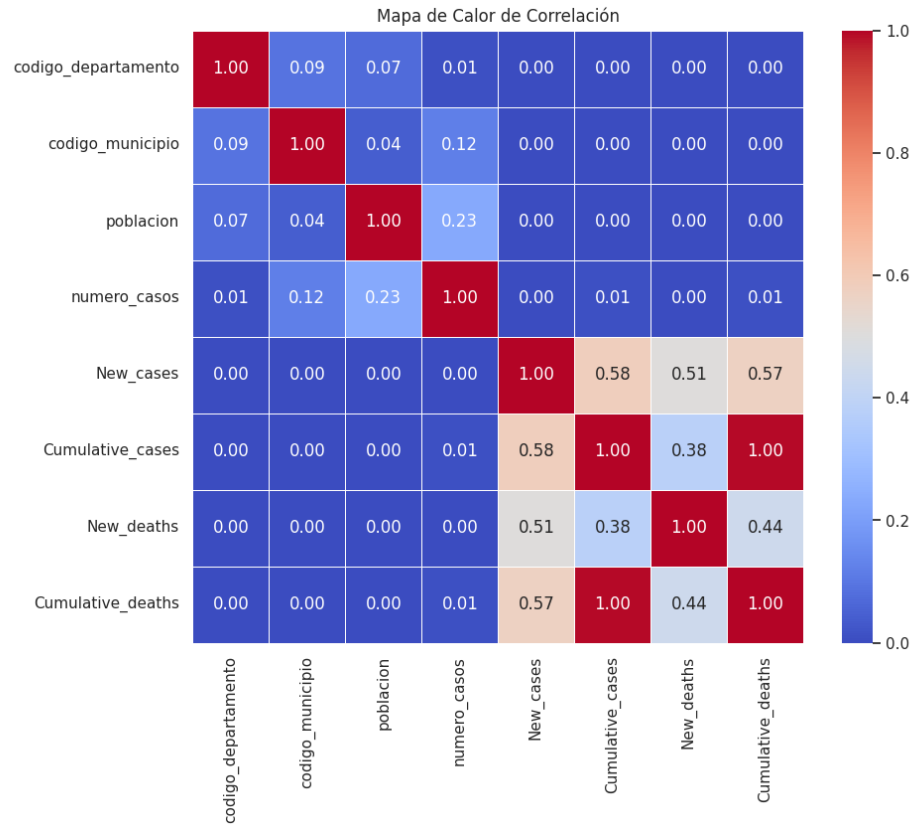
EDA - MULTIVARIABLE - DATOS CUANTITATIVOS

---EDA - MULTIVARIABLE----------

---------DATOS CUANLITATIVOS---------

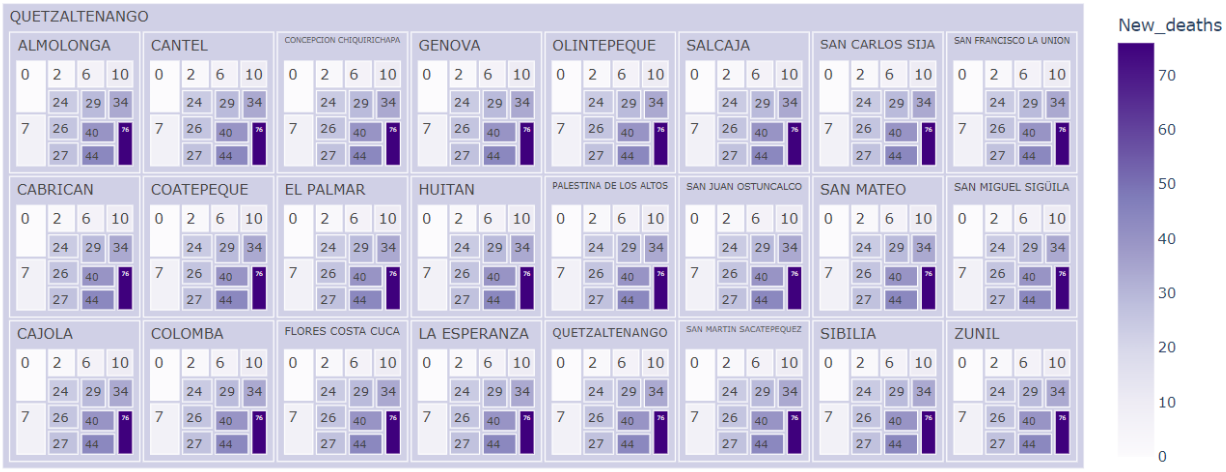
------VARIABLE DE ESTUDIO: DATOS CORRELACIONADOS------

```
[93] df['New_deaths'] = pd.to_numeric(df['New_deaths'])
     correlation_matrix = df.corr().abs()

     plt.figure(figsize=(10, 8))
     sns.set(style="white")

     sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm', square=True, linewidths=.5)

     plt.title('Mapa de Calor de Correlación')
     plt.show()
```

