

analisis_dataset

December 20, 2023

0.1 Obtención de los datos

Procedemos a crear una conexión con la base de datos y cargar los dataframes

```
[2]: import os
from dotenv import load_dotenv
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
load_dotenv('./config/.env')

def generate_histogram(df, column, bins, xsize, ysize, xlabel, ylabel, grid, title, color):
    """Generate a histogram from a Pandas DataFrame"""
    plt.figure(figsize=(xsize, ysize))
    plt.title(title)
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.grid(grid)
    plt.hist(df[column], bins=bins, color=color)
    plt.show()

def generate_box_plot(df, column, xlabel, width, title, color):
    """Generate a box plot from a Pandas DataFrame and print outliers"""
    plt.figure(figsize=(width, 5))
    plt.title(title)
    plt.xlabel(xlabel)
    # Utilizar seaborn para un mejor estilo y manejo de outliers
    sns.boxplot(x=df[column], color=color, showfliers=True) # Mostrar los outliers
    # Identificar y imprimir valores de outliers
    outliers = df[df[column] > df[column].quantile(0.75) + 1.5 * (df[column].quantile(0.75) - df[column].quantile(0.25))]
```

```

print(f'Outliers en {column}: {outliers[column].tolist()}')

# Agregar líneas para indicar percentiles (opcional)
plt.axvline(df[column].median(), color='red', linestyle='dashed', linewidth=2, label='Mediana')
plt.axvline(df[column].quantile(0.25), color='green', linestyle='dashed', linewidth=2, label='Cuartil 1')
plt.axvline(df[column].quantile(0.75), color='blue', linestyle='dashed', linewidth=2, label='Cuartil 3')

plt.legend() # Mostrar la leyenda
plt.grid(True)
plt.show()

def univariate_analysis(df, column, width, height, title, color):
    """
    Generate a bar plot for a single variable (municipios or departamentos) from a Pandas DataFrame.
    """

    Parameters:
    - df: DataFrame
    - column: str, nombre de la columna para el análisis (municipios o departamentos)
    - width: int, ancho de la figura
    - height: int, altura de la figura
    - title: str, título del gráfico
    - color: str, color de las barras
    """
    plt.figure(figsize=(width, height))
    plt.title(title)

    sns.countplot(x=df[column], color=color)

    plt.xlabel(column)
    plt.ylabel('Número de Registros')
    plt.grid(True)

    plt.xticks(rotation=45, ha='right', fontsize=8) # Rotar etiquetas y ajustar tamaño de fuente
    plt.tight_layout() # Ajustar diseño para evitar recorte de etiquetas
    plt.show()

def generate_bar_plot(df, x_column, y_column, width, grid, title, color):
    """
    Generate a bar plot from a Pandas DataFrame.
    """

```

```

Parameters:
- df: DataFrame
- x_column: str, nombre de la columna en el eje x
- y_column: str, nombre de la columna en el eje y
- width: int, ancho de la figura
- title: str, título del gráfico
- color: str, color de las barras
- grid: bool, si se muestra la grilla
"""

plt.figure(figsize=(width, 5))
plt.title(title)

sns.barplot(x=df[x_column], y=df[y_column], color=color)

plt.xlabel(x_column)
plt.ylabel(y_column)
plt.grid(grid)
plt.show()

def conectar():
    """Connect to MySQL database"""
    return mysql.connector.connect(
        host=os.getenv('DB_HOST', 'localhost'),
        user=os.getenv('DB_USER', 'root'),
        password=os.getenv('DB_PASSWORD', 'admin'),
        database=os.getenv('DB_NAME', 'etl_dataset')
    )

def load_tables(query):
    """Load data from MySQL database into a Pandas DataFrame"""
    conexion = None
    cursor = None
    try:
        conexion = conectar()
        cursor = conexion.cursor()
        cursor.execute(query)
        # Get all headers
        columns = [column[0] for column in cursor.description]

        # Get all rows
        rows = cursor.fetchall()

        # return the DataFrame
        df = pd.DataFrame(rows, columns=columns)
        print(f"Datos obtenidos correctamente \n {query}")
        print(f"Registros obtenidos: {len(df)}\n")
        return df
    
```

```

except mysql.connector.Error as error:
    print("Error al obtener los datos de la tabla {}".format(error))
    return None
finally:
    if cursor:
        cursor.close()
    if conexion:
        conexion.close()
print("Conectado a la base de datos\n")

#Datos cuantitativos
queryMonoTown = """SELECT departament, town, population
FROM Town t
INNER JOIN Departament d
    ON t.code_departament = d.code_departament"""
querMonoCovidCases = """SELECT date_reported, new_deaths_f2 as new_deaths,
↪cumulative_deaths_f2 as cumulative_deaths
FROM CovidCases"""

dfMonoTown = load_tables(queryMonoTown)
dfCovidCases = load_tables(querMonoCovidCases)

#Datos cualitativos
queryQualityTown = "SELECT * FROM Town"
queryQualityDepartament = "SELECT * FROM Departament"

queryFull = """
SELECT cc.date_reported, cc.new_deaths_f2 as new_deaths, cc.
↪cumulative_deaths_f2 as cumulative_deaths,
    cc.new_cases_f2 as new_cases, cc.cumulative_cases_f2 as cumulative_cases,
    td.deceases_f1 as deceases, t.code_town, t.town, t.population, t.
↪code_departament, d.departament
FROM CovidCases cc
    INNER JOIN TownDemises td
        ON cc.date_reported=td.date_reported
    INNER JOIN Town t
        ON t.code_town=td.code_town
    INNER JOIN Departament d
        ON d.code_departament=t.code_departament"""

dfTown = load_tables(queryQualityTown)
dfDepartament = load_tables(queryQualityDepartament)
dfFull = load_tables(queryFull)

```

Conecado a la base de datos

Datos obtenidos correctamente

```

SELECT departament, town, population
  FROM Town t
 INNER JOIN Departament d
    ON t.code_departament = d.code_departament
Registros obtenidos: 339

Datos obtenidos correctamente
SELECT date_reported, new_deaths_f2 as new_deaths, cumulative_deaths_f2 as
cumulative_deaths
  FROM CovidCases
Registros obtenidos: 1464

Datos obtenidos correctamente
SELECT * FROM Town
Registros obtenidos: 339

Datos obtenidos correctamente
SELECT * FROM Departament
Registros obtenidos: 22

Datos obtenidos correctamente

SELECT cc.date_reported, cc.new_deaths_f2 as new_deaths, cc.cumulative_deaths_f2
as cumulative_deaths,
      cc.new_cases_f2 as new_cases, cc.cumulative_cases_f2 as cumulative_cases,
      td.deceases_f1 as deceases, t.code_town, t.town, t.population,
t.code_departament, d.departament
  FROM CovidCases cc
    INNER JOIN TownDemises td
      ON cc.date_reported=td.date_reported
    INNER JOIN Town t
      ON t.code_town=td.code_town
    INNER JOIN Departament d
      ON d.code_departament=t.code_departament
Registros obtenidos: 424128

```

0.2 EDA Mono Variable

Análisis Datos Cuantitativos

- Análisis mono variable de la cantidad de nuevas muertes, cantidad de muertes acumuladas y población de los municipios.
- Mostrar estadístico de conteo, valores únicos, promedio, desviación estándar, mínimo, máximo y cuartiles.
- Realizar un histograma y un diagrama de caja con cada una de las variables mencionadas anteriormente.

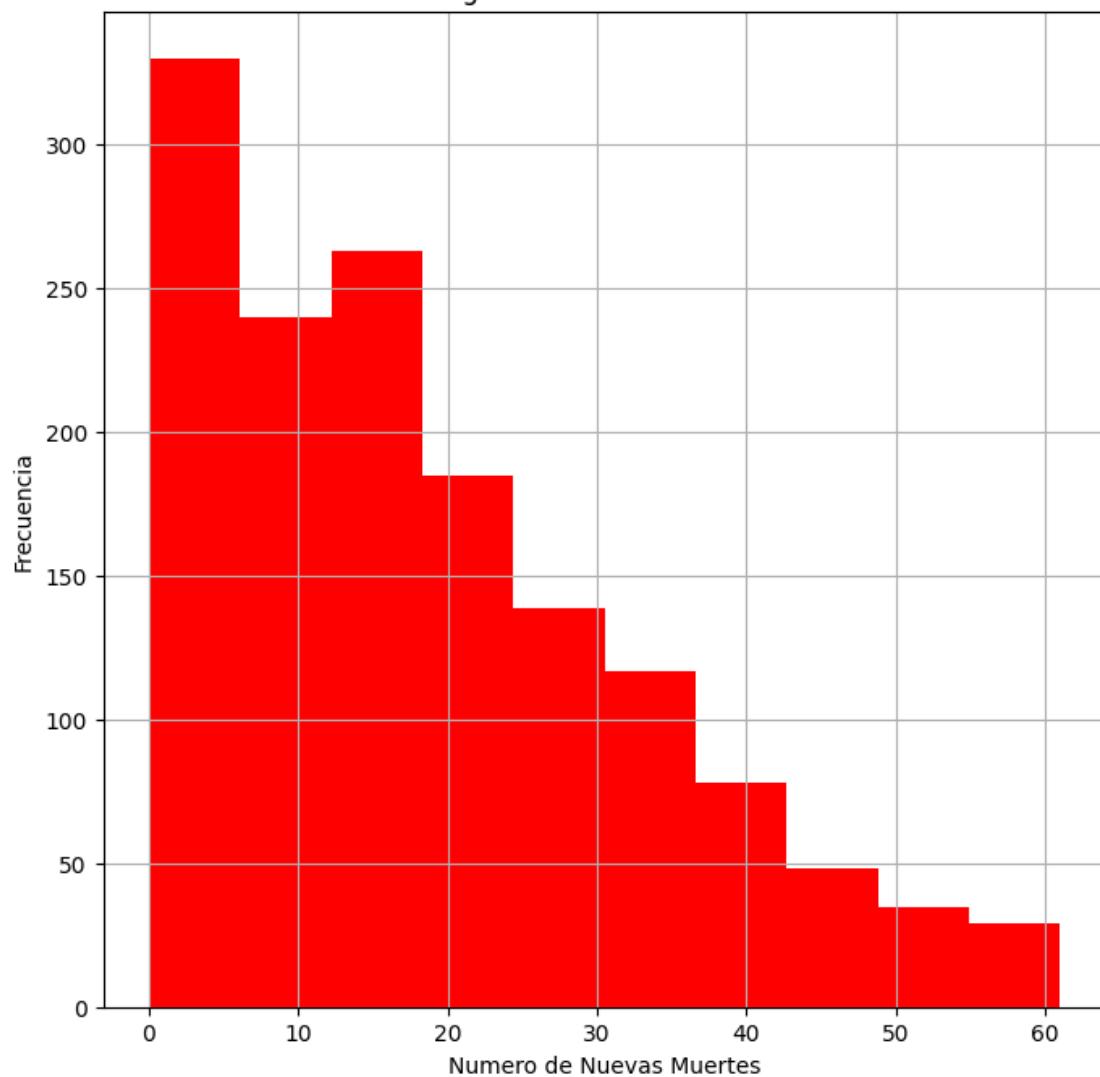
Análisis de la variable nuevas muertes

- La variable usada se llama new_deaths
- Se encontraron 1464 registros
- El promedio de los valores es de 18.820355
- La desviación estándar de 14.474994
- El valor mínimo es 0
- El valor máximo es 61
- El valor más encontrado del cuartil 25% es 7
- El valor más encontrado del cuartil 50% es 16
- El valor más encontrado del cuartil 75% es 27.25
- Valores únicos fueron 53
- El IQR se encuentra aproximadamente entre 7 y 28
- Outliers de 58, 59 y 61

