

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

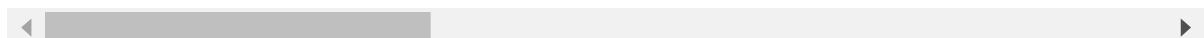
```
In [3]: data = pd.read_csv(r'D:/prodigy/task1 world population/API_SP.POP.TOTL_DS2_en_csv_v
```

```
In [4]: data
```

Out[4]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962
0	Aruba	ABW	Population, total	SP.POP.TOTL	54608.0	55811.0	56682.0
1	Africa Eastern and Southern	AFE	Population, total	SP.POP.TOTL	130692579.0	134169237.0	137835590.0
2	Afghanistan	AFG	Population, total	SP.POP.TOTL	8622466.0	8790140.0	8969047.0
3	Africa Western and Central	AFW	Population, total	SP.POP.TOTL	97256290.0	99314028.0	101445032.0
4	Angola	AGO	Population, total	SP.POP.TOTL	5357195.0	5441333.0	5521400.0
...
261	Kosovo	XKX	Population, total	SP.POP.TOTL	990150.0	1014211.0	1038618.0
262	Yemen, Rep.	YEM	Population, total	SP.POP.TOTL	5542459.0	5646668.0	5753386.0
263	South Africa	ZAF	Population, total	SP.POP.TOTL	16520441.0	16989464.0	17503133.0
264	Zambia	ZMB	Population, total	SP.POP.TOTL	3119430.0	3219451.0	3323427.0
265	Zimbabwe	ZWE	Population, total	SP.POP.TOTL	3806310.0	3925952.0	4049778.0

266 rows × 68 columns

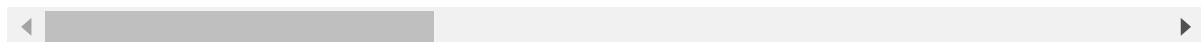


```
In [5]: data.head(5)
```

Out[5]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962
0	Aruba	ABW	Population, total	SP.POP.TOTL	54608.0	55811.0	56682.0
1	Africa Eastern and Southern	AFE	Population, total	SP.POP.TOTL	130692579.0	134169237.0	137835590.0
2	Afghanistan	AFG	Population, total	SP.POP.TOTL	8622466.0	8790140.0	8969047.0
3	Africa Western and Central	AFW	Population, total	SP.POP.TOTL	97256290.0	99314028.0	101445032.0
4	Angola	AGO	Population, total	SP.POP.TOTL	5357195.0	5441333.0	5521400.0

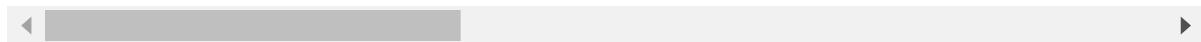
5 rows × 68 columns

In [6]: `data.tail()`

Out[6]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962
261	Kosovo	XKX	Population, total	SP.POP.TOTL	990150.0	1014211.0	1038618.0
262	Yemen, Rep.	YEM	Population, total	SP.POP.TOTL	5542459.0	5646668.0	5753386.0
263	South Africa	ZAF	Population, total	SP.POP.TOTL	16520441.0	16989464.0	17503133.0
264	Zambia	ZMB	Population, total	SP.POP.TOTL	3119430.0	3219451.0	3323427.0
265	Zimbabwe	ZWE	Population, total	SP.POP.TOTL	3806310.0	3925952.0	4049778.0

5 rows × 68 columns

In [7]: `data.shape`

Out[7]: (266, 68)

In [8]: `data.columns`

```
Out[8]: Index(['Country Name', 'Country Code', 'Indicator Name', 'Indicator Code',
       '1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968',
       '1969', '1970', '1971', '1972', '1973', '1974', '1975', '1976', '1977',
       '1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
       '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
       '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004',
       '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013',
       '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021', '2022',
       '2023'],
      dtype='object')
```

```
In [9]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 266 entries, 0 to 265
Data columns (total 68 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Country Name     266 non-null    object  
 1   Country Code     266 non-null    object  
 2   Indicator Name   266 non-null    object  
 3   Indicator Code   266 non-null    object  
 4   1960              264 non-null    float64 
 5   1961              264 non-null    float64 
 6   1962              264 non-null    float64 
 7   1963              264 non-null    float64 
 8   1964              264 non-null    float64 
 9   1965              264 non-null    float64 
 10  1966              264 non-null    float64 
 11  1967              264 non-null    float64 
 12  1968              264 non-null    float64 
 13  1969              264 non-null    float64 
 14  1970              264 non-null    float64 
 15  1971              264 non-null    float64 
 16  1972              264 non-null    float64 
 17  1973              264 non-null    float64 
 18  1974              264 non-null    float64 
 19  1975              264 non-null    float64 
 20  1976              264 non-null    float64 
 21  1977              264 non-null    float64 
 22  1978              264 non-null    float64 
 23  1979              264 non-null    float64 
 24  1980              264 non-null    float64 
 25  1981              264 non-null    float64 
 26  1982              264 non-null    float64 
 27  1983              264 non-null    float64 
 28  1984              264 non-null    float64 
 29  1985              264 non-null    float64 
 30  1986              264 non-null    float64 
 31  1987              264 non-null    float64 
 32  1988              264 non-null    float64 
 33  1989              264 non-null    float64 
 34  1990              265 non-null    float64 
 35  1991              265 non-null    float64 
 36  1992              265 non-null    float64 
 37  1993              265 non-null    float64 
 38  1994              265 non-null    float64 
 39  1995              265 non-null    float64 
 40  1996              265 non-null    float64 
 41  1997              265 non-null    float64 
 42  1998              265 non-null    float64 
 43  1999              265 non-null    float64 
 44  2000              265 non-null    float64 
 45  2001              265 non-null    float64 
 46  2002              265 non-null    float64 
 47  2003              265 non-null    float64 
 48  2004              265 non-null    float64 
 49  2005              265 non-null    float64 
 50  2006              265 non-null    float64
```

```

51 2007           265 non-null   float64
52 2008           265 non-null   float64
53 2009           265 non-null   float64
54 2010           265 non-null   float64
55 2011           265 non-null   float64
56 2012           265 non-null   float64
57 2013           265 non-null   float64
58 2014           265 non-null   float64
59 2015           265 non-null   float64
60 2016           265 non-null   float64
61 2017           265 non-null   float64
62 2018           265 non-null   float64
63 2019           265 non-null   float64
64 2020           265 non-null   float64
65 2021           265 non-null   float64
66 2022           265 non-null   float64
67 2023           265 non-null   float64
dtypes: float64(64), object(4)
memory usage: 141.4+ KB

```

In [10]: `data.describe()`

	1960	1961	1962	1963	1964	1965
count	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02
mean	1.157939e+08	1.173869e+08	1.195401e+08	1.222050e+08	1.248922e+08	1.276182e+08
std	3.639920e+08	3.684672e+08	3.751049e+08	3.837174e+08	3.923714e+08	4.011556e+08
min	2.646000e+03	2.888000e+03	3.171000e+03	3.481000e+03	3.811000e+03	4.161000e+03
25%	5.132212e+05	5.231345e+05	5.337595e+05	5.449288e+05	5.566630e+05	5.651150e+05
50%	3.708088e+06	3.816540e+06	3.931214e+06	4.033994e+06	4.112910e+06	4.194930e+06
75%	2.670606e+07	2.748694e+07	2.830289e+07	2.914708e+07	3.001684e+07	3.084892e+07
max	3.031517e+09	3.072470e+09	3.126894e+09	3.193470e+09	3.260480e+09	3.328243e+09

8 rows × 64 columns

◀ ▶

In [11]: `data.dtypes`

```
Out[11]: Country Name      object
          Country Code     object
          Indicator Name    object
          Indicator Code    object
          1960              float64
                      ...
          2019              float64
          2020              float64
          2021              float64
          2022              float64
          2023              float64
Length: 68, dtype: object
```

```
In [12]: data.duplicated().sum()
```

```
Out[12]: np.int64(0)
```

```
In [13]: data.isna().sum().any()
```

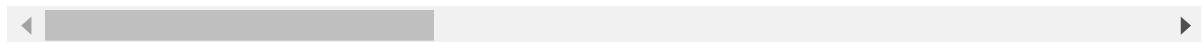
```
Out[13]: np.True_
```

```
In [14]: data=data.fillna(method='ffill')
data.head()
```

C:\Users\chomo\AppData\Local\Temp\ipykernel_5372\935320328.py:1: FutureWarning: Data Frame.fillna with 'method' is deprecated and will raise in a future version. Use obj.bfill() or obj.bfill() instead.
data=data.fillna(method='ffill')

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962
0	Aruba	ABW	Population, total	SP.POP.TOTL	54608.0	55811.0	56682.0
1	Africa Eastern and Southern	AFE	Population, total	SP.POP.TOTL	130692579.0	134169237.0	137835590.0
2	Afghanistan	AFG	Population, total	SP.POP.TOTL	8622466.0	8790140.0	8969047.0
3	Africa Western and Central	AFW	Population, total	SP.POP.TOTL	97256290.0	99314028.0	101445032.0
4	Angola	AGO	Population, total	SP.POP.TOTL	5357195.0	5441333.0	5521400.0

5 rows × 68 columns



```
In [15]: data.isna().sum().any()
```

```
Out[15]: np.False_
```

```
In [16]: data['Country Name'].unique()
```

```
Out[16]: array(['Aruba', 'Africa Eastern and Southern', 'Afghanistan',
   'Africa Western and Central', 'Angola', 'Albania', 'Andorra',
   'Arab World', 'United Arab Emirates', 'Argentina', 'Armenia',
   'American Samoa', 'Antigua and Barbuda', 'Australia', 'Austria',
   'Azerbaijan', 'Burundi', 'Belgium', 'Benin', 'Burkina Faso',
   'Bangladesh', 'Bulgaria', 'Bahrain', 'Bahamas, The',
   'Bosnia and Herzegovina', 'Belarus', 'Belize', 'Bermuda',
   'Bolivia', 'Brazil', 'Barbados', 'Brunei Darussalam', 'Bhutan',
   'Botswana', 'Central African Republic', 'Canada',
   'Central Europe and the Baltics', 'Switzerland', 'Channel Islands',
   'Chile', 'China', "Cote d'Ivoire", 'Cameroon', 'Congo, Dem. Rep.',
   'Congo, Rep.', 'Colombia', 'Comoros', 'Cabo Verde', 'Costa Rica',
   'Caribbean small states', 'Cuba', 'Curacao', 'Cayman Islands',
   'Cyprus', 'Czechia', 'Germany', 'Djibouti', 'Dominica', 'Denmark',
   'Dominican Republic', 'Algeria',
   'East Asia & Pacific (excluding high income)',
   'Early-demographic dividend', 'East Asia & Pacific',
   'Europe & Central Asia (excluding high income)',
   'Europe & Central Asia', 'Ecuador', 'Egypt, Arab Rep.',
   'Euro area', 'Eritrea', 'Spain', 'Estonia', 'Ethiopia',
   'European Union', 'Fragile and conflict affected situations',
   'Finland', 'Fiji', 'France', 'Faroe Islands',
   'Micronesia, Fed. Sts.', 'Gabon', 'United Kingdom', 'Georgia',
   'Ghana', 'Gibraltar', 'Guinea', 'Gambia, The', 'Guinea-Bissau',
   'Equatorial Guinea', 'Greece', 'Grenada', 'Greenland', 'Guatemala',
   'Guam', 'Guyana', 'High income', 'Hong Kong SAR, China',
   'Honduras', 'Heavily indebted poor countries (HIPC)', 'Croatia',
   'Haiti', 'Hungary', 'IBRD only', 'IDA & IBRD total', 'IDA total',
   'IDA blend', 'Indonesia', 'IDA only', 'Isle of Man', 'India',
   'Not classified', 'Ireland', 'Iran, Islamic Rep.', 'Iraq',
   'Iceland', 'Israel', 'Italy', 'Jamaica', 'Jordan', 'Japan',
   'Kazakhstan', 'Kenya', 'Kyrgyz Republic', 'Cambodia', 'Kiribati',
   'St. Kitts and Nevis', 'Korea, Rep.', 'Kuwait',
   'Latin America & Caribbean (excluding high income)', 'Lao PDR',
   'Lebanon', 'Liberia', 'Libya', 'St. Lucia',
   'Latin America & Caribbean',
   'Least developed countries: UN classification', 'Low income',
   'Liechtenstein', 'Sri Lanka', 'Lower middle income',
   'Low & middle income', 'Lesotho', 'Late-demographic dividend',
   'Lithuania', 'Luxembourg', 'Latvia', 'Macao SAR, China',
   'St. Martin (French part)', 'Morocco', 'Monaco', 'Moldova',
   'Madagascar', 'Maldives', 'Middle East & North Africa', 'Mexico',
   'Marshall Islands', 'Middle income', 'North Macedonia', 'Mali',
   'Malta', 'Myanmar',
   'Middle East & North Africa (excluding high income)', 'Montenegro',
   'Mongolia', 'Northern Mariana Islands', 'Mozambique', 'Mauritania',
   'Mauritius', 'Malawi', 'Malaysia', 'North America', 'Namibia',
   'New Caledonia', 'Niger', 'Nigeria', 'Nicaragua', 'Netherlands',
   'Norway', 'Nepal', 'Nauru', 'New Zealand', 'OECD members', 'Oman',
   'Other small states', 'Pakistan', 'Panama', 'Peru', 'Philippines',
   'Palau', 'Papua New Guinea', 'Poland', 'Pre-demographic dividend',
   'Puerto Rico', "Korea, Dem. People's Rep.", 'Portugal', 'Paraguay',
   'West Bank and Gaza', 'Pacific island small states',
   'Post-demographic dividend', 'French Polynesia', 'Qatar',
   'Romania', 'Russian Federation', 'Rwanda', 'South Asia',
   'Saudi Arabia', 'Sudan', 'Senegal', 'Singapore', 'Solomon Islands',
```

```
'Sierra Leone', 'El Salvador', 'San Marino', 'Somalia', 'Serbia',
'Sub-Saharan Africa (excluding high income)', 'South Sudan',
'Sub-Saharan Africa', 'Small states', 'Sao Tome and Principe',
'Suriname', 'Slovak Republic', 'Slovenia', 'Sweden', 'Eswatini',
'Sint Maarten (Dutch part)', 'Seychelles', 'Syrian Arab Republic',
'Turks and Caicos Islands', 'Chad',
'East Asia & Pacific (IDA & IBRD countries)',
'Europe & Central Asia (IDA & IBRD countries)', 'Togo', 'Thailand',
'Tajikistan', 'Turkmenistan',
'Latin America & the Caribbean (IDA & IBRD countries)',
'Timor-Leste', 'Middle East & North Africa (IDA & IBRD countries)',
'Tonga', 'South Asia (IDA & IBRD)',
'Sub-Saharan Africa (IDA & IBRD countries)', 'Trinidad and Tobago',
'Tunisia', 'Turkiye', 'Tuvalu', 'Tanzania', 'Uganda', 'Ukraine',
'Upper middle income', 'Uruguay', 'United States', 'Uzbekistan',
'St. Vincent and the Grenadines', 'Venezuela, RB',
'British Virgin Islands', 'Virgin Islands (U.S.)', 'Viet Nam',
'Vanuatu', 'World', 'Samoa', 'Kosovo', 'Yemen, Rep.',
'South Africa', 'Zambia', 'Zimbabwe'], dtype=object)
```

In [17]: `data['Country Code'].unique()`

Out[17]: array(['ABW', 'AFE', 'AFG', 'AFW', 'AGO', 'ALB', 'AND', 'ARB', 'ARE',
'ARG', 'ARM', 'ASM', 'ATG', 'AUS', 'AUT', 'AZE', 'BDI', 'BEL',
'BEN', 'BFA', 'BGD', 'BGR', 'BHR', 'BHS', 'BIH', 'BLR', 'BLZ',
'BMU', 'BOL', 'BRA', 'BRB', 'BRN', 'BTN', 'BWA', 'CAF', 'CAN',
'CEB', 'CHE', 'CHI', 'CHL', 'CHN', 'CIV', 'CMR', 'COD', 'COG',
'COL', 'COM', 'CPV', 'CRI', 'CSS', 'CUB', 'CUW', 'CYM', 'CYP',
'CZE', 'DEU', 'DEU', 'DMA', 'DNK', 'DOM', 'DZA', 'EAP', 'EAR',
'EAS', 'ECA', 'ECS', 'ECU', 'EGY', 'EMU', 'ERI', 'ESP', 'EST',
'ETH', 'EUU', 'FCS', 'FIN', 'FJI', 'FRA', 'FRO', 'FSM', 'GAB',
'GBR', 'GEO', 'GHA', 'GIB', 'GIN', 'GMB', 'GNB', 'GNQ', 'GRC',
'GRD', 'GRL', 'GTM', 'GUM', 'GUY', 'HIC', 'HKG', 'HND', 'HPC',
'HRV', 'HTI', 'HUN', 'IBD', 'IBT', 'IDA', 'IDB', 'IDN', 'IDX',
'IMN', 'IND', 'INX', 'IRL', 'IRN', 'IRQ', 'ISL', 'ISR', 'ITA',
'JAM', 'JOR', 'JPN', 'KAZ', 'KEN', 'KGZ', 'KHM', 'KIR', 'KNA',
'KOR', 'KWT', 'LAC', 'LAO', 'LBN', 'LBR', 'LBY', 'LCA', 'LCN',
'LDC', 'LIC', 'LIE', 'LKA', 'LMC', 'LMY', 'LSO', 'LTE', 'LTU',
'LUX', 'LVA', 'MAC', 'MAF', 'MAR', 'MCO', 'MDA', 'MDG', 'MDV',
'MEA', 'MEX', 'MHL', 'MIC', 'MKD', 'MLI', 'MLT', 'MMR', 'MNA',
'MNE', 'MNG', 'MNP', 'MOZ', 'MRT', 'MUS', 'MWI', 'MYS', 'NAC',
'NAM', 'NCL', 'NER', 'NGA', 'NIC', 'NLD', 'NOR', 'NPL', 'NRU',
'NZL', 'OED', 'OMN', 'OSS', 'PAK', 'PAN', 'PER', 'PHL', 'PLW',
'PNG', 'POL', 'PRE', 'PRI', 'PRK', 'PRT', 'PRY', 'PSE', 'PSS',
'PST', 'PYF', 'QAT', 'ROU', 'RUS', 'RWA', 'SAS', 'SAU', 'SDN',
'SEN', 'SGP', 'SLB', 'SLE', 'SLV', 'SMR', 'SOM', 'SRB', 'SSA',
'SSD', 'SSF', 'SST', 'STP', 'SUR', 'SVK', 'SVN', 'SWE', 'SWZ',
'SXM', 'SYC', 'SYR', 'TCA', 'TCD', 'TEA', 'TEC', 'TGO', 'THA',
'TJK', 'TKM', 'TLA', 'TLS', 'TMN', 'TON', 'TSA', 'TSS', 'TTO',
'TUN', 'TUR', 'TUV', 'TZA', 'UGA', 'UKR', 'UMC', 'URY', 'USA',
'UZB', 'VCT', 'VEN', 'VGB', 'VIR', 'VNM', 'VUT', 'WLD', 'WSM',
'XKK', 'YEM', 'ZAF', 'ZMB', 'ZWE'], dtype=object)

In [18]: `data['Indicator Name'].unique()`

```
Out[18]: array(['Population, total'], dtype=object)
```

```
In [19]: data['Indicator Code'].unique()
```

```
Out[19]: array(['SP.POP.TOTL'], dtype=object)
```

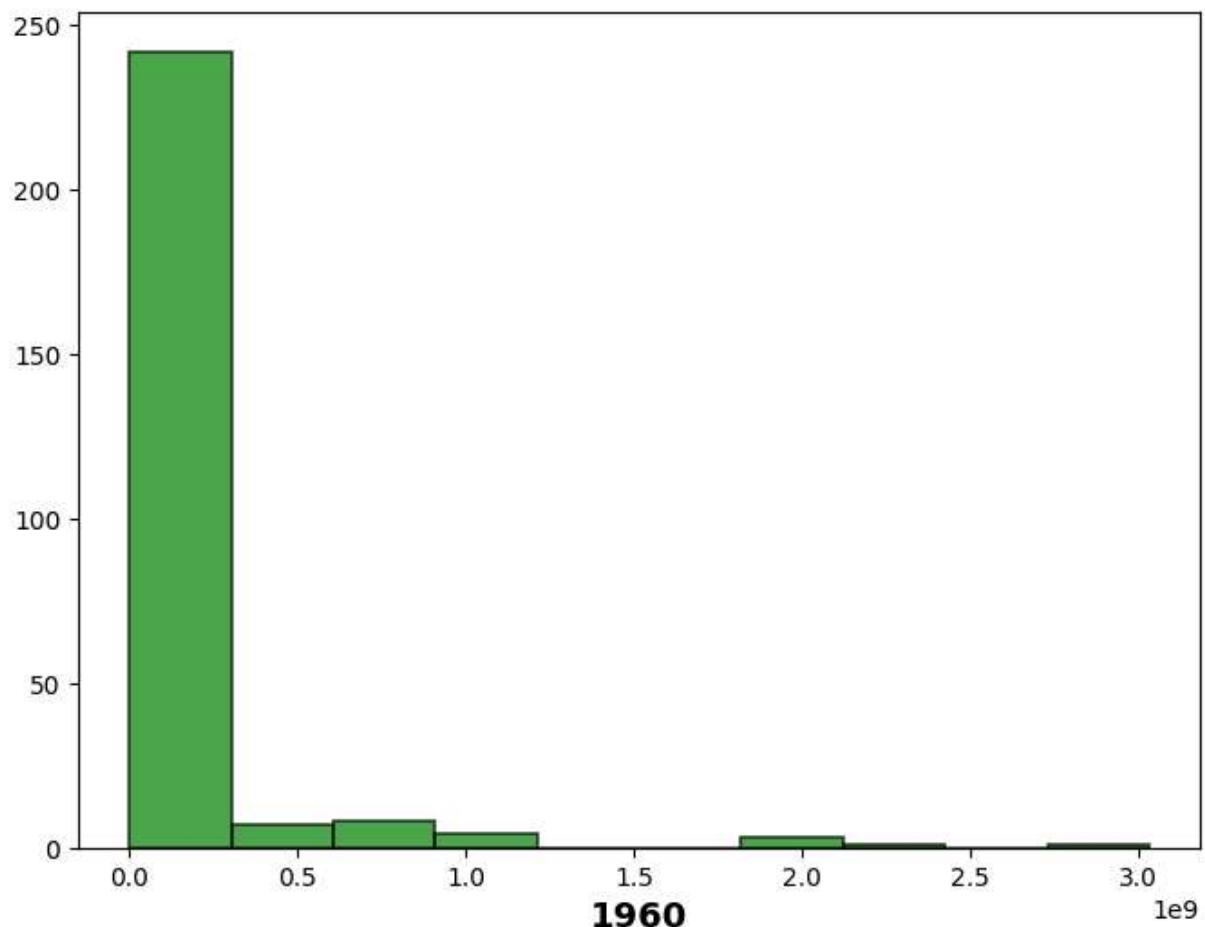
```
In [20]: data.drop(['Indicator Name','Indicator Code','Country Code'], axis=1, inplace=True,
```

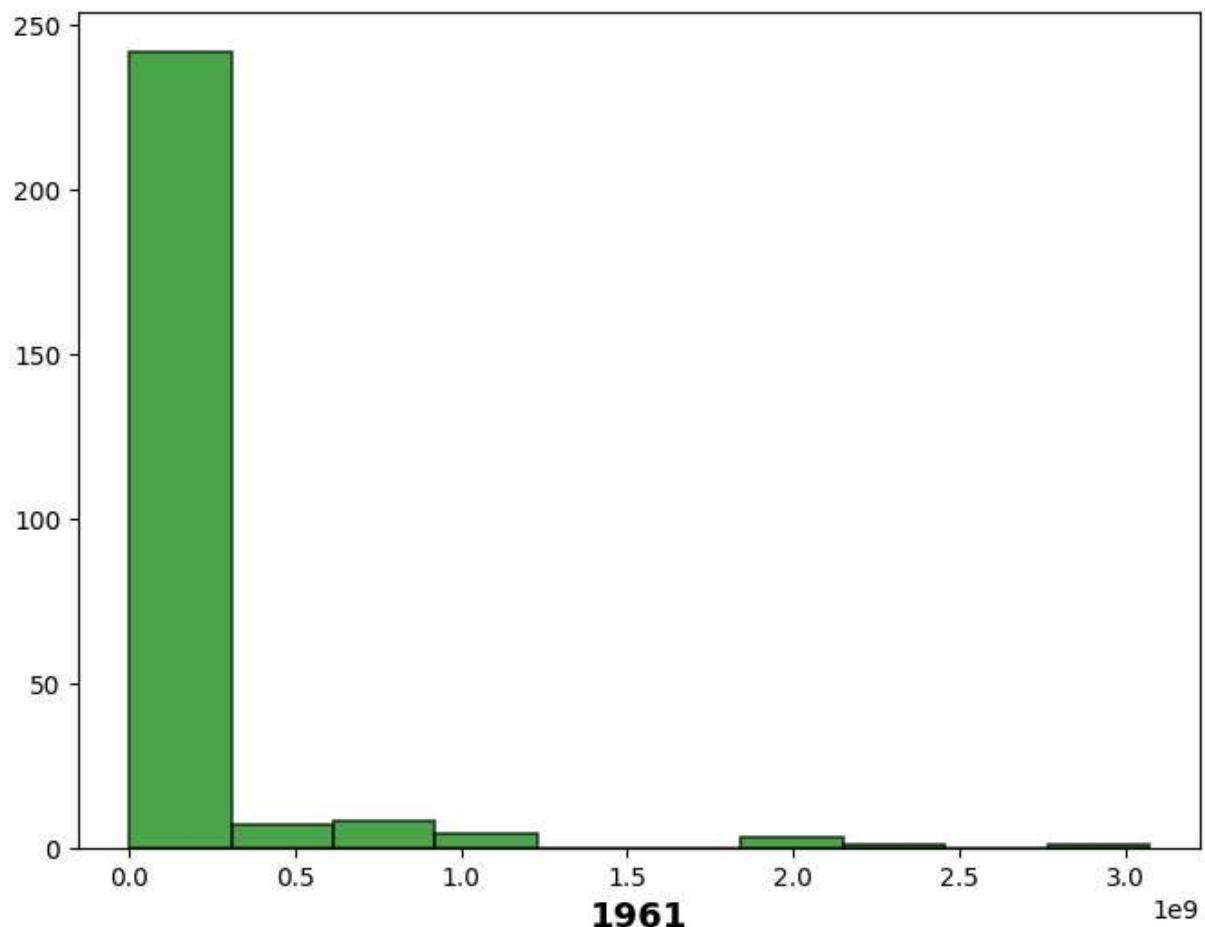
```
In [21]: data.columns
```

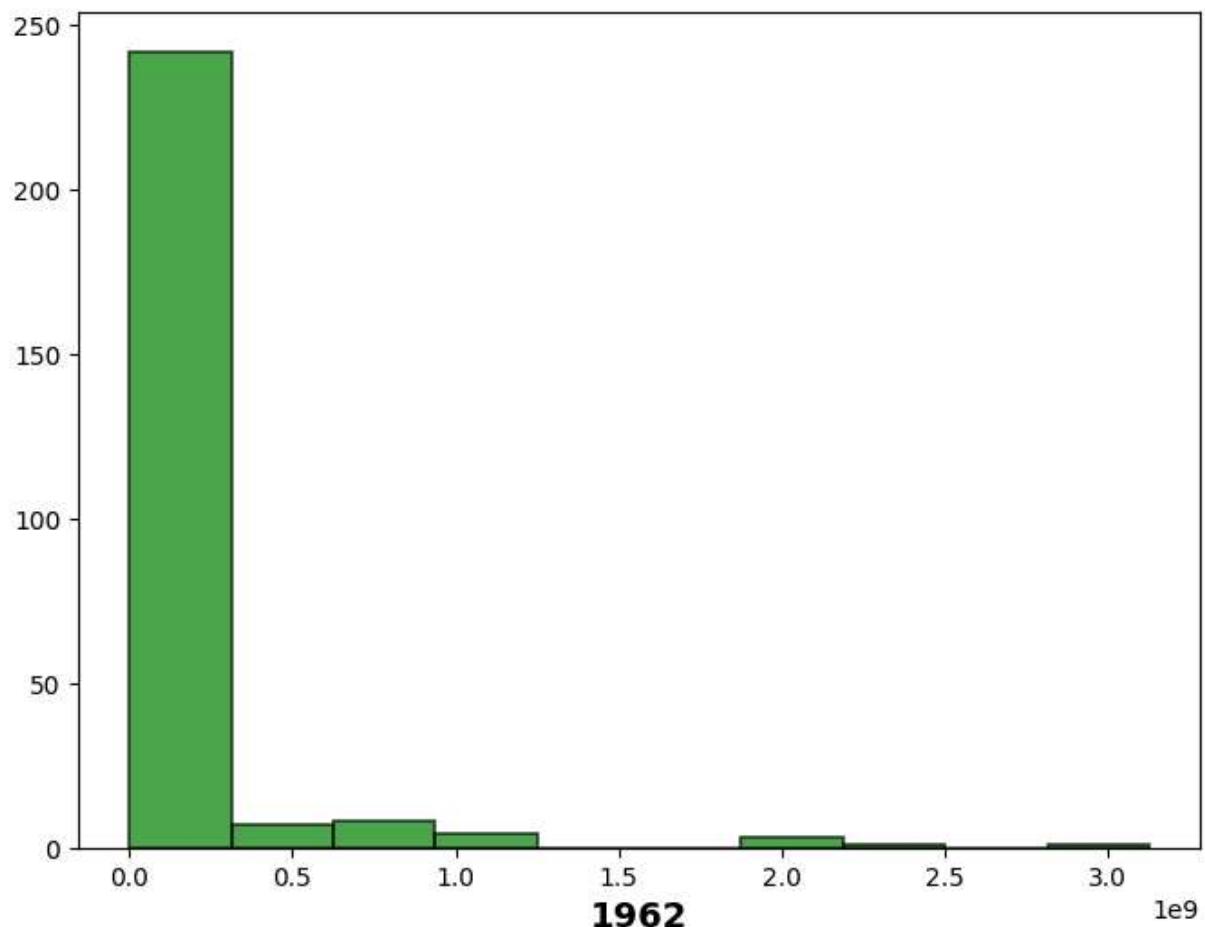
```
Out[21]: Index(['Country Name', '1960', '1961', '1962', '1963', '1964', '1965', '1966',
 '1967', '1968', '1969', '1970', '1971', '1972', '1973', '1974', '1975',
 '1976', '1977', '1978', '1979', '1980', '1981', '1982', '1983', '1984',
 '1985', '1986', '1987', '1988', '1989', '1990', '1991', '1992', '1993',
 '1994', '1995', '1996', '1997', '1998', '1999', '2000', '2001', '2002',
 '2003', '2004', '2005', '2006', '2007', '2008', '2009', '2010', '2011',
 '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019', '2020',
 '2021', '2022', '2023'],
 dtype='object')
```

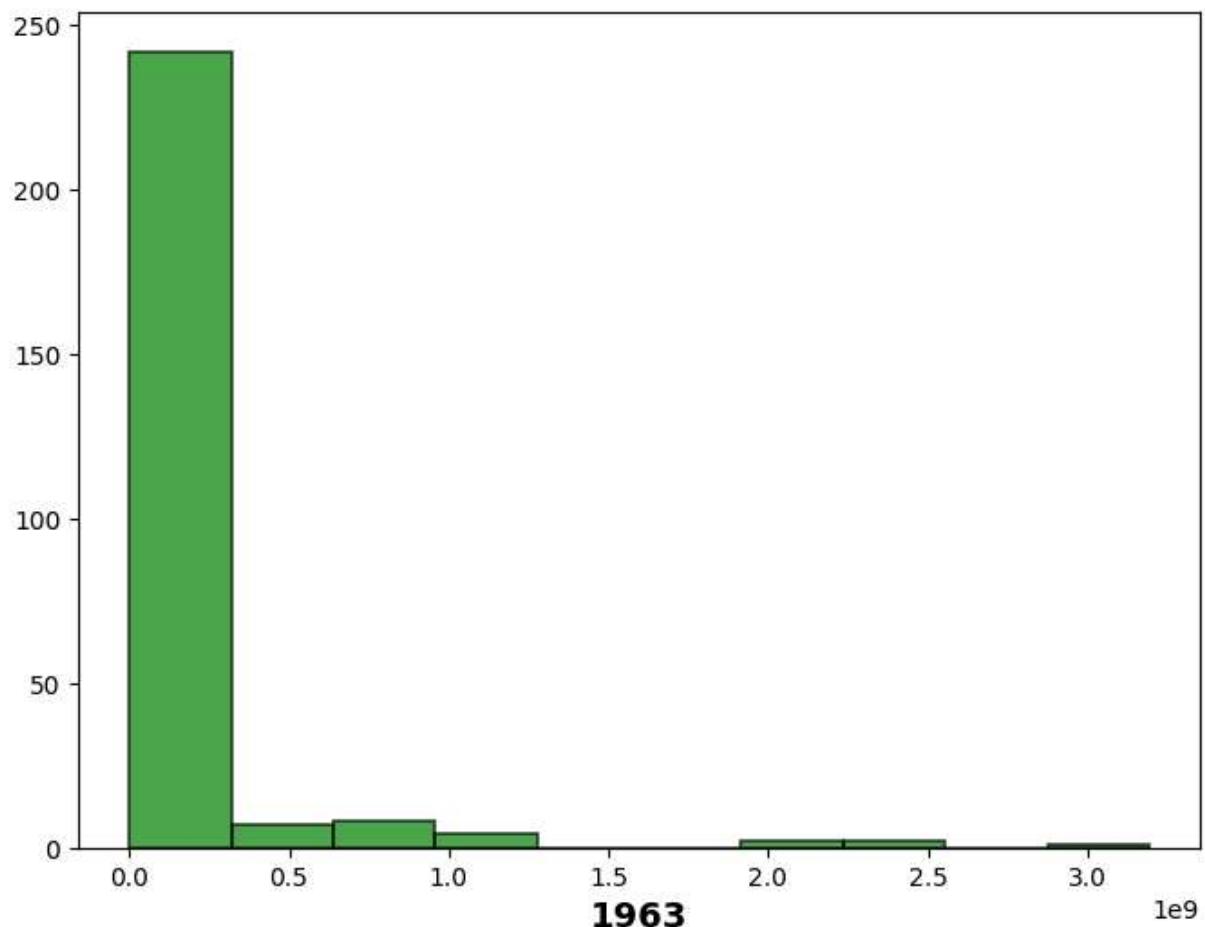
```
In [22]: cols =[ '1960', '1961', '1962', '1963', '1964', '1965', '1966',
 '1967', '1968', '1969', '1970', '1971', '1972', '1973', '1974', '1975',
 '1976', '1977', '1978', '1979', '1980', '1981', '1982', '1983', '1984',
 '1985', '1986', '1987', '1988', '1989', '1990', '1991', '1992', '1993',
 '1994', '1995', '1996', '1997', '1998', '1999', '2000', '2001', '2002',
 '2003', '2004', '2005', '2006', '2007', '2008', '2009', '2010', '2011',
 '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019', '2020',
 '2021', '2022', '2023']
```

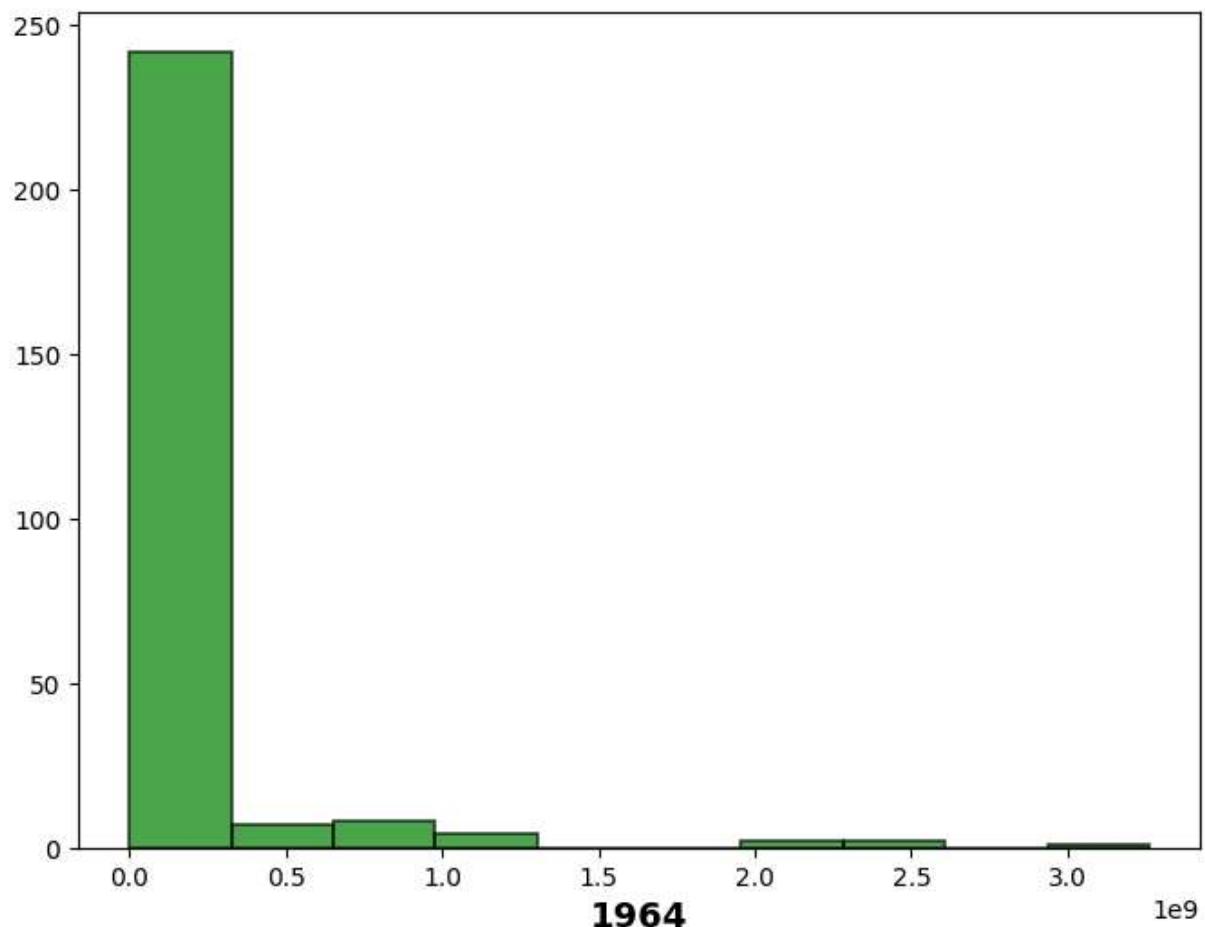
```
In [23]: for i in cols:
    plt.figure(figsize=(8,6))
    plt.hist(data[i], color='Green', bins=10, alpha=0.7, edgecolor='black', linewidth=1)
    plt.xlabel(i, fontsize=14, fontweight='bold')
    plt.show()
```