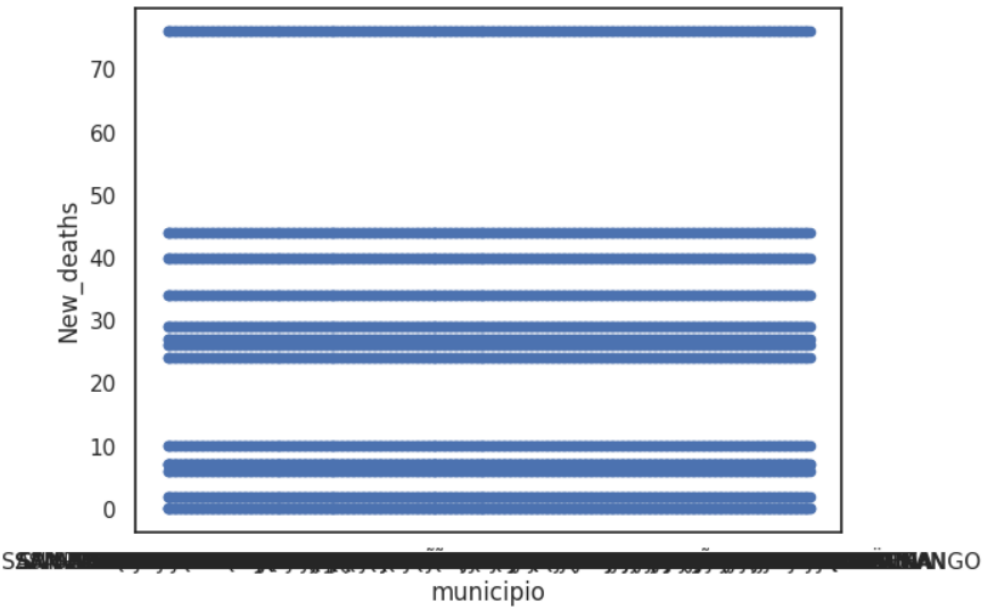


➔ **Graficas de comparacion de las variables “Municipios vs cantidad de nuevas muertes”**

```
fr=df[df['departamento']=='QUETZALTENANGO']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['departamento','municipio','New_deaths'],
                 color='New_deaths', hover_data=['New_deaths','municipio'],color_continuous_scale='Purples')
fig.show()
```