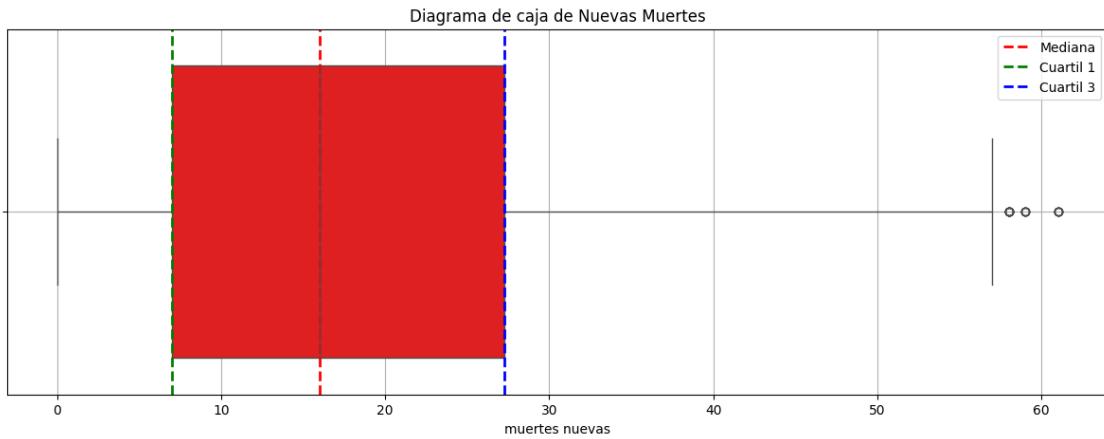
```
[3]: #Obteniendo estadísticas de las variables y valores únicos ordenados forma ↴
      ↴descendente
stats_new_deaths = dfCovidCases['new_deaths'].describe()
unique_new_deaths = sorted(dfCovidCases['new_deaths'].unique(), reverse=True)
# Imprimiendo estadísticas de las variables
print("VARIABLE new_deaths (nuevas muertes)")
print("Estadísticas de la variable")
print(stats_new_deaths)
print(f"Valores únicos de la variable {len(unique_new_deaths)}")
print(', '.join(map(str, unique_new_deaths)))
generate_histogram(dfCovidCases, 'new_deaths', 10, 8, 8, "Número de Nuevas ↴
      ↴Muertes", "Frecuencia",
                  True, 'Histograma de nuevas muertes', 'red')
generate_box_plot(dfCovidCases, 'new_deaths', 'muertes nuevas', 15, 'Diagrama de ↴
      ↴caja de Nuevas Muertes', 'red')
```

```
VARIABLE new_deaths (nuevas muertes)
Estadísticas de la variable
count    1464.000000
mean     18.820355
std      14.474994
min      0.000000
25%     7.000000
50%     16.000000
75%     27.250000
max      61.000000
Name: new_deaths, dtype: float64
Valores únicos de la variable 53
61, 59, 58, 57, 54, 53, 49, 48, 47, 45, 44, 42, 41, 40, 39, 38, 37, 35, 34, 33,
32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13,
12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0
```

Histograma de nuevas muertes



Outliers en new_deaths: [58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 59, 59, 59, 59, 59, 61, 61, 61, 61, 61, 61]



Análisis de la variable muertes acumuladas

- La variable usada se llama cumulative_deaths
- Se encontraron 1464 registros
- El promedio de los valores es de 2275.53
- La desviación estándar de 1596.242666
- El valor mínimo es 0
- El valor máximo es 4803
- El valor más encontrado del cuartil 25% es 531
- El valor más encontrado del cuartil 50% es 2630
- El valor más encontrado del cuartil 75% es 3682
- Valores únicos fueron 53
- El IQR se encuentra aproximadamente entre mayor a 500 y 37000
- No se detectaron outliers

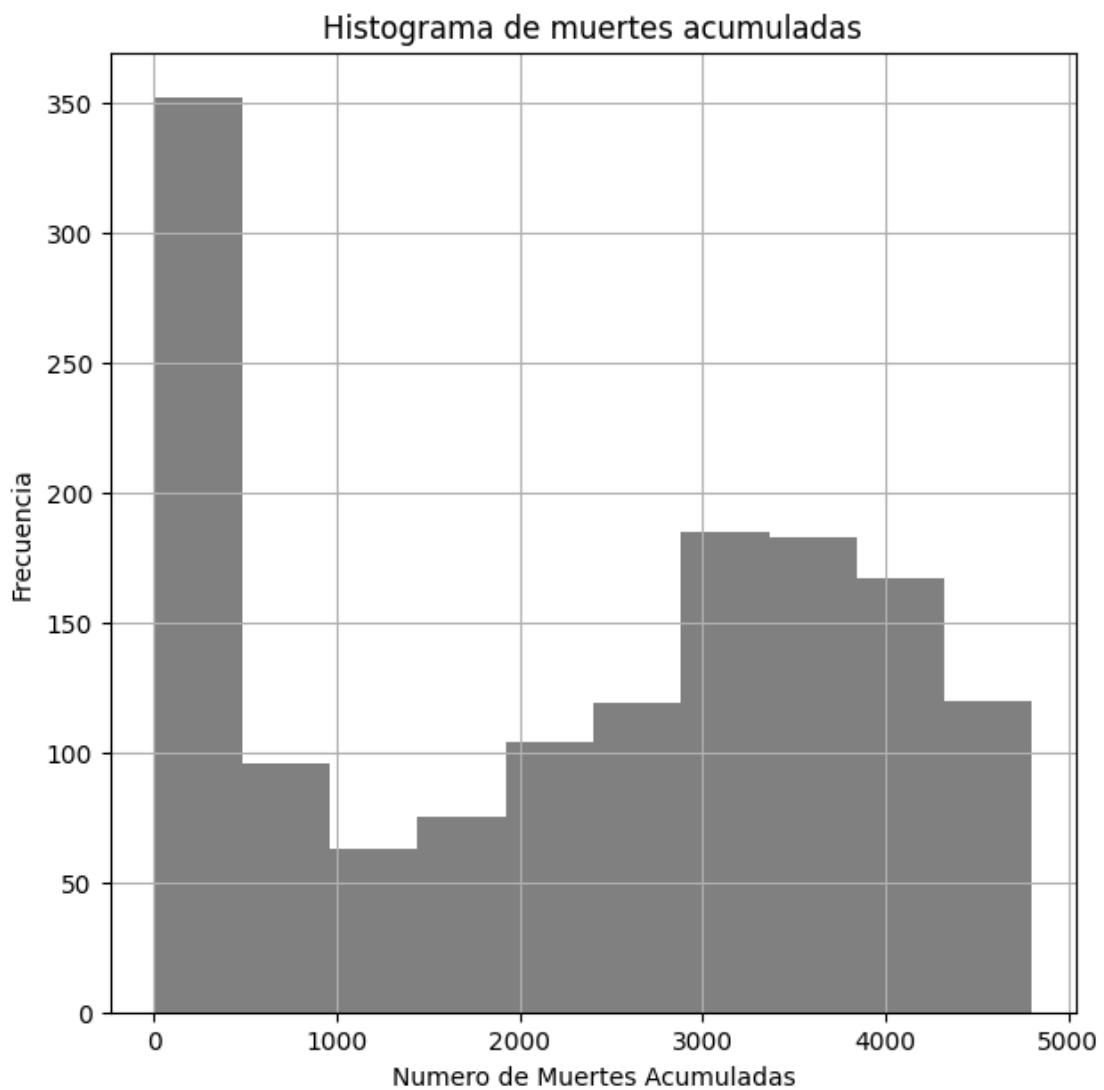
```
[4]: stats_cumulative_deaths = dfCovidCases['cumulative_deaths'].describe()
unique_cumulative_deaths = sorted(dfCovidCases['cumulative_deaths'].unique(), reverse=True)

print("Estadísticas de la variable")
print(stats_cumulative_deaths)
print(f"Valores únicos de la variable {len(unique_new_deaths)}")
print(', '.join(map(str, unique_cumulative_deaths)))
generate_histogram(dfCovidCases, 'cumulative_deaths', 10, 7, 7, "Número de Muertes Acumuladas", "Frecuencia", True, 'Histograma de muertes acumuladas', 'gray')
generate_box_plot(dfCovidCases, 'cumulative_deaths', 'Muertes Acumuladas', 7, 'Diagrama de caja de Muertes Acumuladas', 'gray')
```

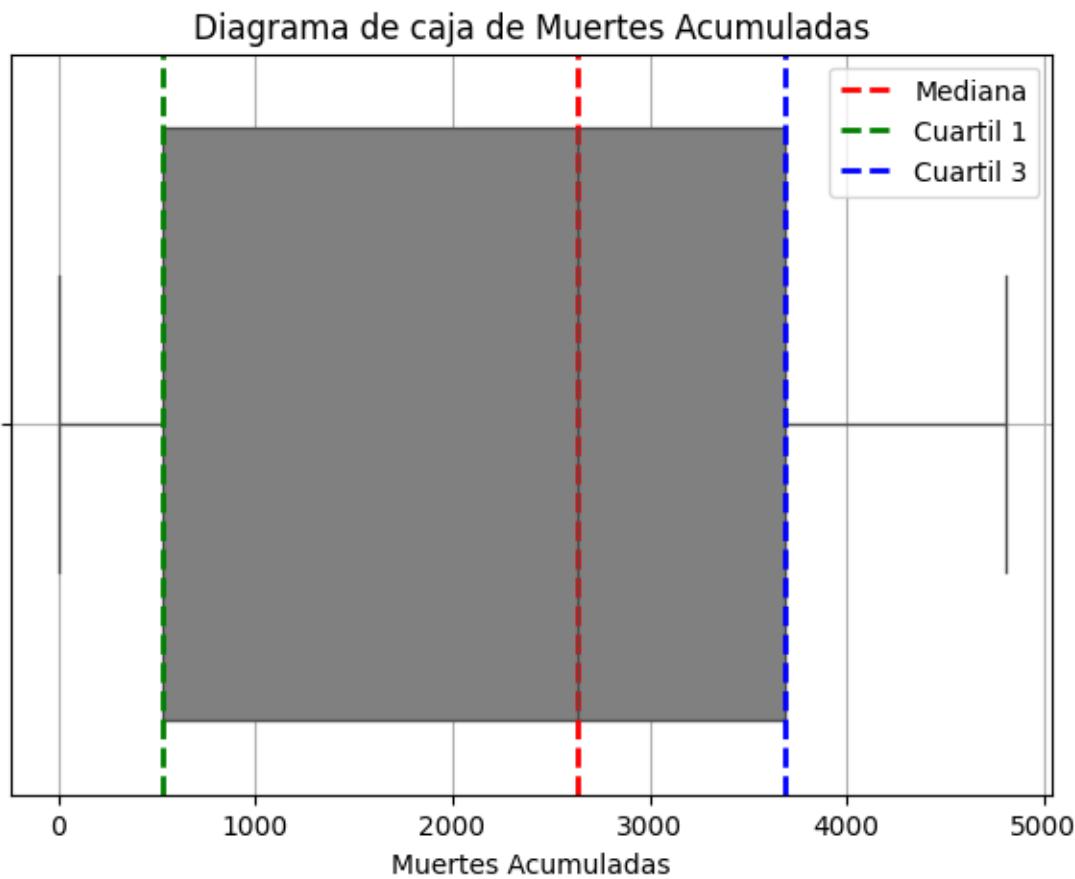
Estadísticas de la variable

count	1464.000000
mean	2275.528689

```
std      1596.242666
min      0.000000
25%     531.000000
50%    2630.000000
75%    3682.000000
max    4803.000000
Name: cumulative_deaths, dtype: float64
Valores únicos de la variable 53
4803, 4781, 4773, 4768, 4763, 4757, 4749, 4739, 4718, 4688, 4656, 4624, 4589,
4551, 4510, 4476, 4445, 4423, 4405, 4376, 4345, 4311, 4286, 4274, 4250, 4239,
4224, 4209, 4191, 4178, 4171, 4166, 4161, 4141, 4133, 4107, 4099, 4092, 4076,
4074, 4067, 4050, 4008, 3947, 3938, 3932, 3920, 3880, 3858, 3845, 3832, 3823,
3821, 3811, 3794, 3766, 3752, 3748, 3738, 3729, 3714, 3704, 3682, 3665, 3651,
3644, 3609, 3594, 3580, 3567, 3546, 3541, 3530, 3515, 3478, 3453, 3430, 3410,
3387, 3384, 3365, 3356, 3347, 3335, 3310, 3302, 3293, 3285, 3267, 3261, 3246,
3238, 3234, 3229, 3213, 3186, 3170, 3154, 3137, 3124, 3119, 3105, 3076, 3036,
3009, 2984, 2972, 2957, 2949, 2929, 2918, 2897, 2890, 2862, 2852, 2845, 2825,
2804, 2790, 2778, 2760, 2740, 2728, 2709, 2685, 2662, 2630, 2611, 2594, 2580,
2532, 2506, 2467, 2419, 2389, 2379, 2355, 2341, 2296, 2267, 2233, 2222, 2211,
2197, 2168, 2119, 2072, 2037, 2013, 1995, 1957, 1924, 1867, 1835, 1782, 1761,
1734, 1699, 1669, 1632, 1573, 1531, 1502, 1449, 1443, 1404, 1350, 1302, 1244,
1219, 1172, 1139, 1092, 1053, 1004, 981, 947, 920, 880, 843, 817, 773, 746, 727,
706, 672, 623, 601, 582, 547, 531, 514, 483, 449, 432, 418, 399, 384, 367, 351,
334, 316, 289, 267, 252, 230, 216, 158, 143, 123, 116, 108, 102, 90, 80, 68, 63,
59, 58, 57, 53, 48, 45, 43, 38, 35, 33, 30, 29, 27, 26, 24, 21, 19, 18, 17, 16,
15, 11, 7, 5, 3, 1, 0
```



