

Task 1

Create a bar chart or histogram to visualize the distribution of a categorical or continuous variable, such as the distribution of ages or genders in a population.

About the Dataset

We have taken a World Population Dataset consisting records of population from the year 2001 to 2022.

Series Code	Meaning
SP.POP.TOTL	Population, total
SP.POP.TOTL.FE.IN	Population, female
SP.POP.TOTL.MA.IN	Population, male
SP.POP.TOTL.FE.ZS	Population, female (% of total population)
SP.POP.TOTL.MA.ZS	Population, male (% of total population)

Importing necessary libraries

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

Reading the dataset

```
In [2]: df = pd.read_csv(r"worldpopulationdata.csv")
```

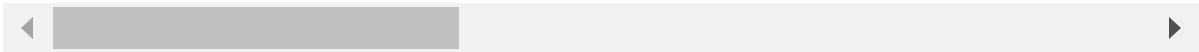
Checking the first 5 rows

```
In [3]: df.head(5)
```

Out[3]:

	Series Name	Series Code	Country Name	Country Code	2022	2021	2020	
0	Population, total	SP.POP.TOTL	Afghanistan	AFG	41128771.0	40099462.0	38972230.0	3776
1	Population, total	SP.POP.TOTL	Albania	ALB	2775634.0	2811666.0	2837849.0	285
2	Population, total	SP.POP.TOTL	Algeria	DZA	44903225.0	44177969.0	43451666.0	4270
3	Population, total	SP.POP.TOTL	American Samoa	ASM	44273.0	45035.0	46189.0	4
4	Population, total	SP.POP.TOTL	Andorra	AND	79824.0	79034.0	77700.0	7

5 rows × 26 columns



Checking the last 5 rows

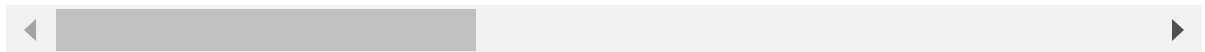
In [4]:

```
df.tail(5)
```

Out[4]:

	Series Name	Series Code	Country Name	Country Code	2022	2021	2020
1080	Population, male (% of total population)	SP.POP.TOTL.MA.ZS	Virgin Islands (U.S.)	VIR	46.613382	46.764444	46.914637
1081	Population, male (% of total population)	SP.POP.TOTL.MA.ZS	West Bank and Gaza	PSE	49.893678	49.877839	49.858957
1082	Population, male (% of total population)	SP.POP.TOTL.MA.ZS	Yemen, Rep.	YEM	50.519031	50.538516	50.554317
1083	Population, male (% of total population)	SP.POP.TOTL.MA.ZS	Zambia	ZMB	49.344602	49.344951	49.338301
1084	Population, male (% of total population)	SP.POP.TOTL.MA.ZS	Zimbabwe	ZWE	47.214139	47.167153	47.130679

5 rows × 26 columns



Checking the shape of the dataset

In [5]: `df.shape`

Out[5]: (1085, 26)

Checking the columns of the dataset

In [6]: `df.columns`

```
Out[6]: Index(['Series Name', 'Series Code', 'Country Name', 'Country Code', '2022',
              '2021', '2020', '2019', '2018', '2017', '2016', '2015', '2014', '2013',
              '2012', '2011', '2010', '2009', '2008', '2007', '2006', '2005', '2004',
              '2003', '2002', '2001'],
              dtype='object')
```

Some information about the dataset

In [7]: `df.info()`

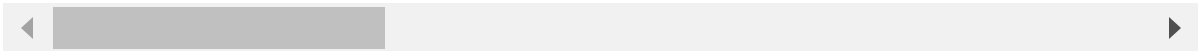
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1085 entries, 0 to 1084
Data columns (total 26 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Series Name     1085 non-null   object
1   Series Code     1085 non-null   object
2   Country Name    1085 non-null   object
3   Country Code    1085 non-null   object
4   2022            1085 non-null   float64
5   2021            1085 non-null   float64
6   2020            1085 non-null   float64
7   2019            1085 non-null   float64
8   2018            1085 non-null   float64
9   2017            1085 non-null   float64
10  2016            1085 non-null   float64
11  2015            1085 non-null   float64
12  2014            1085 non-null   float64
13  2013            1085 non-null   float64
14  2012            1085 non-null   float64
15  2011            1085 non-null   float64
16  2010            1085 non-null   float64
17  2009            1085 non-null   float64
18  2008            1085 non-null   float64
19  2007            1085 non-null   float64
20  2006            1085 non-null   float64
21  2005            1085 non-null   float64
22  2004            1085 non-null   float64
23  2003            1085 non-null   float64
24  2002            1085 non-null   float64
25  2001            1085 non-null   float64
dtypes: float64(22), object(4)
memory usage: 220.5+ KB
```

```
In [8]: df.describe()
```

Out[8]:

	2022	2021	2020	2019	2018	20
count	1.085000e+03	1.085000e+03	1.085000e+03	1.085000e+03	1.085000e+03	1.085000e+
mean	1.461378e+07	1.449711e+07	1.437307e+07	1.422876e+07	1.407966e+07	1.392568e+
std	7.832944e+07	7.801505e+07	7.763257e+07	7.712985e+07	7.657562e+07	7.596457e+
min	2.749000e+01	2.732503e+01	2.735104e+01	2.676295e+01	2.573928e+01	2.508394e+
25%	5.034029e+01	5.035172e+01	5.034171e+01	5.033040e+01	5.033917e+01	5.033041e+
50%	1.465500e+05	1.463660e+05	1.461650e+05	1.459570e+05	1.457520e+05	1.441350e+
75%	5.903468e+06	5.856733e+06	5.831404e+06	5.814422e+06	5.774185e+06	5.686999e+
max	1.417173e+09	1.412360e+09	1.411100e+09	1.407745e+09	1.402760e+09	1.396215e+

8 rows × 22 columns



Checking for duplicate rows

```
In [9]: df.duplicated().sum()
```

```
Out[9]: 0
```

Observation:

- There are no duplicate rows in the dataset

Checking for missing values

```
In [10]: df.isna().sum()
```

```
Out[10]: Series Name      0
Series Code      0
Country Name      0
Country Code      0
2022              0
2021              0
2020              0
2019              0
2018              0
2017              0
2016              0
2015              0
2014              0
2013              0
2012              0
2011              0
2010              0
2009              0
2008              0
2007              0
2006              0
2005              0
2004              0
2003              0
2002              0
2001              0
dtype: int64
```

Observation:

- no missing values present

Checking unique values for columns

```
In [11]: print(df['Country Name'].unique())
print("\nTotal no of unique countries:",df['Country Name'].nunique())
```