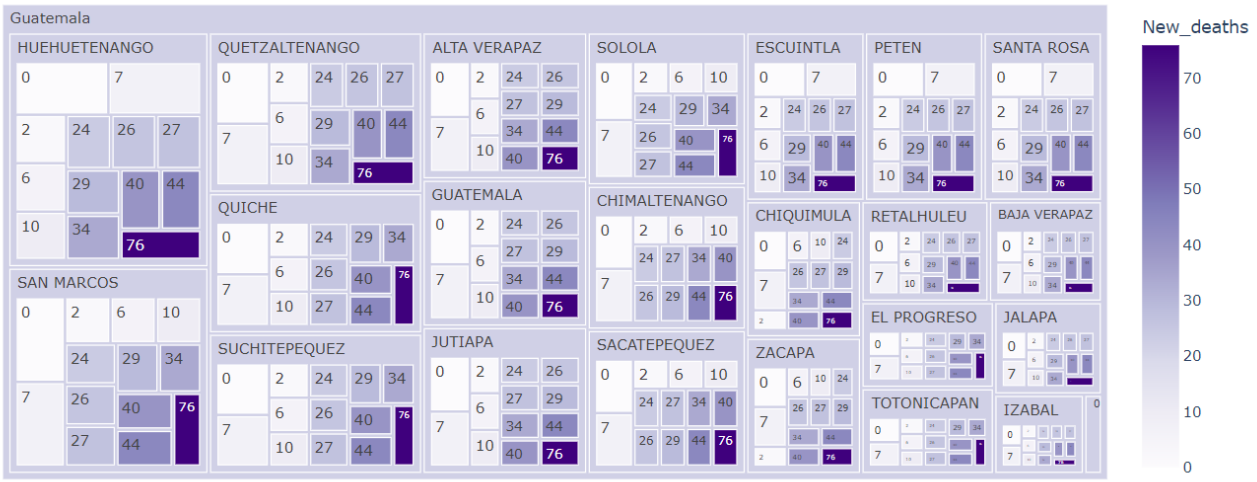

```
df.plot(x='municipio', y='New_deaths',kind='scatter')
plt.show()
```



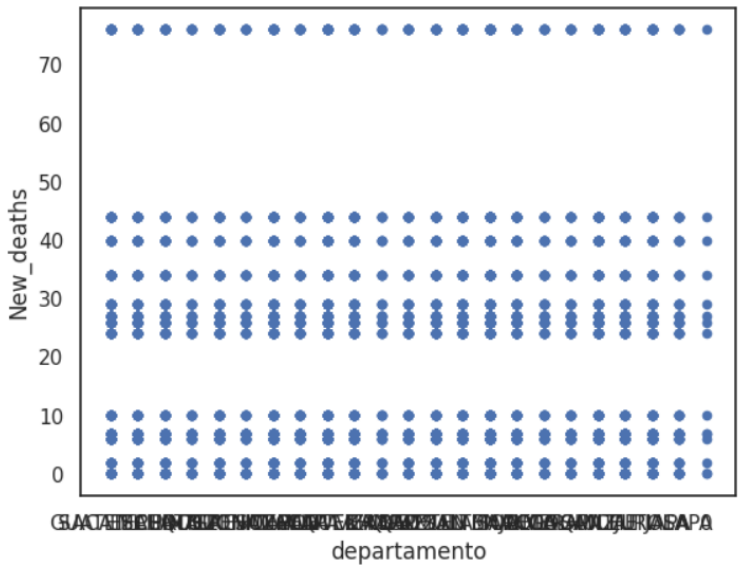
➔ **Graficas de comparacion de las variables “Departamentos vs cantidad de nuevas muertes”**

·VARIABLE DE ESTUDIO: Departamentos vs cantidad de nuevas muertes ------

```
fr=df[df['Country']=='Guatemala']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['Country','departamento','New_deaths'],
                 color='New_deaths', hover_data=['New_deaths','departamento'],color_continuous_scale='Purples')
fig.show()
```