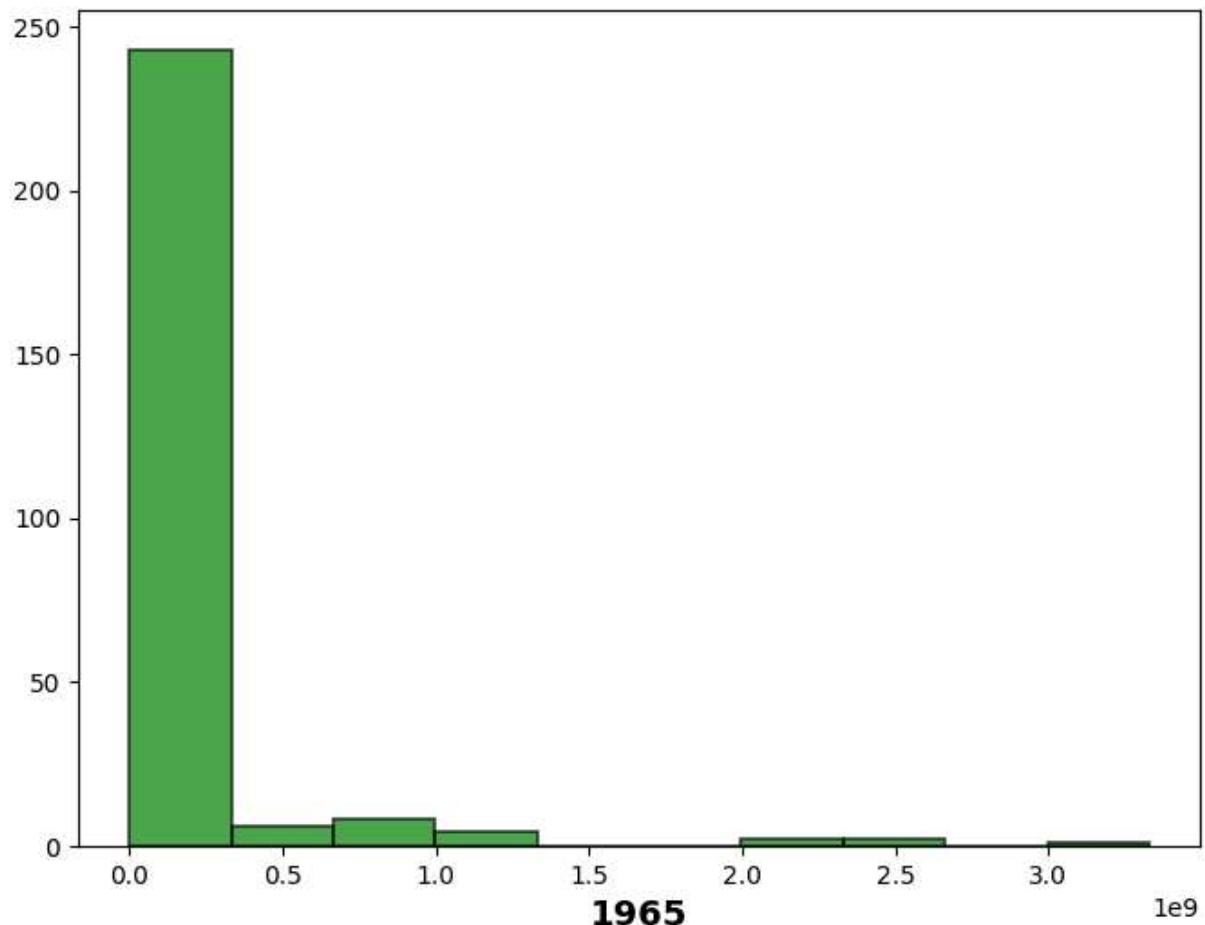


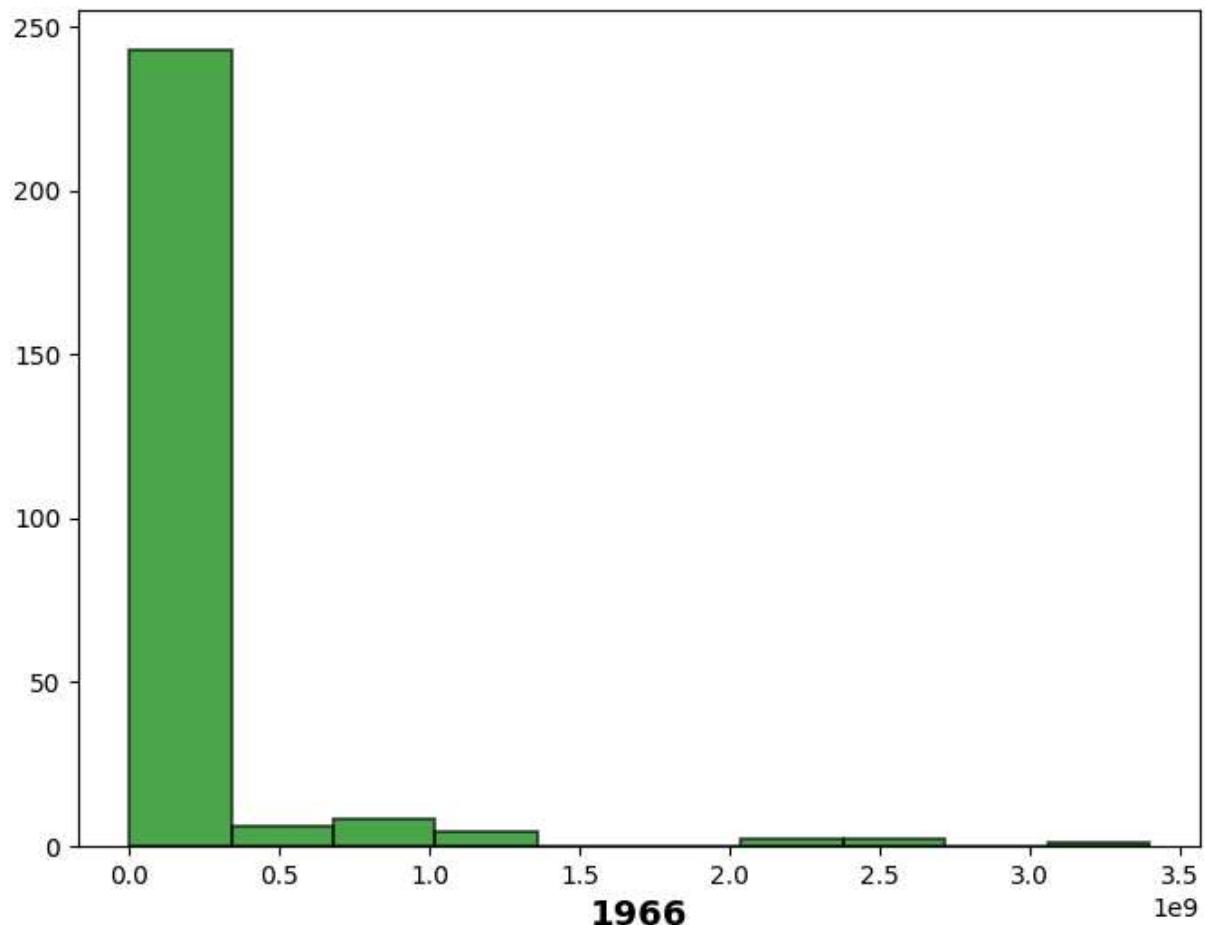


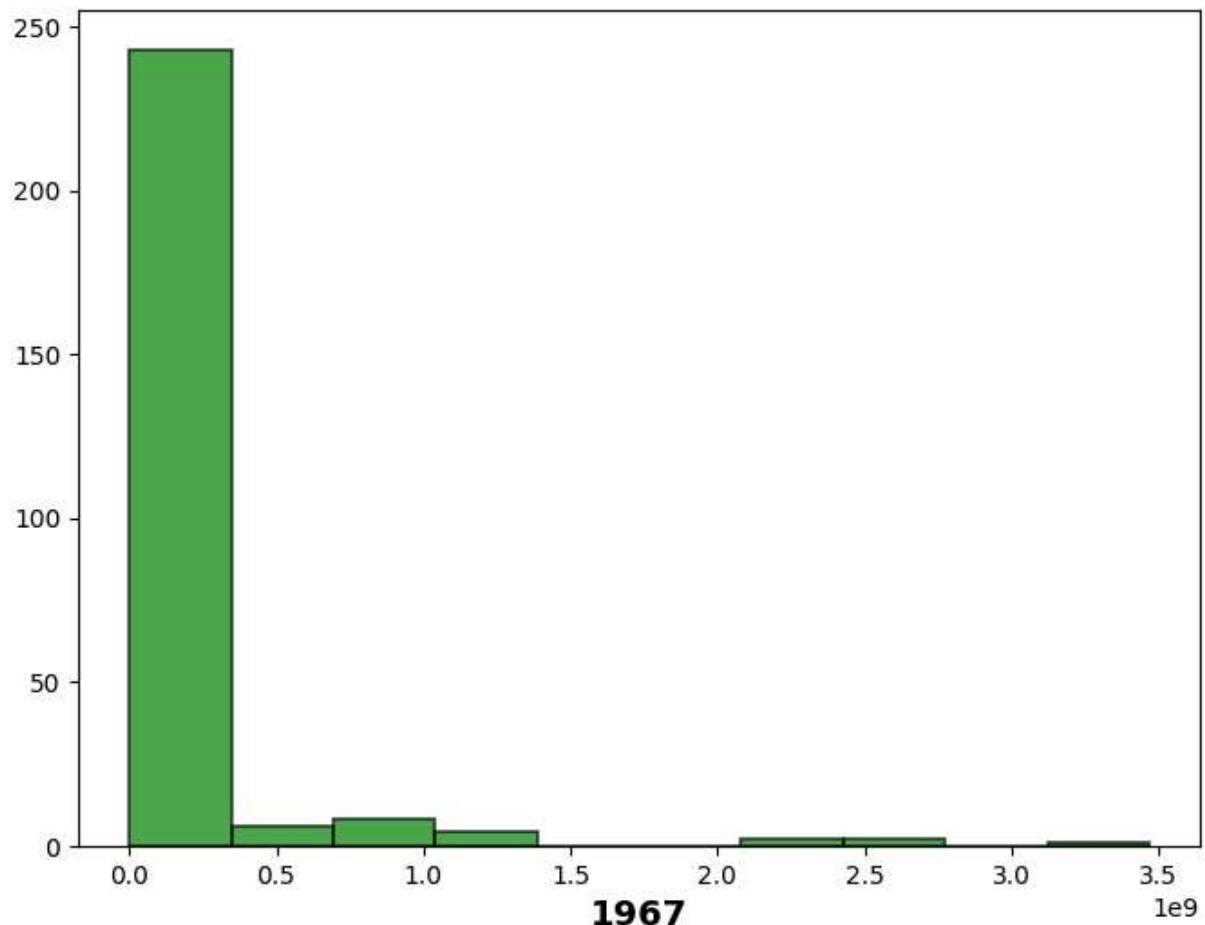


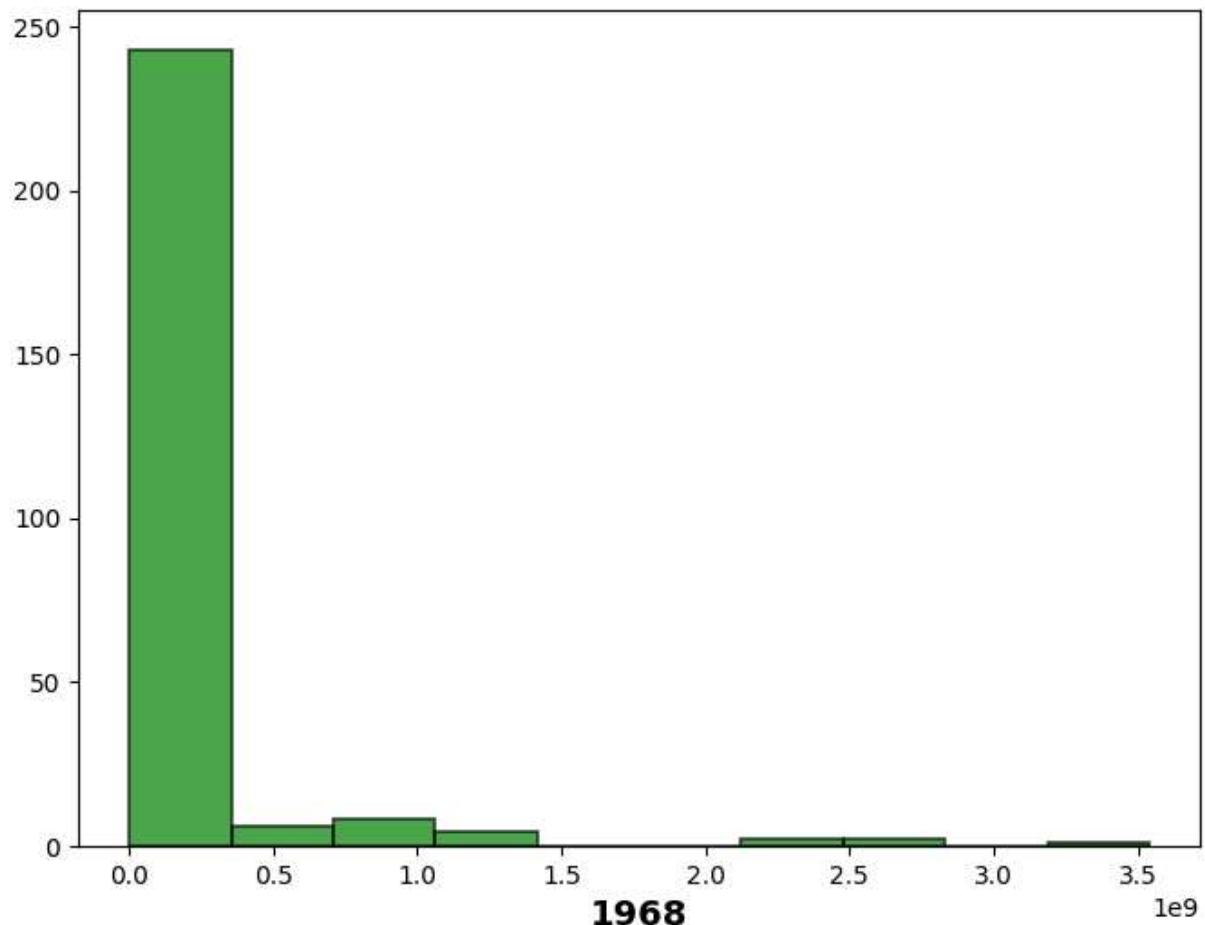


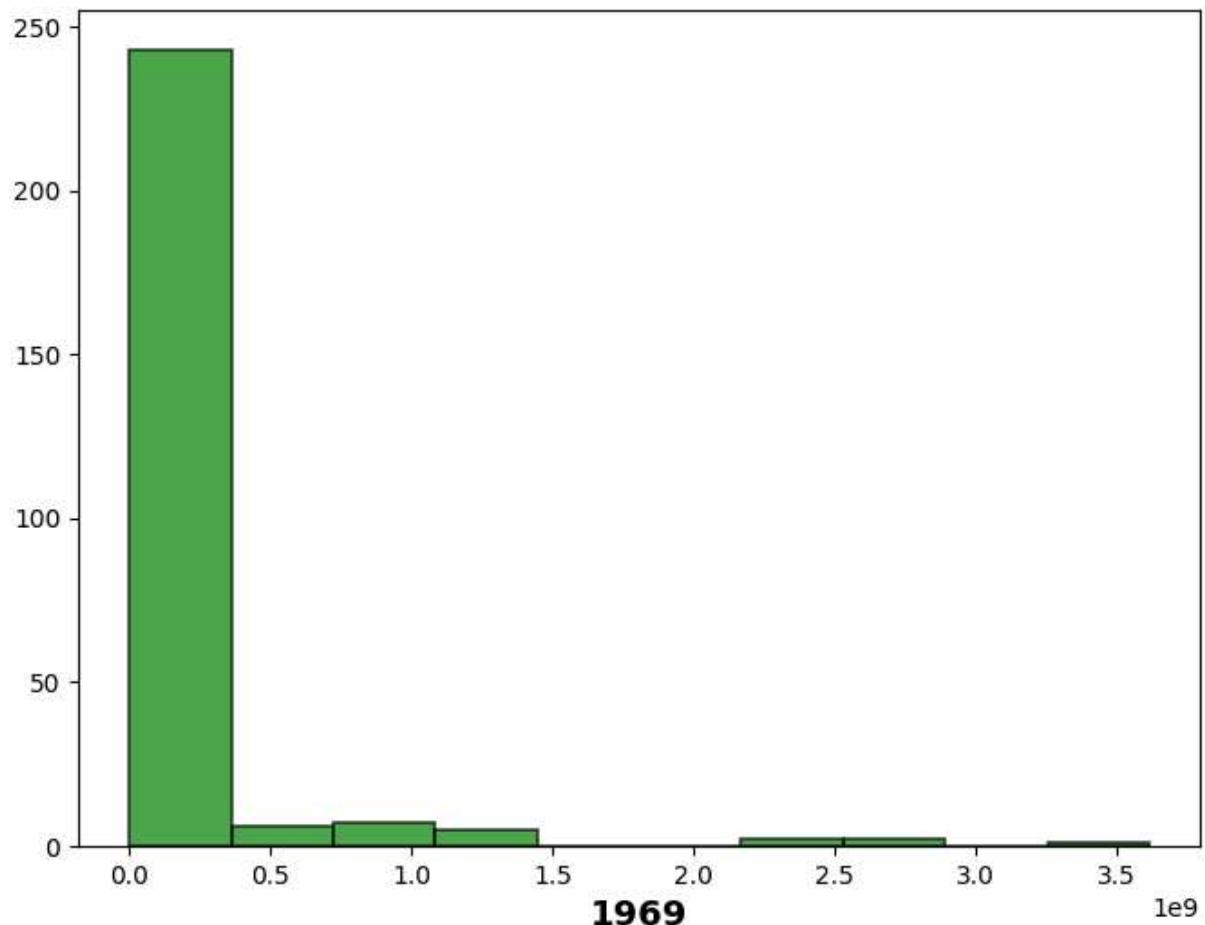


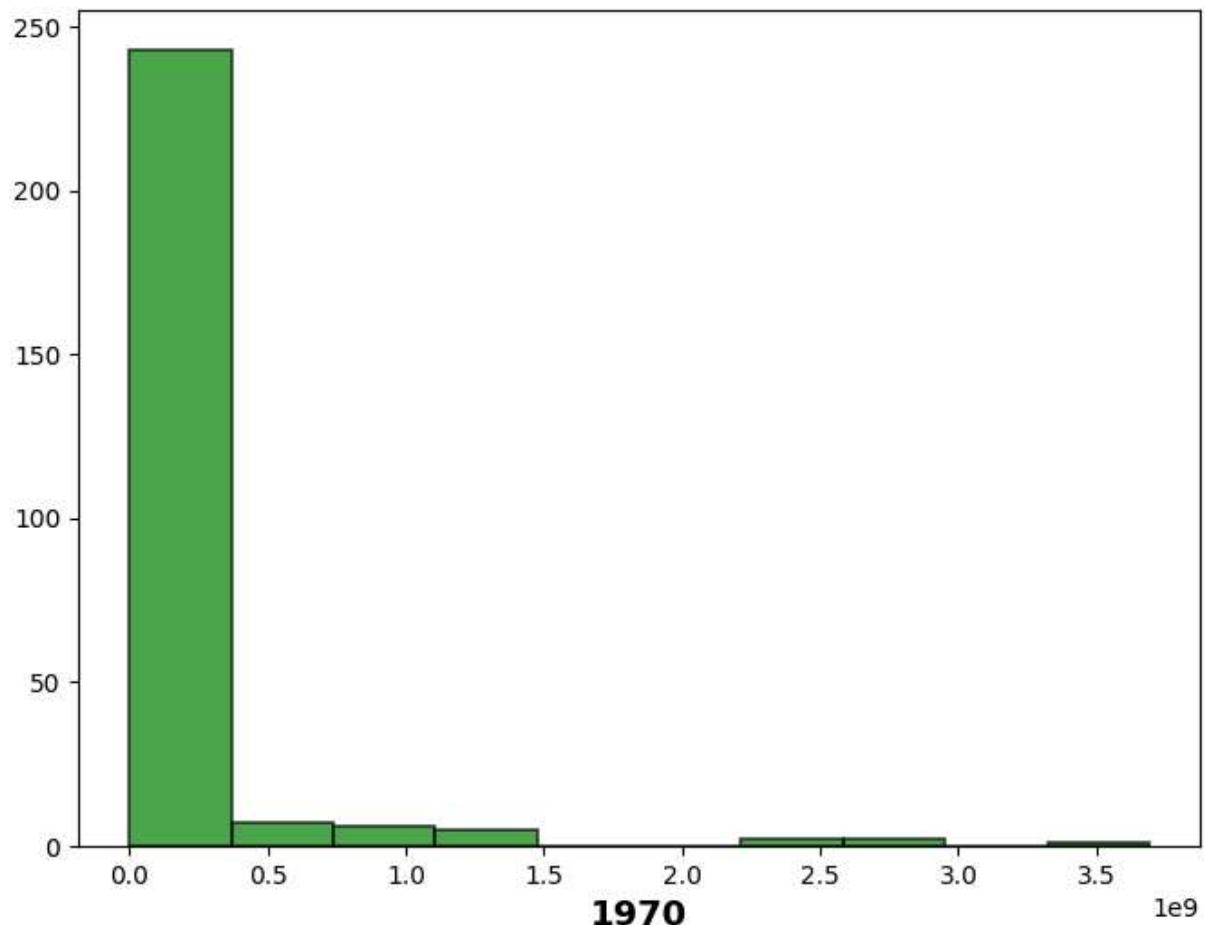


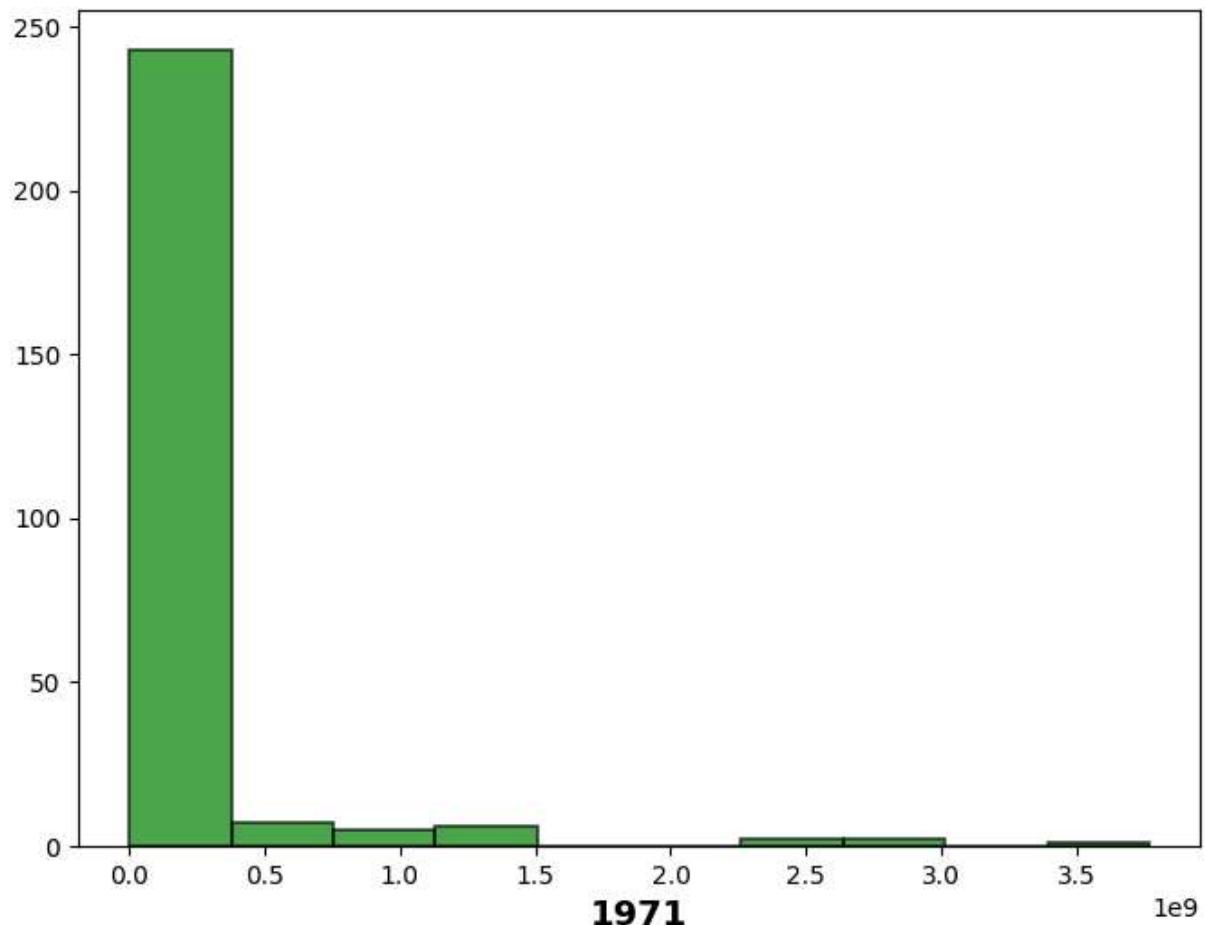


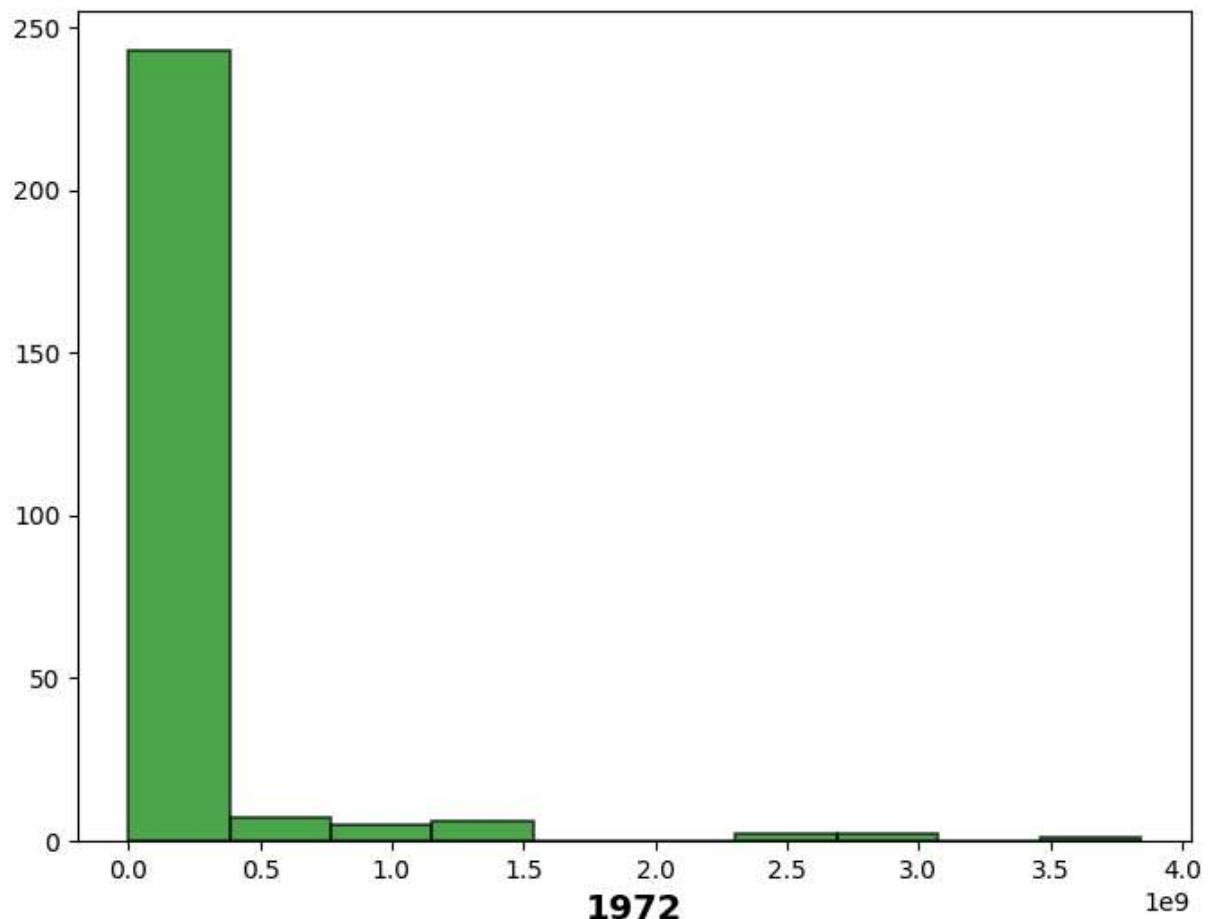


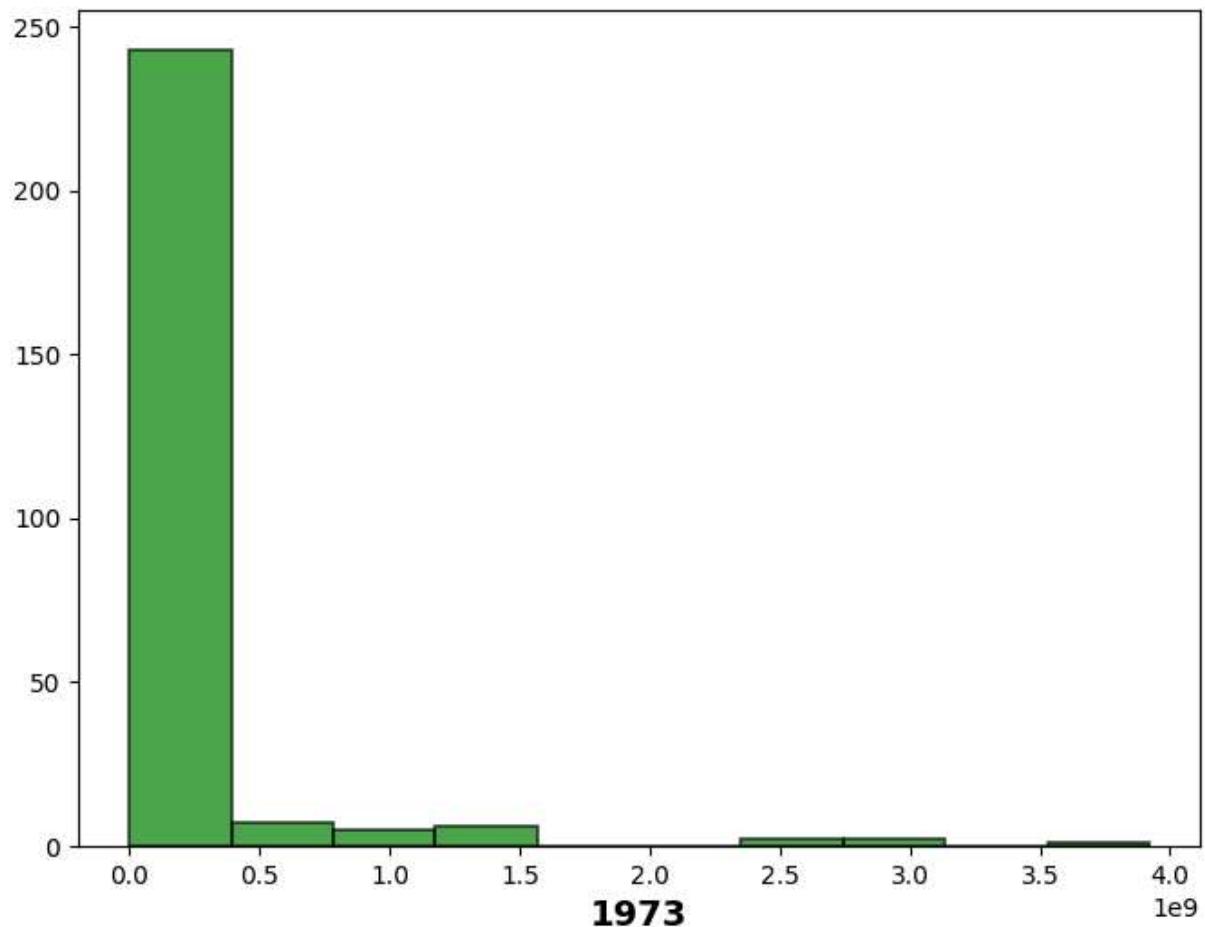


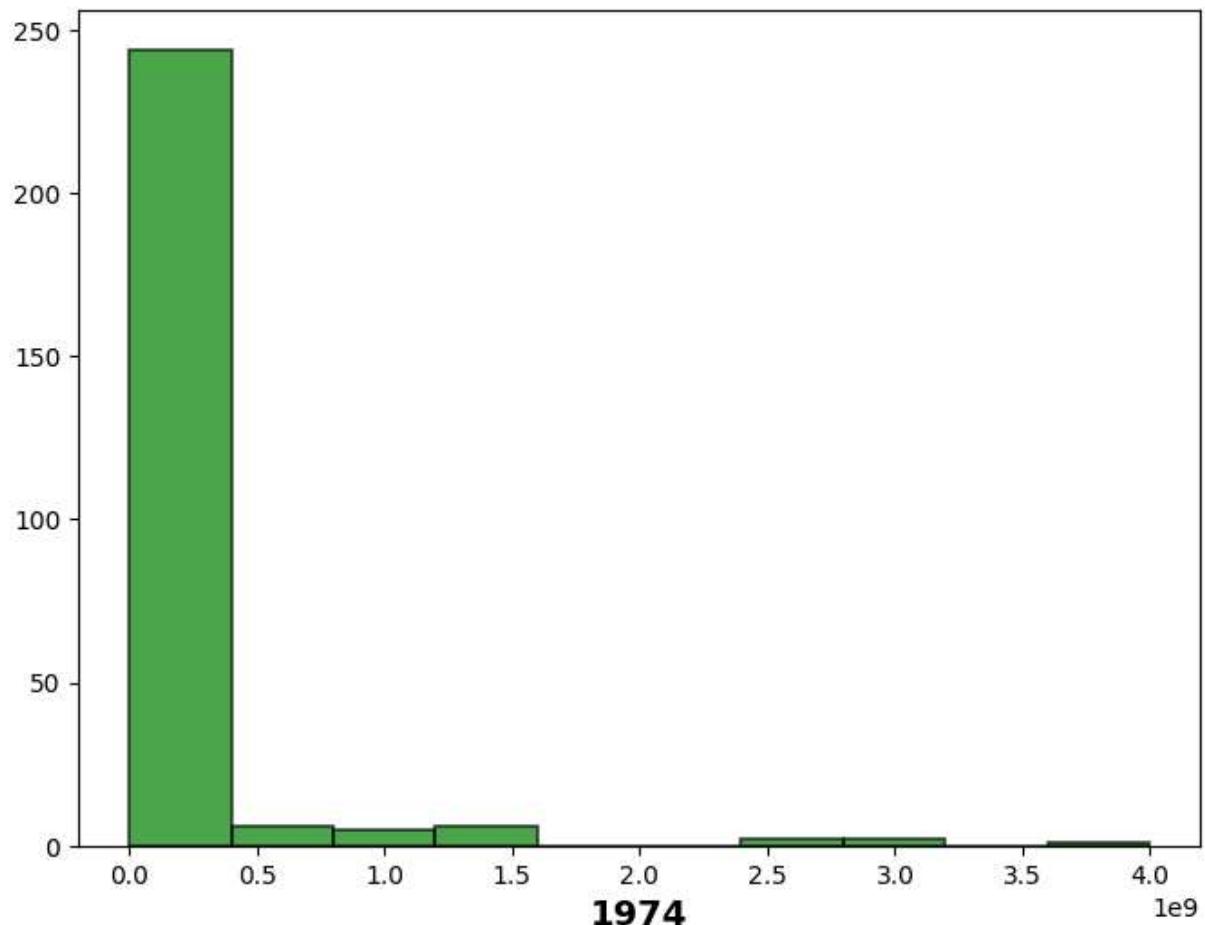


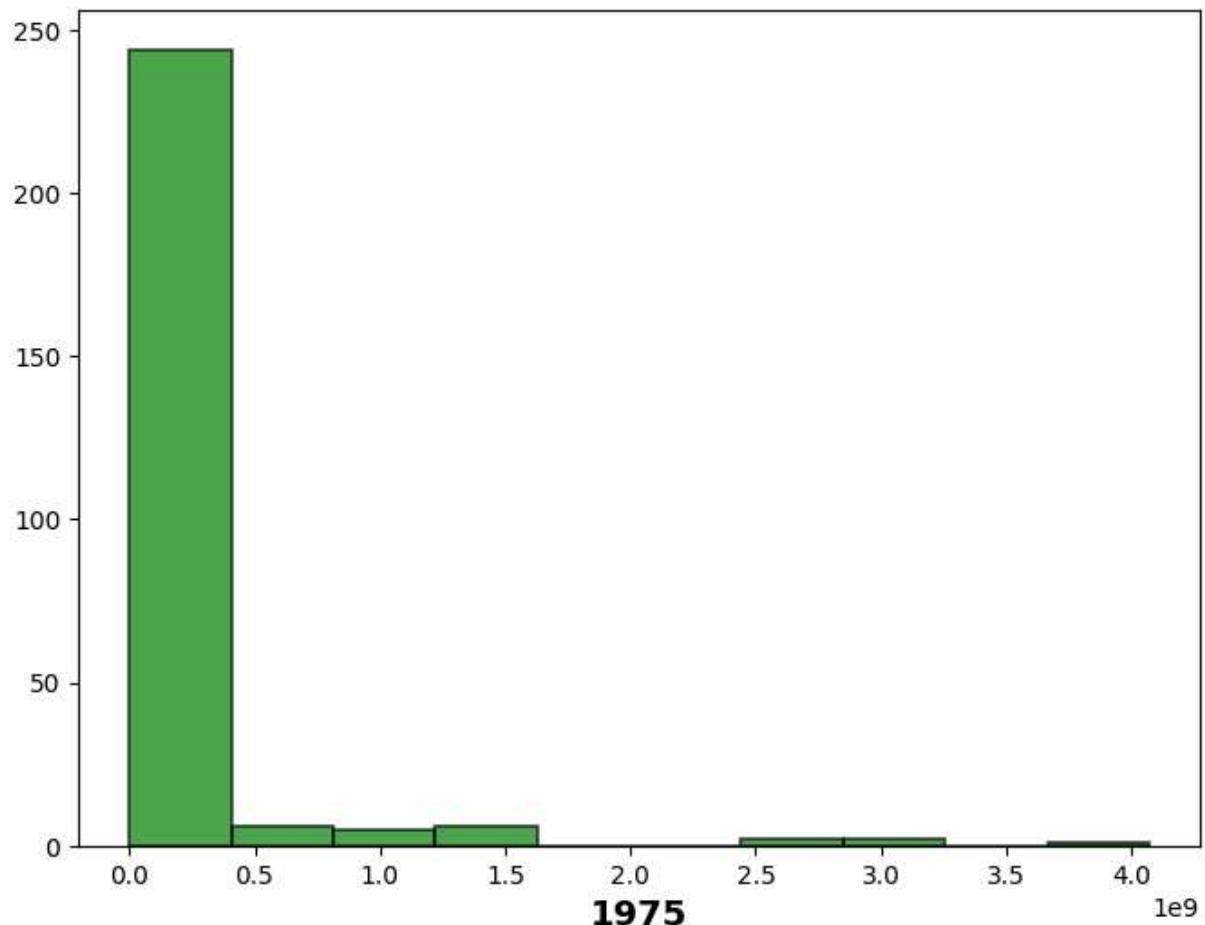


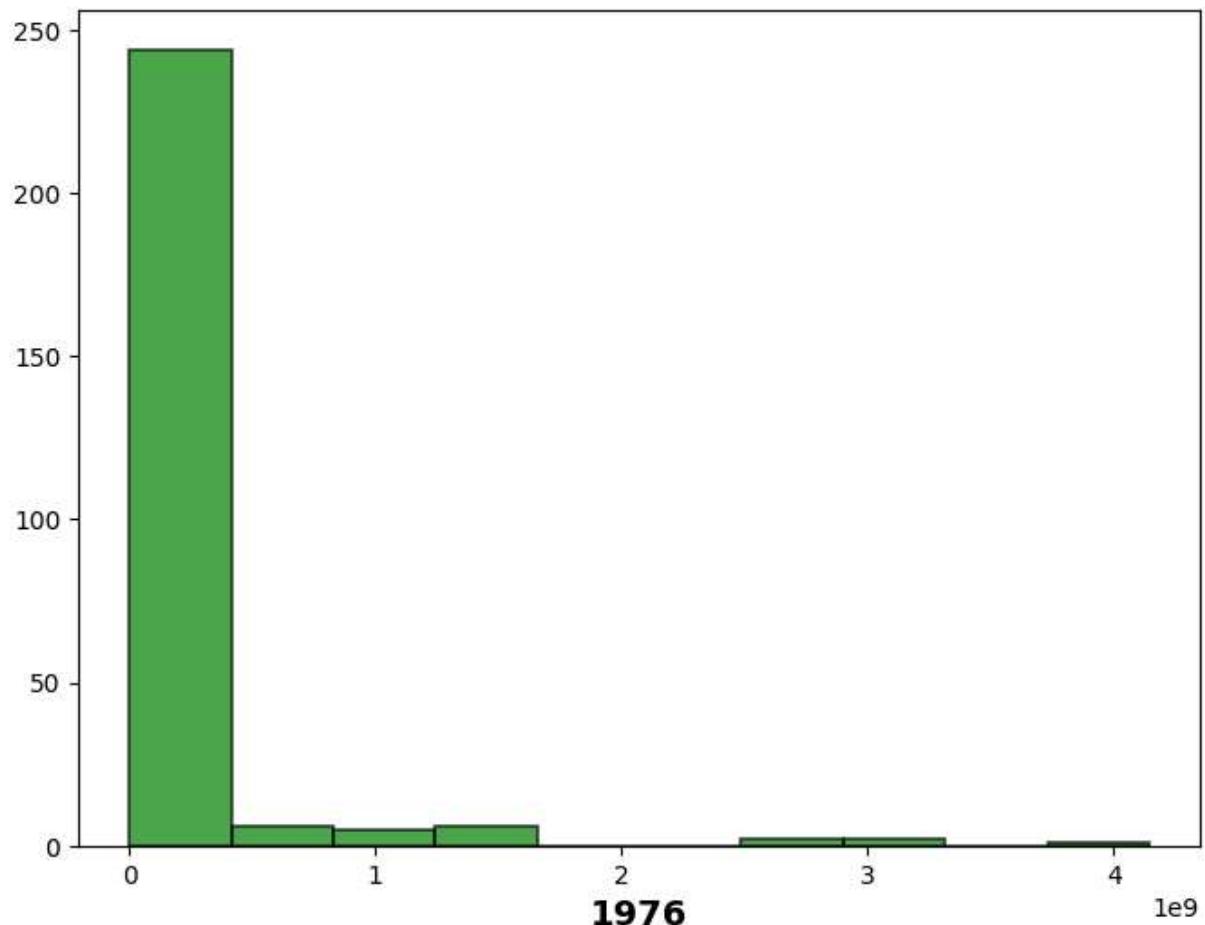


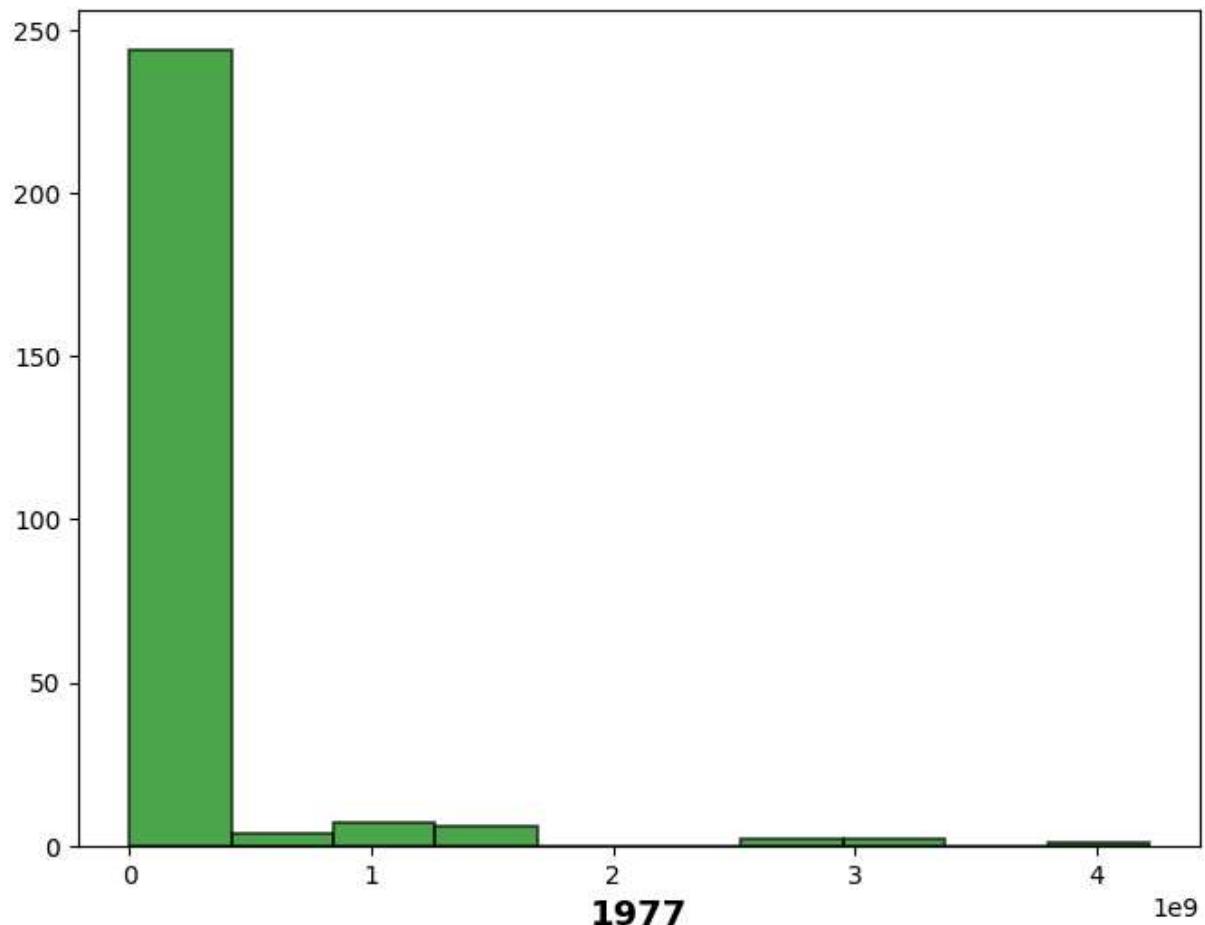


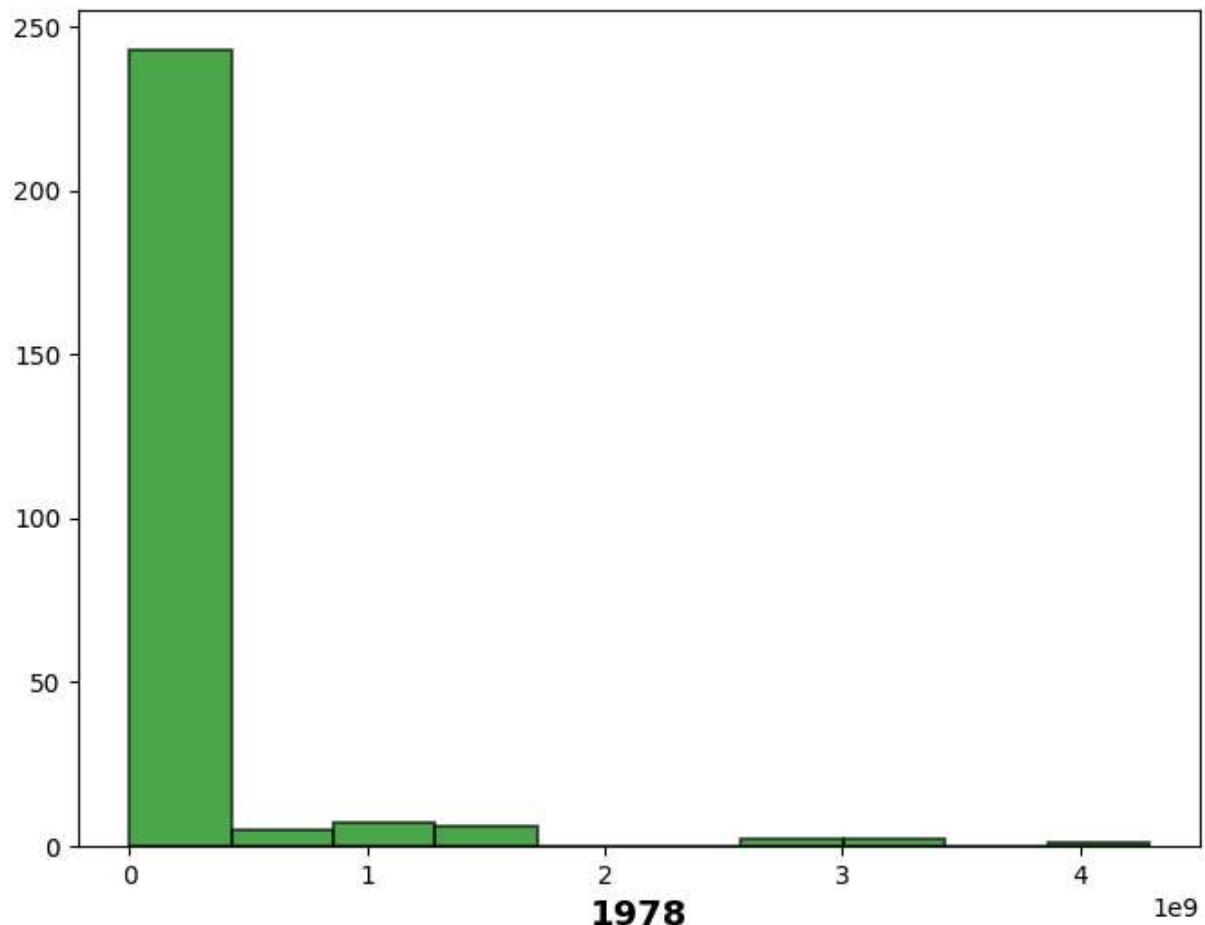


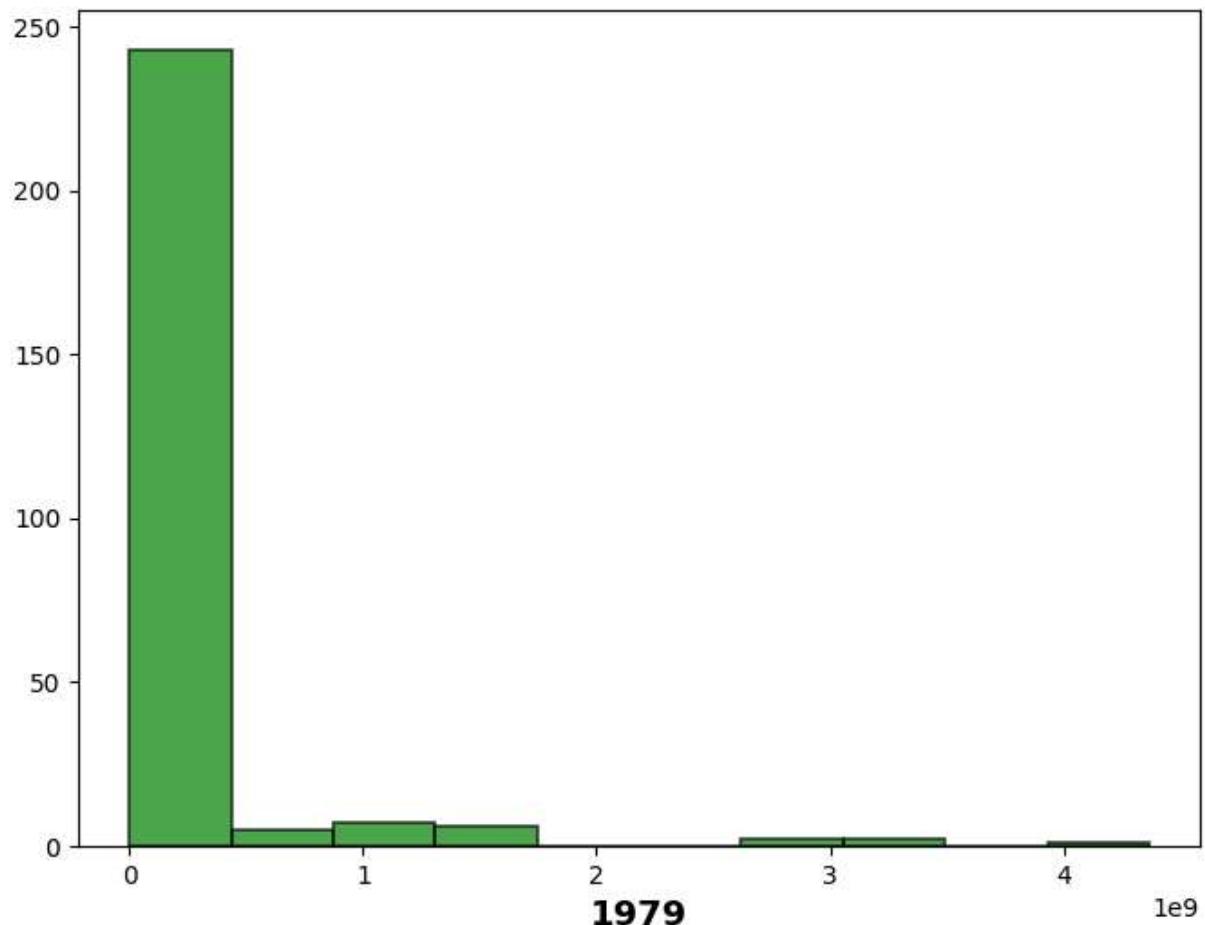


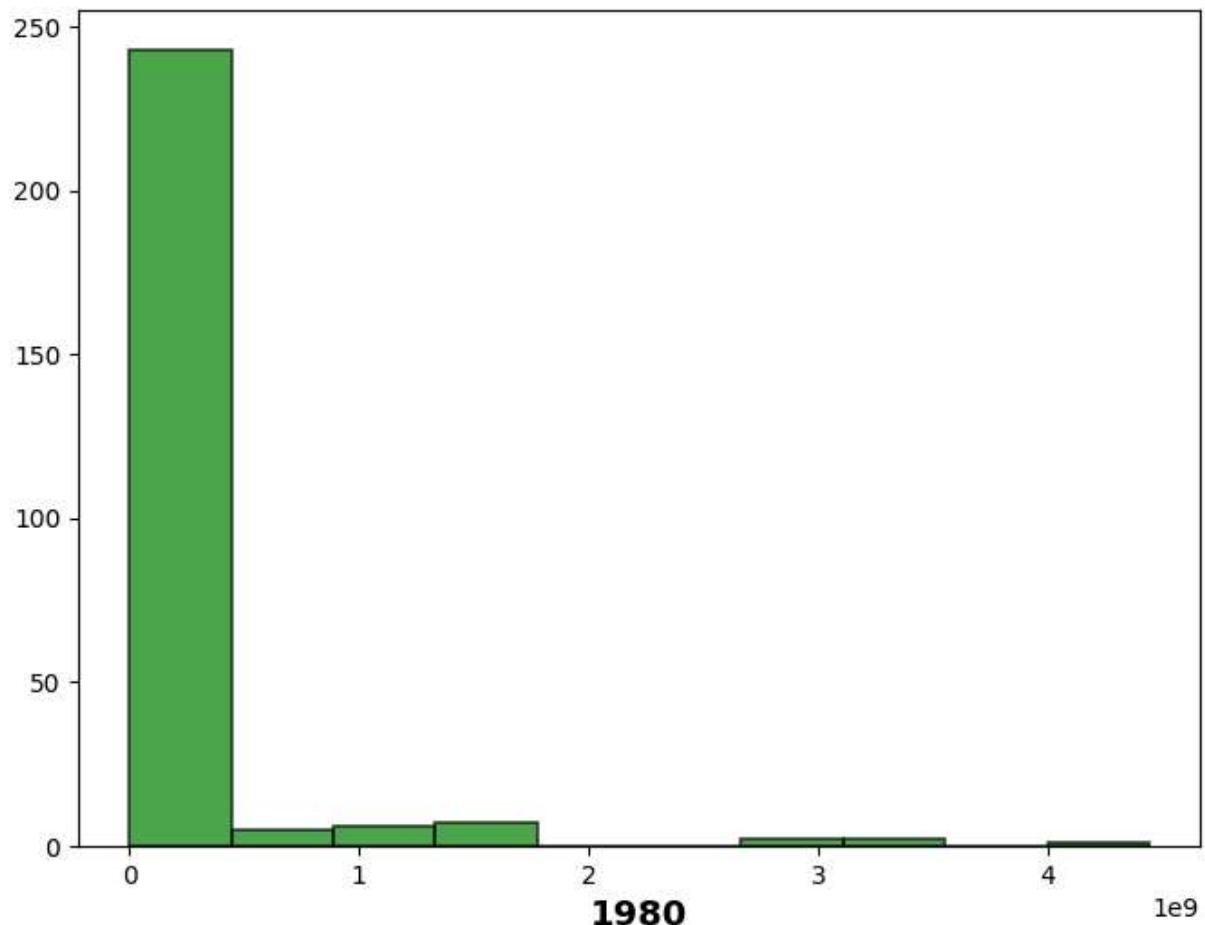


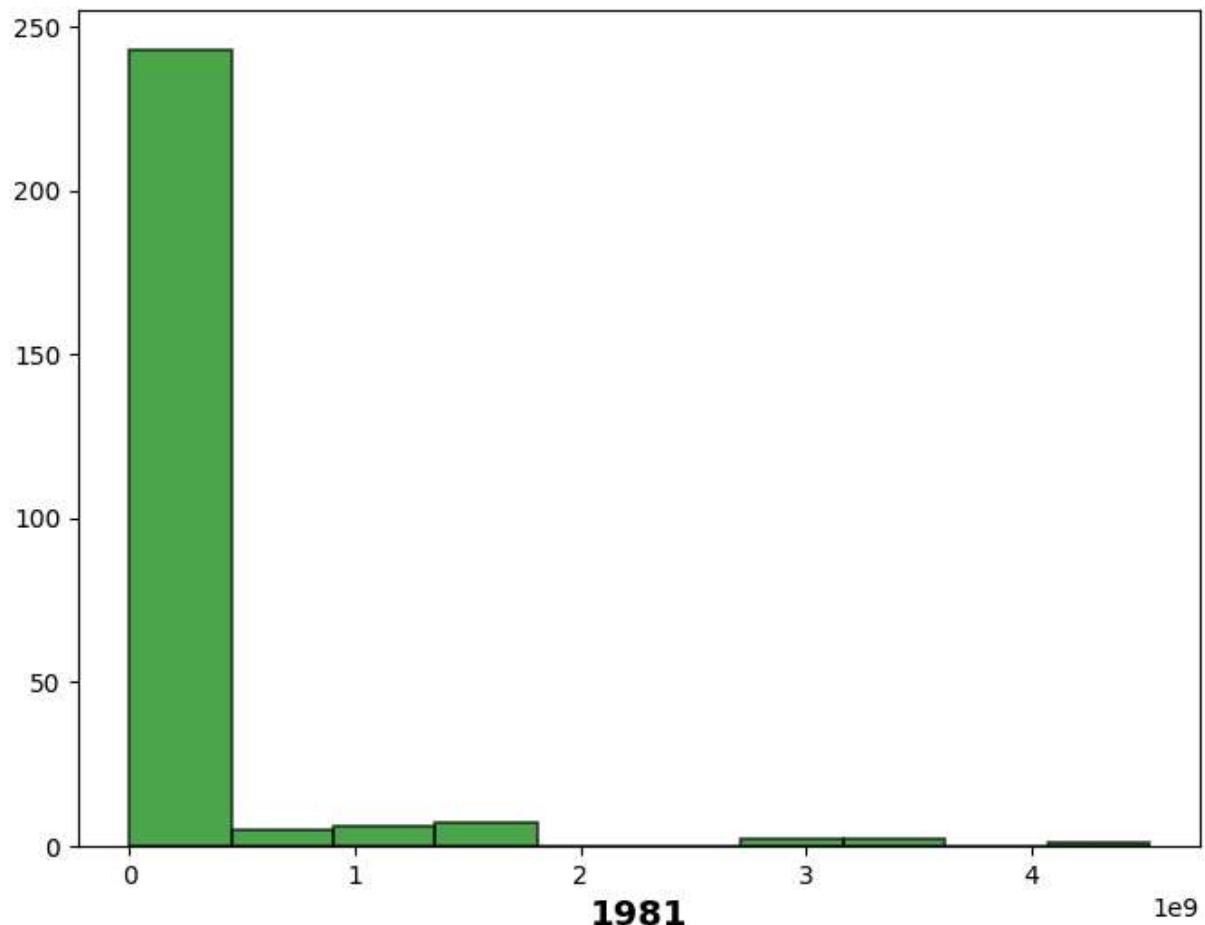


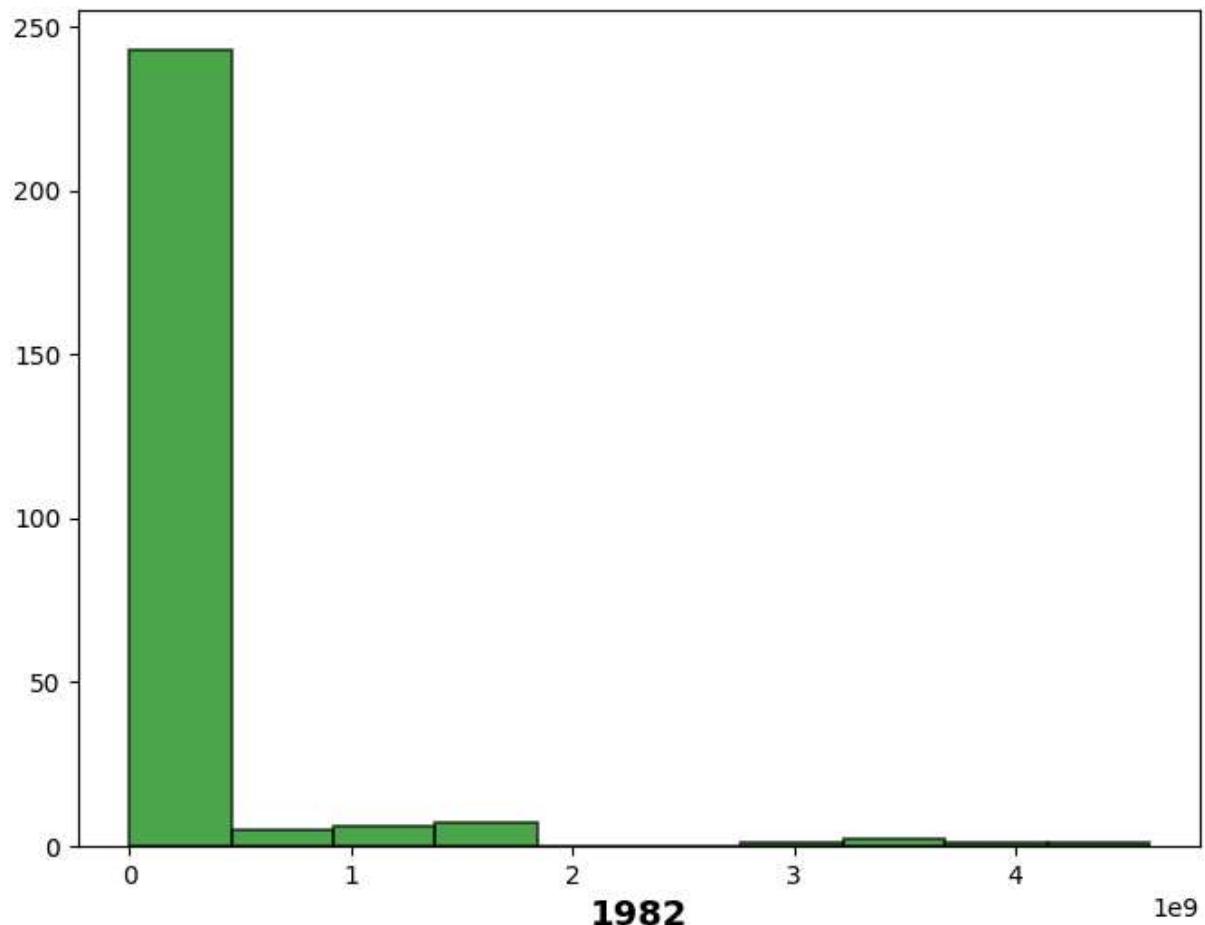


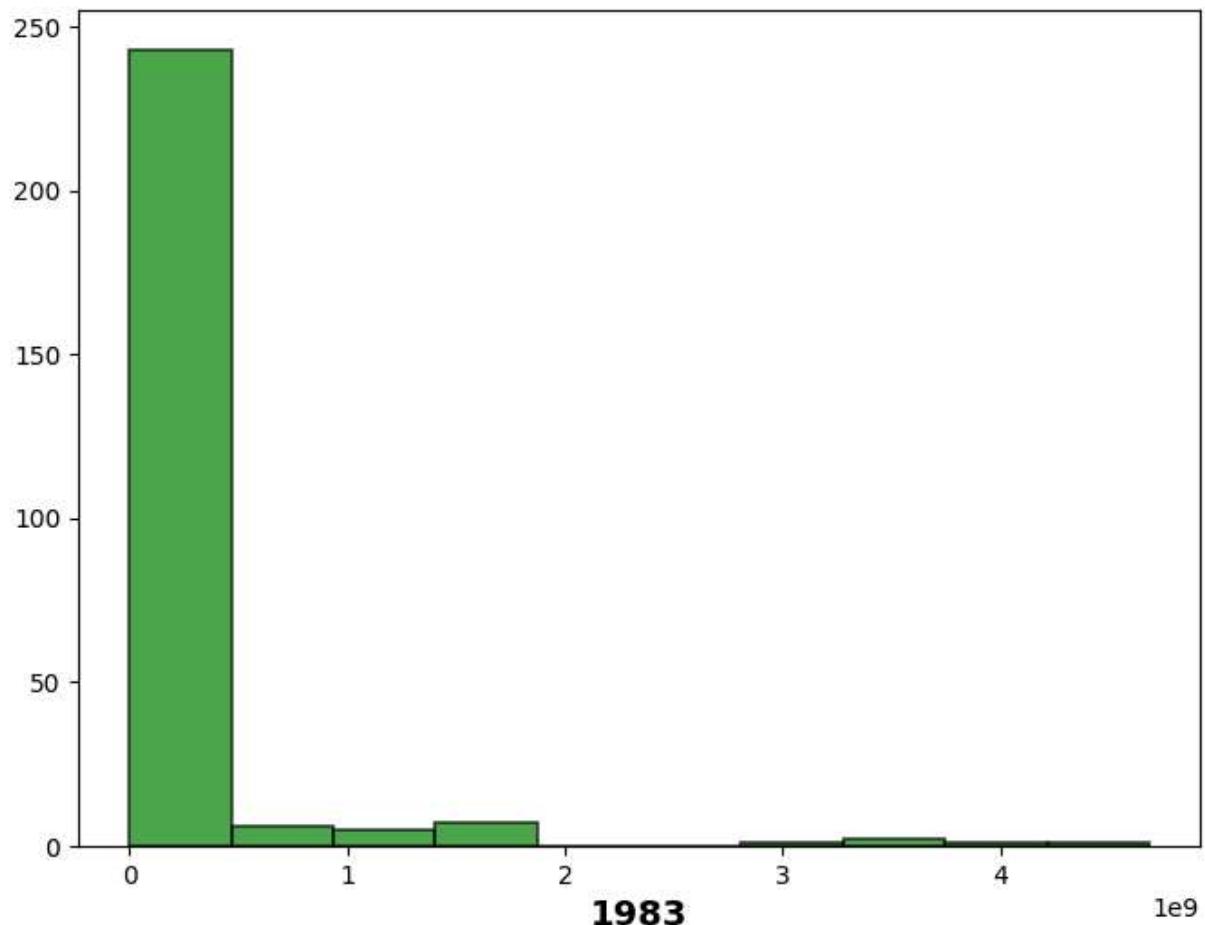


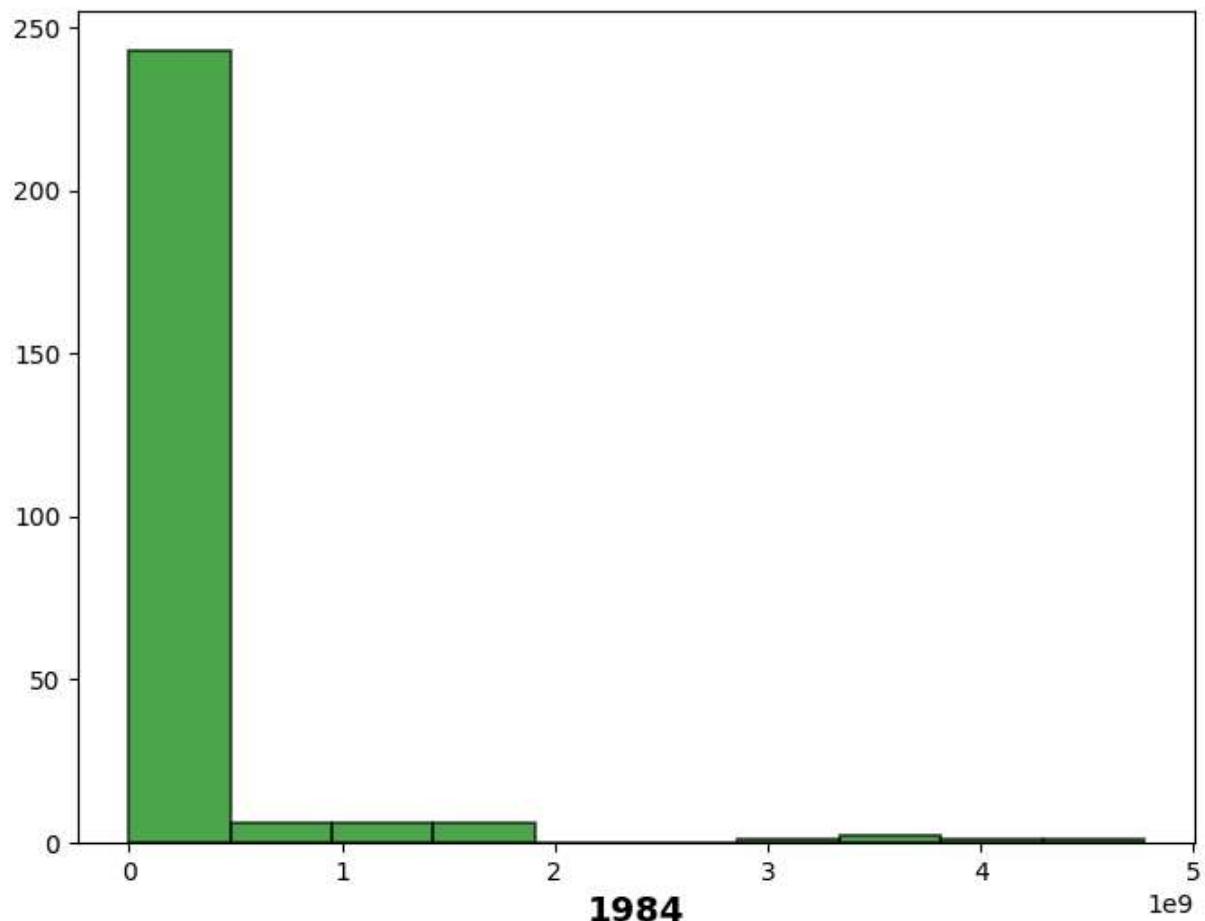


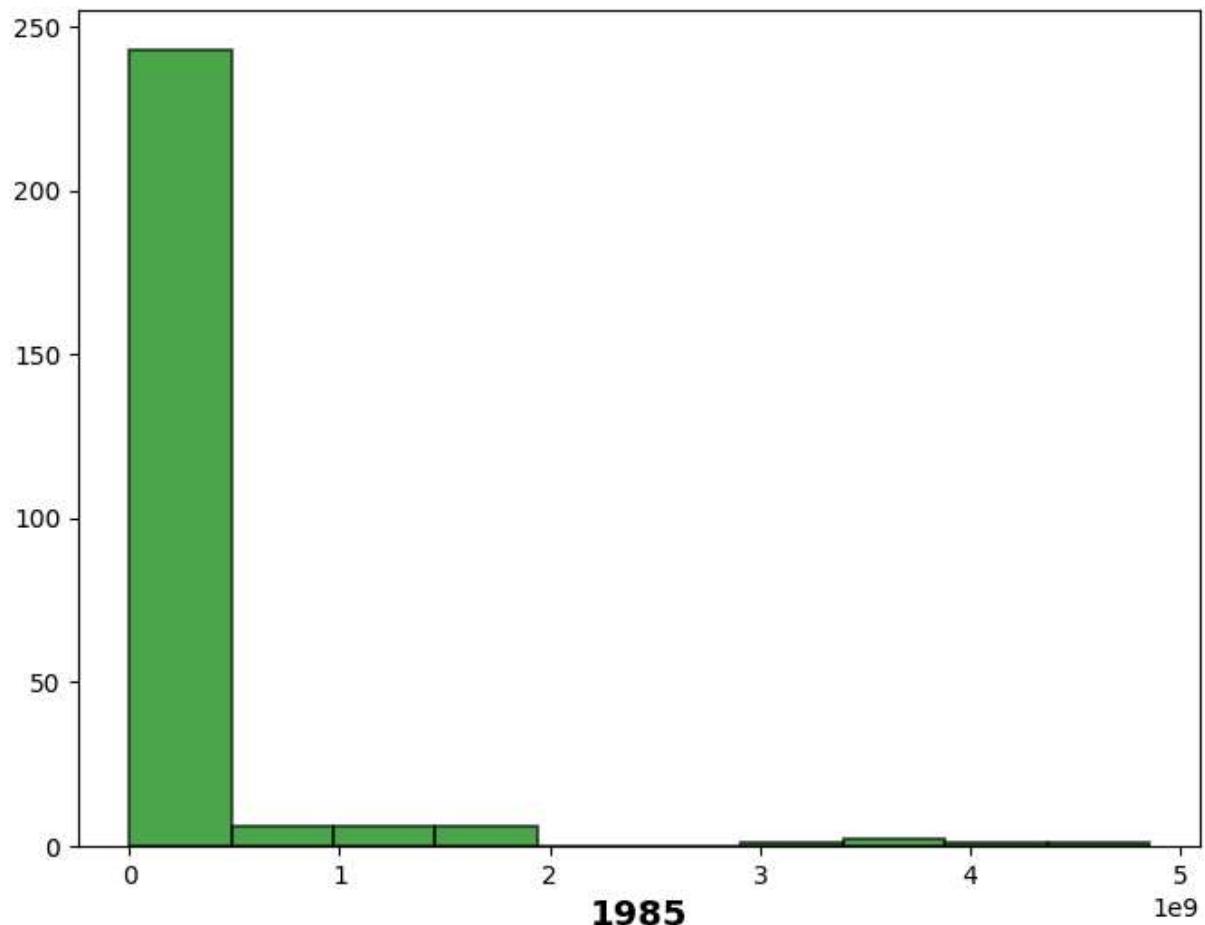


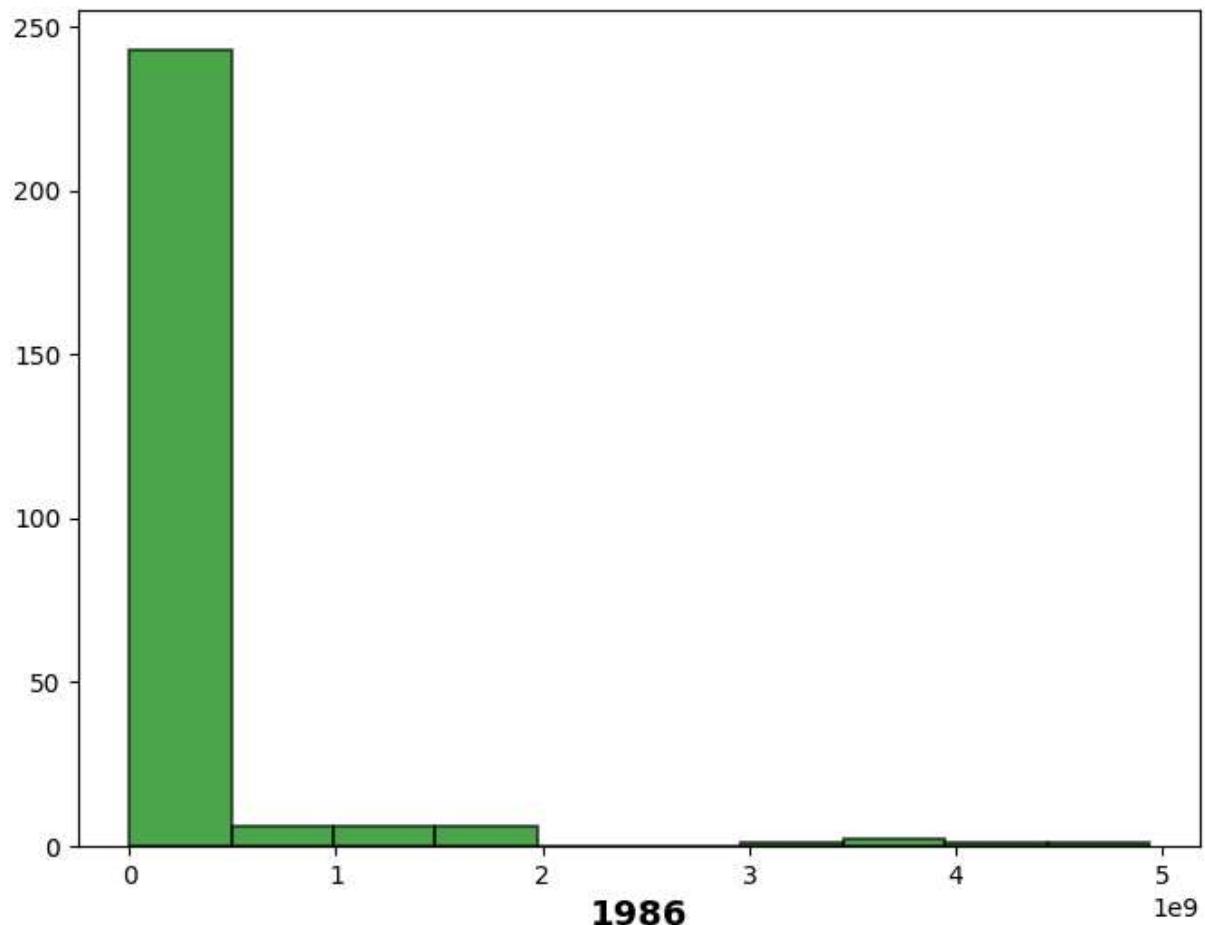


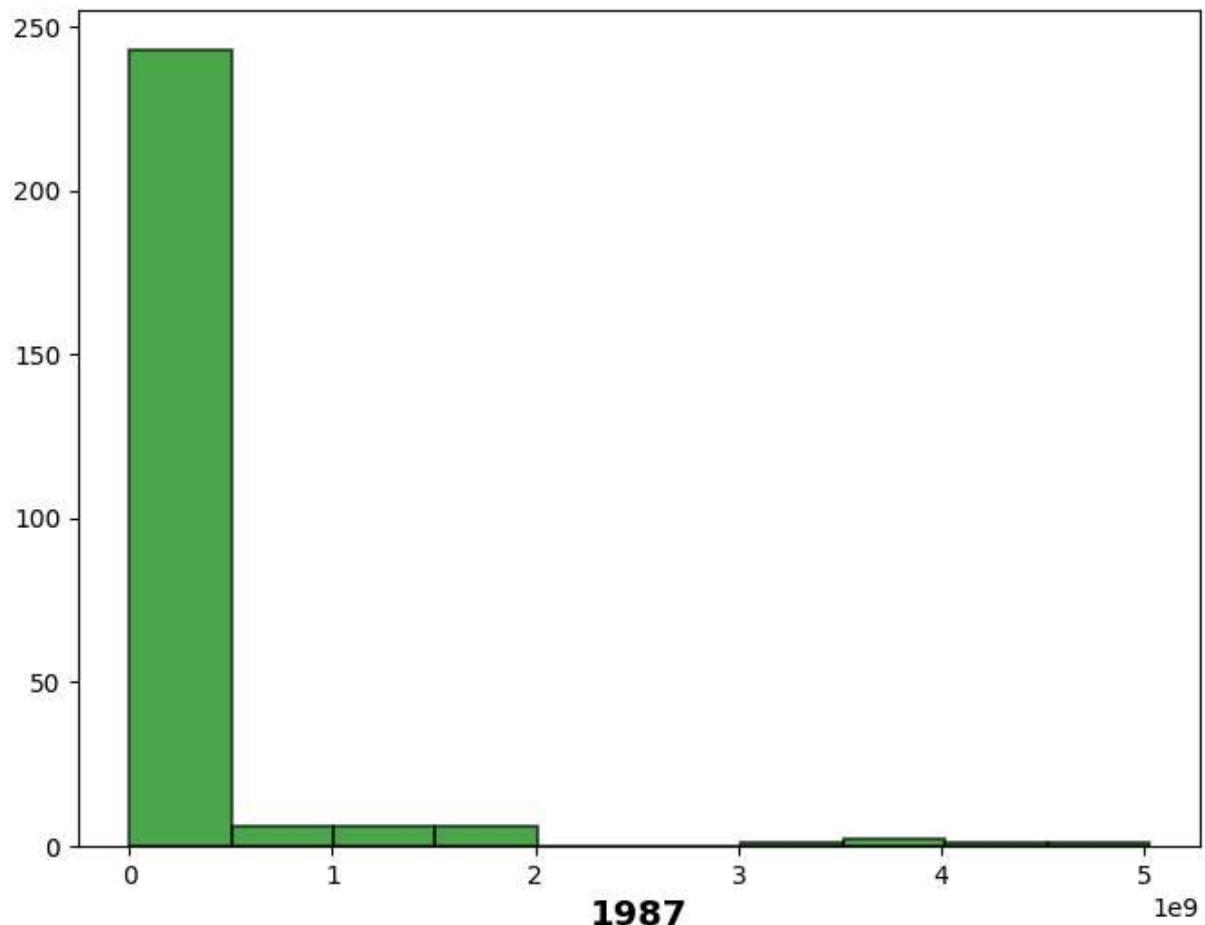


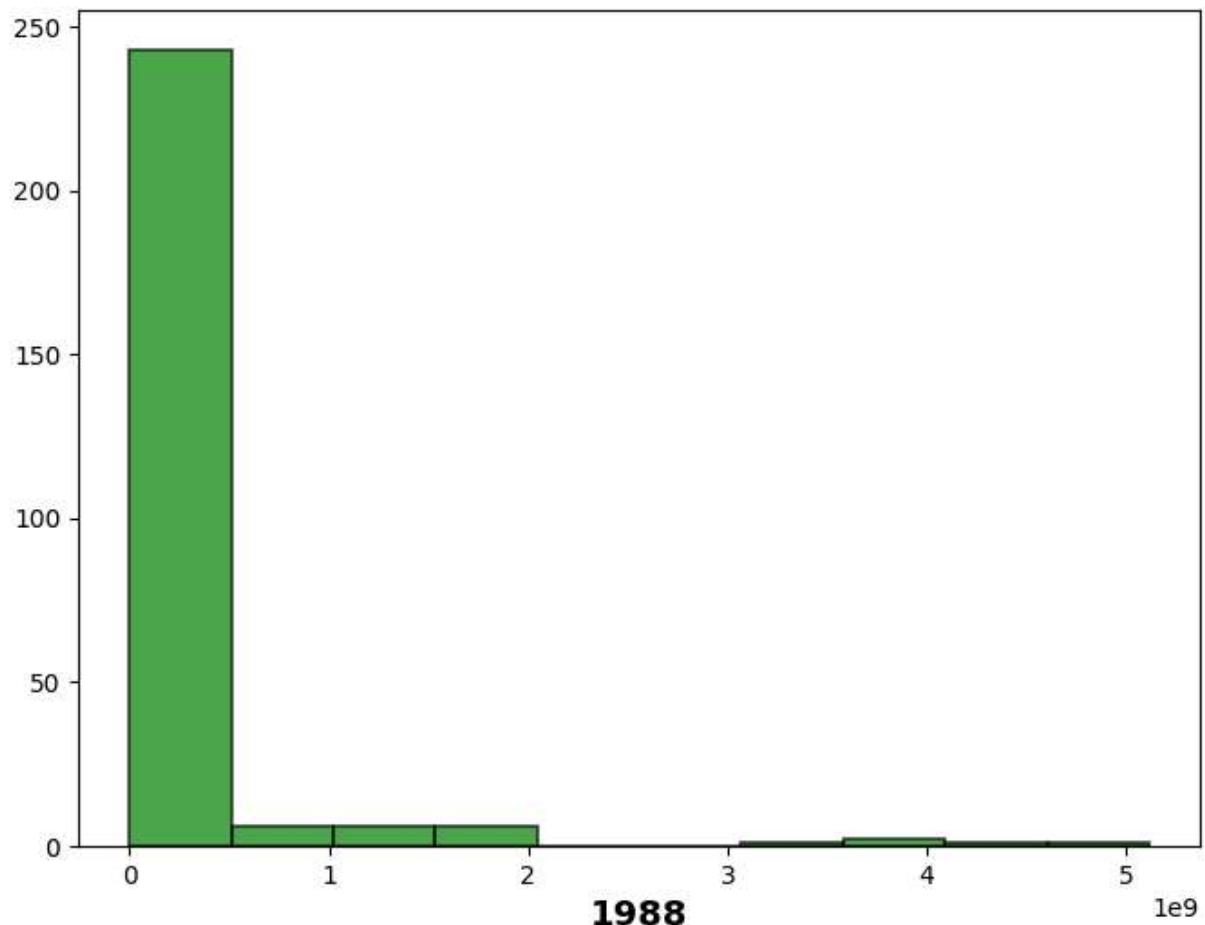


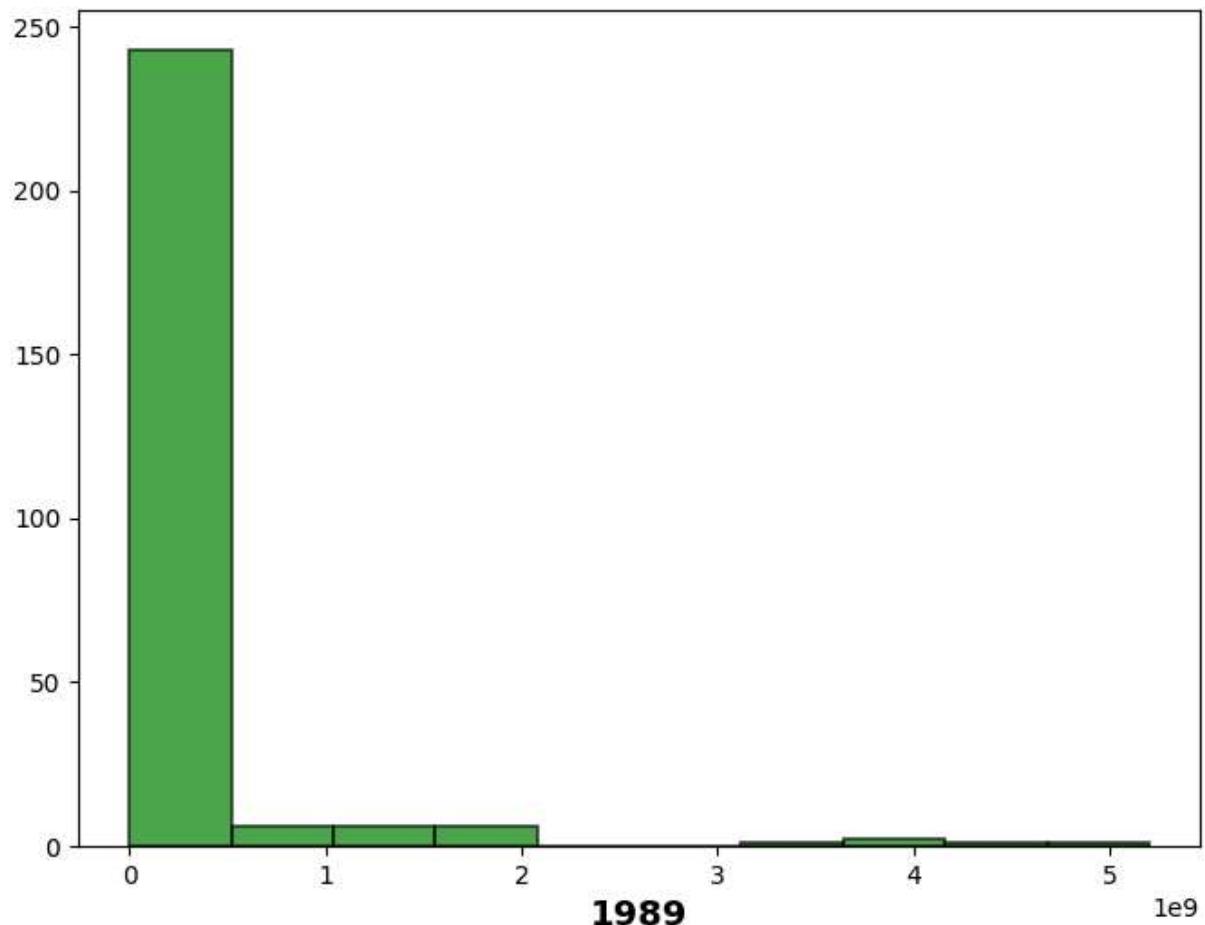


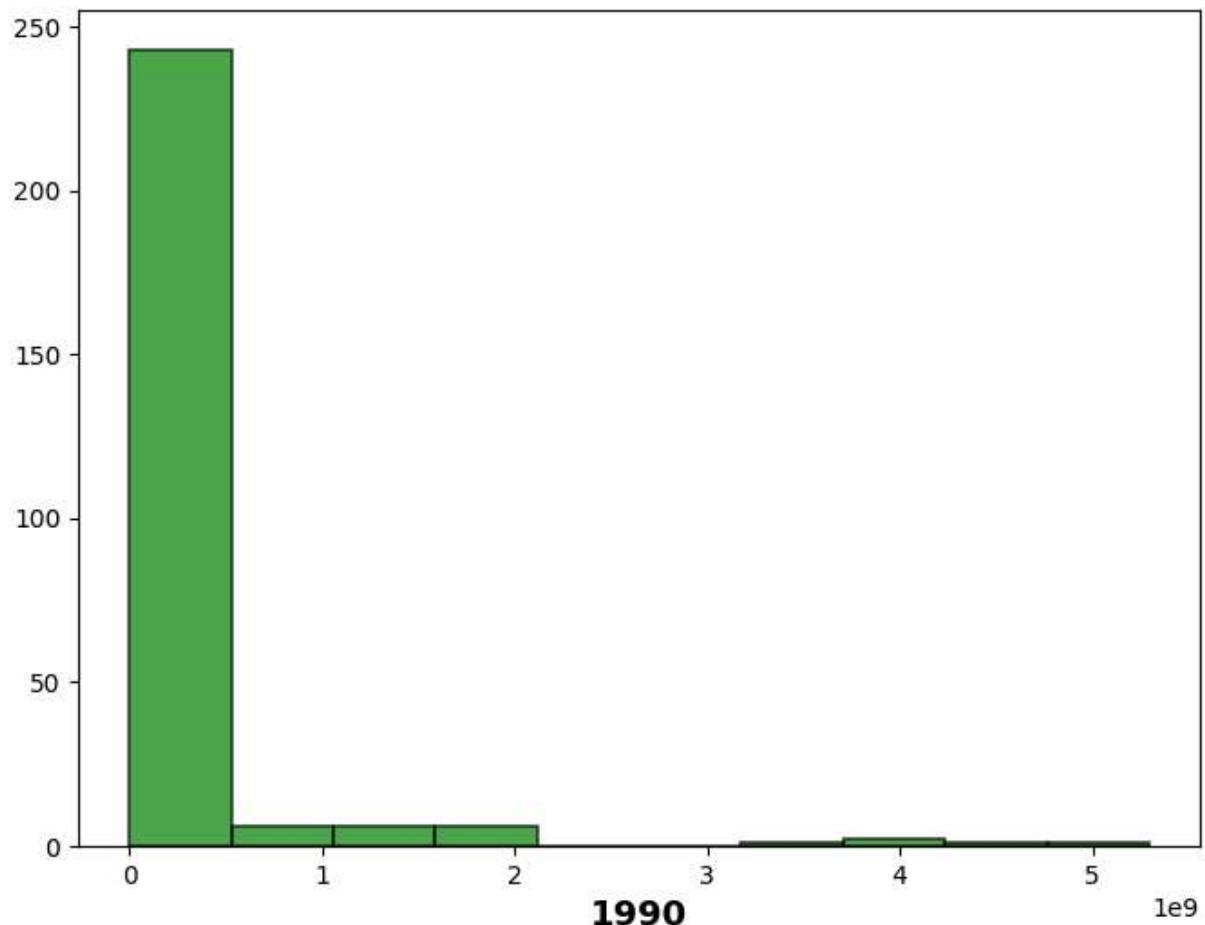


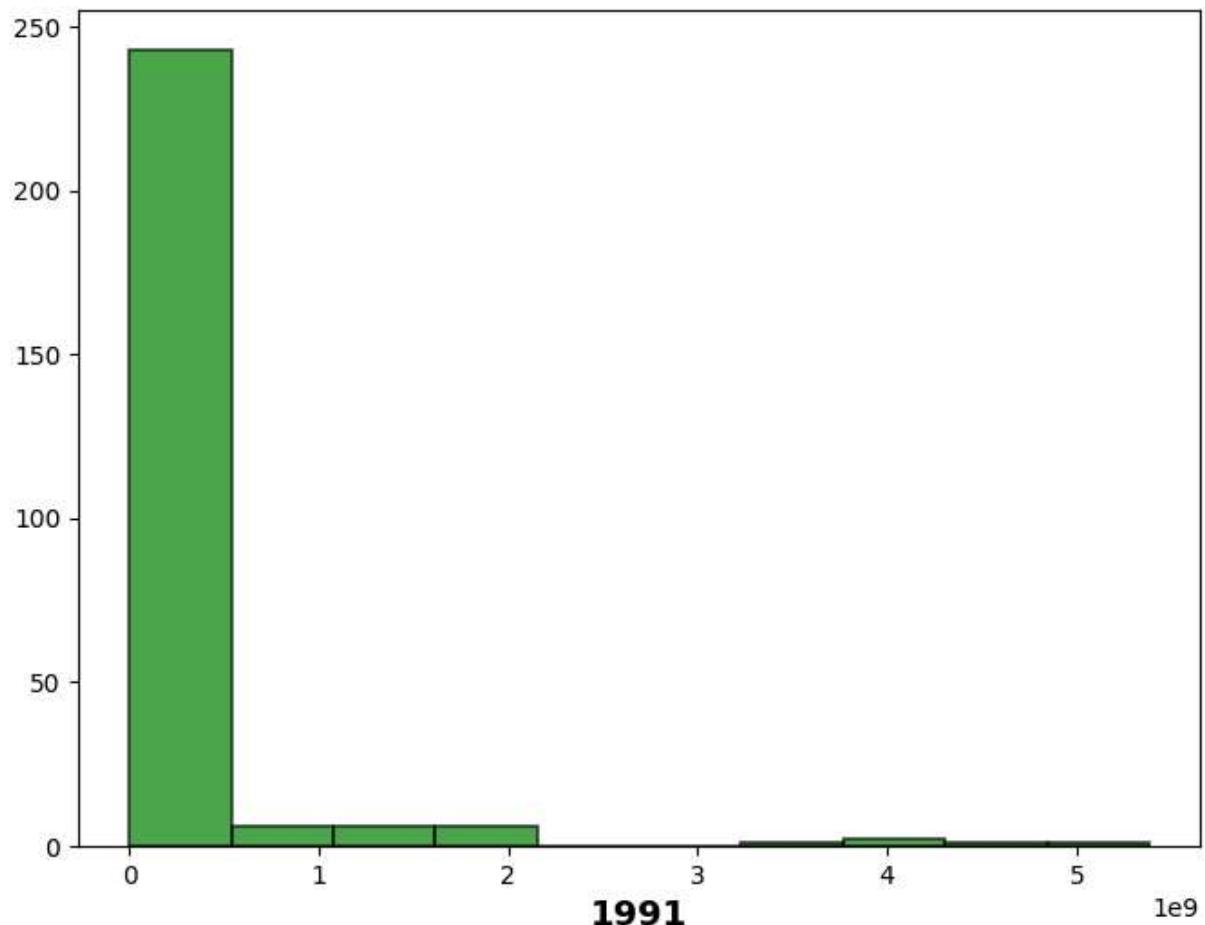


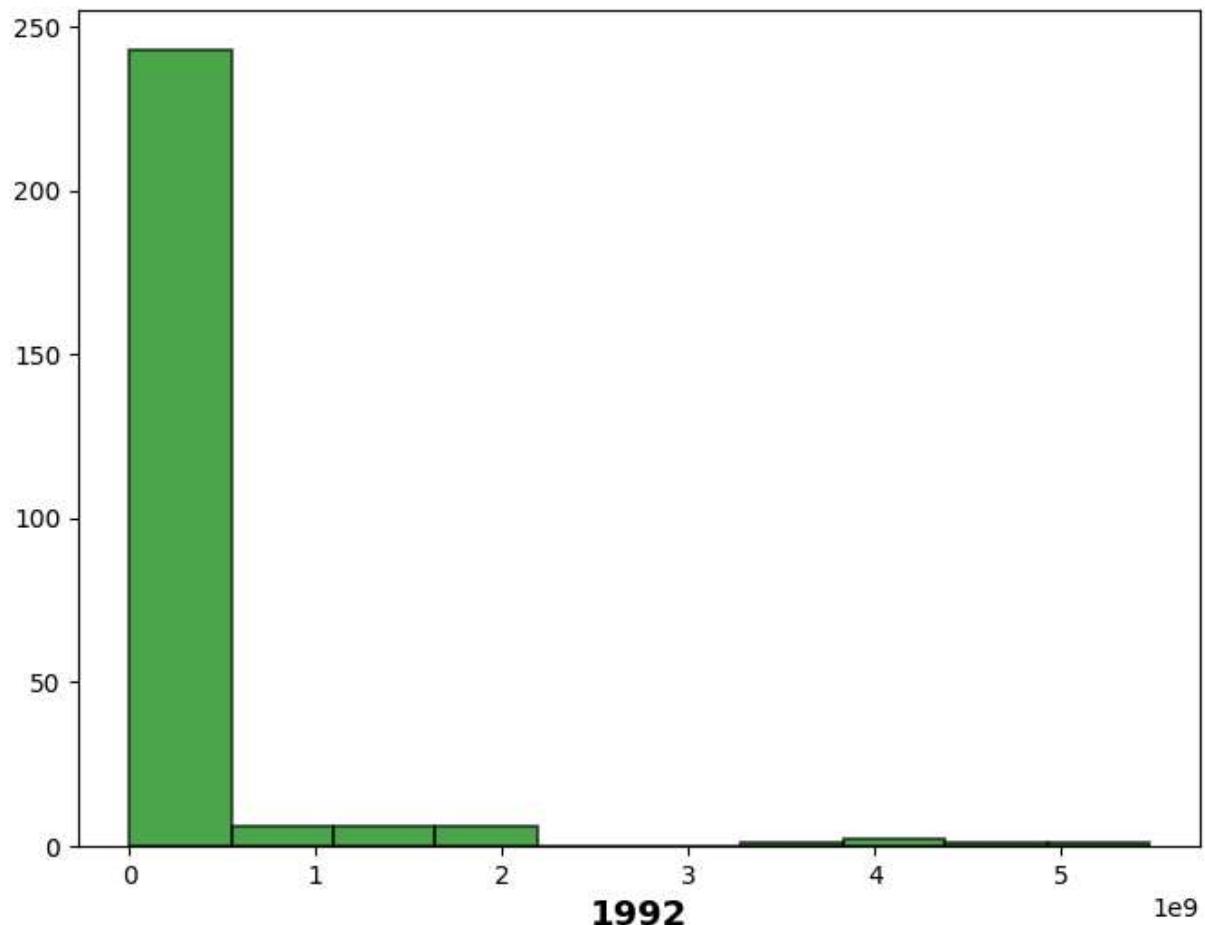


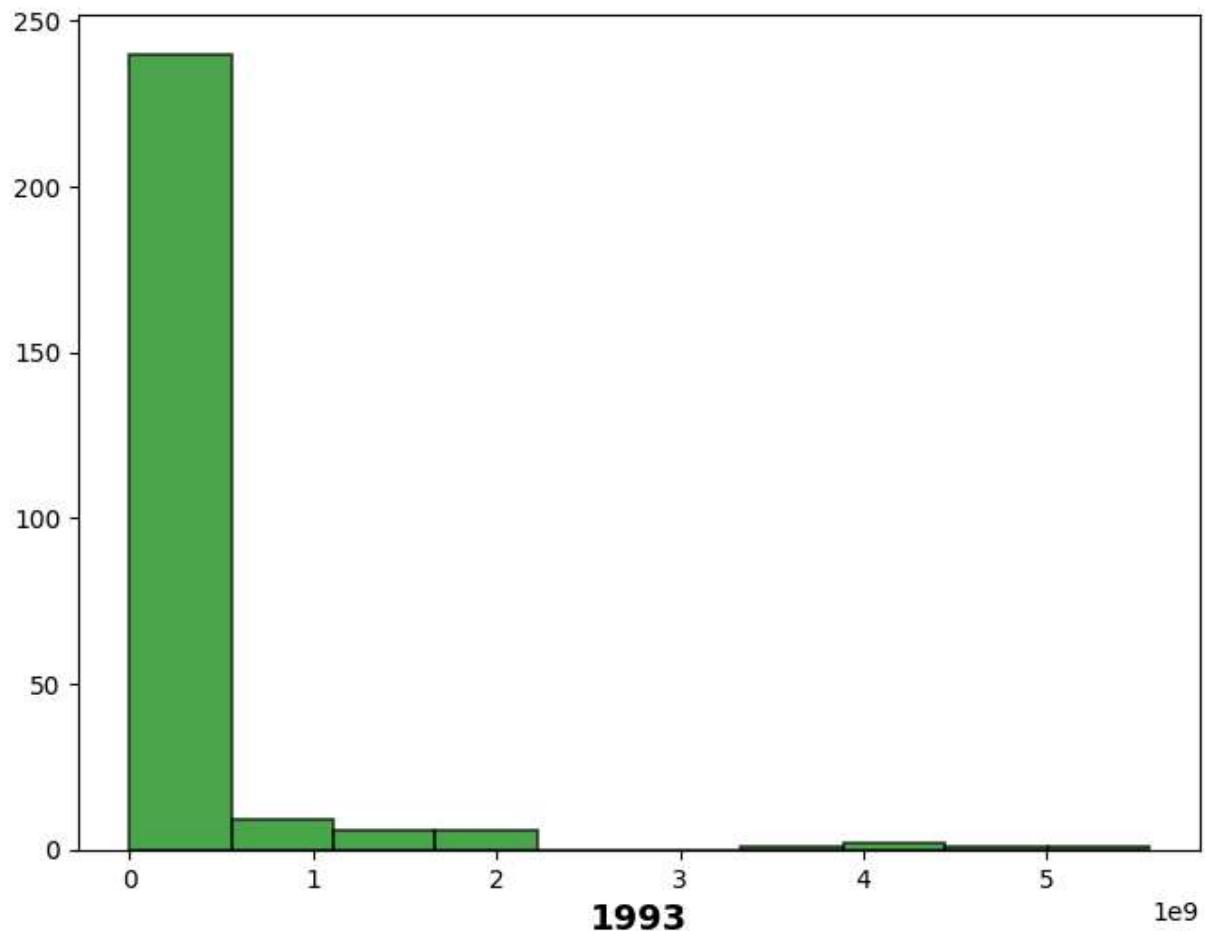


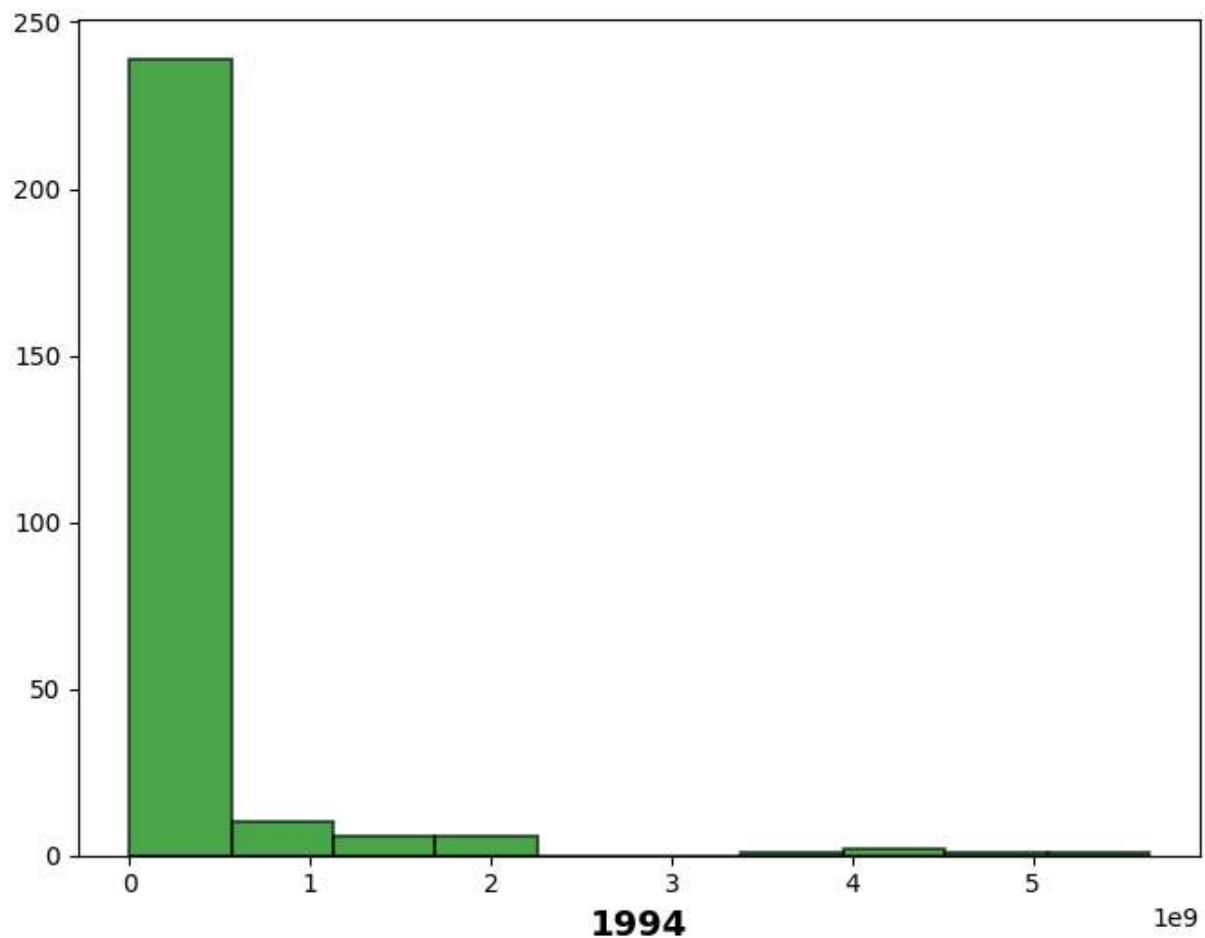


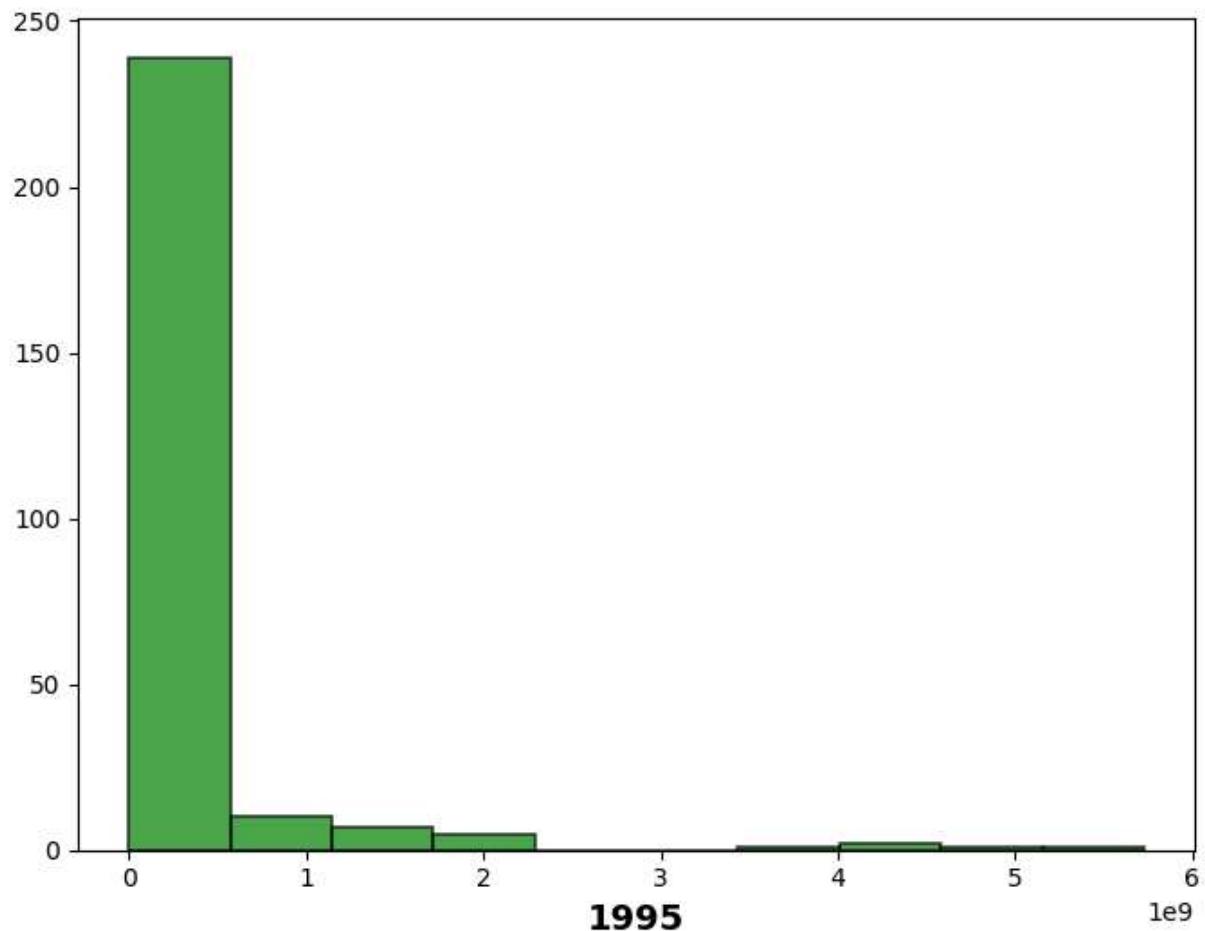


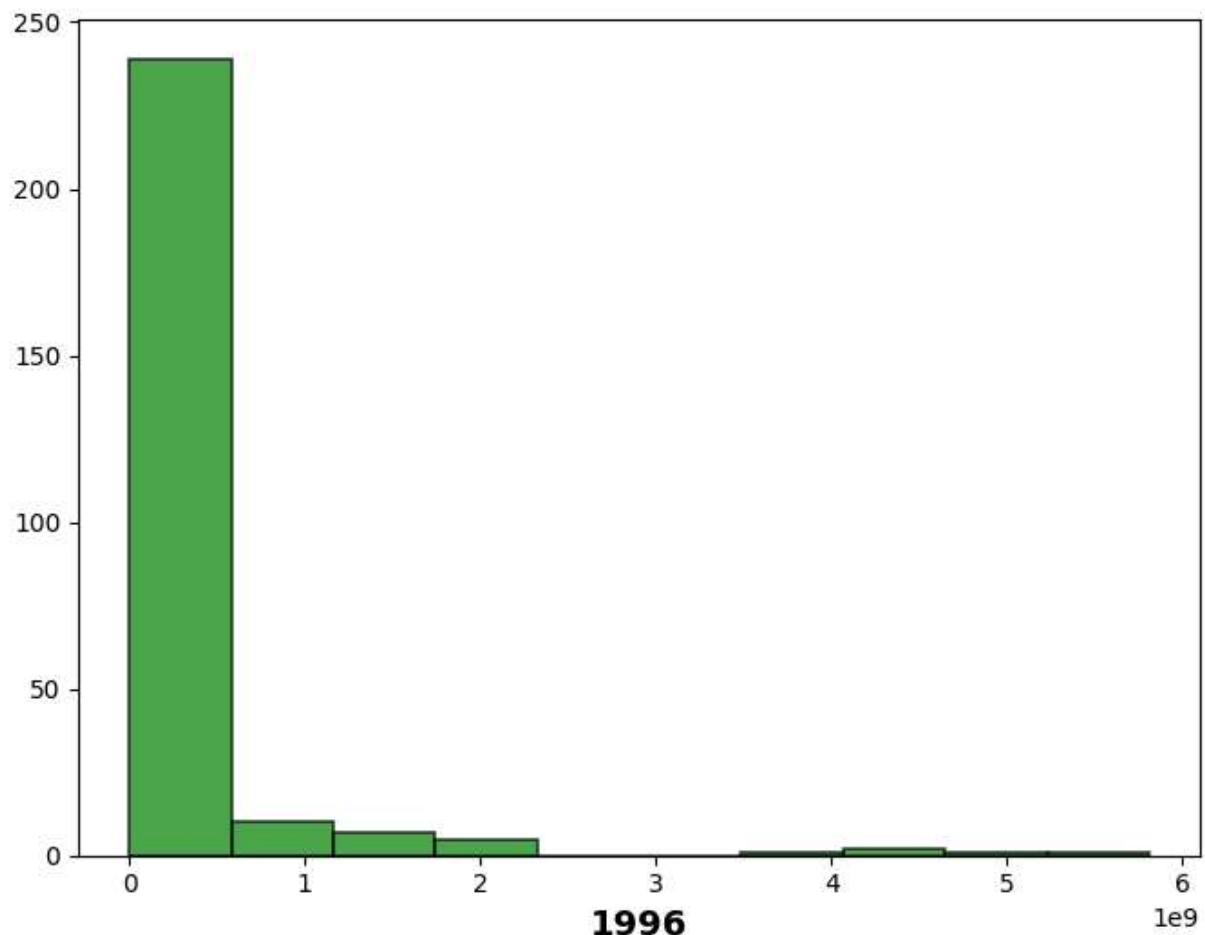


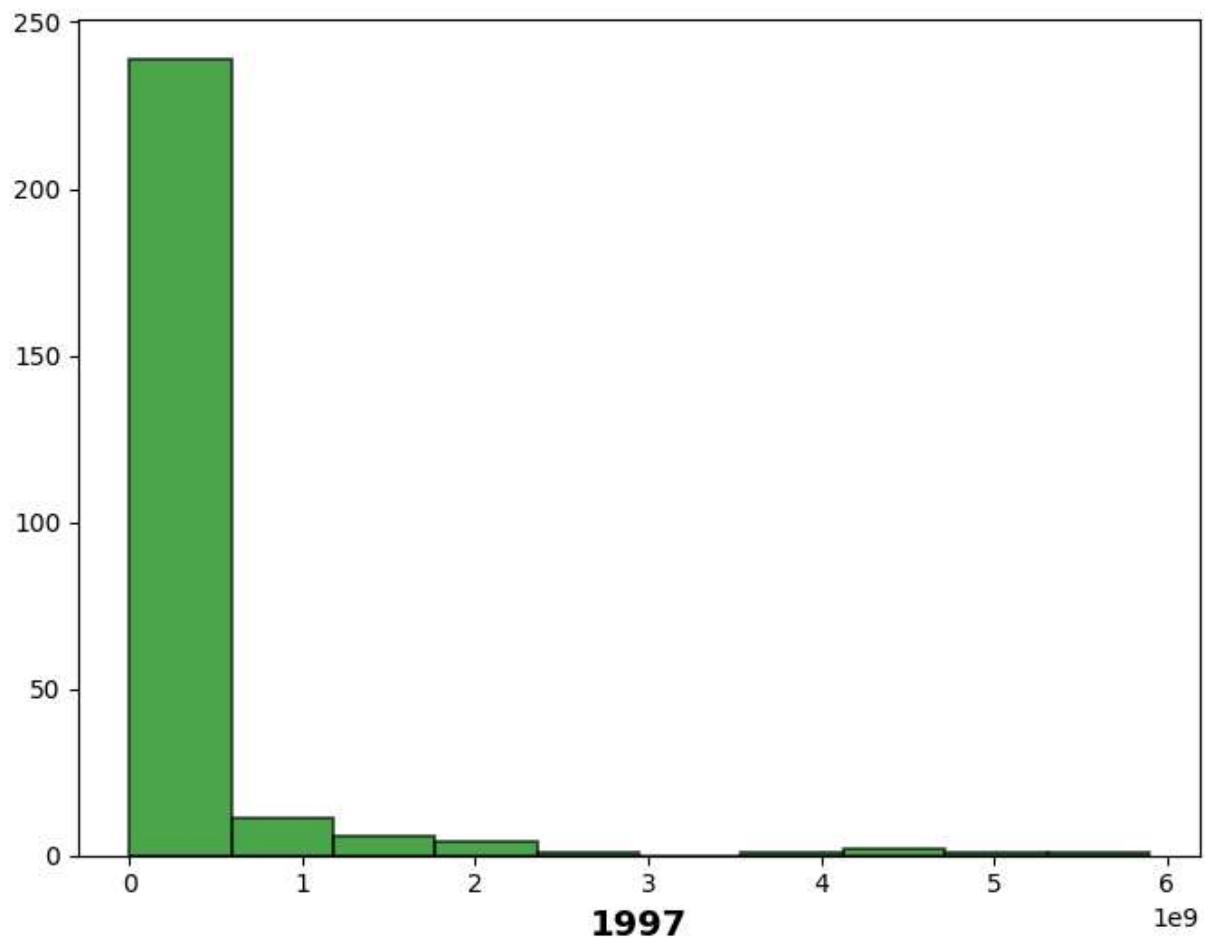


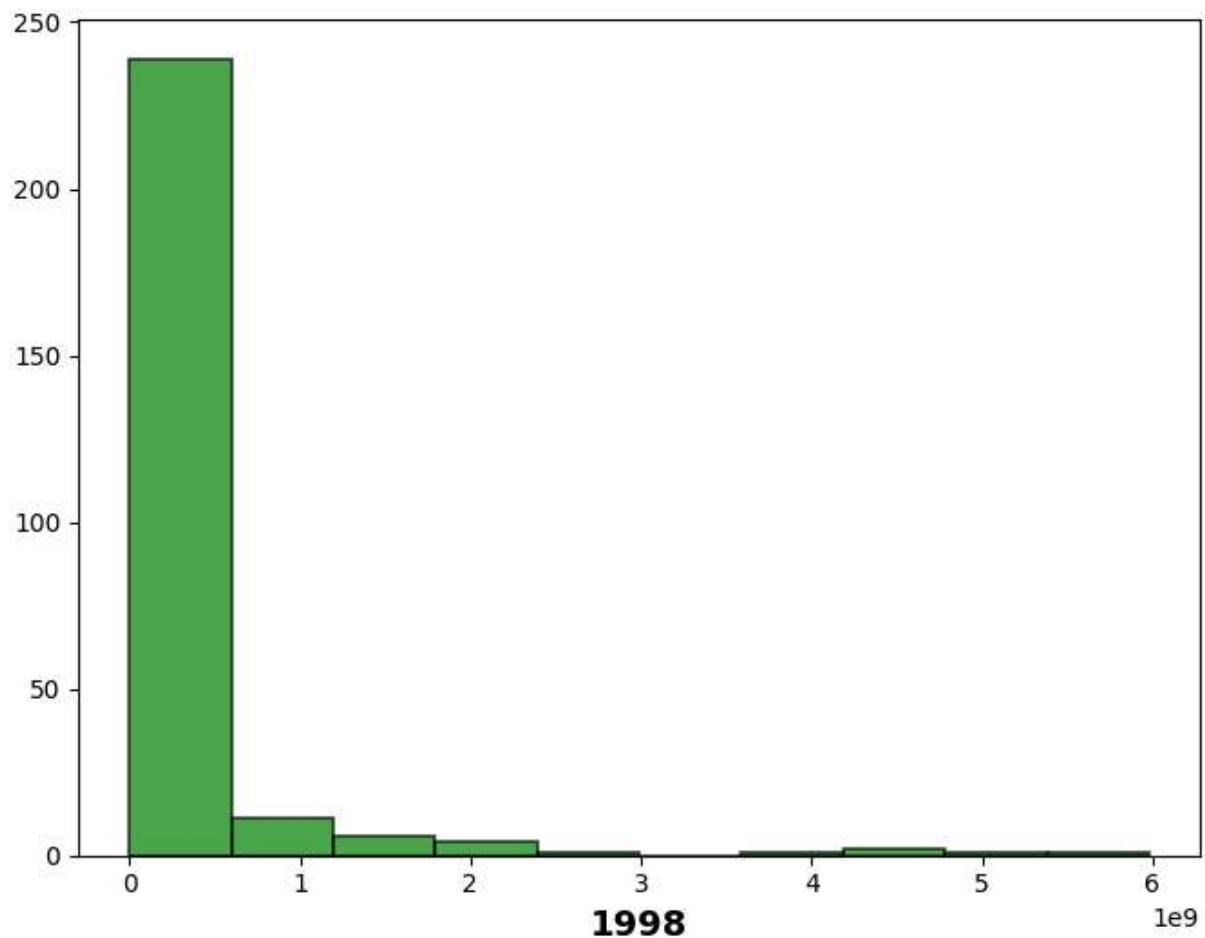


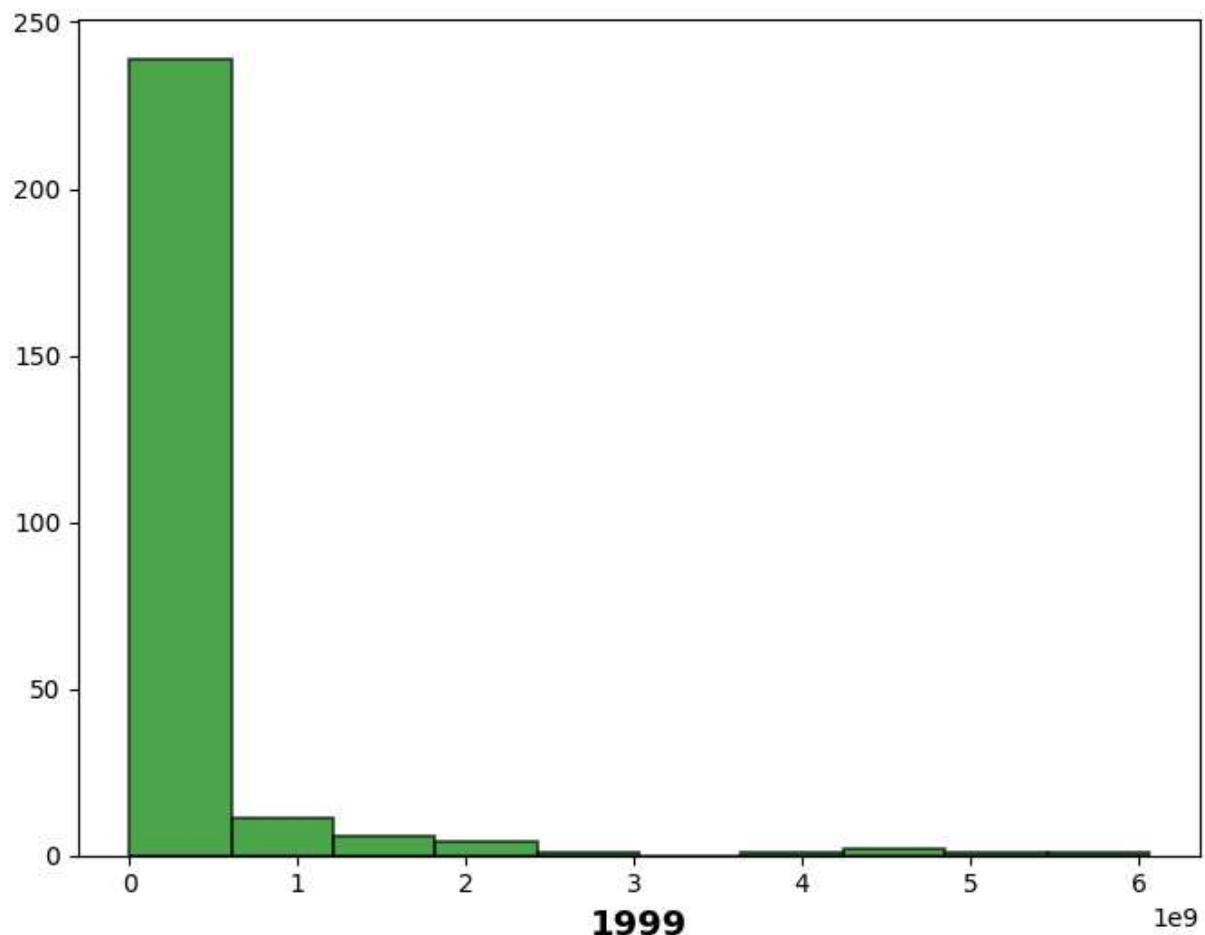


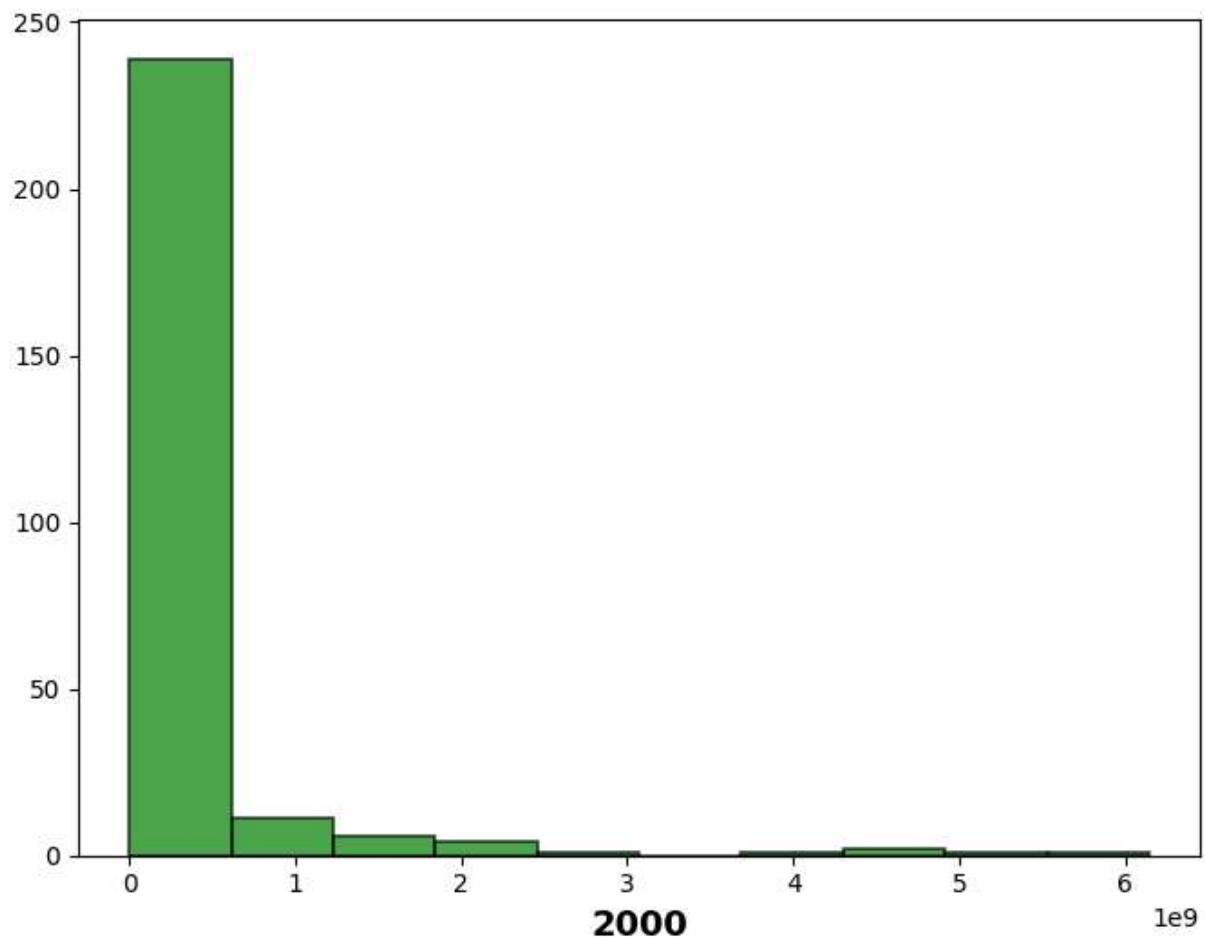


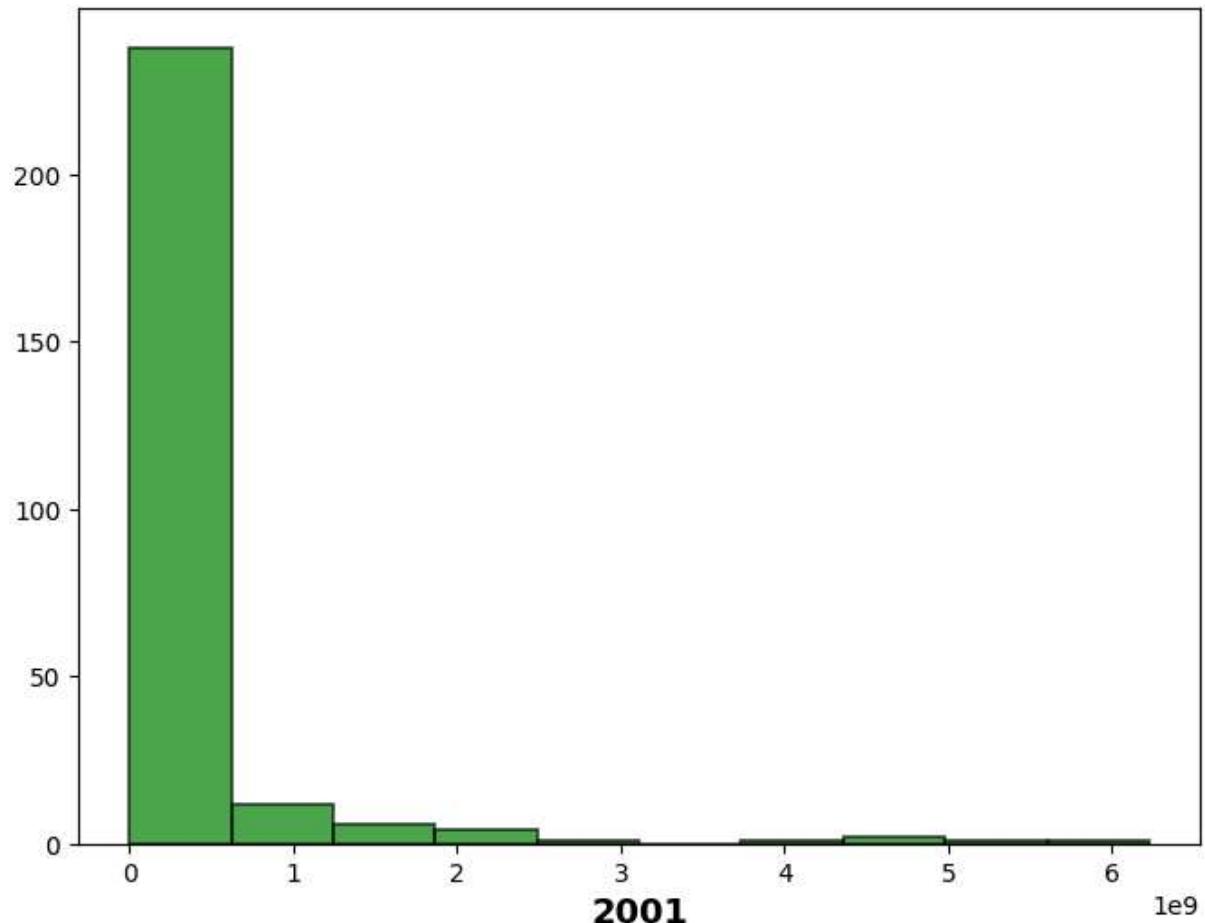


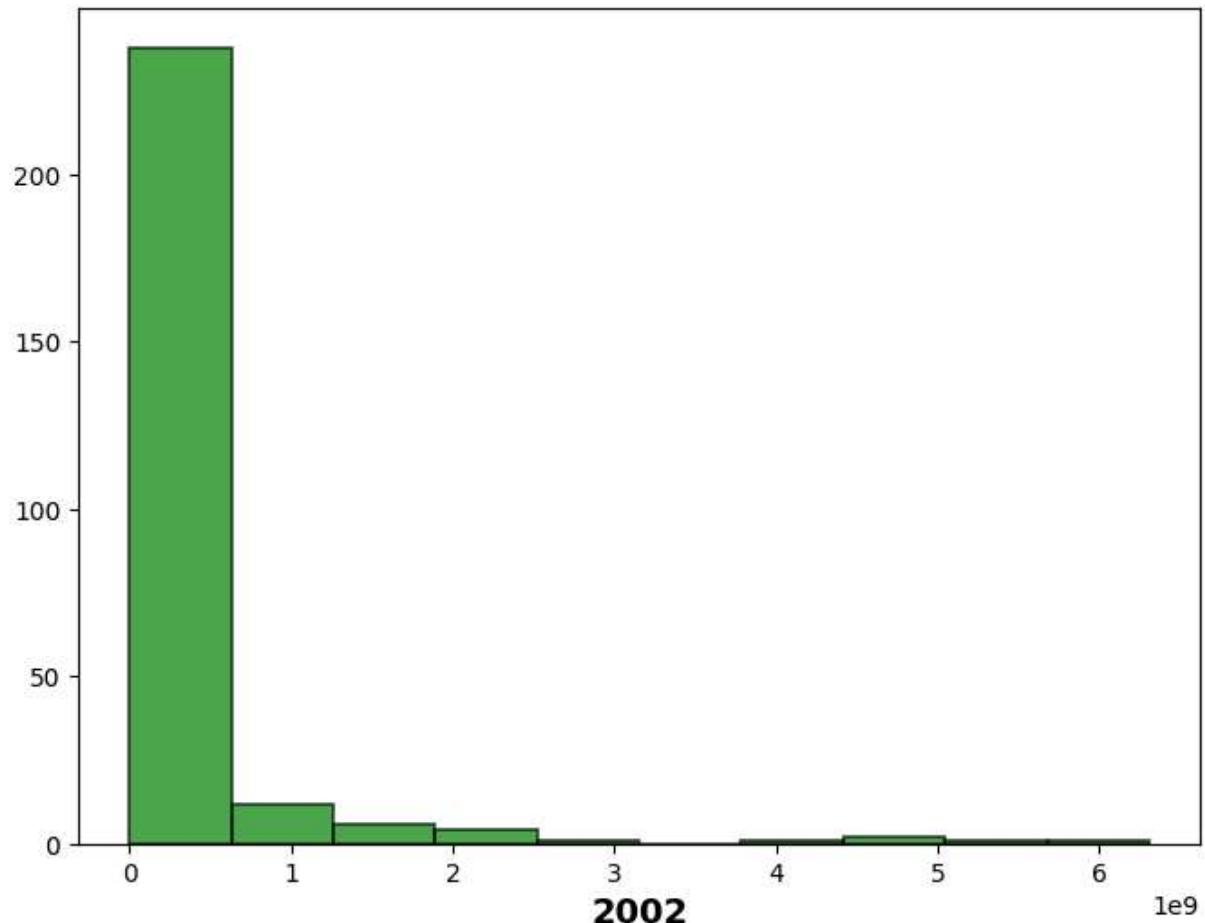


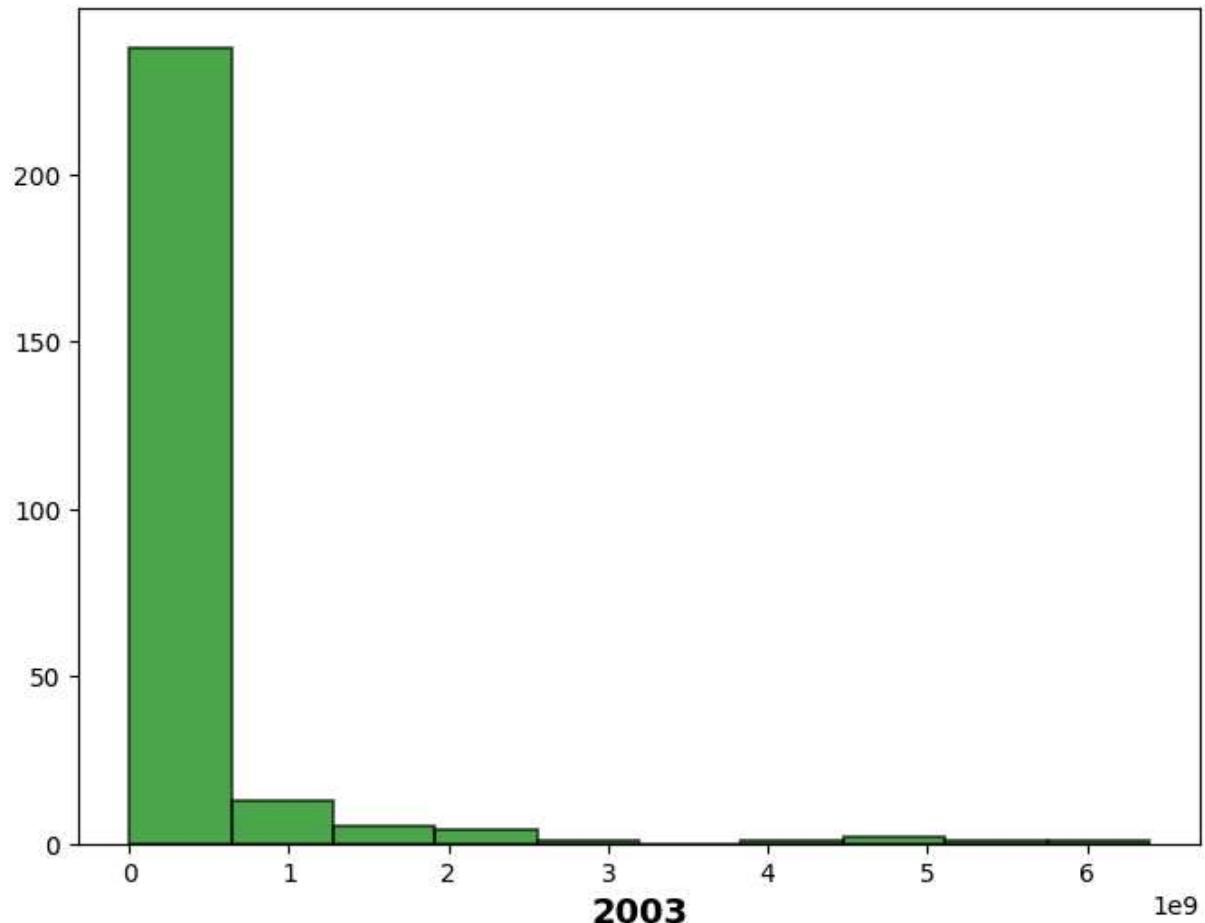


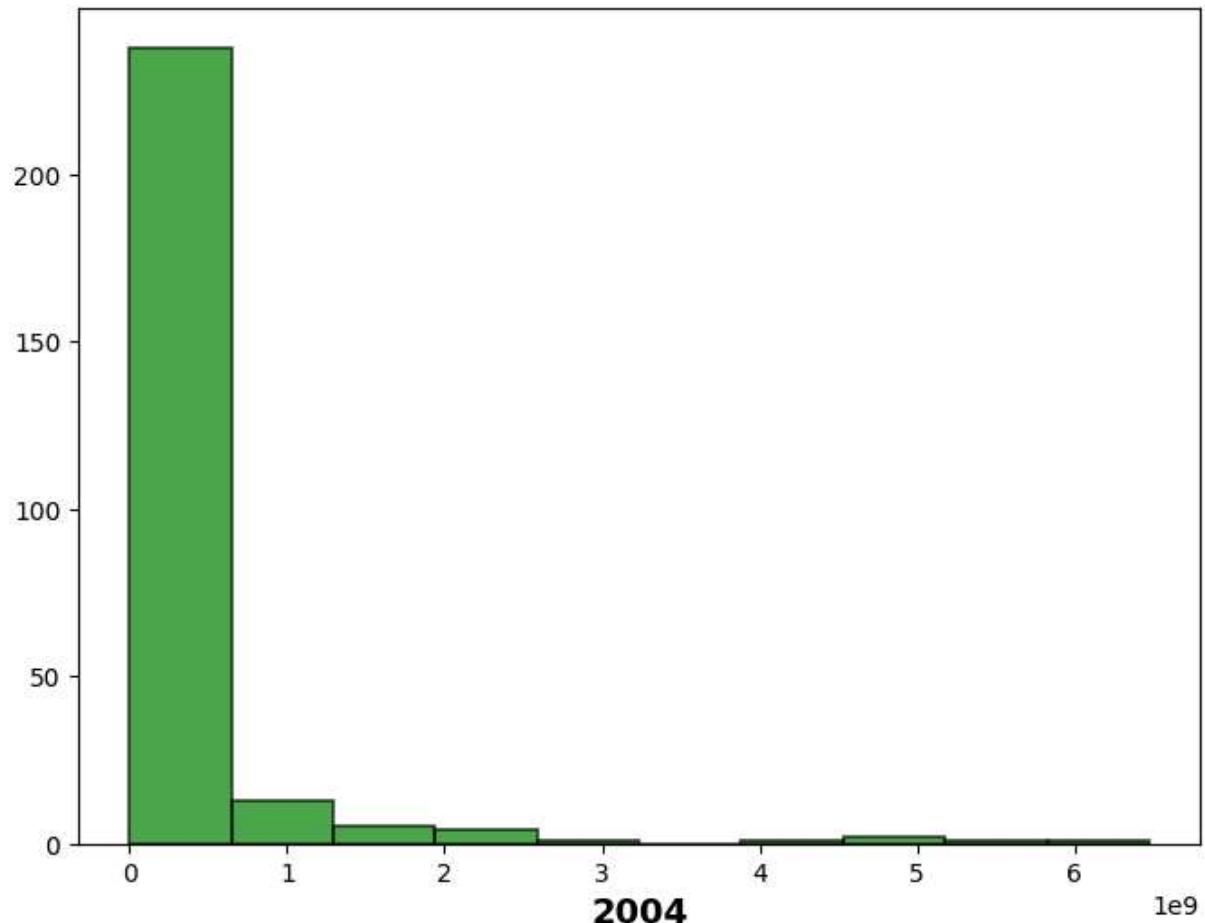


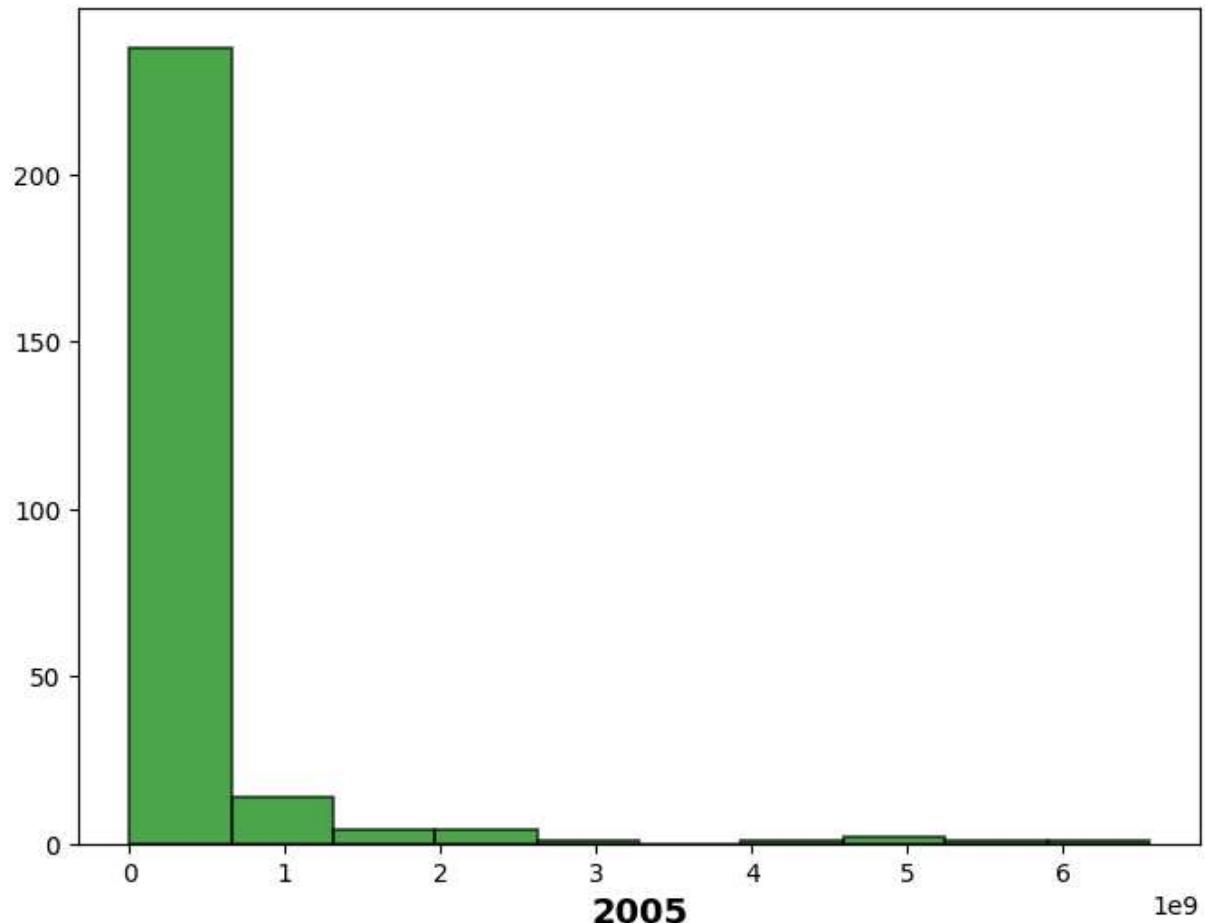


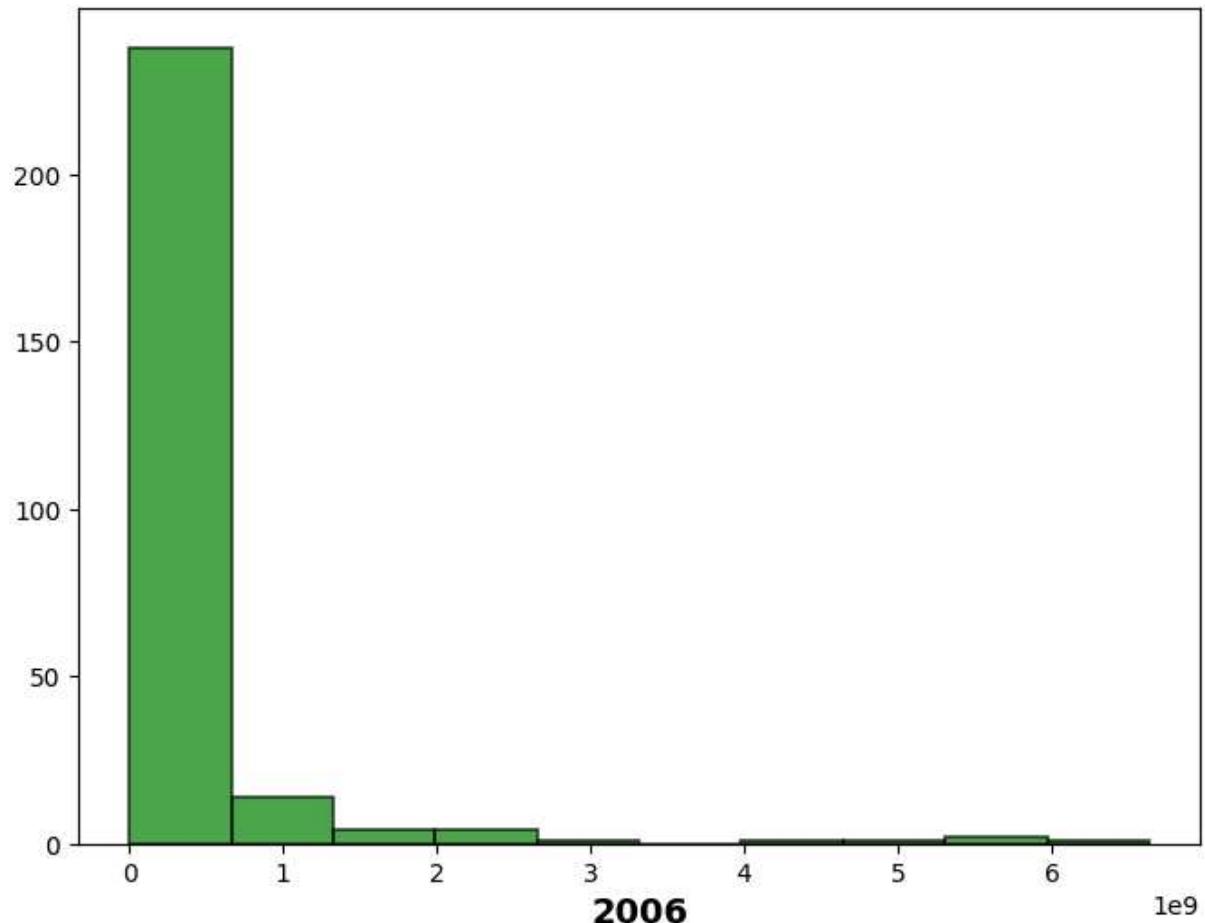


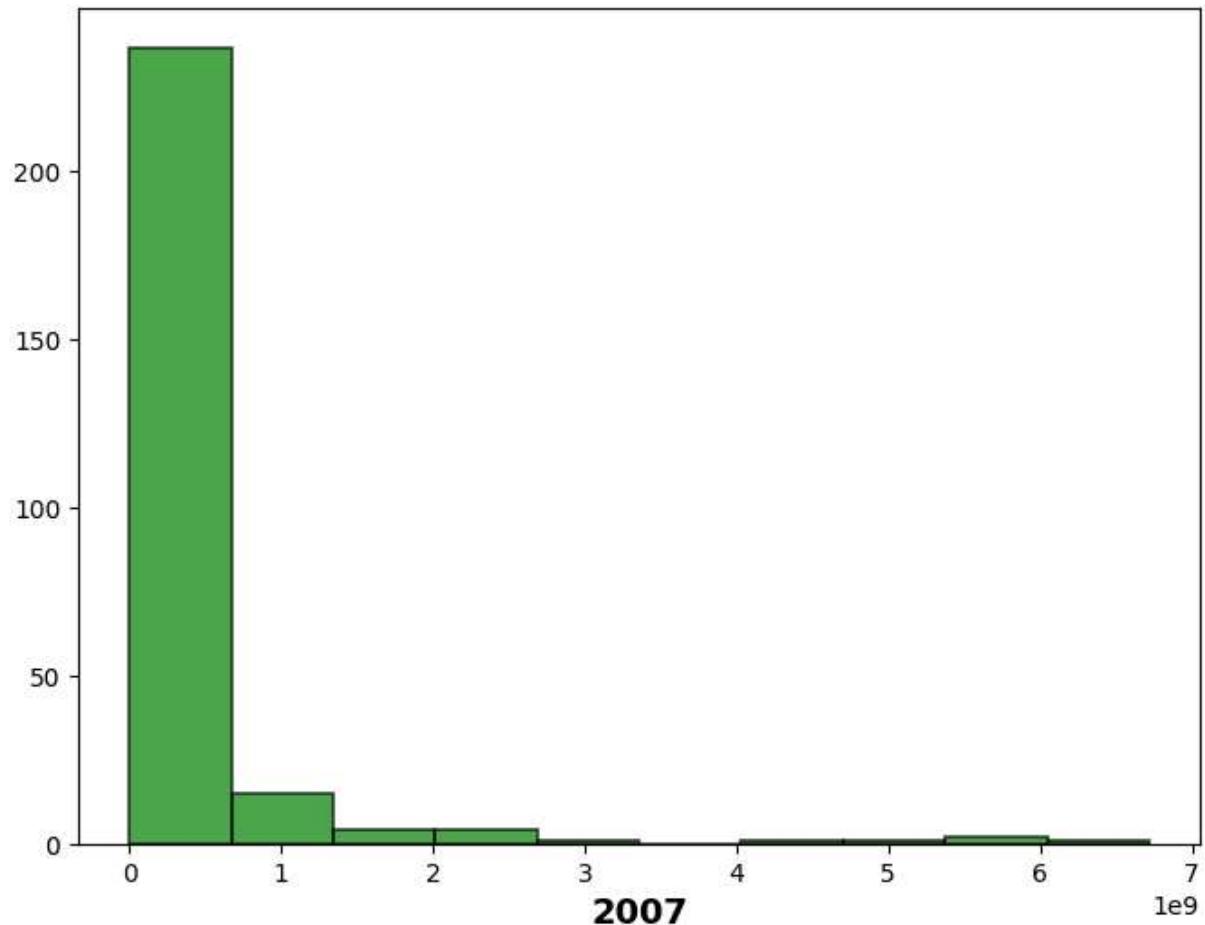


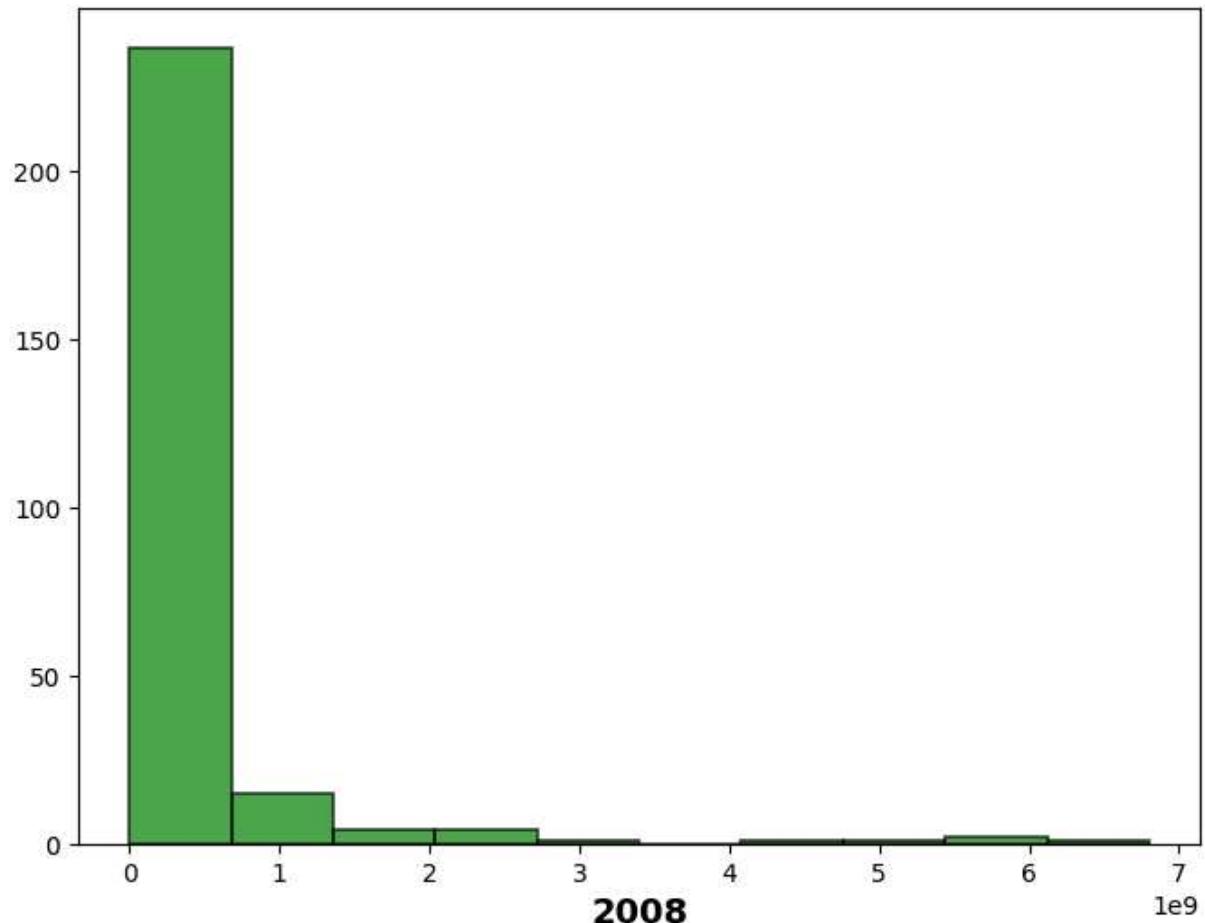


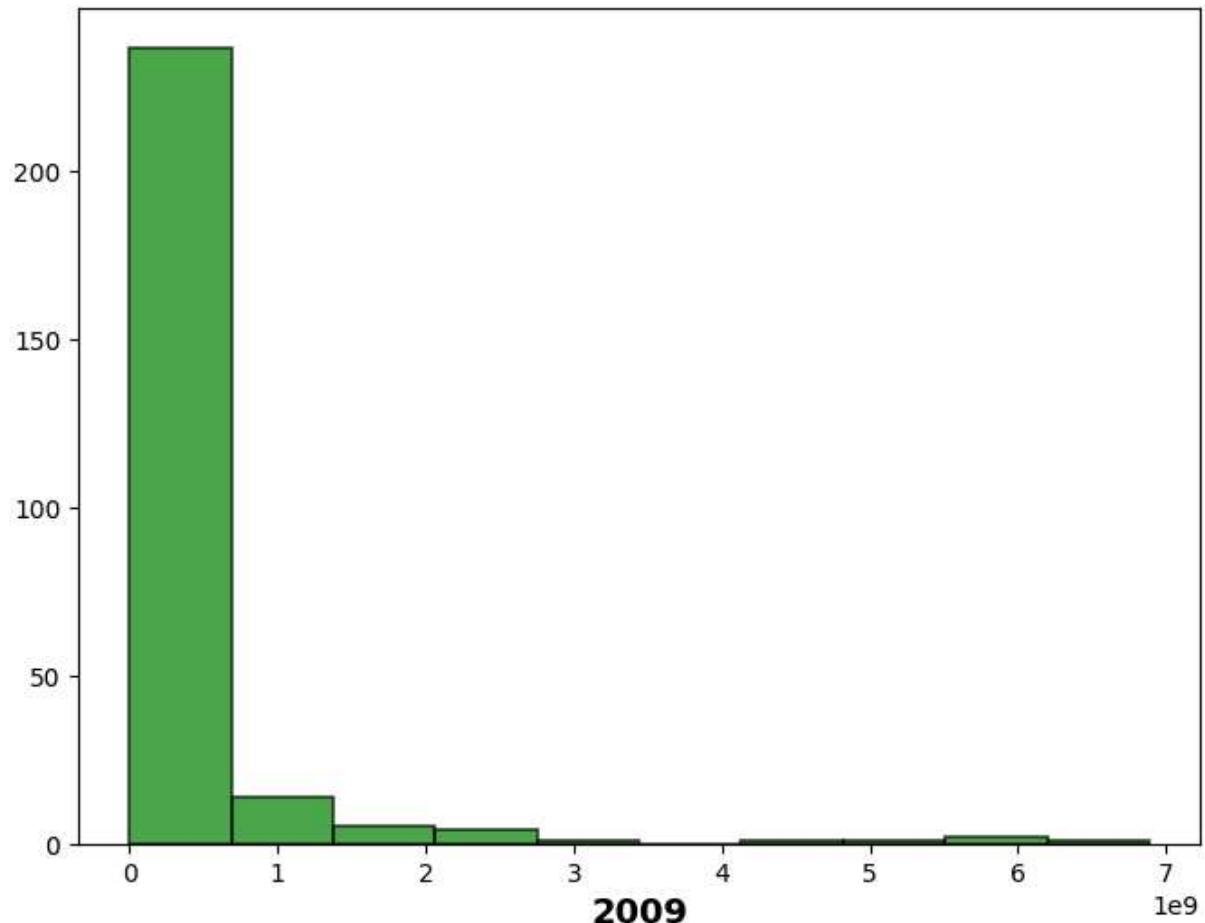


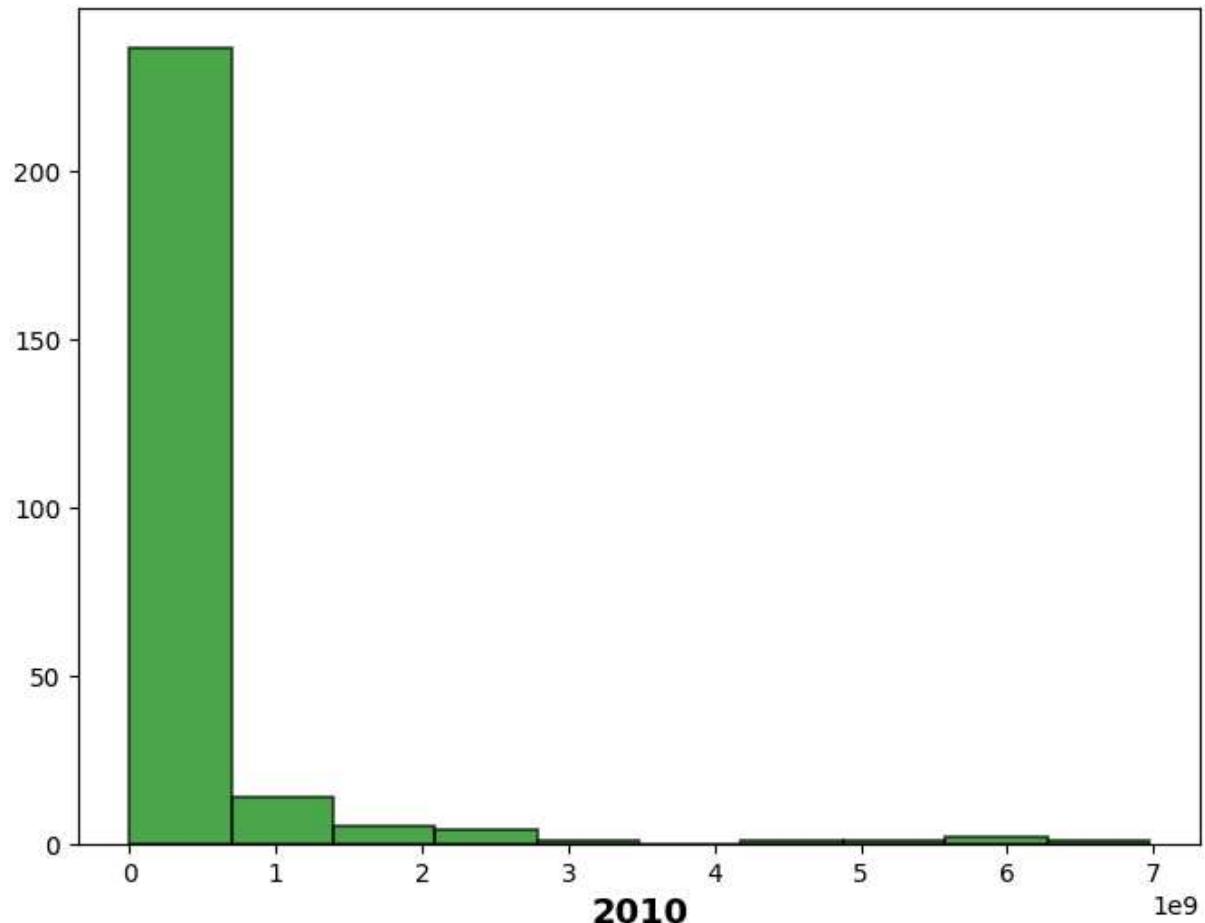


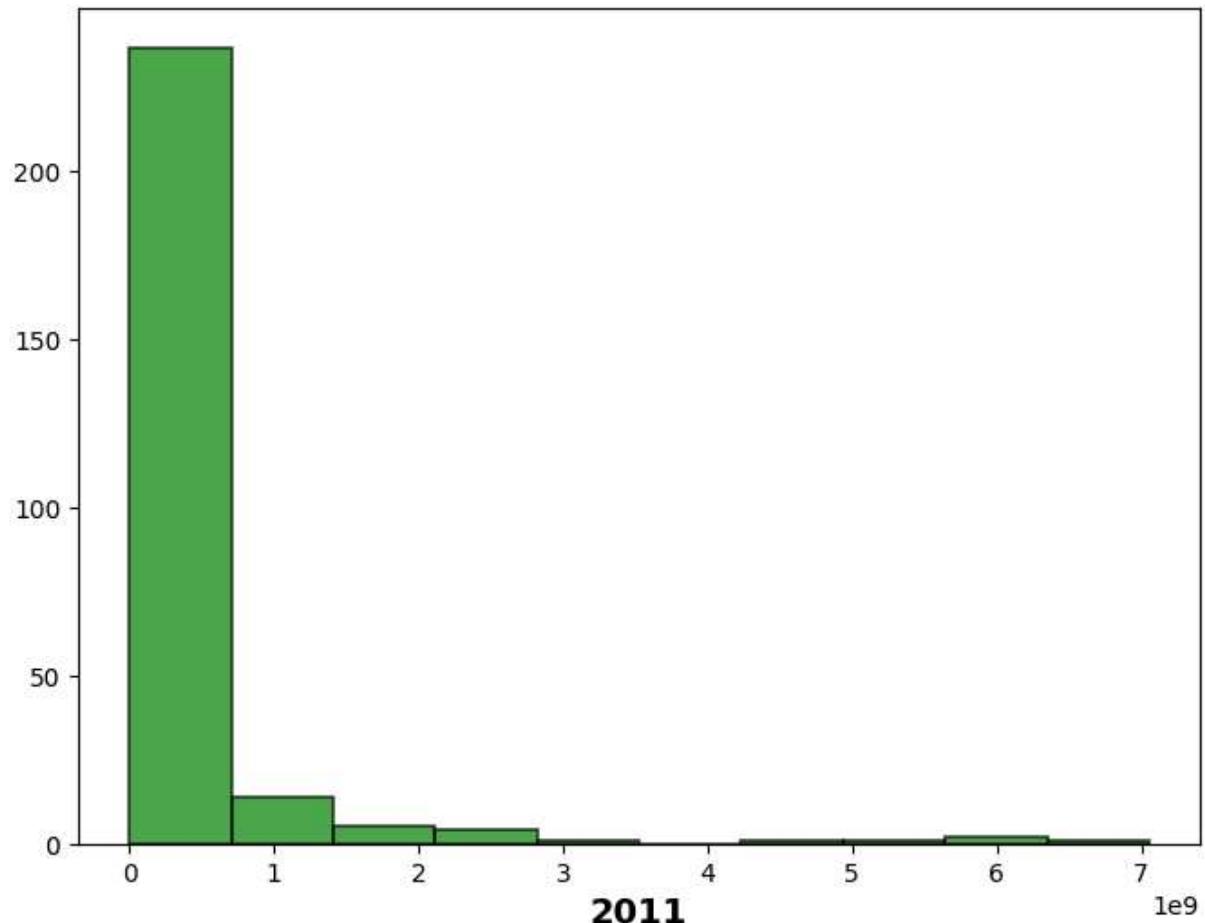


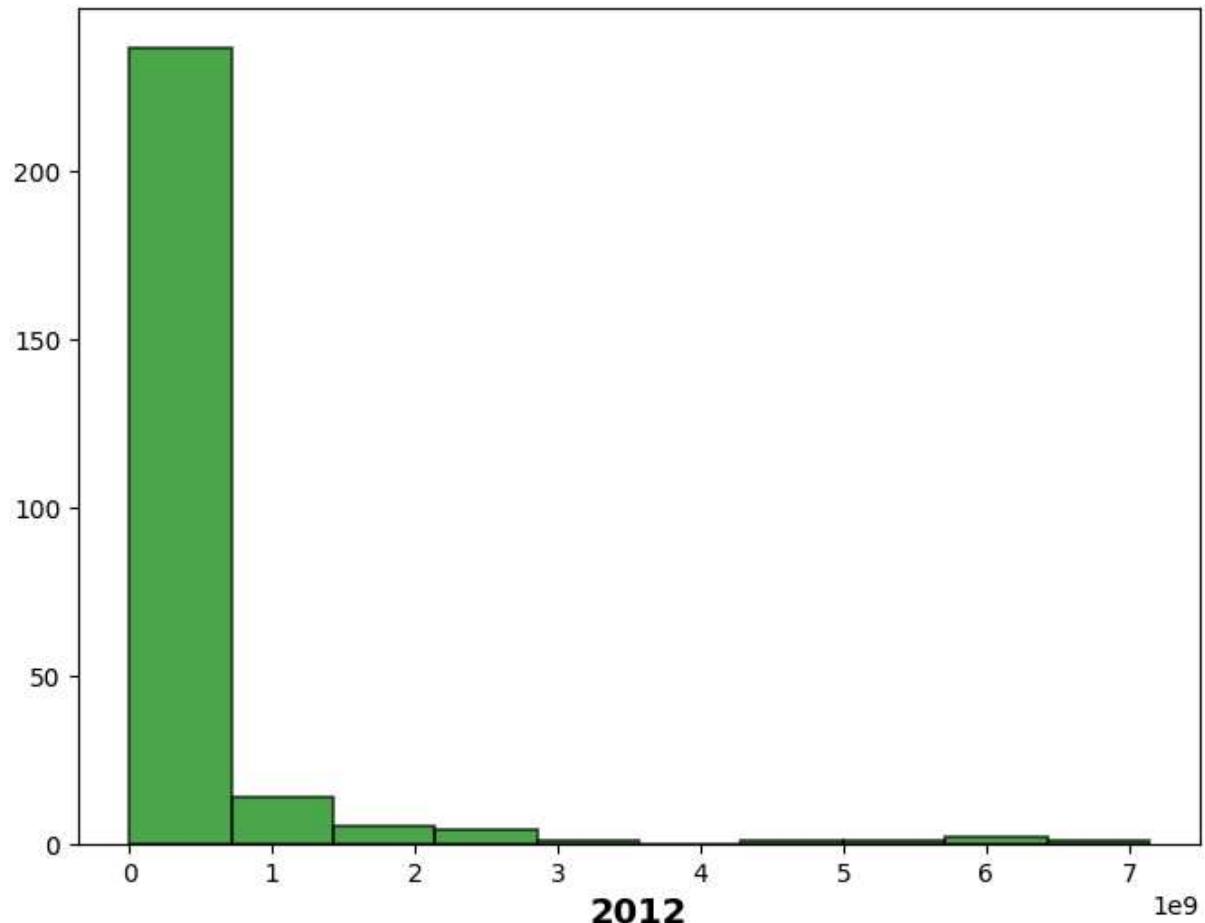


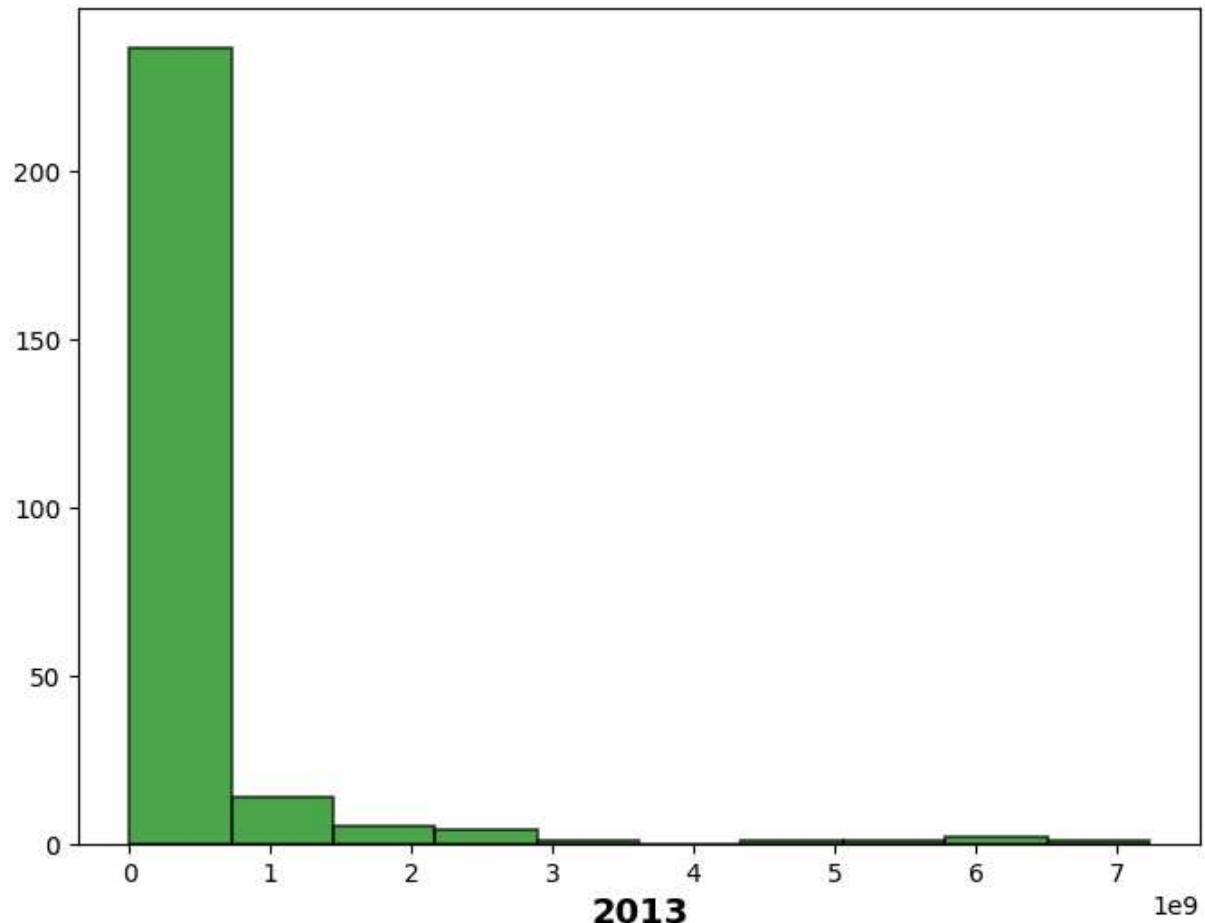


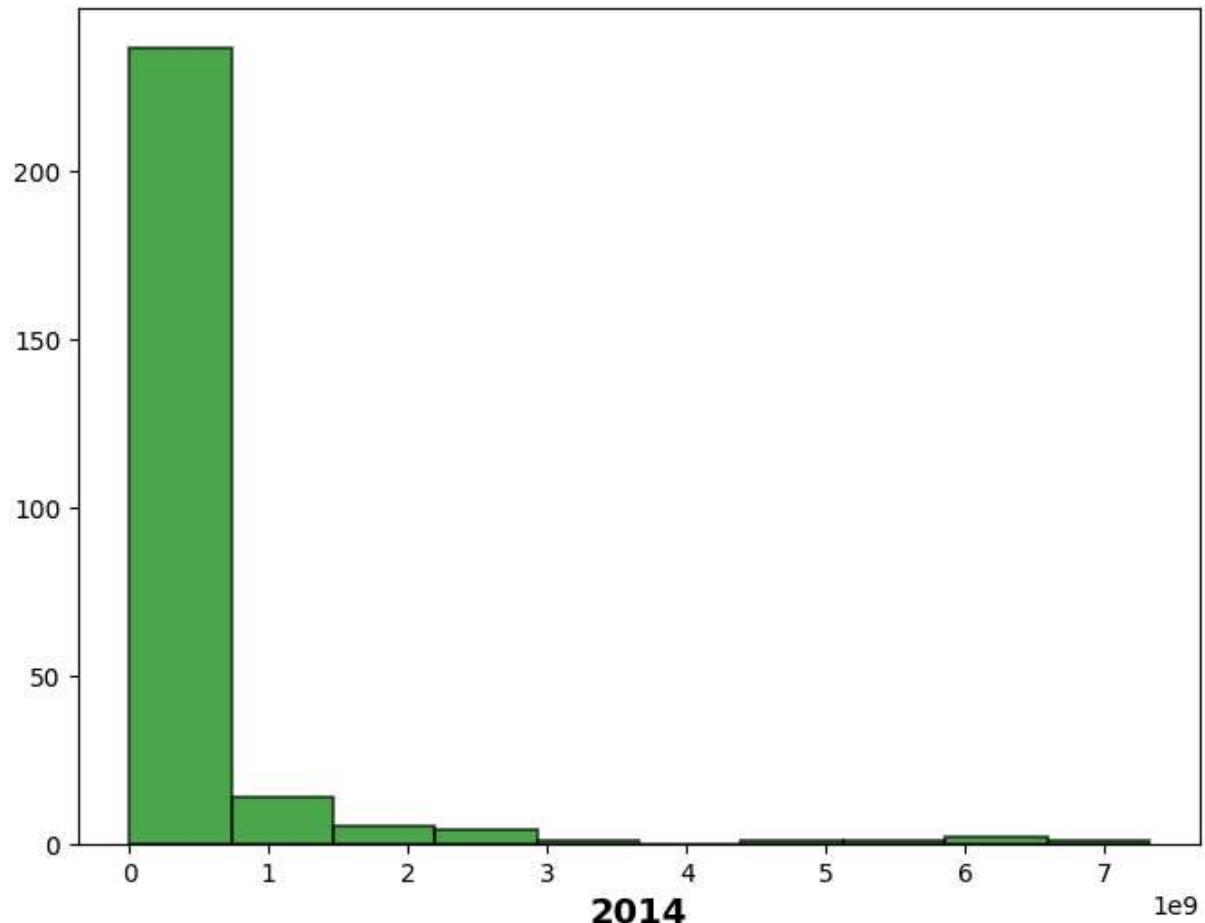


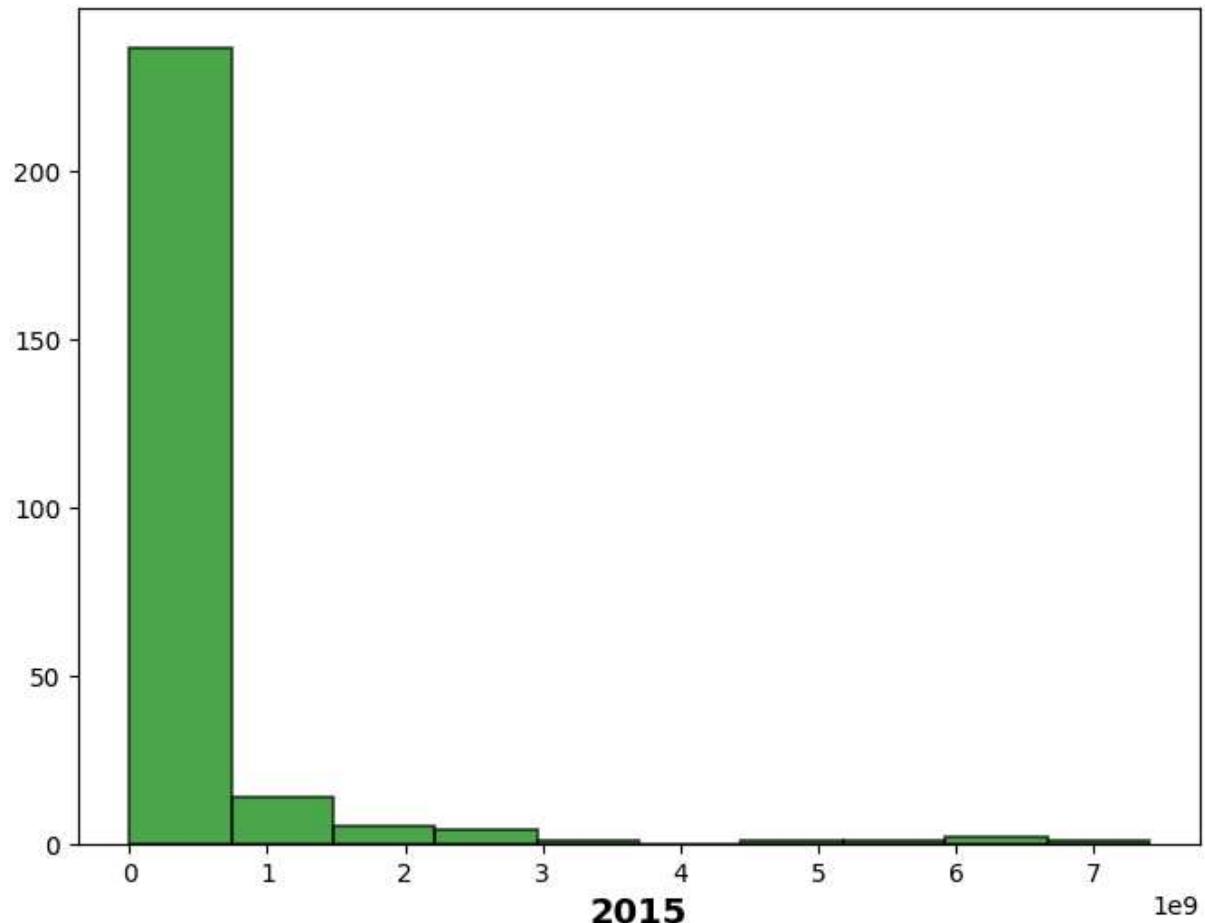


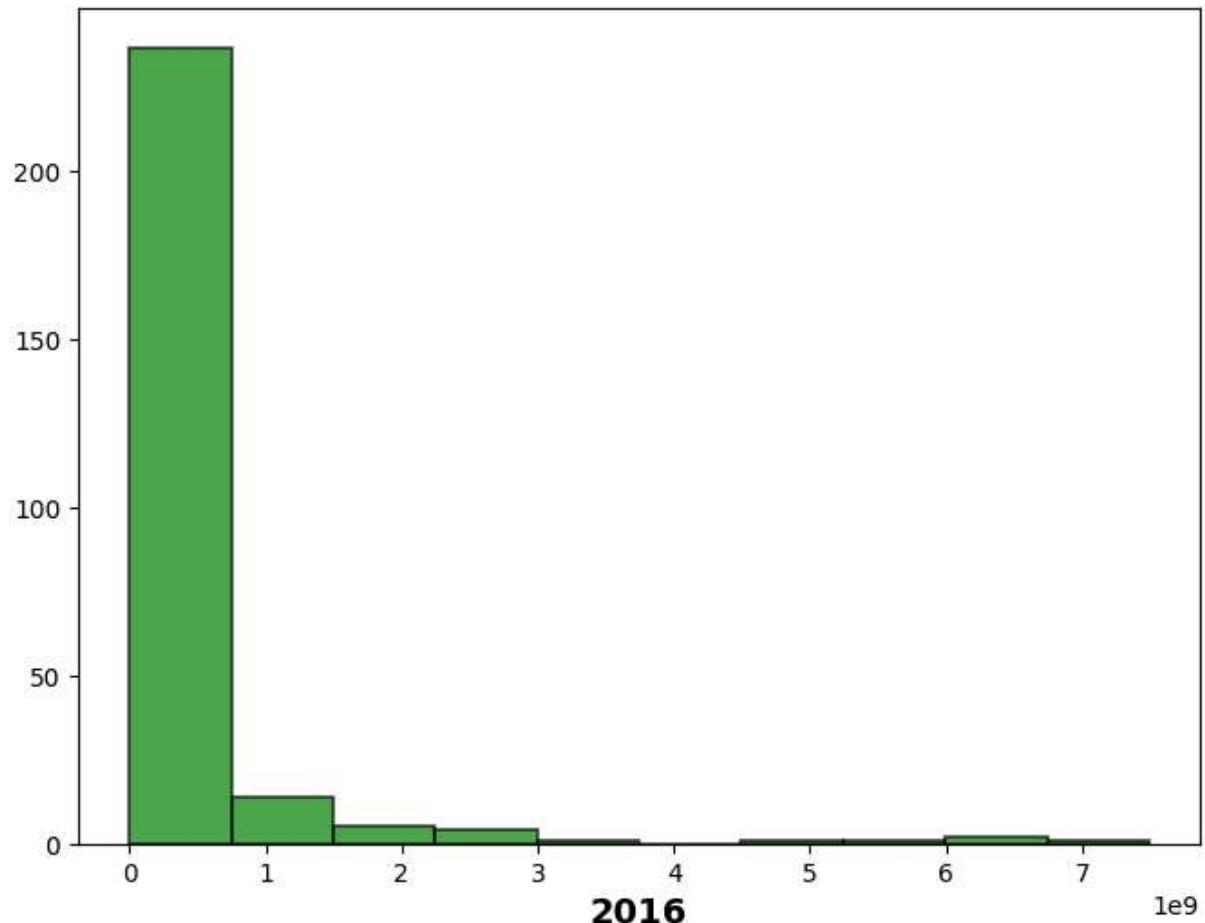


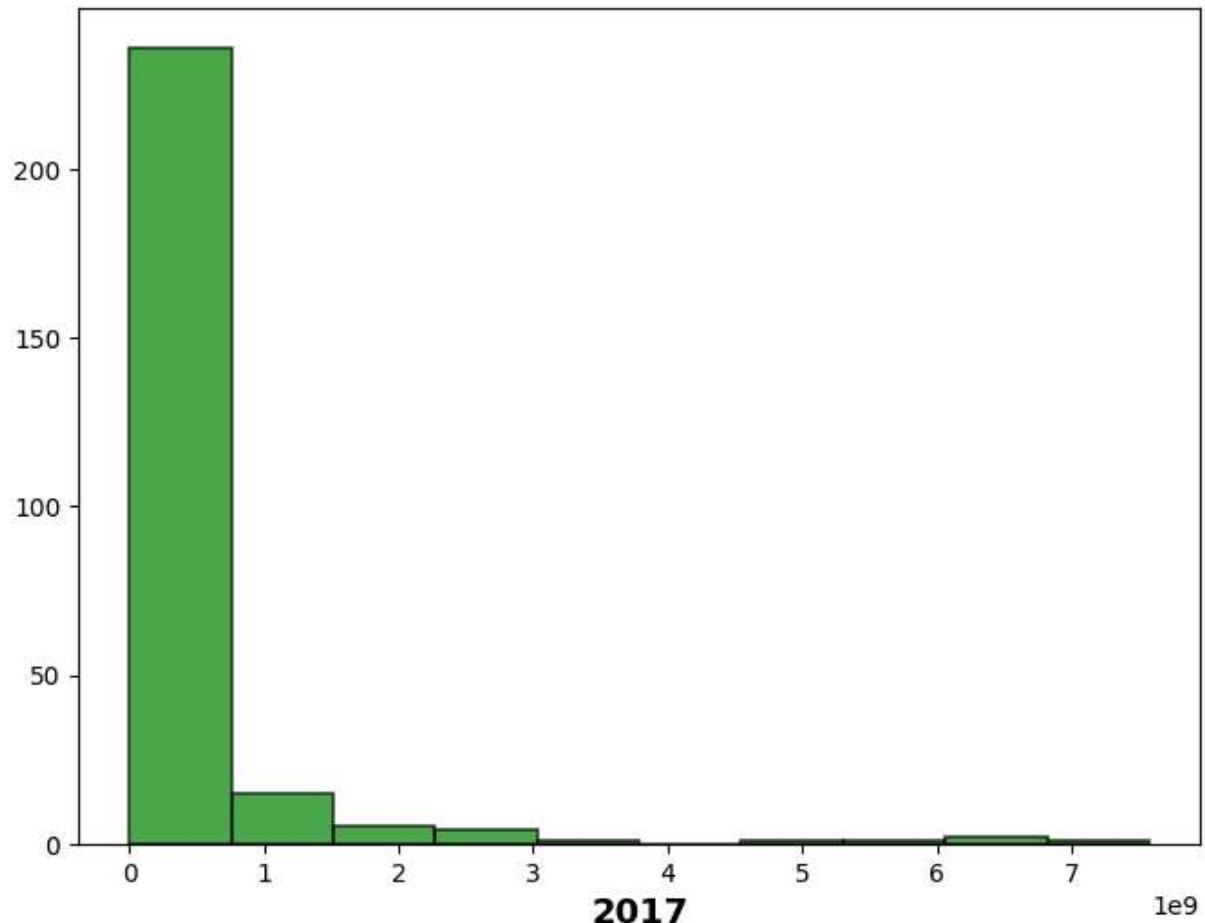


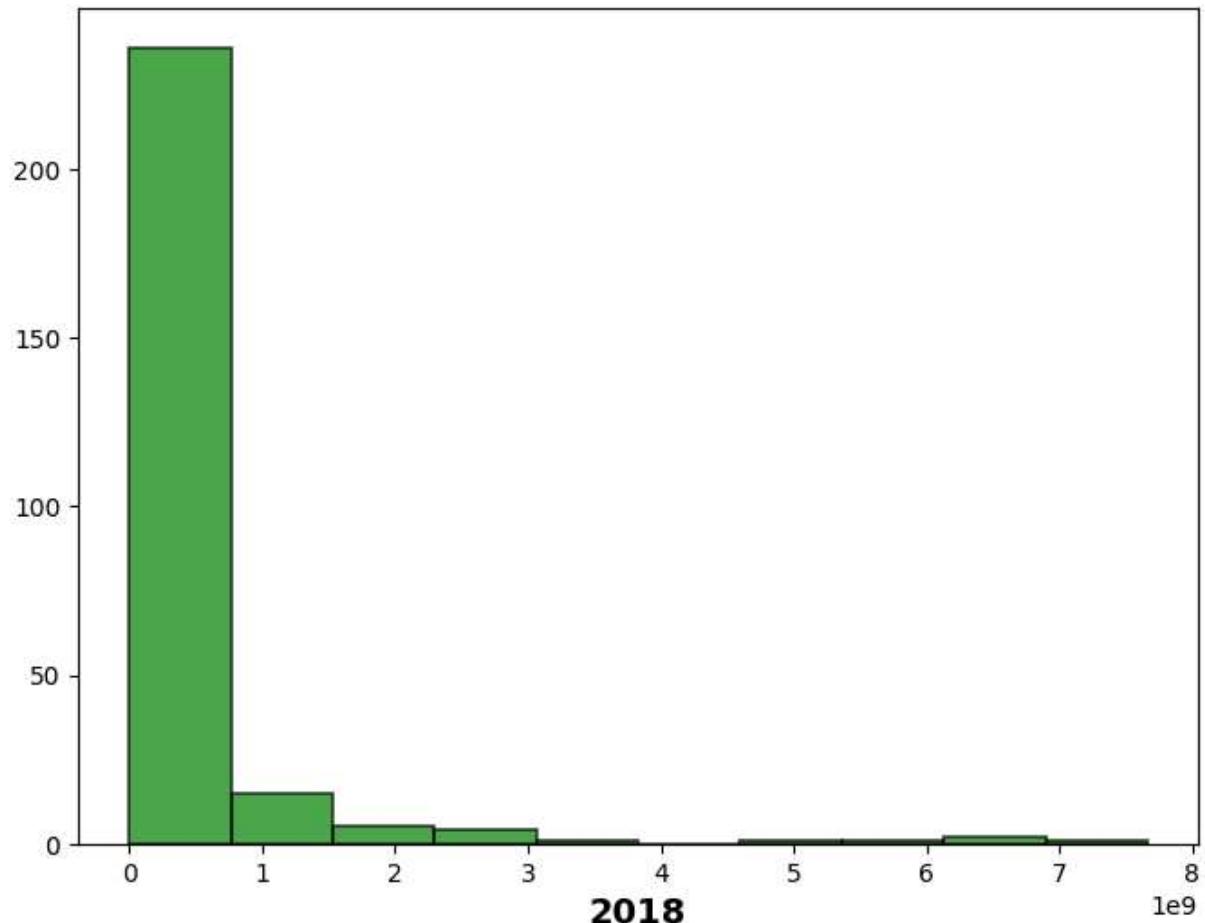


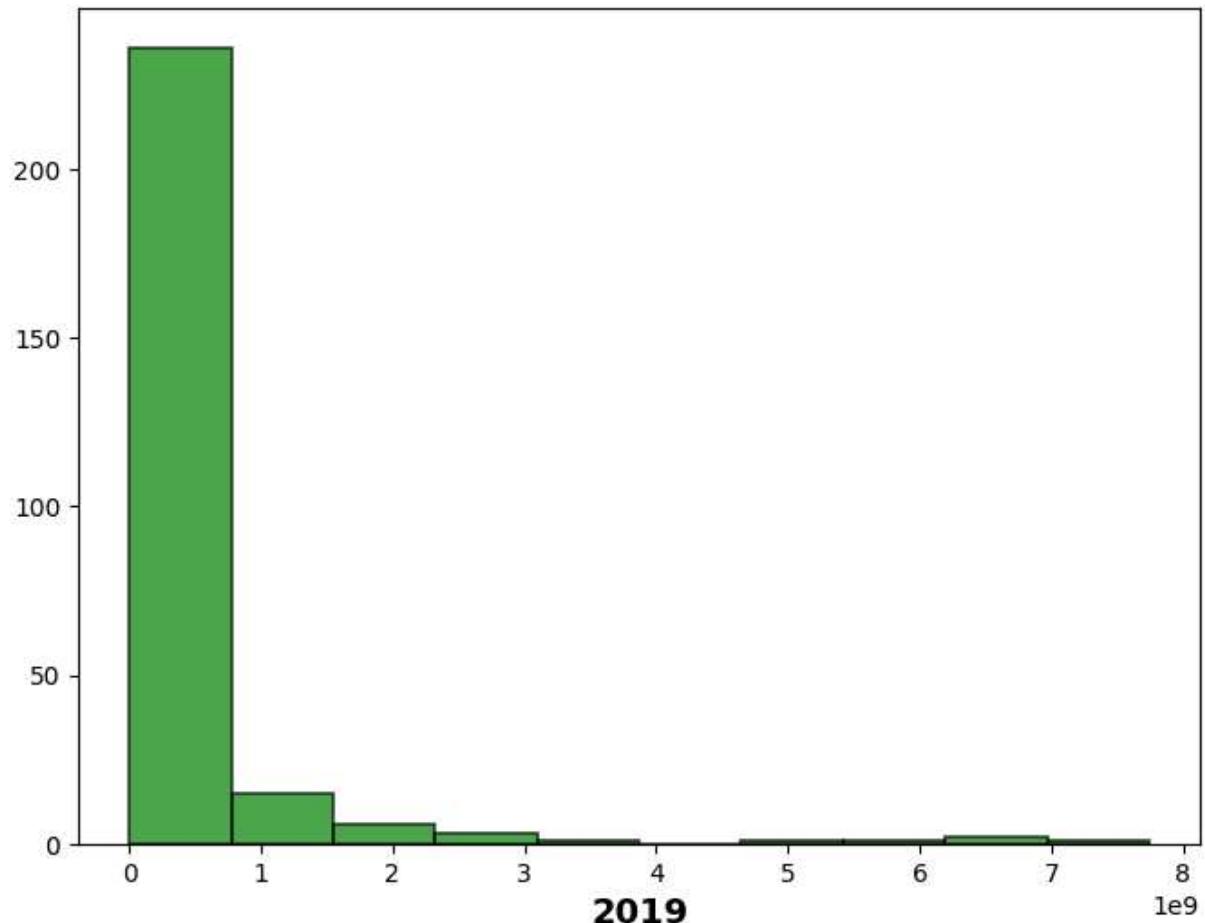


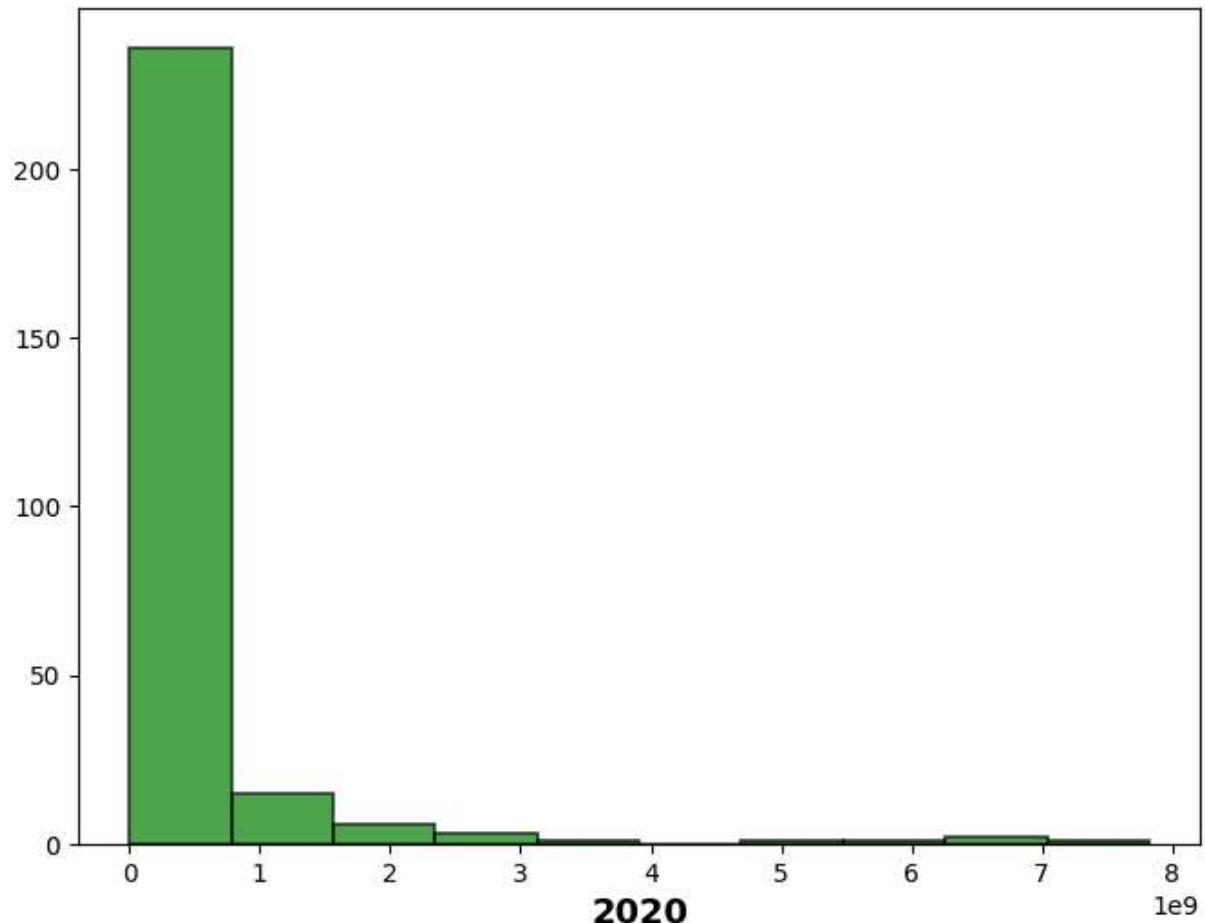


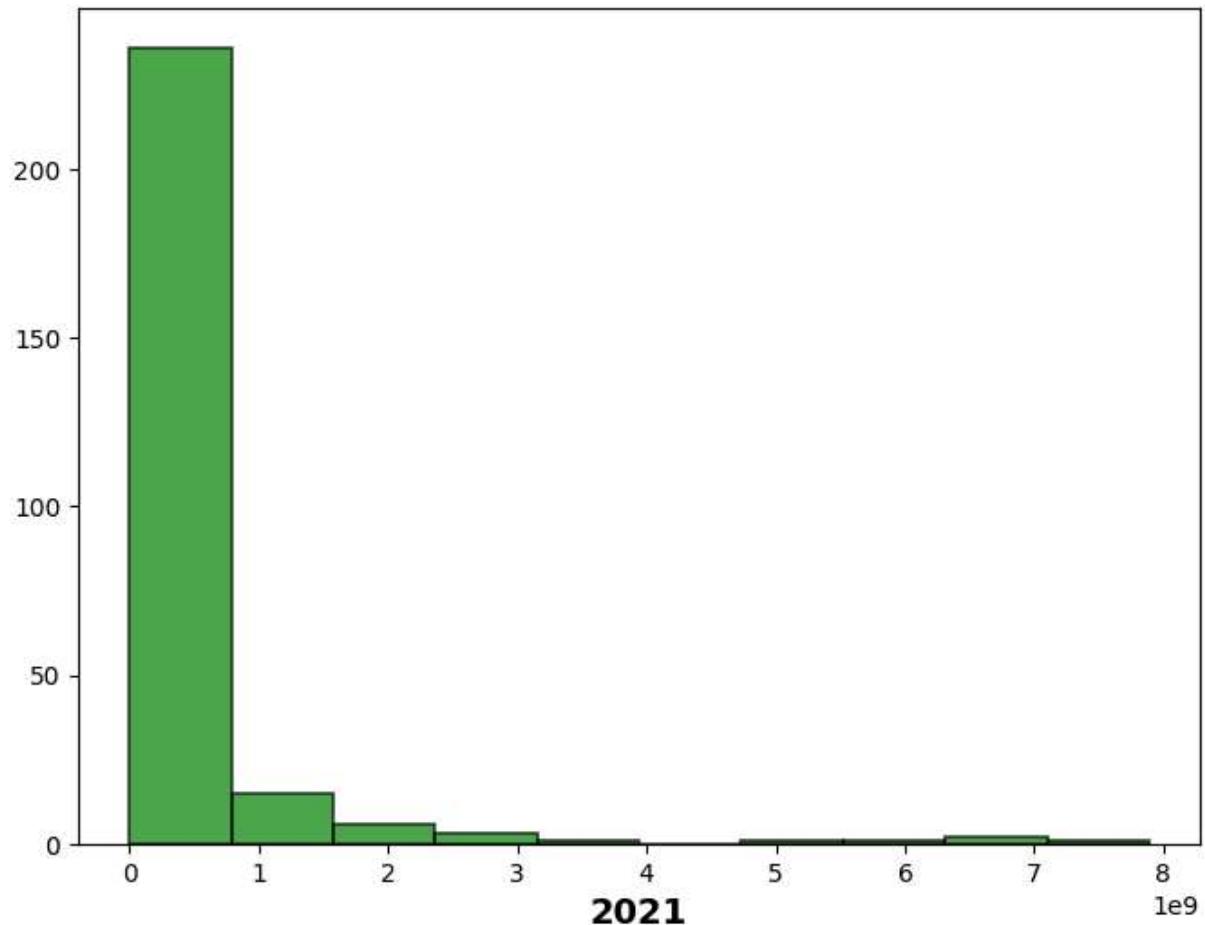


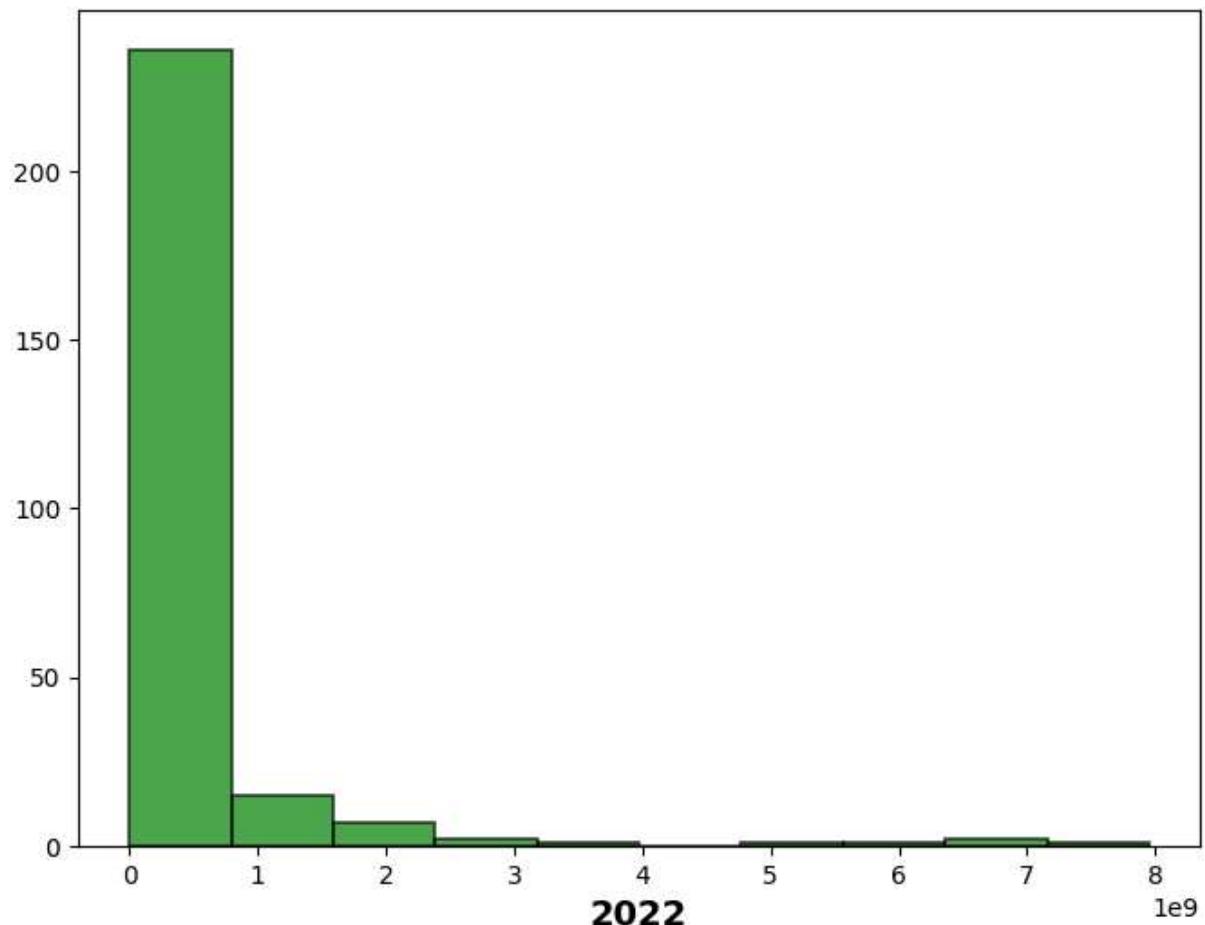


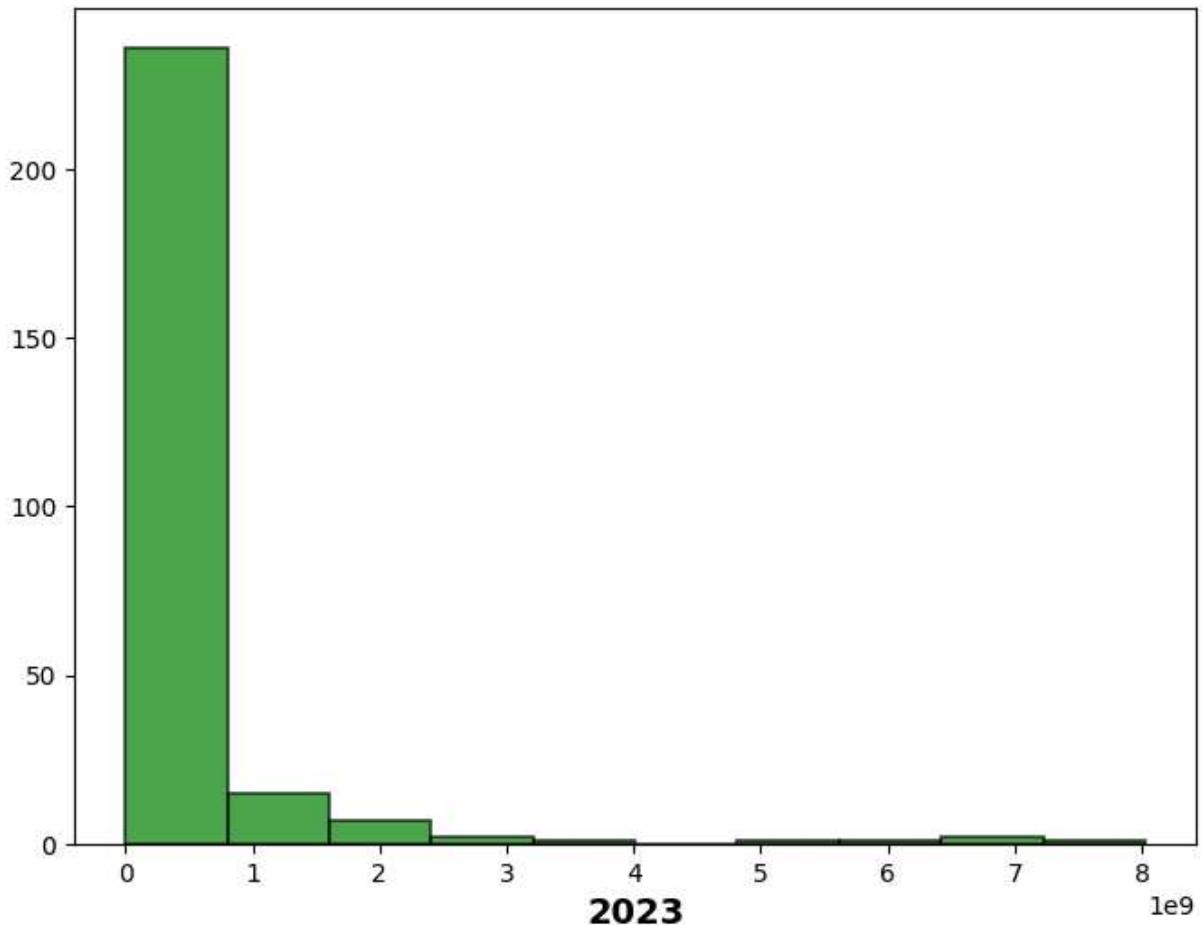












```
In [37]: import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

# Load the dataset
data = pd.read_csv(r'D:/prodigy/task1 world population/API_SP.POP.TOTL_DS2_en_csv_v2_480000.csv')