Outliers en cumulative_deaths: []



Análisis de la variable población

- La variable usada se llama population
- Debido a la desproporción de los datos se aplico el $\log(1+x)$ a los datos para eliminar ceros y ajustar escalas
- Se encontraron 339 registros
- El promedio de los valores es de 4,9978.85
- La desviación estándar de 8,1343.45
- El valor mínimo es 2,563
- El valor máximo es 1,205,668
- El valor más encontrado del cuartil 25% es 17,517
- El valor más encontrado del cuartil 50% es 30,973
- El valor más encontrado del cuartil 75% es 59,011
- Valores únicos fueron 53
- El IQR se encuentra aproximadamente entre mayor a 500 y 37000
- Se encontraron numerosos outliers

```
[5]: stats_population = dfMonoTown['population'].describe()
unique_population = sorted(dfMonoTown['population'].unique(), reverse=True)
```

```

print("Estadísticas de la variable")
print(stats_population)
print(f"Valores únicos de la variable {len(unique_new_deaths)}")
print(' '.join(map(str, unique_population)))
#Aplicar una transformación logarítmica a la variable population log(1+x)
dfMonoTown['population_log'] = np.log(dfMonoTown['population'])
dfFull['population_log'] = np.log(dfFull['population'])
generate_histogram(dfMonoTown, 'population', 10, 7, 7,"Población", ↴
    "Frecuencia", True, 'Histograma de población', 'blue')
generate_box_plot(dfMonoTown, 'population', 'Población', 15, 'Diagrama de caja de Población', 'blue')
generate_histogram(dfMonoTown, 'population_log', 20, 7, 7,"Logaritmo Natural de Nuevas Muertes (1+x)", "Frecuencia", True, 'Histograma de población (transformado)', 'blue')
generate_box_plot(dfMonoTown, 'population_log', 'Población', 10, 'Diagrama de caja de Población Transformado', 'blue')

```

Estadísticas de la variable

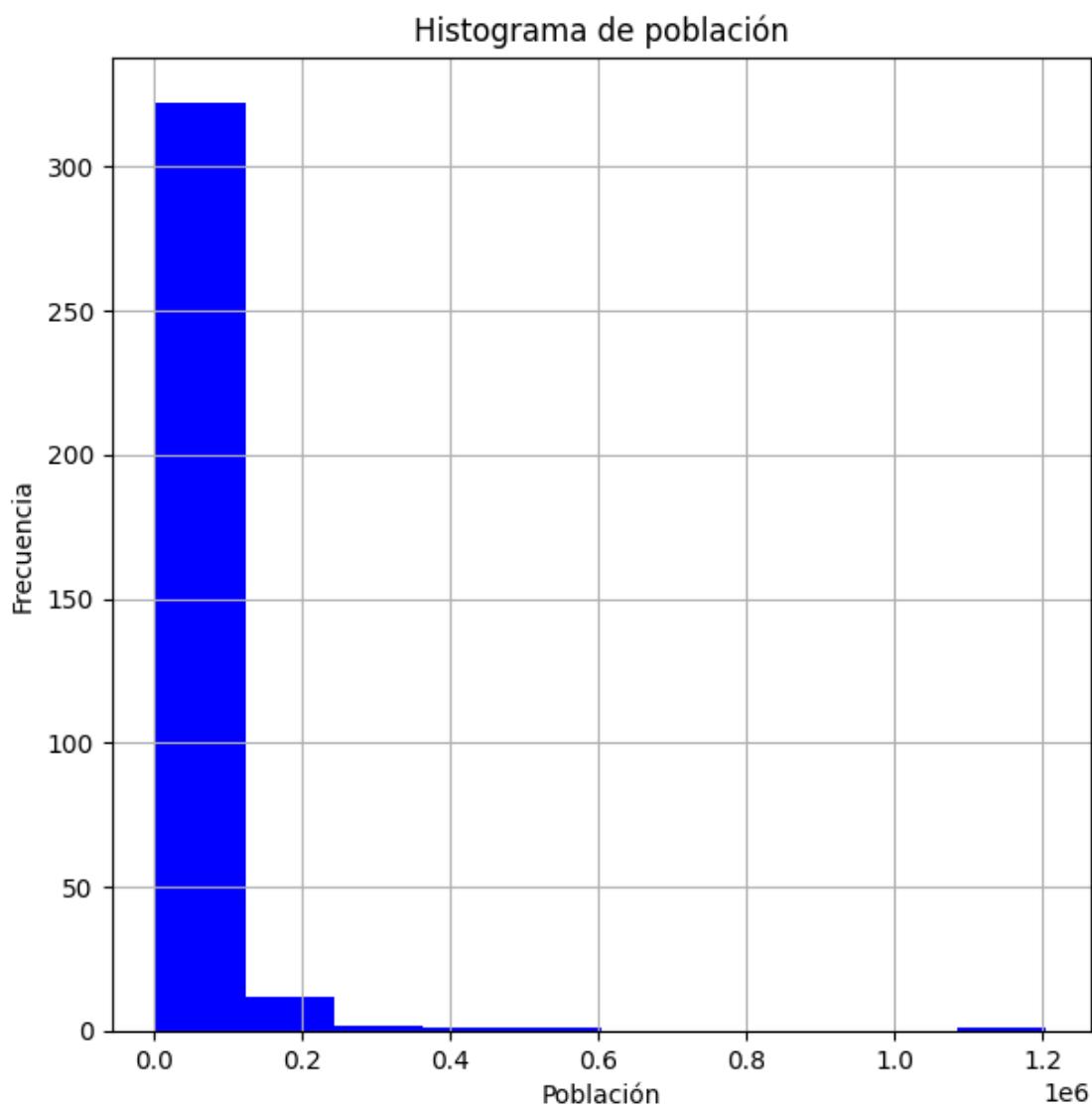
count	3.390000e+02
mean	4.997885e+04
std	8.134345e+04
min	2.563000e+03
25%	1.751700e+04
50%	3.097300e+04
75%	5.901100e+04
max	1.205668e+06

Name: population, dtype: float64

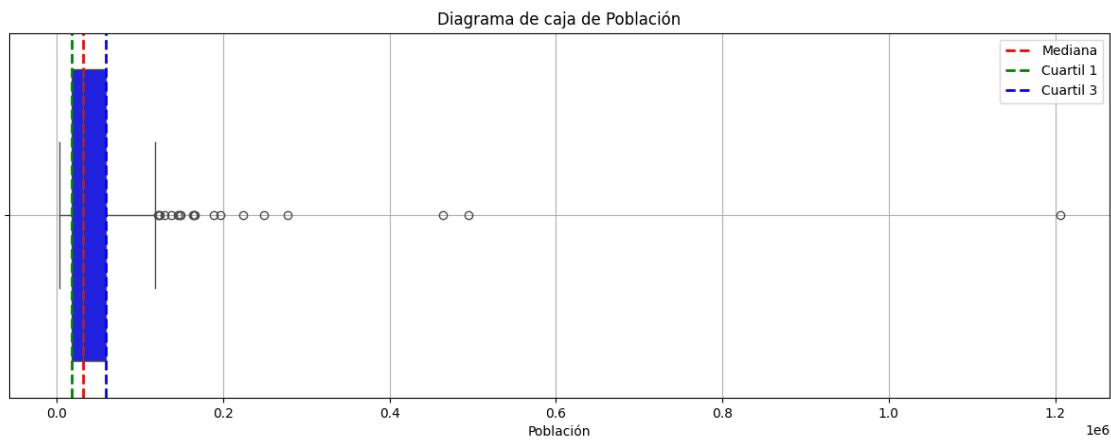
Valores únicos de la variable 53

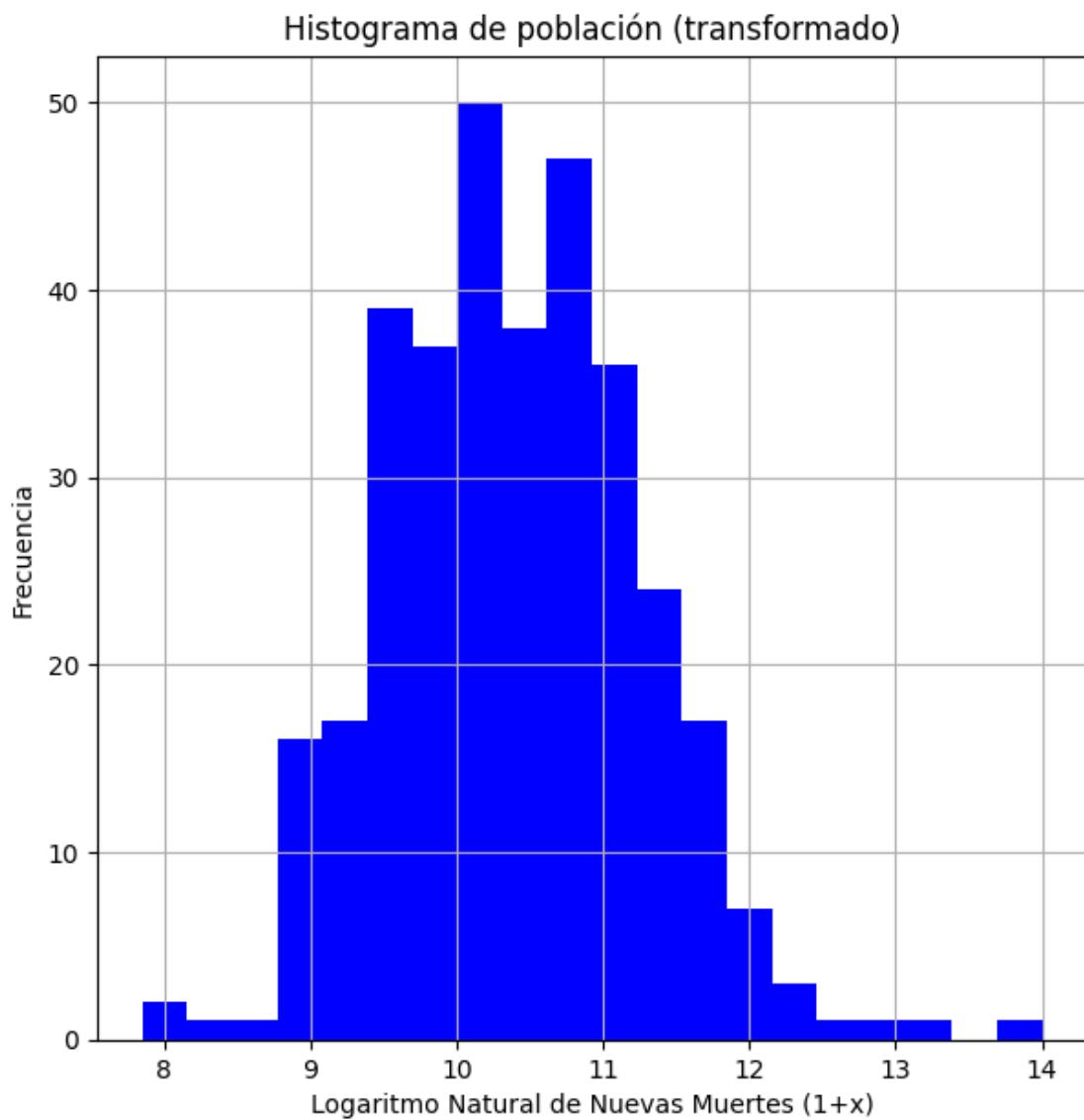
1205668, 494561, 464528, 276836, 248880, 224109, 196867, 188522, 166078, 165026, 163665, 148943, 147604, 145417, 136909, 129232, 123979, 122466, 121721, 117579, 117483, 117014, 112778, 109753, 108893, 108405, 108134, 107265, 106156, 104861, 104428, 98886, 97974, 97296, 97003, 92103, 91954, 91523, 87912, 86150, 84350, 83448, 83375, 82876, 81385, 81005, 80234, 79867, 79777, 79360, 79308, 76128, 76047, 76020, 75979, 73811, 73093, 72955, 70118, 70066, 69560, 69107, 67994, 66883, 66314, 65730, 65594, 65178, 64441, 64292, 64224, 63897, 63841, 62895, 62329, 61664, 61547, 60848, 60791, 60376, 60355, 60200, 59788, 59496, 58526, 58338, 58174, 57977, 57277, 55723, 55290, 55133, 54845, 54623, 54588, 54382, 52439, 51975, 51770, 51299, 51253, 51024, 50814, 50348, 50171, 50093, 49410, 49299, 48644, 48188, 47441, 46629, 46489, 46350, 45937, 45870, 45434, 45249, 45229, 45162, 44819, 44025, 43622, 43512, 43067, 42667, 42601, 42375, 42354, 42342, 41997, 41984, 41552, 41304, 41224, 41035, 40493, 40083, 39350, 39337, 39269, 38981, 38386, 38303, 38270, 38219, 38006, 37915, 37419, 37283, 36409, 36117, 36047, 35895, 35616, 35071, 34981, 34866, 34823, 34591, 33764, 32963, 32771, 32104, 32038, 31581, 31500, 30973, 30912, 30812, 30776, 30444, 30205, 30134, 30017, 29659, 29377, 29356, 29285, 29283, 29121, 28890, 28473, 28445, 28097, 28043, 27787, 27591, 27567, 27522, 27307, 27001, 26714, 26686, 26560,