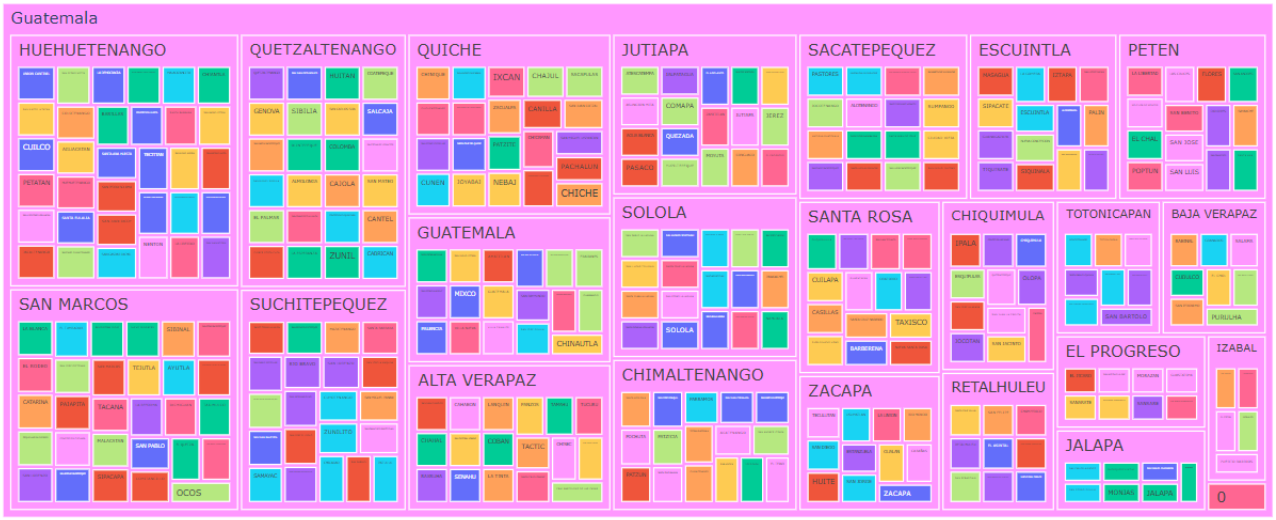

```
df.plot(x='departamento', y='New_deaths',kind='scatter')
plt.show()
```



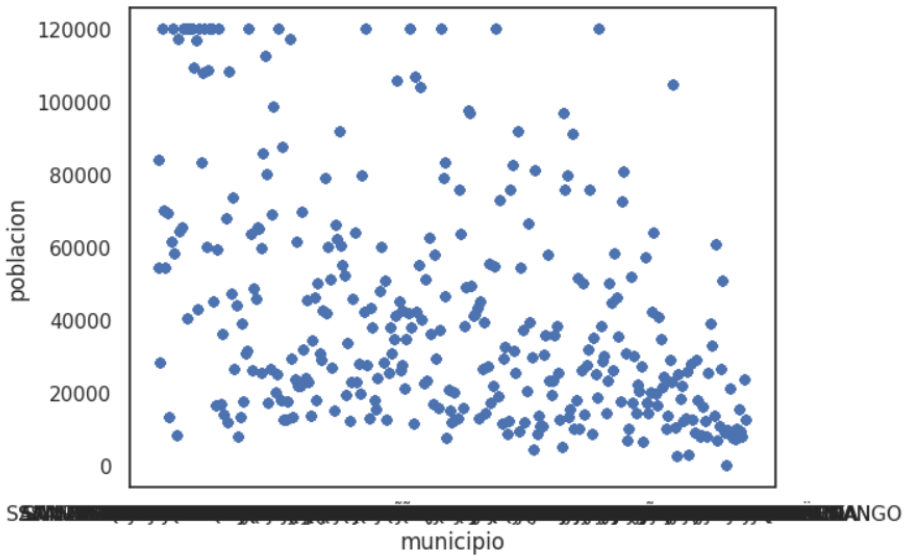
➔ **Graficas de comparacion de las variables “Municipios vs población”**

VARIABLE DE ESTUDIO: Municipios vs población ------

```
fr=df[df['Country']=='Guatemala']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['Country','departamento','municipio'],
                 color='municipio', hover_data=['municipio','departamento'],color_continuous_scale='Purples')
fig.show()
```