# Extract the relevant data and transpose it
population_data = data.set_index('Country Name').loc[:, '1960':'2023'].transpose()

# Sum the population values for each year
total_population_per_year = population_data.sum(axis=1)

# Prepare data for plotting
years = total_population_per_year.index
total_values = total_population_per_year.values

# Generate colors for each year
cmap = plt.get_cmap("gist_rainbow")
colors = cmap(np.linspace(0, 1, len(years)))

# Create the plot
plt.figure(figsize=(30, 30))
bars = plt.barh(years, total_values, color=colors)

# Add gridlines to the x-axis
plt.grid(axis='x', linestyle='--', alpha=0.7)
```

```
# Add Labels to each bar
for bar in bars:
    width = bar.get_width()
    plt.text(width, bar.get_y() + bar.get_height() / 2, f'{width:.2f}', ha='left', va='center', fontsize=12, color='black')

# Add Labels and title
plt.xlabel('Total Values', fontsize=20)
plt.ylabel('Year', fontsize=20)
plt.title('Total Population per Year', fontsize=24)

# Show the plot
plt.show()
```

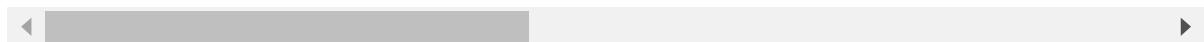


In [26]: country_by_1960=data.sort_values(by='1960').head(20)
country_by_1960

Out[26]:

	Country Name	1960	1961	1962	1963	1964	1965	1966	1967
225	Sint Maarten (Dutch part)	2646.0	2888.0	3171.0	3481.0	3811.0	4161.0	4531.0	4930.0
147	St. Martin (French part)	4135.0	4258.0	4388.0	4524.0	4666.0	4832.0	5044.0	5294.0
179	Nauru	4582.0	4753.0	4950.0	5198.0	5484.0	5804.0	6021.0	6114.0
245	Tuvalu	5404.0	5436.0	5471.0	5503.0	5525.0	5548.0	5591.0	5657.0
228	Turks and Caicos Islands	5604.0	5625.0	5633.0	5634.0	5642.0	5650.0	5652.0	5662.0
255	British Virgin Islands	7850.0	7885.0	7902.0	7919.0	7949.0	8018.0	8139.0	8337.0
52	Cayman Islands	8473.0	8626.0	8799.0	8985.0	9172.0	9366.0	9566.0	9771.0
164	Northern Mariana Islands	8702.0	8965.0	9252.0	9561.0	9890.0	10229.0	10577.0	10720.0 1
6	Andorra	9443.0	10216.0	11014.0	11839.0	12690.0	13563.0	14546.0	15745.0 1
188	Palau	9446.0	9639.0	9851.0	10076.0	10318.0	10563.0	10813.0	10992.0 1
155	Marshall Islands	15374.0	15867.0	16387.0	16947.0	17537.0	18154.0	18794.0	19665.0 2
212	San Marino	15556.0	15895.0	16242.0	16583.0	16926.0	17273.0	17588.0	17907.0 1
137	Liechtenstein	16472.0	16834.0	17221.0	17625.0	18058.0	18500.0	18957.0	19467.0 2
11	American Samoa	20085.0	20626.0	21272.0	21949.0	22656.0	23391.0	24122.0	24848.0 2
149	Monaco	21797.0	21907.0	22106.0	22442.0	22766.0	23022.0	23198.0	23281.0 2
84	Gibraltar	21822.0	21907.0	22249.0	22796.0	23347.0	23910.0	24477.0	25047.0 2
256	Virgin Islands (U.S.)	32500.0	34300.0	35000.0	39800.0	40800.0	43500.0	46200.0	49100.0 5
91	Greenland	32500.0	33700.0	35000.0	36400.0	37600.0	39200.0	40500.0	41900.0 4
78	Faroe Islands	34154.0	34572.0	34963.0	35385.0	35841.0	36346.0	36825.0	37234.0 3
200	Qatar	36385.0	40111.0	45123.0	50950.0	57531.0	64843.0	73102.0	82517.0 9