```
[ 'Afghanistan' 'Albania' 'Algeria' 'American Samoa' 'Andorra' 'Angola'
  'Antigua and Barbuda' 'Argentina' 'Armenia' 'Aruba' 'Australia' 'Austria'
  'Azerbaijan' 'Bahamas, The' 'Bahrain' 'Bangladesh' 'Barbados' 'Belarus'
  'Belgium' 'Belize' 'Benin' 'Bermuda' 'Bhutan' 'Bolivia'
  'Bosnia and Herzegovina' 'Botswana' 'Brazil' 'British Virgin Islands'
  'Brunei Darussalam' 'Bulgaria' 'Burkina Faso' 'Burundi' 'Cabo Verde'
  'Cambodia' 'Cameroon' 'Canada' 'Cayman Islands'
  'Central African Republic' 'Chad' 'Channel Islands' 'Chile' 'China'
  'Colombia' 'Comoros' 'Congo, Dem. Rep.' 'Congo, Rep.' 'Costa Rica'
  "Cote d'Ivoire" 'Croatia' 'Cuba' 'Curacao' 'Cyprus' 'Czechia' 'Denmark'
  'Djibouti' 'Dominica' 'Dominican Republic' 'Ecuador' 'Egypt, Arab Rep.'
  'El Salvador' 'Equatorial Guinea' 'Eritrea' 'Estonia' 'Eswatini'
  'Ethiopia' 'Faroe Islands' 'Fiji' 'Finland' 'France' 'French Polynesia'
  'Gabon' 'Gambia, The' 'Georgia' 'Germany' 'Ghana' 'Gibraltar' 'Greece'
  'Greenland' 'Grenada' 'Guam' 'Guatemala' 'Guinea' 'Guinea-Bissau'
  'Guyana' 'Haiti' 'Honduras' 'Hong Kong SAR, China' 'Hungary' 'Iceland'
  'India' 'Indonesia' 'Iran, Islamic Rep.' 'Iraq' 'Ireland' 'Isle of Man'
  'Israel' 'Italy' 'Jamaica' 'Japan' 'Jordan' 'Kazakhstan' 'Kenya'
  'Kiribati' "Korea, Dem. People's Rep." 'Korea, Rep.' 'Kosovo' 'Kuwait'
  'Kyrgyz Republic' 'Lao PDR' 'Latvia' 'Lebanon' 'Lesotho' 'Liberia'
  'Libya' 'Liechtenstein' 'Lithuania' 'Luxembourg' 'Macao SAR, China'
  'Madagascar' 'Malawi' 'Malaysia' 'Maldives' 'Mali' 'Malta'
  'Marshall Islands' 'Mauritania' 'Mauritius' 'Mexico'
  'Micronesia, Fed. Sts.' 'Moldova' 'Monaco' 'Mongolia' 'Montenegro'
  'Morocco' 'Mozambique' 'Myanmar' 'Namibia' 'Nauru' 'Nepal' 'Netherlands'
  'New Caledonia' 'New Zealand' 'Nicaragua' 'Niger' 'Nigeria'
  'North Macedonia' 'Northern Mariana Islands' 'Norway' 'Oman' 'Pakistan'
  'Palau' 'Panama' 'Papua New Guinea' 'Paraguay' 'Peru' 'Philippines'
  'Poland' 'Portugal' 'Puerto Rico' 'Qatar' 'Romania' 'Russian Federation'
  'Rwanda' 'Samoa' 'San Marino' 'Sao Tome and Principe' 'Saudi Arabia'
  'Senegal' 'Serbia' 'Seychelles' 'Sierra Leone' 'Singapore'
  'Sint Maarten (Dutch part)' 'Slovak Republic' 'Slovenia'
  'Solomon Islands' 'Somalia' 'South Africa' 'South Sudan' 'Spain'
  'Sri Lanka' 'St. Kitts and Nevis' 'St. Lucia' 'St. Martin (French part)'
  'St. Vincent and the Grenadines' 'Sudan' 'Suriname' 'Sweden'
  'Switzerland' 'Syrian Arab Republic' 'Tajikistan' 'Tanzania' 'Thailand'
  'Timor-Leste' 'Togo' 'Tonga' 'Trinidad and Tobago' 'Tunisia' 'Turkiye'
  'Turkmenistan' 'Turks and Caicos Islands' 'Tuvalu' 'Uganda' 'Ukraine'
  'United Arab Emirates' 'United Kingdom' 'United States' 'Uruguay'
  'Uzbekistan' 'Vanuatu' 'Venezuela, RB' 'Vietnam' 'Virgin Islands (U.S.)'
  'West Bank and Gaza' 'Yemen, Rep.' 'Zambia' 'Zimbabwe']
```

Total no of unique countries: 217

```
In [12]: print(df['Country Code'].unique())
         print("\nTotal no of unique country code:",df['Country Code'].nunique())
```

```
[ 'AFG' 'ALB' 'DZA' 'ASM' 'AND' 'AGO' 'ATG' 'ARG' 'ARM' 'ABW' 'AUS' 'AUT'
  'AZE' 'BHS' 'BHR' 'BGD' 'BRB' 'BLR' 'BEL' 'BLZ' 'BEN' 'BMU' 'BTN' 'BOL'
  'BIH' 'BWA' 'BRA' 'VGB' 'BRN' 'BGR' 'BFA' 'BDI' 'CPV' 'KHM' 'CMR' 'CAN'
  'CYM' 'CAF' 'TCD' 'CHI' 'CHL' 'CHN' 'COL' 'COM' 'COD' 'COG' 'CRI' 'CIV'
  'HRV' 'CUB' 'CUW' 'CYP' 'CZE' 'DNK' 'DJI' 'DMA' 'DOM' 'ECU' 'EGY' 'SLV'
  'GNQ' 'ERI' 'EST' 'SWZ' 'ETH' 'FRO' 'FJI' 'FIN' 'FRA' 'PYF' 'GAB' 'GMB'
  'GEO' 'DEU' 'GHA' 'GIB' 'GRC' 'GRL' 'GRD' 'GUM' 'GTM' 'GIN' 'GNB' 'GUY'
  'HTI' 'HND' 'HKG' 'HUN' 'ISL' 'IND' 'IDN' 'IRN' 'IRQ' 'IRL' 'IMN' 'ISR'
  'ITA' 'JAM' 'JPN' 'JOR' 'KAZ' 'KEN' 'KIR' 'PRK' 'KOR' 'XKX' 'KWT' 'KGZ'
  'LAO' 'LVA' 'LBN' 'LSO' 'LBR' 'LBY' 'LIE' 'LTU' 'LUX' 'MAC' 'MDG' 'MWI'
  'MYS' 'MDV' 'MLI' 'MLT' 'MHL' 'MRT' 'MUS' 'MEX' 'FSM' 'MDA' 'MCO' 'MNG'
  'MNE' 'MAR' 'MOZ' 'MMR' 'NAM' 'NRU' 'NPL' 'NLD' 'NCL' 'NZL' 'NIC' 'NER'
  'NGA' 'MKD' 'MNP' 'NOR' 'OMN' 'PAK' 'PLW' 'PAN' 'PNG' 'PRY' 'PER' 'PHL'
  'POL' 'PRT' 'PRI' 'QAT' 'ROU' 'RUS' 'RWA' 'WSM' 'SMR' 'STP' 'SAU' 'SEN'
  'SRB' 'SYC' 'SLE' 'SGP' 'SXM' 'SVK' 'SVN' 'SLB' 'SOM' 'ZAF' 'SSD' 'ESP'
  'LKA' 'KNA' 'LCA' 'MAF' 'VCT' 'SDN' 'SUR' 'SWE' 'CHE' 'SYR' 'TJK' 'TZA'
  'THA' 'TLS' 'TGO' 'TON' 'TTO' 'TUN' 'TUR' 'TKM' 'TCA' 'TUV' 'UGA' 'UKR'
  'ARE' 'GBR' 'USA' 'URY' 'UZB' 'VUT' 'VEN' 'VNM' 'VIR' 'PSE' 'YEM' 'ZMB'
  'ZWE']
```

Total no of unique country code: 217