26472, 26350, 26191, 26170, 26146, 25859, 25716, 25677, 25479, 25475, 25461, 25332, 25189, 25165, 24580, 24289, 24199, 23858, 23851, 23499, 23435, 23394, 23311, 23166, 23160, 23030, 23021, 22618, 22423, 22048, 22012, 21906, 21855, 21695, 21416, 21088, 20851, 20579, 20384, 20299, 20245, 20032, 19778, 19693, 19514, 19389, 19080, 18896, 18291, 18222, 18128, 18085, 17957, 17923, 17833, 17811, 17569, 17465, 17393, 17322, 17234, 17116, 16817, 16786, 16679, 16339, 15840, 15835, 15639, 15586, 15414, 15231, 15225, 14640, 14522, 14440, 14380, 14121, 13971, 13811, 13803, 13786, 13785, 13620, 13535, 13508, 13384, 13294, 13124, 13123, 13071, 12905, 12674, 12599, 12577, 12573, 12569, 12509, 12438, 12374, 12330, 12261, 11980, 11948, 11870, 11635, 11600, 11057, 10996, 10859, 10646, 10341, 10340, 10229, 10225, 10200, 9652, 9607, 9361, 9238, 9088, 8766, 8724, 8658, 8519, 8317, 8272, 8139, 7950, 7945, 7851, 7817, 7544, 7462, 7103, 6861, 6706, 5128, 4480, 2911, 2563

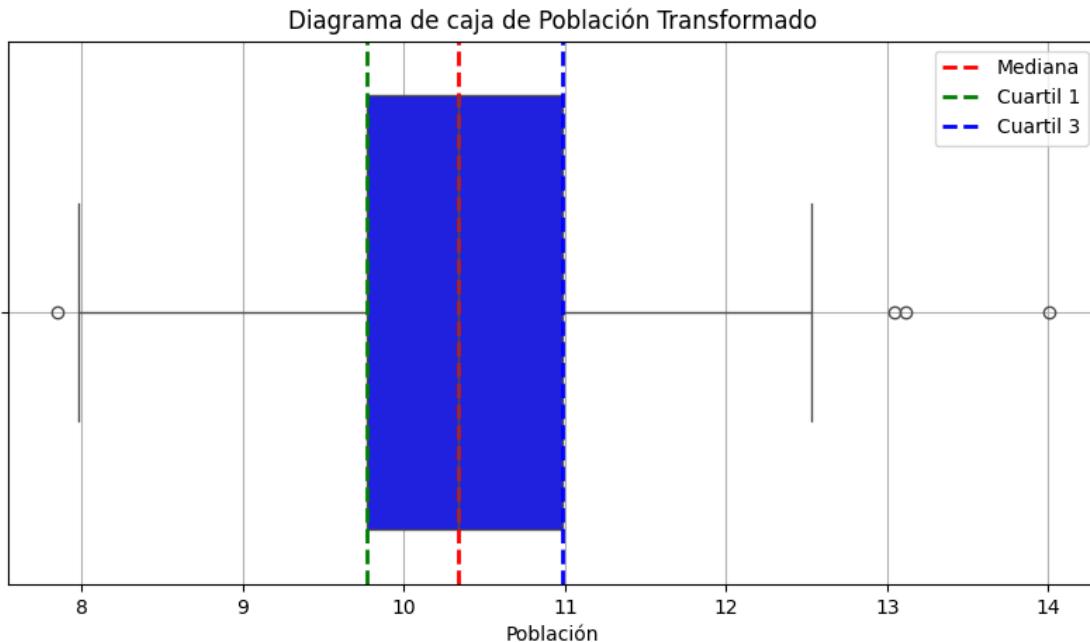


Outliers en population: [1205668, 121721, 494561, 276836, 147604, 464528, 165026, 145417, 129232, 122466, 148943, 224109, 248880, 188522, 163665, 166078, 123979, 136909, 196867]





Outliers en population_log: [14.002544328154372, 13.111425779363252, 13.0487771152901]



Análisis de Datos Cualitativos Diagrama de barras de los departamentos y municipios - El formato normalizado son la separación de los valores en el dataset y eliminando duplicaciones de entradas. - El formato desnormalizado son la unión y aparición de los valores en el dataset repitiéndose pero atendiendo distintos casos. - Tomar en cuenta que las gráficas se comportarán similar por ser sólo una multiplicación de conjuntos en sus datos. - Hay 339 registros en los municipios y 22 registros en los departamentos en el formato Normalizado. - Hay 424,128 registros en los municipios y departamentos de formato Desnormalizado. - Los municipios que más veces aparecieron fue Santa Barbara y La democracia tanto en el formato Normalizado y Desnormalizado. - Hay 22 departamentos en total.

```
univariate_analysis(dfFull, 'departament', width=15, height=5, title='Diagrama de Barras - Municipios Desnormalizada', color='skyblue')
```

Diagrama de barras de los municipios con 339 registros (Normalizada)

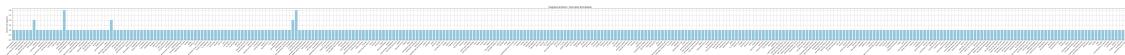


Diagrama de barras de los municipios aparecidos en casos de muertes con 424128 registros (Desnormalizada)

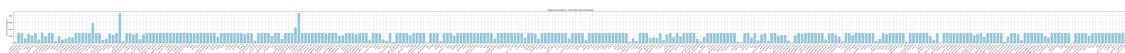


Diagrama de barras de los departamentos con 22 registros (Normalizada)

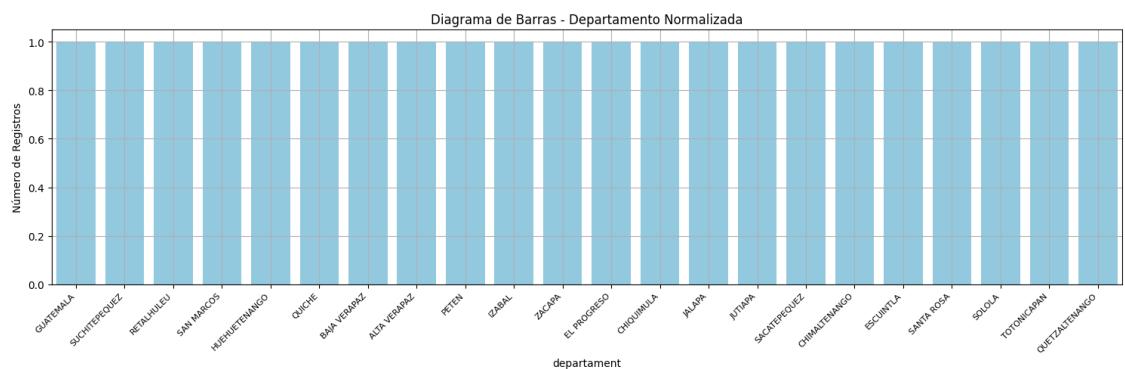
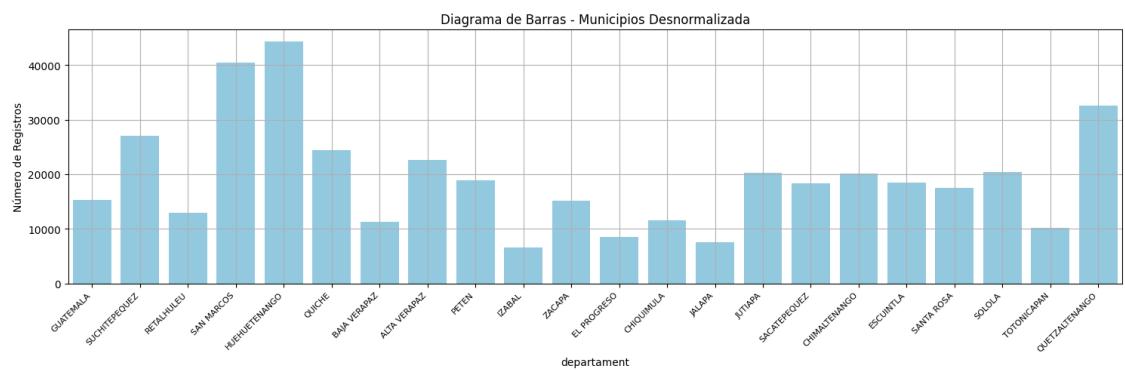


Diagrama de barras de los departamentos aparecidos en casos de muertes con 424128 registros (Desnormalizada)



0.3 EDA Multivariable

- Realizar gráficas de dispersión a los datos cuantitivos que son las variables cantidad de nuevas muertes, cantidad de muertes acumuladas y población de los municipios
- Realizar gráficas de barras, mapas de calor u otro tipo de gráfica para comparar las variables municipios y departamentos con las variables cuantitativas
- Se trabajara con la variable population_log en lugar de la variable population por cuestiones de facilitación de datos.

Análisis de los Datos Cuantitativos

- En el gráfico de dispersión tridimensional observamos que entre los rangos de 0-25 nuevos muertos se acumulaban en todos los datos de 3,000 a 5,000 muertos y que ocurrian en poblaciones de municipios desde las 2980 personas hasta las 162,754 personas
- En el gráfico de dispersión de Nuevas Muertes VS Muertes Acumuladas observamos que cuando ocurren muertes nuevas entre 0 a 40 ocupan mayor cantidad de muertos acumulados, donde en cada municipio morían alrededor de 0 a 4 personas por día. Mientras que cuando es muertes superiores a 40 se llegan a acumular peores muertos. Por lo que concluimos que la mayor cantidad de muertos se llego a alcanzar cuando se morían pocos al dia.
- Solo en un municipio que tenía 1,202,604 ocurrieron más de 20 muertos diarios.

```
[7]: def scatter_plots(df, x_column, y_column, hue_column, title):  
    """  
    Generate scatter plots for quantitative variables from a Pandas DataFrame.  
  
    Parameters:  
    - df: DataFrame  
    - x_column: str, nombre de la columna para el eje x  
    - y_column: str, nombre de la columna para el eje y  
    - hue_column: str, nombre de la columna para el parámetro 'hue' (color)  
    - title: str, título del gráfico  
    """  
    plt.figure(figsize=(10, 6))  
    plt.title(title)  
  
    # Usar scatter plot con 'hue' para distinguir diferentes valores en la  
    # columna 'hue'  
    sns.scatterplot(x=df[x_column], y=df[y_column], hue=df[hue_column],  
                    palette='coolwarm', edgecolor='w', s=30)  
  
    plt.xlabel(x_column)  
    plt.ylabel(y_column)  
    plt.legend(title=hue_column)  
    plt.grid(True)  
    plt.show()  
  
def scatter_plot_tridimensional(df, x_column, y_column, z_column, title):
```

```

"""
Generate a tridimensional scatter plot for quantitative variables from a
↳Pandas DataFrame.

Parameters:
- df: DataFrame
- x_column: str, nombre de la columna para el eje x
- y_column: str, nombre de la columna para el eje y
- z_column: str, nombre de la columna para el eje z
- title: str, título del gráfico
"""

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
colors = np.arange(len(df))

scatter = ax.scatter(df[x_column], df[y_column], df[z_column], c=colors, ↳
cmap='viridis', marker='o', s=50, linewidths=0.5)

ax.set_xlabel(x_column)
ax.set_ylabel(y_column)
ax.set_zlabel(z_column)

# Añadir barra de colores
cbar = plt.colorbar(scatter, ax=ax)
cbar.set_label('Índice de Color')

plt.title(title)
plt.show()

print(f"Diagrama tridimensional de Dispersion entre Nuevas Muertes vs Muertes ↳
↳Acumuladas vs Poblacion de Municipios")
scatter_plot_tridimensional(dfFull, 'new_deaths', 'cumulative_deaths', ↳
'population_log', 'Gráfico de Dispersion Tridimensional')
print(f"Diagrama de dispersión de Nuevas Muertes vs Muertes Acumuladas")
scatter_plots(dfFull, 'new_deaths', 'cumulative_deaths', 'deceases', 'Gráfico ↳
de Dispersion Nuevas Muertes vs Muertes Acumuladas')
print(f"Diagrama de dispersión de Nuevas Muertes vs Población Municipios")
scatter_plots(dfFull, 'new_deaths', 'population_log', 'deceases', 'Gráfico de ↳
Dispersion de Nuevas Muertes vs Población Municipios')
print(f"Diagrama de dispersión de Muertes Acumuladas vs Población Municipios")
scatter_plots(dfFull, 'cumulative_deaths', 'population_log', 'deceases', ↳
'Gráfico de Dispersion de Muertes Acumuladas vs Población Municipios')