20 rows × 65 columns



In [30]: country_by_1960_t=country_by_1960.set_index('Country Name').T

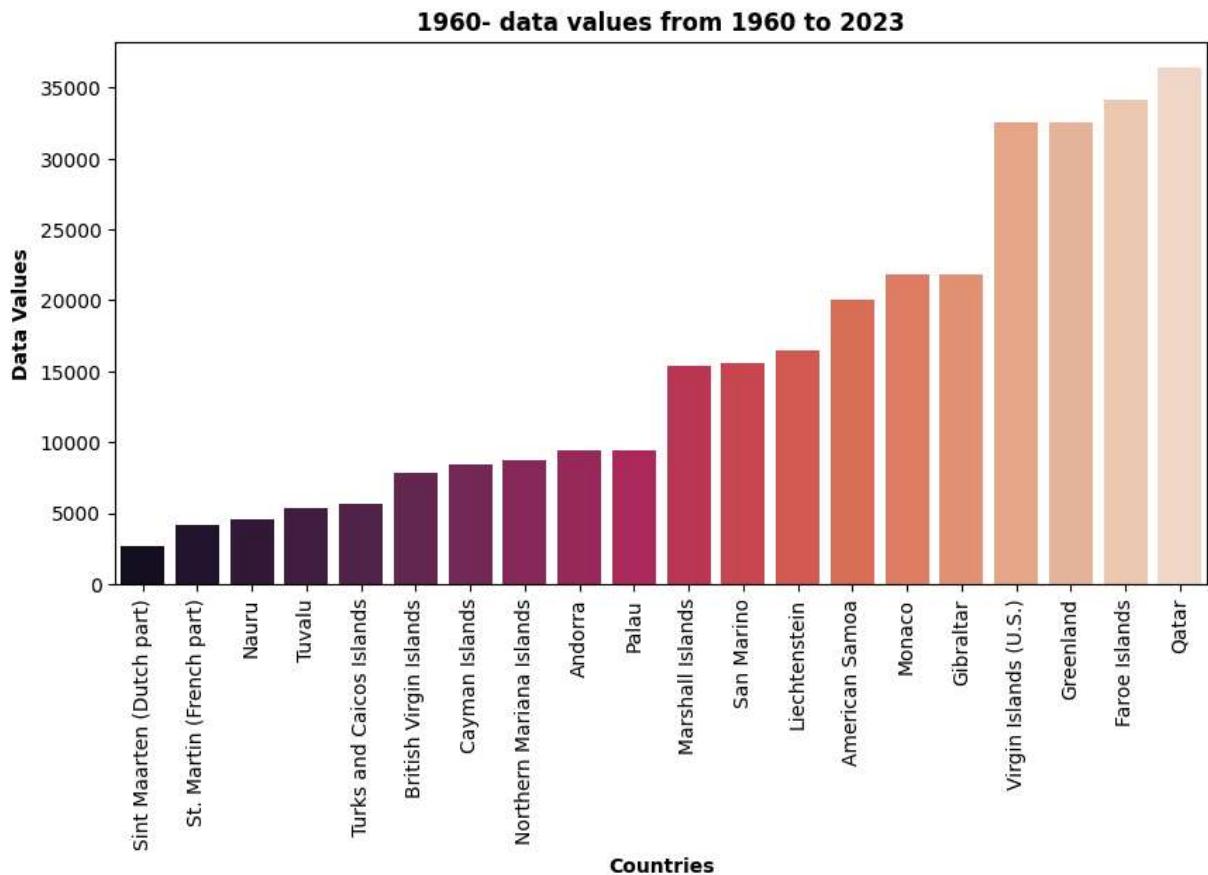
```

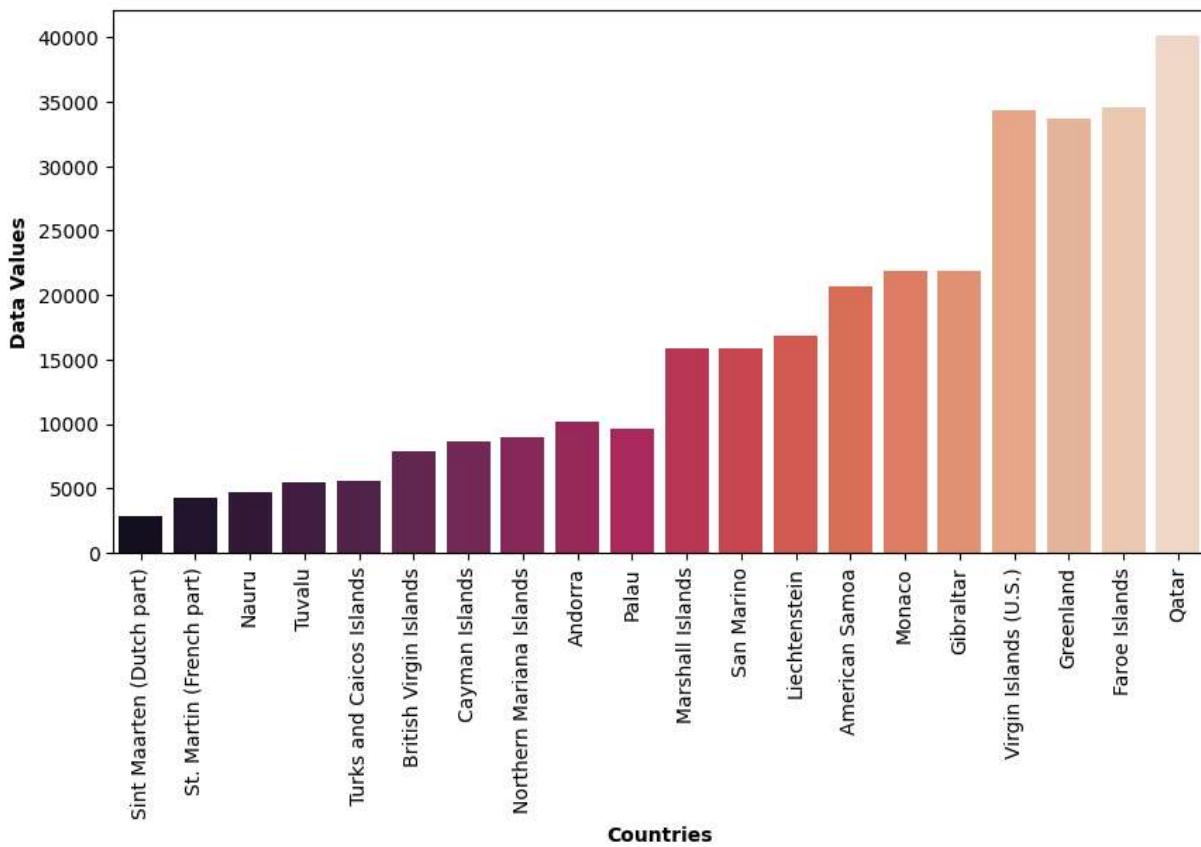
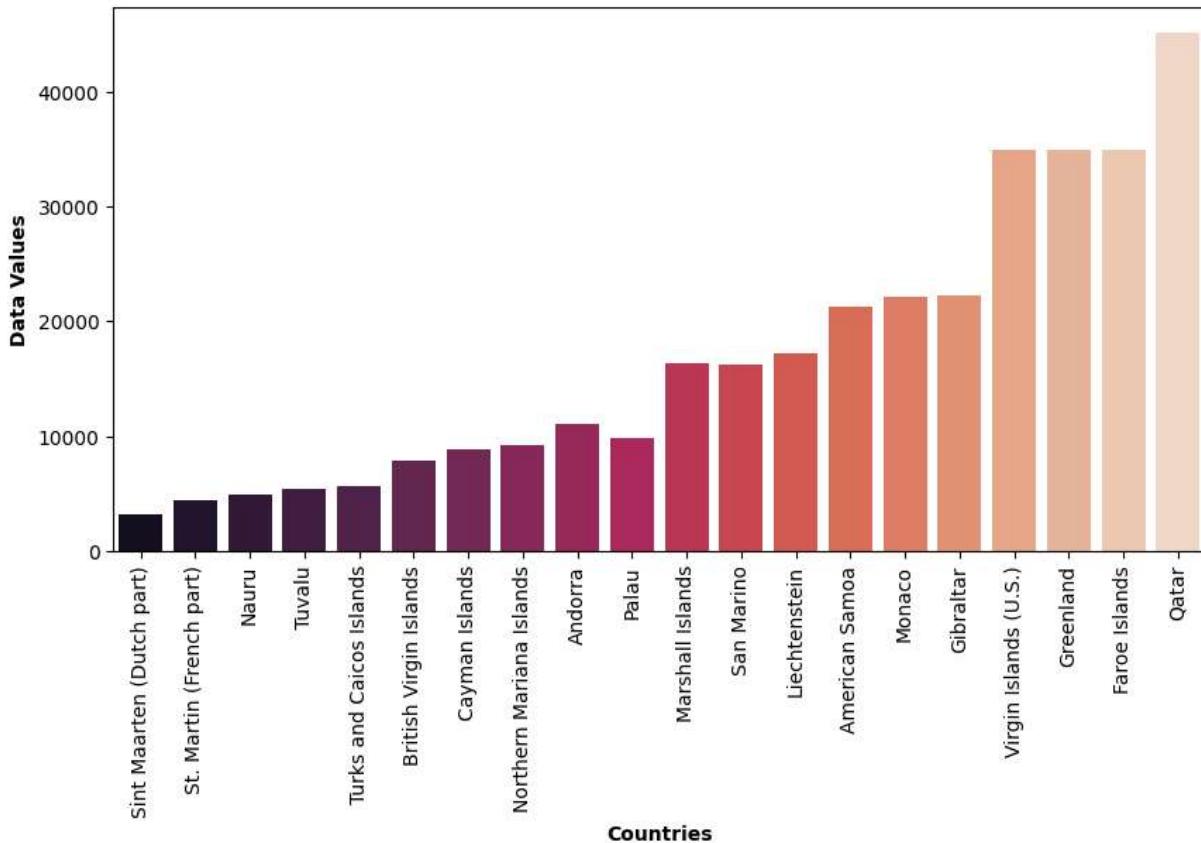
color_palette = sns.color_palette("rocket", n_colors=20)
for country_name, data_values in country_by_1960_t.iterrows():
    fig=plt.figure(figsize=(10,5))

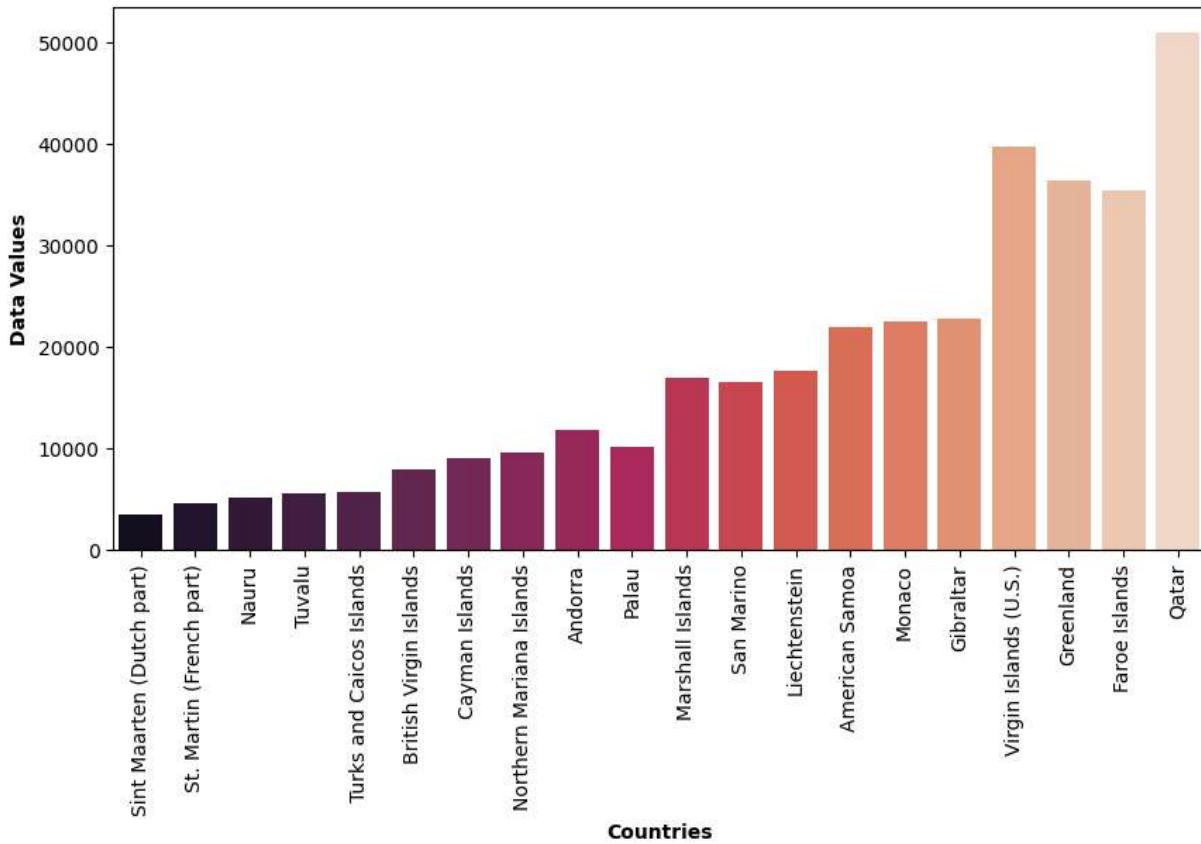
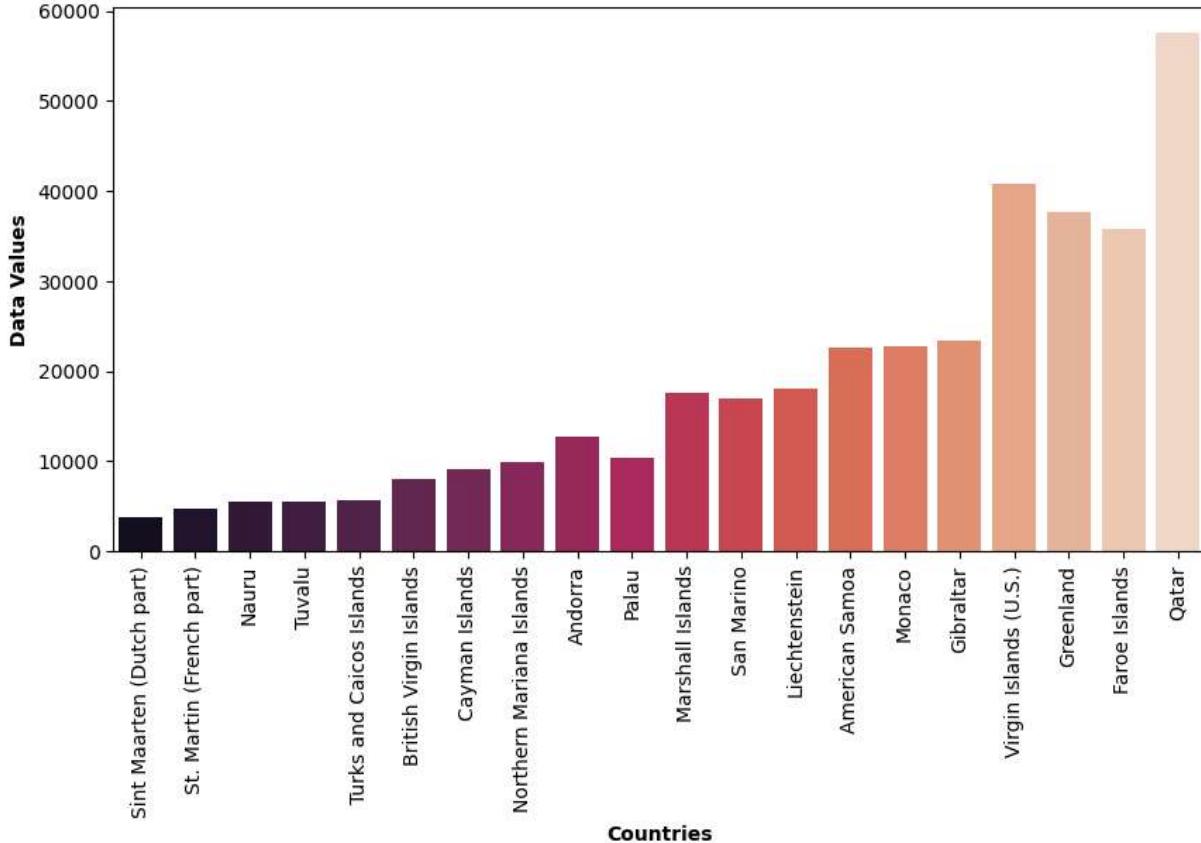
    sns.barplot(x=data_values.index, y=data_values.values, hue=data_values.index, pal
    plt.xlabel("Countries", fontweight="bold")
    plt.ylabel("Data Values", fontweight="bold")

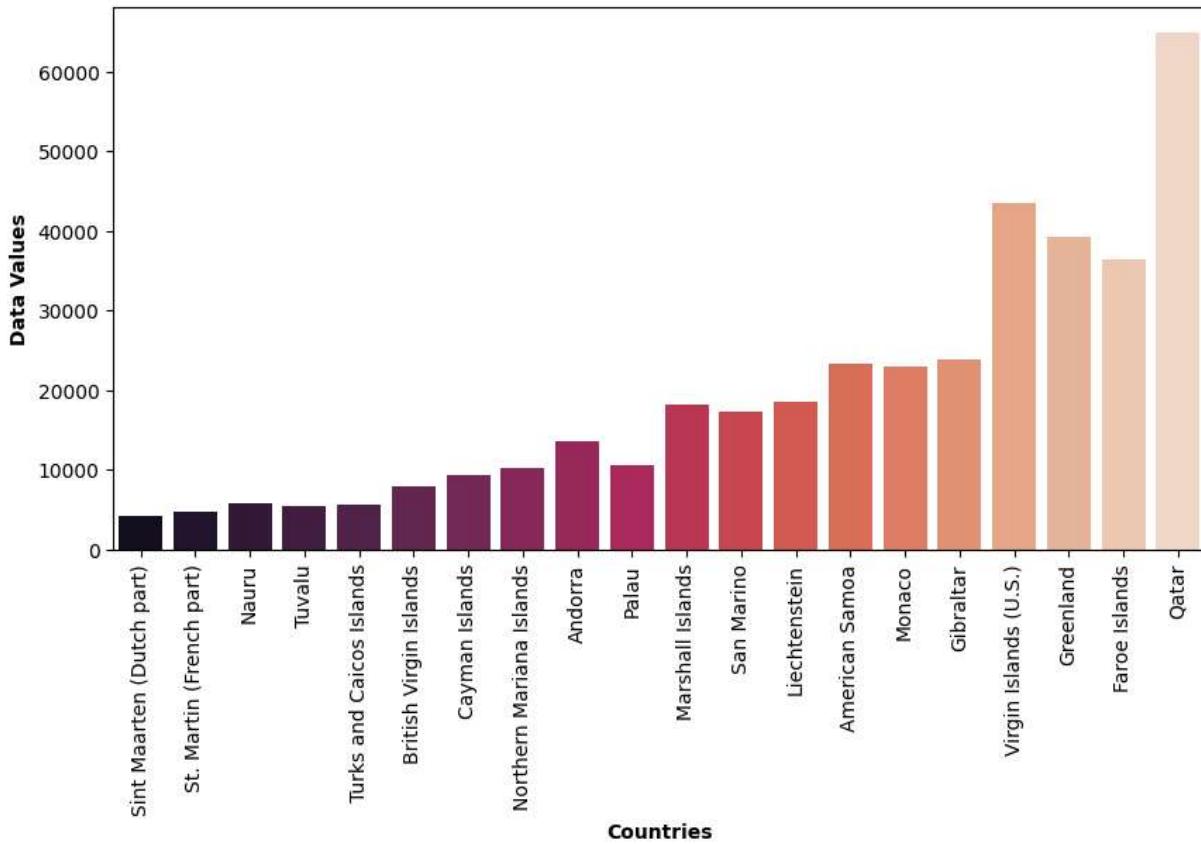
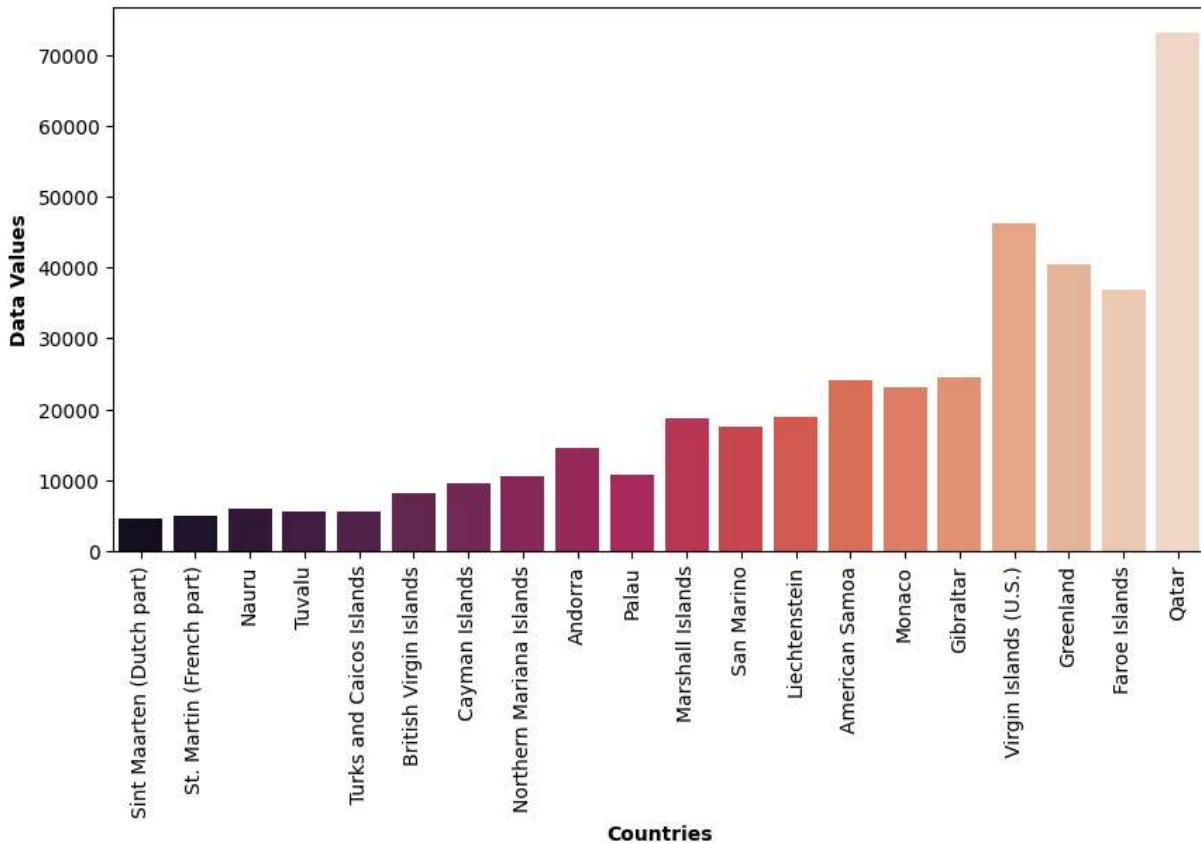
    plt.title(f"{country_name}- data values from 1960 to 2023", fontweight="bold")
    plt.xticks(rotation=90)
    plt.show()

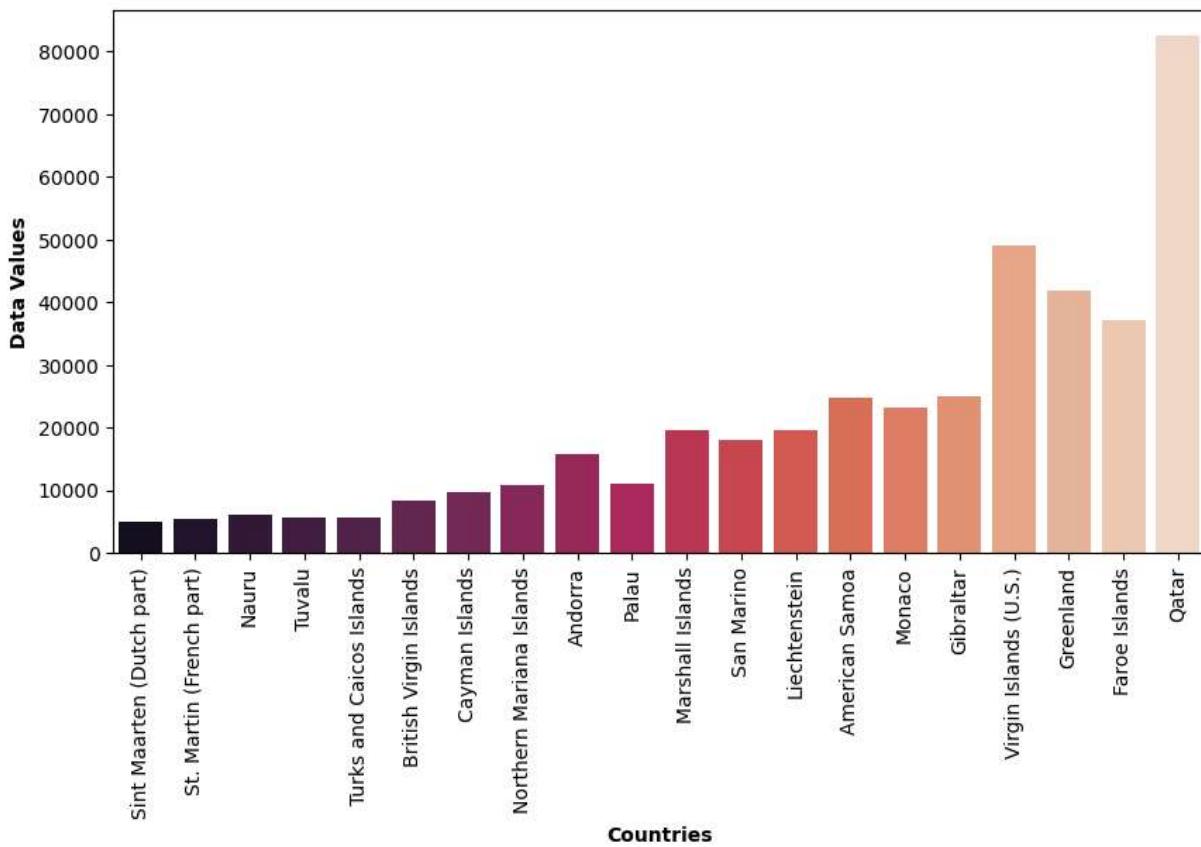
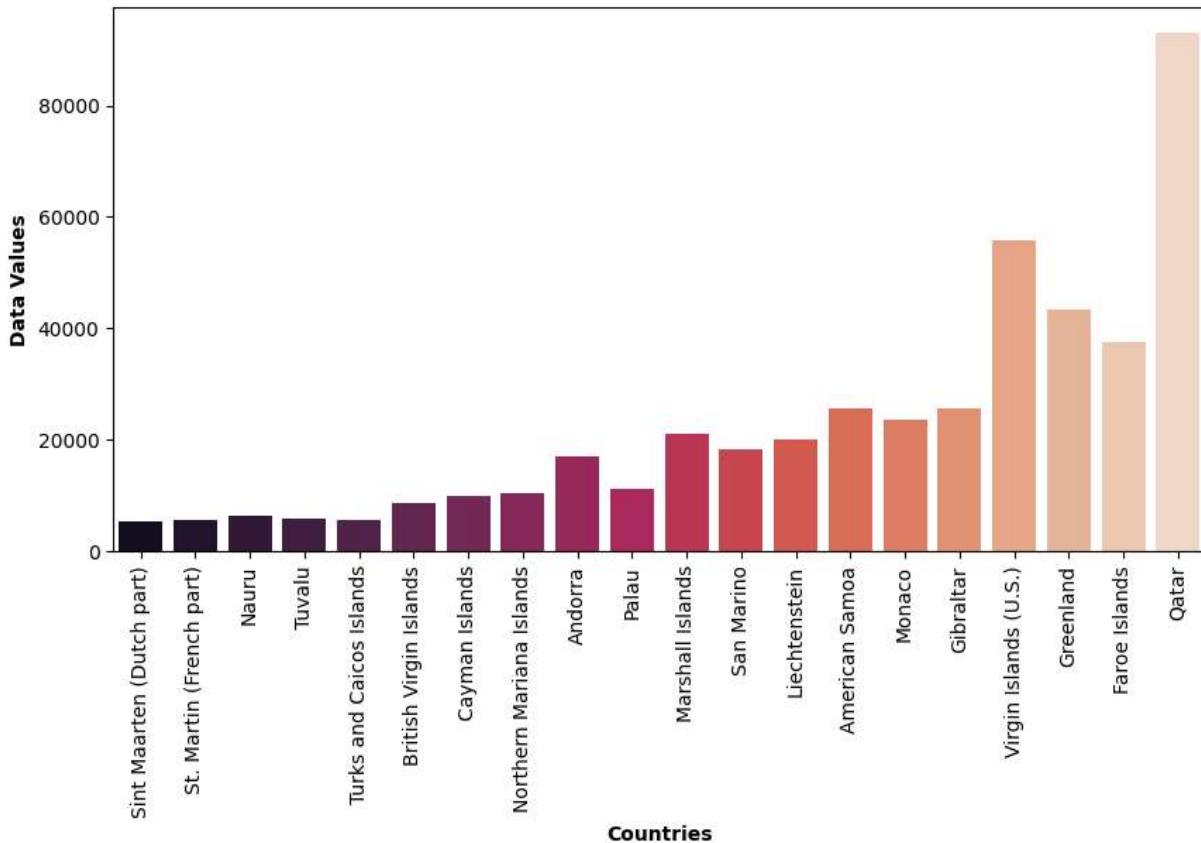
```

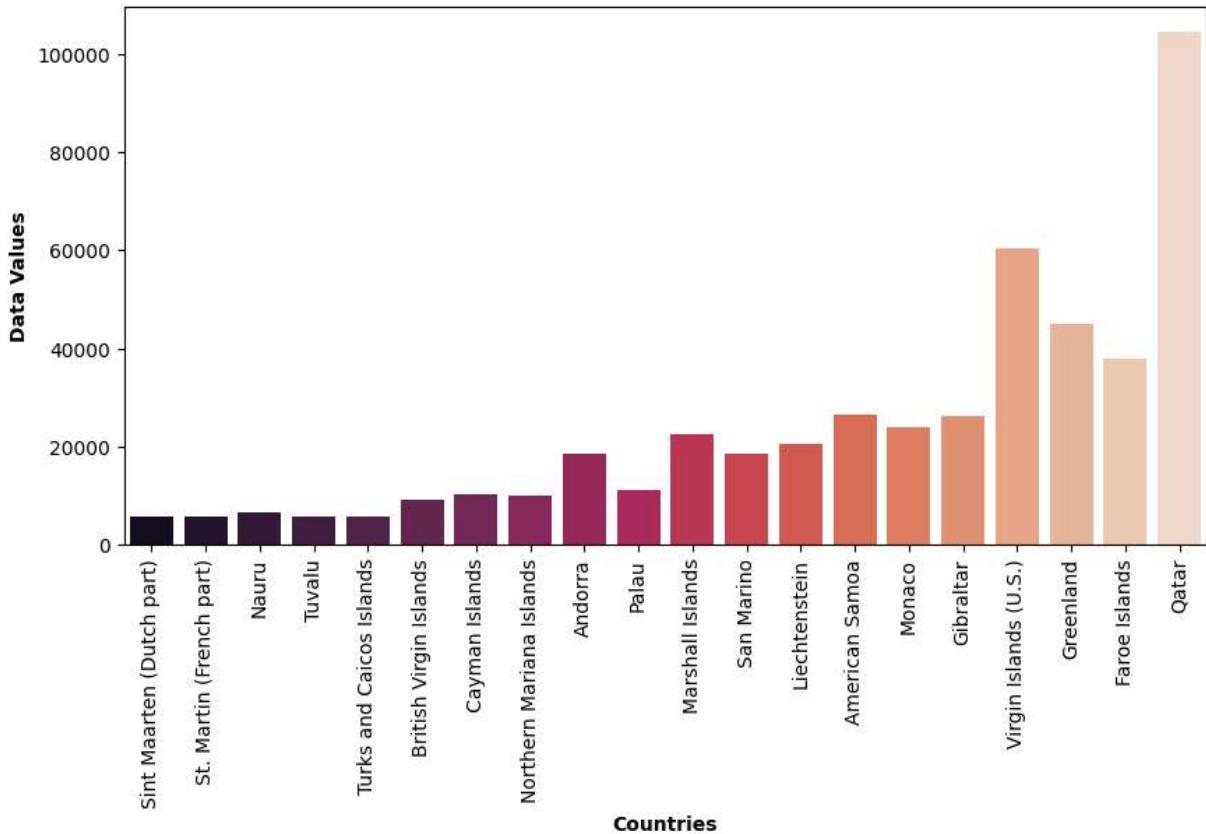
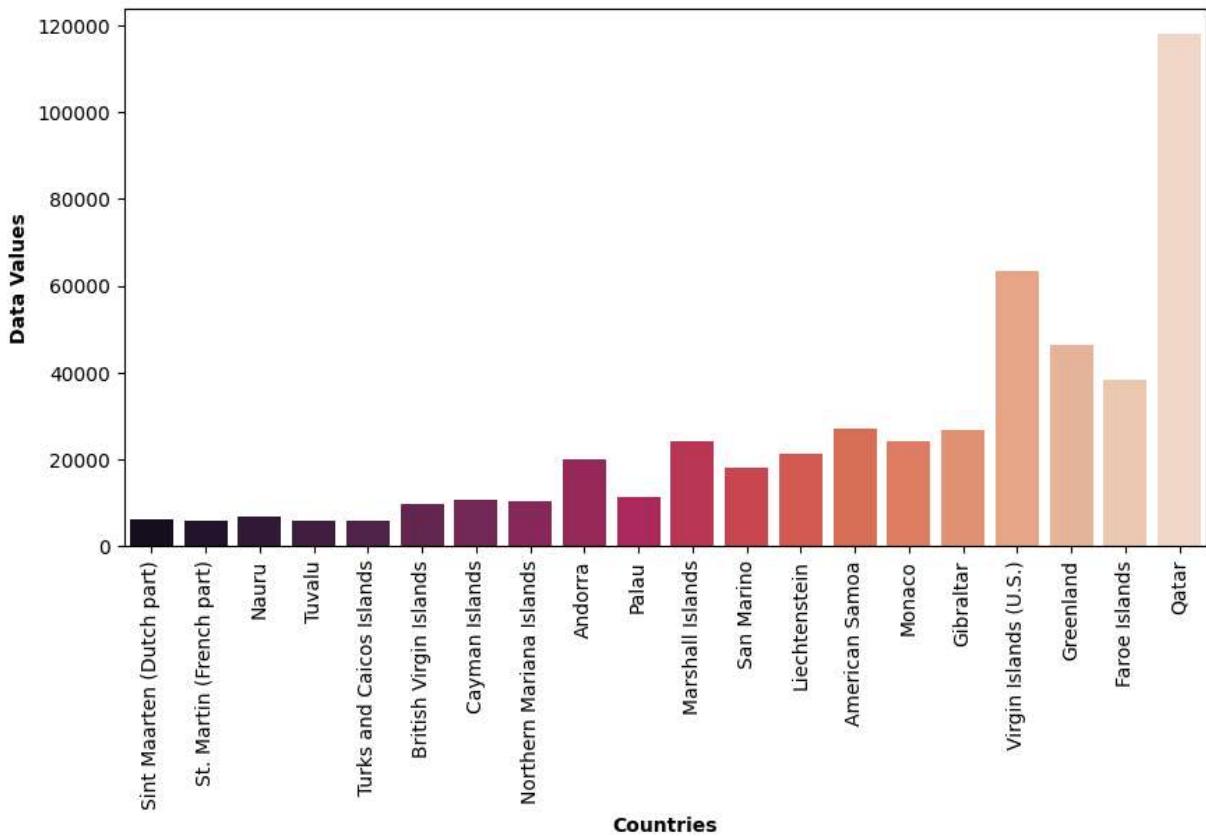


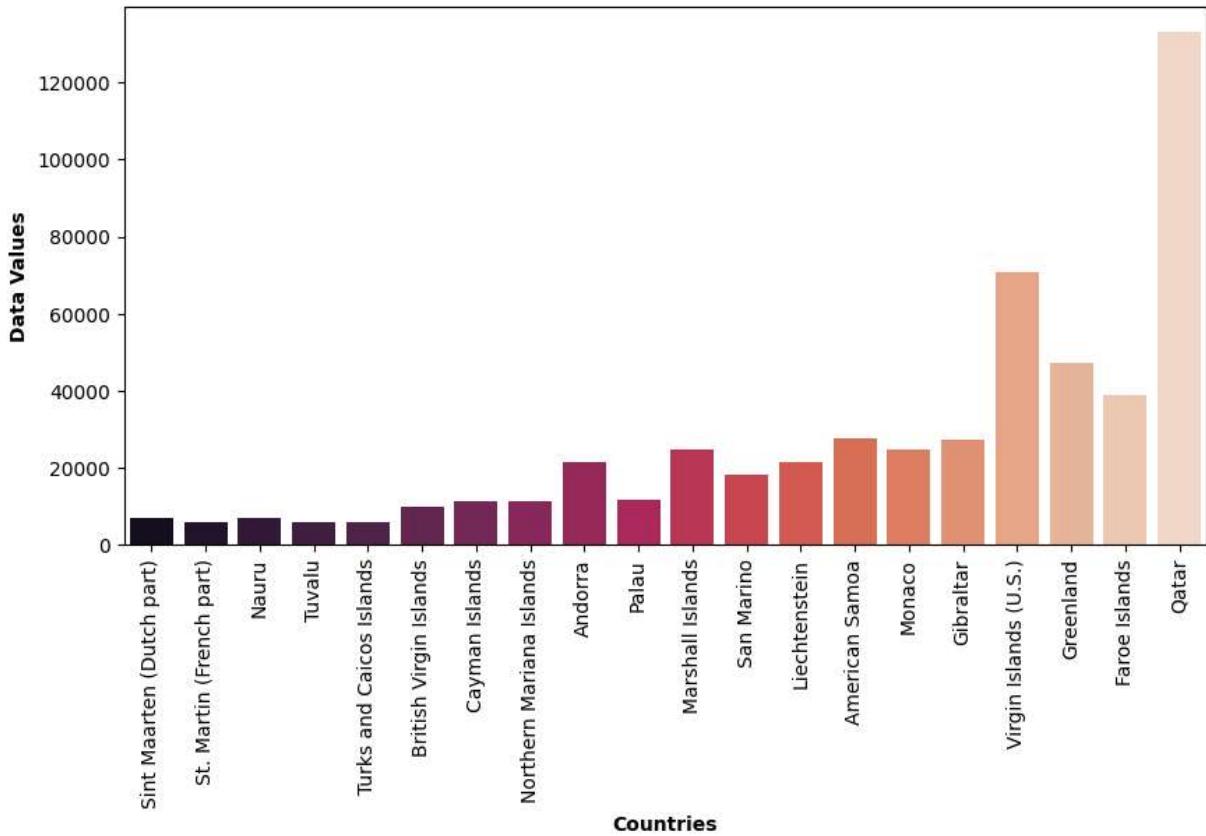
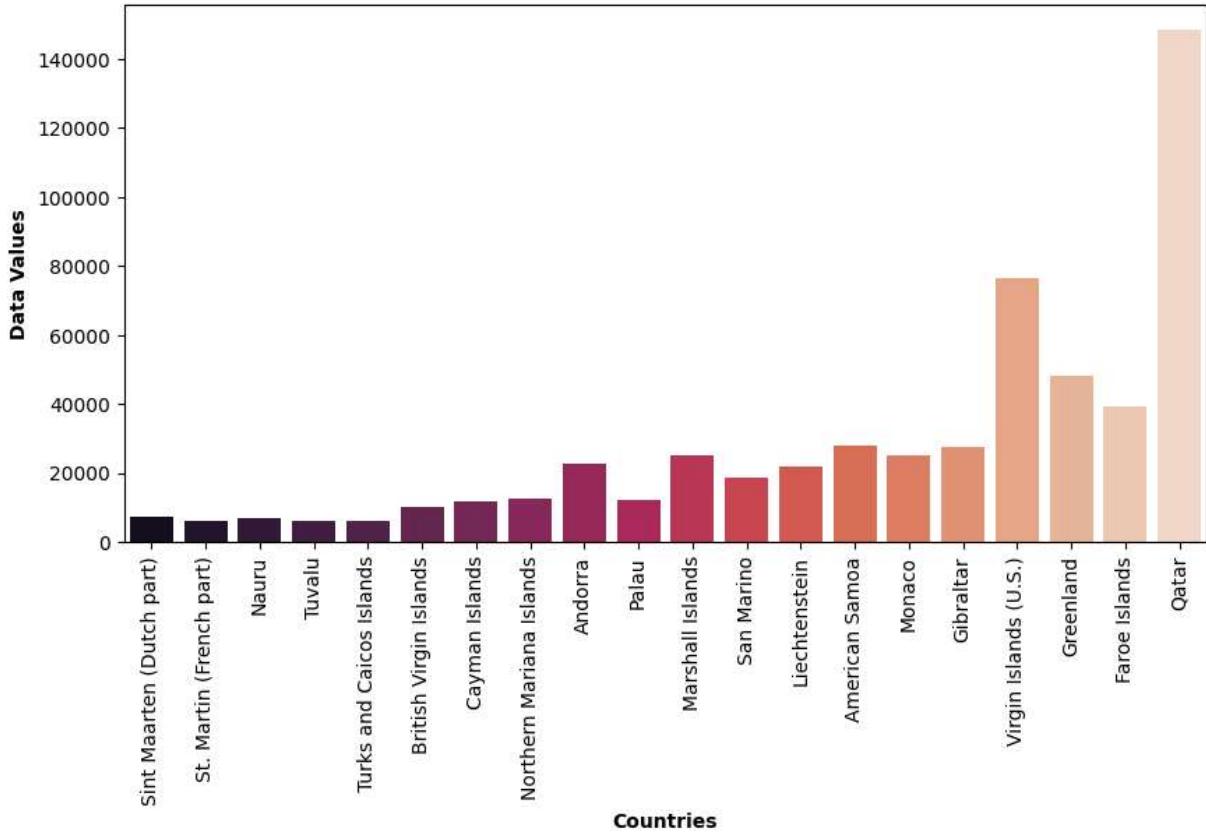
1961- data values from 1960 to 2023**1962- data values from 1960 to 2023**

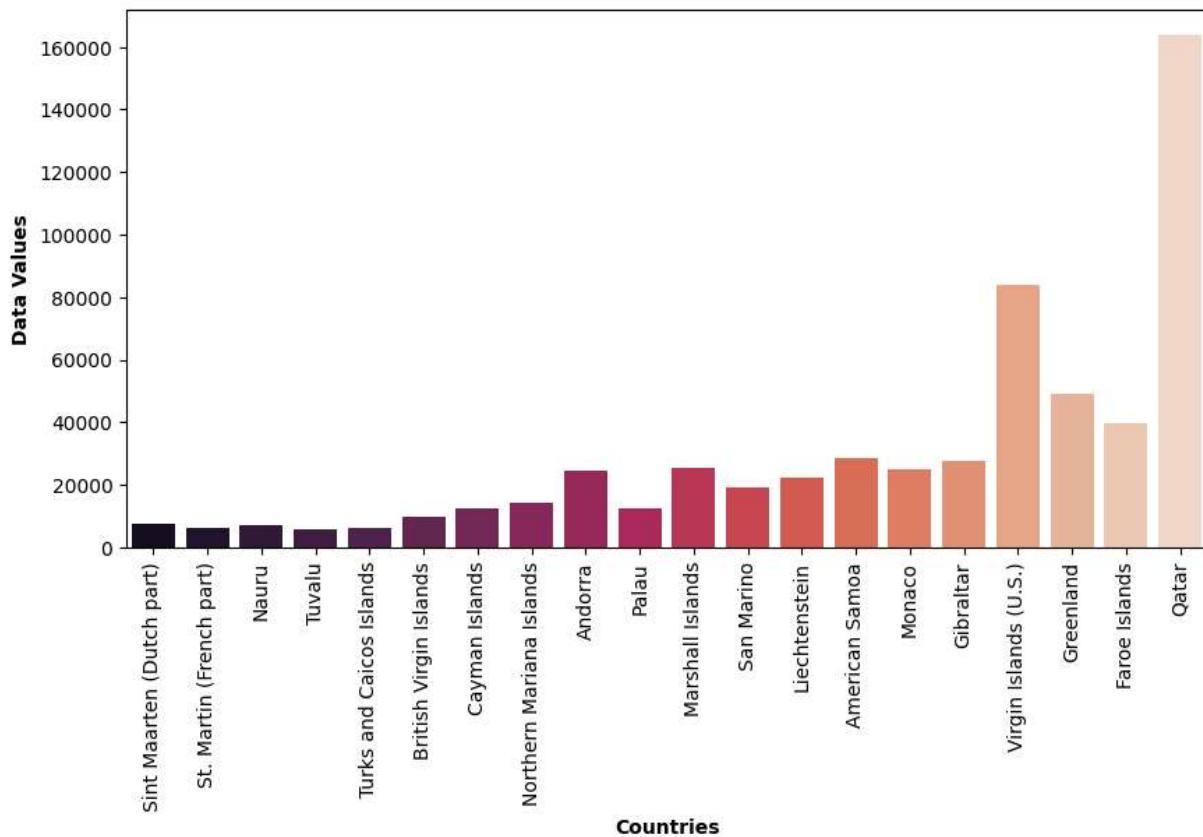
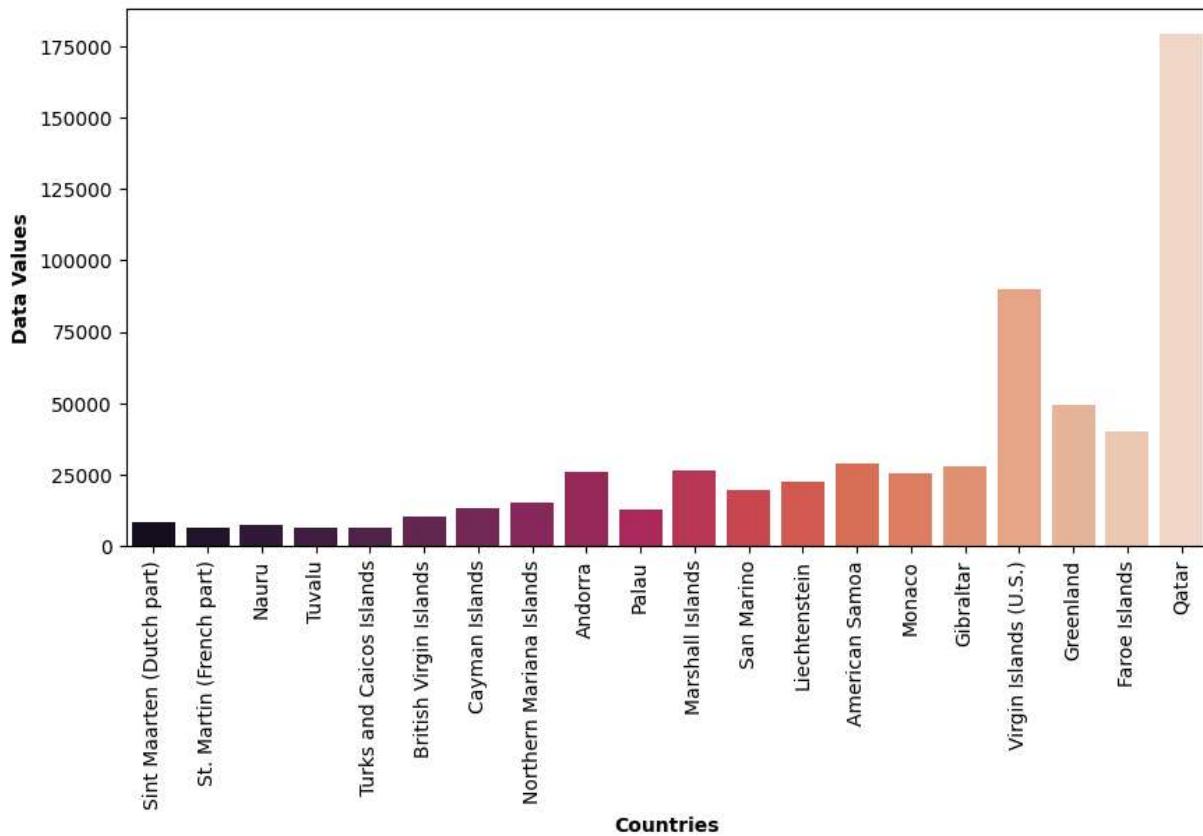
1963- data values from 1960 to 2023**1964- data values from 1960 to 2023**

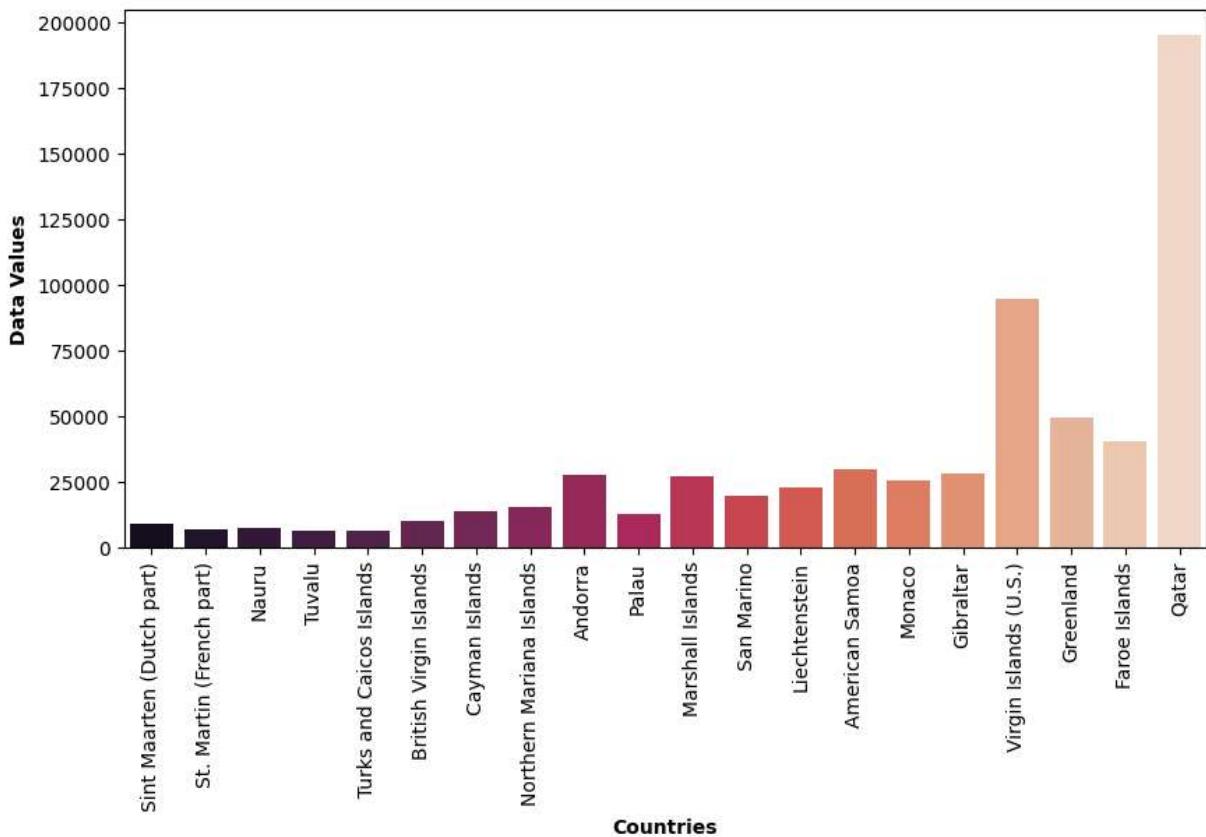
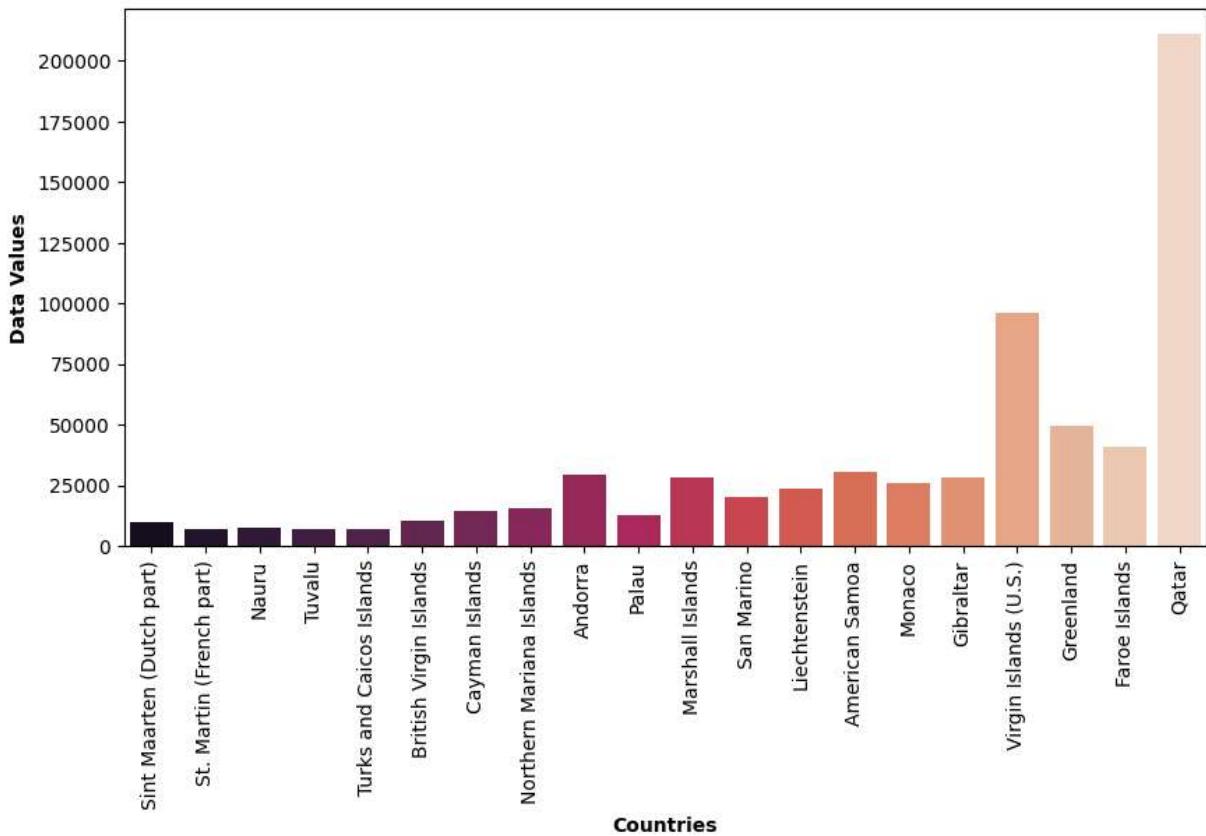
1965- data values from 1960 to 2023**1966- data values from 1960 to 2023**

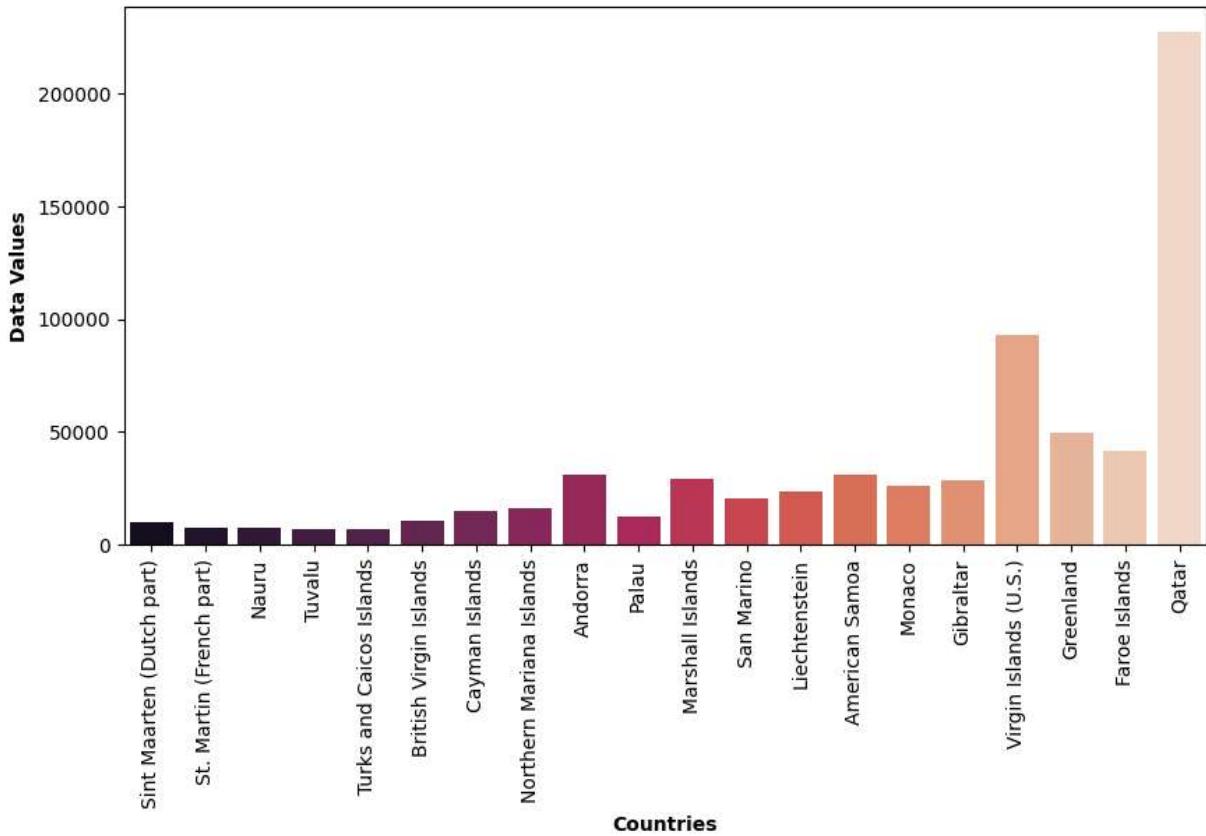
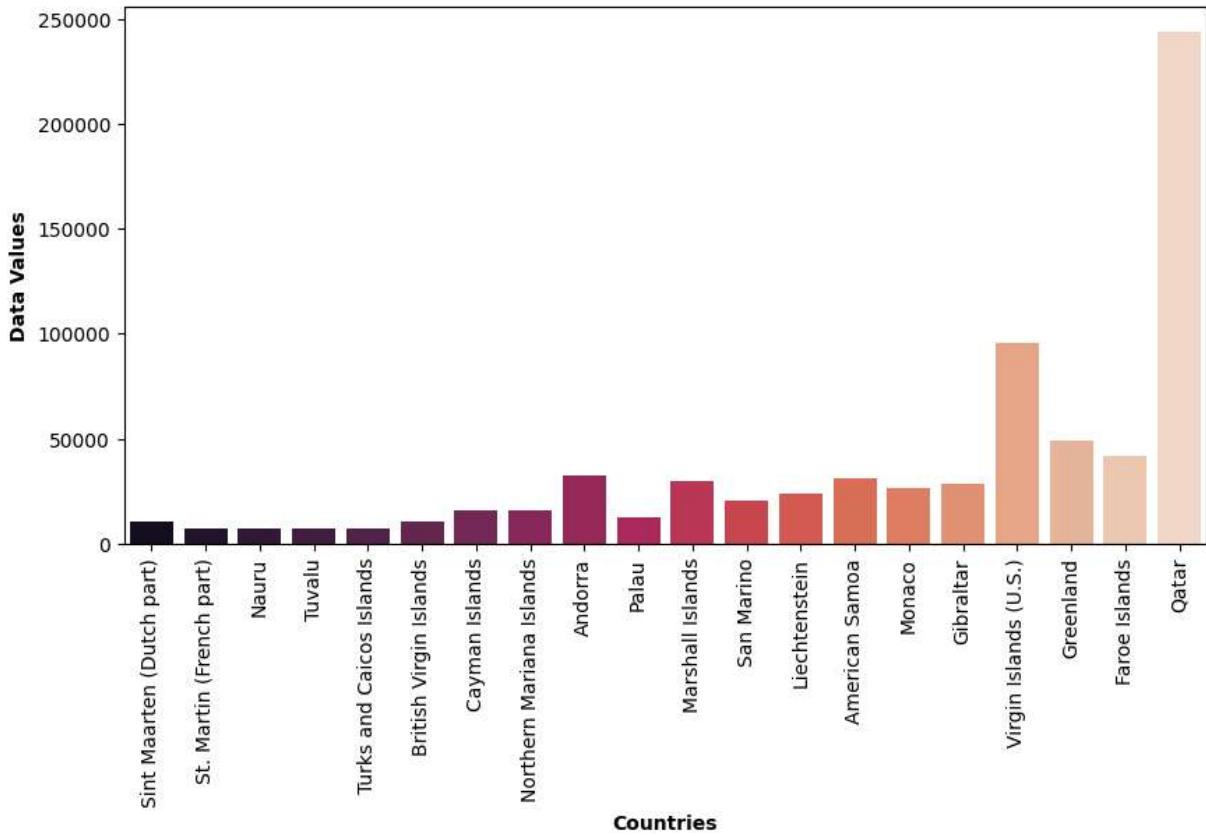
1967- data values from 1960 to 2023**1968- data values from 1960 to 2023**

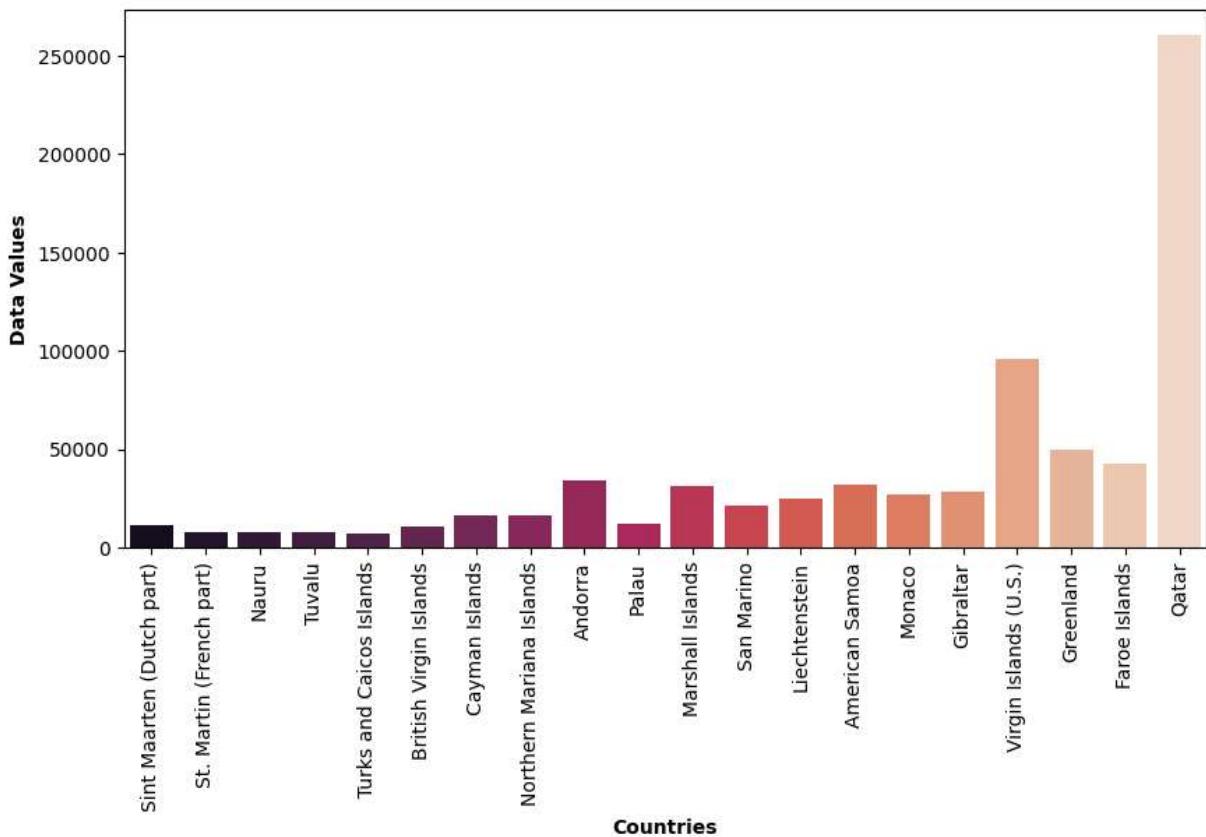
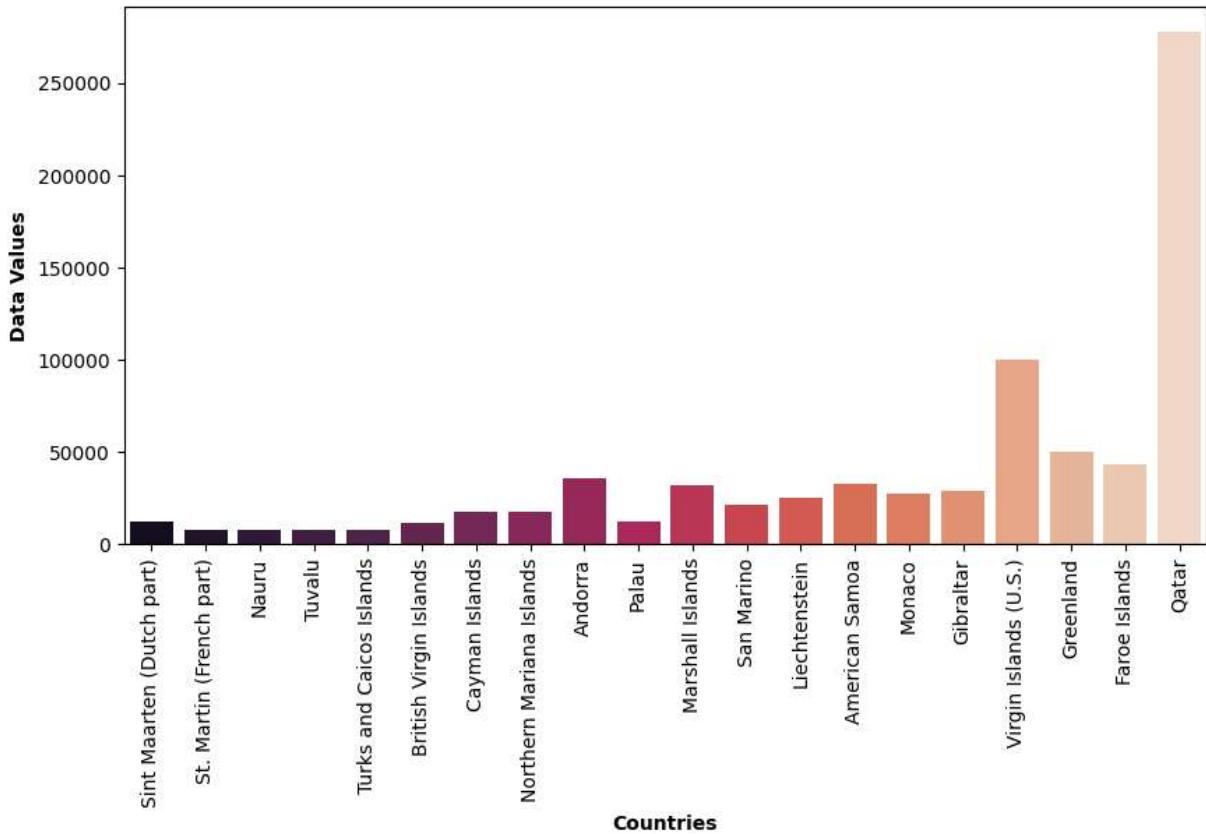
1969- data values from 1960 to 2023**1970- data values from 1960 to 2023**

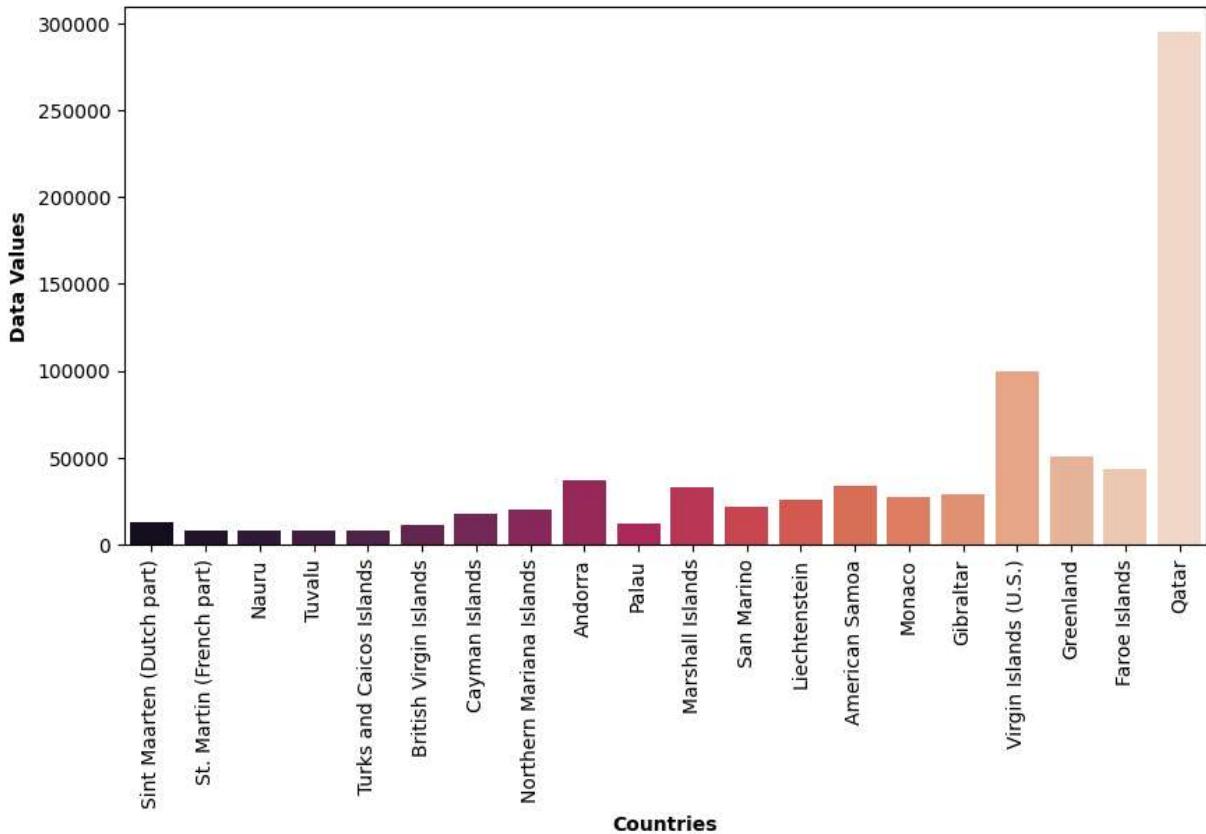
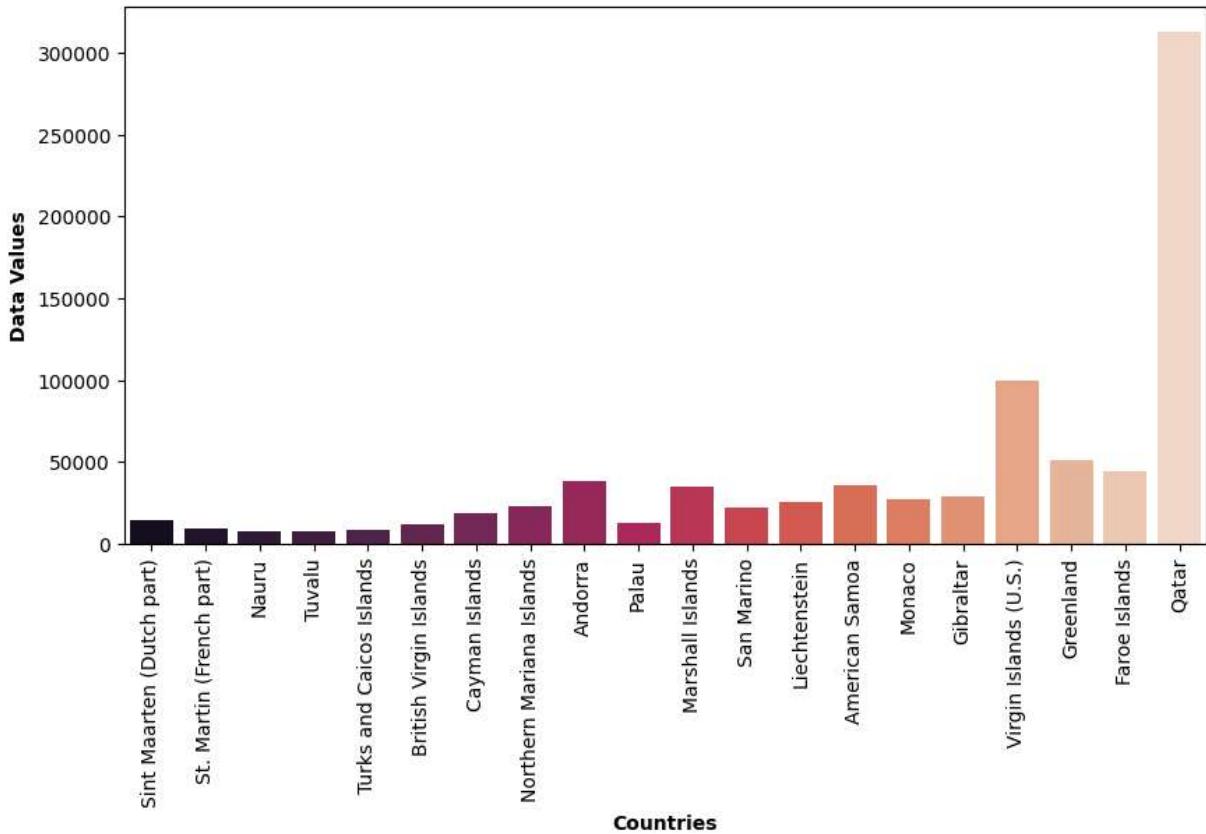
1971- data values from 1960 to 2023**1972- data values from 1960 to 2023**

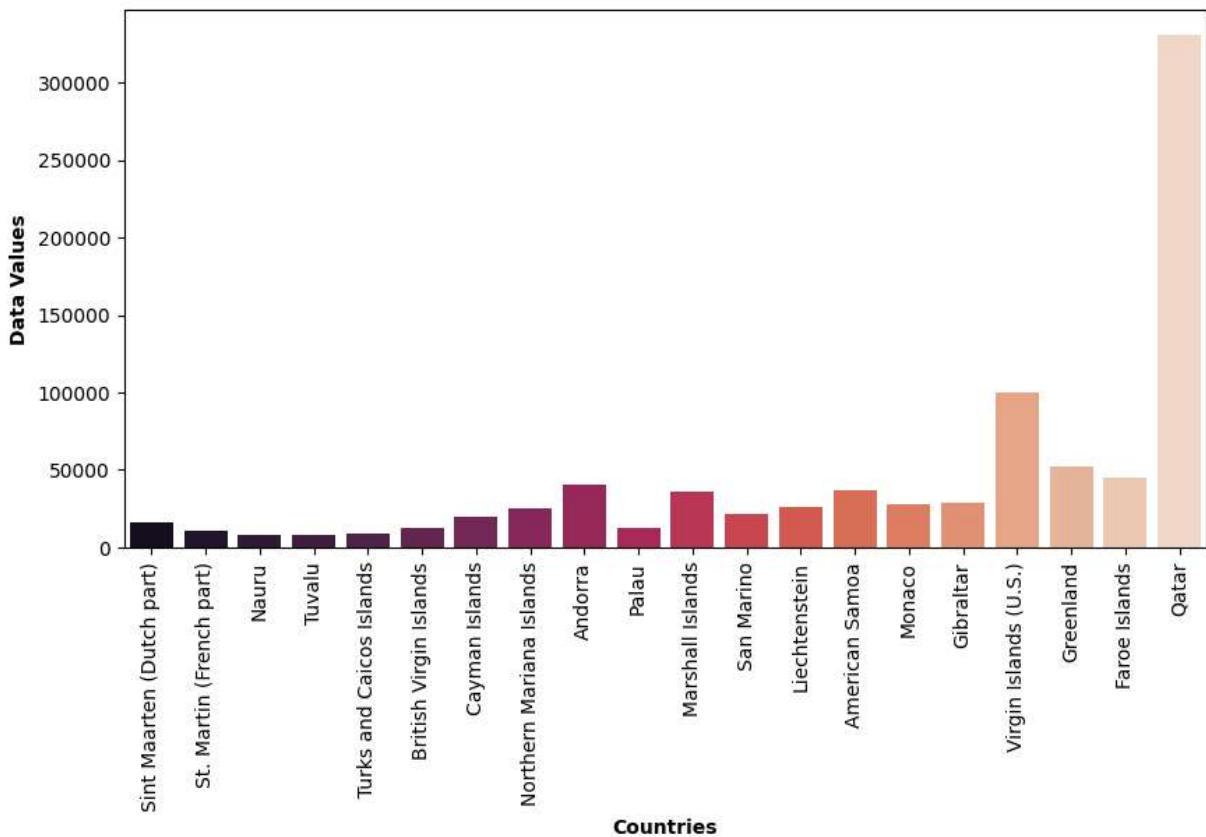
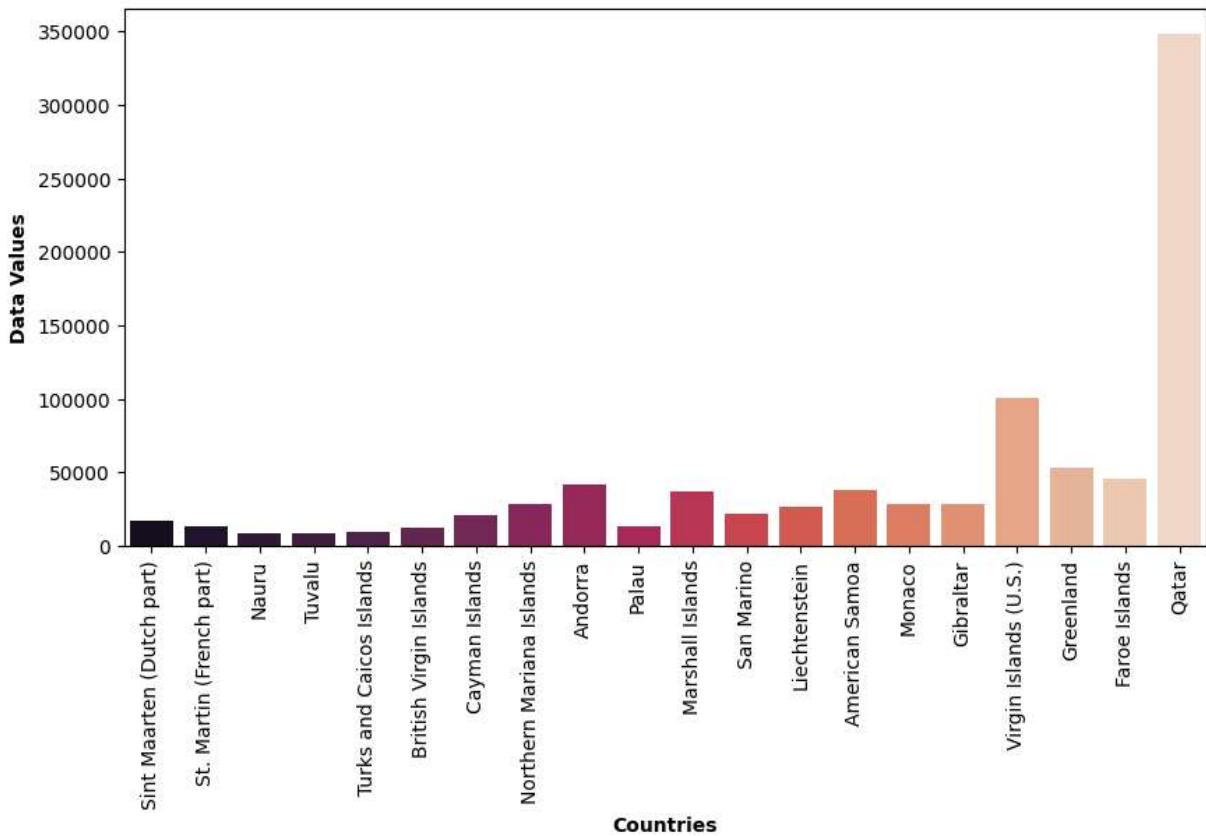
1973- data values from 1960 to 2023**1974- data values from 1960 to 2023**

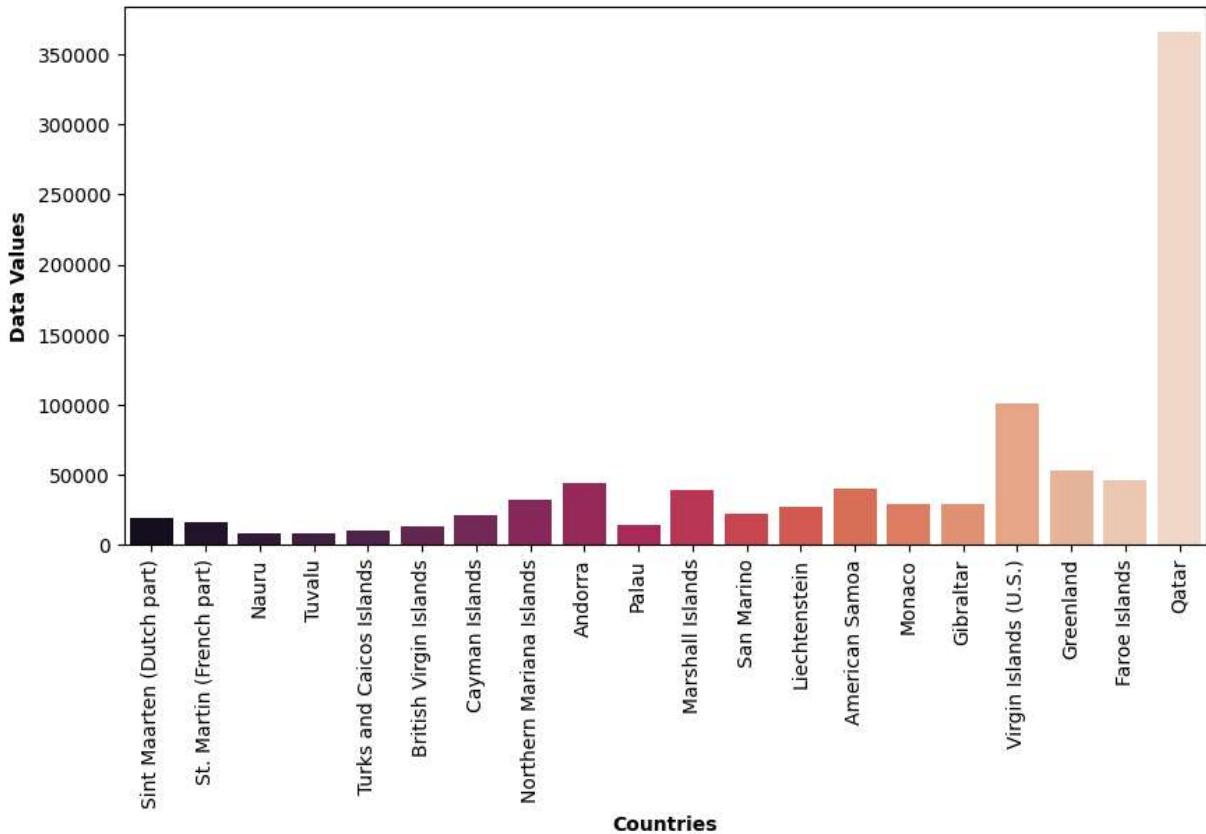
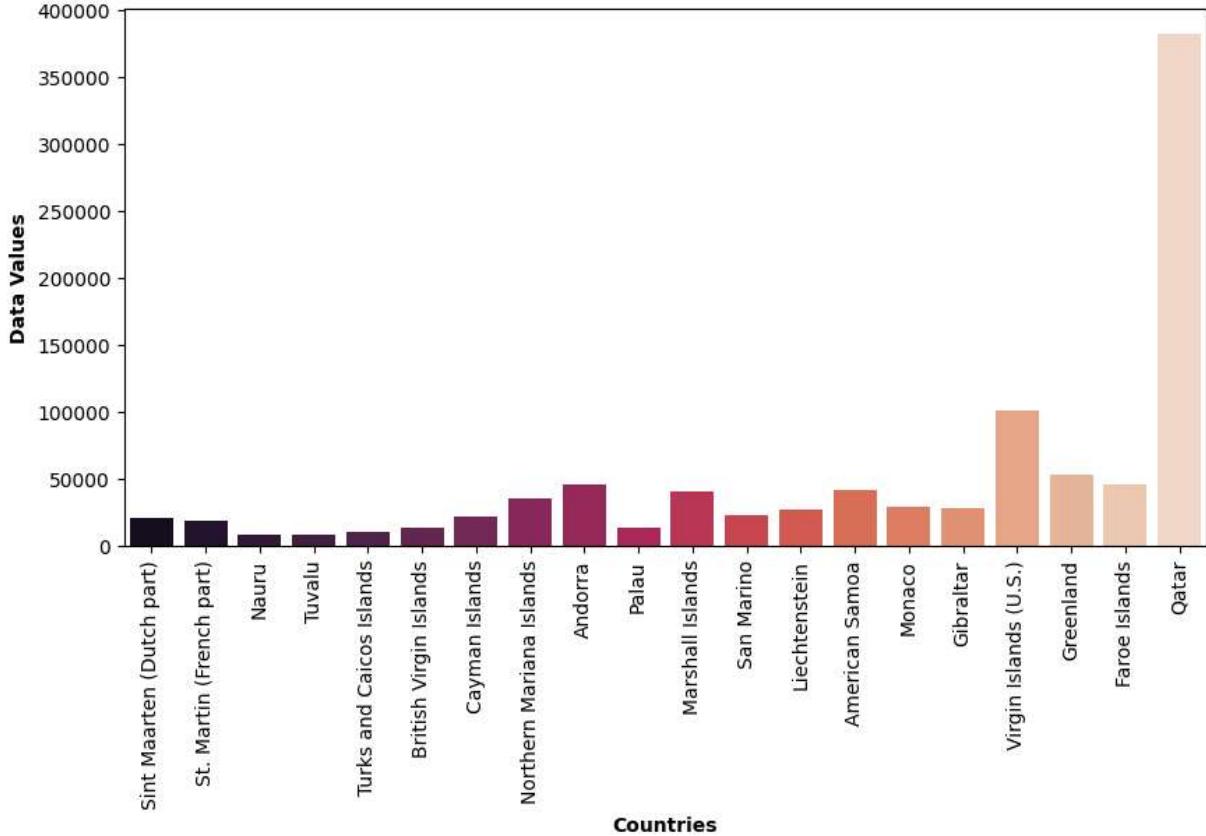
1975- data values from 1960 to 2023**1976- data values from 1960 to 2023**