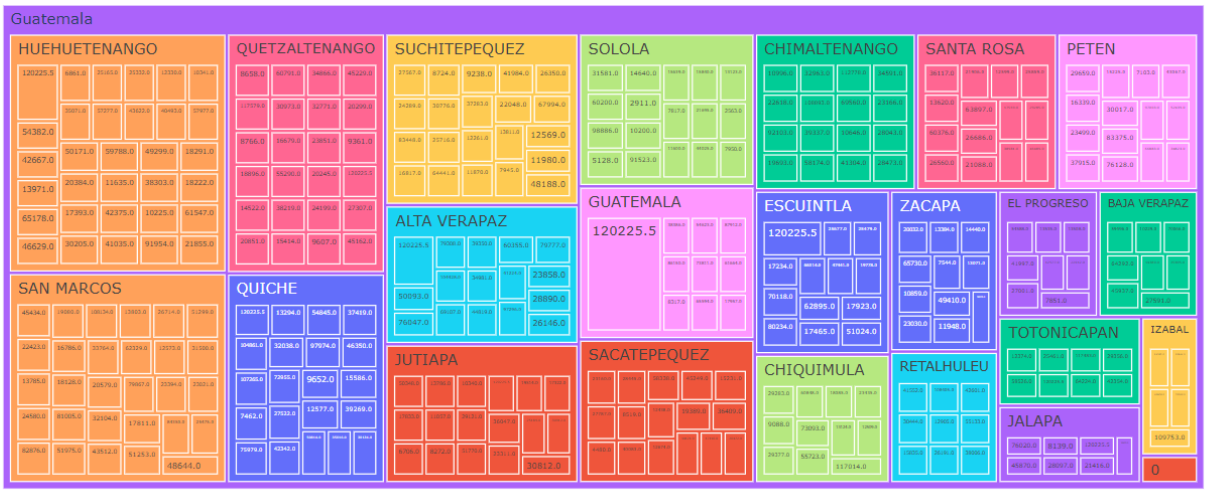

```
df.plot(x='municipio', y='poblacion',kind='scatter')
plt.show()
```



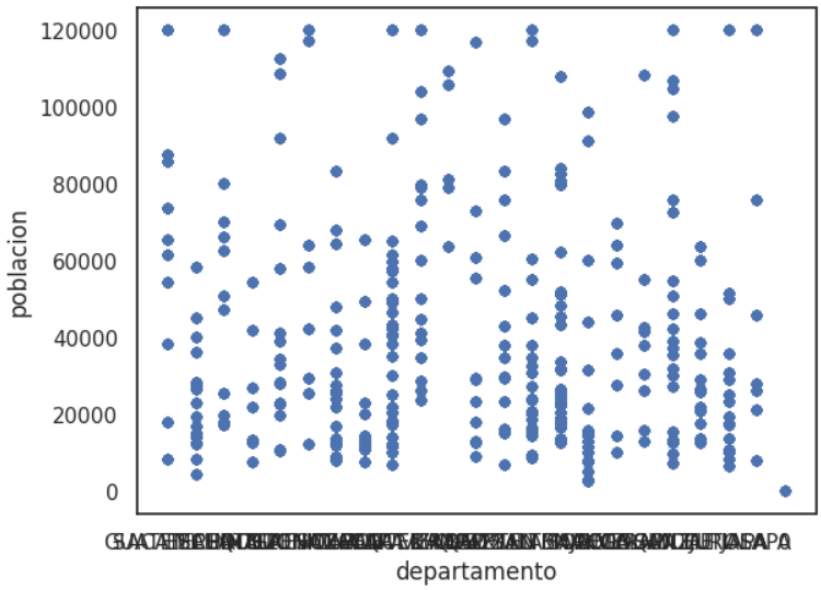
➔ Graficas de comparacion de las variables “Departamentos vs población”

-VARIABLE DE ESTUDIO: Departamentos vs población ------

```
fr=df[df['Country']=='Guatemala']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['Country','departamento', 'poblacion'],
                 color='departamento', hover_data=['municipio','poblacion'],color_continuous_scale='Purples')
fig.show()
```