```
In [13]: df['Series Name'].unique()
```

```
Out[13]: array(['Population, total', 'Population, female', 'Population, male',
                'Population, female (% of total population)',
                'Population, male (% of total population)'], dtype=object)
```

```
In [14]: df['Series Code'].unique()
```

```
Out[14]: array(['SP.POP.TOTL', 'SP.POP.TOTL.FE.IN', 'SP.POP.TOTL.MA.IN',
                'SP.POP.TOTL.FE.ZS', 'SP.POP.TOTL.MA.ZS'], dtype=object)
```

Dropping unnecessary columns

```
In [15]: df.drop(['Series Name', 'Country Code'], axis=1, inplace=True)
```

```
In [16]: df.columns
```

```
Out[16]: Index(['Series Code', 'Country Name', '2022', '2021', '2020', '2019', '2018',
                '2017', '2016', '2015', '2014', '2013', '2012', '2011', '2010', '2009',
                '2008', '2007', '2006', '2005', '2004', '2003', '2002', '2001'],
                dtype='object')
```

Extraction of top-10 countries with respect to total population

```
In [17]: # Filter data for total population
total_population_data = df[df['Series Code'] == 'SP.POP.TOTL']

# Sort data based on the total population for 2022
total_population_sorted = total_population_data.sort_values(by="2022", ascending=False)
```

```
# Get the top ten countries with the highest total population for 2022
total_top_ten_countries = total_population_sorted.head(10)
print("Top ten countries of total population\n")
print(total_top_ten_countries[['Country Name']])
```

Top ten countries of total population

	Country Name
89	India
41	China
206	United States
90	Indonesia
149	Pakistan
144	Nigeria
26	Brazil
15	Bangladesh
161	Russian Federation
127	Mexico

Bar Plot

Top ten countries of total population in year 2022 and 2016

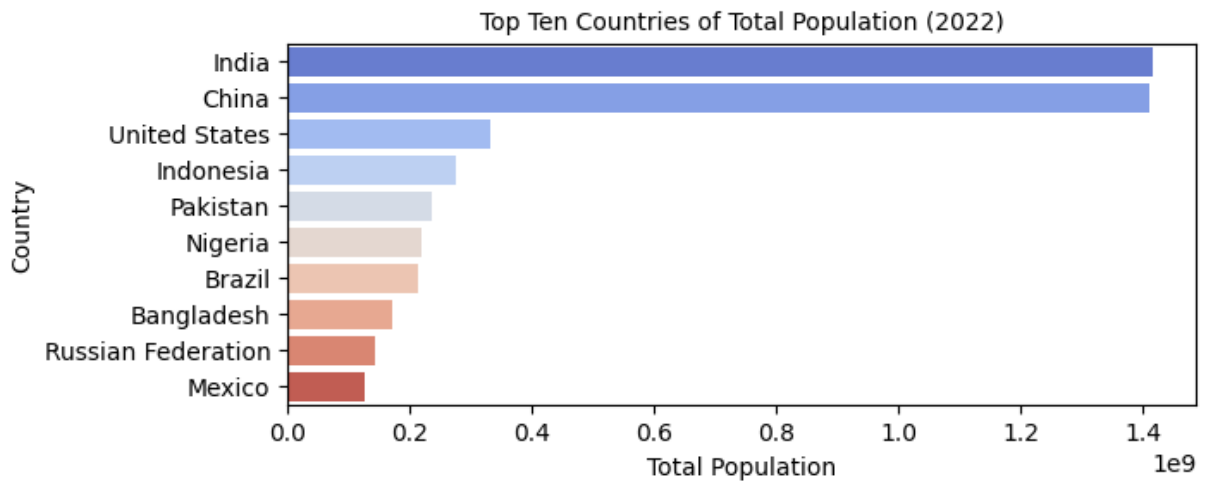
```
In [18]: # Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2022", y="Country Name", data=total_top_ten_countries, palette="coolw")
plt.title("Top Ten Countries of Total Population (2022)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()

plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2016", y="Country Name", data=total_top_ten_countries, palette="coolw")
plt.title("Top Ten Countries with Total Population (2016)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1358906815.py:4: FutureWarning:

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

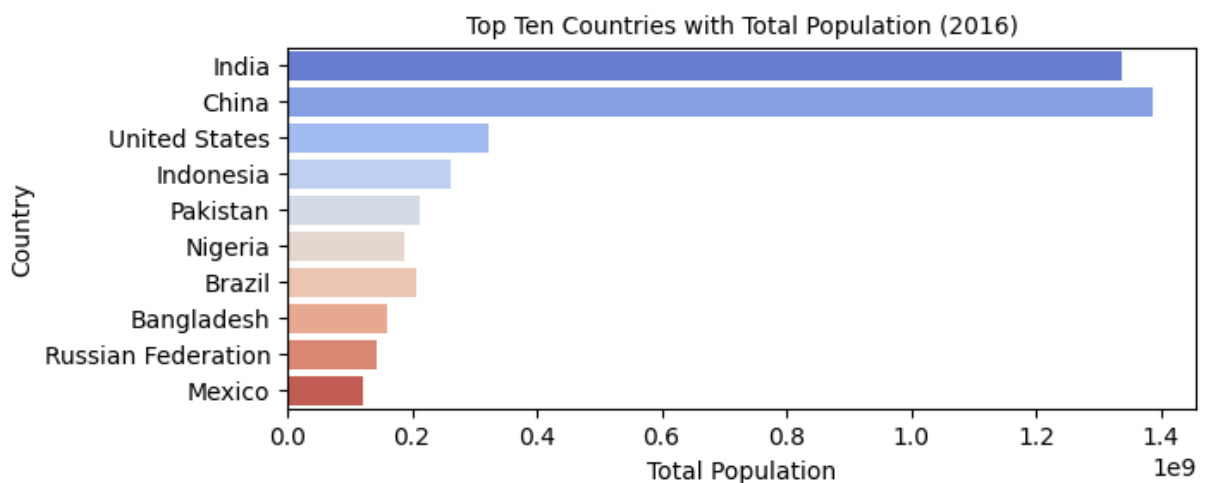
```
sns.barplot(x="2022", y="Country Name", data=total_top_ten_countries, palette="coolw")
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1358906815.py:12: FutureWarning:

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

```
sns.barplot(x="2016", y="Country Name", data=total_top_ten_countries, palette="coolwarm")
```



Top ten countries of total population in year 2010 and 2001

```
In [19]: # Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2010", y="Country Name", data=total_top_ten_countries, palette="coolwarm")
plt.title("Top Ten Countries of Total Population (2010)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()

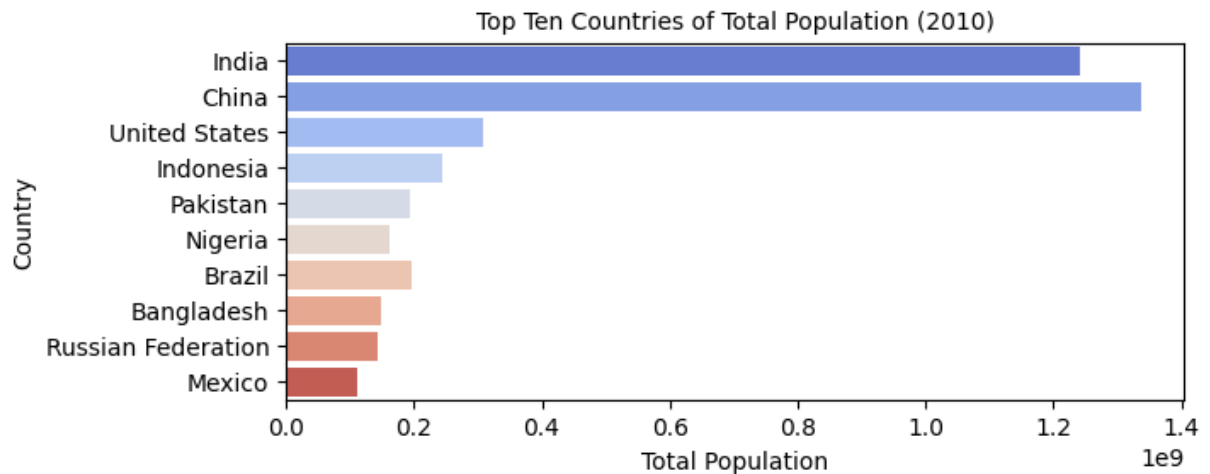
plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2001", y="Country Name", data=total_top_ten_countries, palette="coolwarm")
plt.title("Top Ten Countries with Total Population (2001)", fontsize=10)
```

```
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1609751976.py:4: FutureWarning:

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

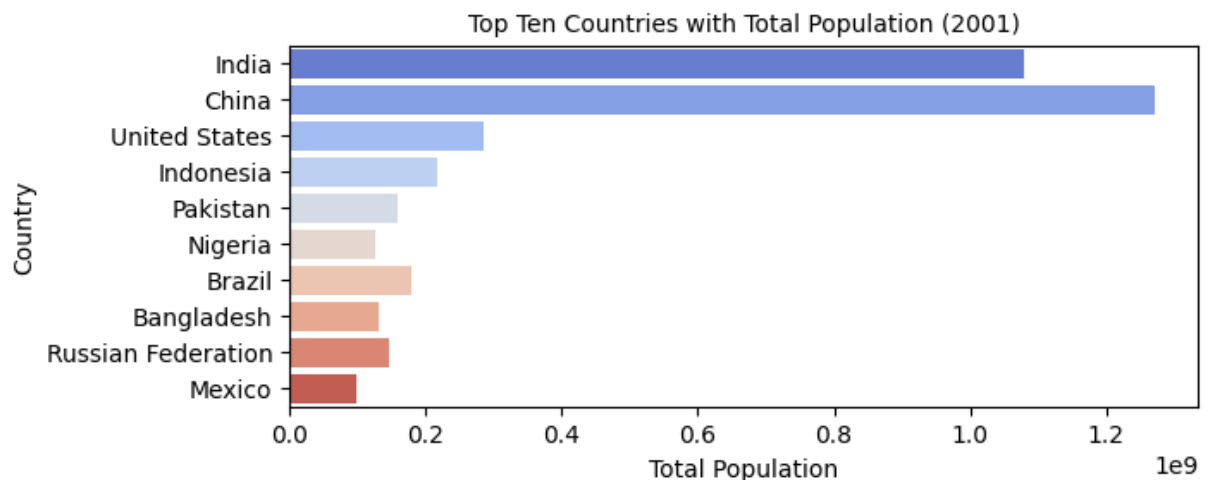
```
sns.barplot(x="2010", y="Country Name", data=total_top_ten_countries, palette="coolwarm")
```



C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1609751976.py:12: FutureWarning:

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

```
sns.barplot(x="2001", y="Country Name", data=total_top_ten_countries, palette="coolwarm")
```



Extraction of bottom-10 countries with respect to total population

```
In [20]: # Sort data based on the total population for 2022
total_population_sorted1 = total_population_data.sort_values(by="2022", ascending=True)
```

```
# Get the bottom ten countries with the highest total population for 2022
total_bottom_ten_countries = total_population_sorted1.head(10)
print("Bottom ten countries of total population\n")
print(total_bottom_ten_countries[['Country Name']] )
```

Bottom ten countries of total population

	Country Name
201	Tuvalu
137	Nauru
150	Palau
27	British Virgin Islands
183	St. Martin (French part)
75	Gibraltar
164	San Marino
130	Monaco
114	Liechtenstein
124	Marshall Islands