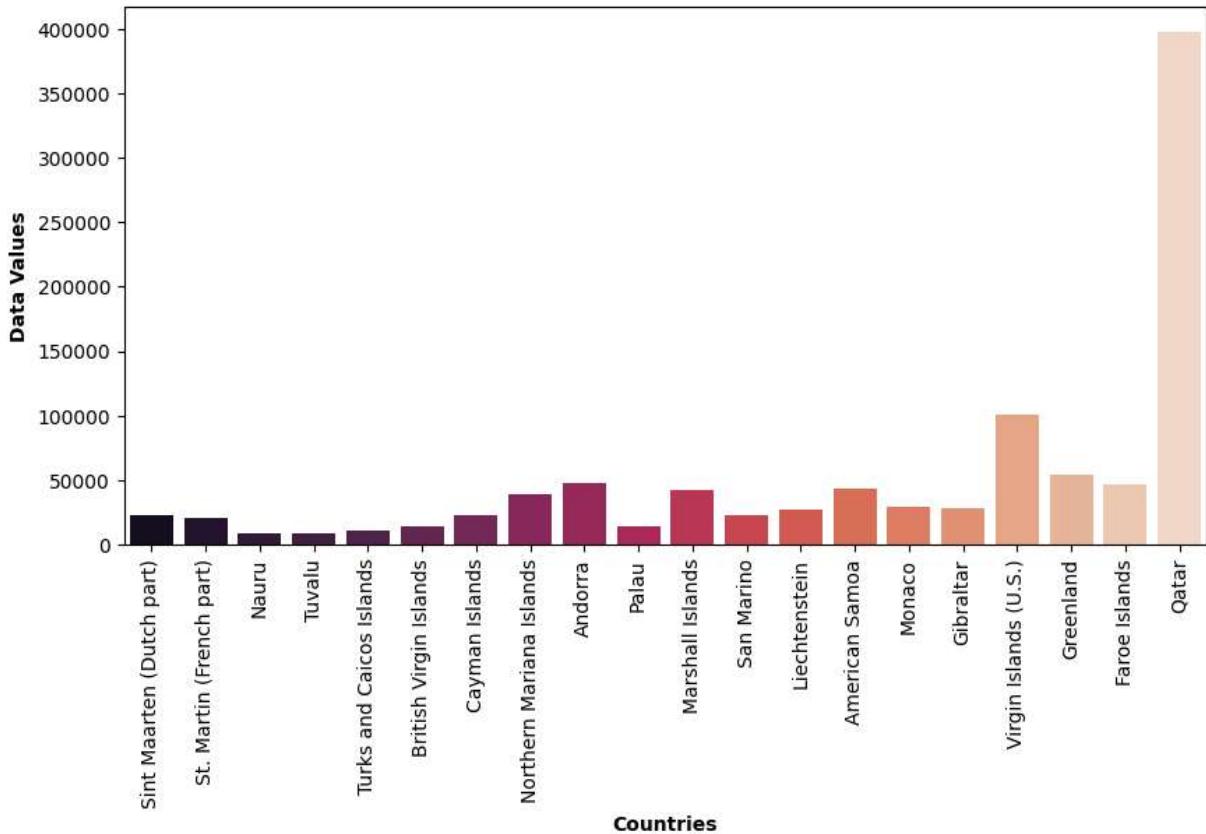
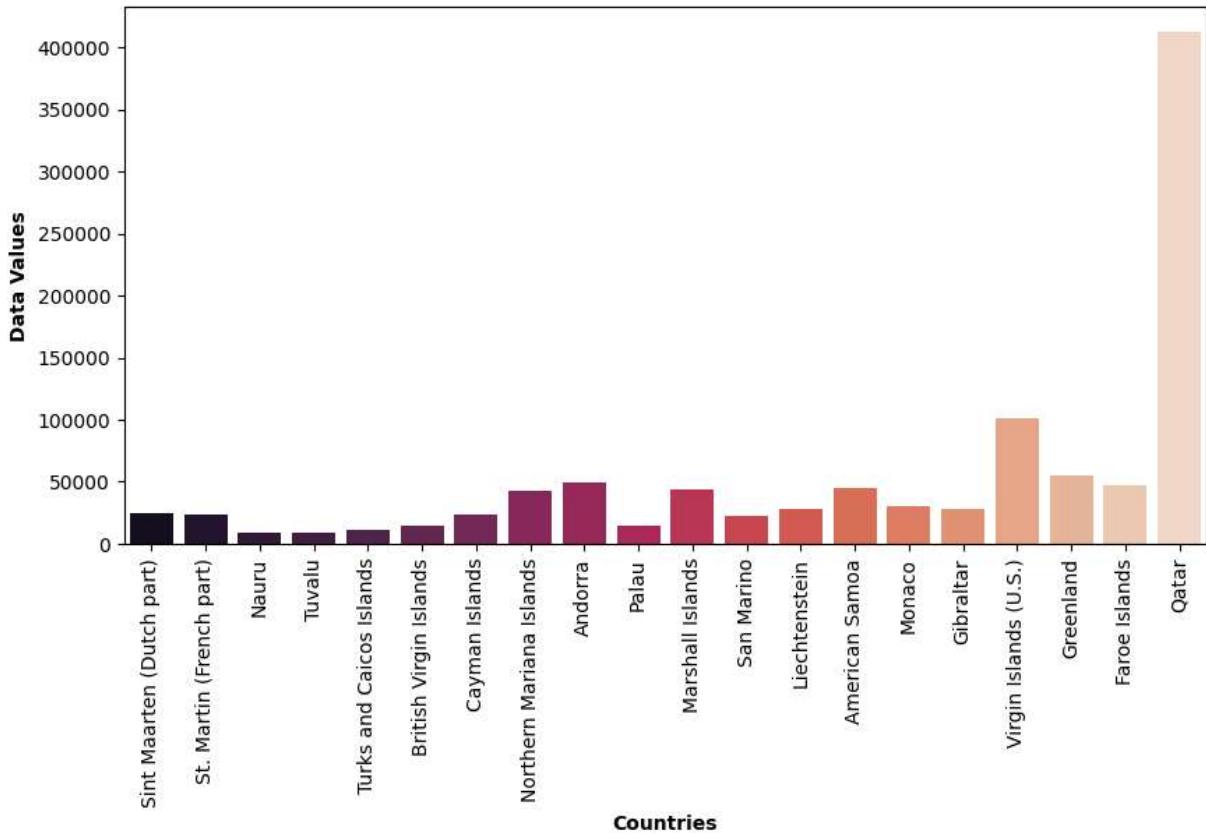
1977- data values from 1960 to 2023**1978- data values from 1960 to 2023**

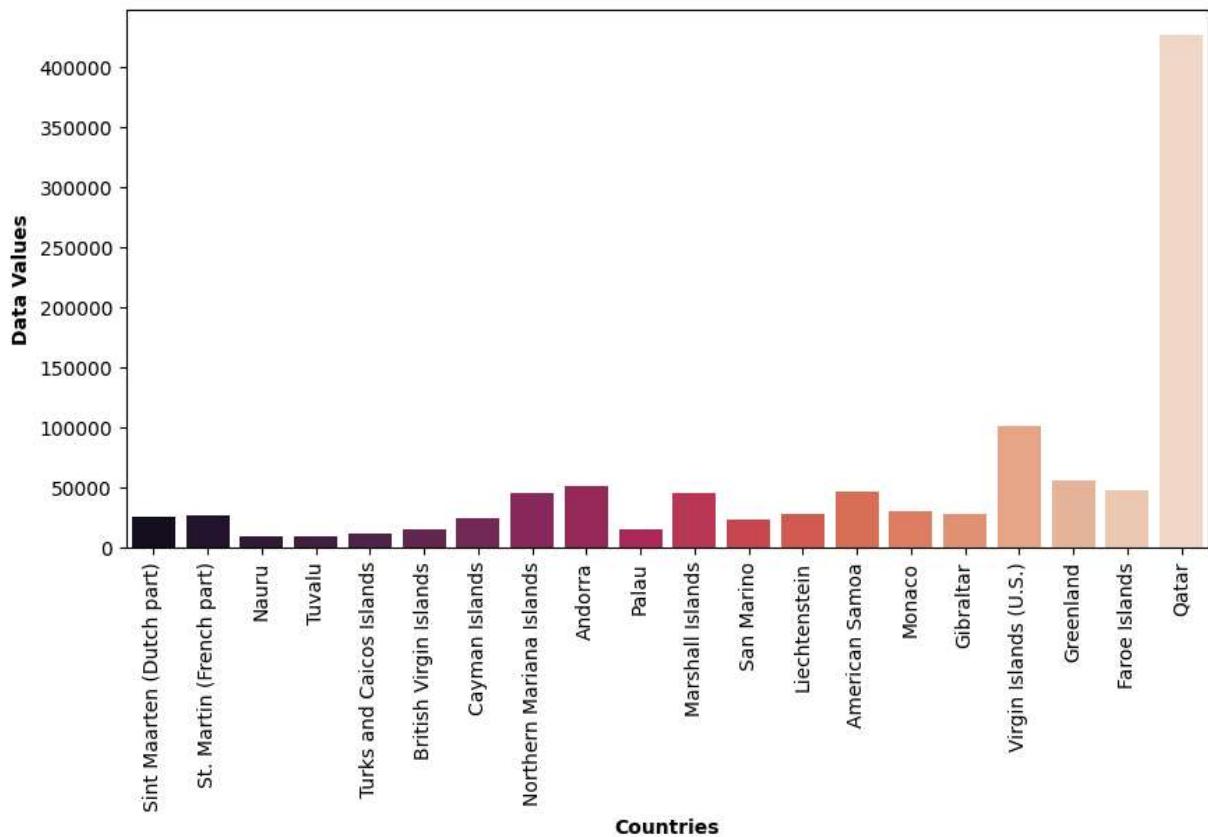
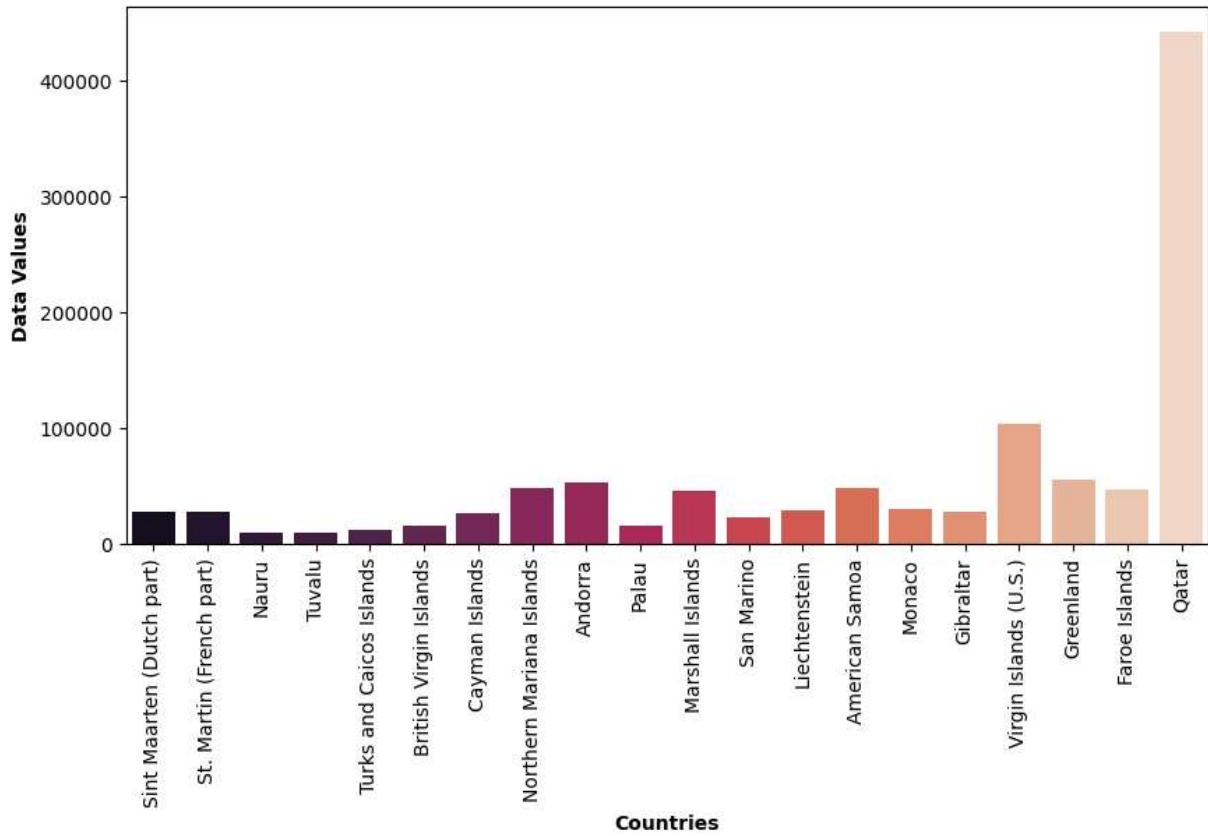
1979- data values from 1960 to 2023**1980- data values from 1960 to 2023**

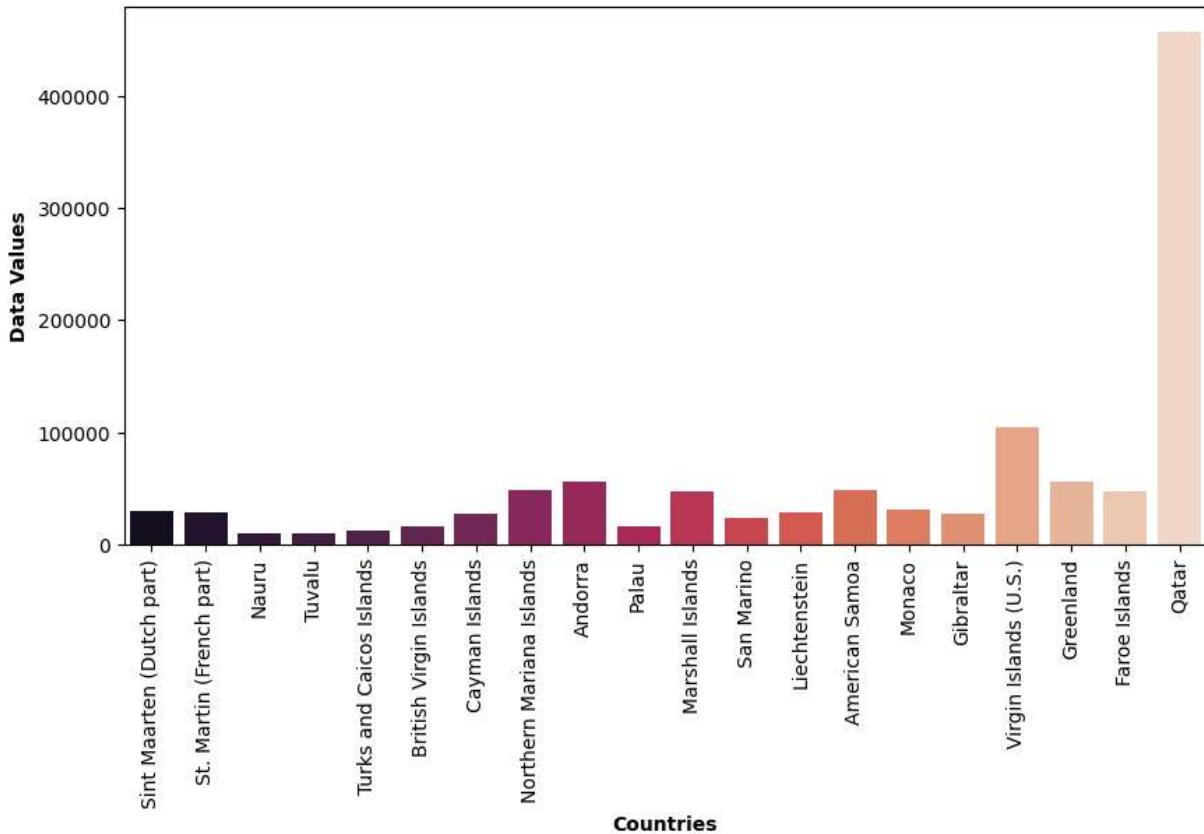
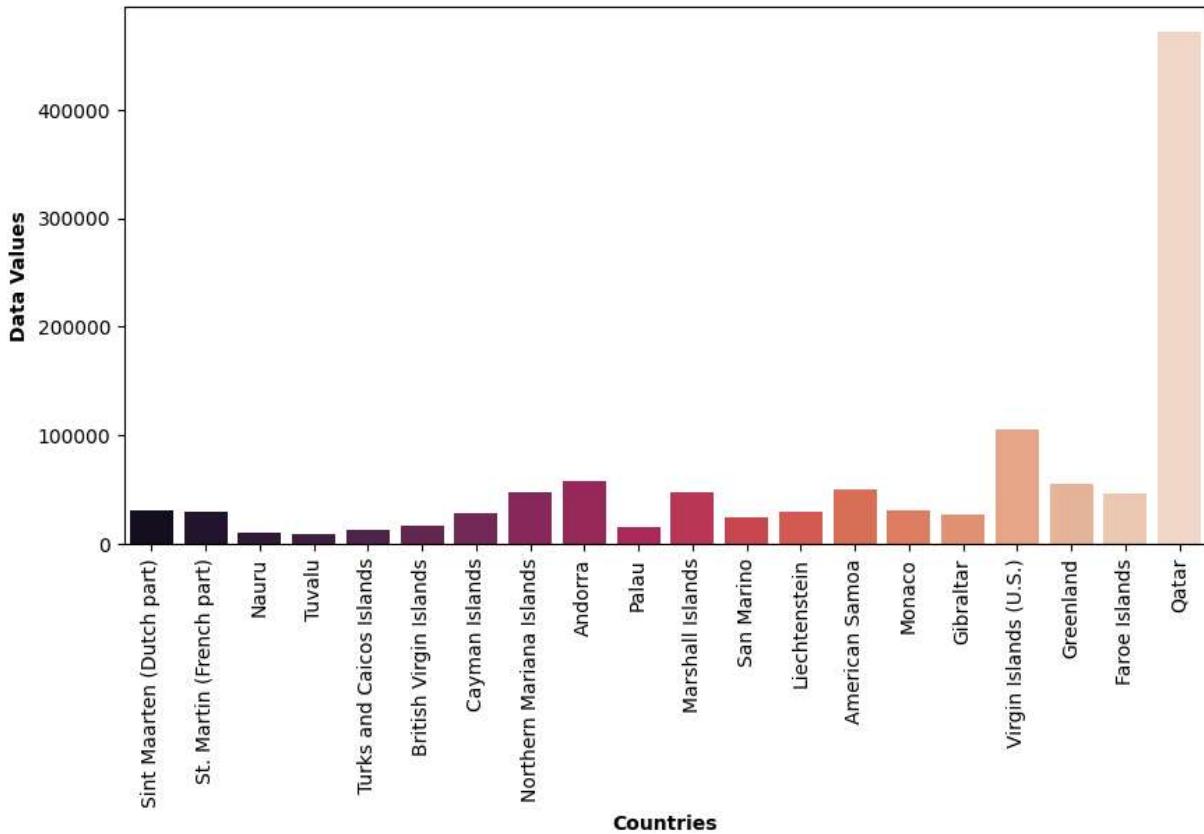
1981- data values from 1960 to 2023**1982- data values from 1960 to 2023**

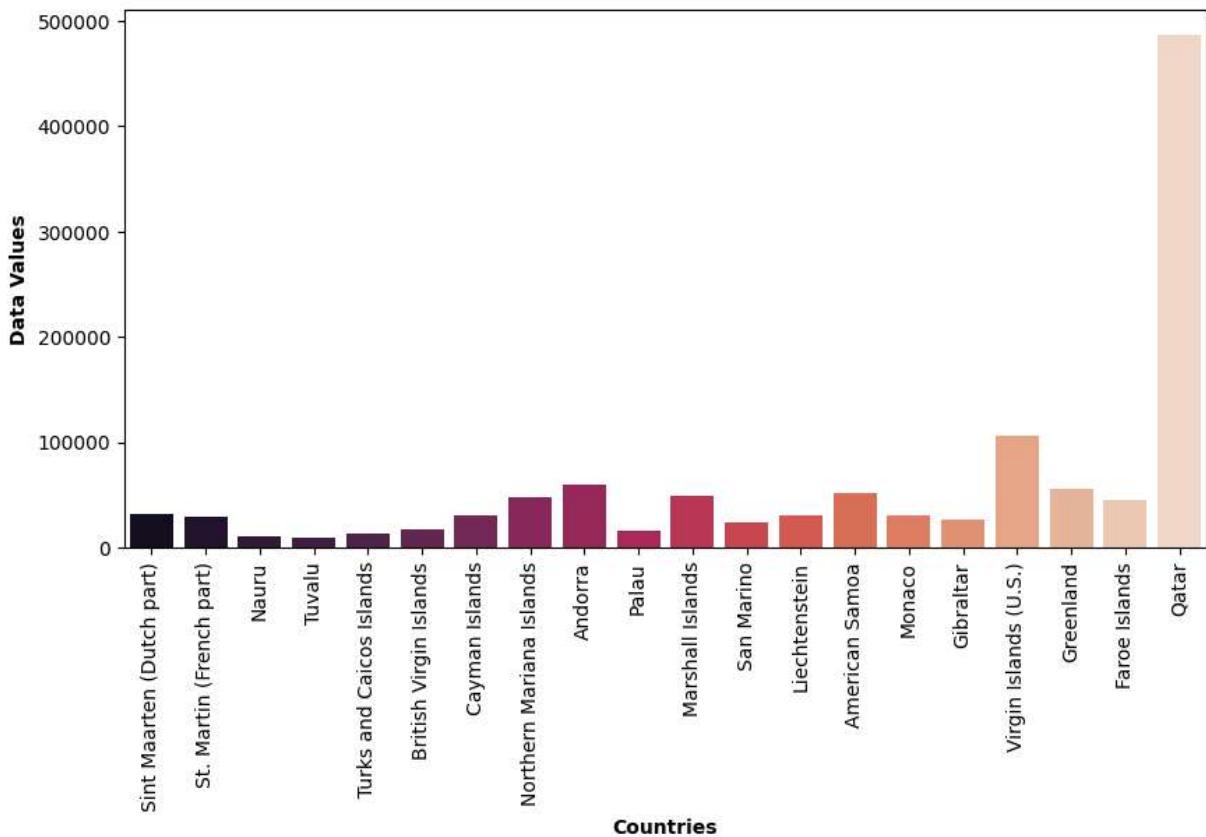
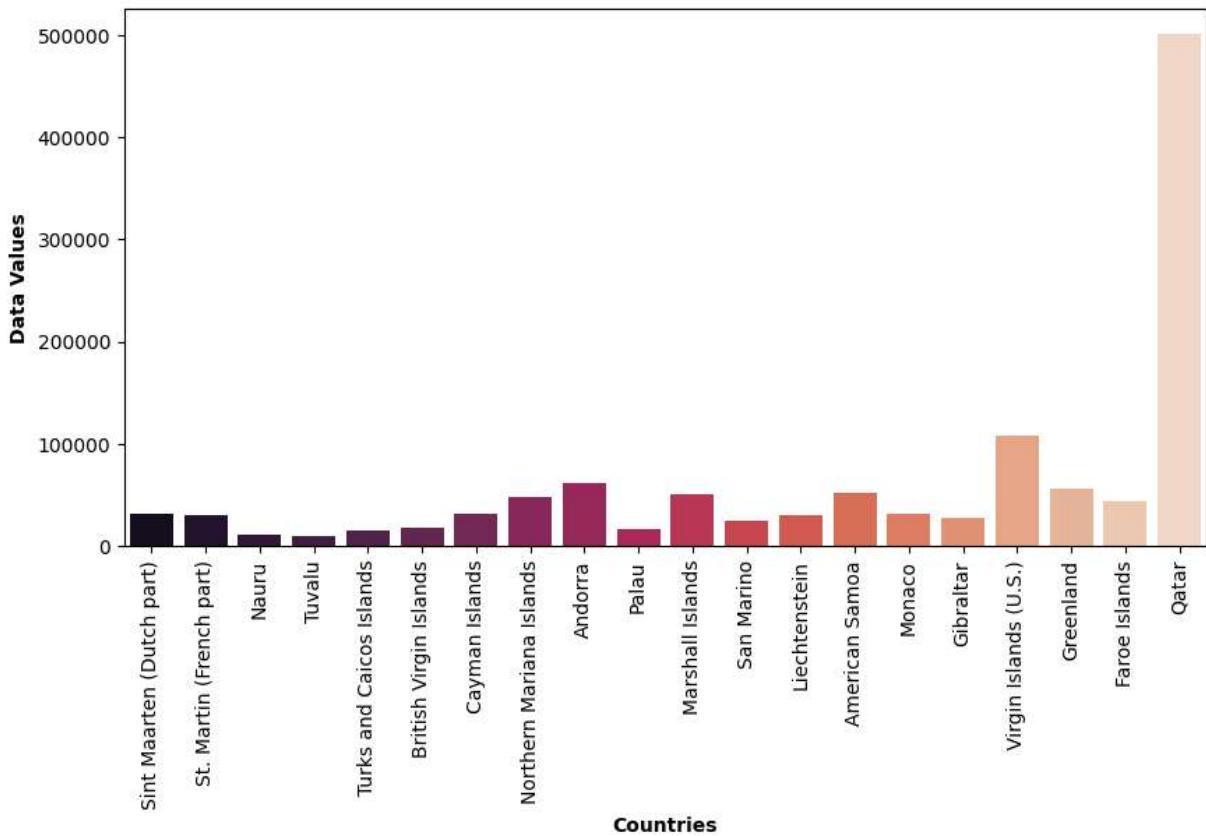
1983- data values from 1960 to 2023**1984- data values from 1960 to 2023**

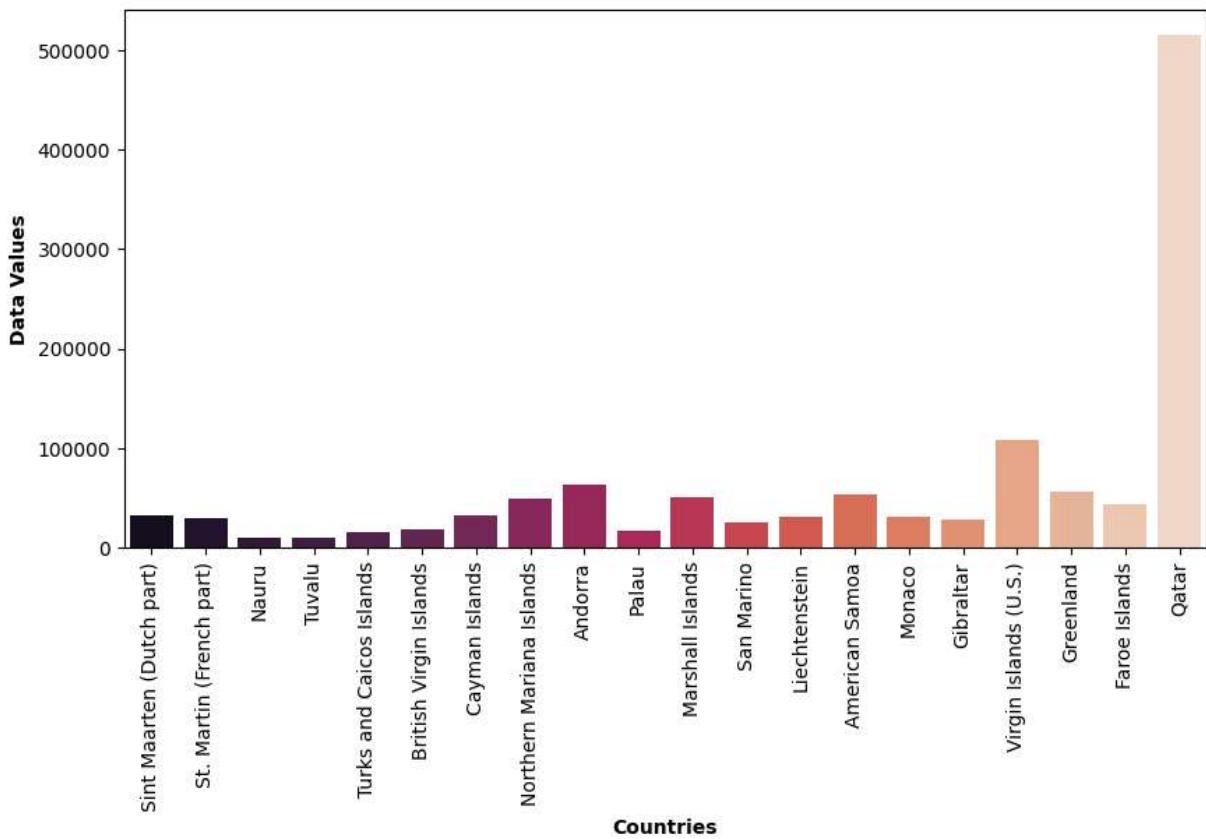
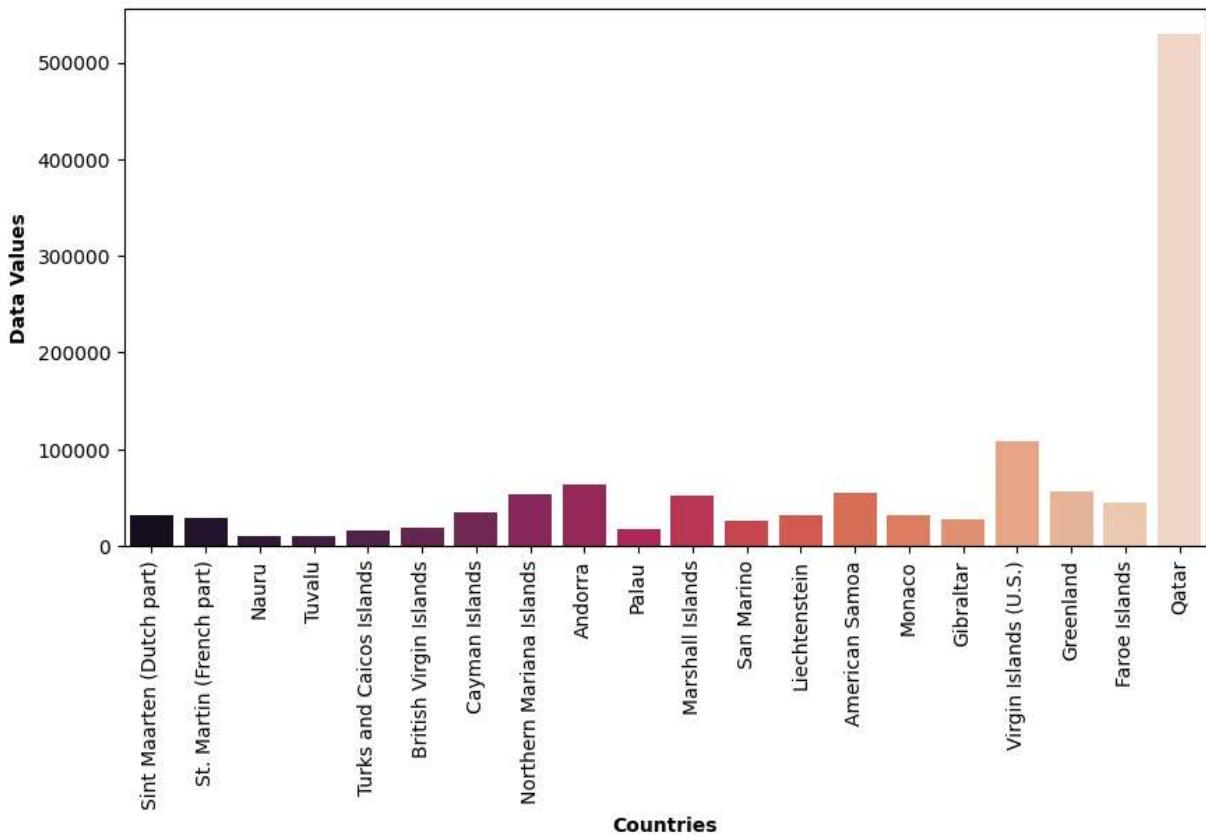
1985- data values from 1960 to 2023**1986- data values from 1960 to 2023**

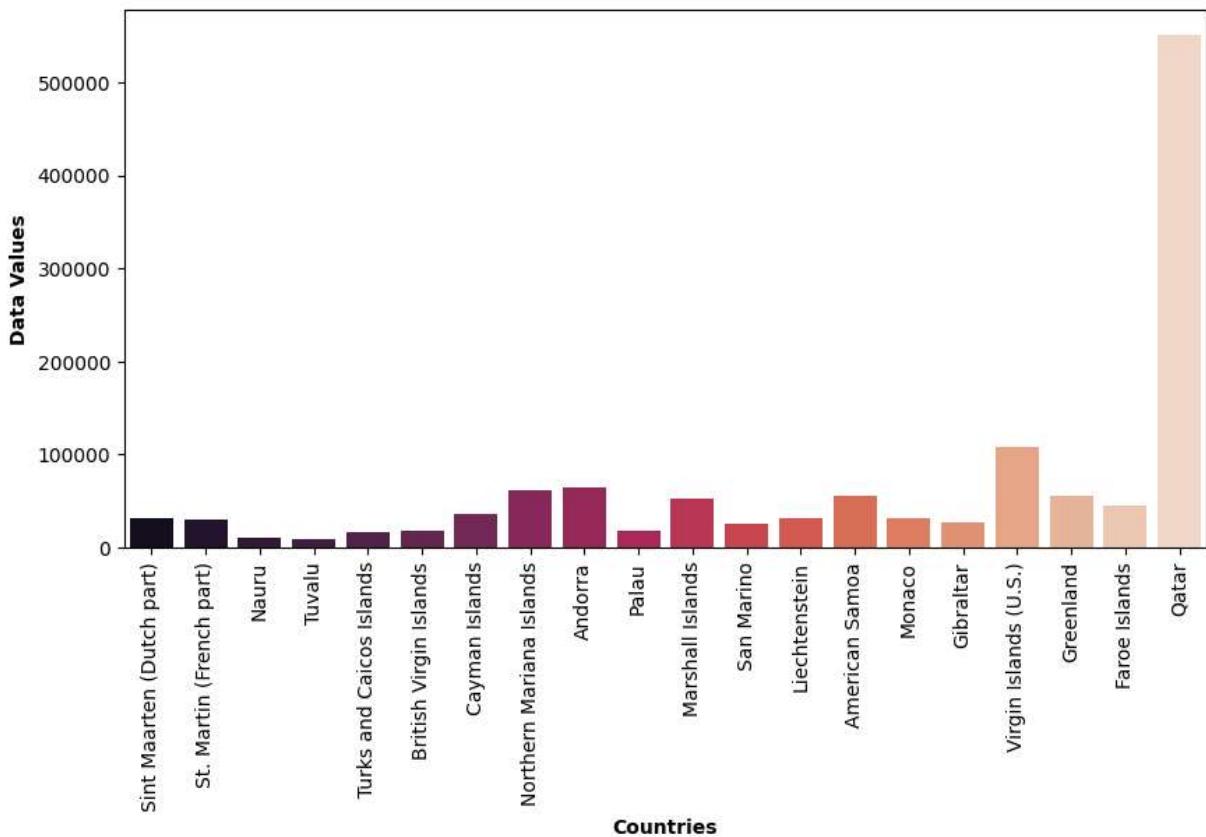
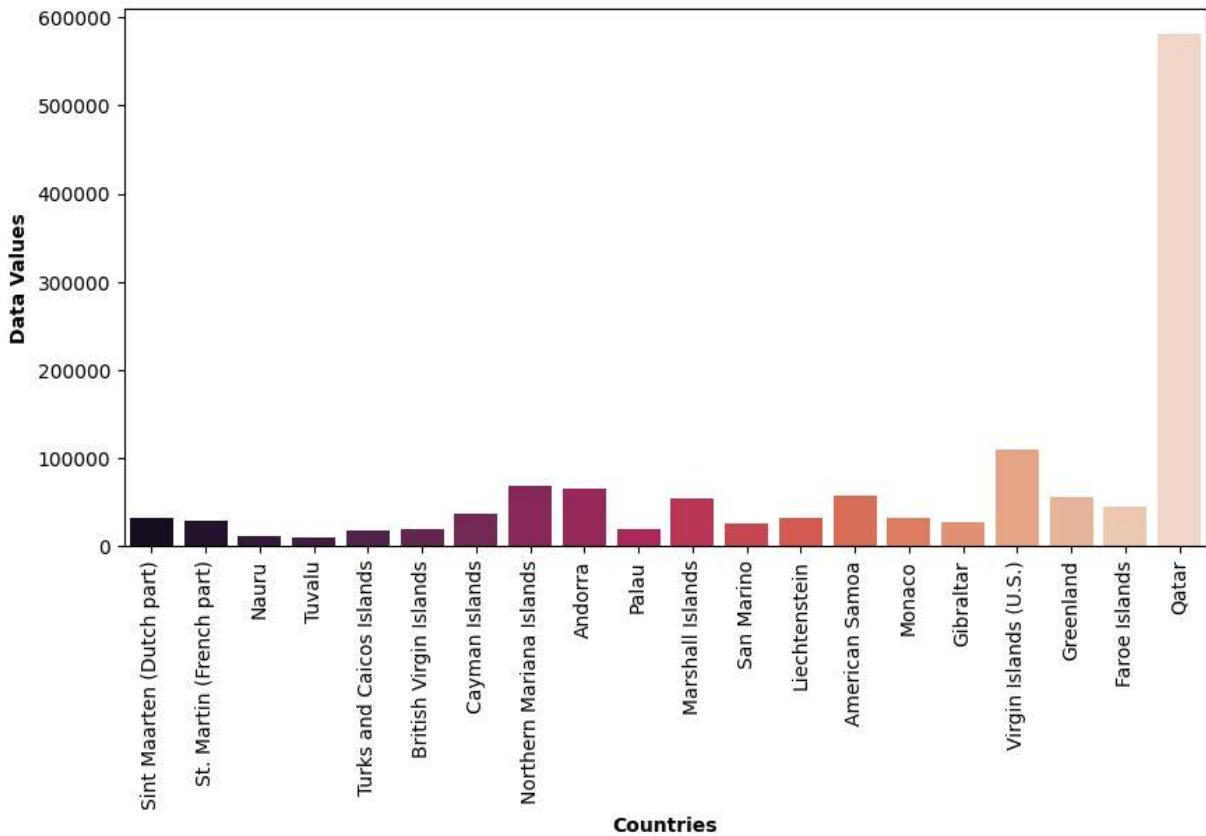
1987- data values from 1960 to 2023**1988- data values from 1960 to 2023**

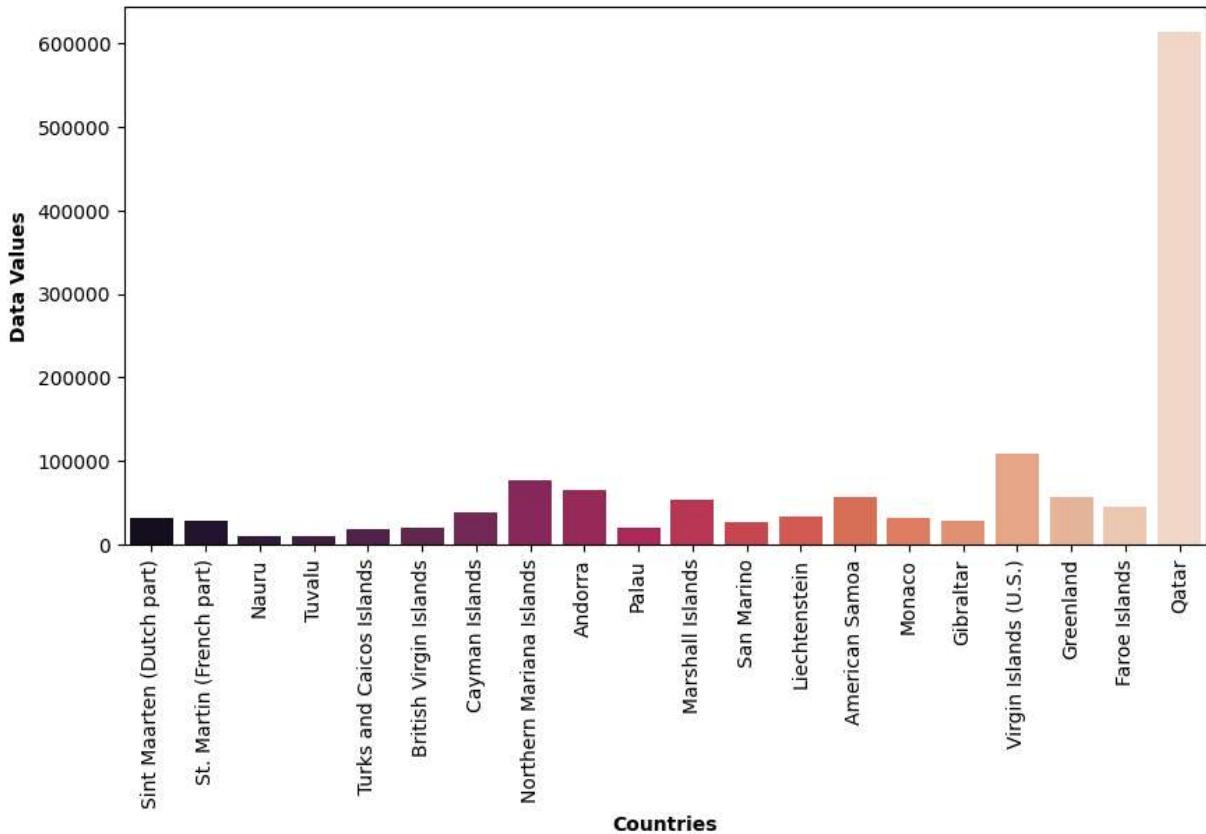
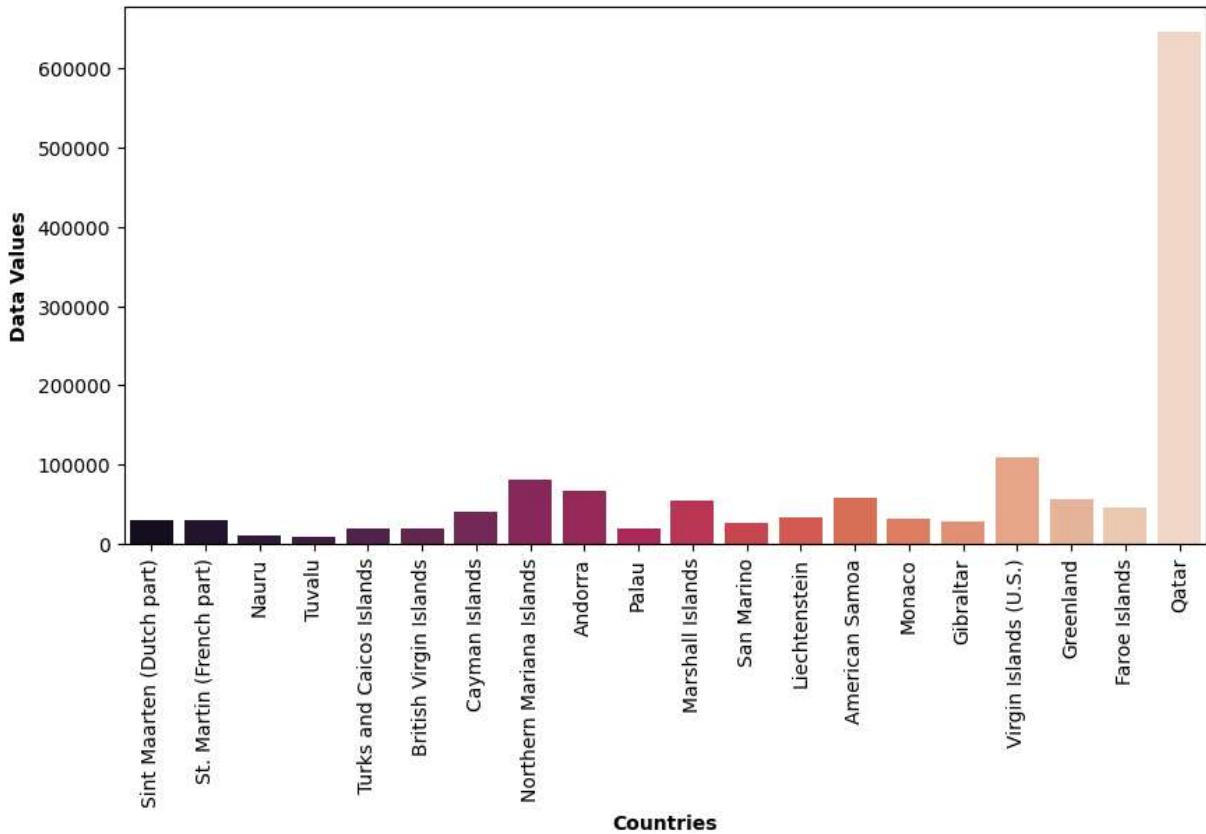
1989- data values from 1960 to 2023**1990- data values from 1960 to 2023**

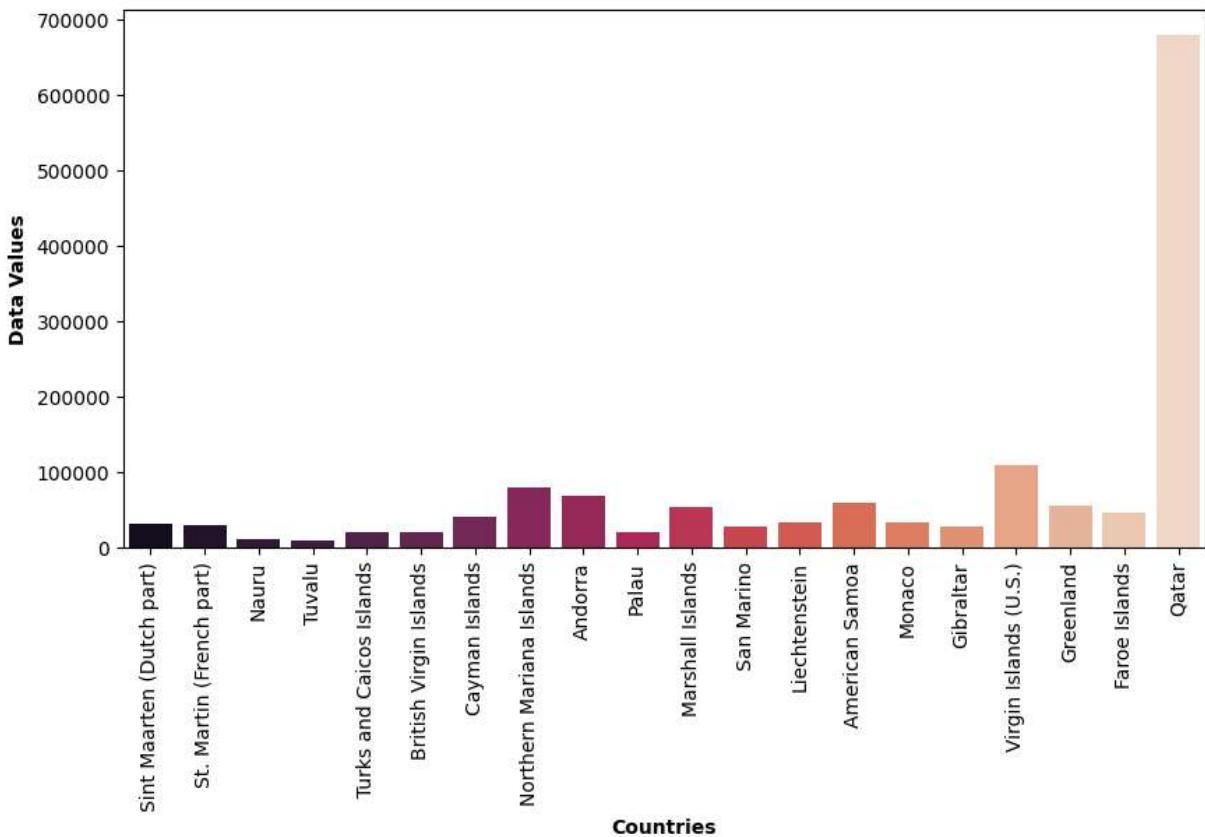
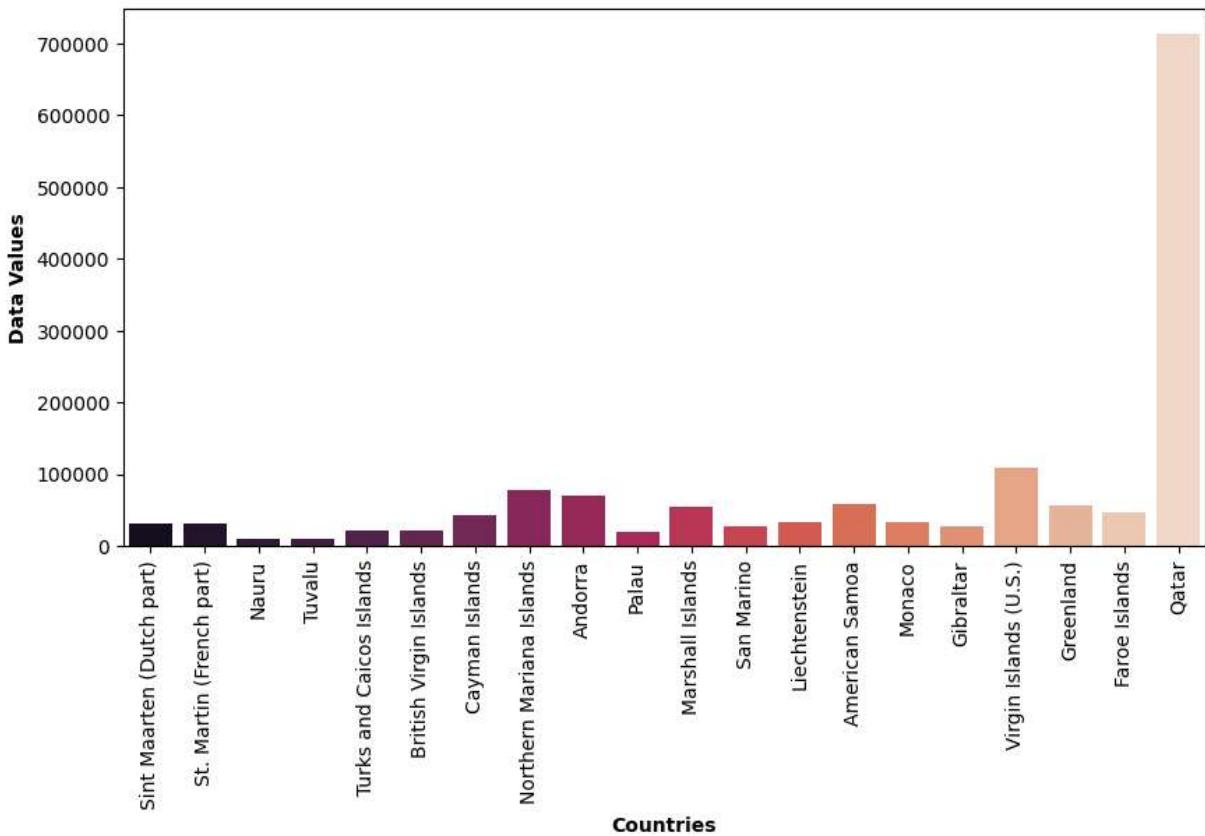
1991- data values from 1960 to 2023**1992- data values from 1960 to 2023**

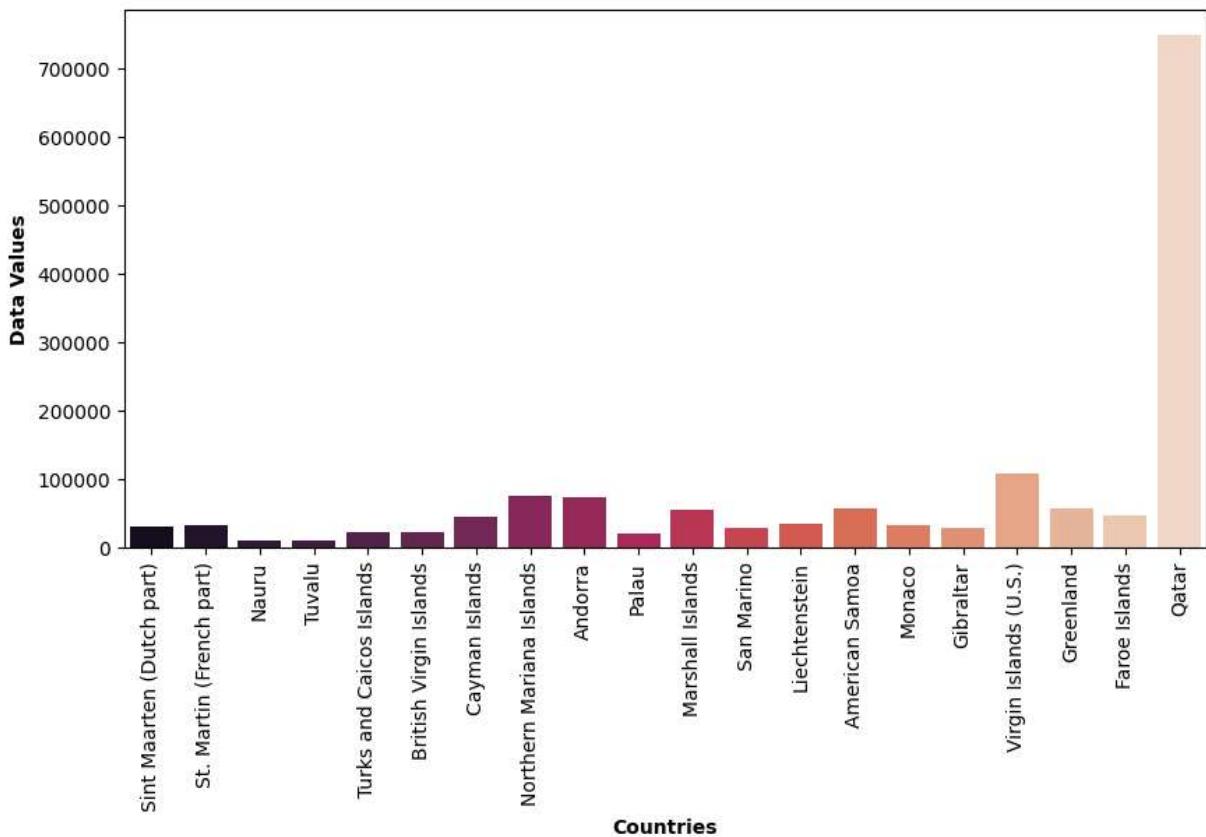
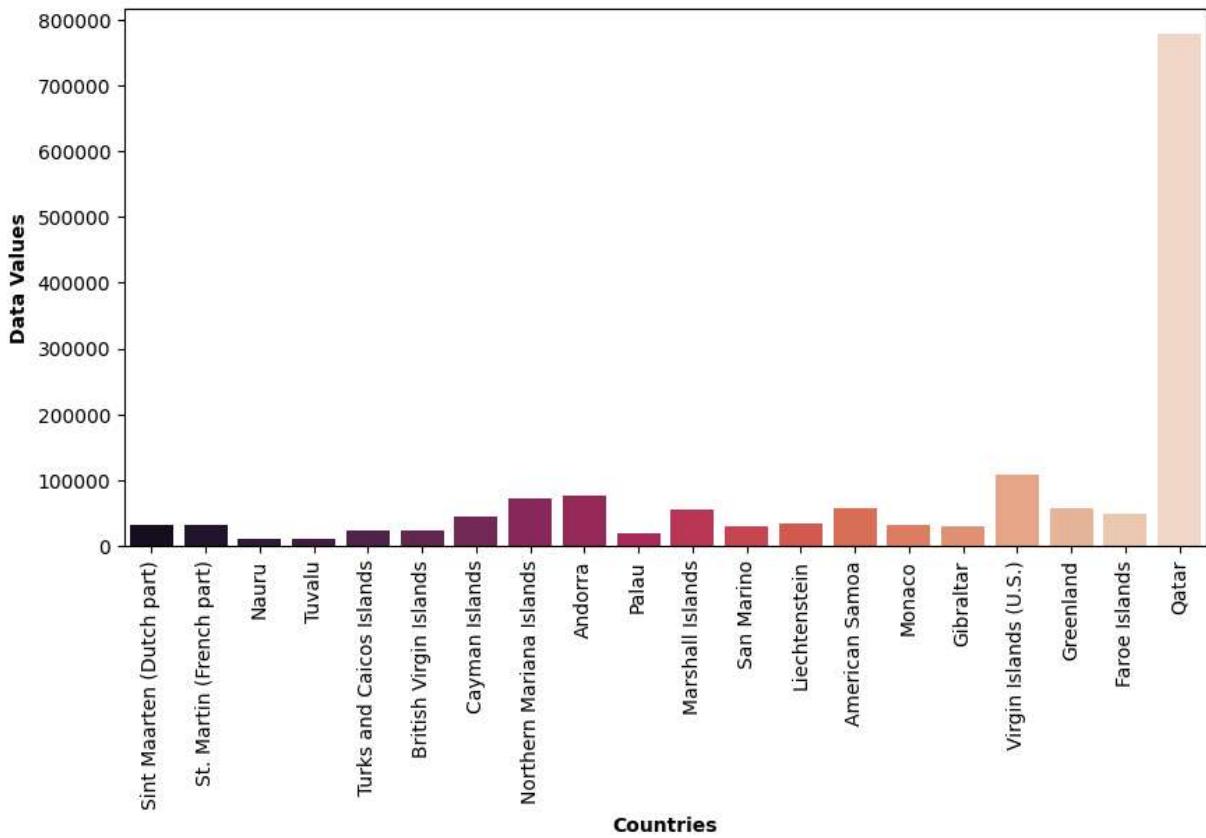
1993- data values from 1960 to 2023**1994- data values from 1960 to 2023**

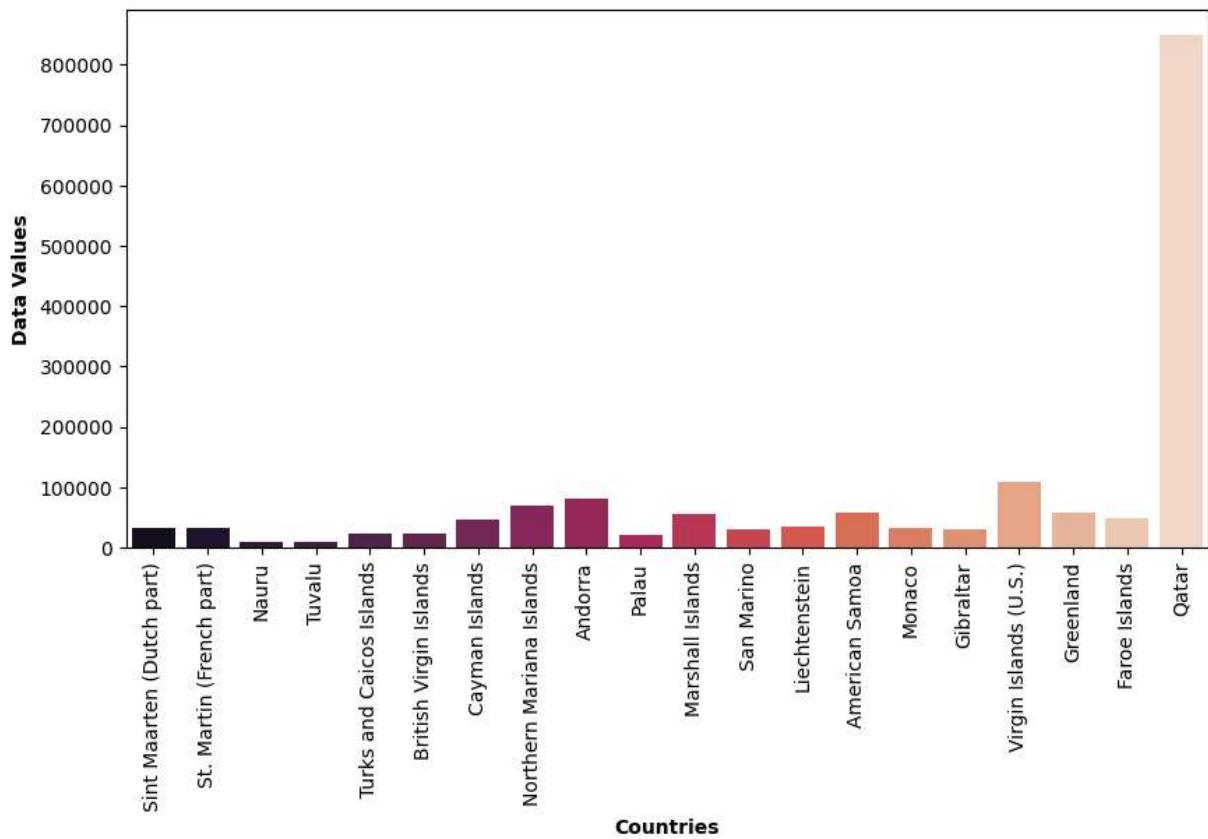
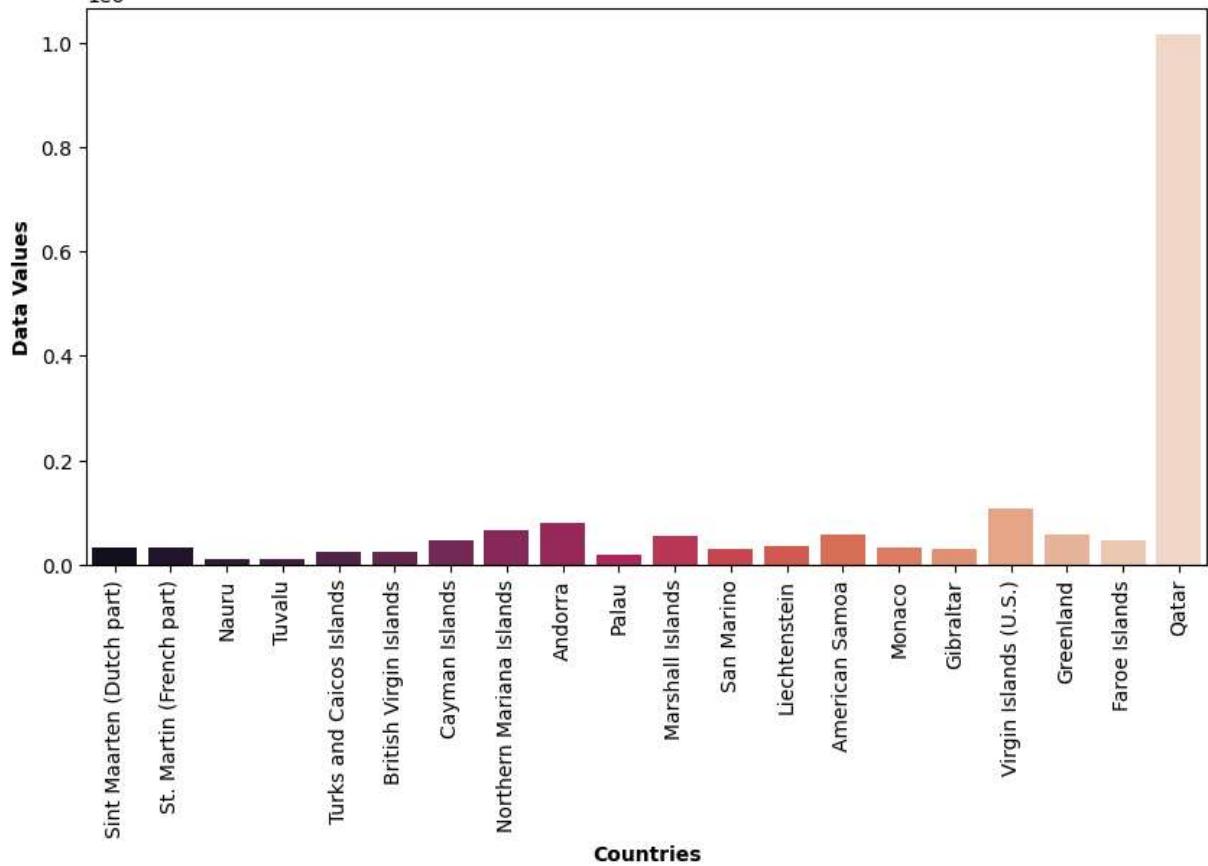
1995- data values from 1960 to 2023**1996- data values from 1960 to 2023**

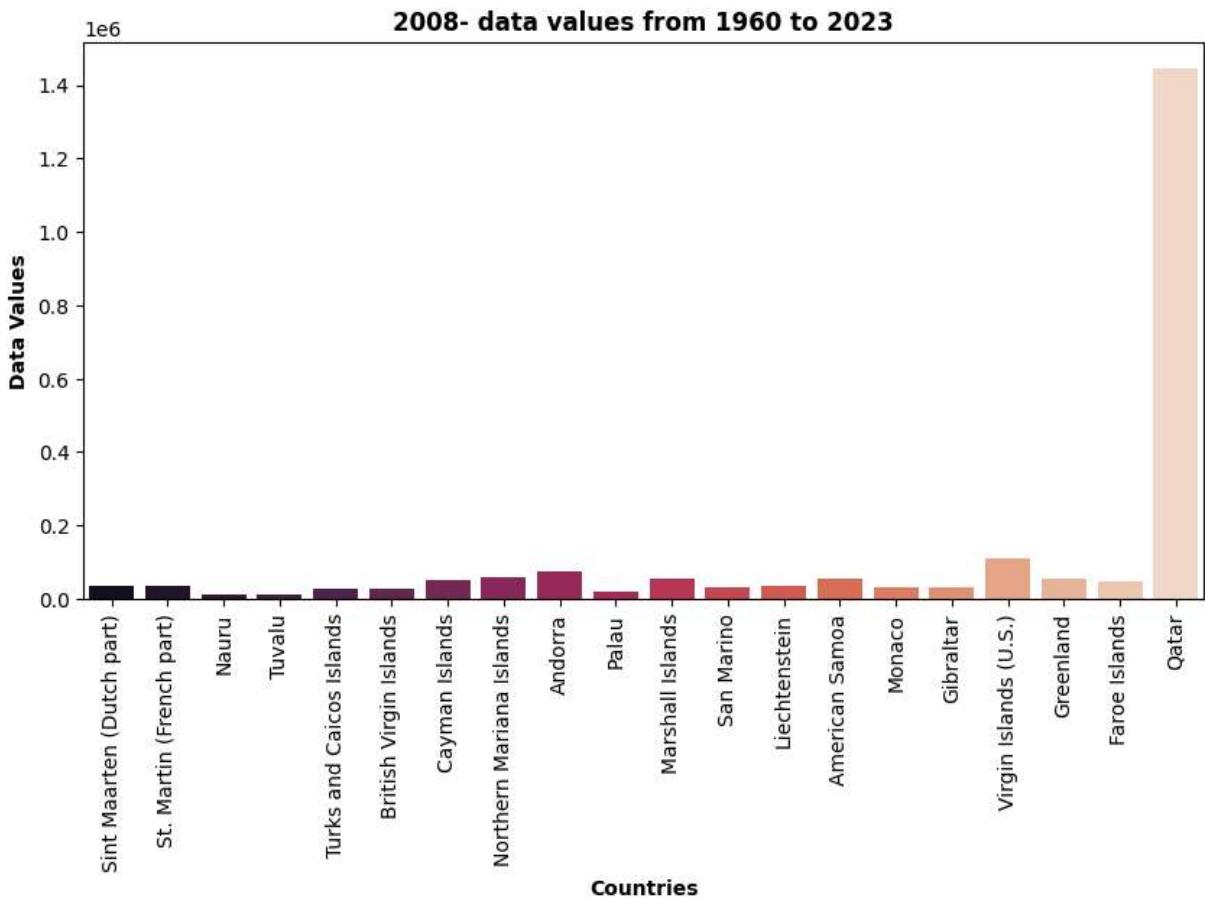
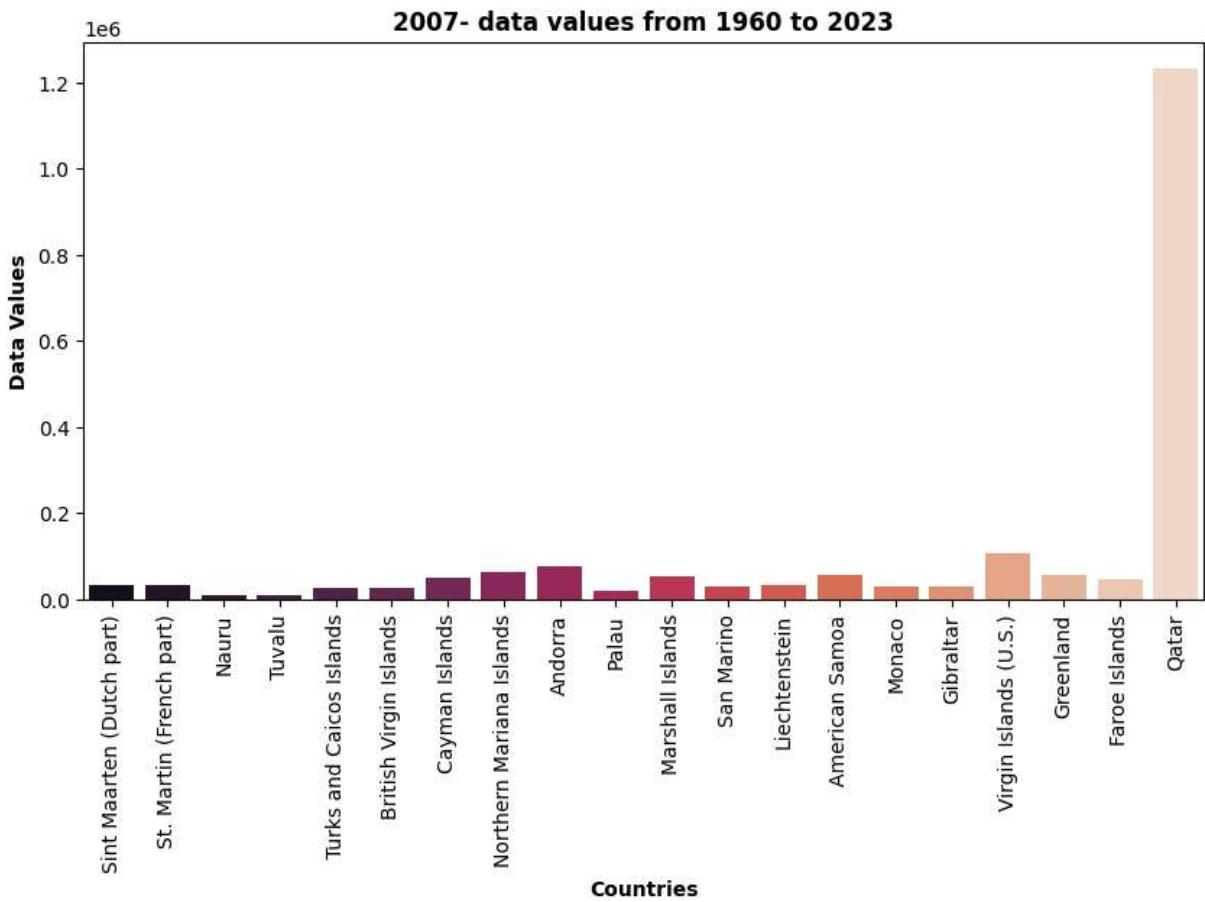
1997- data values from 1960 to 2023**1998- data values from 1960 to 2023**

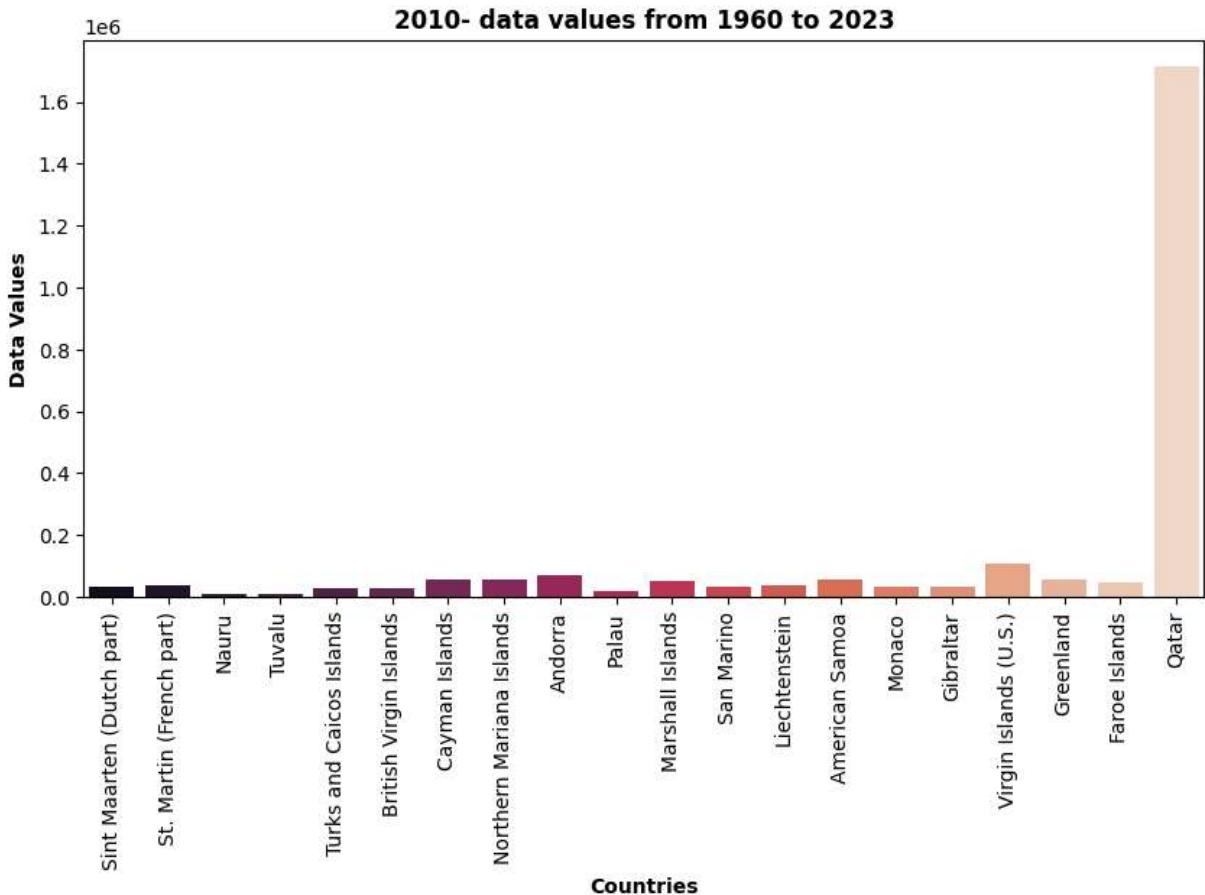
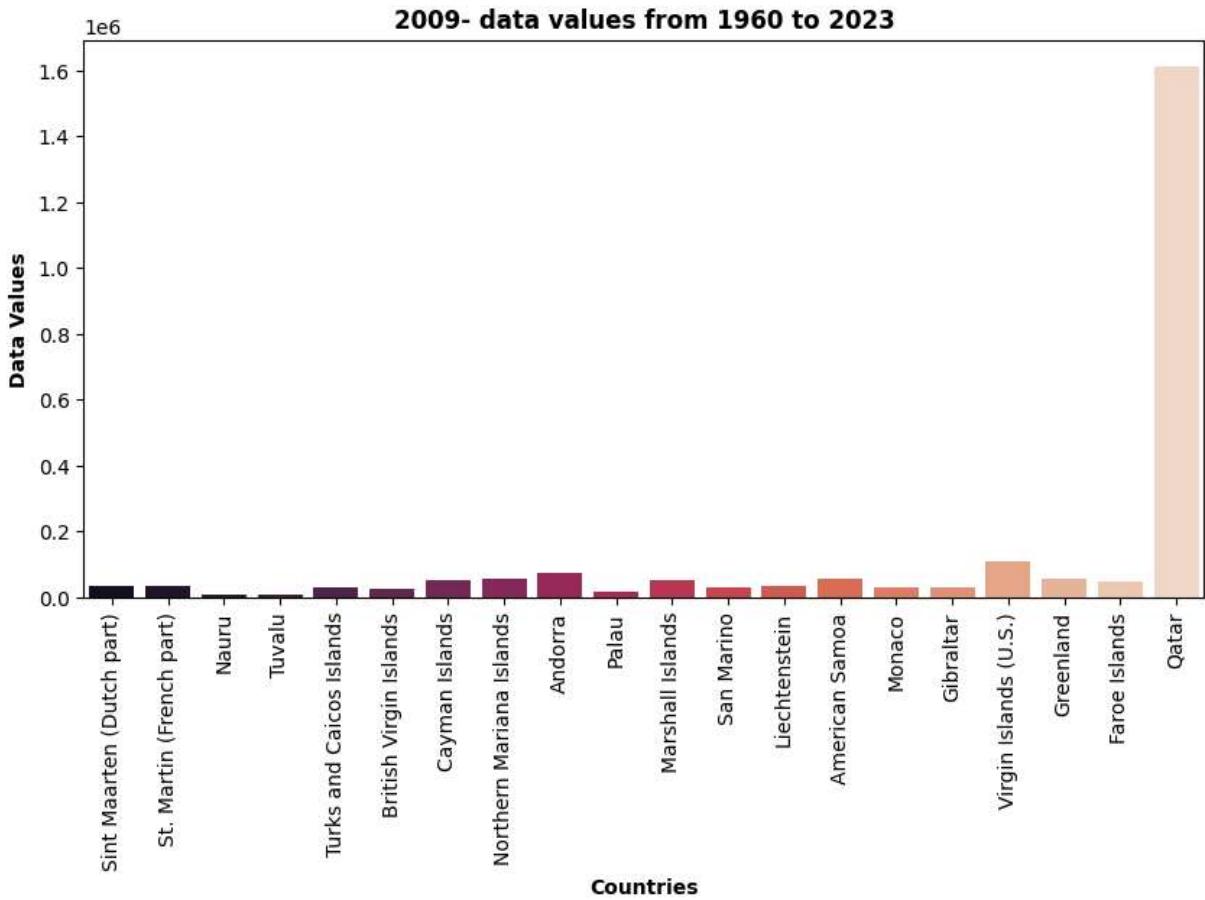
1999- data values from 1960 to 2023**2000- data values from 1960 to 2023**

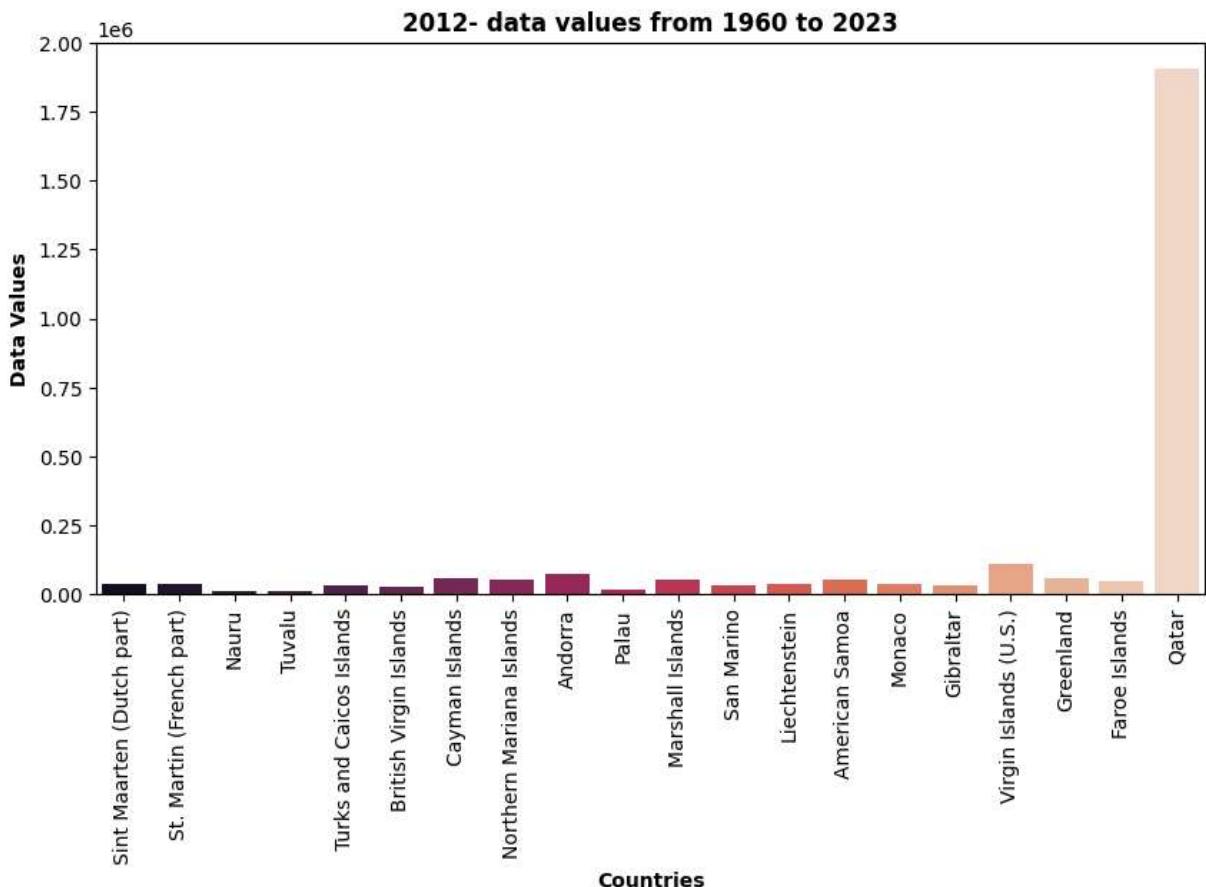
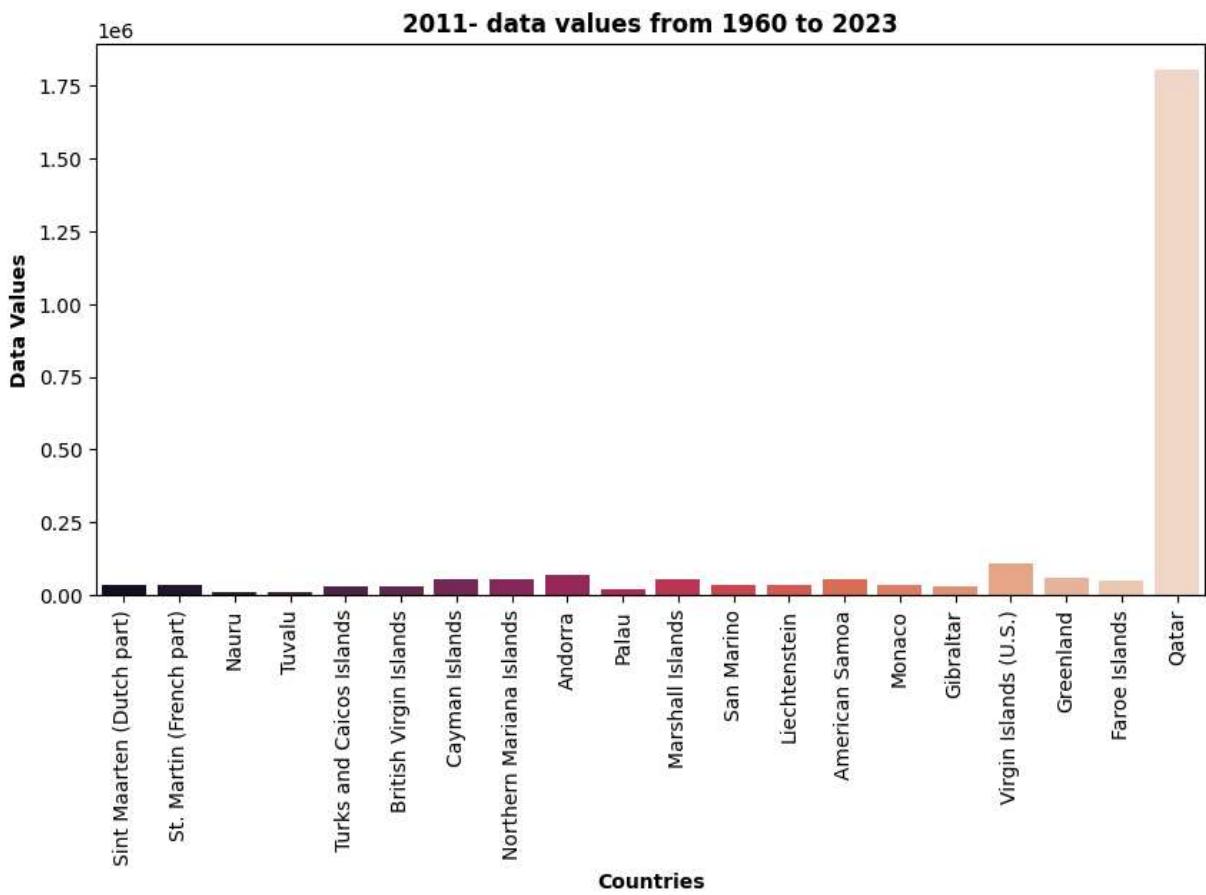
2001- data values from 1960 to 2023**2002- data values from 1960 to 2023**