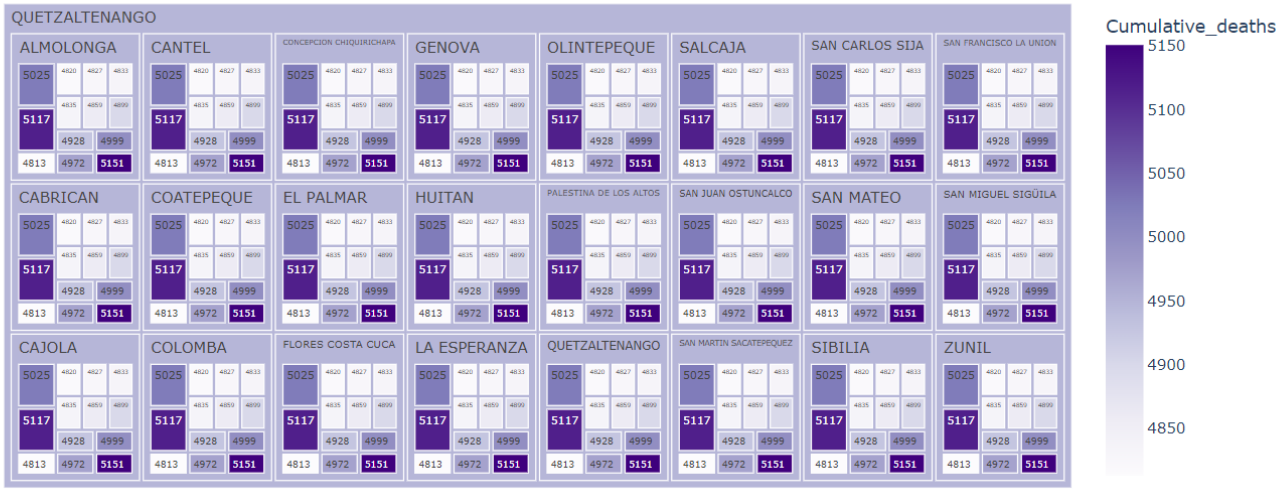

```
df.plot(x='departamento', y='poblacion',kind='scatter')
plt.show()
```



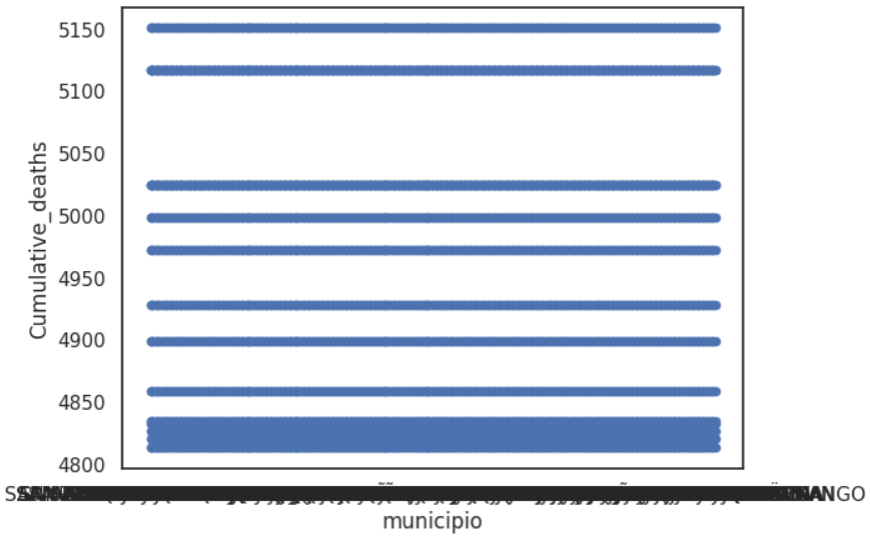
➔ **Graficas de comparacion de las variables “Municipios vs cantidad de muertes acumuladas”**

VARIABLE DE ESTUDIO: Municipios vs cantidad de muertes acumuladas ------

```
fr=df[df['departamento']=='QUETZALTENANGO']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['departamento','municipio','Cumulative_deaths'],
                 color='Cumulative_deaths', hover_data=['Cumulative_deaths','municipio'],color_continuous_scale='Purp
fig.show()
```




```
df.plot(x='municipio', y='Cumulative_deaths',kind='scatter')
plt.show()
```