```

Diagrama tridimensional de Dispersion entre Nuevas Muertes vs Muertes Acumuladas
vs Poblacion de Municipios

Gráfico de Dispersion Tridimensional

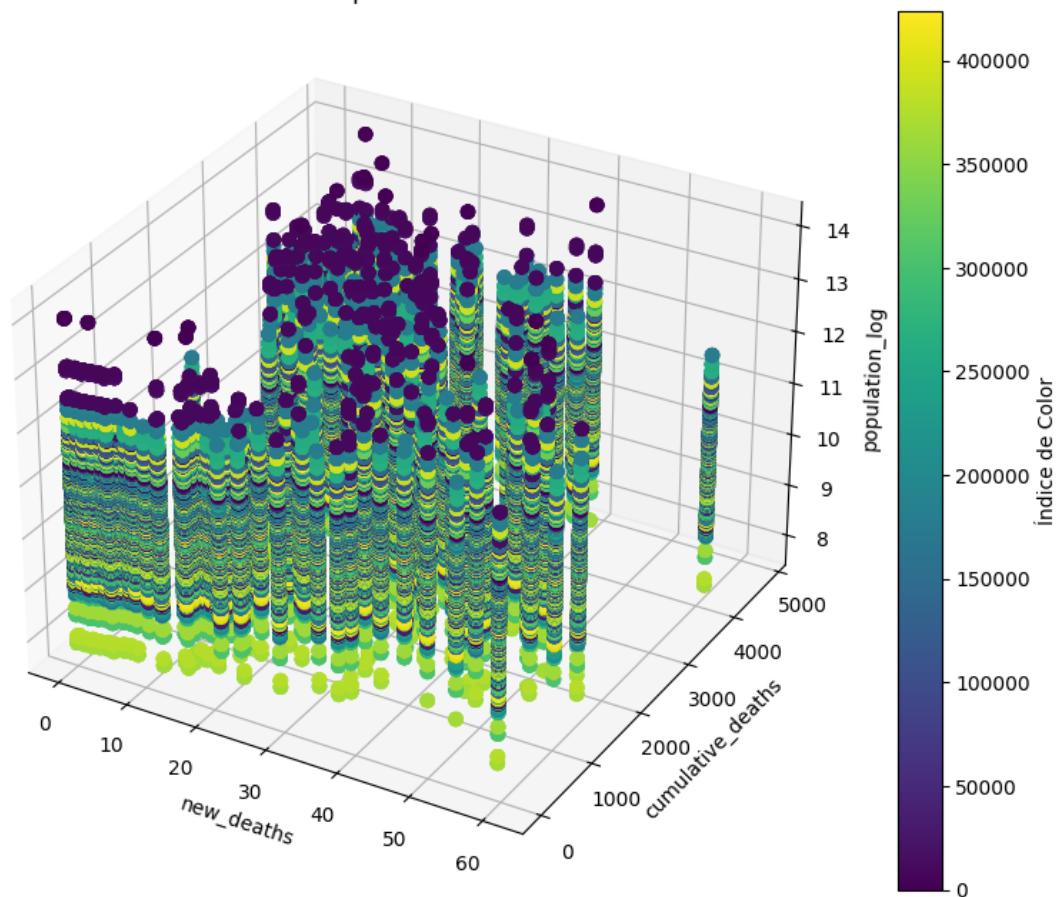


Diagrama de dispersión de Nuevas Muertes vs Muertes Acumuladas

```
C:\Users\Camran1234\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_  
qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-  
packages\IPython\core\pylabtools.py:152: UserWarning: Creating legend with  
loc="best" can be slow with large amounts of data.  
fig.canvas.print_figure(bytes_io, **kw)
```

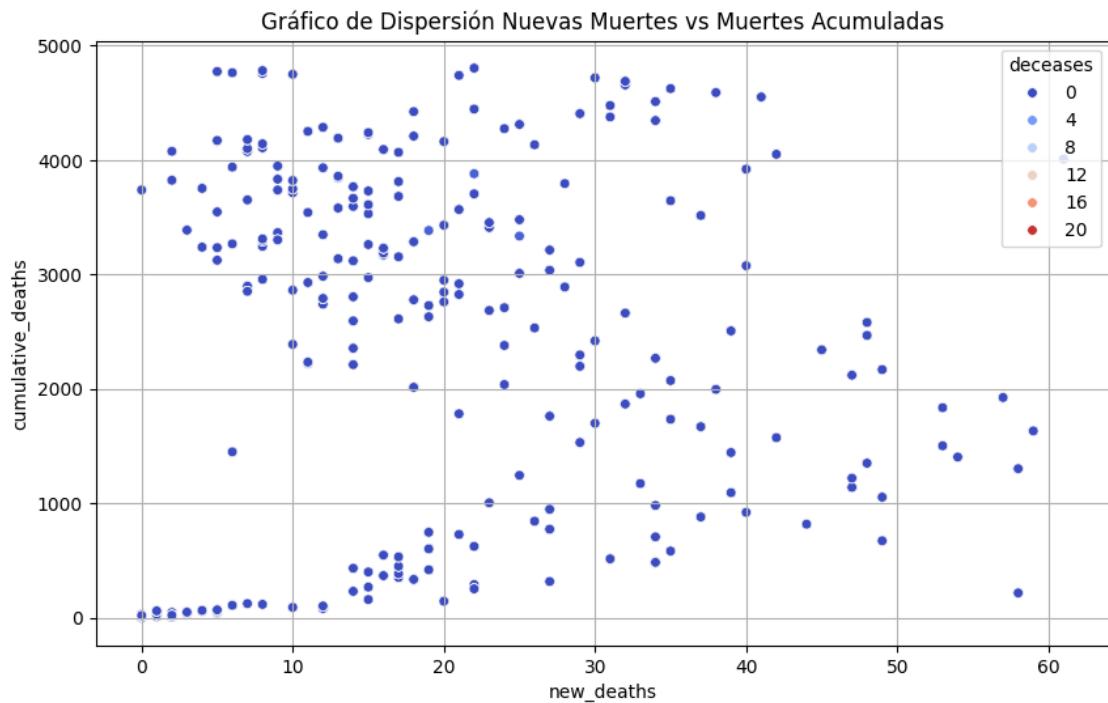


Diagrama de dispersión de Nuevas Muertes vs Población Municipios

```
C:\Users\Camran1234\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_ 
qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\IPython\core\pylabtools.py:152: UserWarning: Creating legend with
loc="best" can be slow with large amounts of data.
fig.canvas.print_figure(bytes_io, **kw)
```

Gráfico de Dispersion de Nuevas Muertes vs Población Municipios

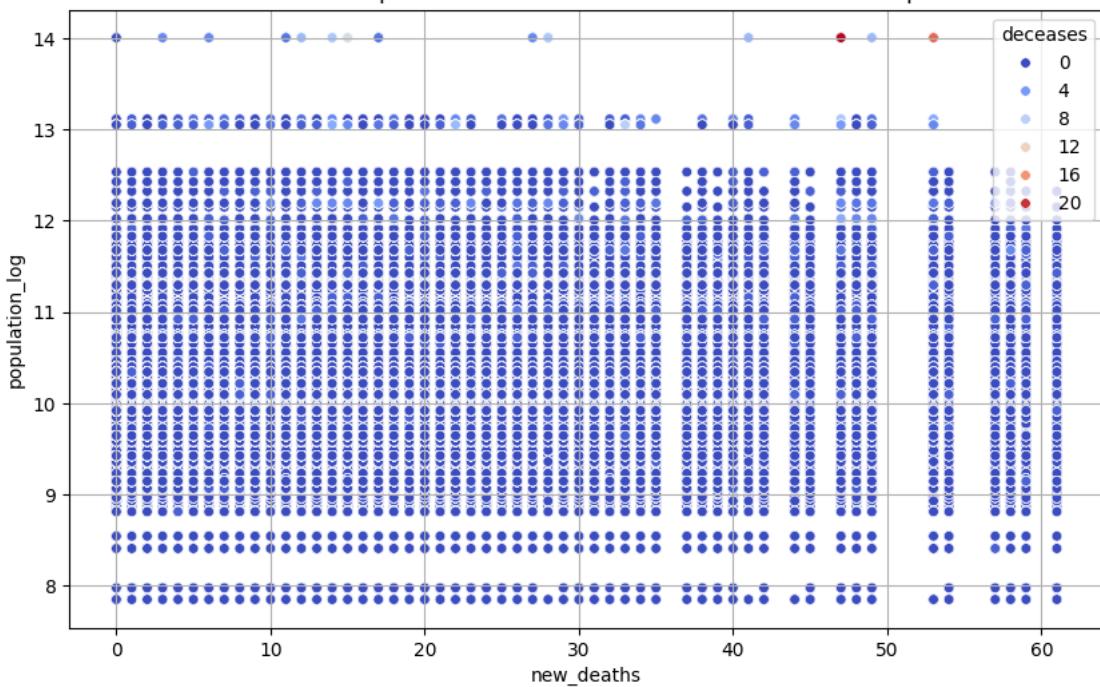


Diagrama de dispersión de Muertes Acumuladas vs Población Municipios

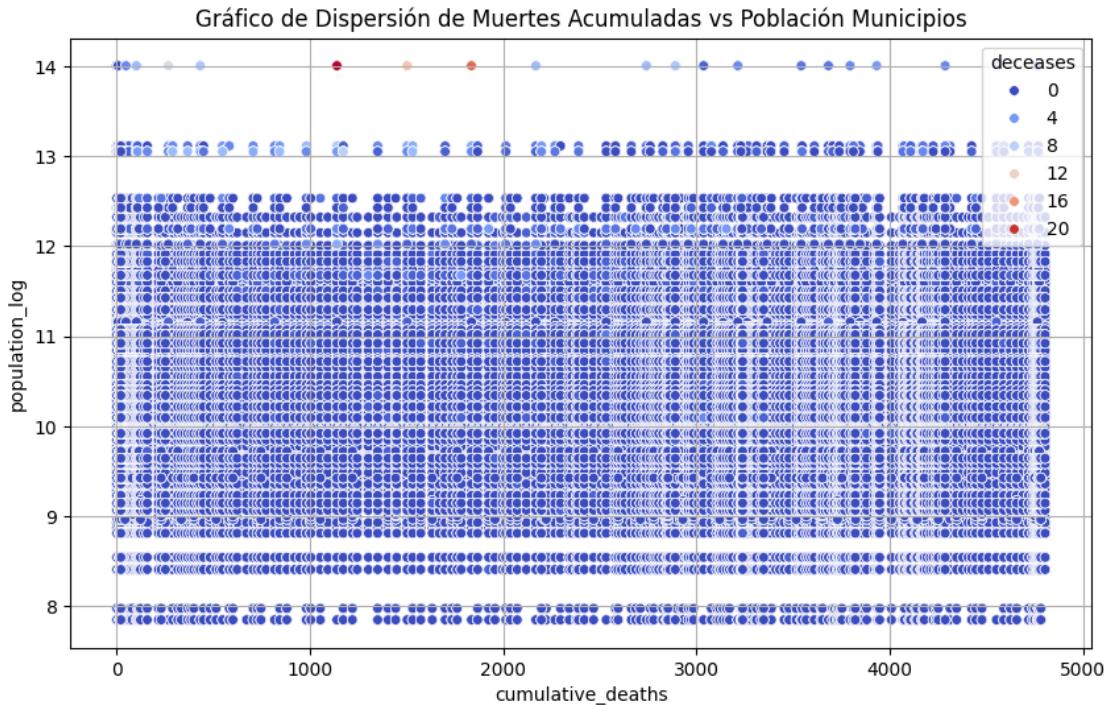
```
C:\Users\Camran1234\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_
```

```
qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
```

```
packages\IPython\core\pylabtools.py:152: UserWarning: Creating legend with
```

```
loc="best" can be slow with large amounts of data.
```

```
fig.canvas.print_figure(bytes_io, **kw)
```