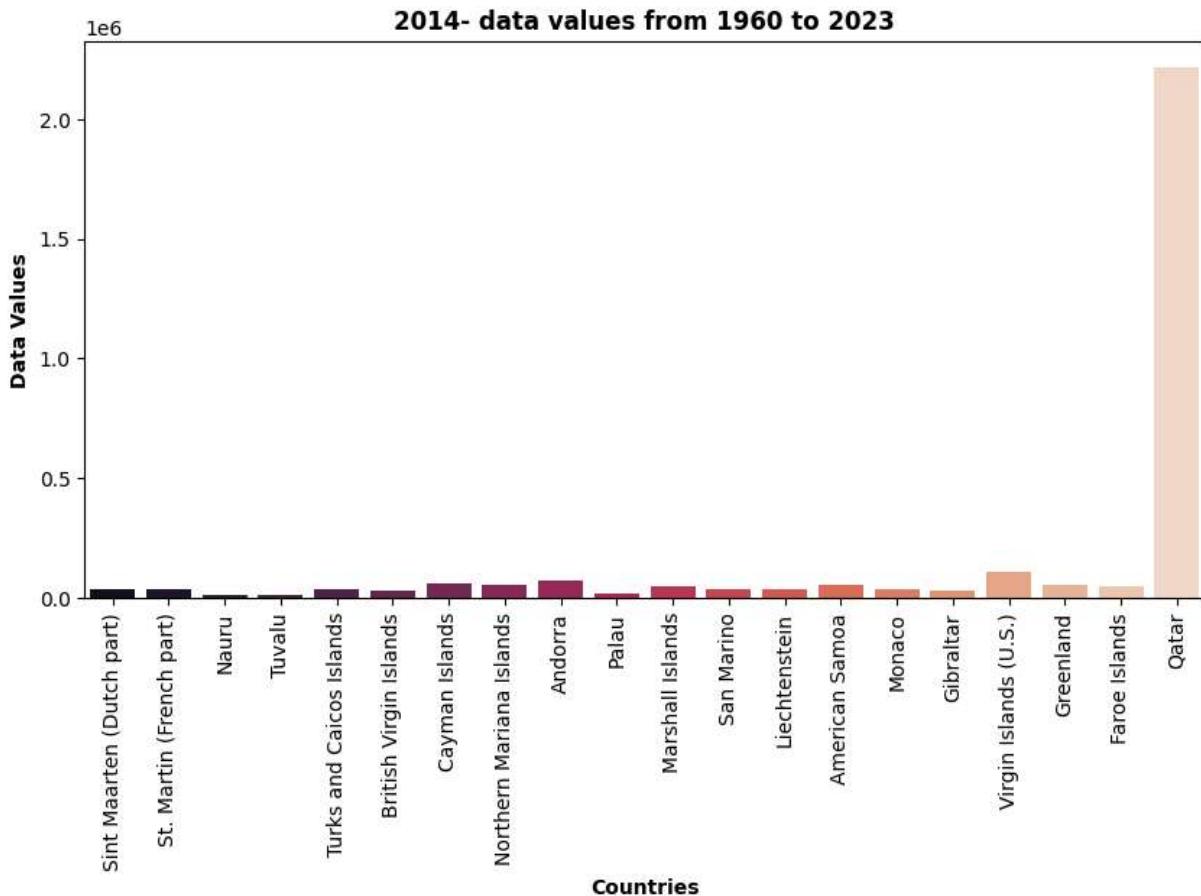
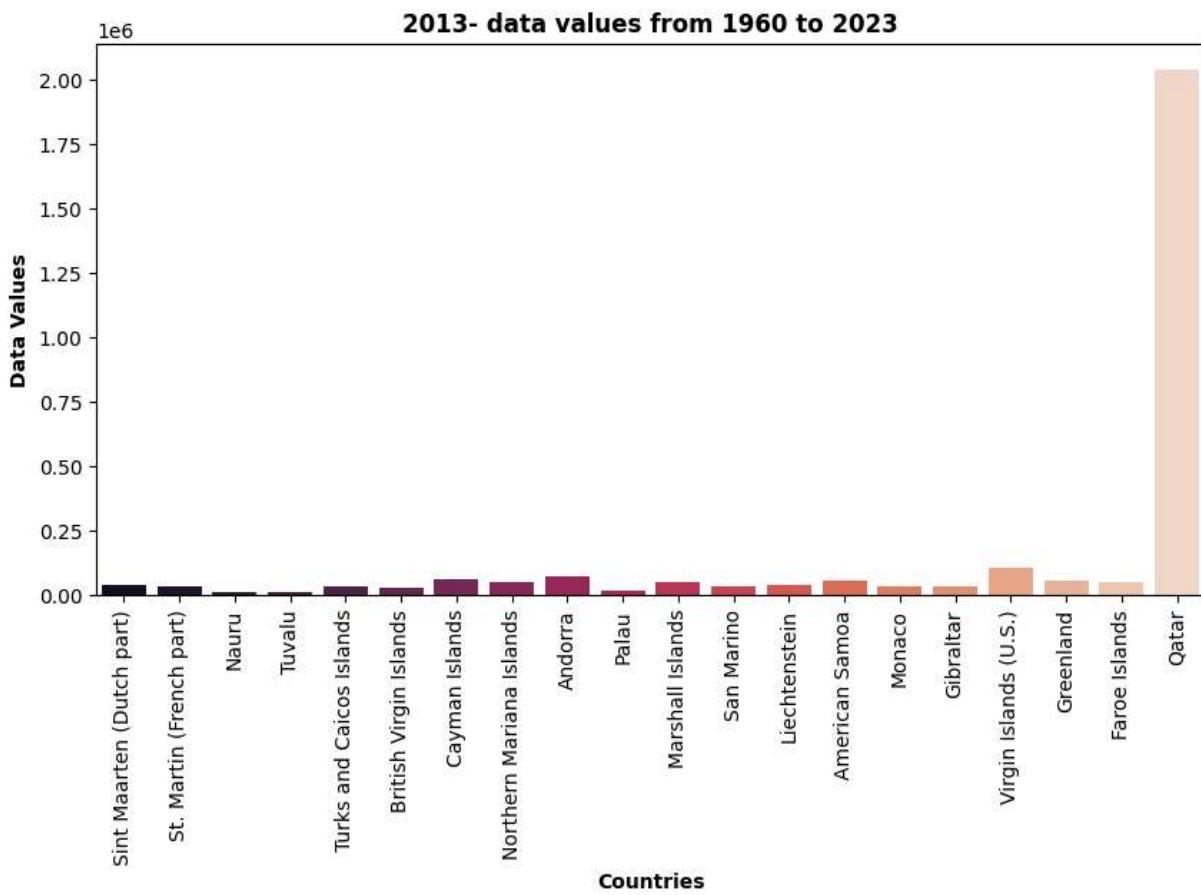
2003- data values from 1960 to 2023**2004- data values from 1960 to 2023**

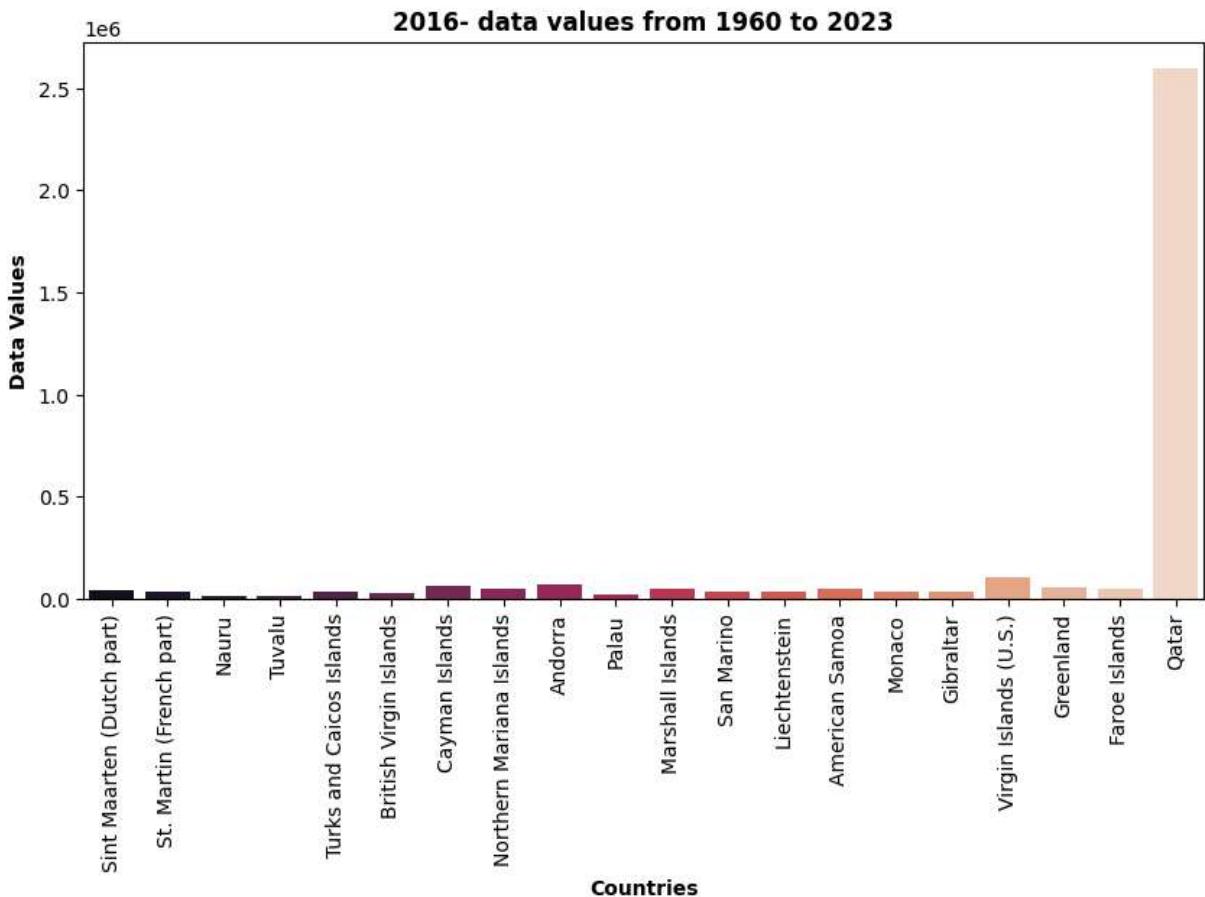
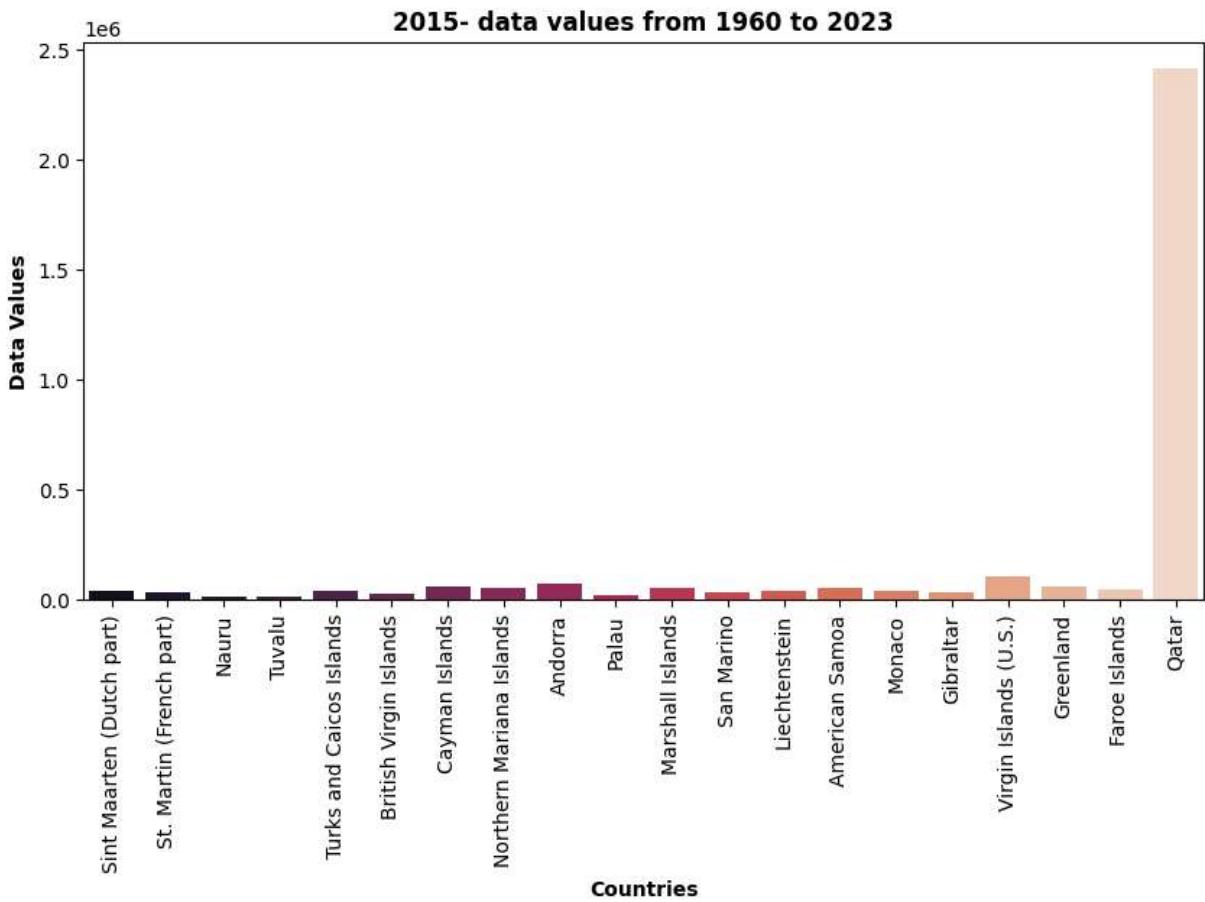
2005- data values from 1960 to 2023**2006- data values from 1960 to 2023**

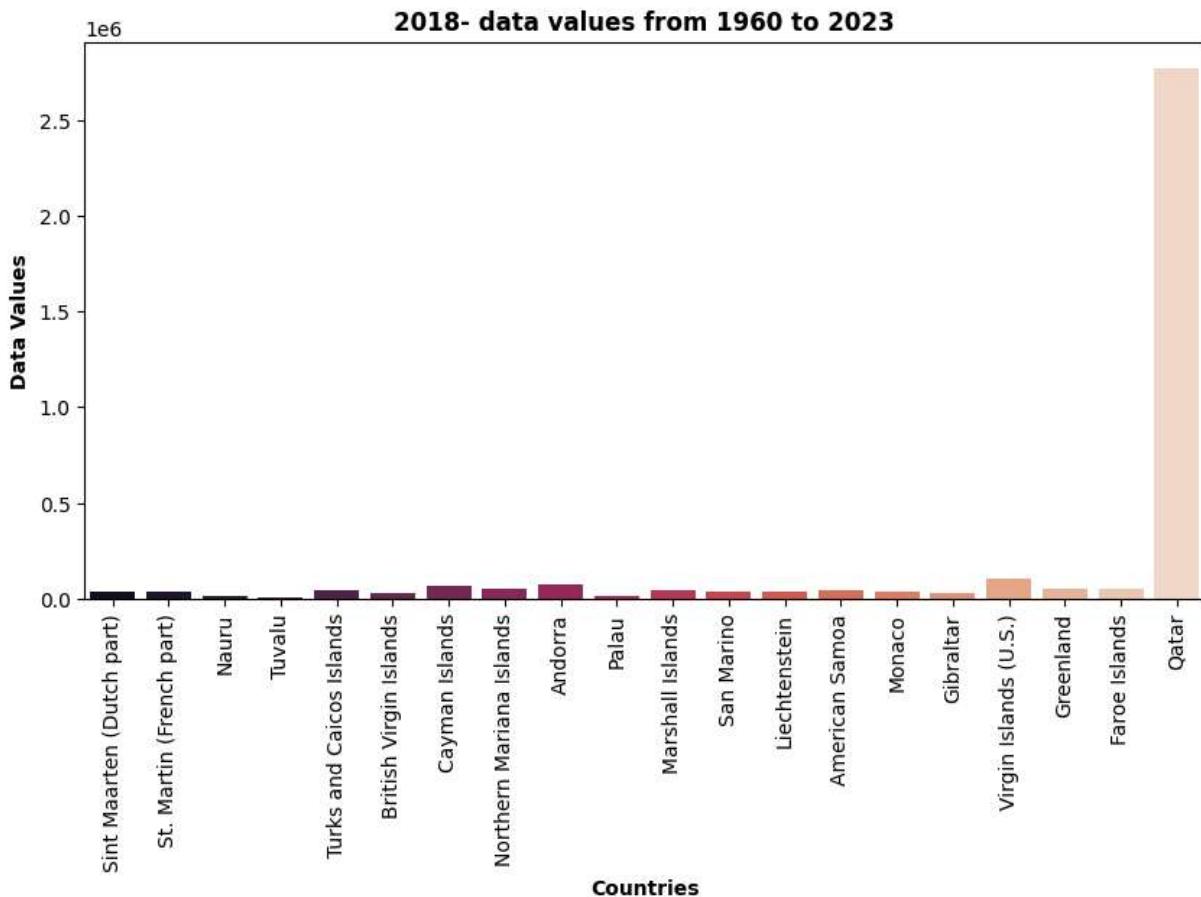
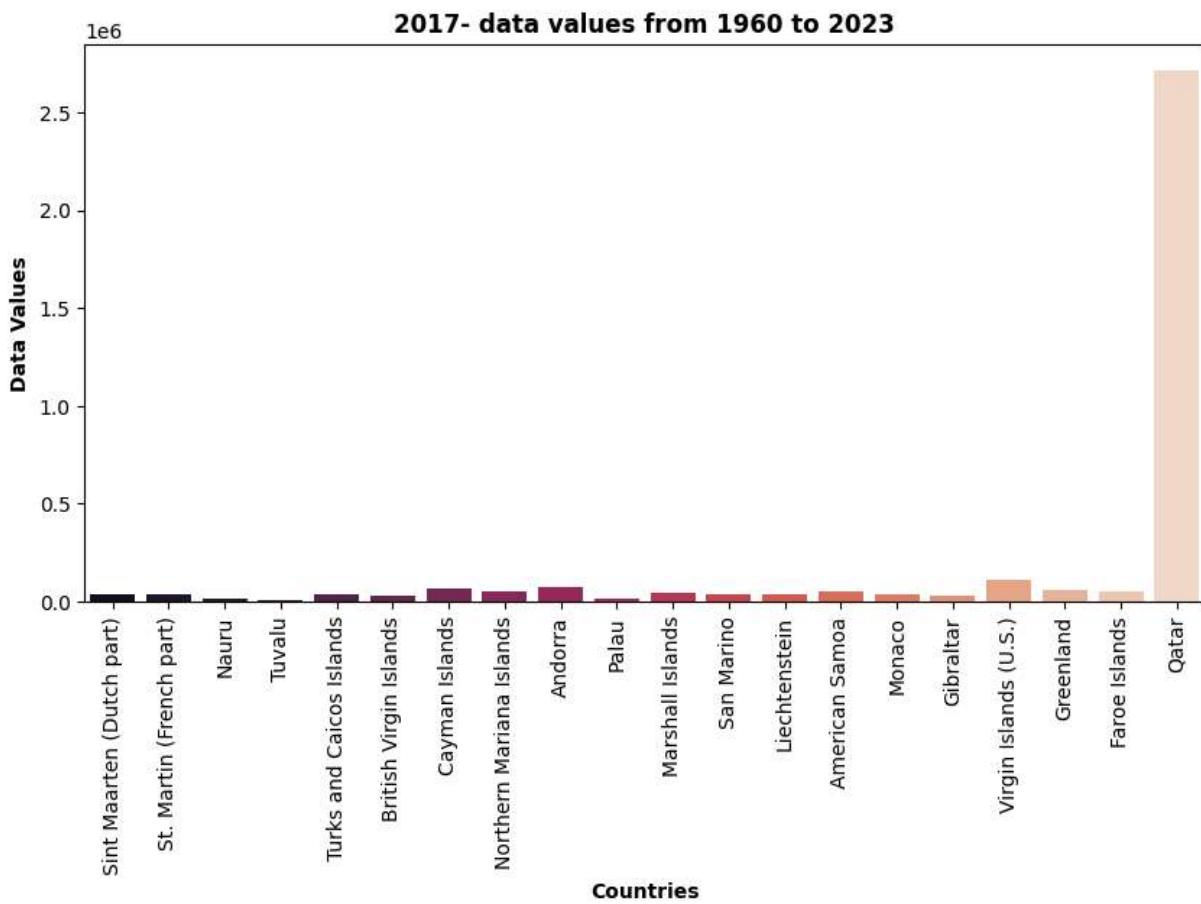


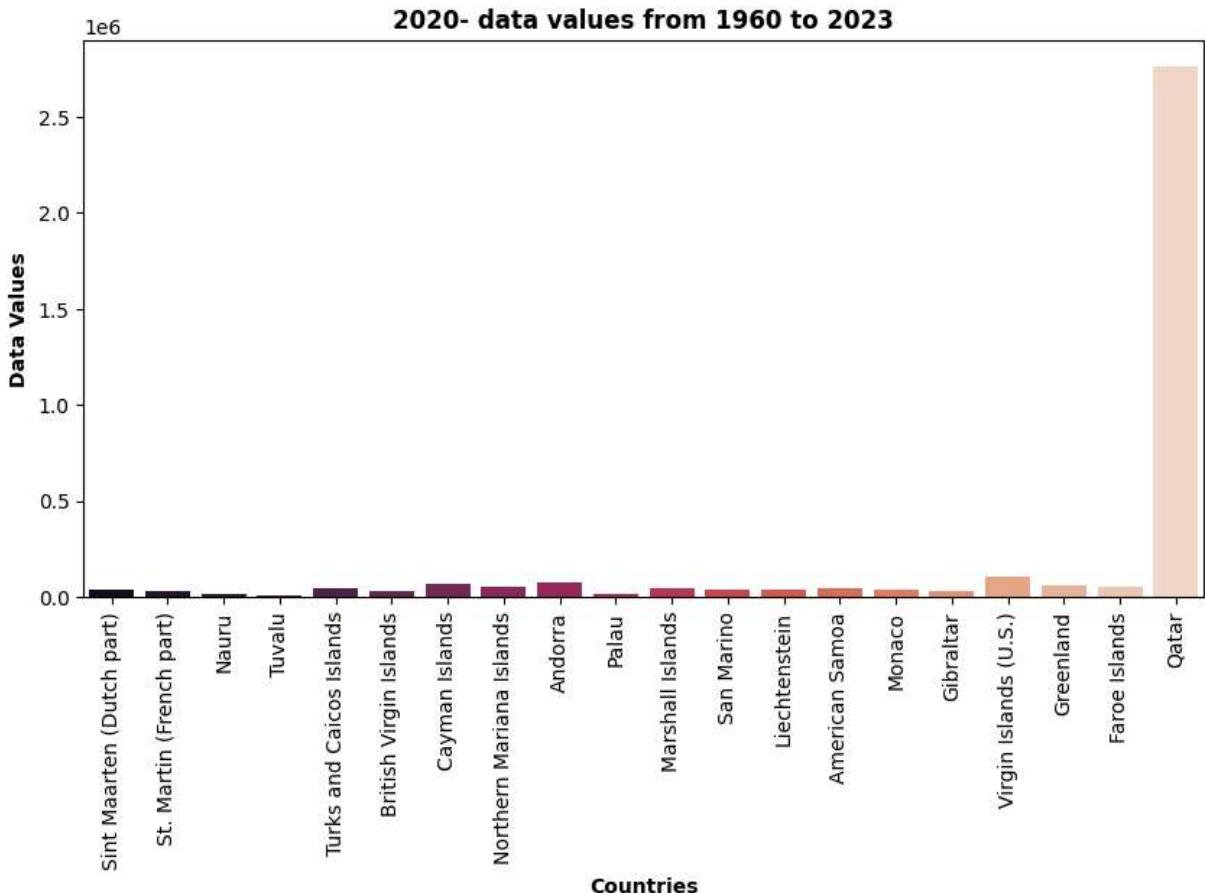
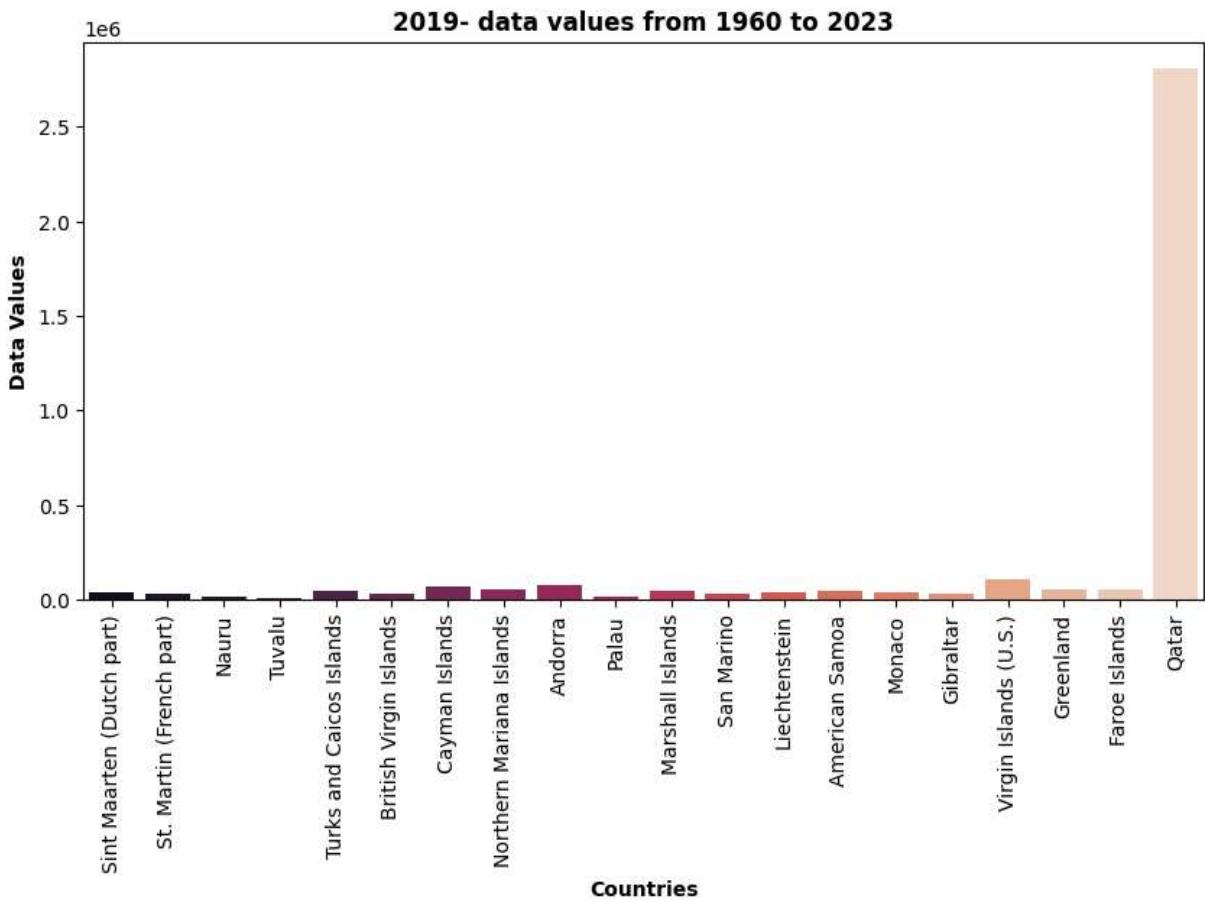


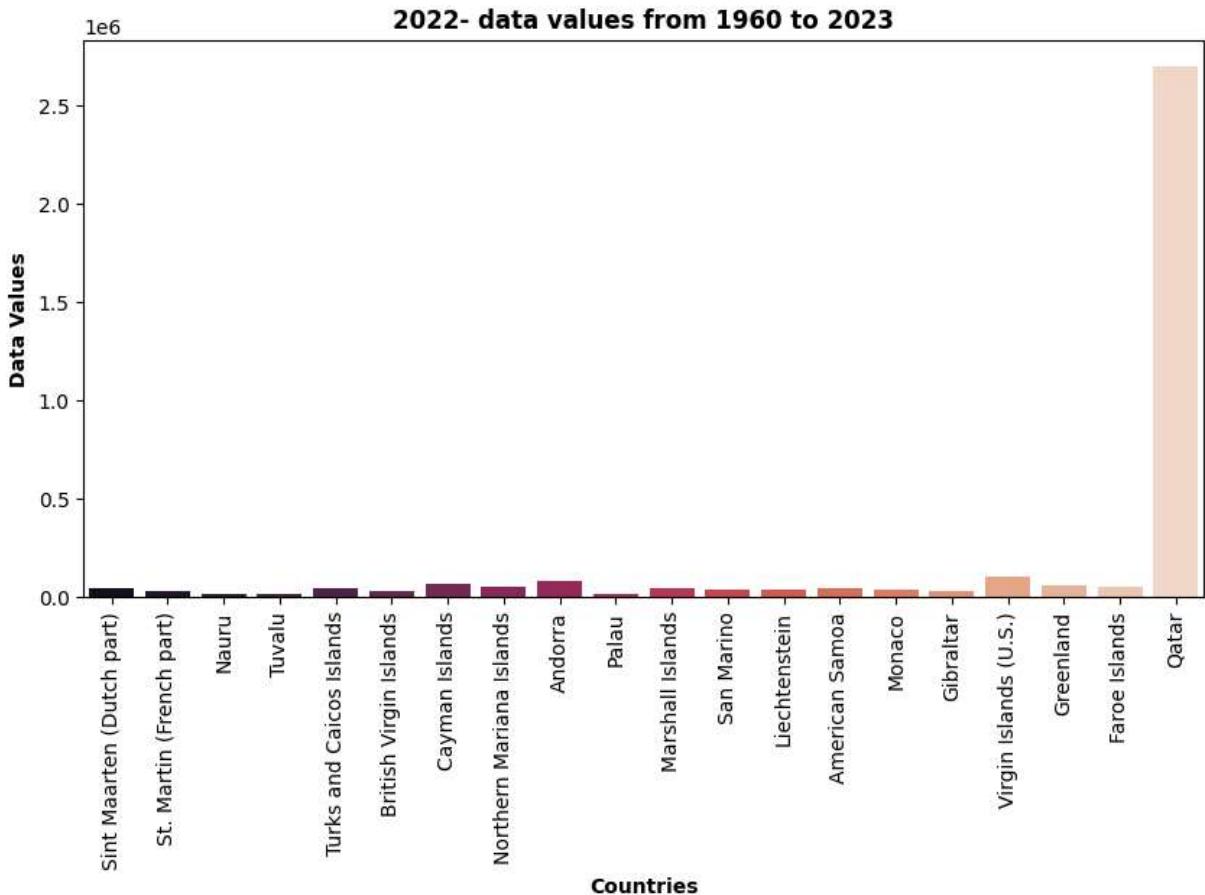
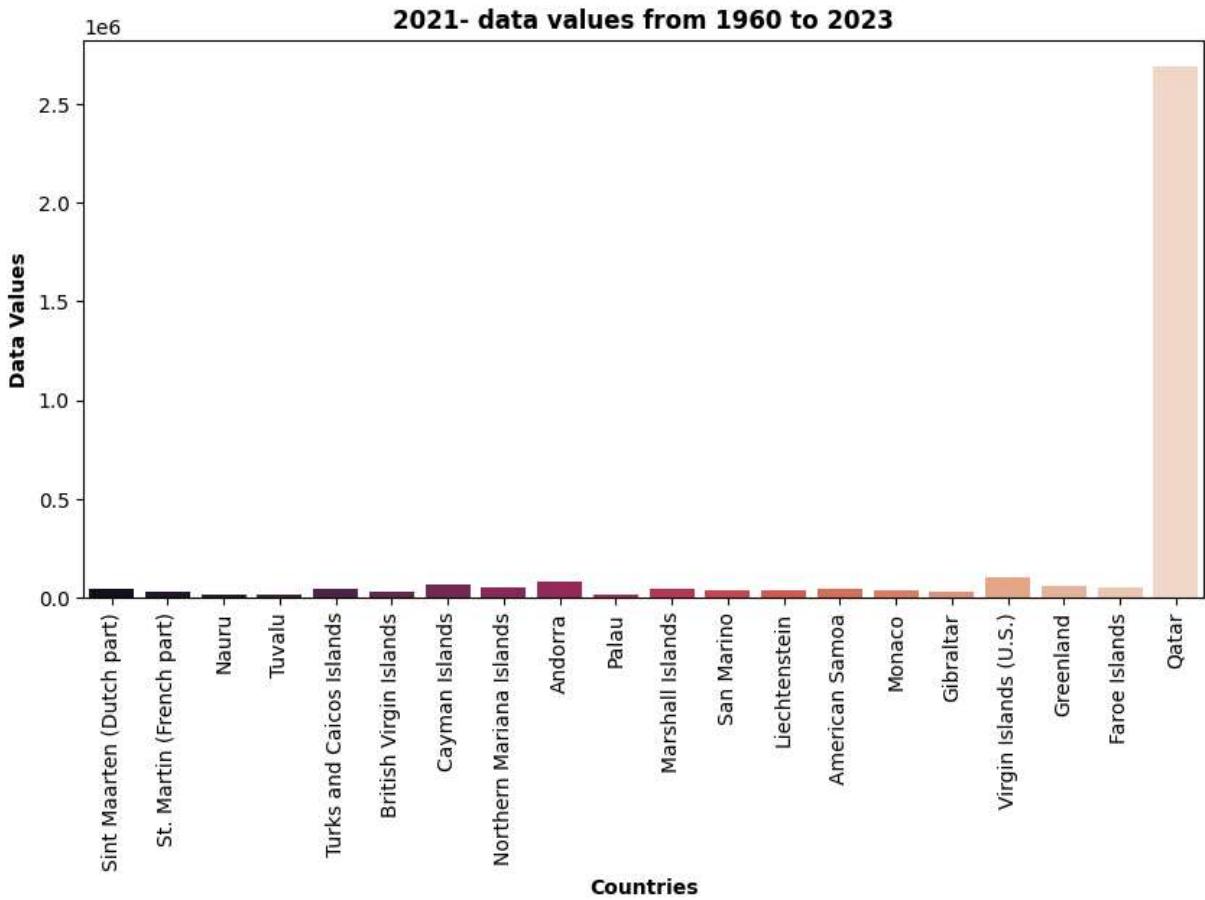


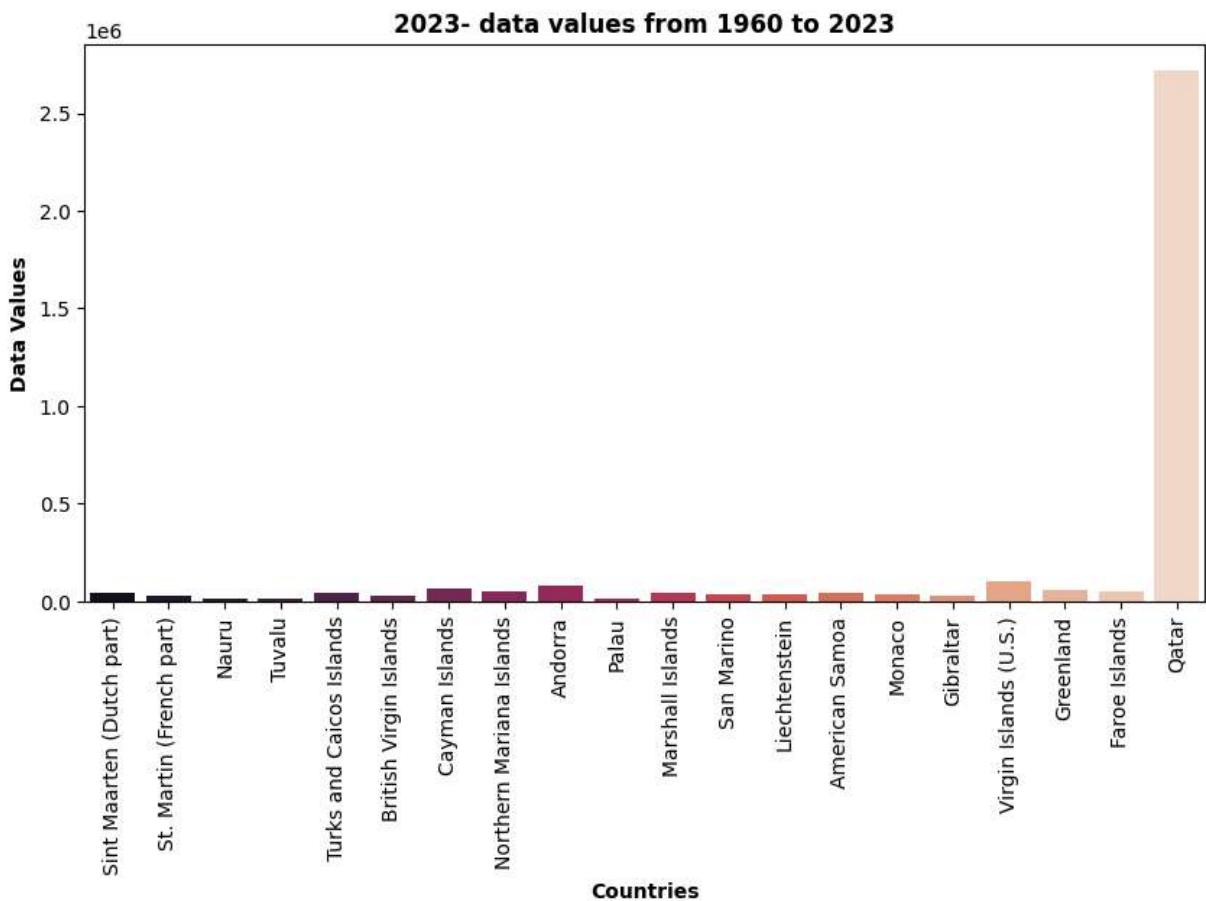












```
In [31]: country_by_2023= data.sort_values(by='2023').head(20)
country_by_2023
```

Out[31]:

	Country Name	1960	1961	1962	1963	1964	1965	1966	1967
245	Tuvalu	5404.0	5436.0	5471.0	5503.0	5525.0	5548.0	5591.0	5657.0
179	Nauru	4582.0	4753.0	4950.0	5198.0	5484.0	5804.0	6021.0	6114.0
188	Palau	9446.0	9639.0	9851.0	10076.0	10318.0	10563.0	10813.0	10992.0
255	British Virgin Islands	7850.0	7885.0	7902.0	7919.0	7949.0	8018.0	8139.0	8337.0
147	St. Martin (French part)	4135.0	4258.0	4388.0	4524.0	4666.0	4832.0	5044.0	5294.0
84	Gibraltar	21822.0	21907.0	22249.0	22796.0	23347.0	23910.0	24477.0	25047.0
212	San Marino	15556.0	15895.0	16242.0	16583.0	16926.0	17273.0	17588.0	17907.0
149	Monaco	21797.0	21907.0	22106.0	22442.0	22766.0	23022.0	23198.0	23281.0
137	Liechtenstein	16472.0	16834.0	17221.0	17625.0	18058.0	18500.0	18957.0	19467.0
225	Sint Maarten (Dutch part)	2646.0	2888.0	3171.0	3481.0	3811.0	4161.0	4531.0	4930.0
155	Marshall Islands	15374.0	15867.0	16387.0	16947.0	17537.0	18154.0	18794.0	19665.0
11	American Samoa	20085.0	20626.0	21272.0	21949.0	22656.0	23391.0	24122.0	24848.0
228	Turks and Caicos Islands	5604.0	5625.0	5633.0	5634.0	5642.0	5650.0	5652.0	5662.0
125	St. Kitts and Nevis	56660.0	56247.0	55404.0	54391.0	53255.0	52016.0	50683.0	49269.0
164	Northern Mariana Islands	8702.0	8965.0	9252.0	9561.0	9890.0	10229.0	10577.0	10720.0
78	Faroe Islands	34154.0	34572.0	34963.0	35385.0	35841.0	36346.0	36825.0	37234.0
91	Greenland	32500.0	33700.0	35000.0	36400.0	37600.0	39200.0	40500.0	41900.0
27	Bermuda	44400.0	45500.0	46600.0	47700.0	48900.0	50100.0	51000.0	52000.0
52	Cayman Islands	8473.0	8626.0	8799.0	8985.0	9172.0	9366.0	9566.0	9771.0
57	Dominica	59379.0	60395.0	61224.0	62031.0	62843.0	63744.0	64728.0	65760.0

20 rows × 65 columns

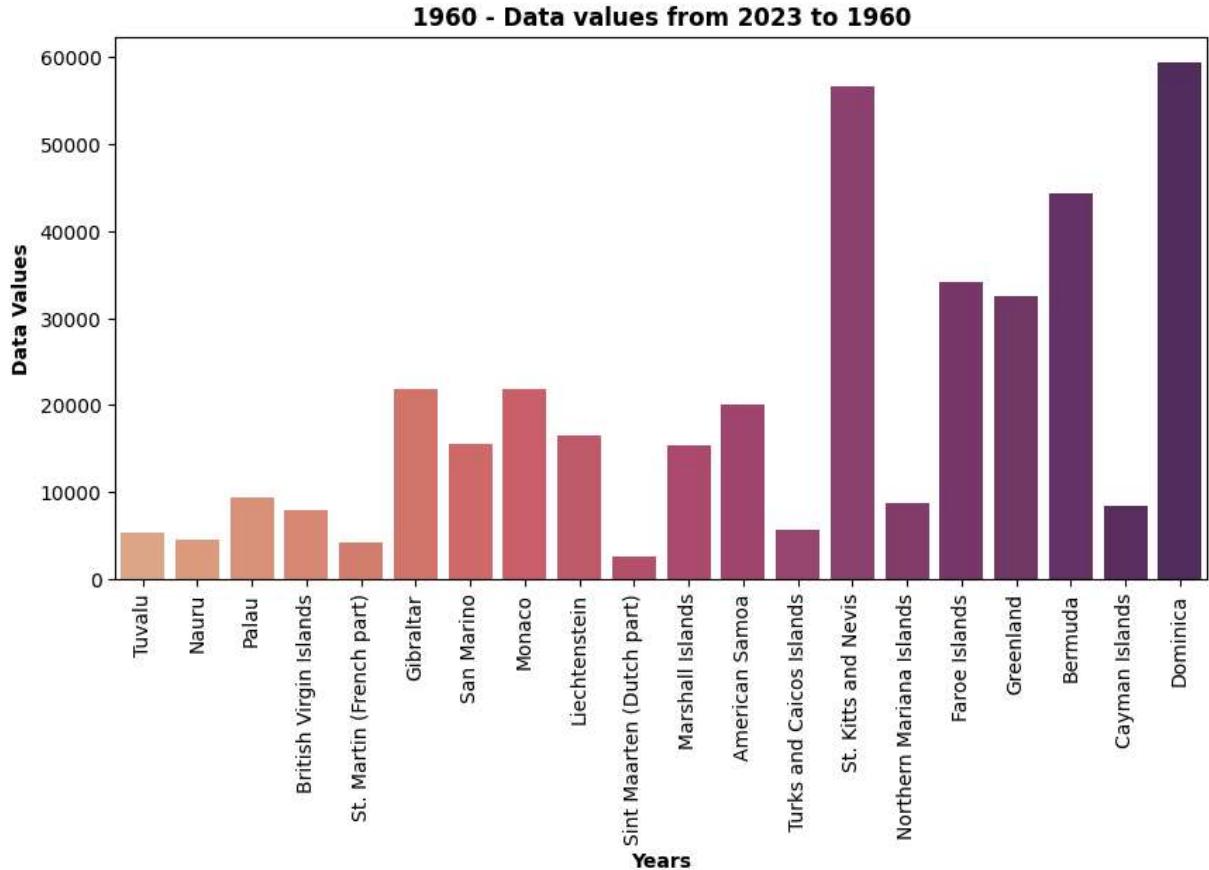


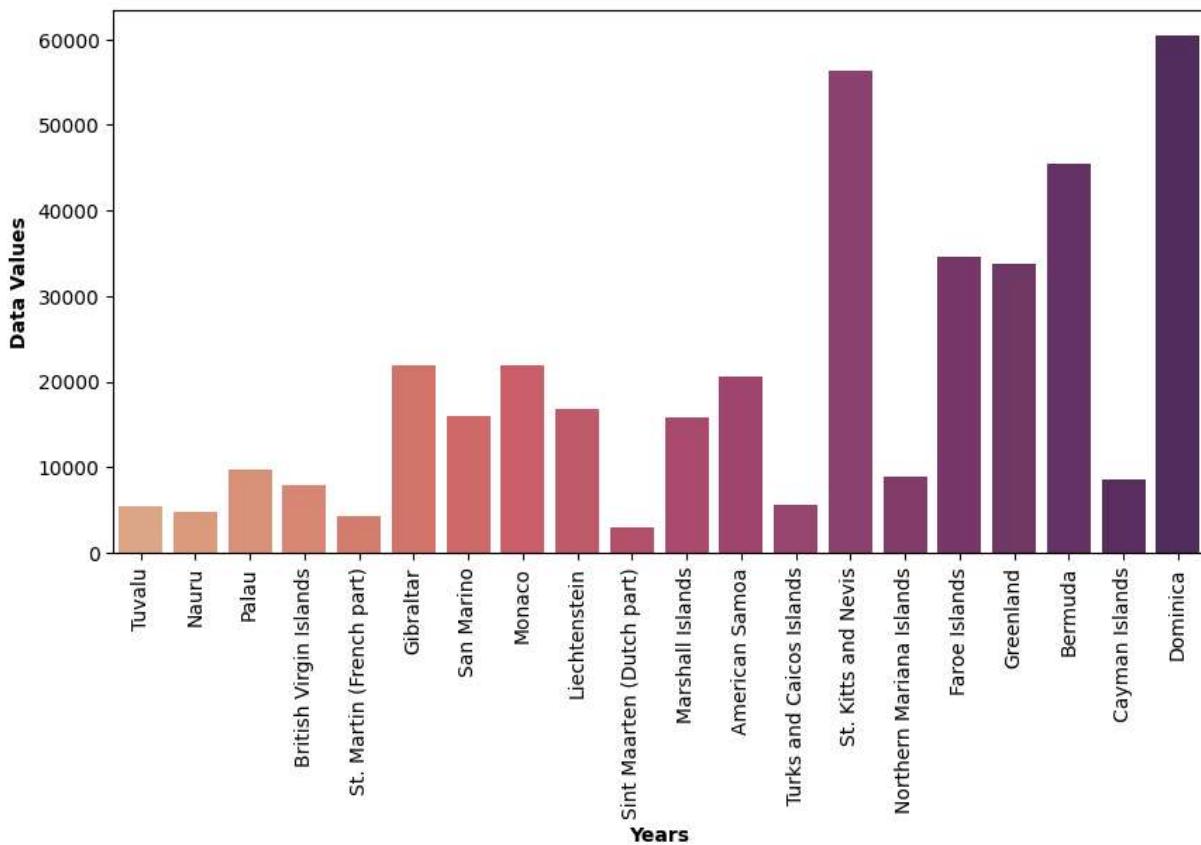
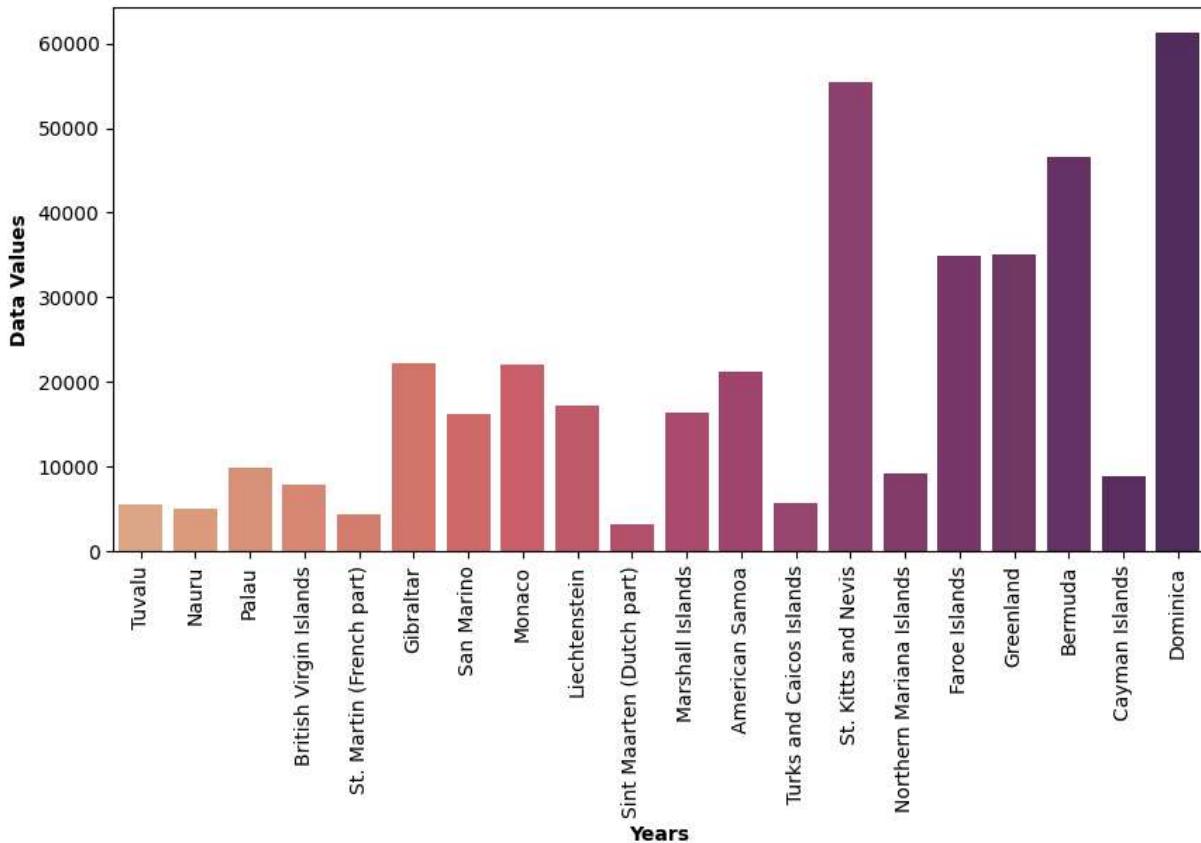
In [32]: country_by_2023_t=country_by_2023.set_index('Country Name').T

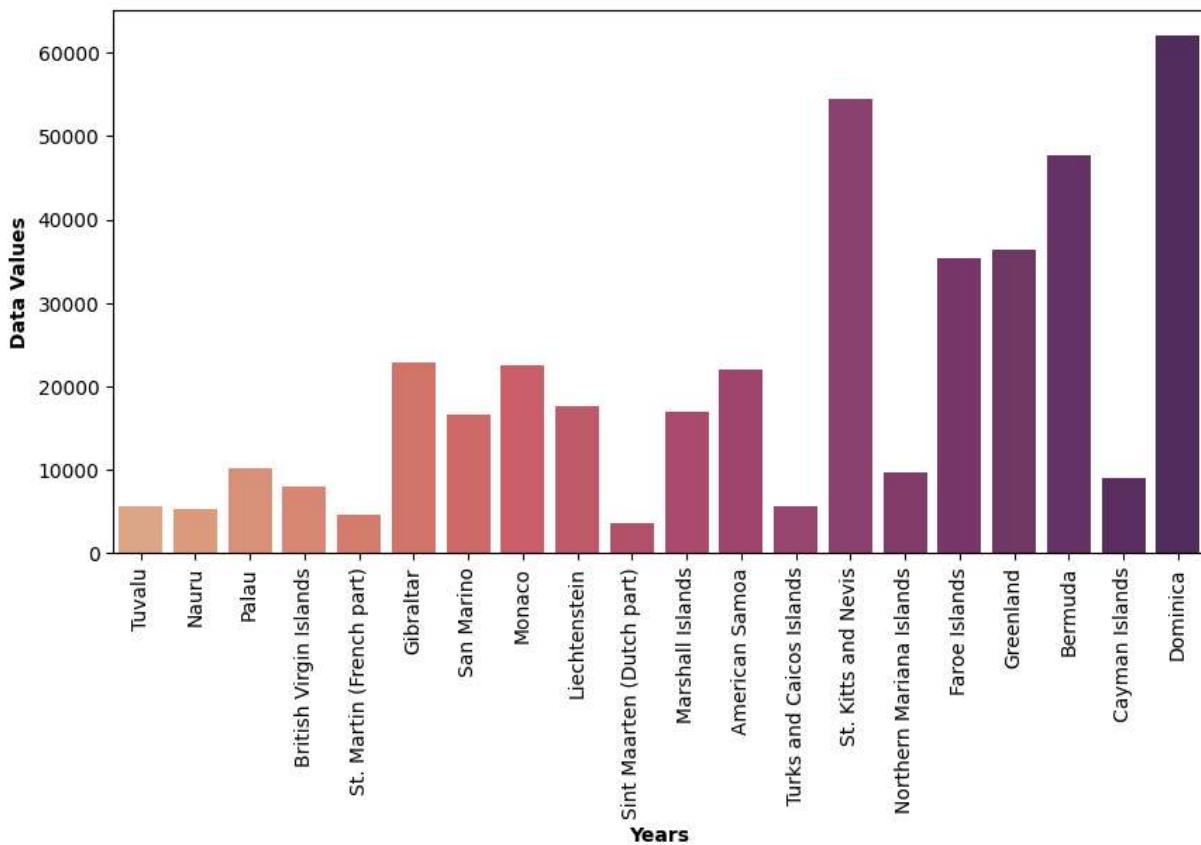
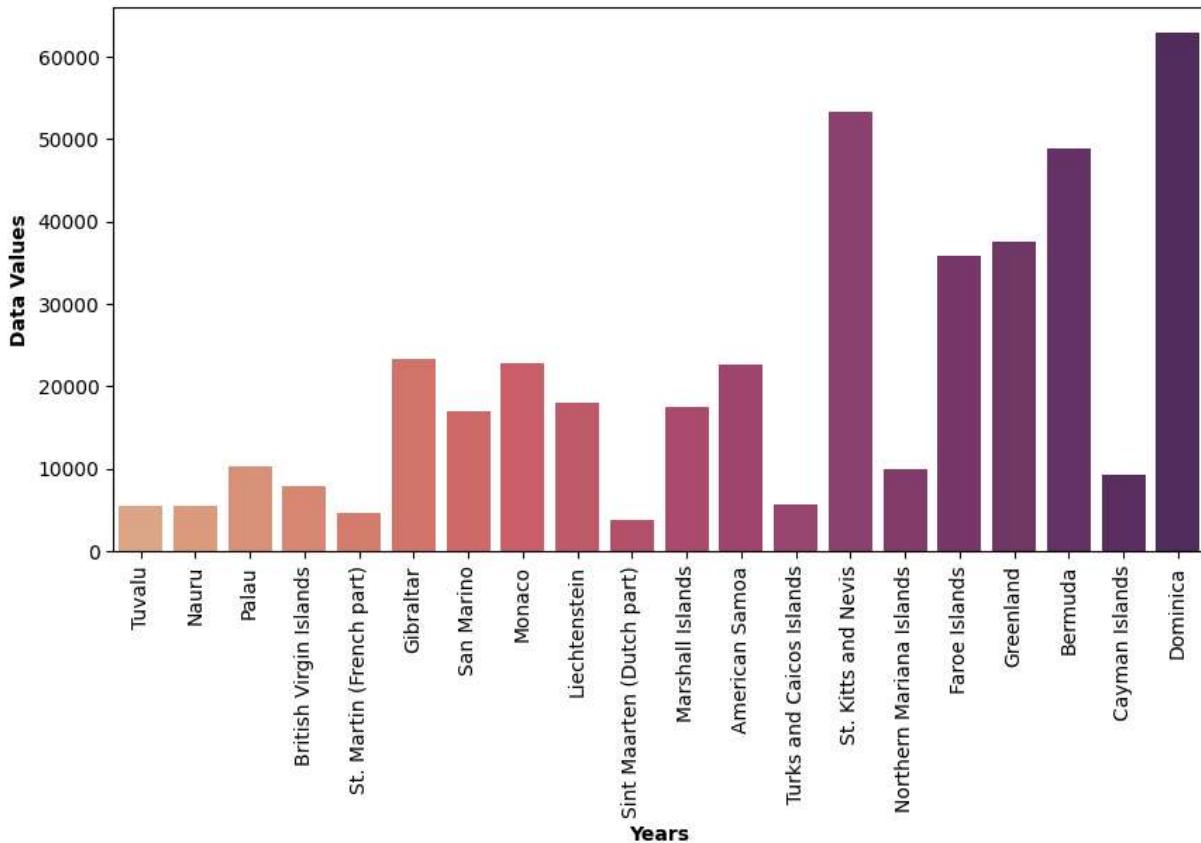
```
color_palette = sns.color_palette("flare",n_colors=len(country_by_2023_t.columns))
for country_name, data_values in country_by_2023_t.iterrows():
    fig=plt.figure(figsize=(10,5))

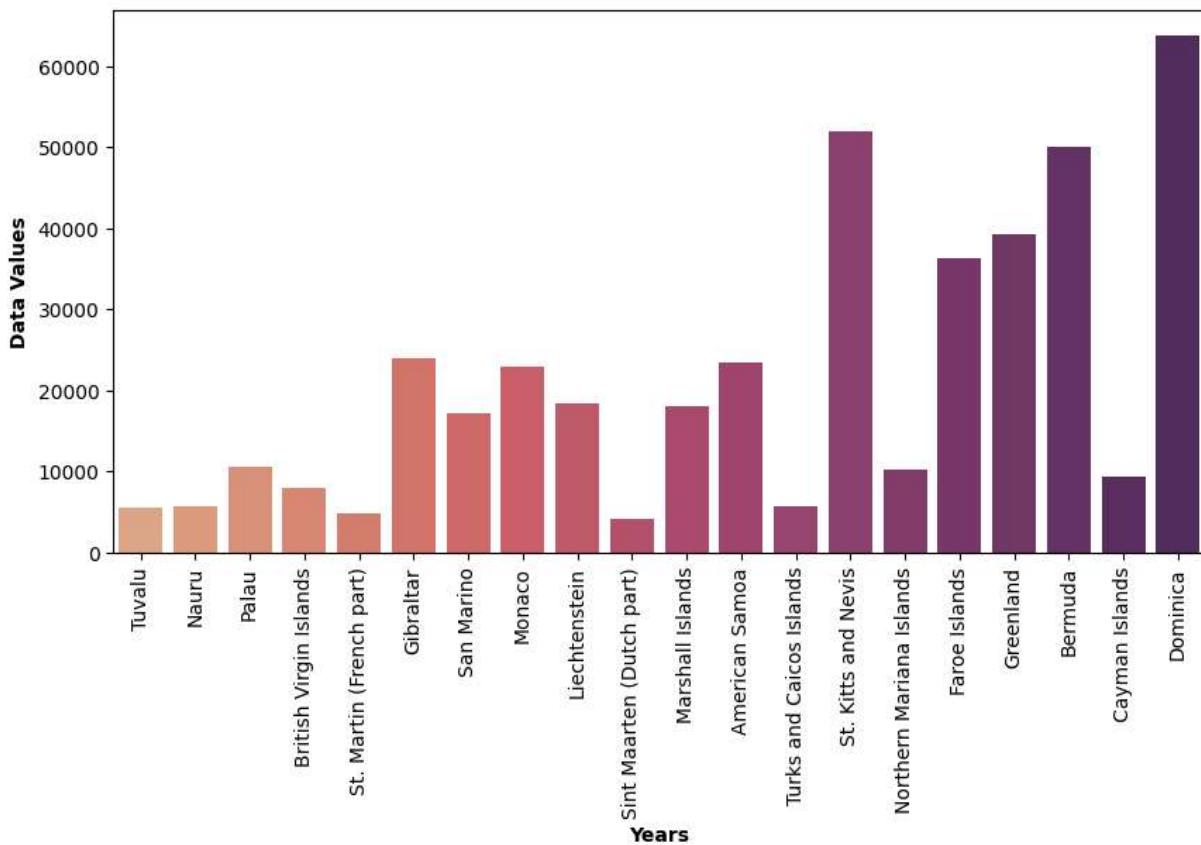
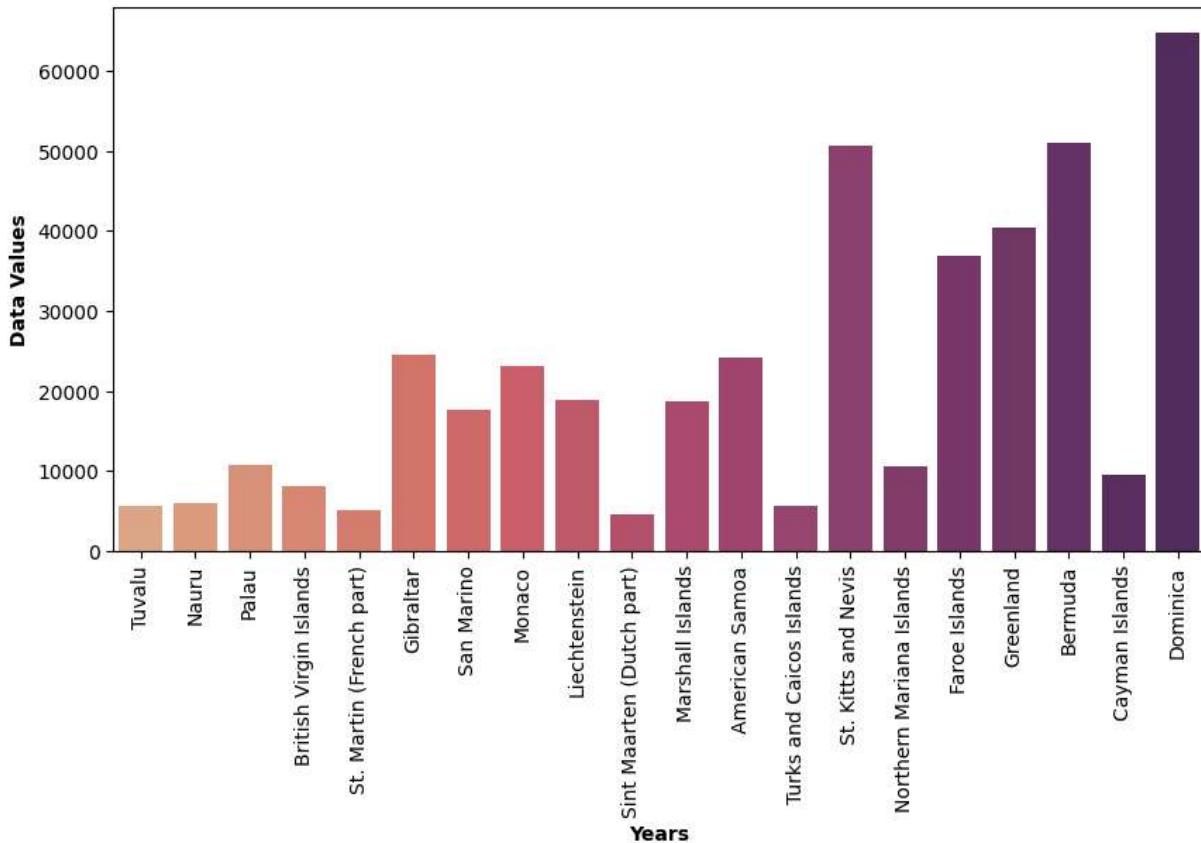
    sns.barplot(x=data_values.index, y=data_values.values, hue=data_values.index, pal
    plt.xlabel("Years",fontweight="bold")
    plt.ylabel("Data Values",fontweight="bold")

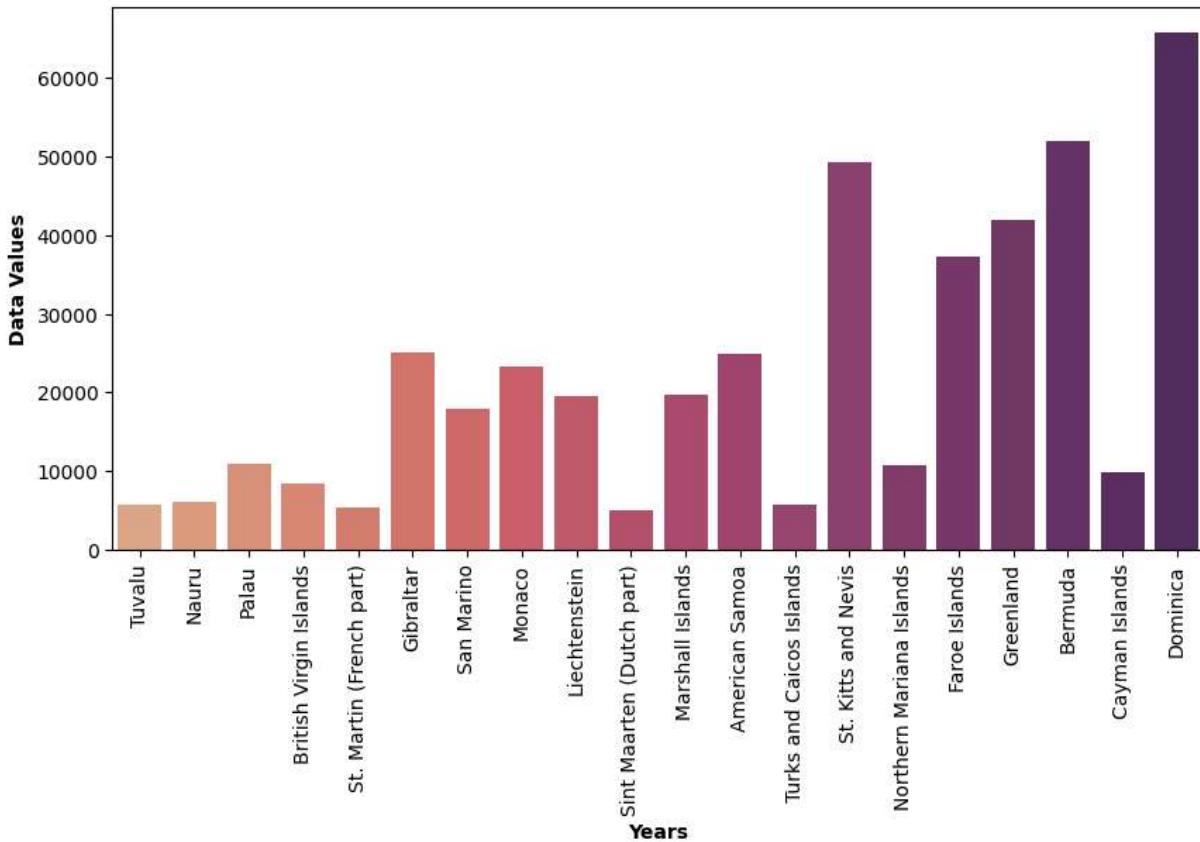
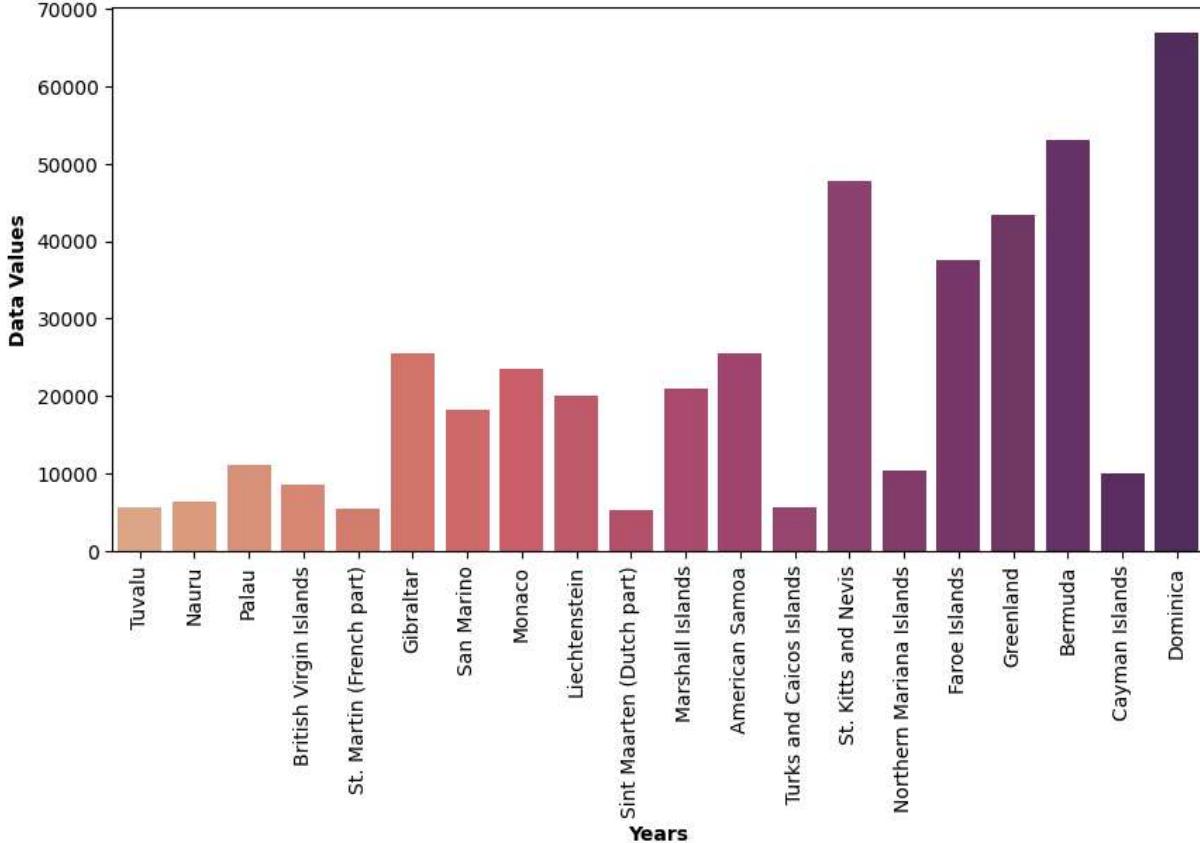
    plt.title(f"{country_name} - Data values from 2023 to 1960",fontweight="bold",)
    plt.xticks(rotation=90)
    plt.show()
```

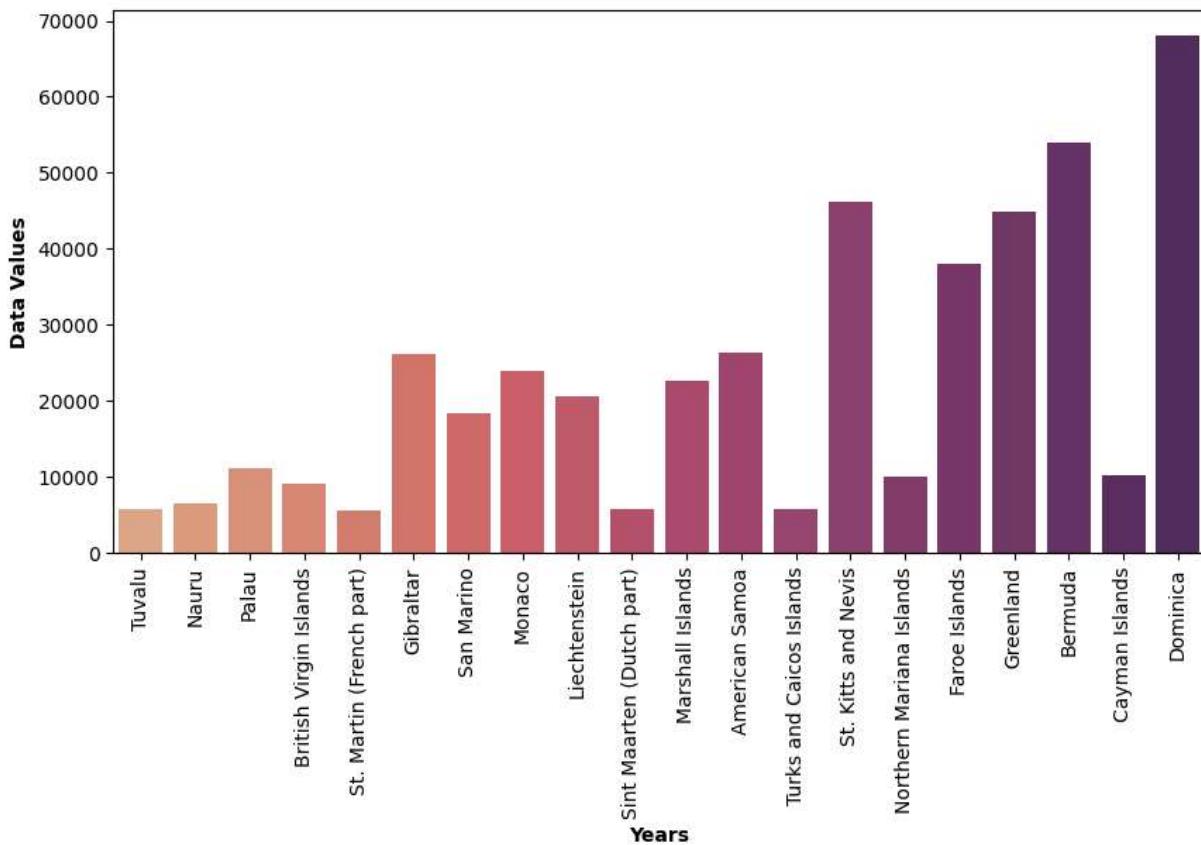
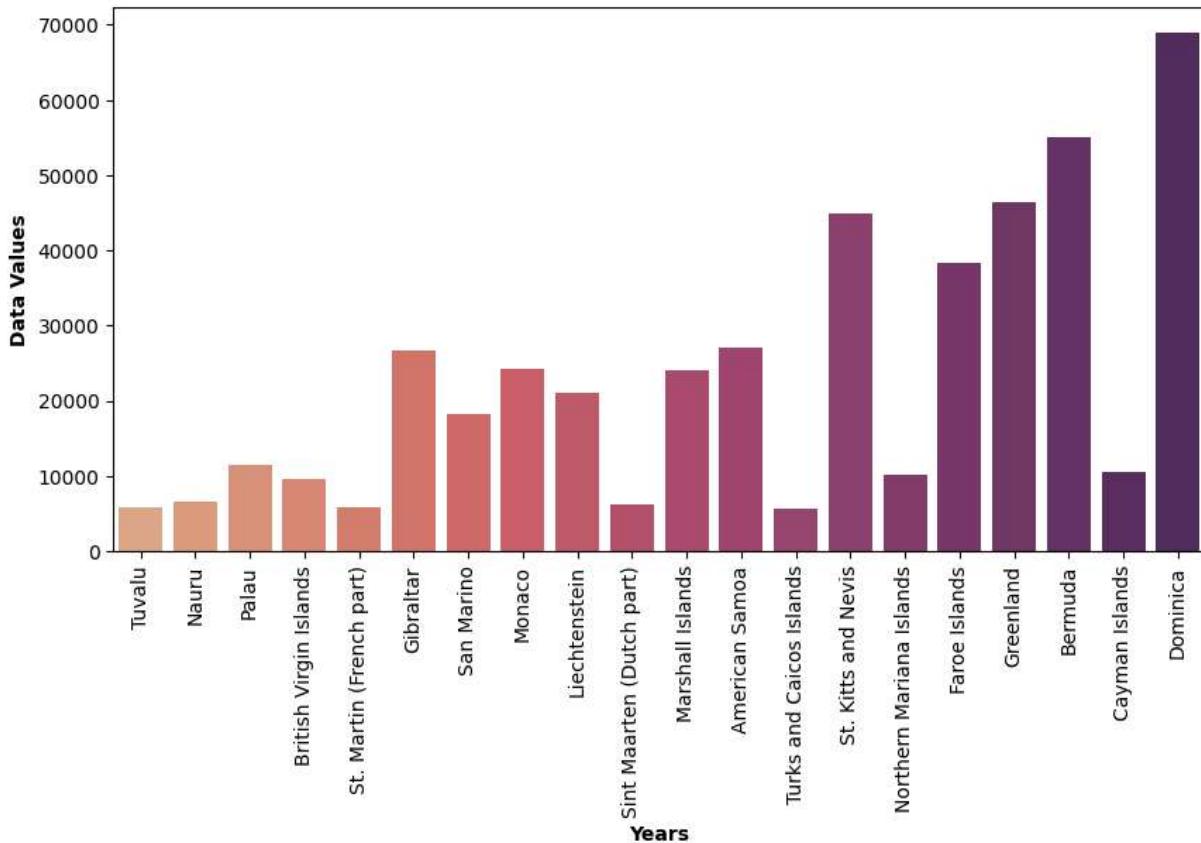


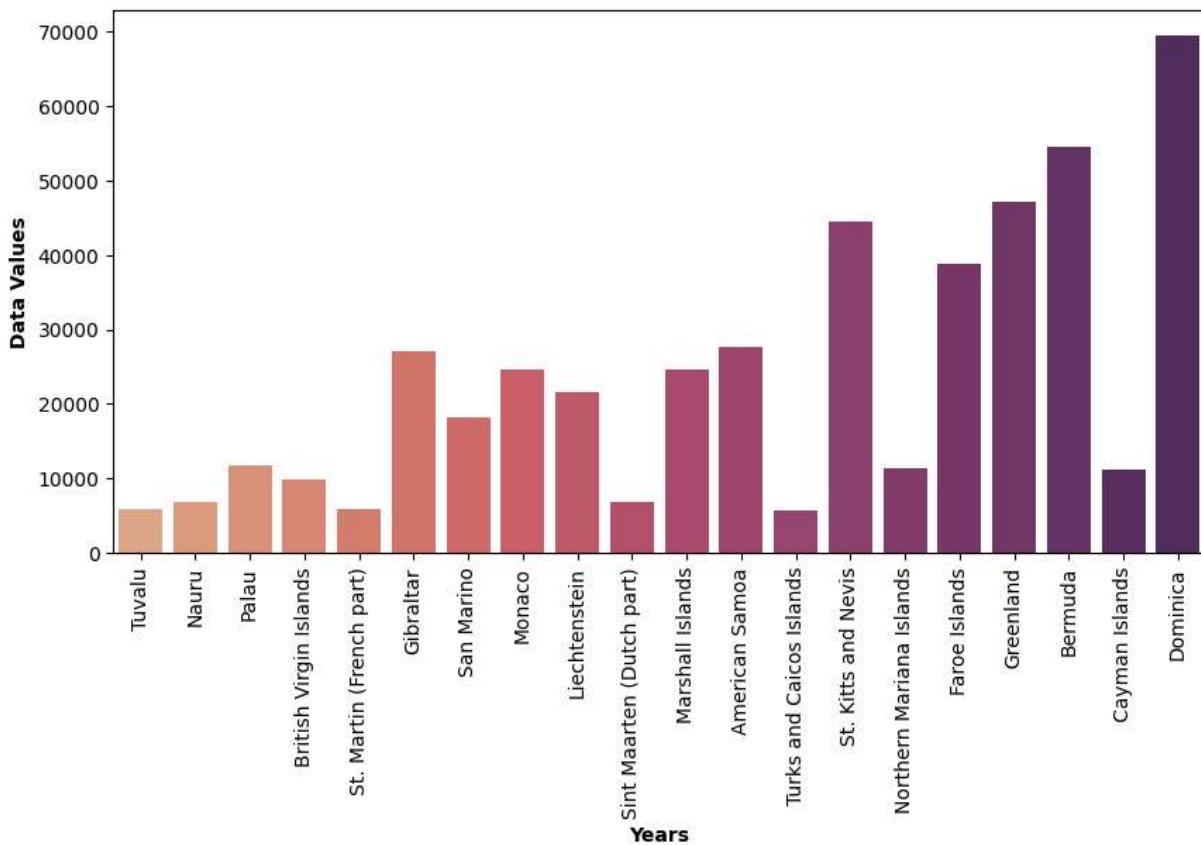
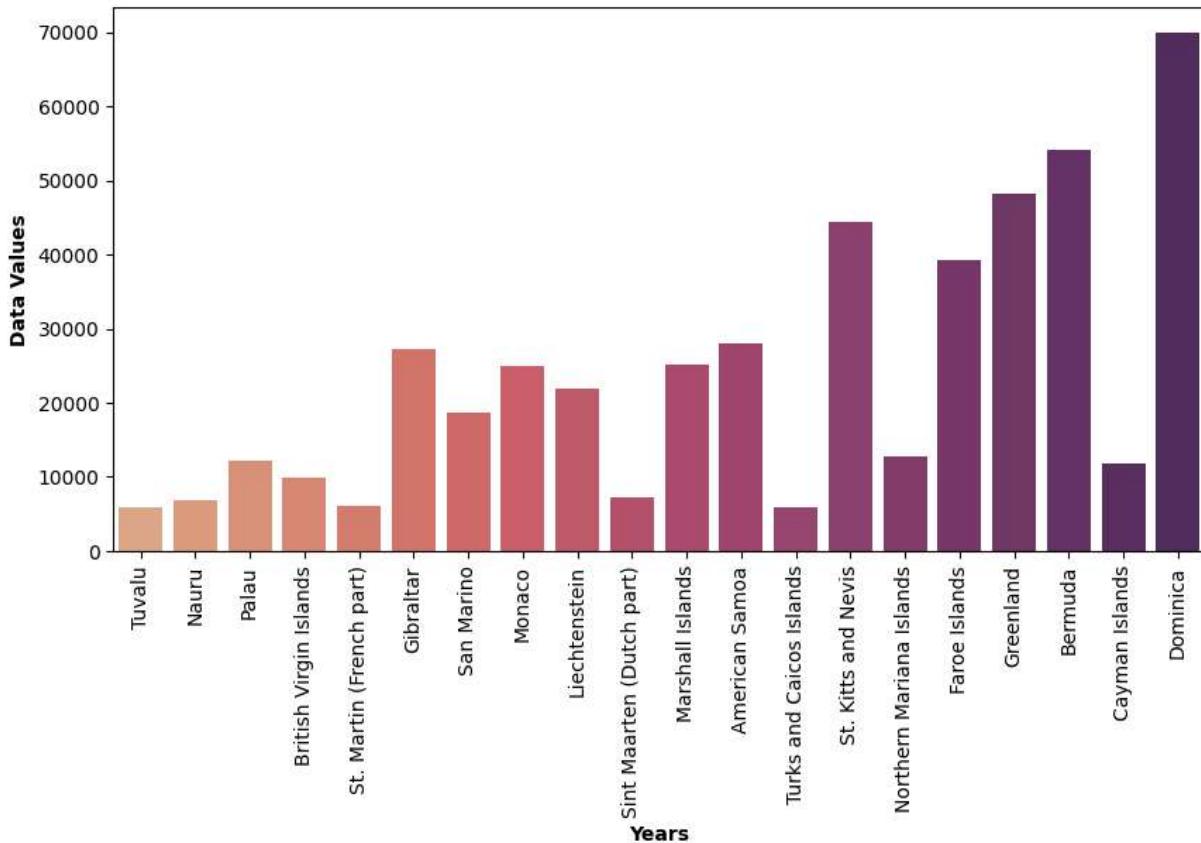
1961 - Data values from 2023 to 1960**1962 - Data values from 2023 to 1960**