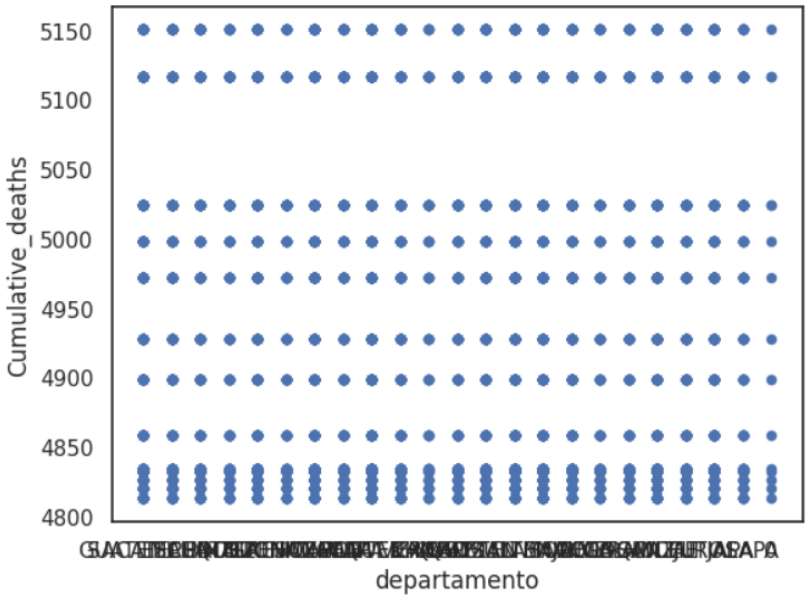


➔ **Graficas de comparacion de las variables “Departamentos vs cantidad de muertes acumuladas”**

VARIABLE DE ESTUDIO: Departamentos vs cantidad de muertes acumuladas ------

```
fr=df[df['Country']=='Guatemala']
nannef=fr.dropna()
import plotly.express as px
fig = px.treemap(nannef, path=['Country','departamento','Cumulative_deaths'],
                 color='Cumulative_deaths', hover_data=['Cumulative_deaths','departamento'],color_continuous_scale='P
fig.show()
```


```
df.plot(x='departamento', y='Cumulative_deaths',kind='scatter')
plt.show()
```



CONCLUSIONES (TOMA DE DATOS):

1. LOS DEPARTAMENTOS DONDE EXISTE UNA MAYOR CANTIDAD DE MUERTES NUEVAS SON: HUEHUETENANGO, SAN MARCOS, QUETZALTENANGO, QUICHE Y SUCHITEPEQUEZ, ESTO VA EN FUNCION DE LA CANTIDAD DE POBLACION Y EL NUMERO DE CASOS REGISTRADOS EXITOSAMENTE. POR LO QUE SE ACONSEJA TOMAR MEJORES MEDIDAS DE SEGURIDAD SOCIAL PARA EVITAR QUE ESTE NUMERO CREZCA

2. LOS DEPARTAMENTOS DONDE HAY UNA MENOR CANTIDAD DE MUERTES ACUMULADAS SON: IZABAL, JALAPA, EL PROGRESO, BAJA VERAPAZ Y RETALHULEU, ESTO VA EN FUNCION DE LA CANTIDAD DE POBLACION Y EL NUMERO DE CASOS REGISTRADOS EXITOSAMENTE. LAS MEDIDAS DE SEGURIDAD DEBEN SEGUIR SIN DESCARTAR UN ESTUDIO DE CAMPO MAS PRECISO PARA COMPROBAR LA EFICIENCIA DE LOS METODOS PARA MANTENER LA SALUD DE LA POBLACION DE ESOS DEPARTAMENTOS.

3. LOS DEPARTAMENTOS DONDE HAY UNA CANTIDAD PROMEDIO DE MUERTES ACUMULADAS SON: ALTA VERAPAZ, GUATEMALA, JUTIAPA, SOLOLA, CHIMALTENANGO Y SACATEPEQUEZ, ESTO VA EN FUNCION DE LA CANTIDAD DE POBLACION Y EL NUMERO DE CASOS REGISTRADOS EXITOSAMENTE. LAS MEDIDAS DE SEGURIDAD DEBEN SEGUIR SIN DESCARTAR UN ESTUDIO DE CAMPO.