Análisis Cualitativo Relacionar los datos cuantitativos (nuevas muertes, muertes acumuladas, población de los municipios) con los datos cualitativos (departamentos, municipios)

- Municipios vs cantidad de nuevas muertes
- Departamentos vs cantidad de nuevas muertes
- Municipios vs población
- Departamentos vs población
- Municipios vs cantidad de muertes acumuladas
- Departamentos vs cantidad de muertes acumuladas

Nota: Usar gráfica de barras, mapa de calor y alguna que otra gráfica para cada inciso

```
[8]: def bar_plot(df, x_column, y_column, width, height, hue_column, ↴
    ↴palette='coolwarm', title='Gráfico de Barras'):
    plt.figure(figsize=(width, height))
    plt.title(title)
    sns.barplot(x=x_column, y=y_column, hue=hue_column, data=df, ↴
    ↴palette=palette, linewidth=5)
    plt.xlabel(x_column)
    plt.ylabel(y_column)
    plt.legend(title=hue_column, loc='upper left')
    plt.xticks(rotation=45, ha='right', fontsize=8) # Rotar etiquetas y ↴
    ↴ajustar tamaño de fuente
    plt.tight_layout()
```

```

plt.show()

def heatmap(df, x_column, y_column, title='Mapa de Calor'):
    plt.figure(figsize=(10, 6))
    plt.title(title)
    sns.heatmap(df[[x_column, y_column]].corr(), annot=True, cmap='Spectral', u
    ↪ linewidths=.5)
    plt.show()

def create_boxplot(df, x_column, y_column, width, height, title='Diagrama de_
    ↪ Caja', color='skyblue'):
    """
    Create a boxplot from a DataFrame.

    Parameters:
    - df: DataFrame
    - x_column: str, nombre de la columna para el eje x
    - y_column: str, nombre de la columna para el eje y
    - title: str, título del gráfico
    - color: str, color de las cajas
    """
    plt.figure(figsize=(width, height))
    plt.title(title)

    sns.boxplot(x=x_column, y=y_column, data=df, color=color, palette='Set3', u
    ↪ showfliers=True)

    plt.xlabel(x_column)
    plt.ylabel(y_column)
    plt.xticks(rotation=45, ha='right', fontsize=8) # Rotar etiquetas y
    ↪ ajustar tamaño de fuente
    plt.tight_layout()
    plt.show()

def create_area_chart(df, x_column, y_column, width, height, color='skyblue', u
    ↪ title='Gráfico de Área'):
    """
    Create an area chart from a DataFrame.

    Parameters:
    - df: DataFrame
    - x_column: str, nombre de la columna para el eje x
    - y_column: str, nombre de la columna para el eje y
    - title: str, título del gráfico
    """
    plt.figure(figsize=(width, height))

```

```

plt.title(title)

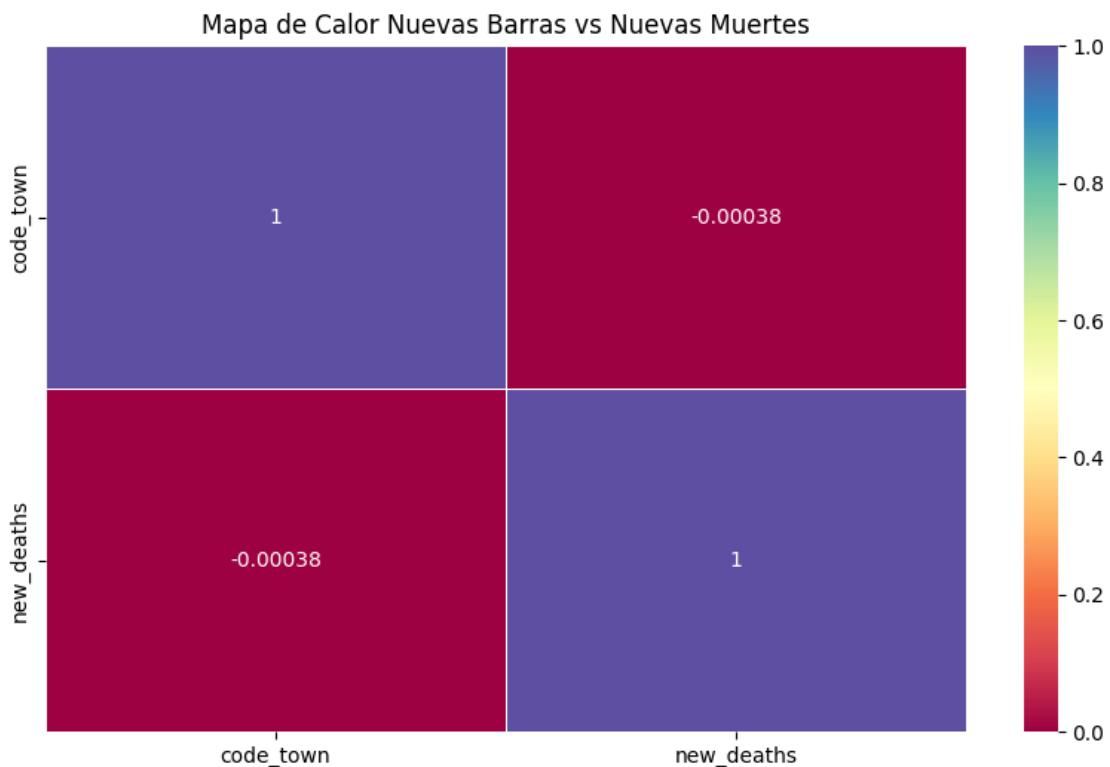
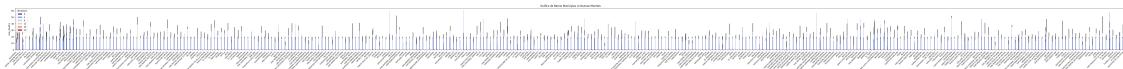
plt.fill_between(df[x_column], df[y_column], color=color, alpha=0.6)
plt.xlabel(x_column)
plt.ylabel(y_column)
plt.xticks(rotation=45, ha='right', fontsize=8) # Rotar etiquetas y
→ajustar tamaño de fuente
plt.tight_layout()
plt.show()

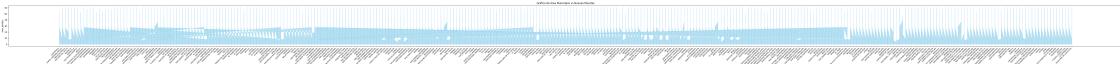
```

Análisis de Municipios VS Muertes Nuevas

```
[9]: print(f"Analisis de Municipios vs cantidad de nuevas muertes")
bar_plot(dfFull, 'town', 'new_deaths', 75, 5, 'deceases', title='Gráfico de
→Barras Municipios vs Nuevas Muertes')
heatmap(dfFull, 'code_town', 'new_deaths', title='Mapa de Calor Nuevas Barras
→vs Nuevas Muertes')
create_area_chart(dfFull, 'town', 'new_deaths', 75, 5, title='Gráfico de Área
→Municipios vs Nuevas Muertes')
```

Analisis de Municipios vs cantidad de nuevas muertes



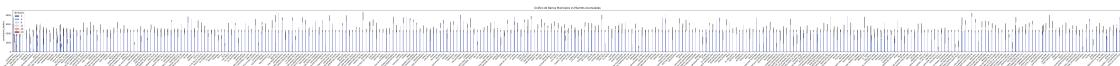


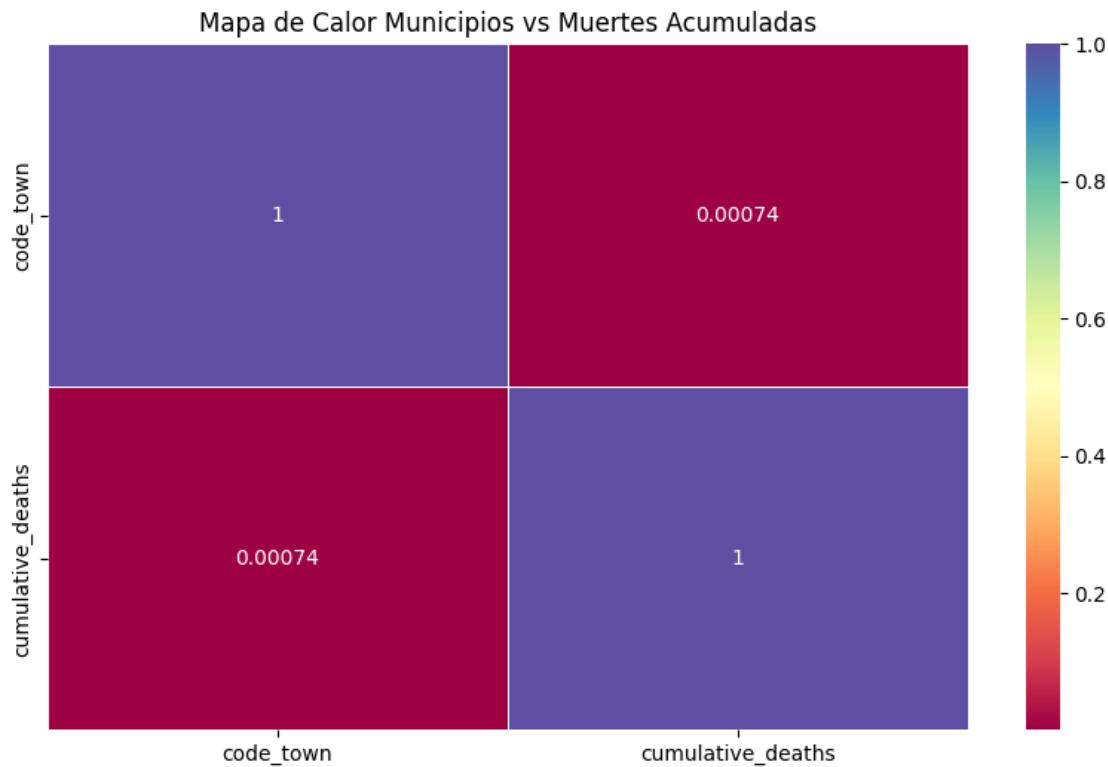
Análisis de Municipios VS Muertes Acumuladas

- El municipio que tuvo mayor muertes al día y muertes acumuladas por covid fue la capital.
- Si existe una correlación entre las nuevas muertes y los municipios
- Solo fueron 72 municipios que estuvieron debajo del promedio de las muertes nuevas, siendo el municipio de casillas el que menos muertes nuevas llegó a reportar.
- De las muertes acumuladas San Miguel Sigáoeila ha sido el único con tener una anomalía en la media de muertes acumuladas.

```
[10]: print(f"Analisis de Municipios vs cantidad de muertes acumuladas")
bar_plot(dfFull, 'town', 'cumulative_deaths', 75, 5, 'deceases', title='Gráfico de Barras Municipios vs Muertes Acumuladas')
heatmap(dfFull, 'code_town', 'cumulative_deaths', title='Mapa de Calor Municipios vs Muertes Acumuladas')
create_boxplot(dfFull, 'town', 'cumulative_deaths', 75, 5, title='Diagrama de Caja Municipios vs Muertes Acumuladas')
create_area_chart(dfFull, 'town', 'cumulative_deaths', 75, 5, title='Gráfico de Área Municipios vs Muertes Acumuladas')
```

Analisis de Municipios vs cantidad de muertes acumuladas

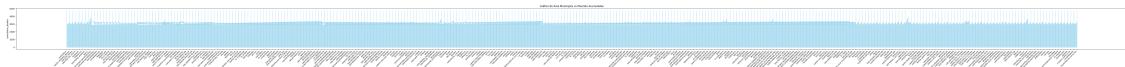
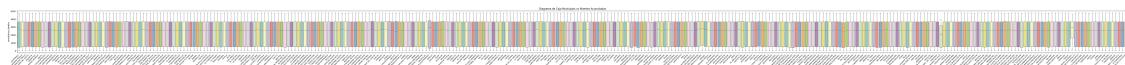




```
C:\Users\Camran1234\AppData\Local\Temp\ipykernel_8268\3842029547.py:32:
FutureWarning:
```

```
Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.
```

```
sns.boxplot(x=x_column, y=y_column, data=df, color=color, palette='Set3',
showfliers=True)
```

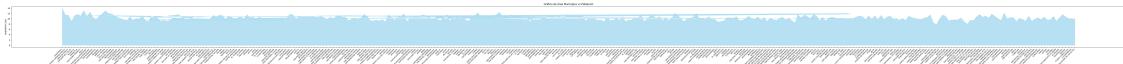
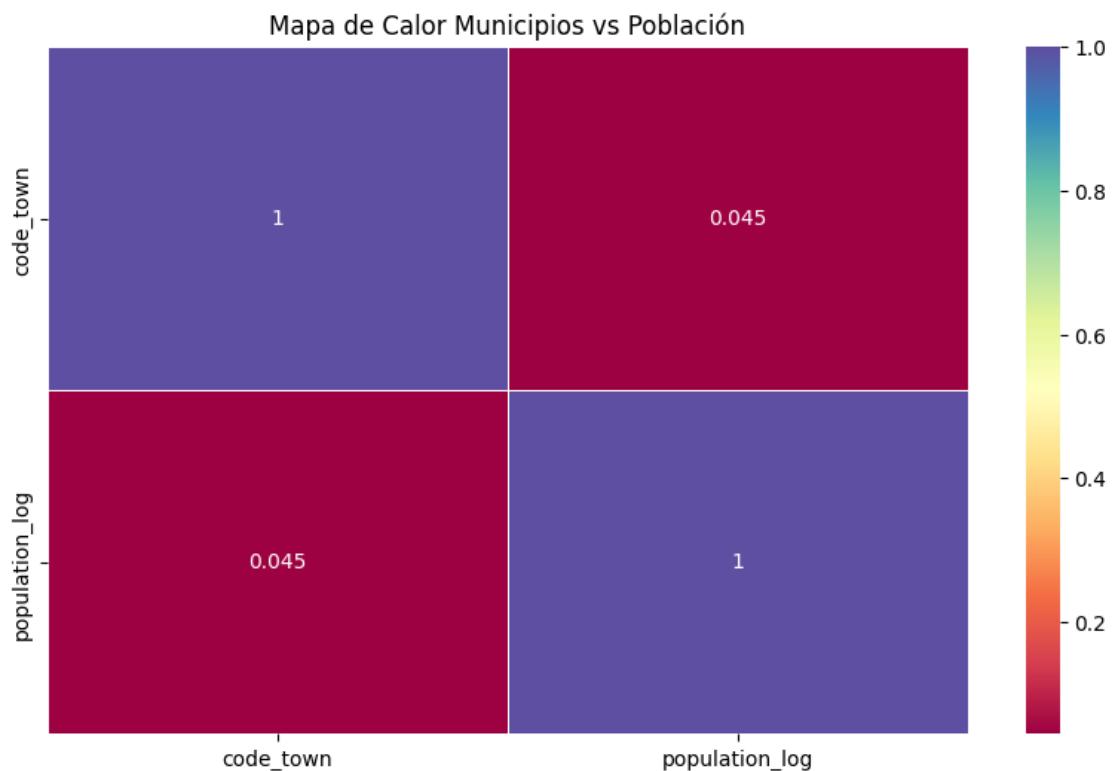
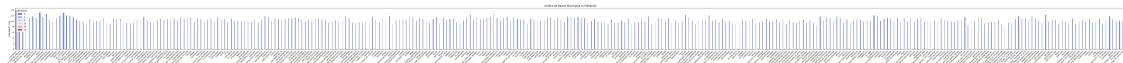


Análisis de Municipios VS Población de Municipios

- La población de la mayoría de los municipios es similar.

```
[11]: print(f"Analisis de Municipios vs poblacion de municipios")
bar_plot(dfFull, 'town', 'population_log', 75, 5, 'deceases', title='Gráfico de Barra Municipios vs Población')
heatmap(dfFull, 'code_town', 'population_log', title='Mapa de Calor Municipios vs Población')
create_area_chart(dfFull, 'town', 'population_log', 75, 5, title='Gráfico de Área Municipios vs Población')
```

Analisis de Municipios vs poblacion de municipios

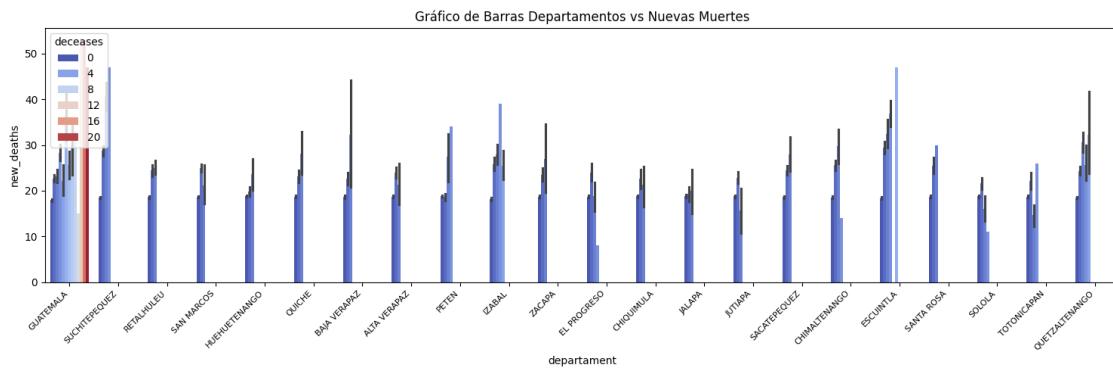


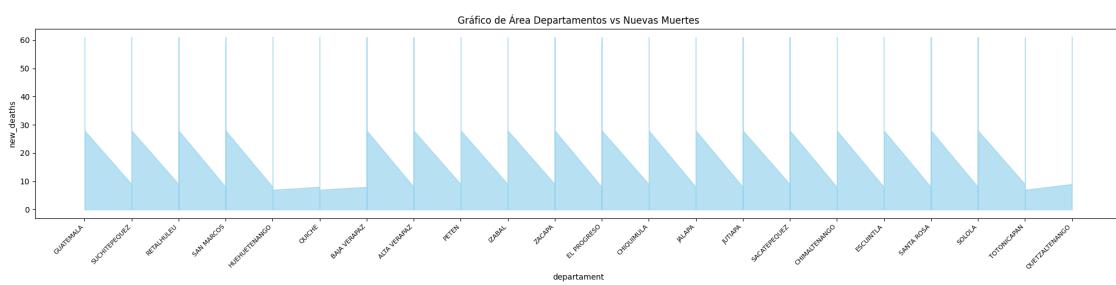
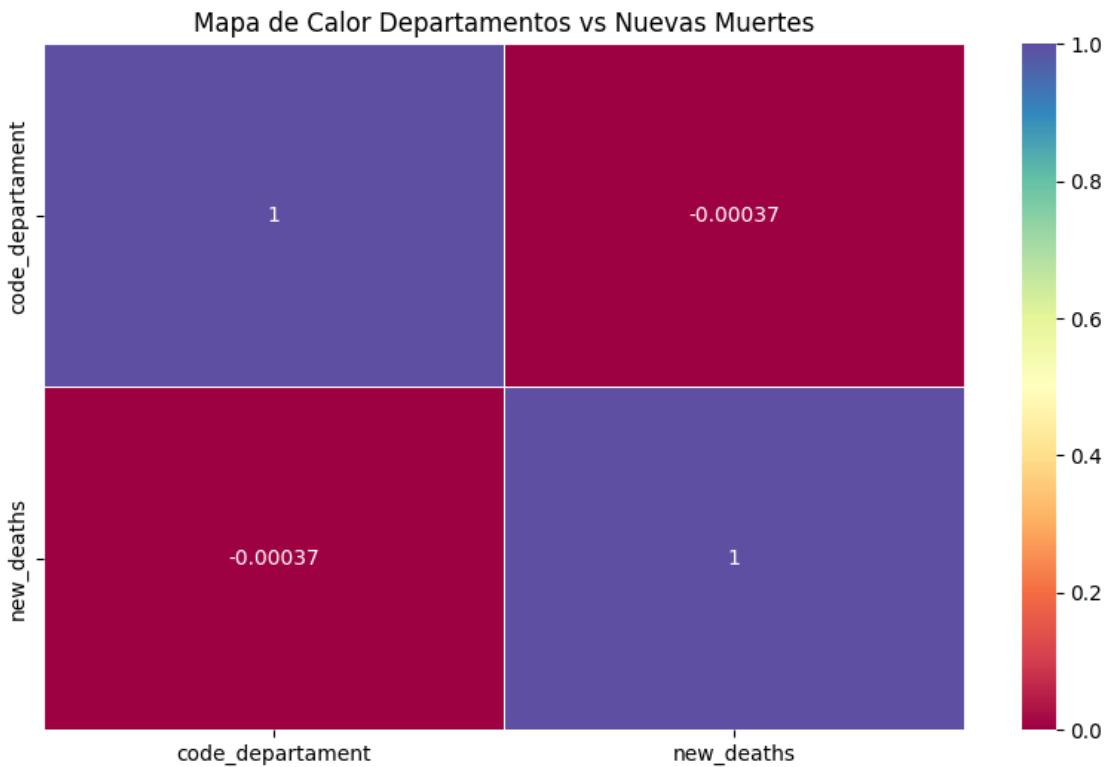
Análisis de Departamentos VS Muertes Nuevas

- Guatemala fue el departamento que mayor cantidad de muertes nuevas reportaba al día, mientras que Solola fue el que menor cantidad de muertes nuevas reportaba.

```
[12]: print(f"Analisis de Departamentos vs cantidad de nuevas muertes")
bar_plot(dfFull, 'departament', 'new_deaths', 15, 5, 'deceases', title='Gráfico de Barras Departamentos vs Nuevas Muertes')
heatmap(dfFull, 'code_departament', 'new_deaths', title='Mapa de Calor de Departamentos vs Nuevas Muertes')
create_area_chart(dfFull, 'departament', 'new_deaths', 20, 5, title='Gráfico de Área Departamentos vs Nuevas Muertes')
create_boxplot(dfFull, 'departament', 'new_deaths', 25, 5, title='Diagrama de Caja Departamentos vs Nuevas Muertes')
```

Analisis de Departamentos vs cantidad de nuevas muertes

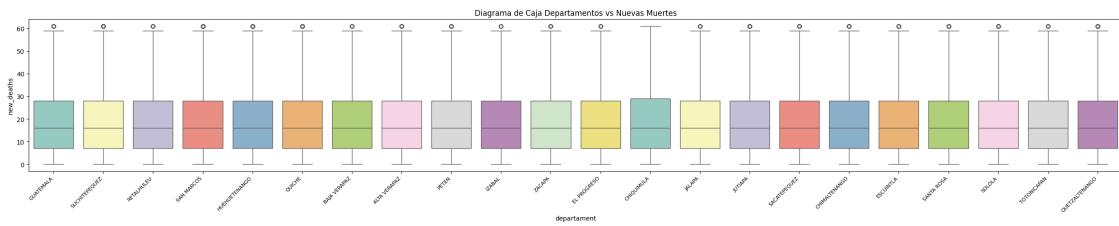




```
C:\Users\Camran1234\AppData\Local\Temp\ipykernel_8268\3842029547.py:32:
FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.boxplot(x=x_column, y=y_column, data=df, color=color, palette='Set3',
showfliers=True)
```



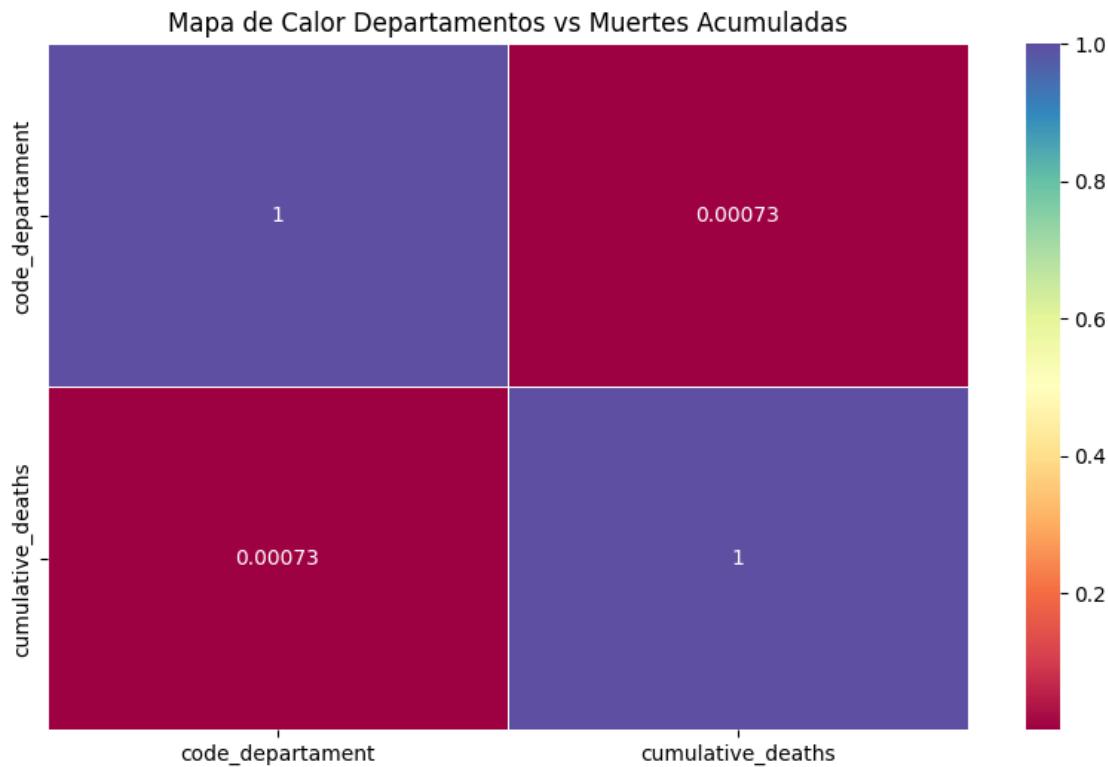
Análisis de Departamentos VS Muertes Acumuladas

- El departamento de Guatemala fue el que mayor cantidad de muertos acumulados tuvo y también el que mayor cantidad de muertos tenía al día en sus municipios.
- Aunque en totonicapan si se llegó a tener aproximadamente 3,500 muertos acumulados reportados en una fecha.
- La tasas de muertes acumuladas de Totonicapan fue muy exponencial y de pocos días mientras que la de Guatemala se acumulaba la misma cantidad de muertos al día.

```
[63]: print(f"Analisis de Departamentos vs cantidad de muertes acumuladas")
bar_plot(dfFull, 'departament', 'cumulative_deaths', 20, 5, 'deceases',
         title='Gráfico de Barras Departamentos vs Muertes Acumuladas')
heatmap(dfFull, 'code_departament', 'cumulative_deaths', title='Mapa de Calor',
        subtitle='Departamentos vs Muertes Acumuladas')
create_boxplot(dfFull, 'departament', 'cumulative_deaths', 30, 5,
               title='Diagrama de Caja Departamentos vs Muertes Acumuladas')
```

Analisis de Departamentos vs cantidad de muertes acumuladas

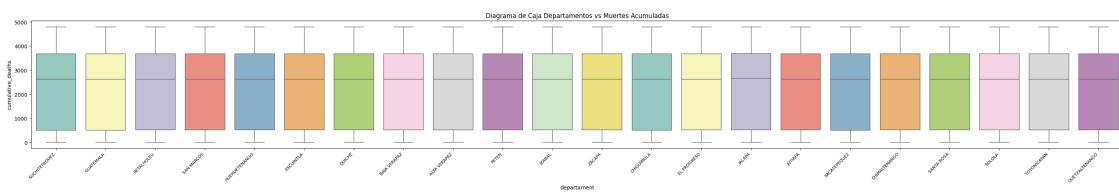




```
C:\Users\Camran1234\AppData\Local\Temp\ipykernel_5188\3842029547.py:32:
FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.boxplot(x=x_column, y=y_column, data=df, color=color, palette='Set3',
showfliers=True)
```



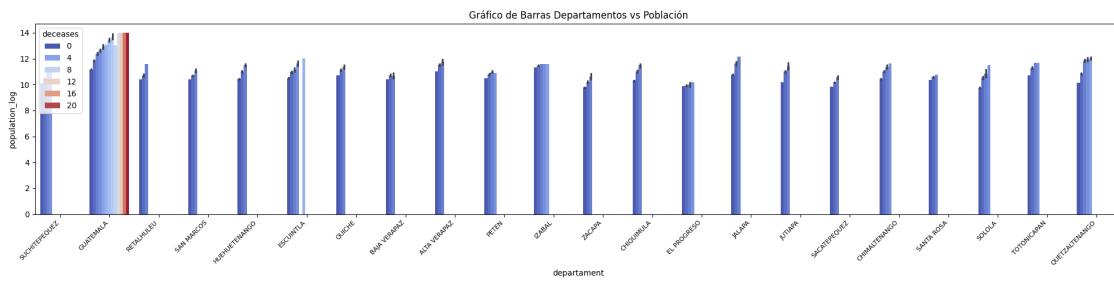
0.3.1 Análisis de Departamentos VS Población de Municipios

- El departamento de Izabal tienen casi la misma cantidad de pobladores en sus municipios, aproximadamente de 98,715 personas.

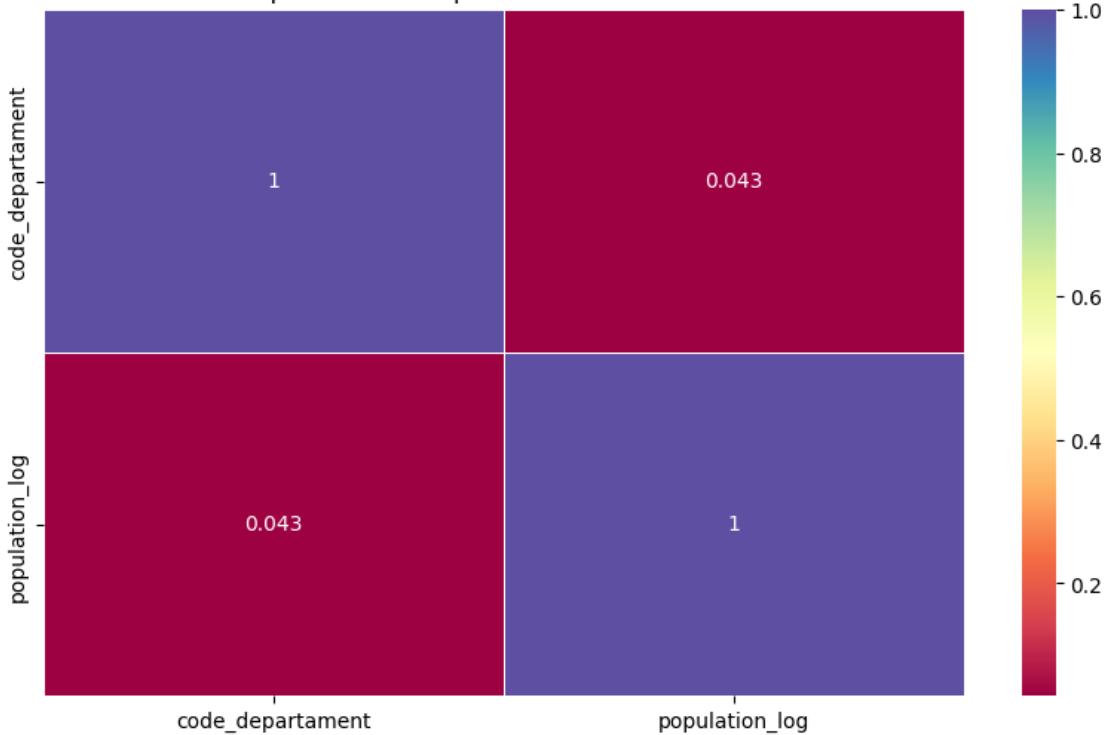
- Mientras que el Departamento de Guatemala es el que más cantidad variada de pobladores tiene en sus municipios llegando a los 1,202,604 personas y la menor de 8,103 personas.

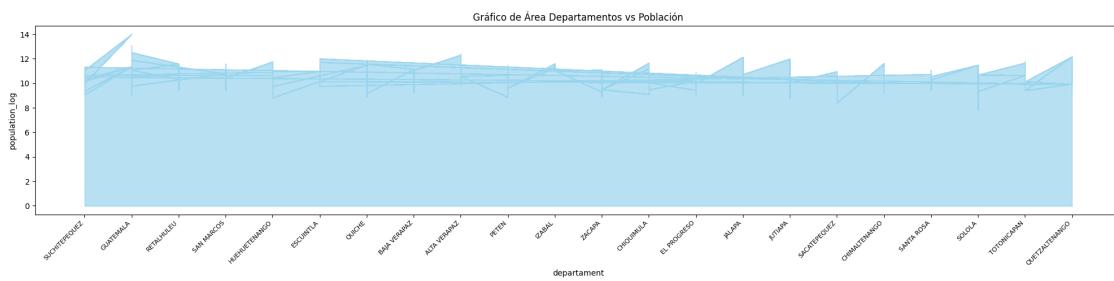
```
[64]: print(f"Analisis de Departamentos vs poblacion de municipios")
bar_plot(dfFull, 'departament', 'population_log', 20, 5, 'deceases', □
    ↪title='Gráfico de Barras Departamentos vs Población')
heatmap(dfFull, 'code_departament', 'population_log', title='Mapa de Calor □
    ↪Departamentos vs Población')
create_area_chart(dfFull, 'departament', 'population_log', 20, 5, □
    ↪title='Gráfico de Área Departamentos vs Población')
create_boxplot(dfFull, 'departament', 'population_log', 20, 5, title='Diagrama □
    ↪de Caja Departamentos vs Población')
```

Analisis de Departamentos vs poblacion de municipios



Mapa de Calor Departamentos vs Población

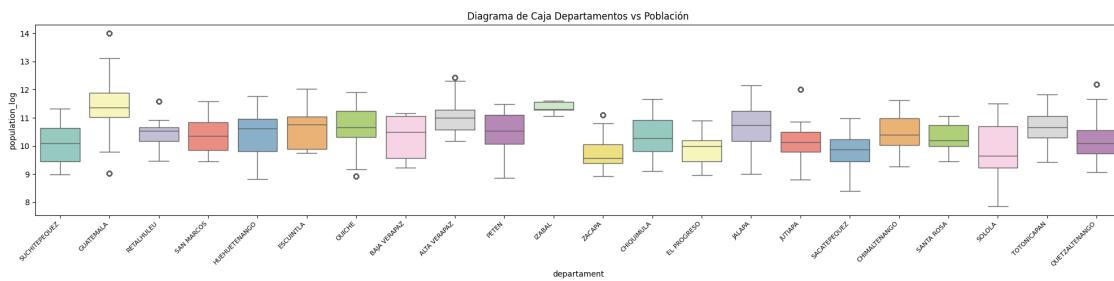




```
C:\Users\Camran1234\AppData\Local\Temp\ipykernel_5188\3842029547.py:32:
FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.boxplot(x=x_column, y=y_column, data=df, color=color, palette='Set3',
showfliers=True)
```



0.4 Conclusiones

- El departamento de Guatemala fue el más afectado y también el que mayor cantidad de muertos tenia al día en sus municipios. Siendo el municipio de Guatemala el que más afectado terminó.
- El departamento de Totonicapan fue el que más cantidad de muertos tuvo en pocos días, llegando a picos de 3500 muertos acumulados
- La mediana de muertes nuevas al día era de 16, su promedio de 18 y el valor máximo de muertos fue de 61, siguiéndole 58 y 59. La mayoría en la capital y el promedio de las muertes acumuladas en todo el país fue de 2,275.