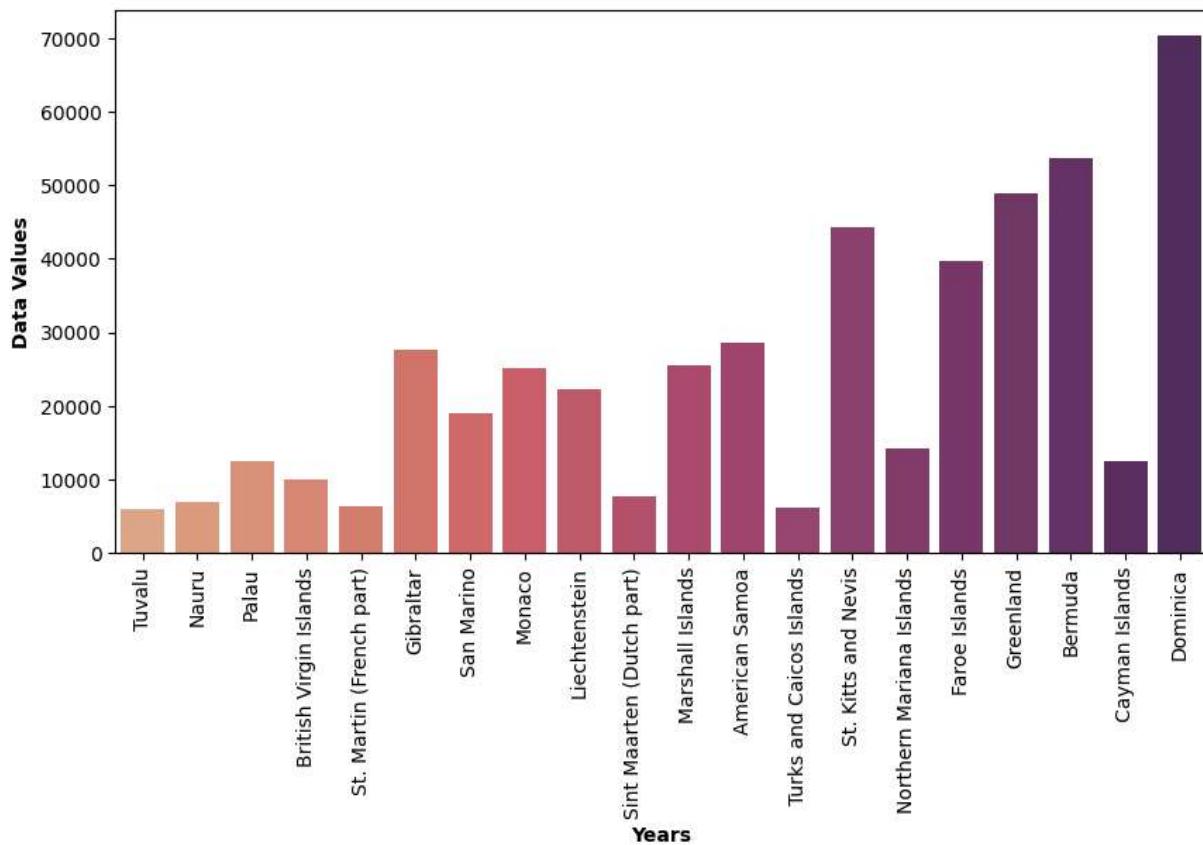
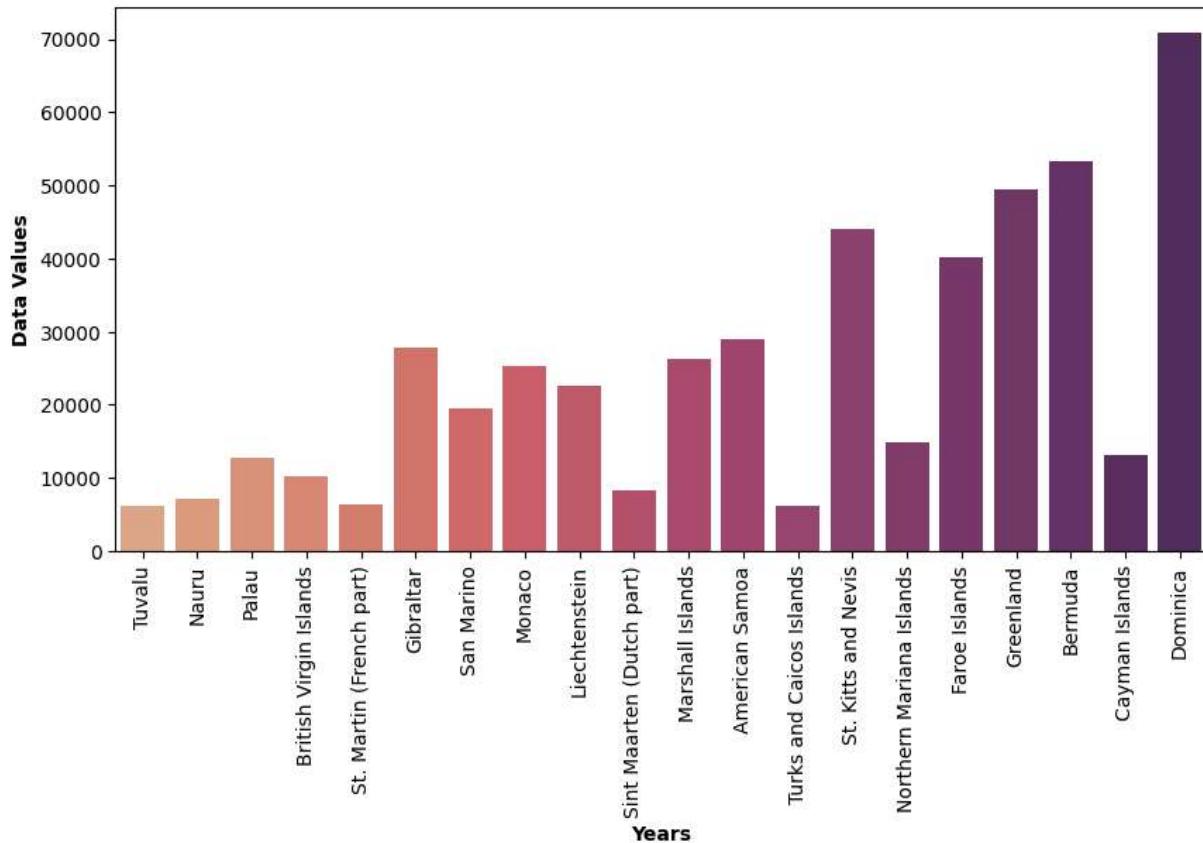
1963 - Data values from 2023 to 1960**1964 - Data values from 2023 to 1960**

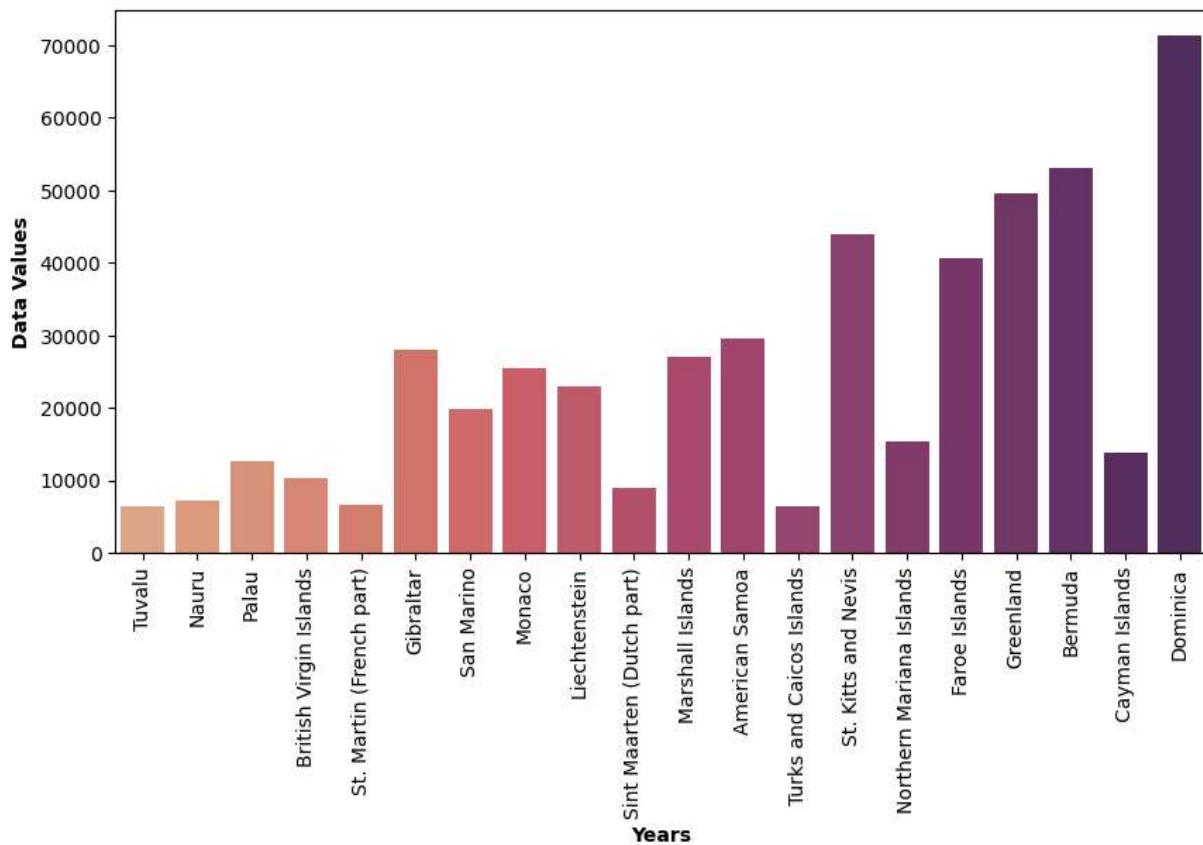
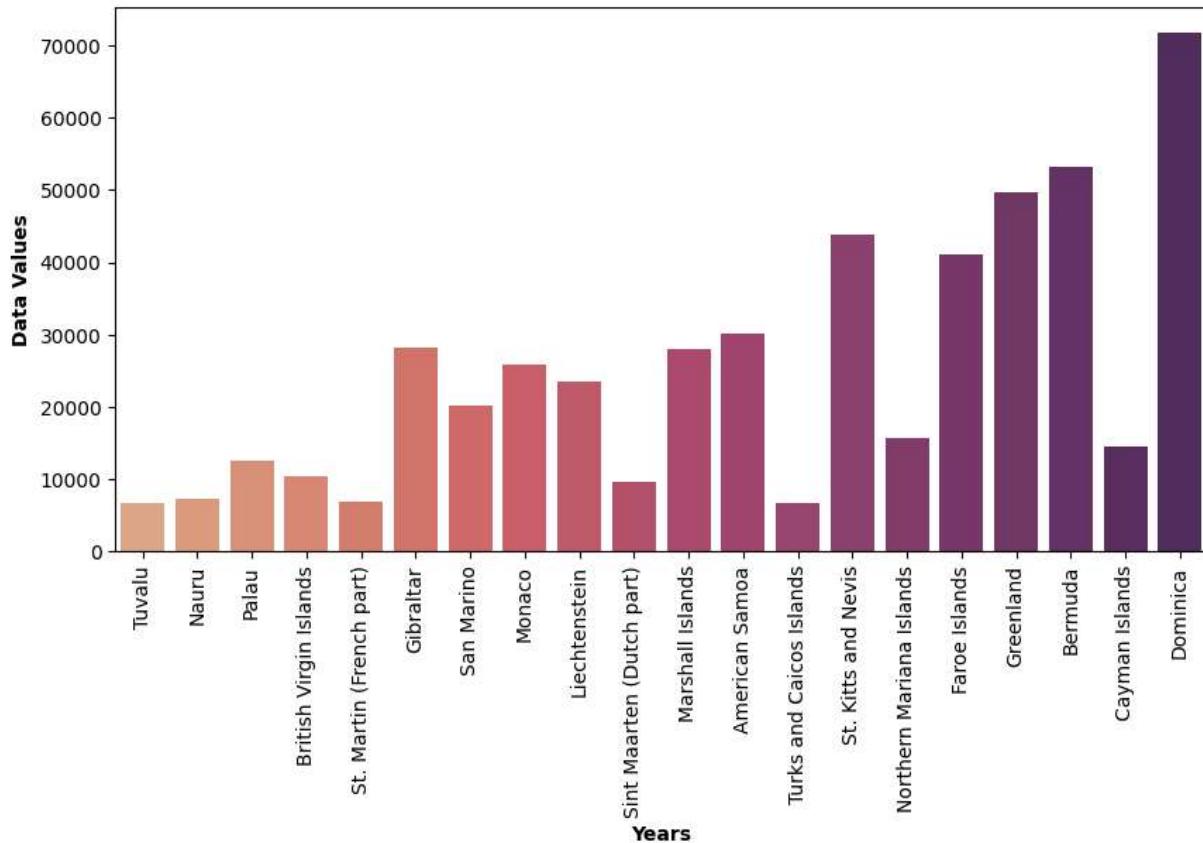
1965 - Data values from 2023 to 1960**1966 - Data values from 2023 to 1960**

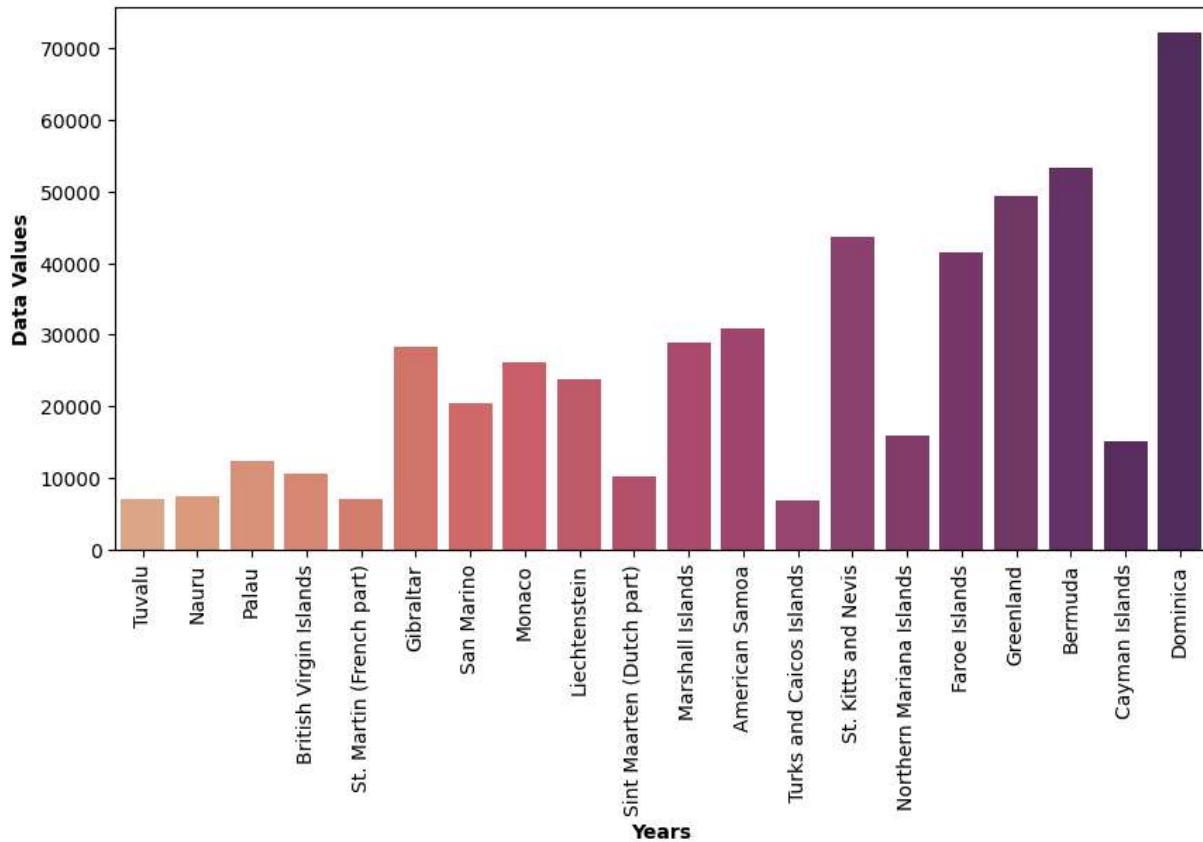
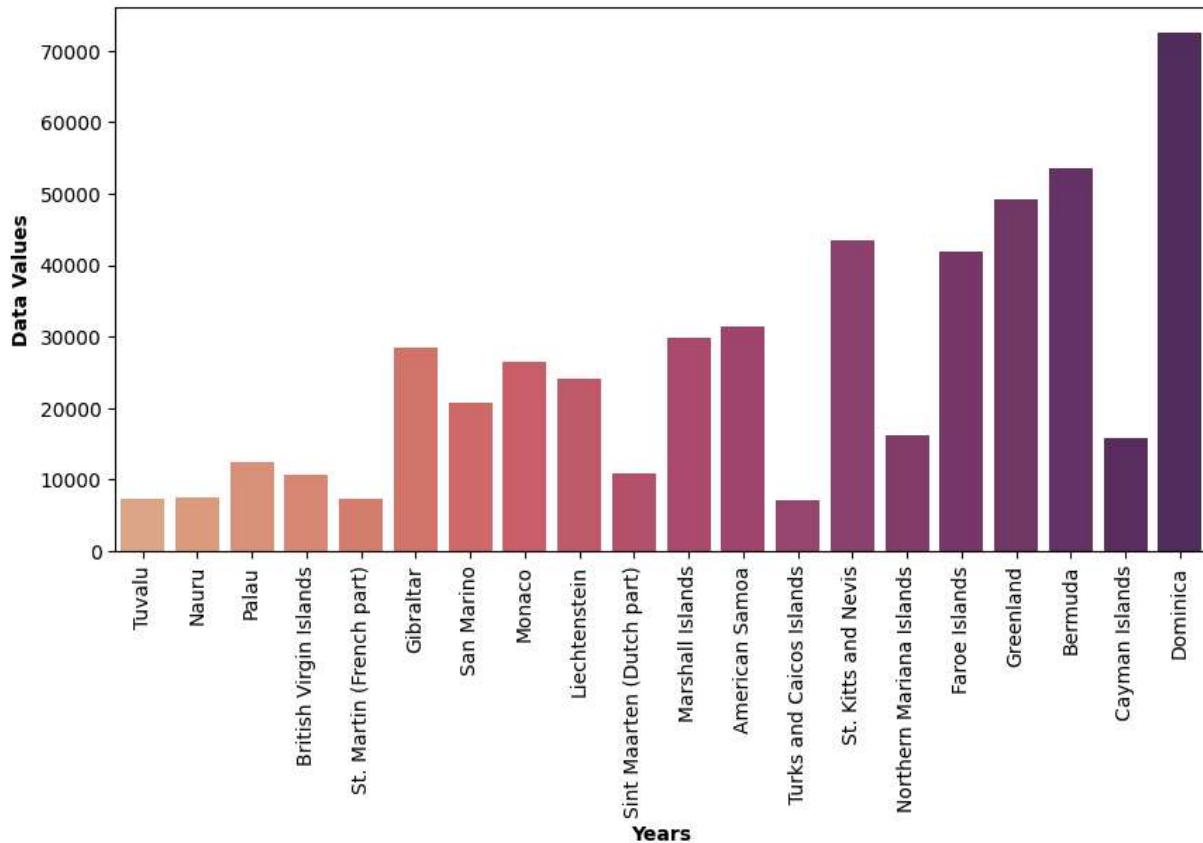
1967 - Data values from 2023 to 1960**1968 - Data values from 2023 to 1960**

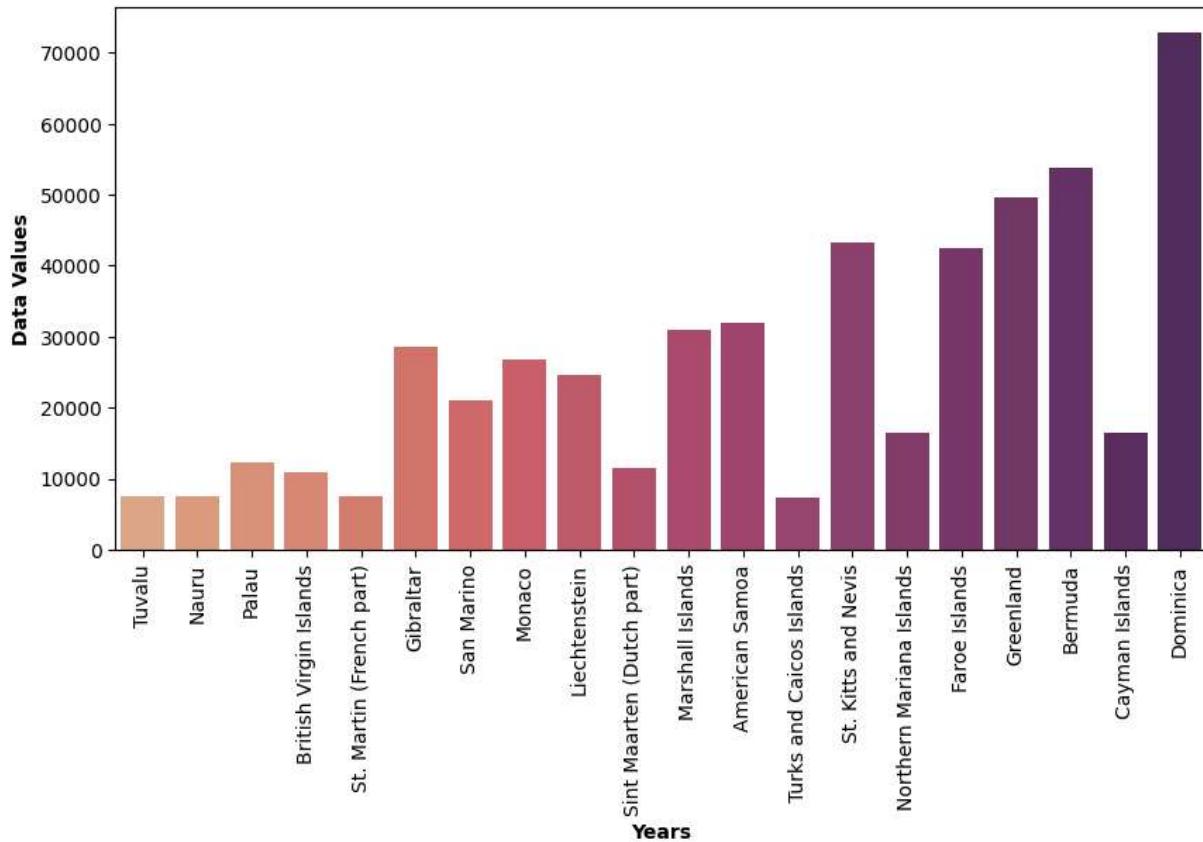
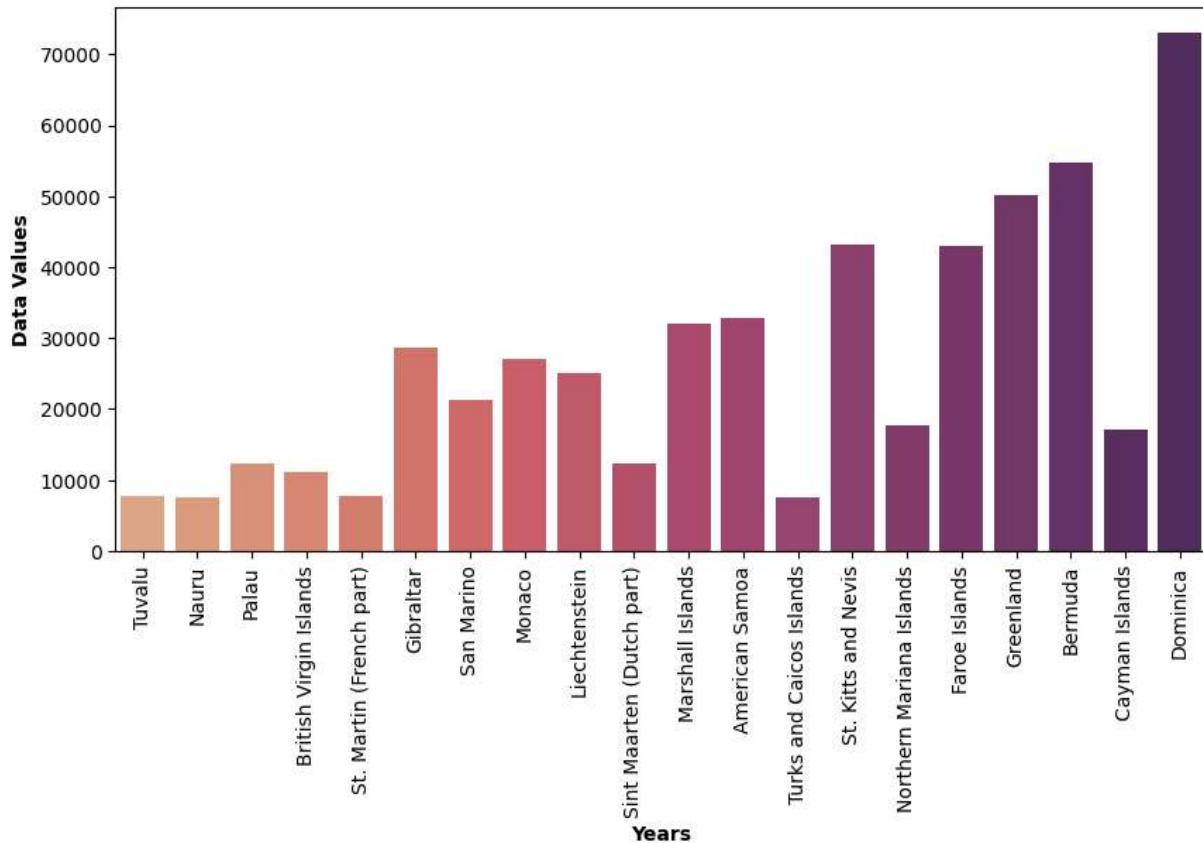
1969 - Data values from 2023 to 1960**1970 - Data values from 2023 to 1960**

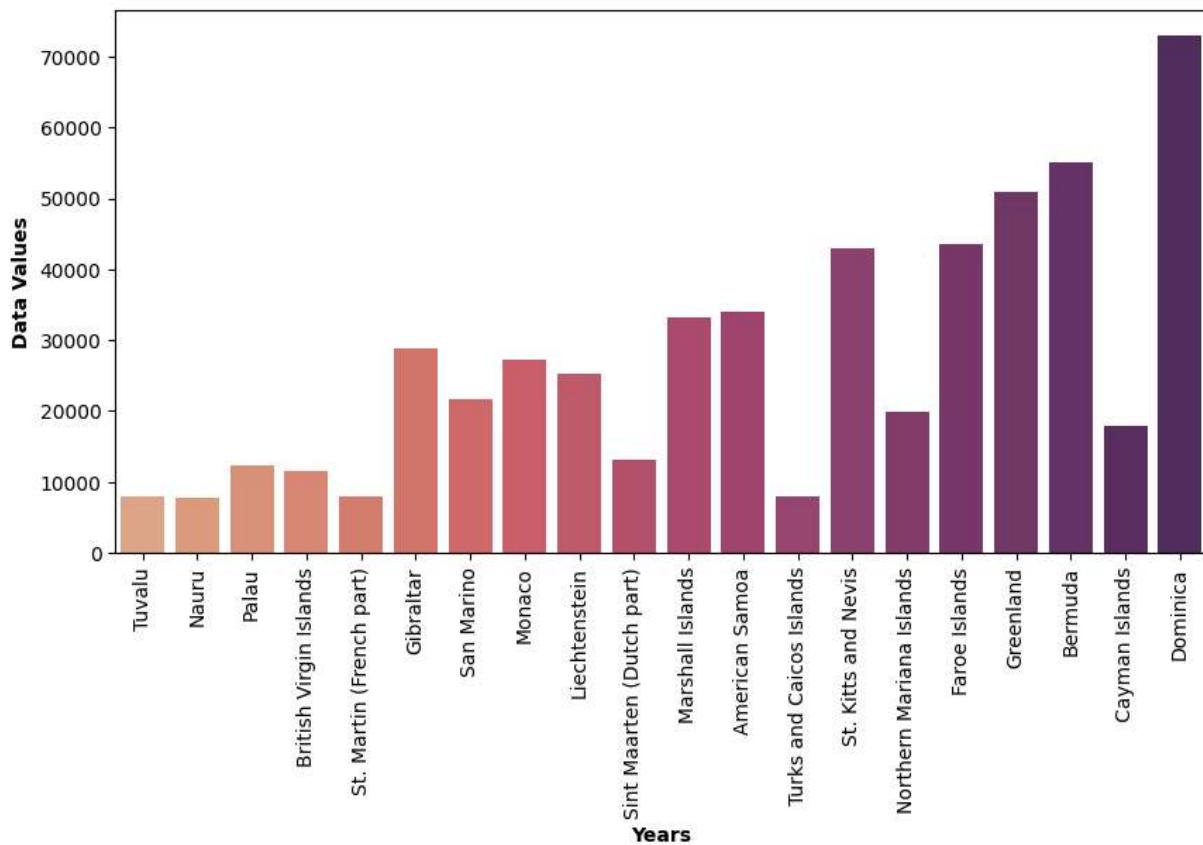
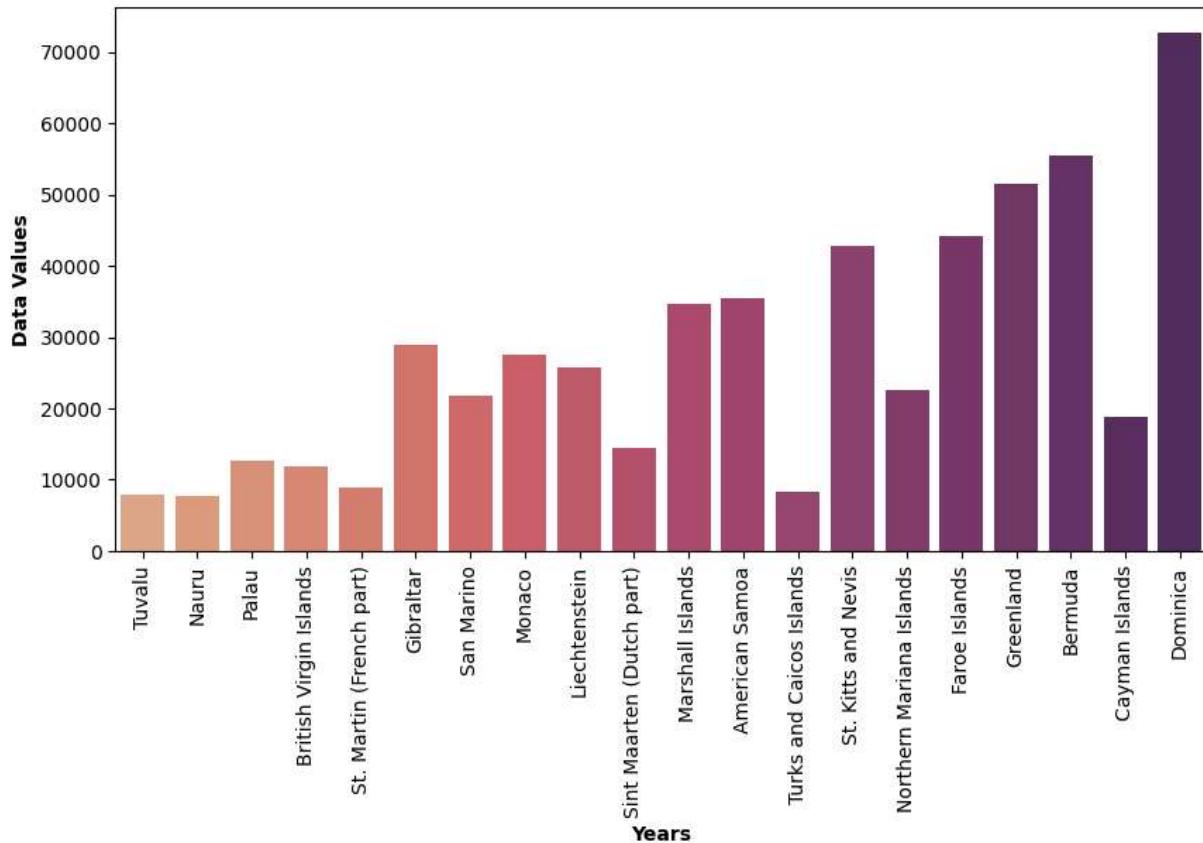
1971 - Data values from 2023 to 1960**1972 - Data values from 2023 to 1960**

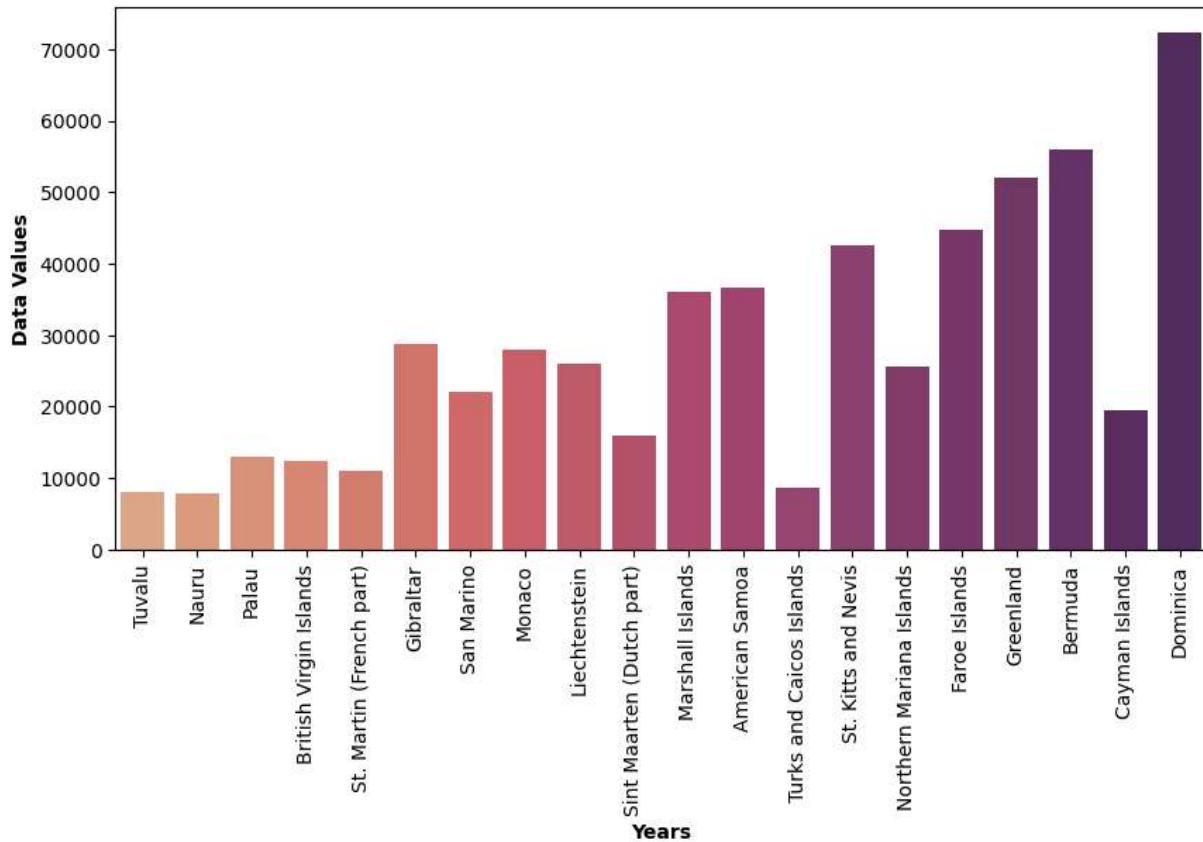
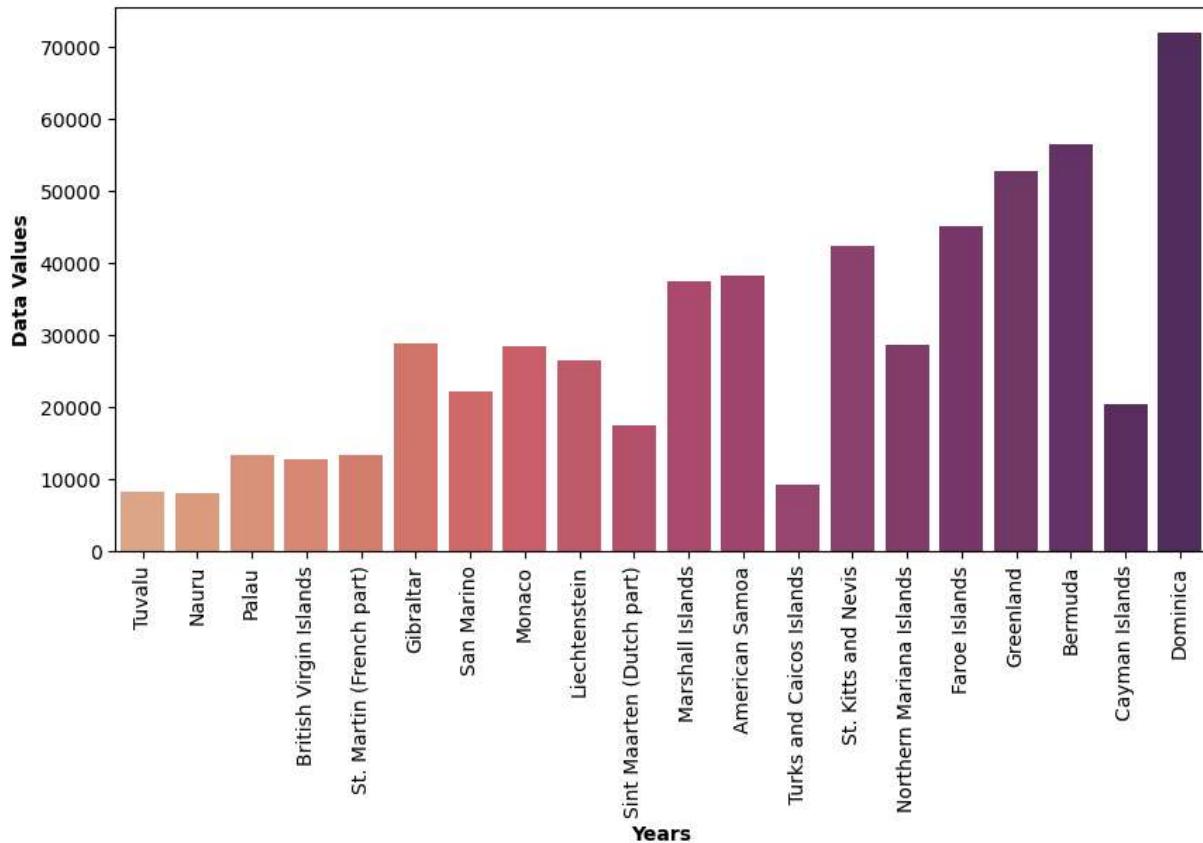
1973 - Data values from 2023 to 1960**1974 - Data values from 2023 to 1960**

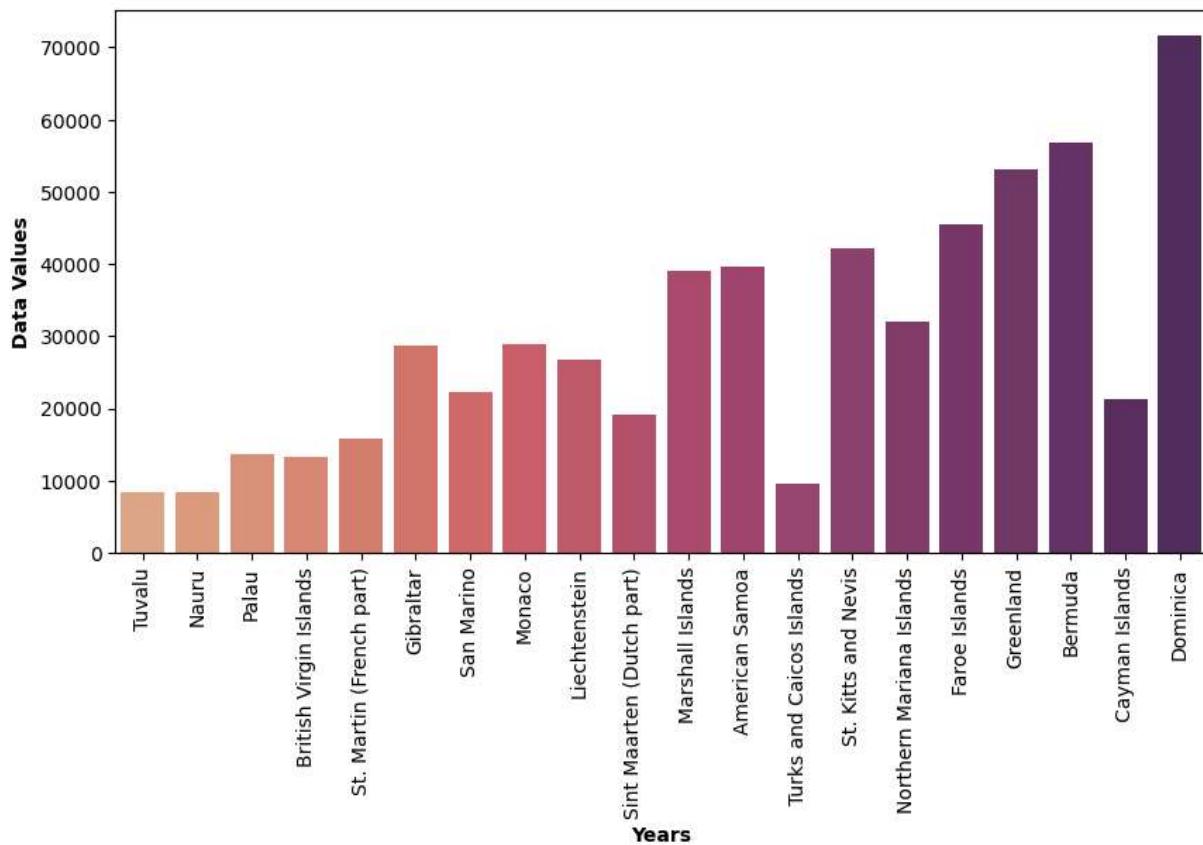
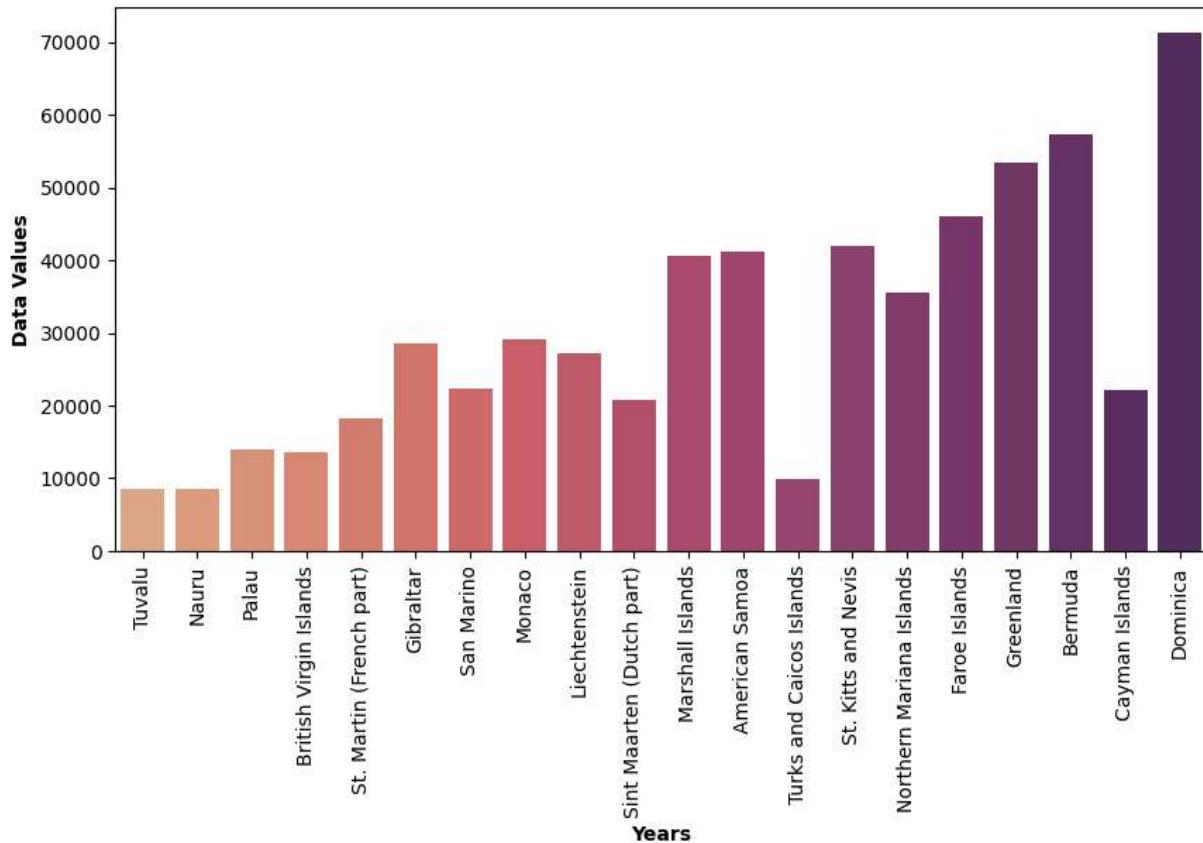
1975 - Data values from 2023 to 1960**1976 - Data values from 2023 to 1960**

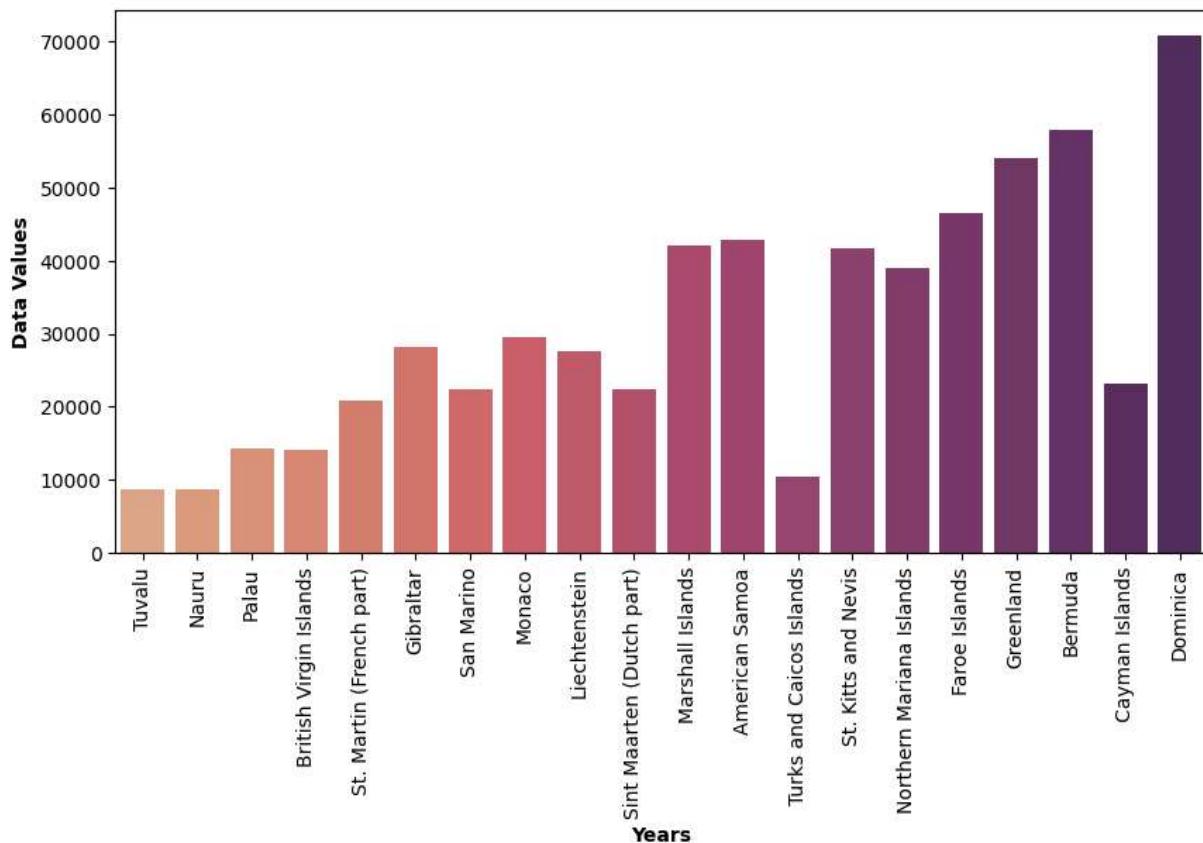
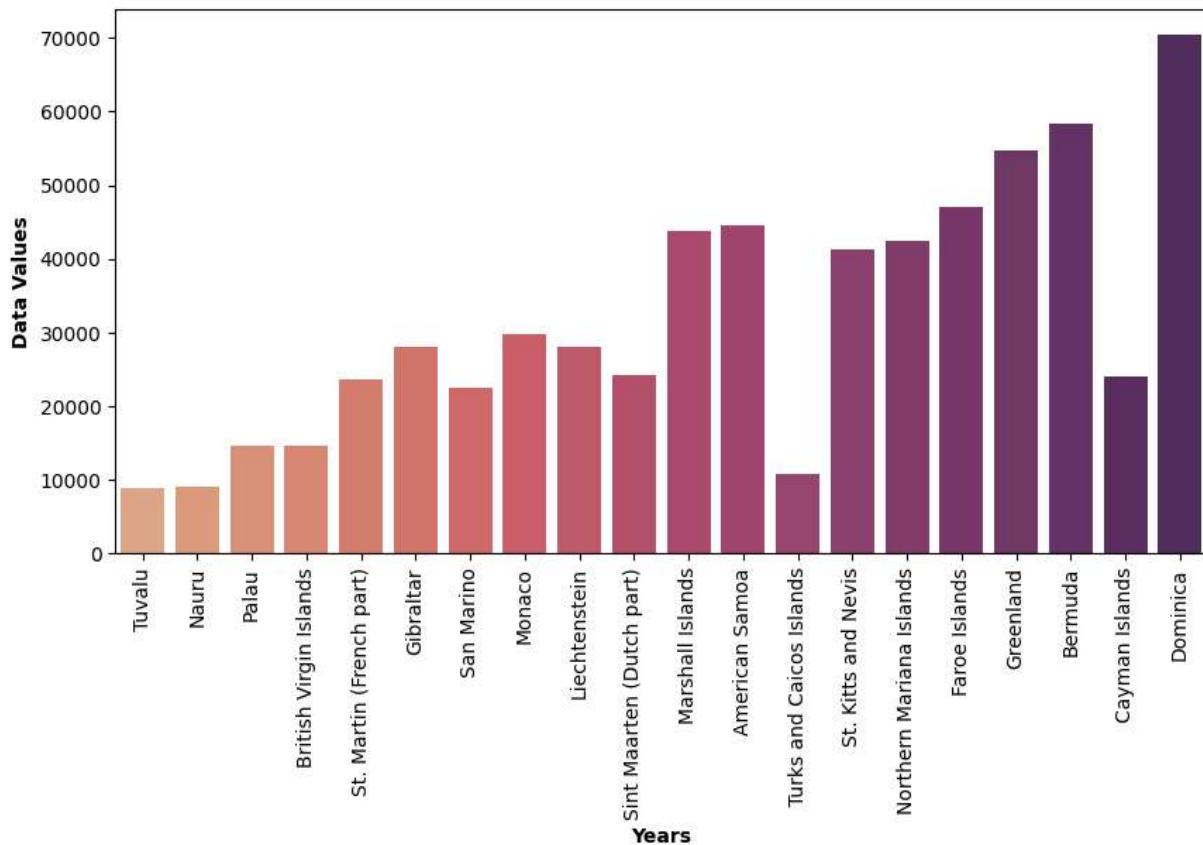
1977 - Data values from 2023 to 1960**1978 - Data values from 2023 to 1960**

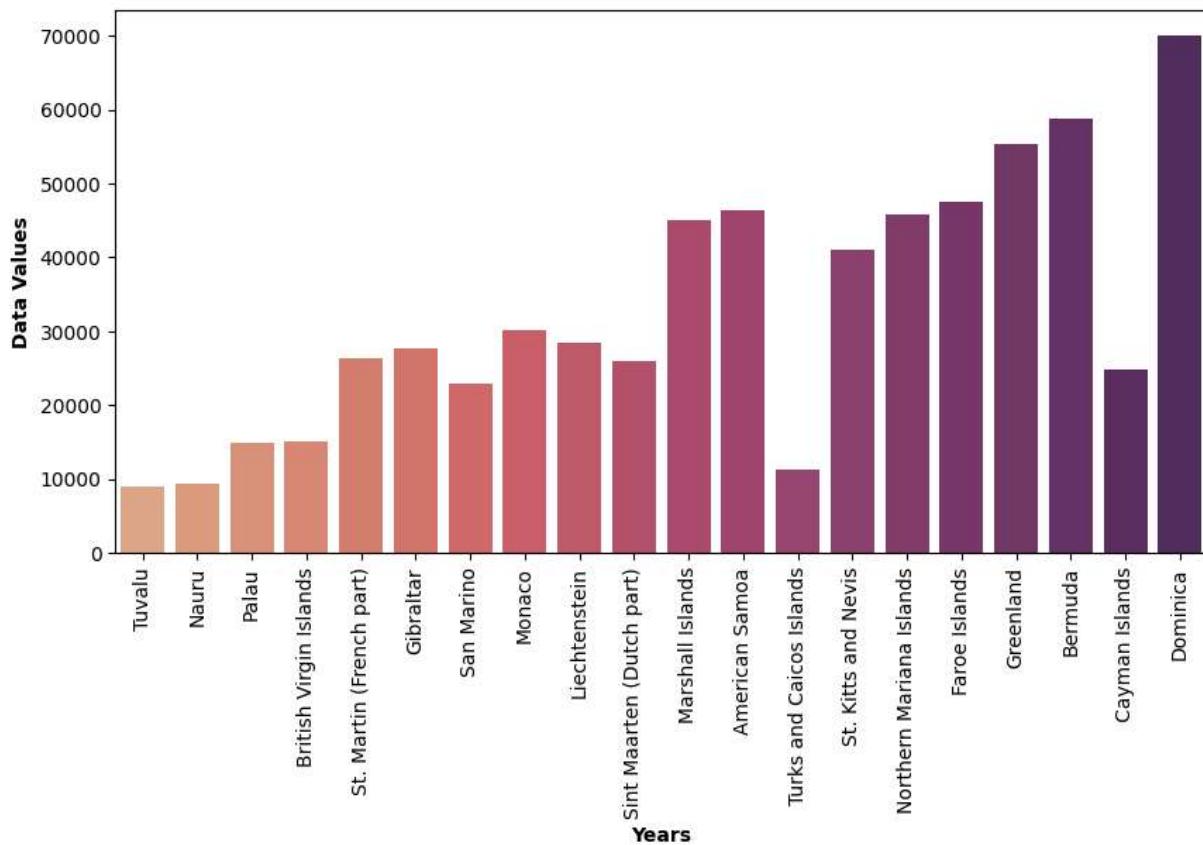
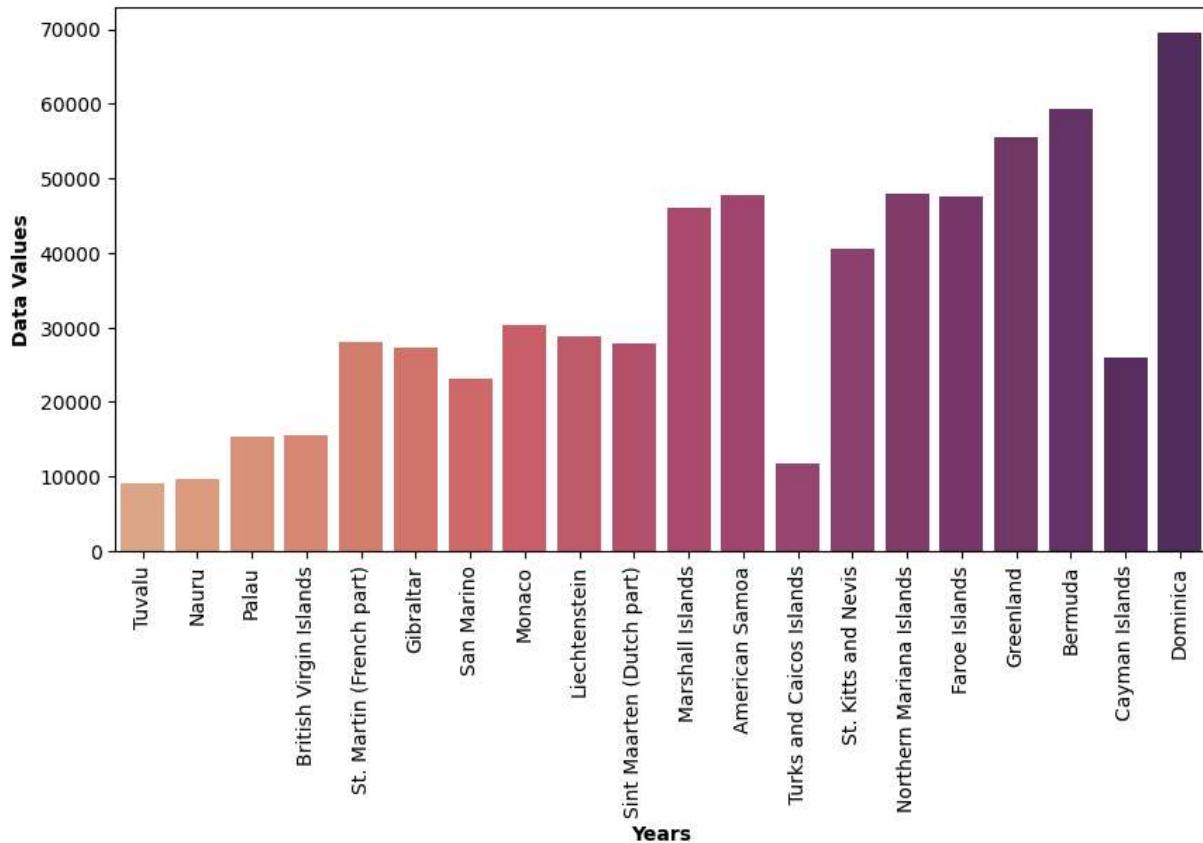
1979 - Data values from 2023 to 1960**1980 - Data values from 2023 to 1960**

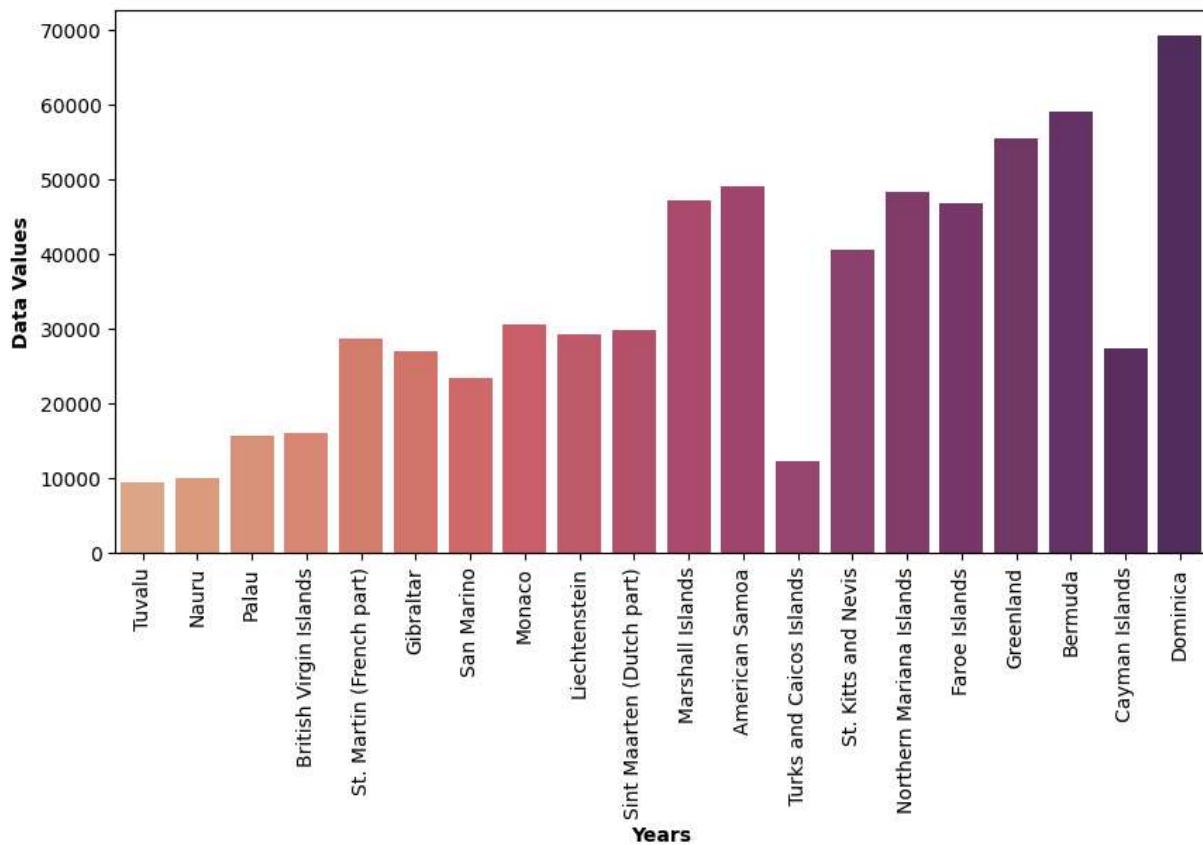
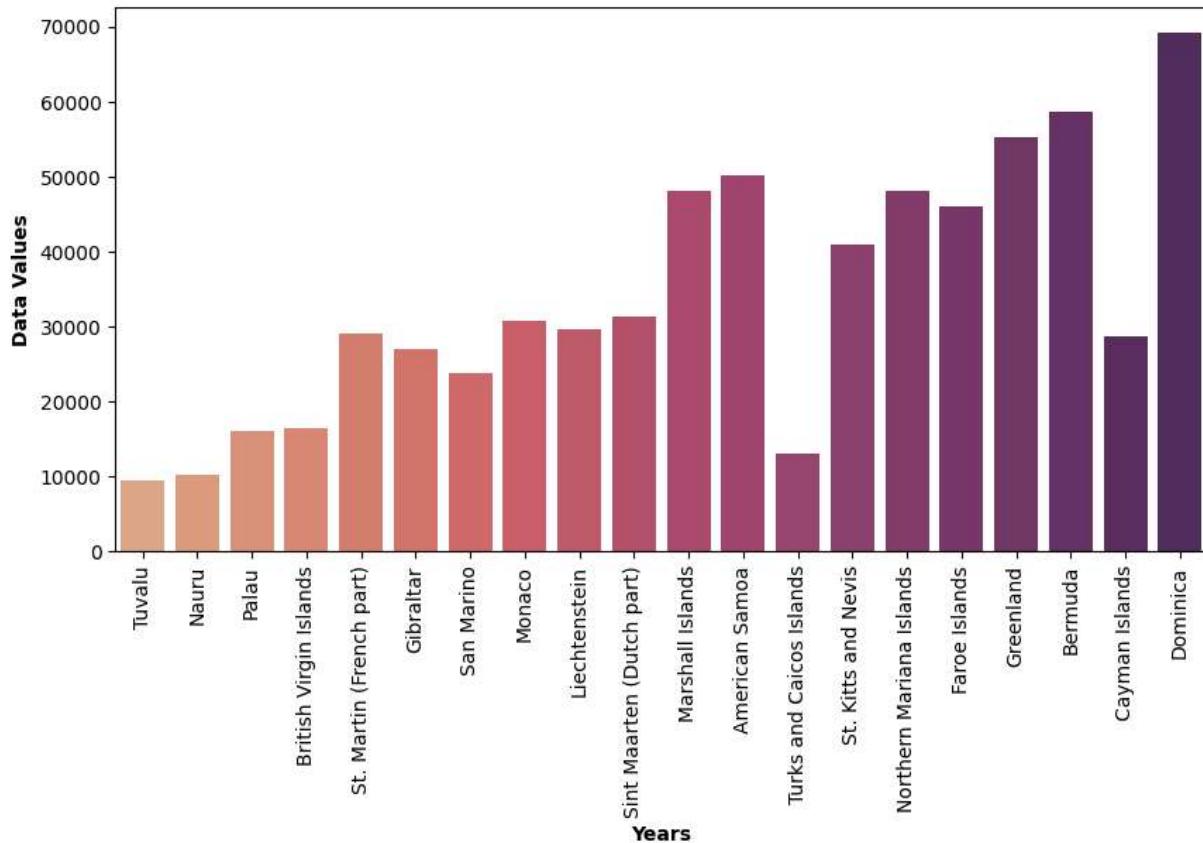
1981 - Data values from 2023 to 1960**1982 - Data values from 2023 to 1960**

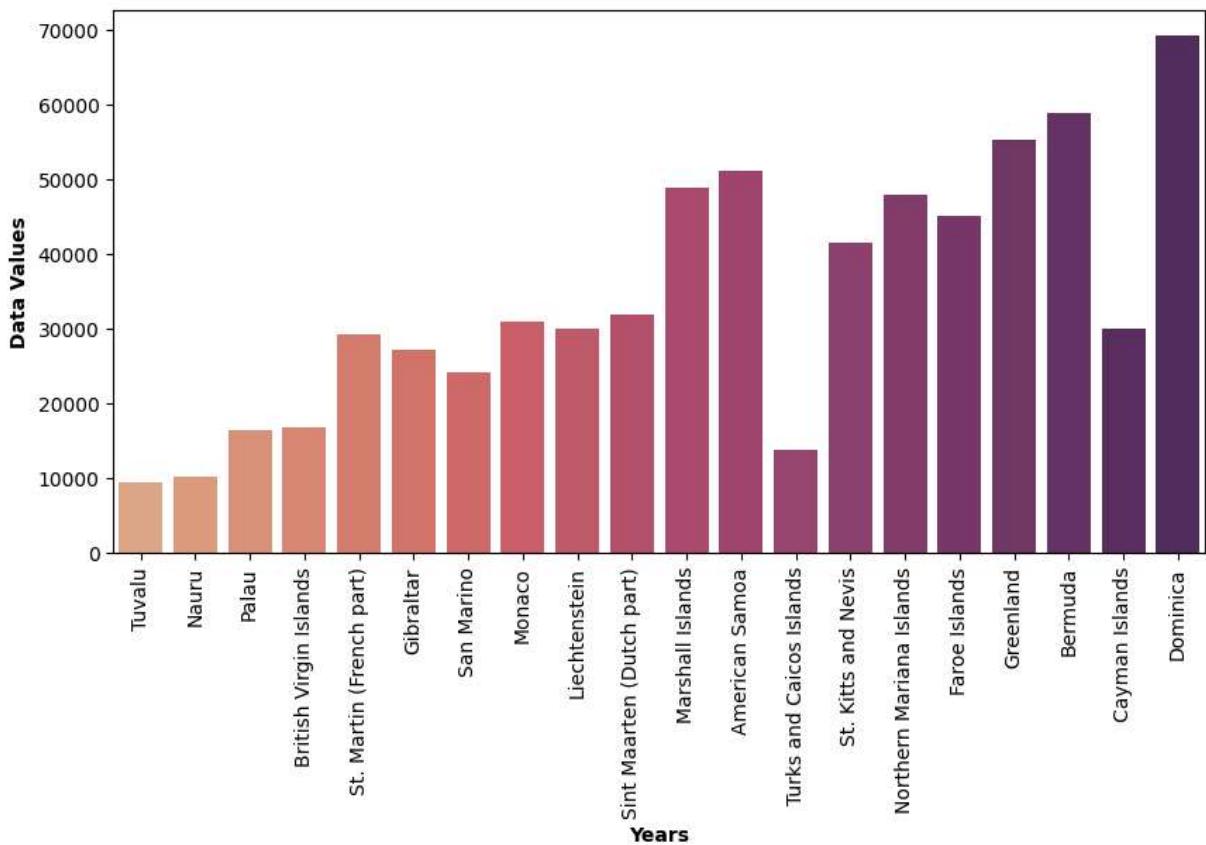
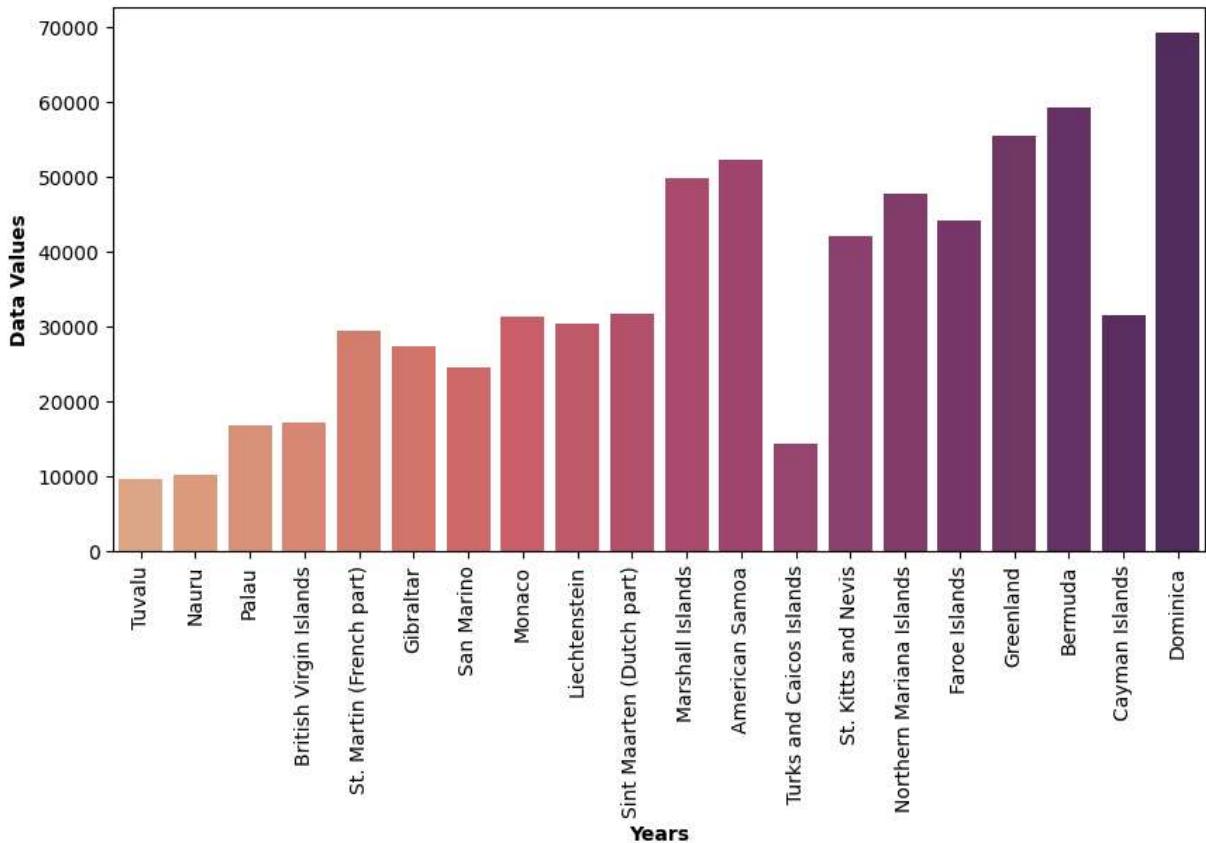
1983 - Data values from 2023 to 1960**1984 - Data values from 2023 to 1960**

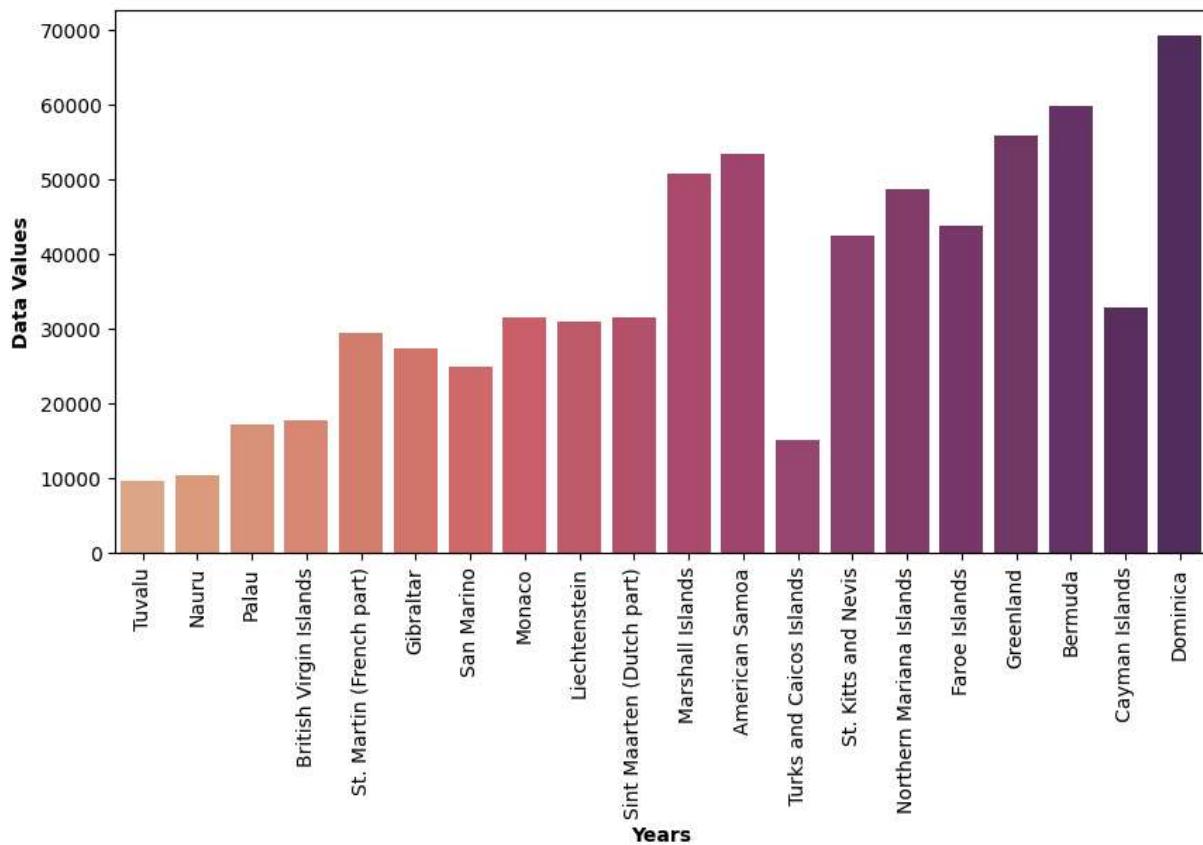
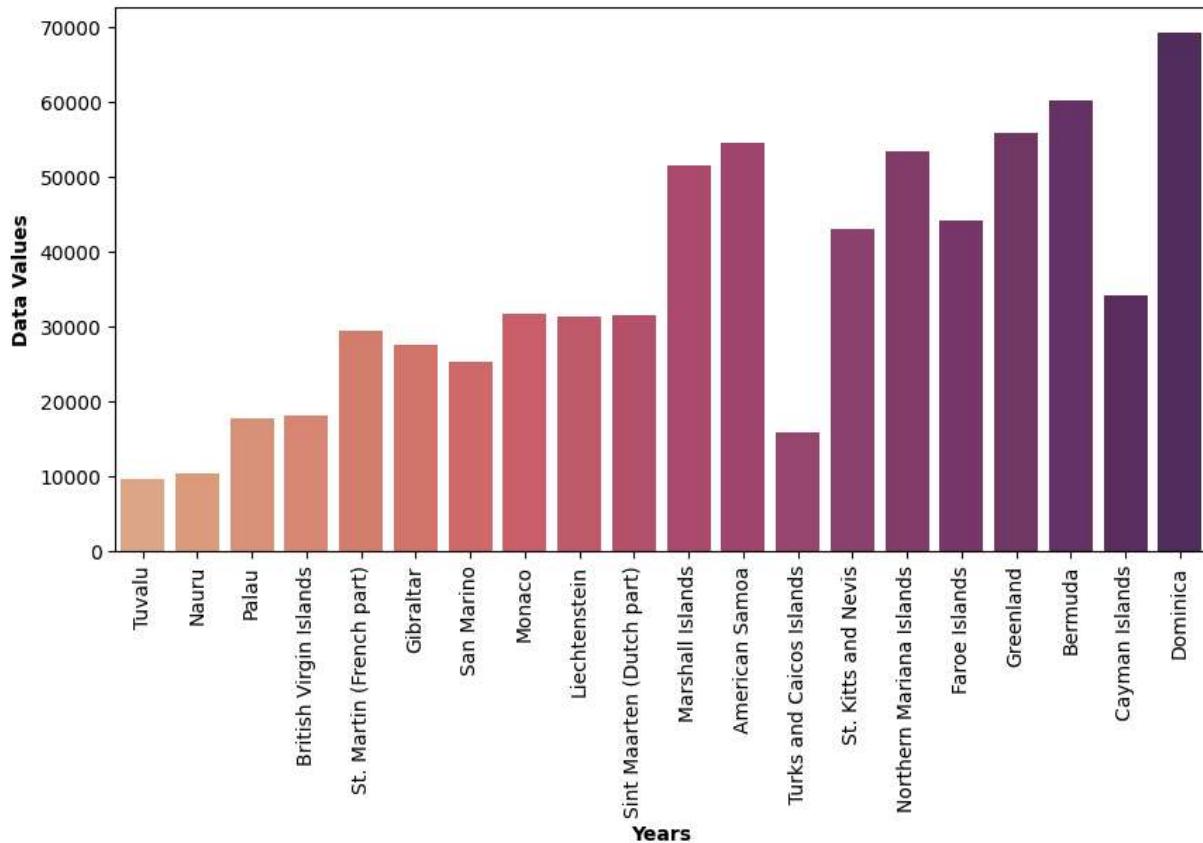
1985 - Data values from 2023 to 1960**1986 - Data values from 2023 to 1960**

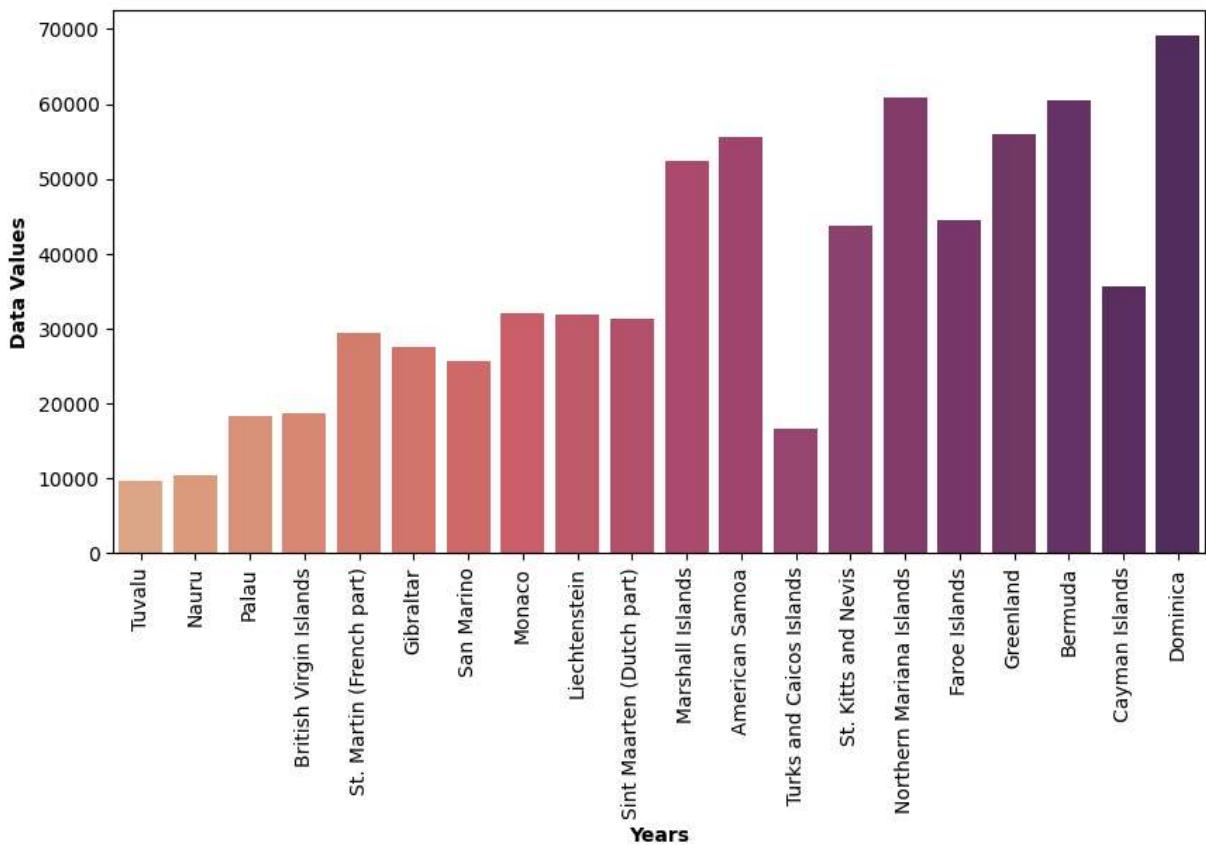
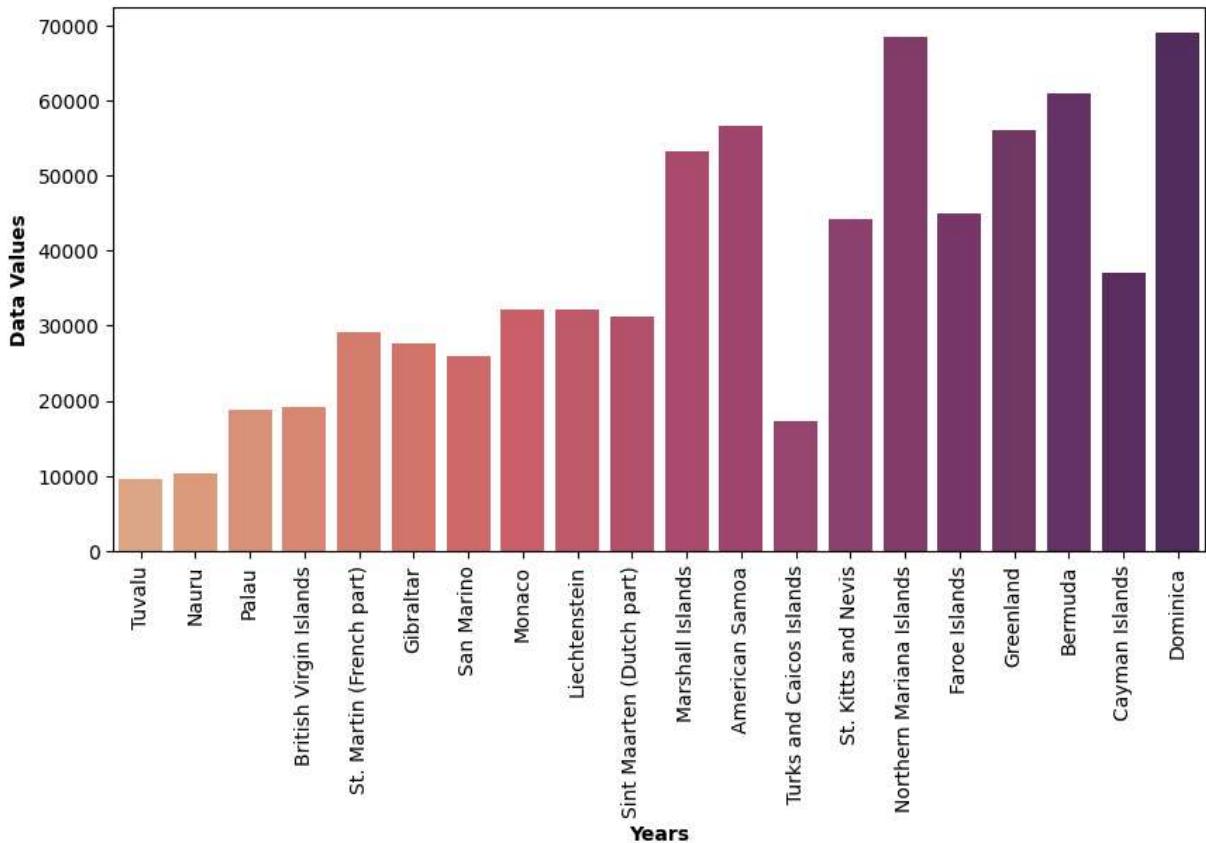
1987 - Data values from 2023 to 1960**1988 - Data values from 2023 to 1960**