Bottom ten countries of total population in year 2022 and 2016

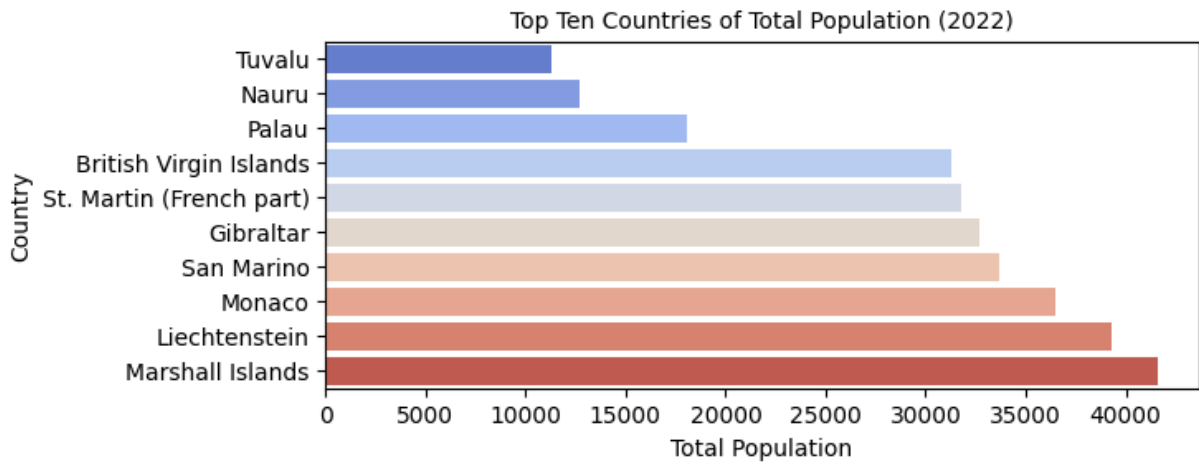
```
In [21]: # Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2022", y="Country Name", data=total_bottom_ten_countries, palette="coolwarm")
plt.title("Top Ten Countries of Total Population (2022)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()

plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2016", y="Country Name", data=total_bottom_ten_countries, palette="coolwarm")
plt.title("Top Ten Countries with Total Population (2016)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\2378637957.py:4: FutureWarning:

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

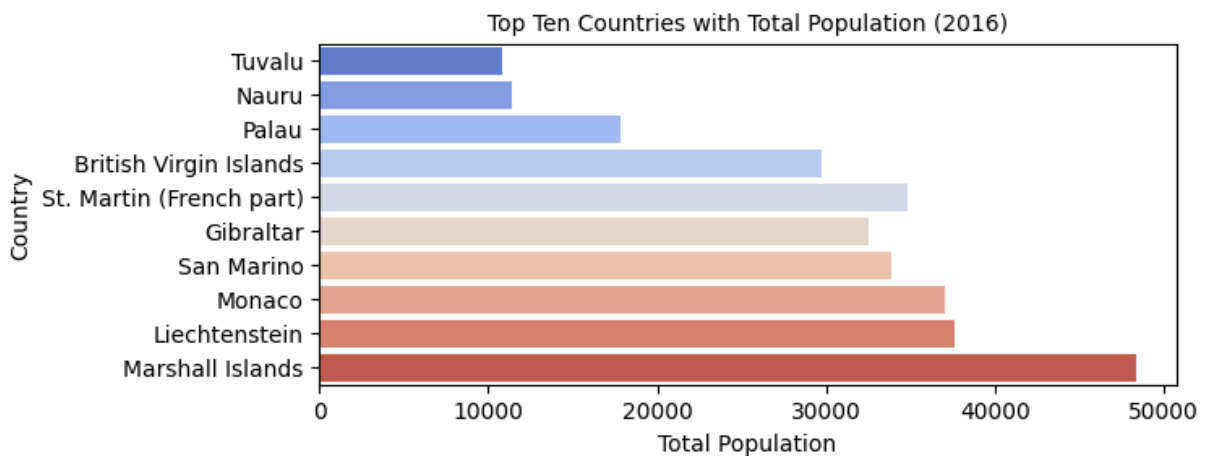
```
sns.barplot(x="2022", y="Country Name", data=total_bottom_ten_countries, palette="coolwarm")
```



C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\2378637957.py:12: FutureWarning:

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

```
sns.barplot(x="2016", y="Country Name", data=total_bottom_ten_countries, palette="colorwarm")
```



Bottom ten countries of total population in year 2010 and 2001

```
In [22]: # Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2010", y="Country Name", data=total_bottom_ten_countries, palette="colorwarm")
plt.title("Top Ten Countries of Total Population (2010)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
plt.ylabel("Country", fontsize=10)
plt.show()

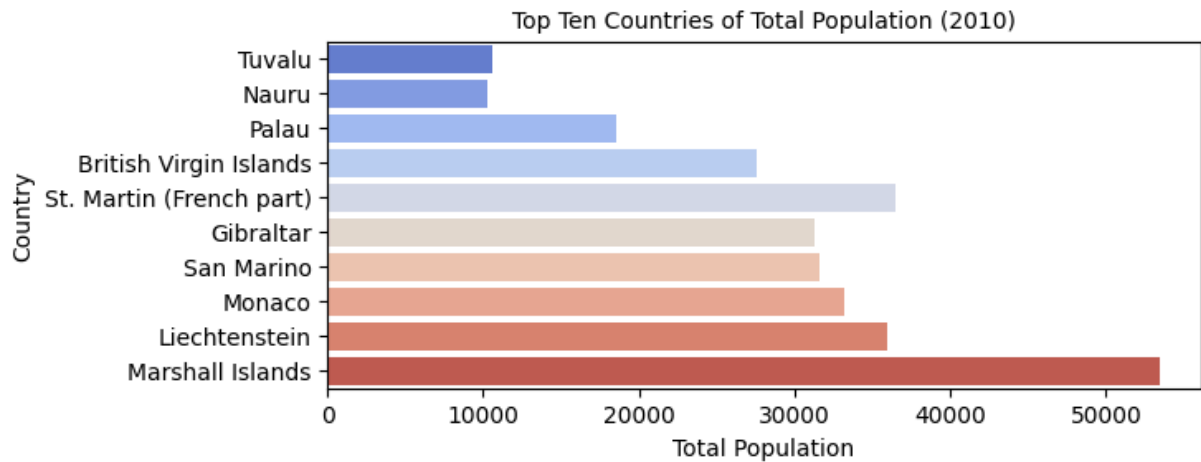
plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2001", y="Country Name", data=total_bottom_ten_countries, palette="colorwarm")
plt.title("Top Ten Countries with Total Population (2001)", fontsize=10)
plt.xlabel("Total Population", fontsize=10)
```

```
plt.ylabel("Country", fontsize=10)
plt.show()
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1516624802.py:4: FutureWarning:

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

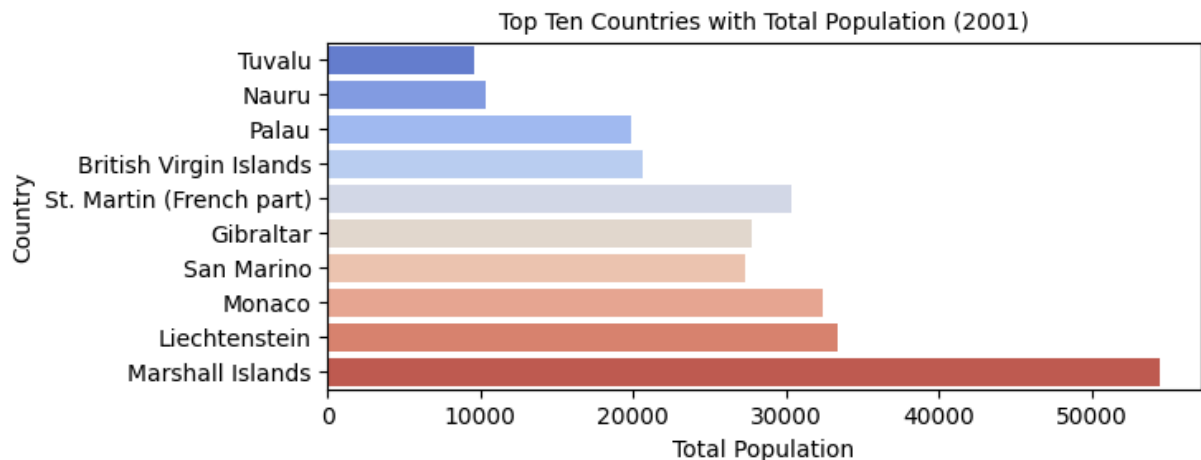
```
sns.barplot(x="2010", y="Country Name", data=total_bottom_ten_countries, palette="coolwarm")
```



C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1516624802.py:12: FutureWarning:

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

```
sns.barplot(x="2001", y="Country Name", data=total_bottom_ten_countries, palette="coolwarm")
```



Extraction of top ten countries with highest male population

```
In [23]: # Filter data for male population
male_population_data = df[df["Series Code"] == "SP.POP.TOTL.MA.IN"]

# Sort data based on the male population for 2022
```

```
male_population_sorted = male_population_data.sort_values(by="2022", ascending=False)

# Get the top ten countries with the highest male population for 2022
male_top_ten_countries = male_population_sorted.head(10)
print("Top ten countries of male population")
print(male_top_ten_countries[['Country Name']])
```

Top ten countries of male population

	Country Name
523	India
475	China
640	United States
524	Indonesia
583	Pakistan
578	Nigeria
460	Brazil
449	Bangladesh
595	Russian Federation
561	Mexico

Extraction of top ten countries with highest female population

```
In [24]: # Filter data for female population
female_population_data = df[df["Series Code"] == "SP.POP.TOTL.FE.IN"]

# Sort data based on the female population for 2022
female_population_sorted = female_population_data.sort_values(by="2022", ascending=False)

# Get the top ten countries with the highest female population for 2022
female_top_ten_countries = female_population_sorted.head(10)
print("Top ten countries of female population")
print(female_top_ten_countries[['Country Name']])
```

Top ten countries of female population

	Country Name
258	China
306	India
423	United States
307	Indonesia
366	Pakistan
243	Brazil
361	Nigeria
232	Bangladesh
378	Russian Federation
344	Mexico

Top ten countries with highest male and female population in 2022

```
In [25]: # Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
```

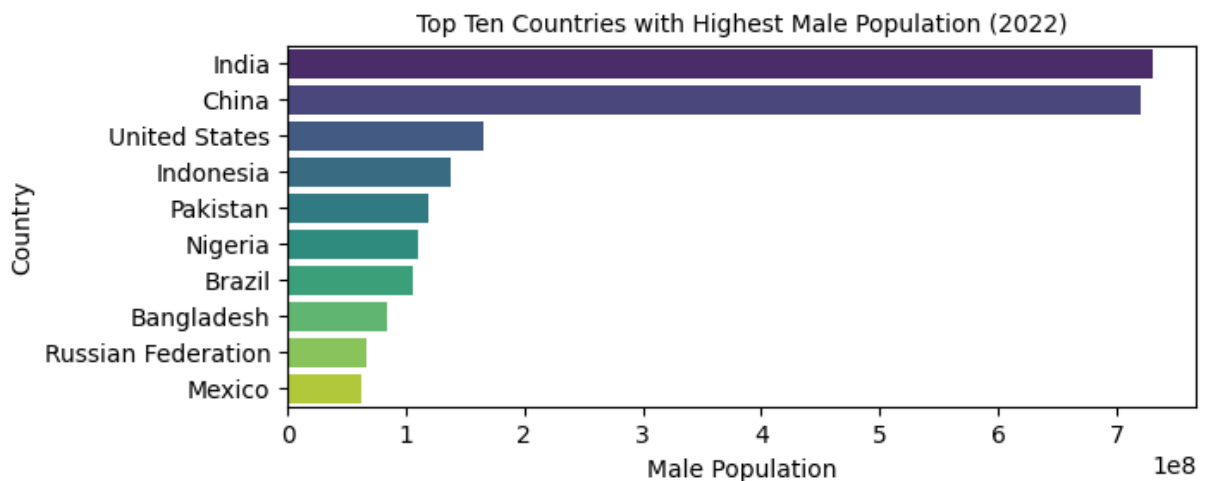
```
sns.barplot(x="2022", y="Country Name", data=male_top_ten_countries, palette="viridis")
plt.title("Top Ten Countries with Highest Male Population (2022)",size=10)
plt.xlabel("Male Population",size=10)
plt.ylabel("Country",size=10)
plt.show()

plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2022", y="Country Name", data=female_top_ten_countries, palette="viridis")
plt.title("Top Ten Countries with Highest Female Population (2022)",size=10)
plt.xlabel("Female Population",size=10)
plt.ylabel("Country",size=10)
plt.show()
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\2901527560.py:4: FutureWarning:

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

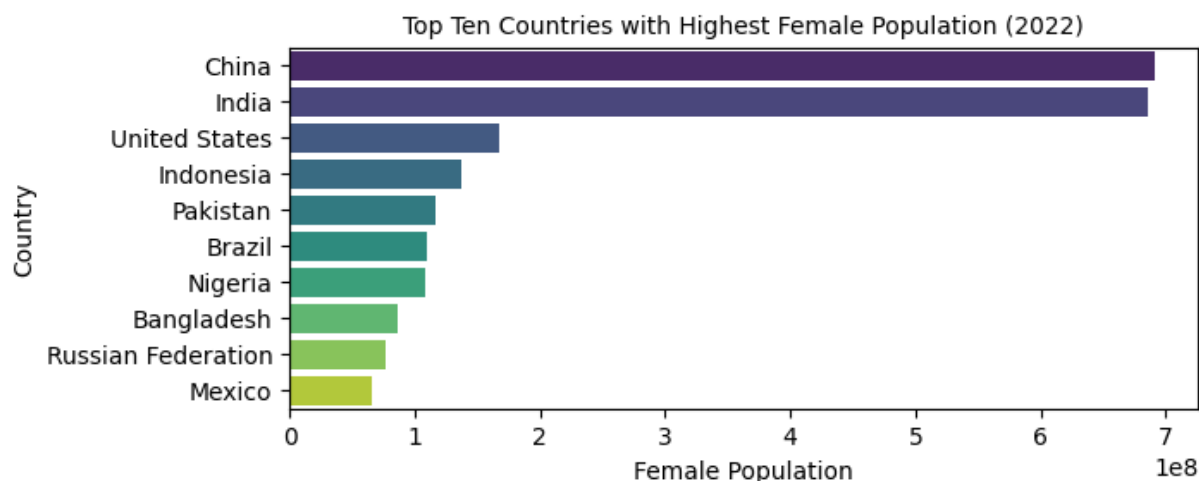
```
sns.barplot(x="2022", y="Country Name", data=male_top_ten_countries, palette="viridis")
```



C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\2901527560.py:12: FutureWarning:

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

```
sns.barplot(x="2022", y="Country Name", data=female_top_ten_countries, palette="viridis")
```



Extraction of top ten countries with lowest male population

```
In [26]: male_lowest_ten_countries = male_population_sorted.tail(10)
print("Top ten countries of lowest male population")
print(male_lowest_ten_countries[['Country Name']])
```

Top ten countries of lowest male population

	Country Name
558	Marshall Islands
548	Liechtenstein
564	Monaco
598	San Marino
509	Gibraltar
617	St. Martin (French part)
461	British Virgin Islands
584	Palau
571	Nauru
635	Tuvalu

Extraction of top ten countries with lowest female population

```
In [27]: female_lowest_ten_countries = female_population_sorted.tail(10)
print("Top ten countries of lowest female population")
print(female_lowest_ten_countries[['Country Name']])
```


Top ten countries of lowest female population

	Country Name
389	Sint Maarten (Dutch part)
331	Liechtenstein
347	Monaco
381	San Marino
400	St. Martin (French part)
292	Gibraltar
244	British Virgin Islands
367	Palau
354	Nauru
418	Tuvalu

Top ten countries with lowest male and female population in 2022

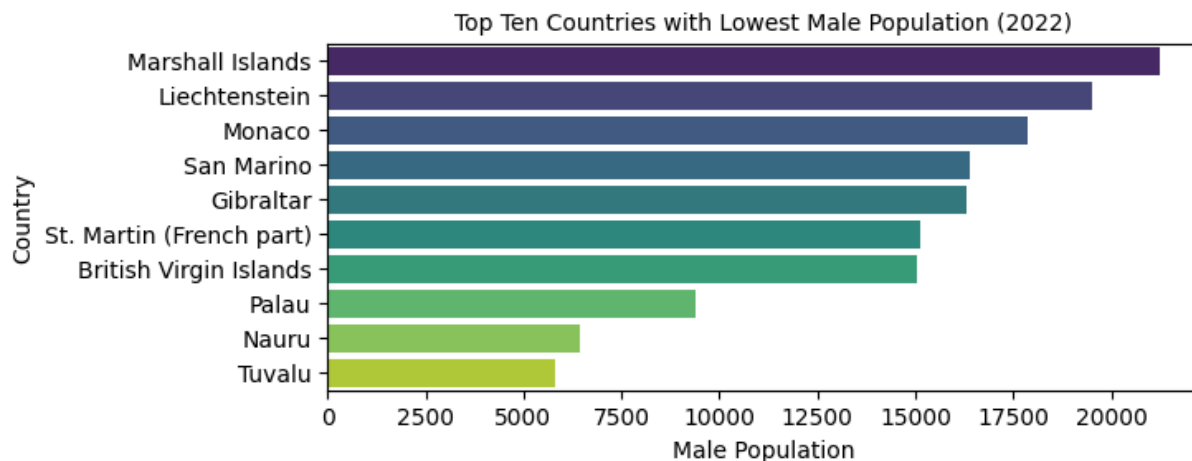
```
In [28]: # Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2022", y="Country Name", data=male_lowest_ten_countries, palette="viridis")
plt.title("Top Ten Countries with Lowest Male Population (2022)",size=10)
plt.xlabel("Male Population",size=10)
plt.ylabel("Country",size=10)
plt.show()

plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2022", y="Country Name", data=female_lowest_ten_countries, palette="viridis")
plt.title("Top Ten Countries with Lowest Female Population (2022)",size=10)
plt.xlabel("Female Population",size=10)
plt.ylabel("Country",size=10)
plt.show()
```

C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1602059660.py:4: FutureWarning:

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

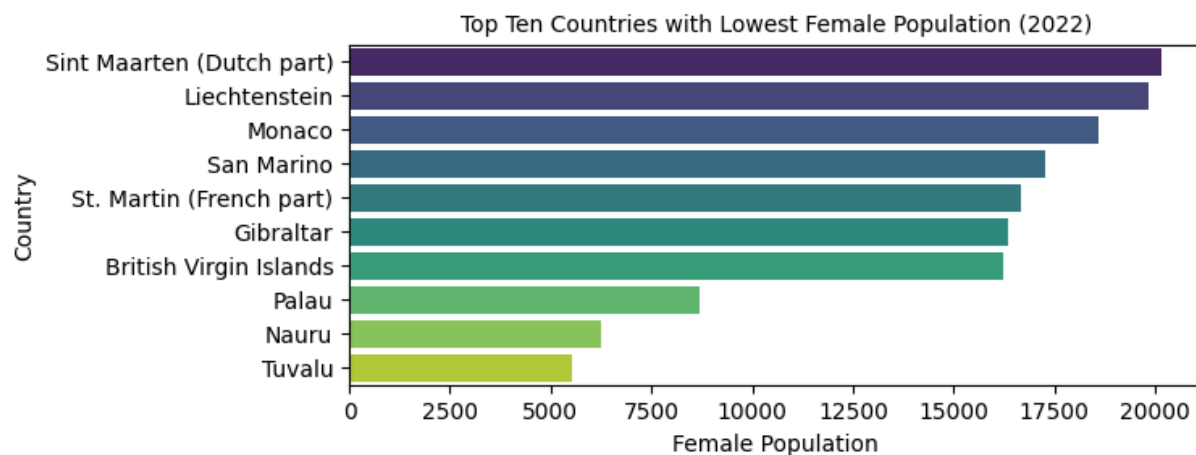
```
sns.barplot(x="2022", y="Country Name", data=male_lowest_ten_countries, palette="viridis")
```



C:\Users\Soumaya\AppData\Local\Temp\ipykernel_996\1602059660.py:12: FutureWarning:

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

```
sns.barplot(x="2022", y="Country Name", data=female_lowest_ten_countries, palette="v
iridis")
```



Stacked Bar Plot

Top 10 Countries with Male and Female Populations (2022)

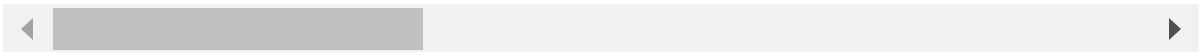
```
In [29]: # Merge male and female population data on 'Country Name'
merged_data = pd.merge(male_population_data, female_population_data, on="Country Name")
```

```
In [30]: merged_data
```

Out[30]:

	Series	Code_male	Country Name	2022_male	2021_male	2020_male	2019_male	2018_
0	SP.POP.TOTL.MA.IN		Afghanistan	20766442.0	20254878.0	19692301.0	19090409.0	18549
1	SP.POP.TOTL.MA.IN		Albania	1384548.0	1404454.0	1419264.0	1428828.0	1435
2	SP.POP.TOTL.MA.IN		Algeria	22862237.0	22497244.0	22132899.0	21756903.0	21362
3	SP.POP.TOTL.MA.IN		American Samoa	21873.0	22289.0	22921.0	23535.0	24
4	SP.POP.TOTL.MA.IN		Andorra	40786.0	40361.0	39615.0	38842.0	38
...
212	SP.POP.TOTL.MA.IN		Virgin Islands (U.S.)	49137.0	49510.0	49866.0	50196.0	50
213	SP.POP.TOTL.MA.IN		West Bank and Gaza	2516444.0	2455361.0	2394860.0	2334948.0	2275
214	SP.POP.TOTL.MA.IN		Yemen, Rep.	17023203.0	16668432.0	16320979.0	15953578.0	15578
215	SP.POP.TOTL.MA.IN		Zambia	9877642.0	9609004.0	9338613.0	9066397.0	8794
216	SP.POP.TOTL.MA.IN		Zimbabwe	7705601.0	7543690.0	7385220.0	7231989.0	7086

217 rows × 47 columns



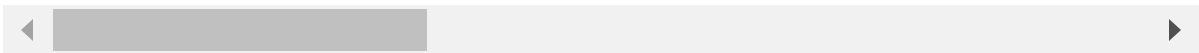
```
In [31]: # Calculate the total population for each country (male + female)
merged_data["Total Population"] = merged_data["2022_male"] + merged_data["2022_female"]
```

```
In [32]: merged_data.head()
```

Out[32]:

	Series	Code_male	Country Name	2022_male	2021_male	2020_male	2019_male	2018_ma
0	SP.POP.TOTL.MA.IN		Afghanistan	20766442.0	20254878.0	19692301.0	19090409.0	1854986
1	SP.POP.TOTL.MA.IN		Albania	1384548.0	1404454.0	1419264.0	1428828.0	143588
2	SP.POP.TOTL.MA.IN		Algeria	22862237.0	22497244.0	22132899.0	21756903.0	2136260
3	SP.POP.TOTL.MA.IN		American Samoa	21873.0	22289.0	22921.0	23535.0	2413
4	SP.POP.TOTL.MA.IN		Andorra	40786.0	40361.0	39615.0	38842.0	3807

5 rows × 48 columns

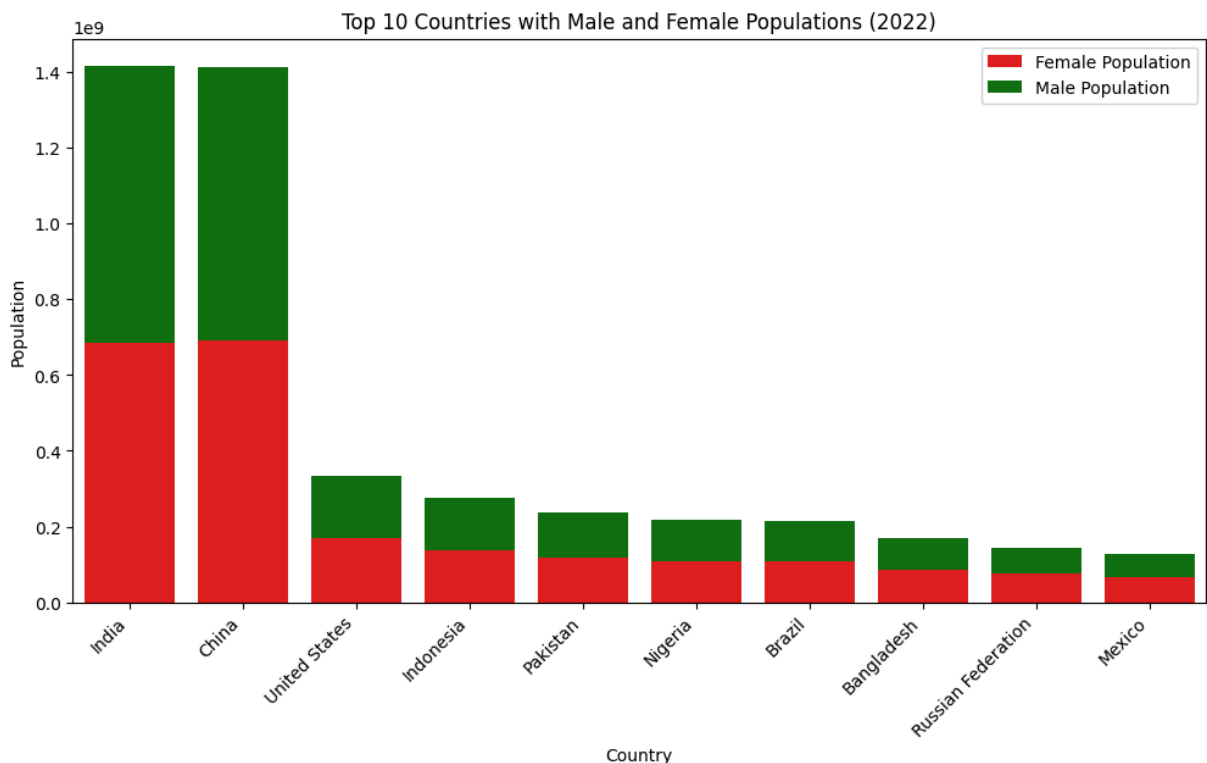


```
In [33]: # Sort data based on total population in descending order
sorted_data = merged_data.sort_values(by="Total Population", ascending=False)

# Select the top 10 countries with the highest total population
top_10_countries = sorted_data.head(10)
```

```
In [34]: # Create the stacked bar plot
plt.figure(figsize=(12, 6))

sns.barplot(x="Country Name", y="2022_female", data=top_10_countries, color="red", label="Female Population")
sns.barplot(x="Country Name", y="2022_male", data=top_10_countries, bottom=top_10_countries["2022_female"], color="green", label="Male Population")
plt.title("Top 10 Countries with Male and Female Populations (2022)")
plt.xlabel("Country")
plt.ylabel("Population")
plt.legend()
plt.xticks(rotation=45, ha="right")
plt.show()
```



Bottom 10 Countries with Male and Female Populations (2022)

```
In [35]: # Select the top 10 countries with the highest total population
bottom_10_countries = sorted_data.tail(10)
```

```
In [36]: # Create the stacked bar plot
plt.figure(figsize=(12, 6))

sns.barplot(x="Country Name", y="2022_female", data=bottom_10_countries, color="red", label="Female Population")
sns.barplot(x="Country Name", y="2022_male", data=bottom_10_countries, bottom=bottom_10_countries["2022_female"], color="green", label="Male Population")
```

```
plt.title("Bottom 10 Countries with Male and Female Populations (2022)")
plt.xlabel("Country")
plt.ylabel("Population")
plt.legend()
plt.xticks(rotation=45, ha="right")
plt.show()
```