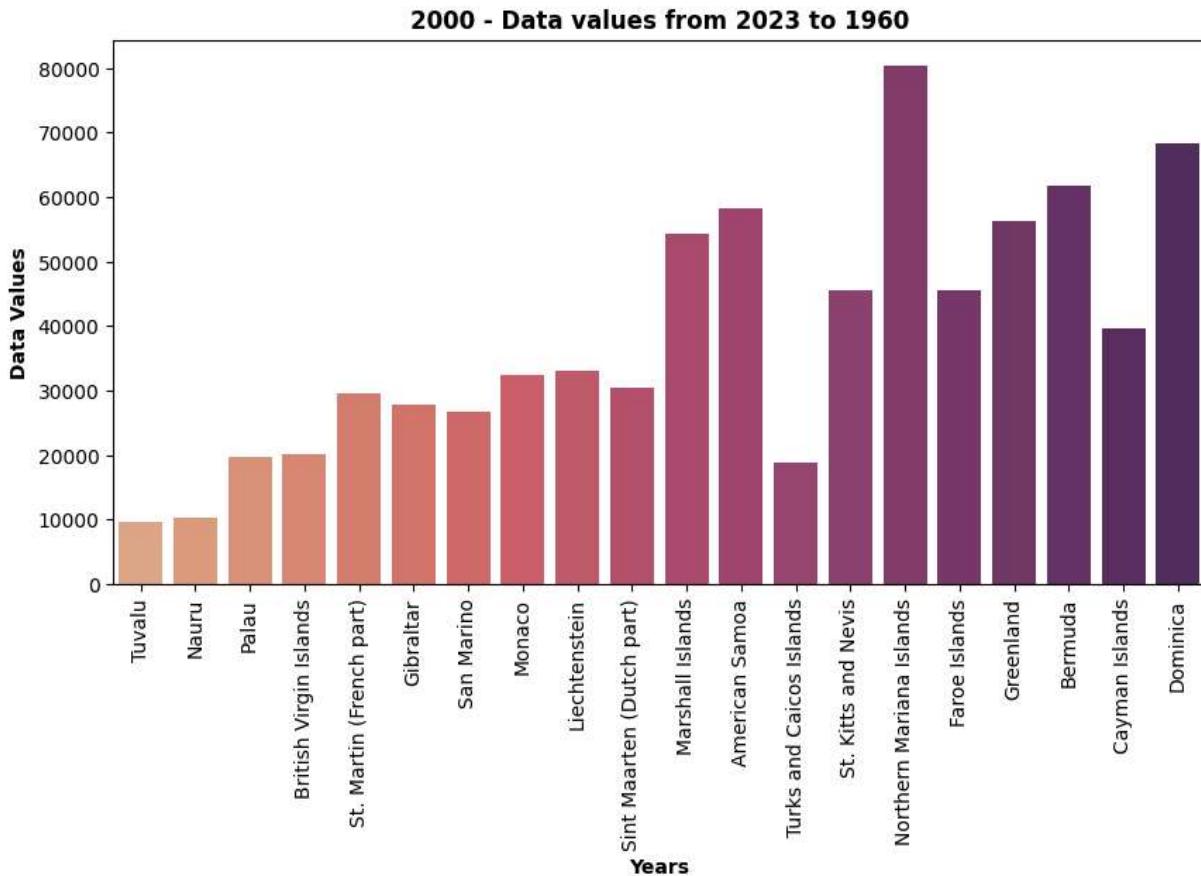
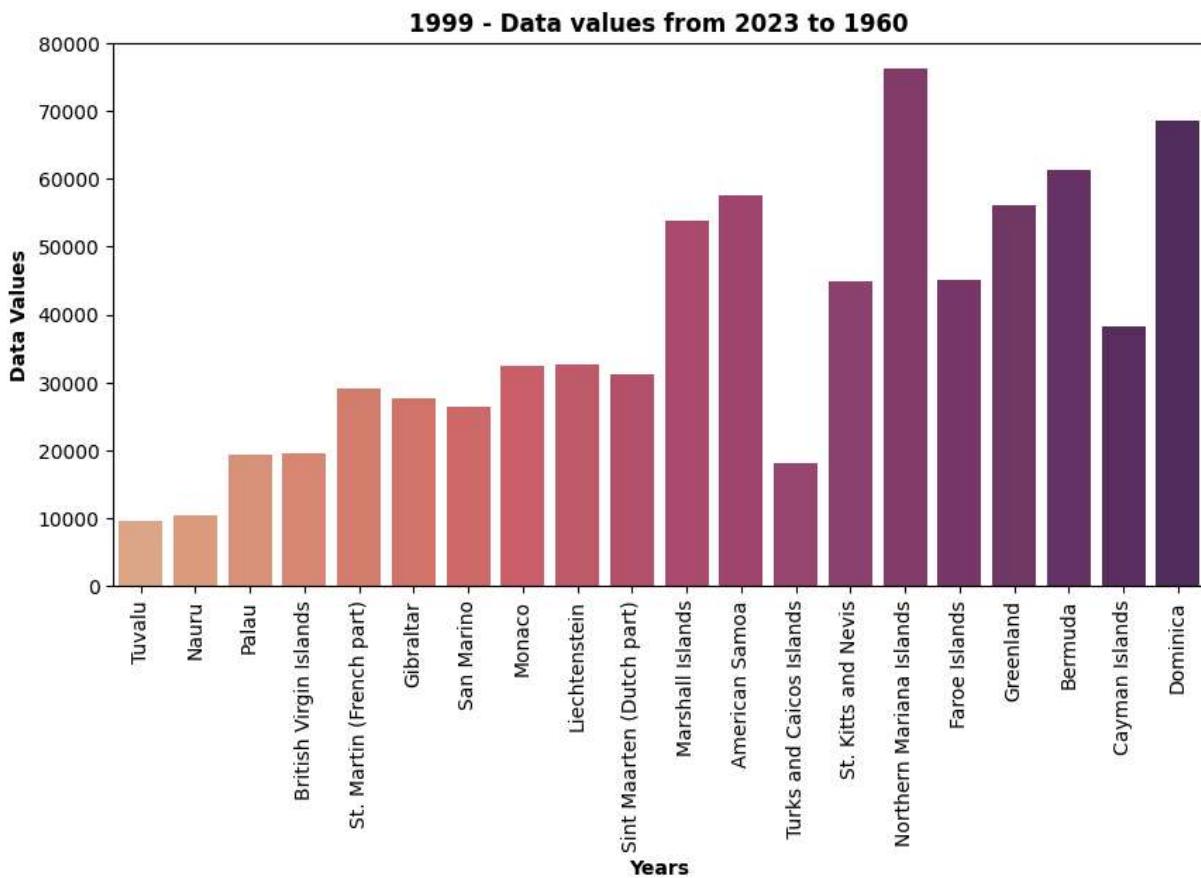
1989 - Data values from 2023 to 1960**1990 - Data values from 2023 to 1960**

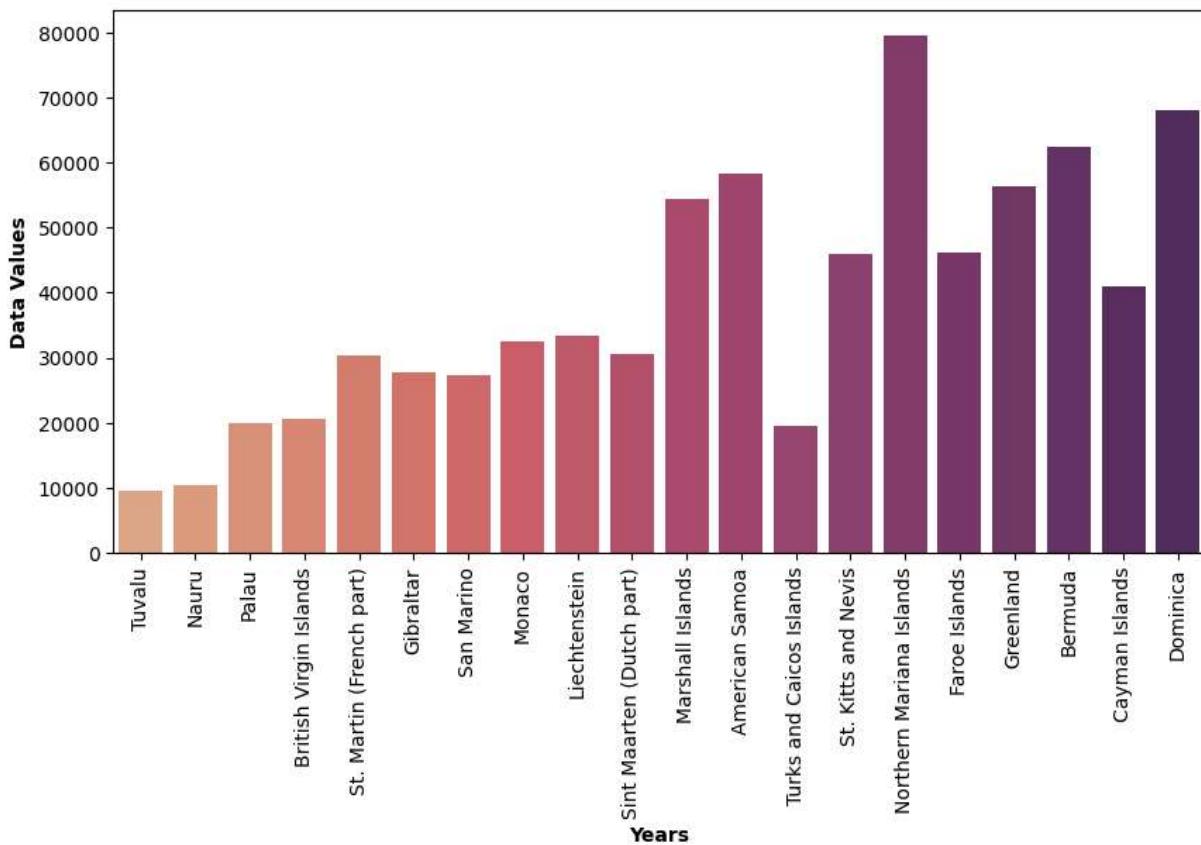
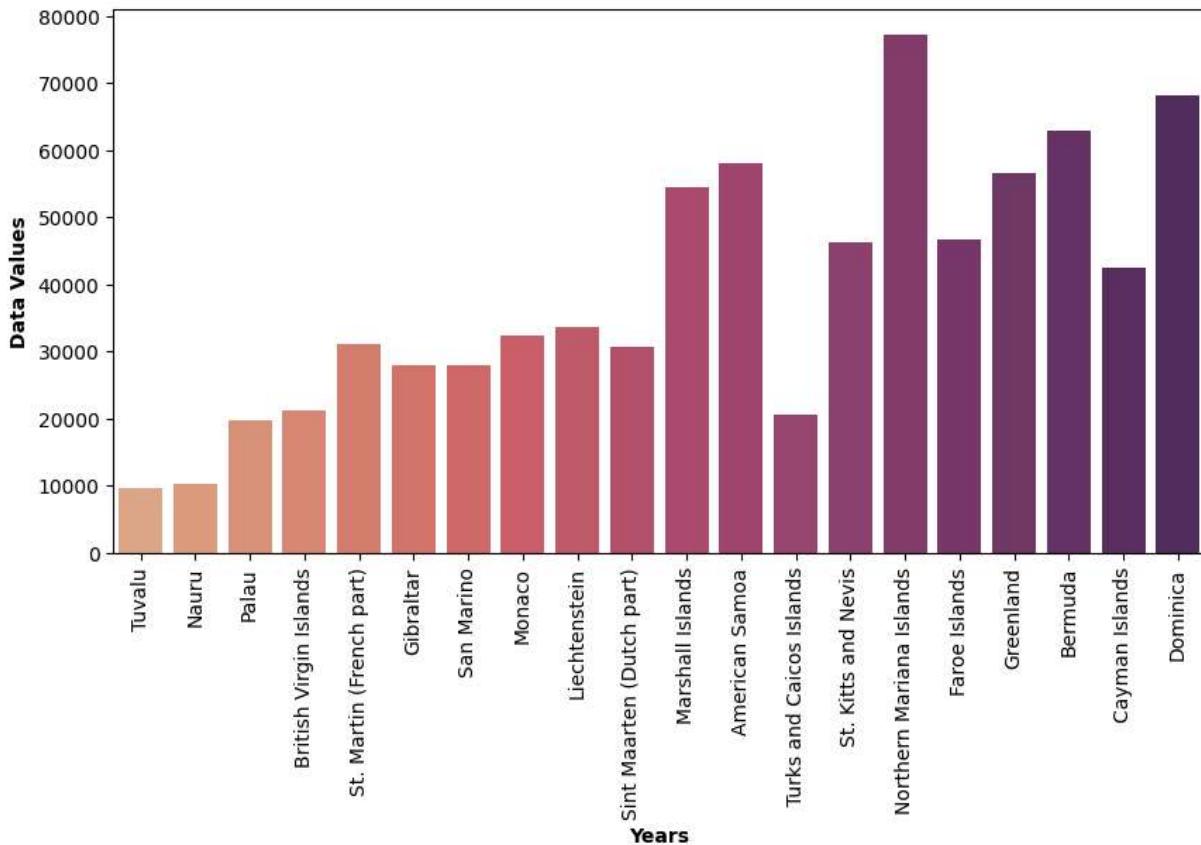
1991 - Data values from 2023 to 1960**1992 - Data values from 2023 to 1960**

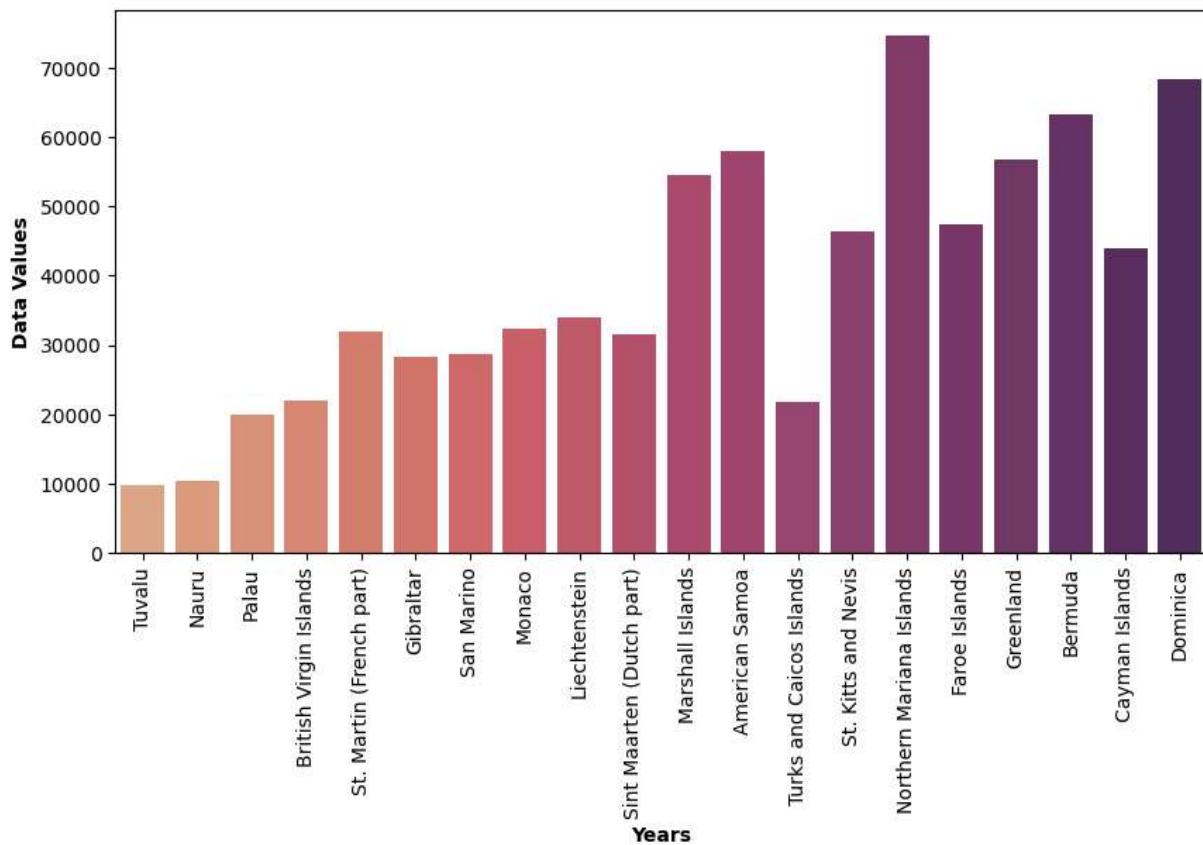
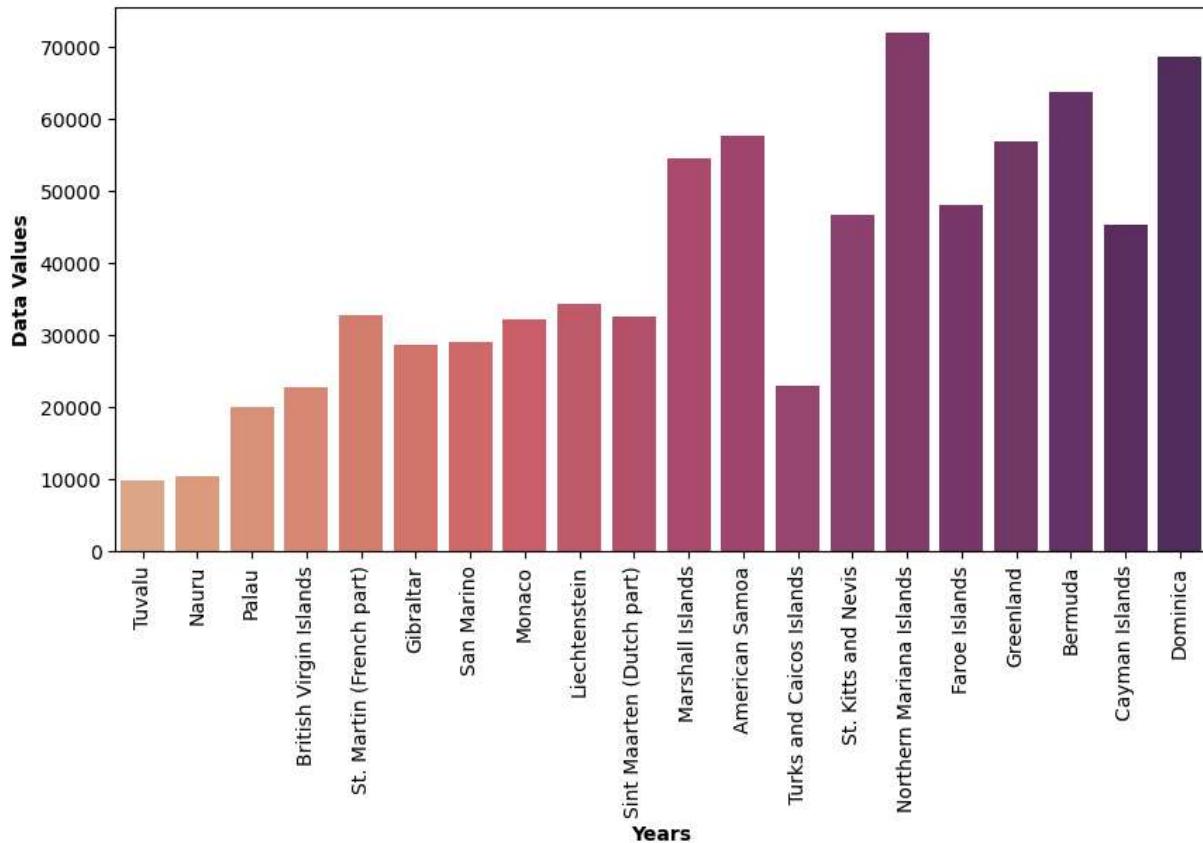
1993 - Data values from 2023 to 1960**1994 - Data values from 2023 to 1960**

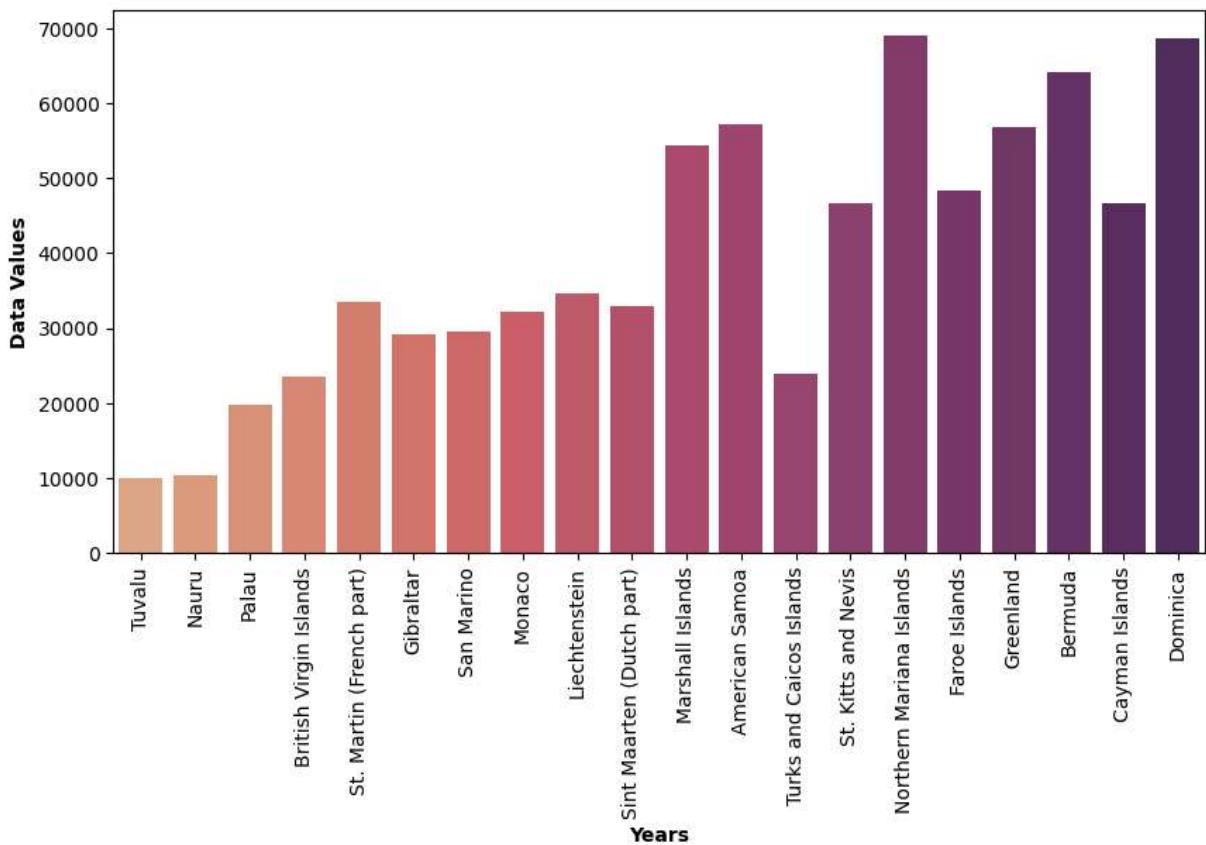
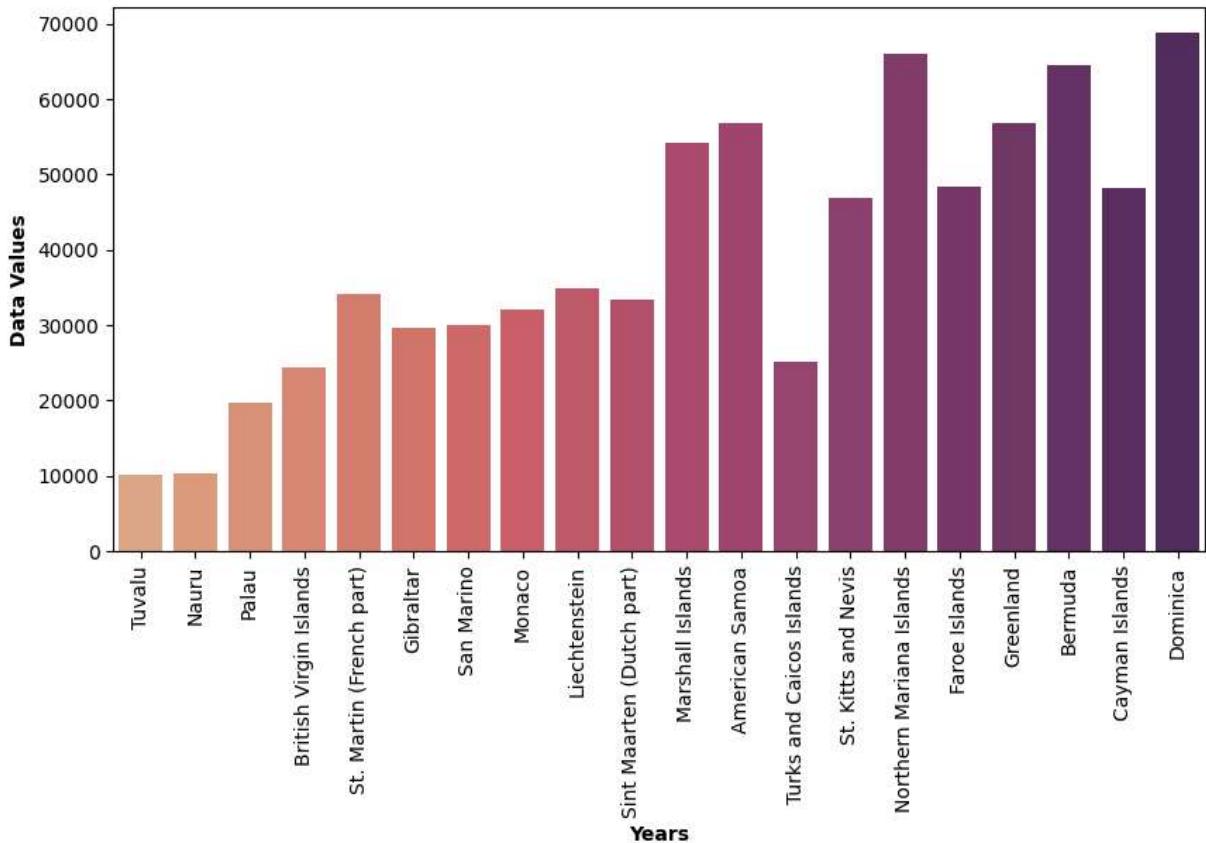
1995 - Data values from 2023 to 1960**1996 - Data values from 2023 to 1960**

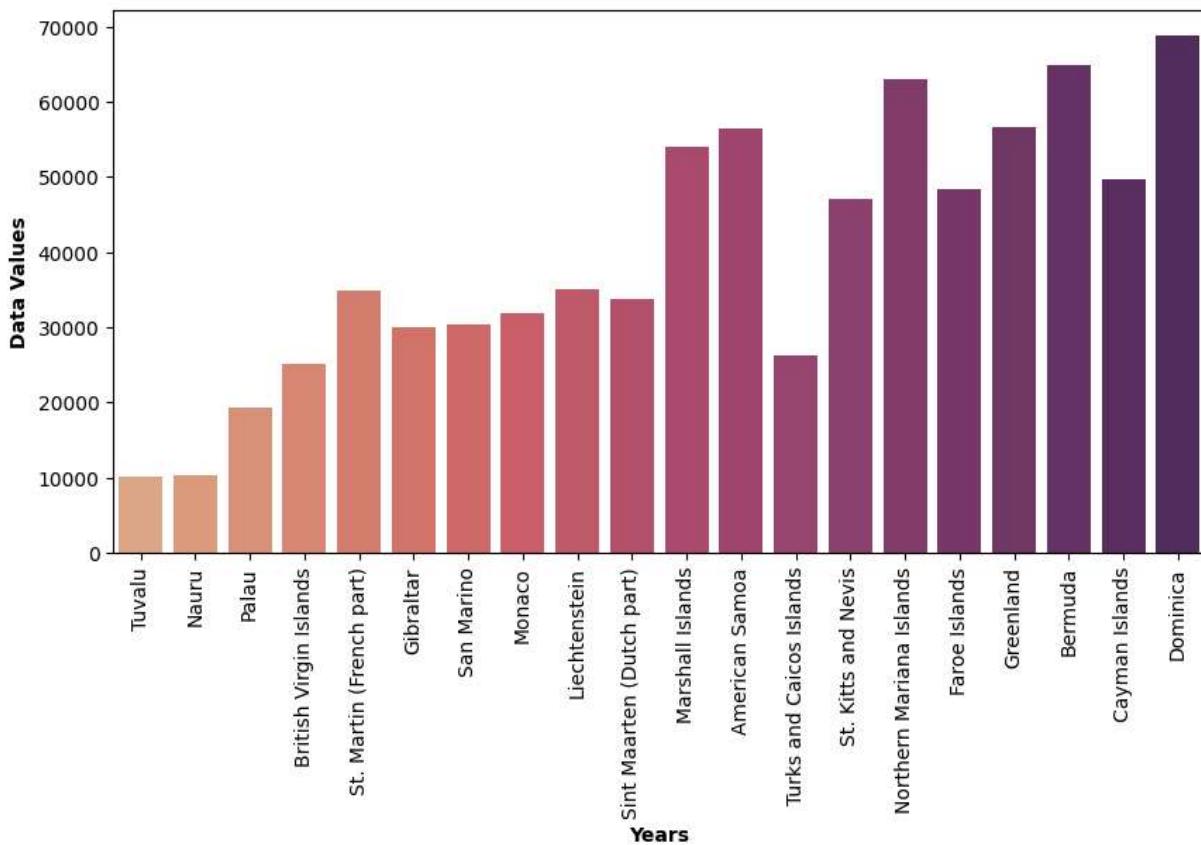
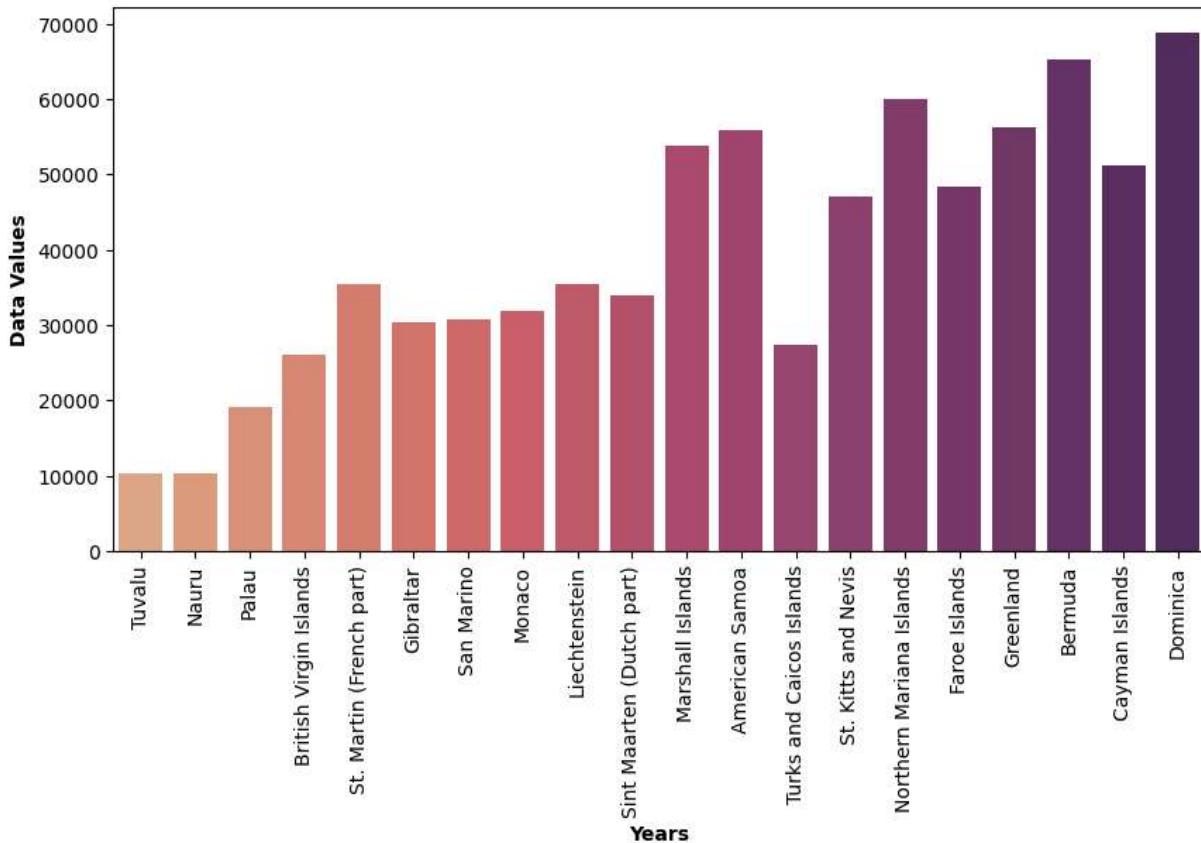
1997 - Data values from 2023 to 1960**1998 - Data values from 2023 to 1960**

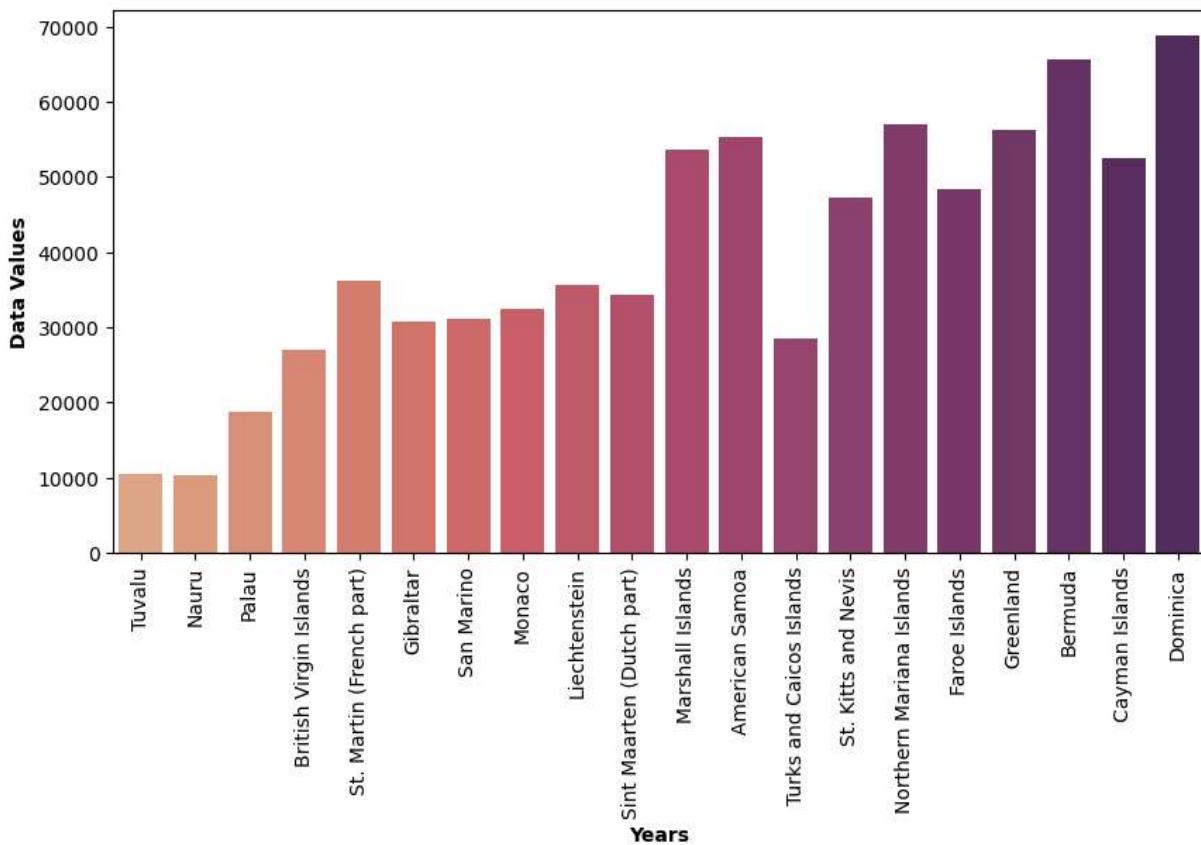
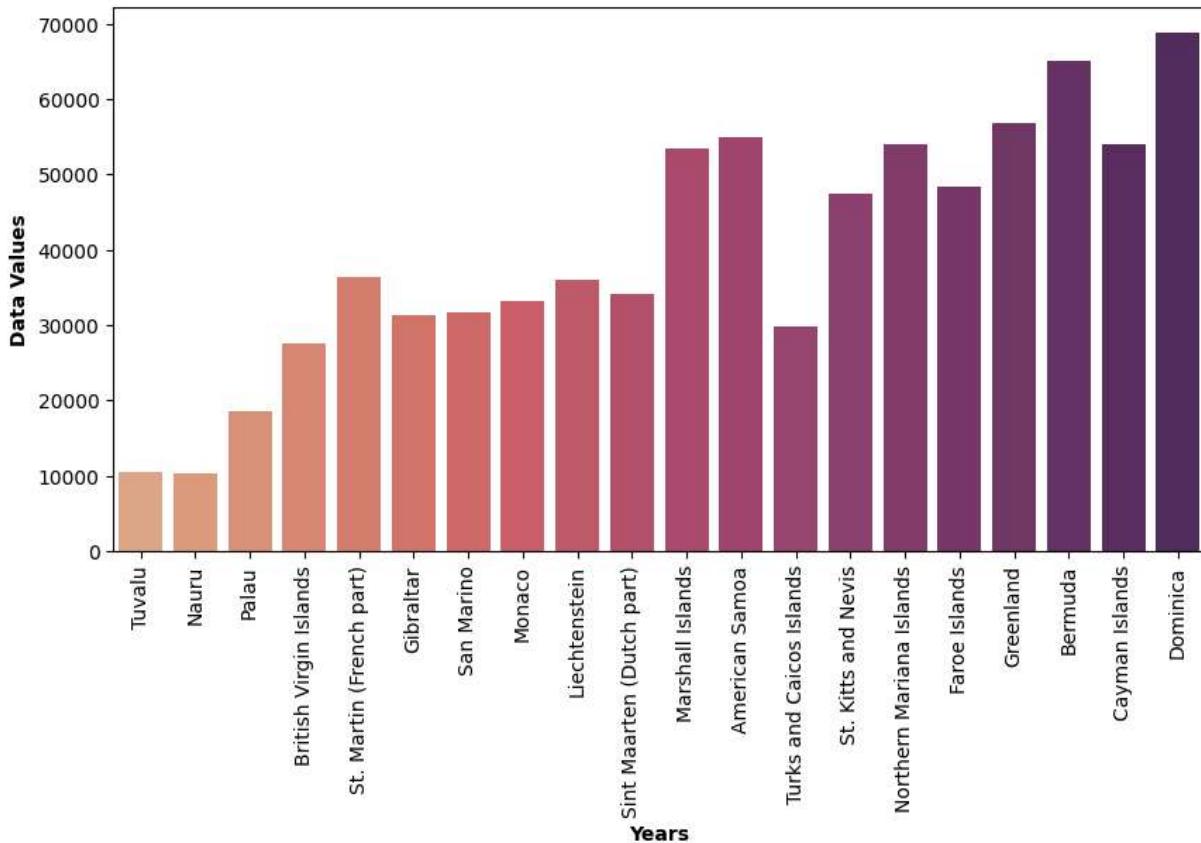


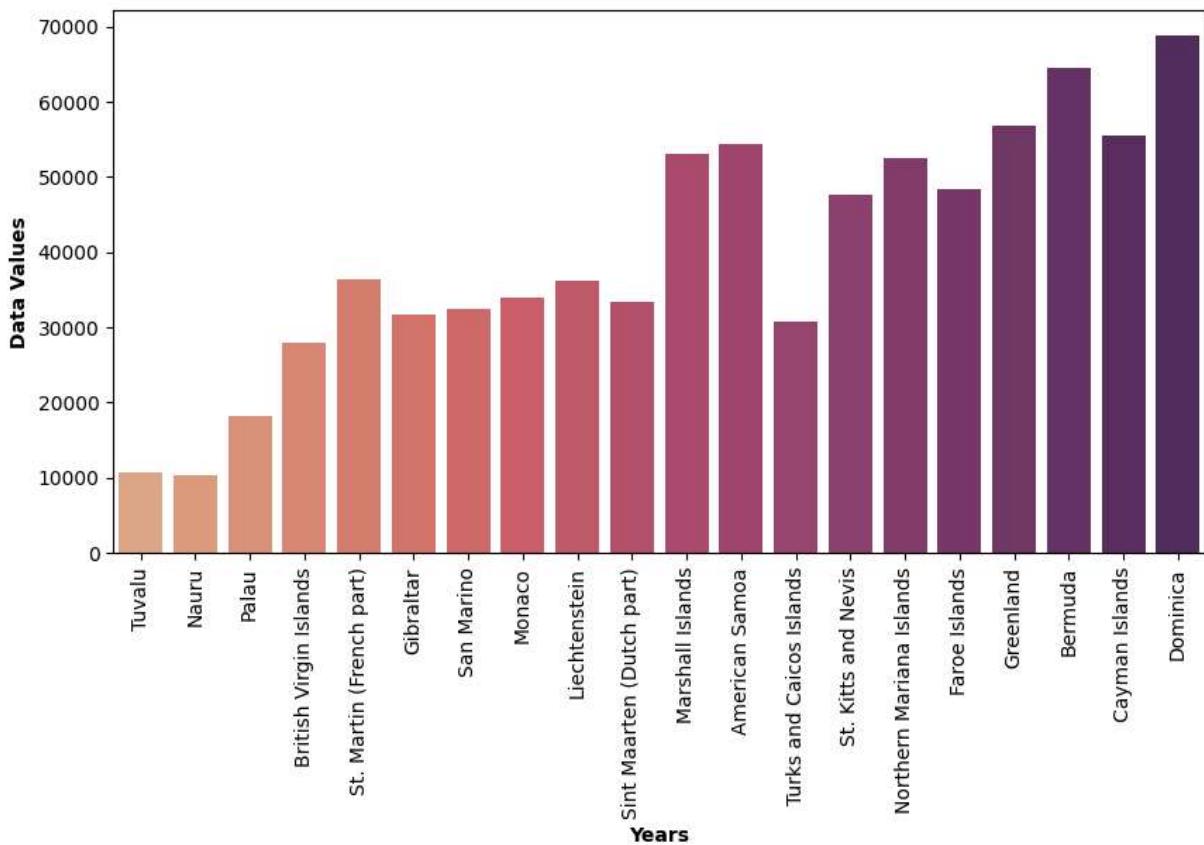
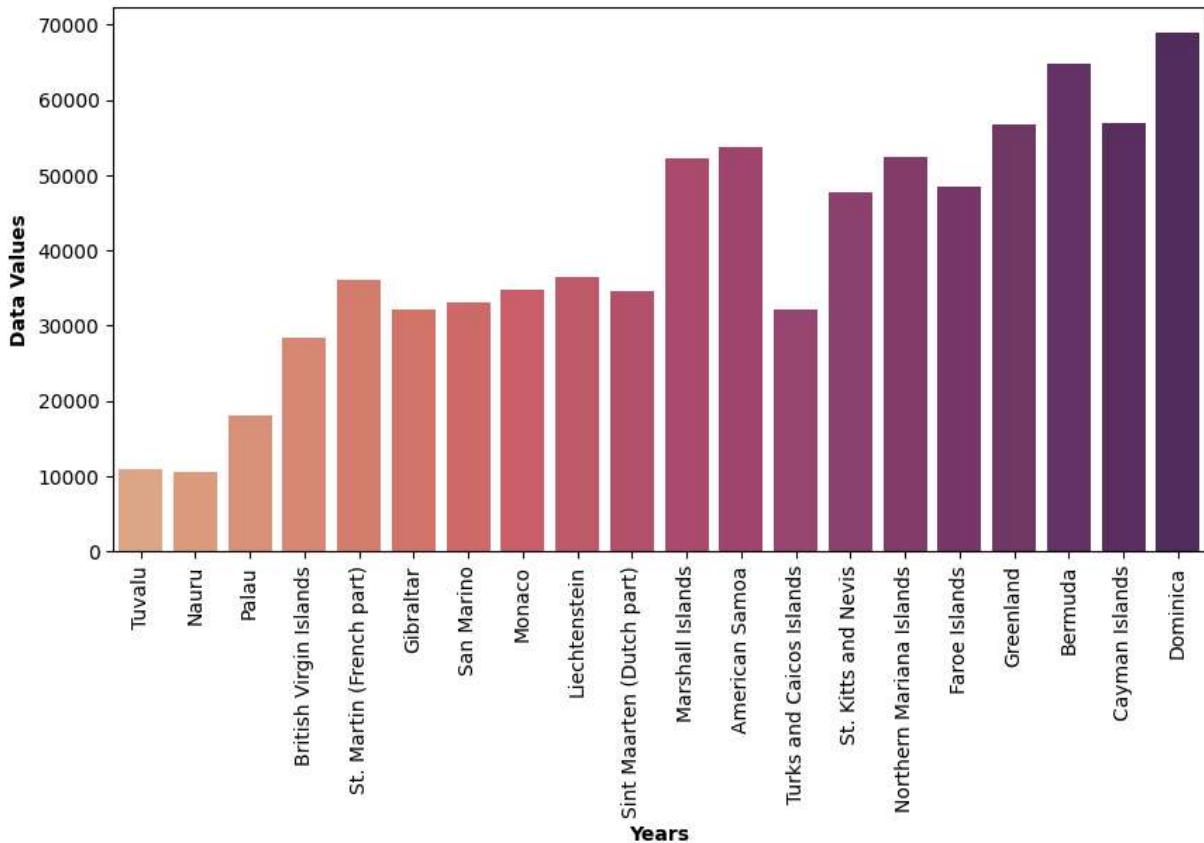
2001 - Data values from 2023 to 1960**2002 - Data values from 2023 to 1960**

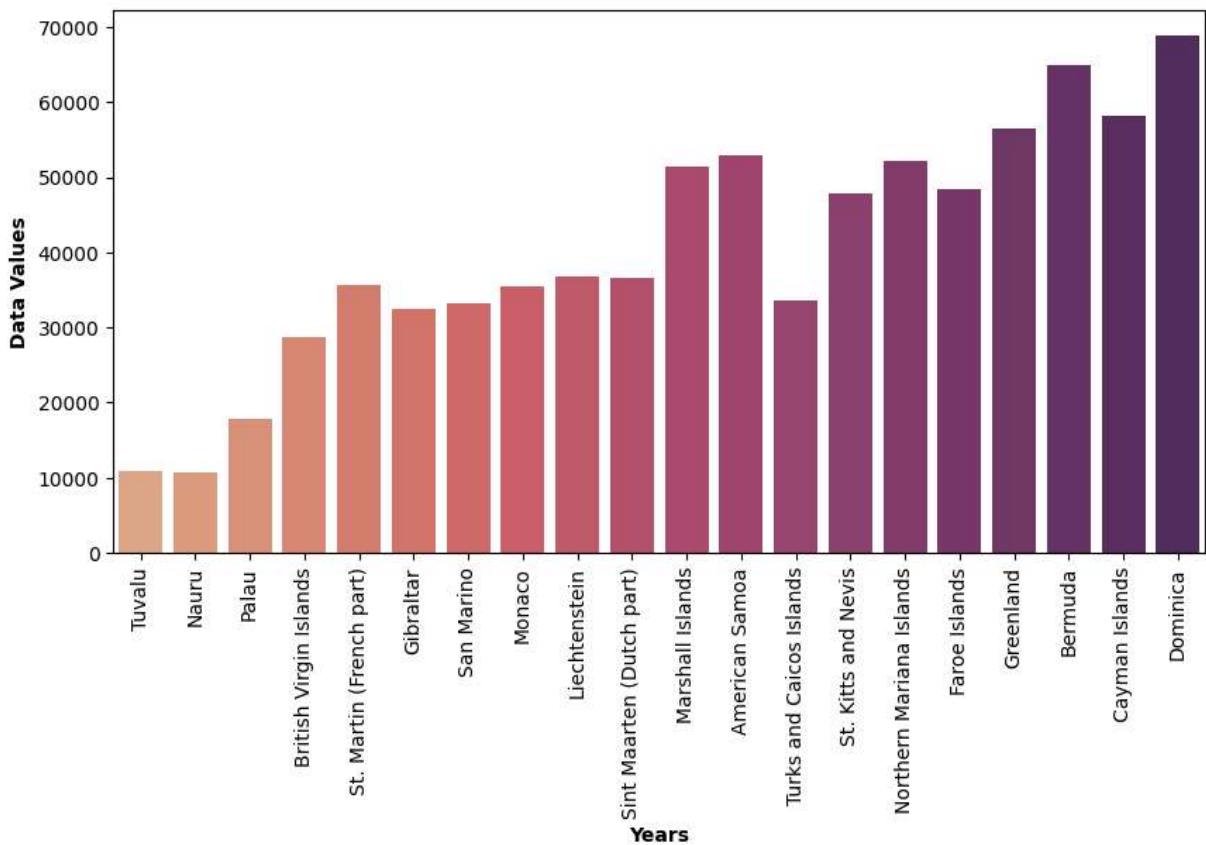
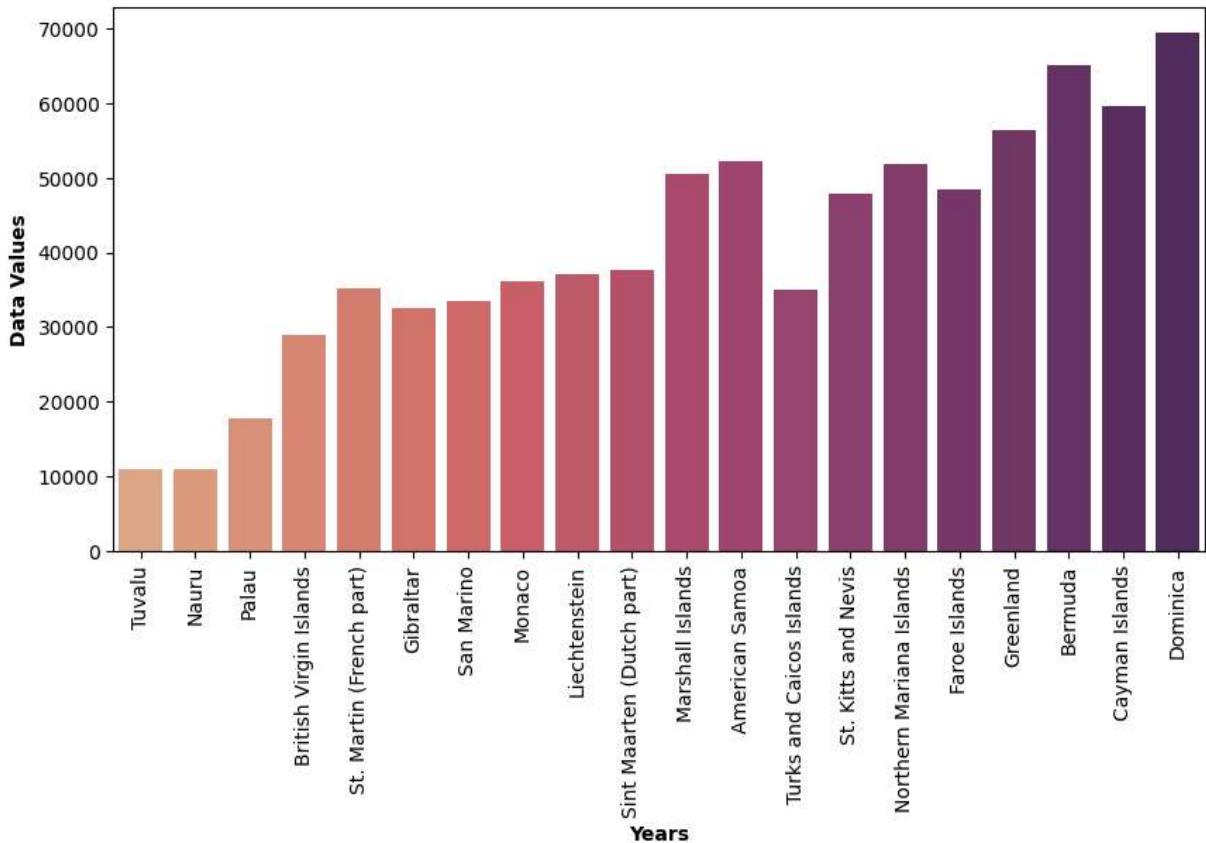
2003 - Data values from 2023 to 1960**2004 - Data values from 2023 to 1960**

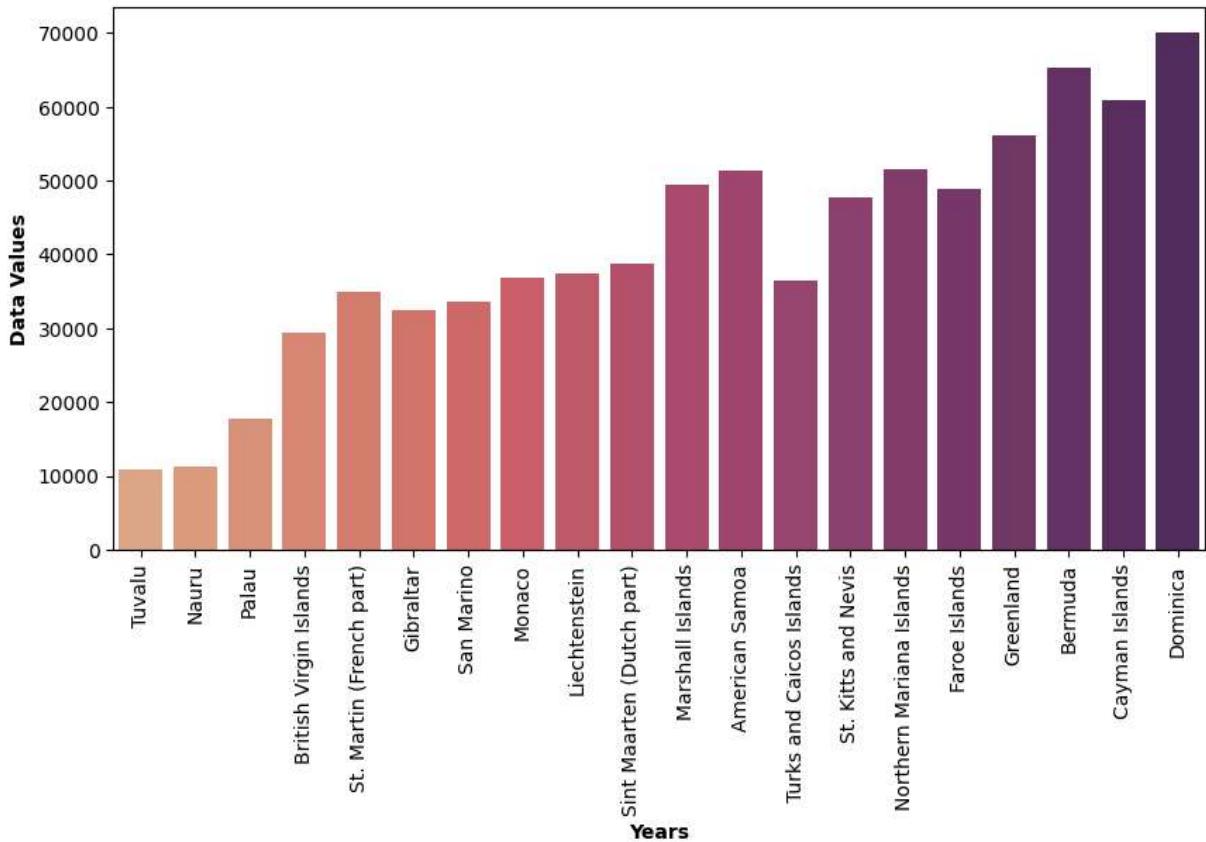
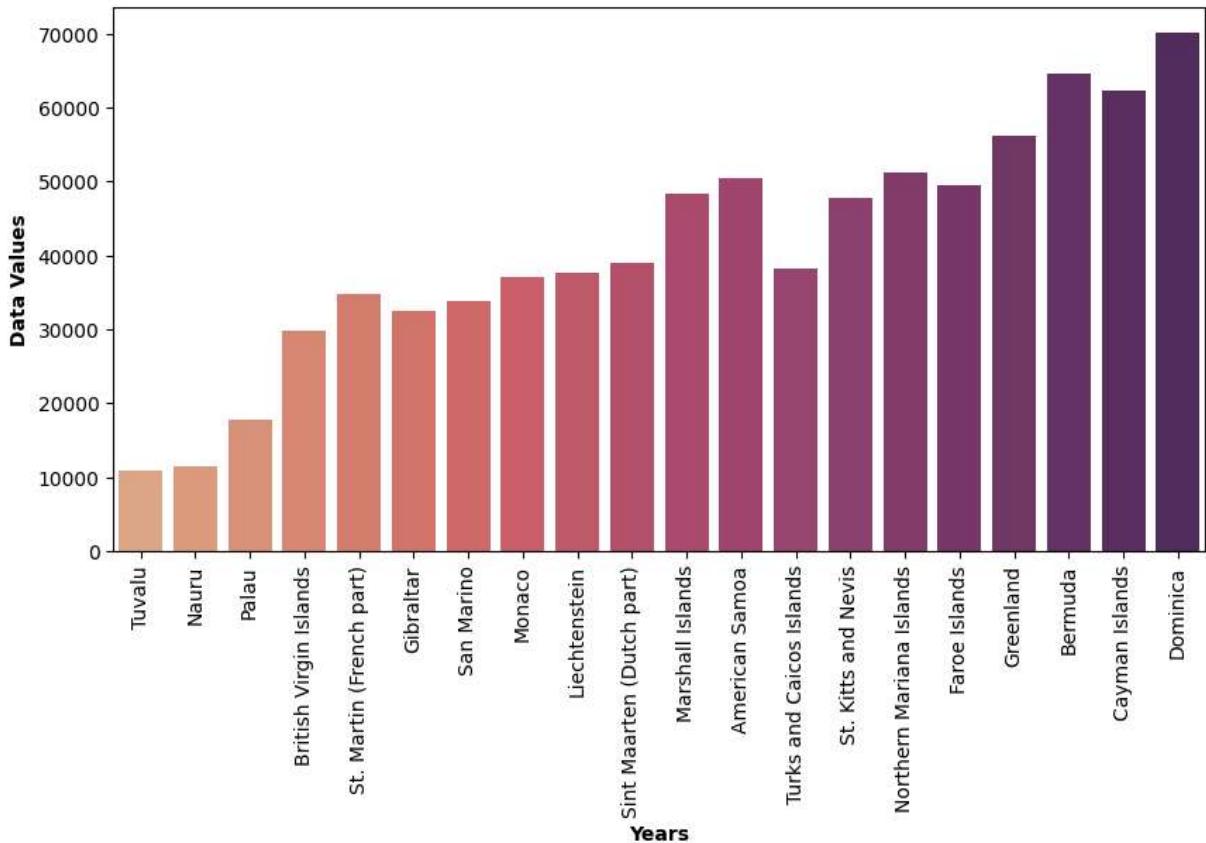
2005 - Data values from 2023 to 1960**2006 - Data values from 2023 to 1960**

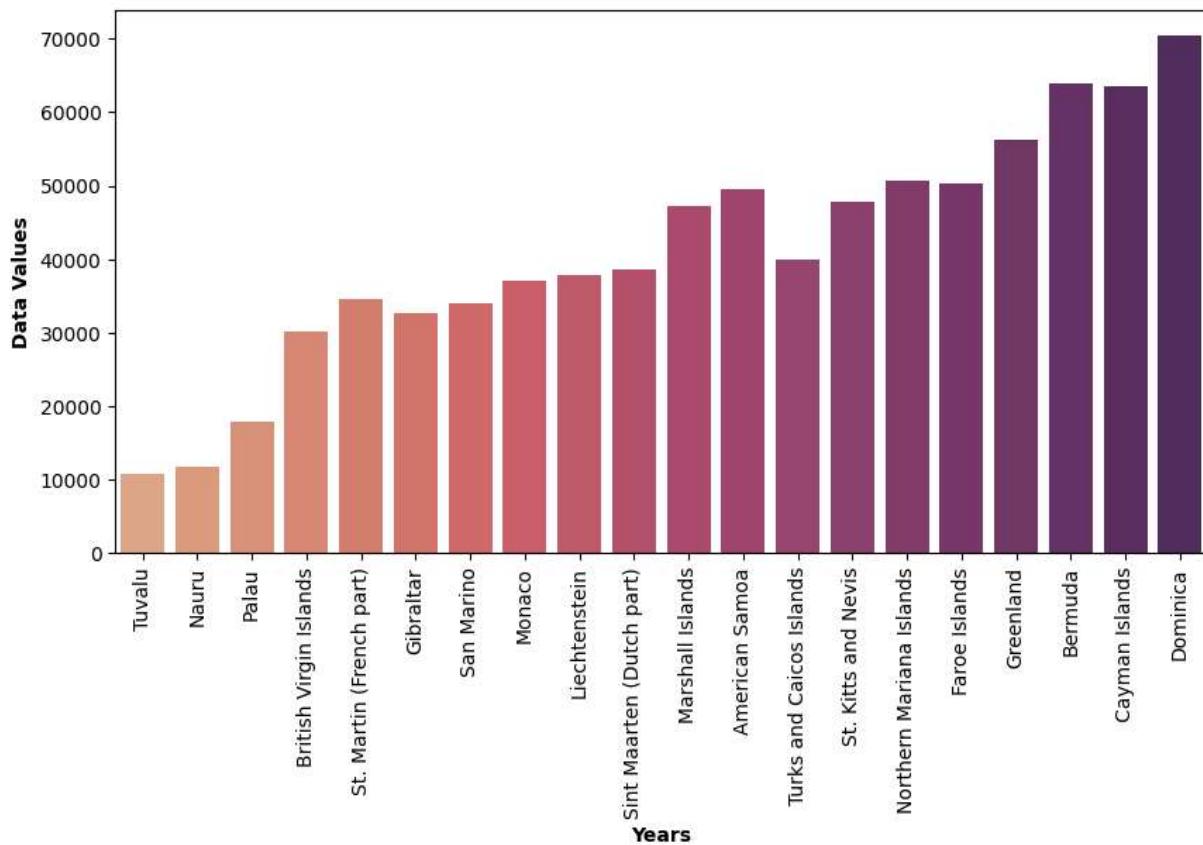
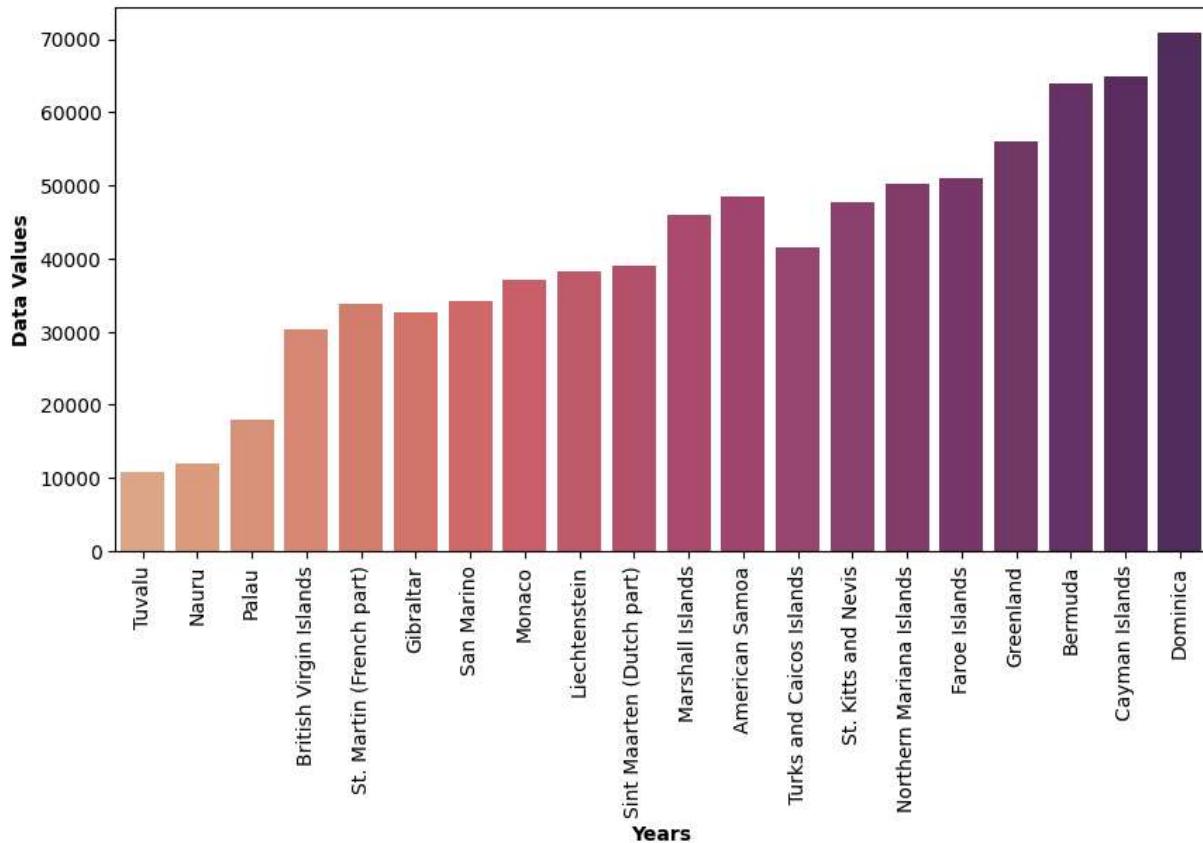
2007 - Data values from 2023 to 1960**2008 - Data values from 2023 to 1960**

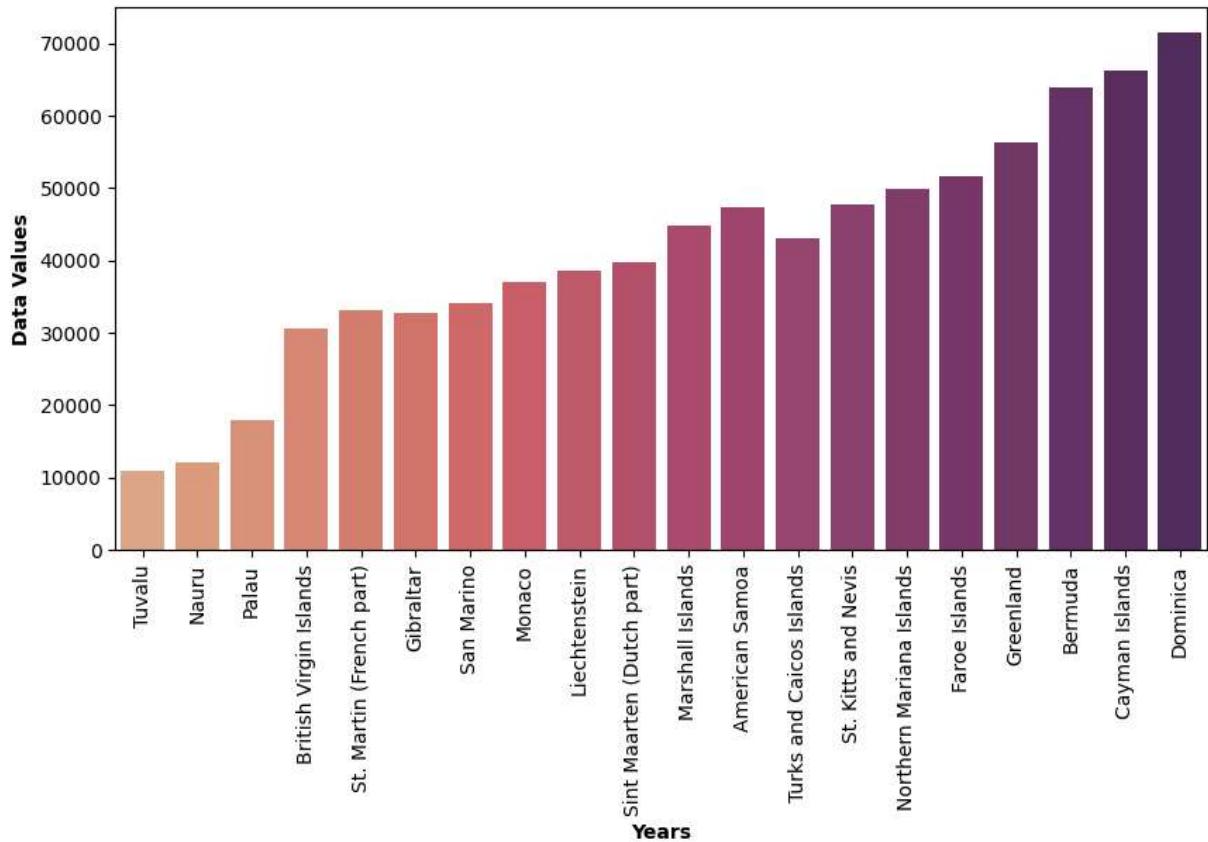
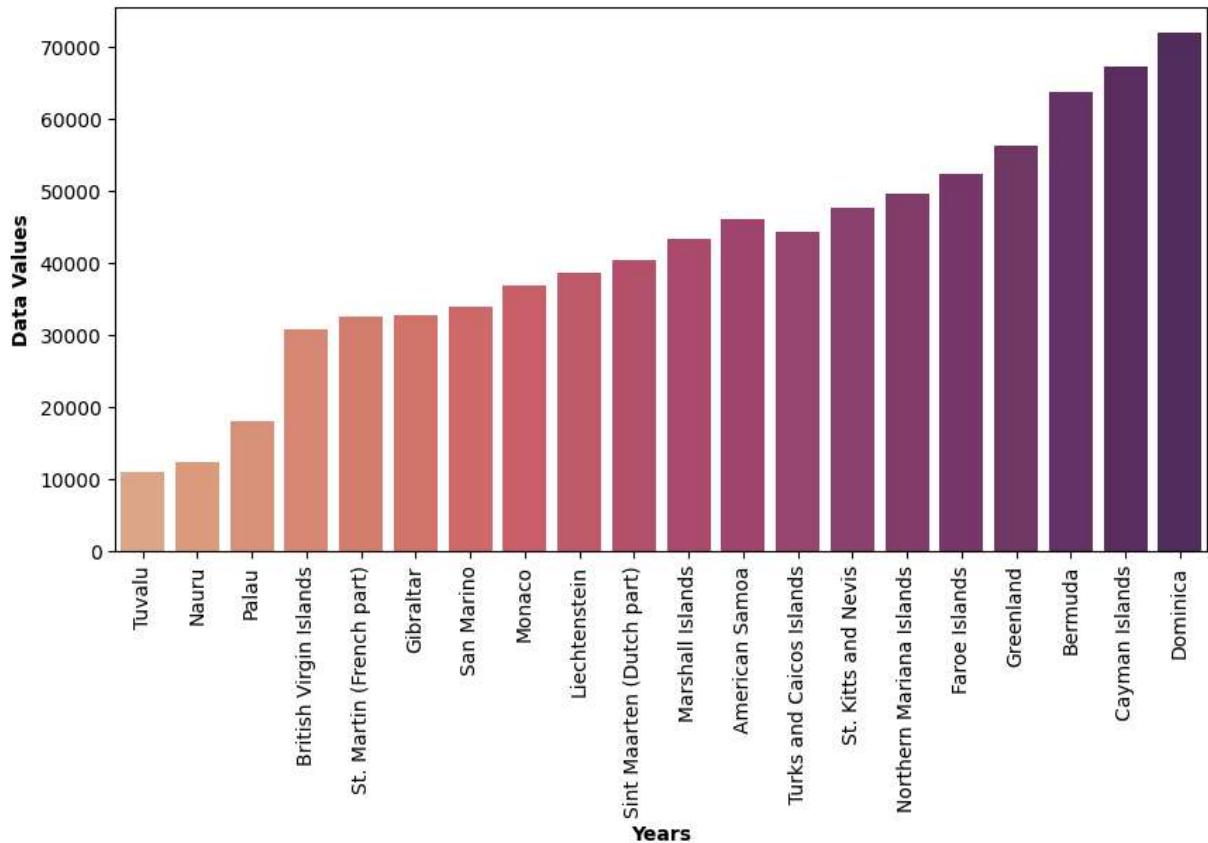
2009 - Data values from 2023 to 1960**2010 - Data values from 2023 to 1960**

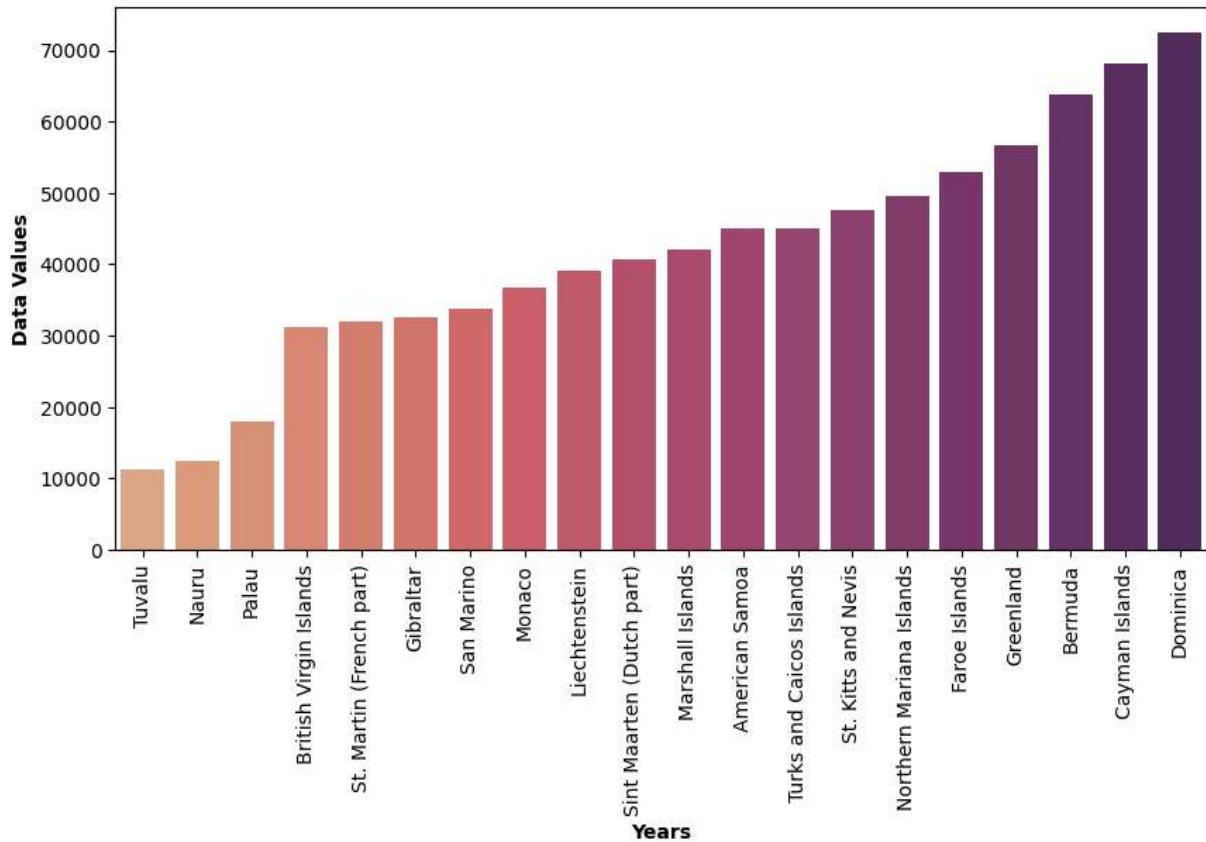
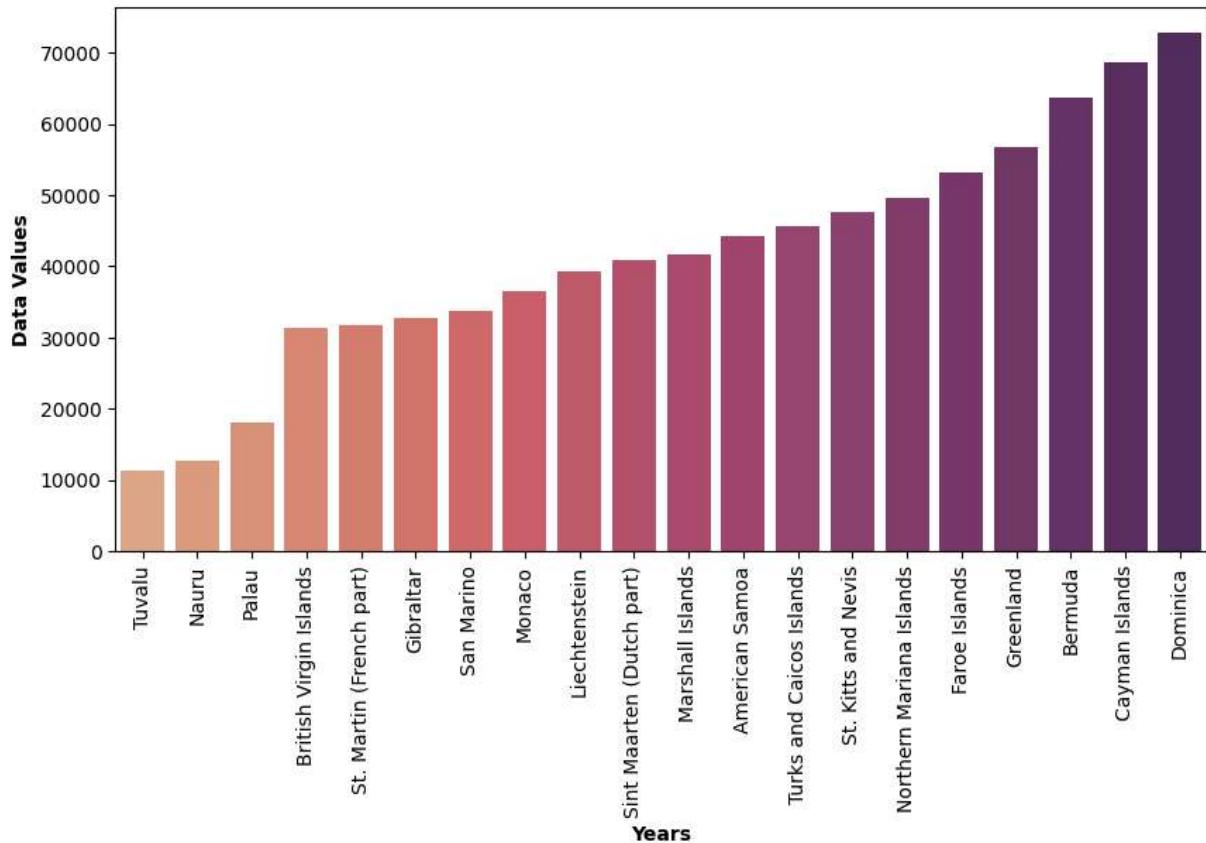
2011 - Data values from 2023 to 1960**2012 - Data values from 2023 to 1960**

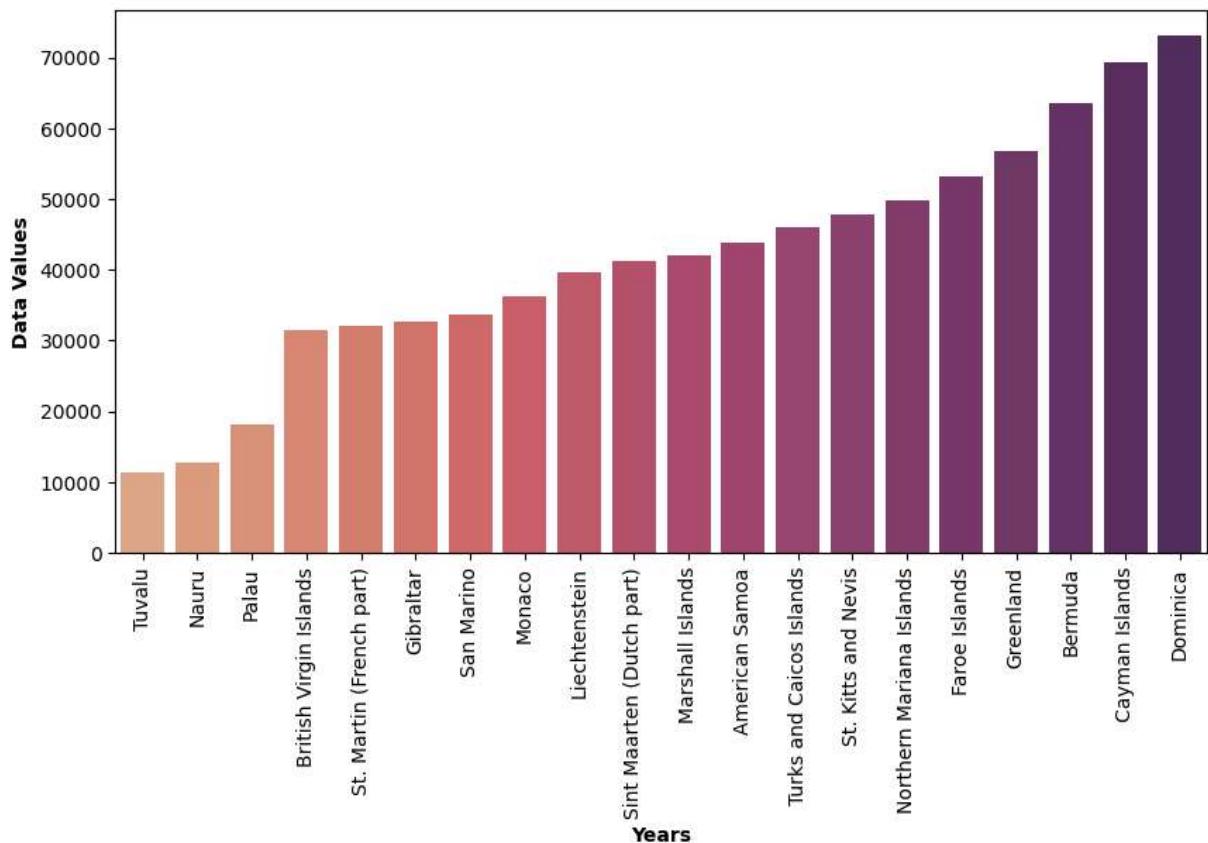
2013 - Data values from 2023 to 1960**2014 - Data values from 2023 to 1960**

2015 - Data values from 2023 to 1960**2016 - Data values from 2023 to 1960**

2017 - Data values from 2023 to 1960**2018 - Data values from 2023 to 1960**

2019 - Data values from 2023 to 1960**2020 - Data values from 2023 to 1960**

2021 - Data values from 2023 to 1960**2022 - Data values from 2023 to 1960**

2023 - Data values from 2023 to 1960

In []:

In []: