INPUT

```
from google.colab import drive
drive.mount('/content/drive')
```

INPUT

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

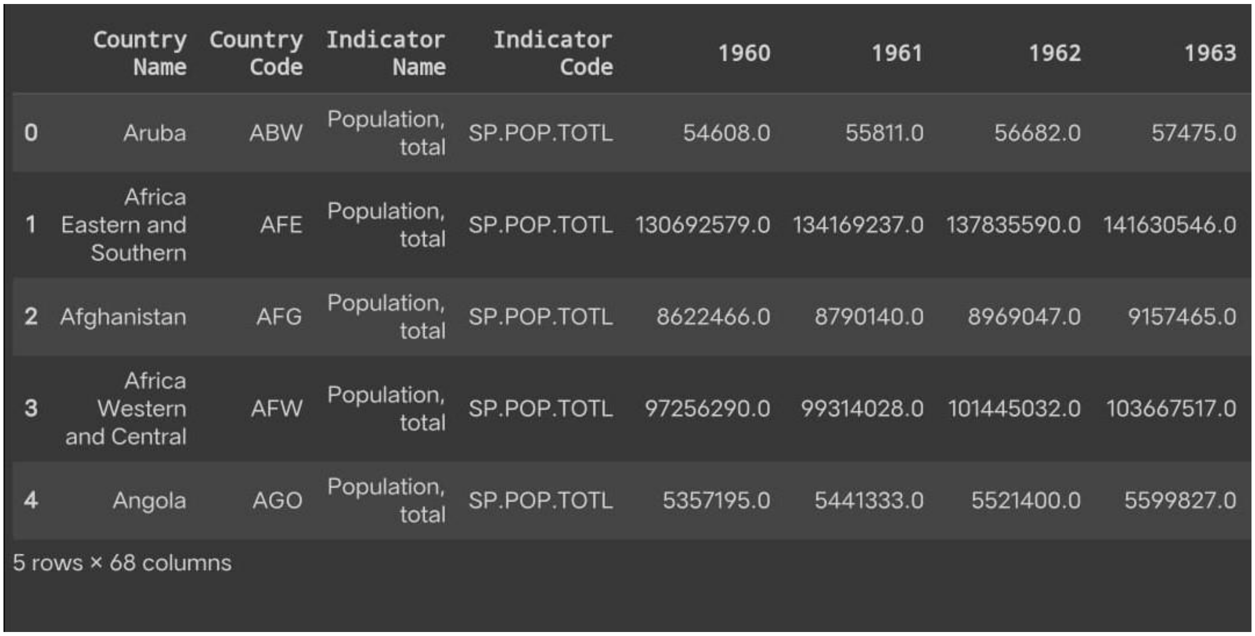
INPUT

```
df=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/World_population_data.csv"
```

INPUT

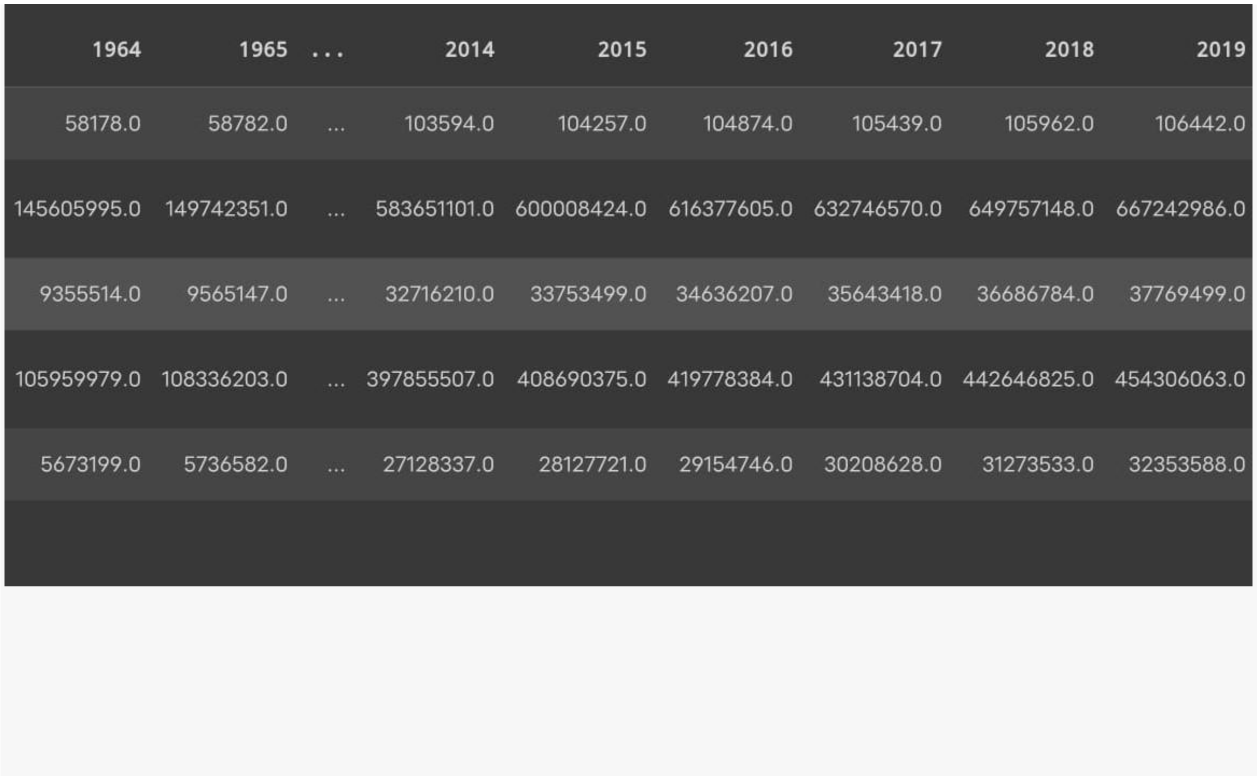
```
df.head(5)
```

OUTPUT

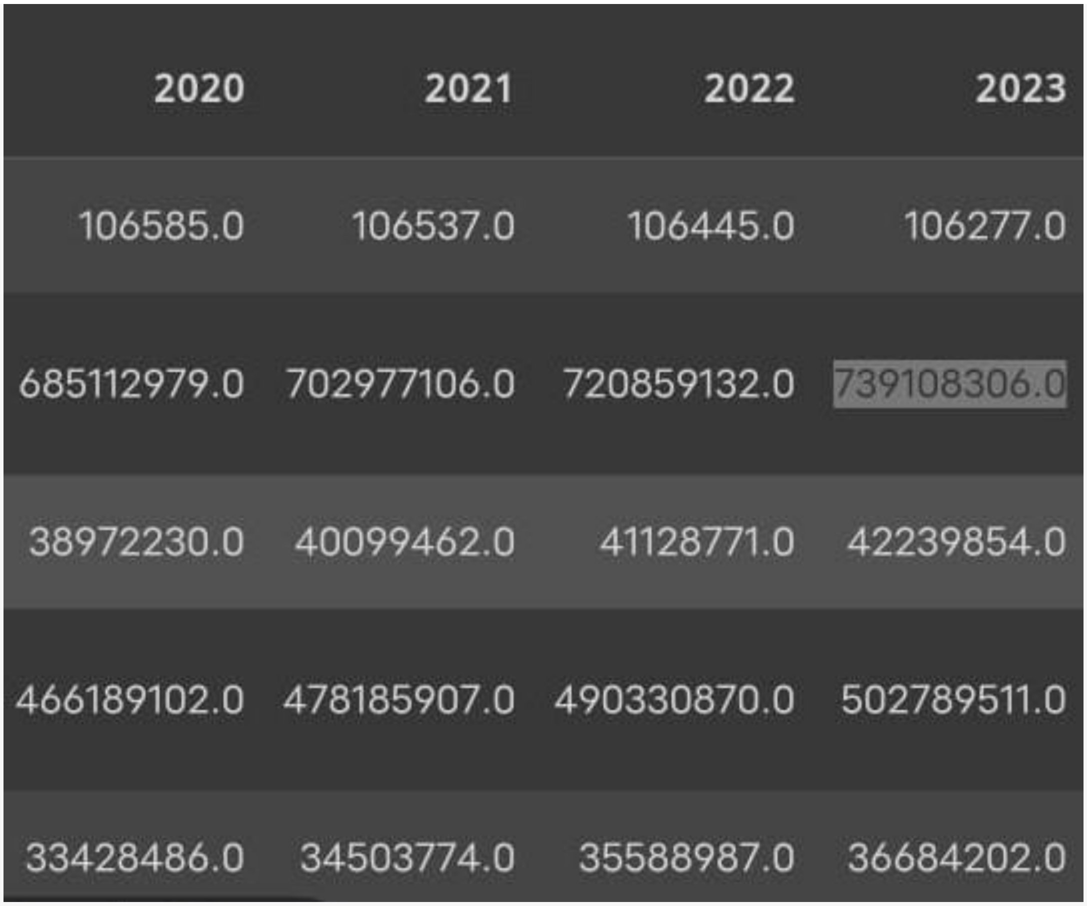


	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963
0	Aruba	ABW	Population, total	SP.POP.TOTL	54608.0	55811.0	56682.0	57475.0
1	Africa Eastern and Southern	AFE	Population, total	SP.POP.TOTL	130692579.0	134169237.0	137835590.0	141630546.0
2	Afghanistan	AFG	Population, total	SP.POP.TOTL	8622466.0	8790140.0	8969047.0	9157465.0
3	Africa Western and Central	AFW	Population, total	SP.POP.TOTL	97256290.0	99314028.0	101445032.0	103667517.0
4	Angola	AGO	Population, total	SP.POP.TOTL	5357195.0	5441333.0	5521400.0	5599827.0

5 rows × 68 columns



1964	1965	...	2014	2015	2016	2017	2018	2019
58178.0	58782.0	...	103594.0	104257.0	104874.0	105439.0	105962.0	106442.0
145605995.0	149742351.0	...	583651101.0	600008424.0	616377605.0	632746570.0	649757148.0	667242986.0
9355514.0	9565147.0	...	32716210.0	33753499.0	34636207.0	35643418.0	36686784.0	37769499.0
105959979.0	108336203.0	...	397855507.0	408690375.0	419778384.0	431138704.0	442646825.0	454306063.0
5673199.0	5736582.0	...	27128337.0	28127721.0	29154746.0	30208628.0	31273533.0	32353588.0

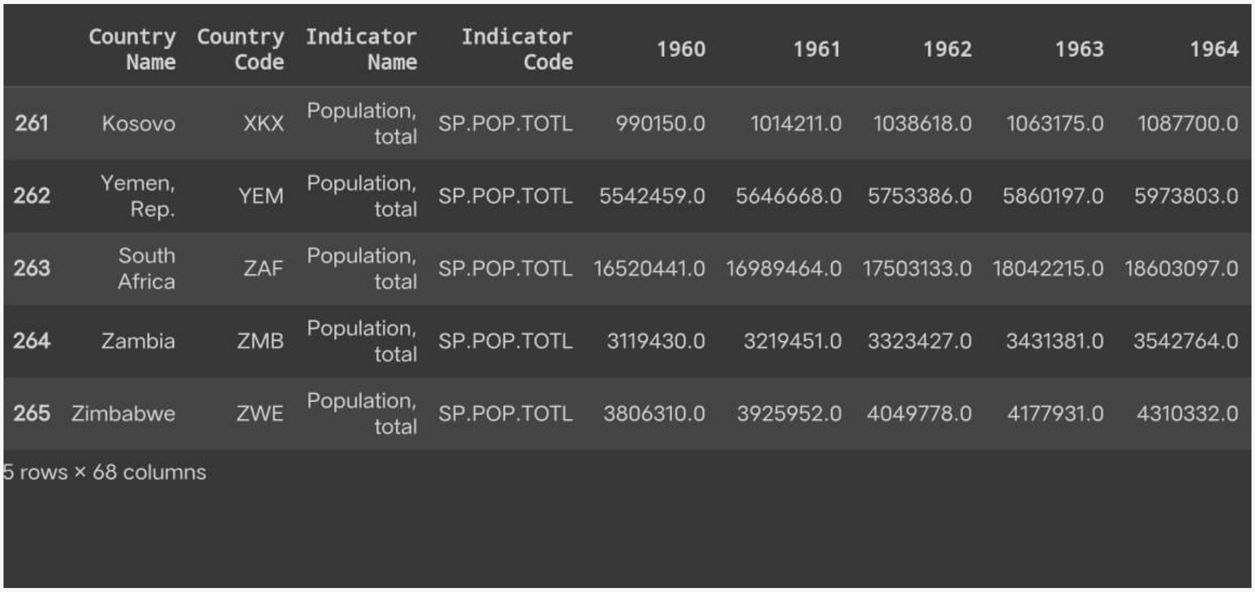


2020	2021	2022	2023
106585.0	106537.0	106445.0	106277.0
685112979.0	702977106.0	720859132.0	739108306.0
38972230.0	40099462.0	41128771.0	42239854.0
466189102.0	478185907.0	490330870.0	502789511.0
33428486.0	34503774.0	35588987.0	36684202.0

INPUT

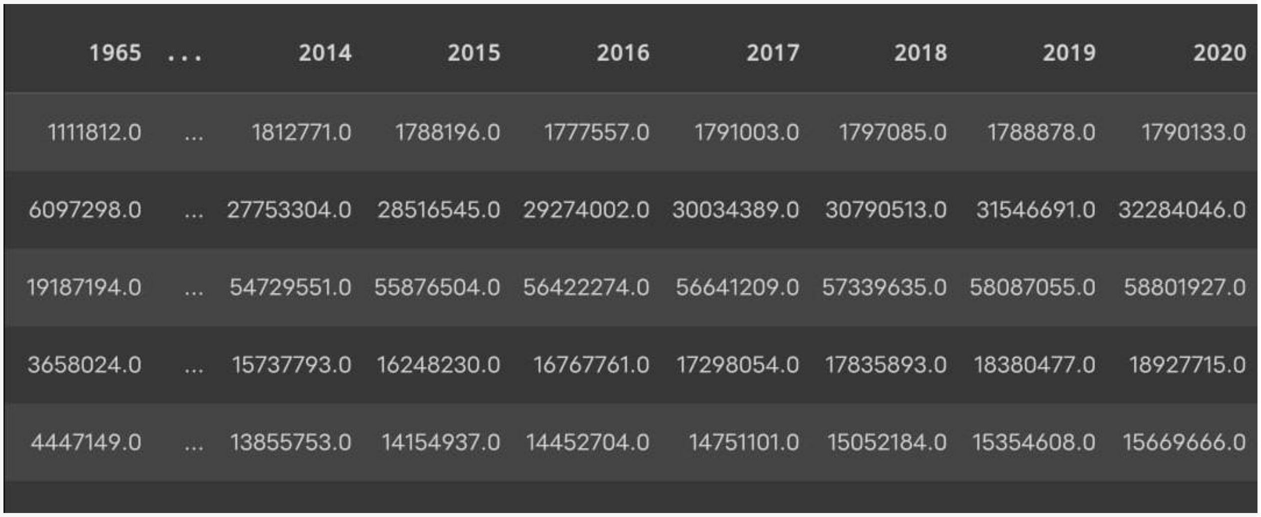
```
df.tail(5)
```

OUTPUT

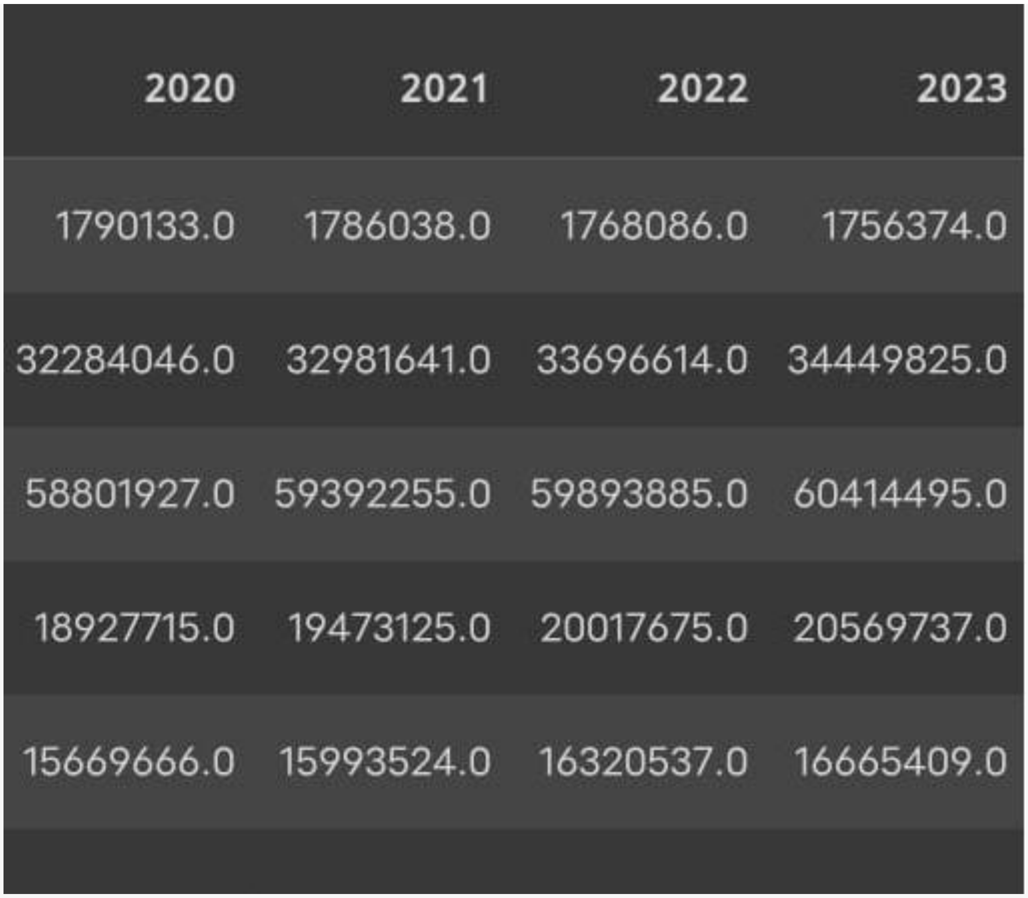


	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964
261	Kosovo	XKX	Population, total	SP.POP.TOTL	990150.0	1014211.0	1038618.0	1063175.0	1087700.0
262	Yemen, Rep.	YEM	Population, total	SP.POP.TOTL	5542459.0	5646668.0	5753386.0	5860197.0	5973803.0
263	South Africa	ZAF	Population, total	SP.POP.TOTL	16520441.0	16989464.0	17503133.0	18042215.0	18603097.0
264	Zambia	ZMB	Population, total	SP.POP.TOTL	3119430.0	3219451.0	3323427.0	3431381.0	3542764.0
265	Zimbabwe	ZWE	Population, total	SP.POP.TOTL	3806310.0	3925952.0	4049778.0	4177931.0	4310332.0

5 rows × 68 columns



1965	...	2014	2015	2016	2017	2018	2019	2020
1111812.0	...	1812771.0	1788196.0	1777557.0	1791003.0	1797085.0	1788878.0	1790133.0
6097298.0	...	27753304.0	28516545.0	29274002.0	30034389.0	30790513.0	31546691.0	32284046.0
19187194.0	...	54729551.0	55876504.0	56422274.0	56641209.0	57339635.0	58087055.0	58801927.0
3658024.0	...	15737793.0	16248230.0	16767761.0	17298054.0	17835893.0	18380477.0	18927715.0
4447149.0	...	13855753.0	14154937.0	14452704.0	14751101.0	15052184.0	15354608.0	15669666.0



2020	2021	2022	2023
1790133.0	1786038.0	1768086.0	1756374.0
32284046.0	32981641.0	33696614.0	34449825.0
58801927.0	59392255.0	59893885.0	60414495.0
18927715.0	19473125.0	20017675.0	20569737.0
15669666.0	15993524.0	16320537.0	16665409.0

INPUT

```
df.columns
```

OUTPUT

```
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')
```

INPUT

```
df.info()
```

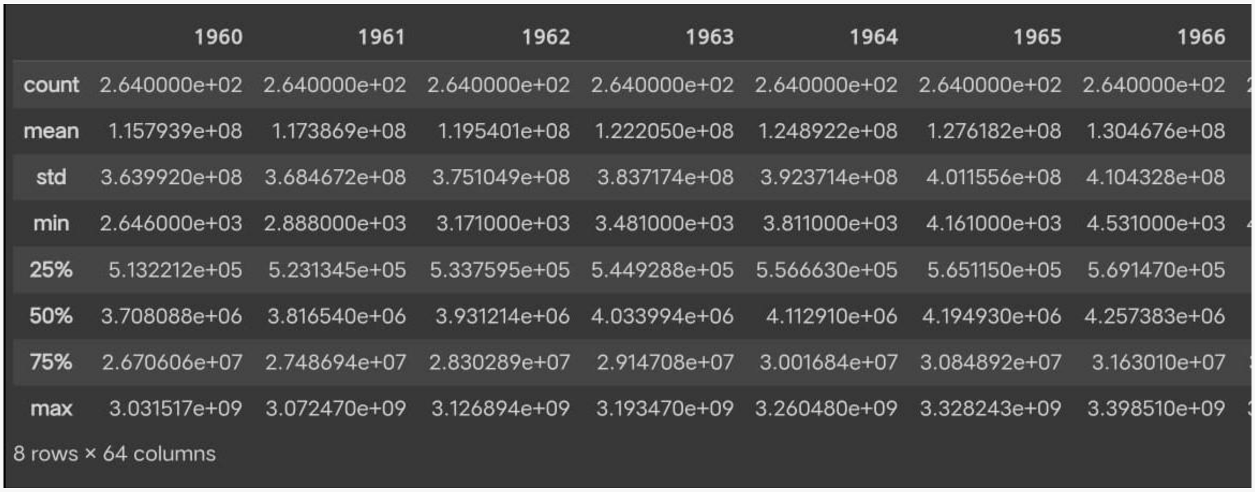
OUTPUT

```
<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
```

INPUT

```
df.describe()
```

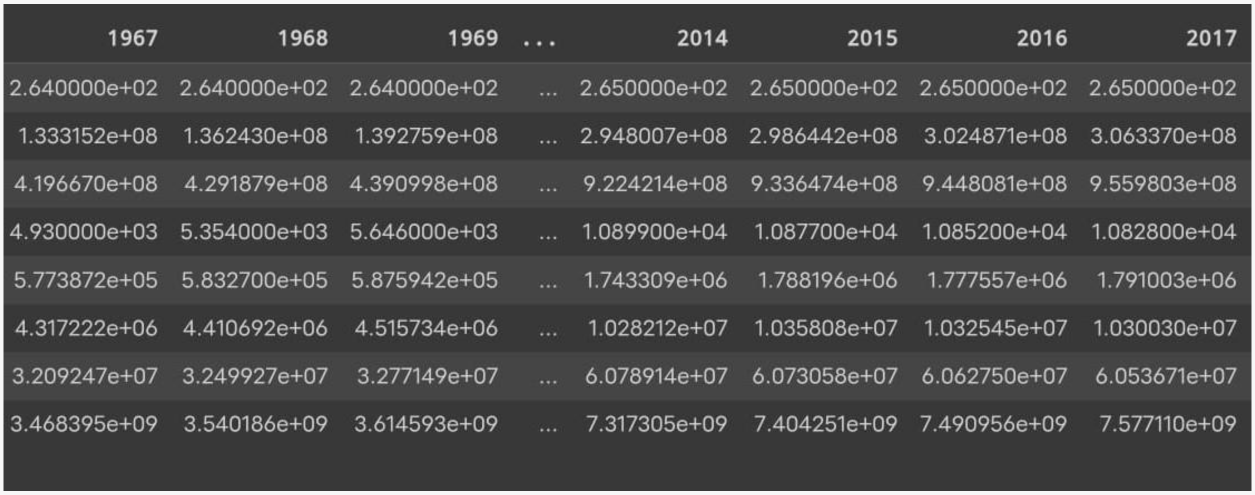
OUTPUT



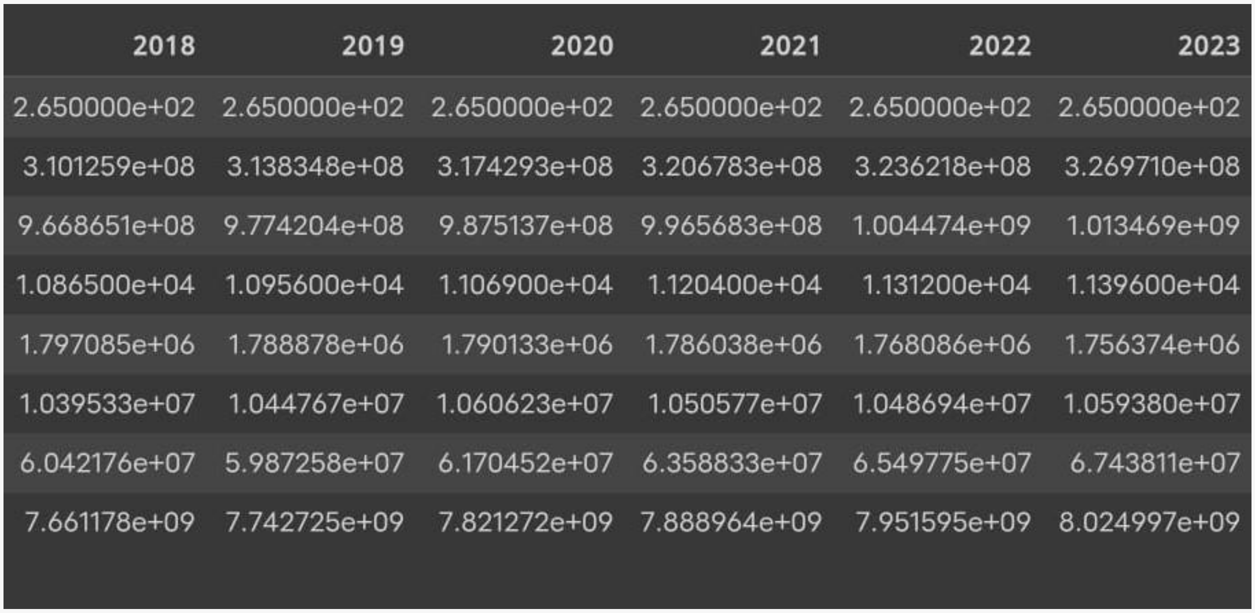
	1960	1961	1962	1963	1964	1965	1966
count	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02	2.640000e+02
mean	1.157939e+08	1.173869e+08	1.195401e+08	1.222050e+08	1.248922e+08	1.276182e+08	1.304676e+08
std	3.639920e+08	3.684672e+08	3.751049e+08	3.837174e+08	3.923714e+08	4.011556e+08	4.104328e+08
min	2.646000e+03	2.888000e+03	3.171000e+03	3.481000e+03	3.811000e+03	4.161000e+03	4.531000e+03
25%	5.132212e+05	5.231345e+05	5.337595e+05	5.449288e+05	5.566630e+05	5.651150e+05	5.691470e+05
50%	3.708088e+06	3.816540e+06	3.931214e+06	4.033994e+06	4.112910e+06	4.194930e+06	4.257383e+06
75%	2.670606e+07	2.748694e+07	2.830289e+07	2.914708e+07	3.001684e+07	3.084892e+07	3.163010e+07
max	3.031517e+09	3.072470e+09	3.126894e+09	3.193470e+09	3.260480e+09	3.328243e+09	3.398510e+09

8 rows × 64 columns

```
0
```



1967	1968	1969	...	2014	2015	2016	2017
2.640000e+02	2.640000e+02	2.640000e+02	...	2.650000e+02	2.650000e+02	2.650000e+02	2.650000e+02
1.333152e+08	1.362430e+08	1.392759e+08	...	2.948007e+08	2.986442e+08	3.024871e+08	3.063370e+08
4.196670e+08	4.291879e+08	4.390998e+08	...	9.224214e+08	9.336474e+08	9.448081e+08	9.559803e+08
4.930000e+03	5.354000e+03	5.646000e+03	...	1.089900e+04	1.087700e+04	1.085200e+04	1.082800e+04
5.773872e+05	5.832700e+05	5.875942e+05	...	1.743309e+06	1.788196e+06	1.777557e+06	1.791003e+06
4.317222e+06	4.410692e+06	4.515734e+06	...	1.028212e+07	1.035808e+07	1.032545e+07	1.030030e+07
3.209247e+07	3.249927e+07	3.277149e+07	...	6.078914e+07	6.073058e+07	6.062750e+07	6.053671e+07
3.468395e+09	3.540186e+09	3.614593e+09	...	7.317305e+09	7.404251e+09	7.490956e+09	7.577110e+09



2018	2019	2020	2021	2022	2023
2.650000e+02	2.650000e+02	2.650000e+02	2.650000e+02	2.650000e+02	2.650000e+02
3.101259e+08	3.138348e+08	3.174293e+08	3.206783e+08	3.236218e+08	3.269710e+08
9.668651e+08	9.774204e+08	9.875137e+08	9.965683e+08	1.004474e+09	1.013469e+09
1.086500e+04	1.095600e+04	1.106900e+04	1.120400e+04	1.131200e+04	1.139600e+04
1.797085e+06	1.788878e+06	1.790133e+06	1.786038e+06	1.768086e+06	1.756374e+06
1.039533e+07	1.044767e+07	1.060623e+07	1.050577e+07	1.048694e+07	1.059380e+07
6.042176e+07	5.987258e+07	6.170452e+07	6.358833e+07	6.549775e+07	6.743811e+07
7.661178e+09	7.742725e+09	7.821272e+09	7.888964e+09	7.951595e+09	8.024997e+09

INPUT

```
df.duplicated().sum()
```

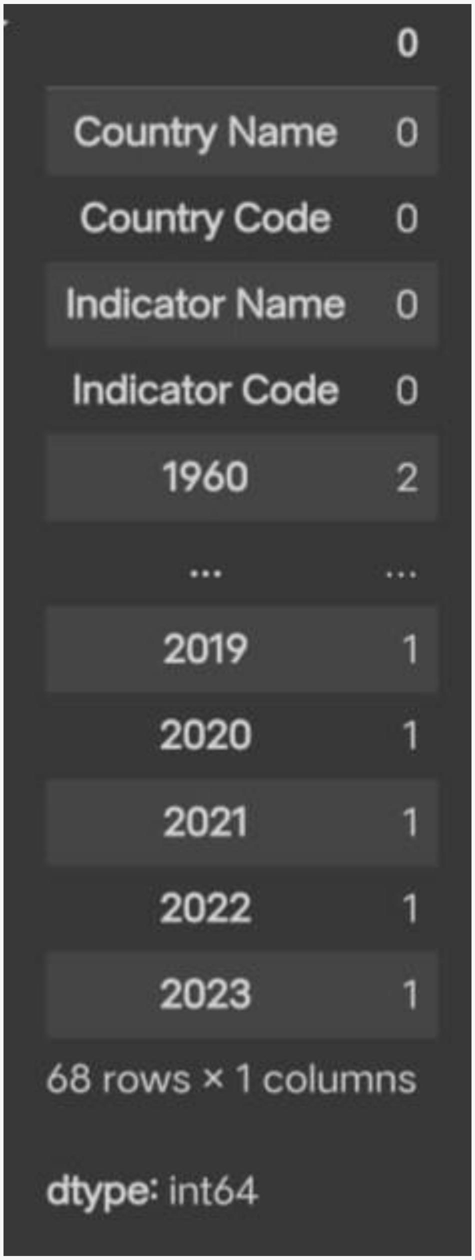
OUTPUT

```
0
```

INPUT

```
df.isna().sum()
```

OUTPUT



```
                    0
  Country Name      0
  Country Code      0
  Indicator Name    0
  Indicator Code    0
      1960          2
       ...        ...
      2019          1
      2020          1
      2021          1
      2022          1
      2023          1
68 rows × 1 columns

dtype: int64
```

INPUT

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

OUTPUT

```
['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']

 Total no of unique countries: 266
```

INPUT

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

OUTPUT

```
['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' 'DJI' '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' 'XKX' 'YEM' 'ZAF'
 'ZMB' 'ZWE']
```

```
 Total no of unique country code: 266
```

INPUT

```python
df.drop(['Indicator Name','Country Name'],axis=1,inplace=True)
```

INPUT

```python
# Filter data for total population
total_population_data = df[df['Indicator 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 Code']]
```

OUTPUT

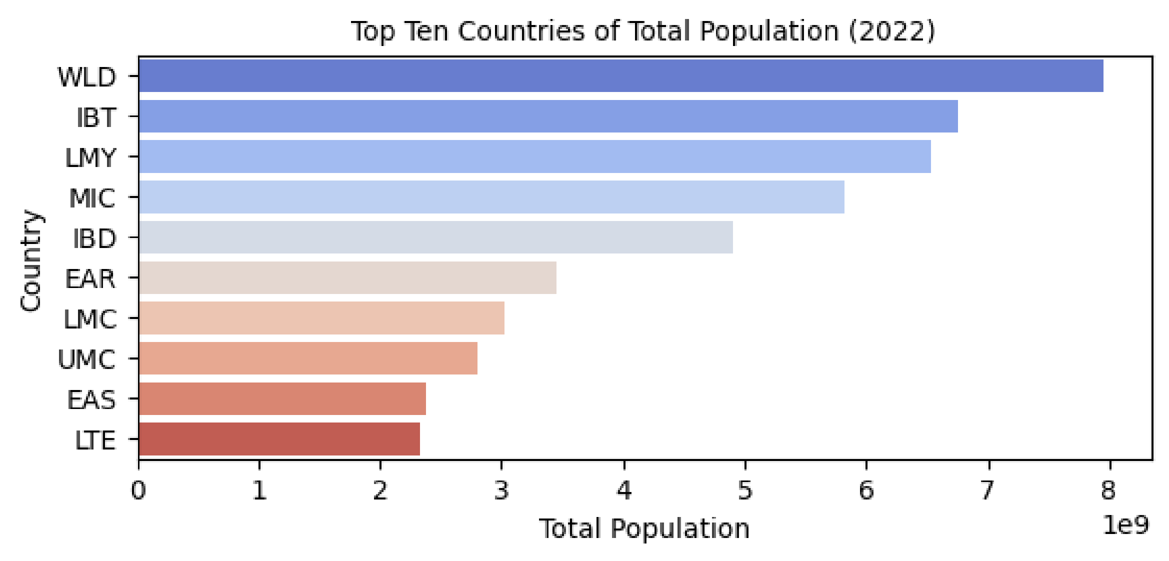
```
Top ten countries of total population

    Country Code
259          WLD
103          IBT
140          LMY
156          MIC
102          IBD
62           EAR
139          LMC
249          UMC
63           EAS
142          LTE
```

INPUT

```
# Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2022", y="Country Code",
data=total_top_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()
```

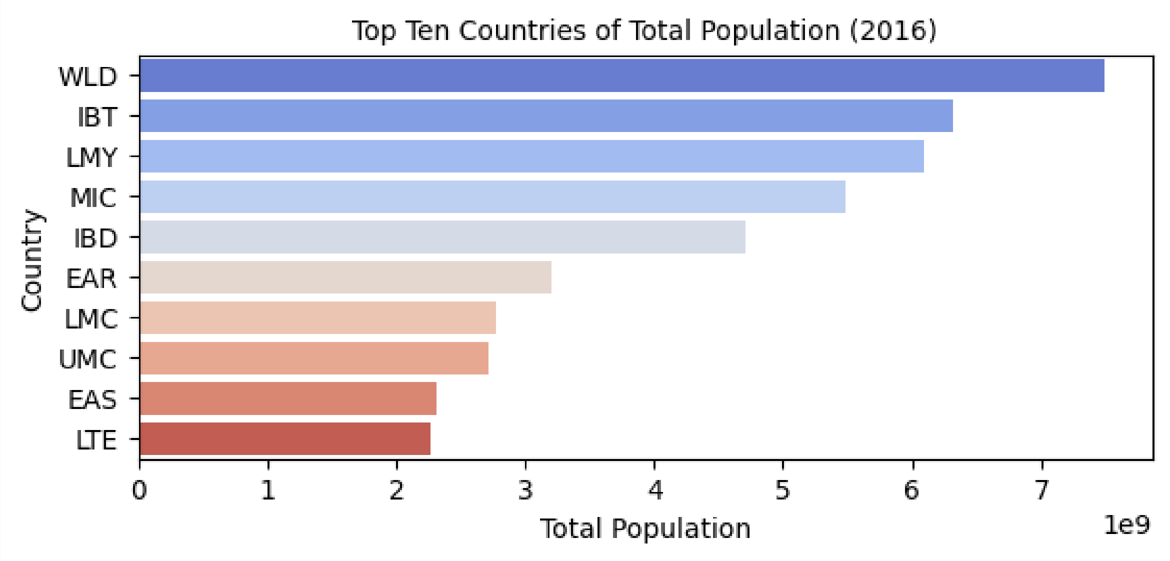
OUTPUT



INPUT

```
# Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2016", y="Country Code",
data=total_top_ten_countries,
palette="coolwarm")
plt.title("Top Ten Countries of Total
Population (2016)",fontsize=10)
plt.xlabel("Total Population",fontsize=10)
plt.ylabel("Country",fontsize=10)
plt.show()
```

OUTPUT



INPUT

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

# Get the top 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
Code']])
```

OUTPUT

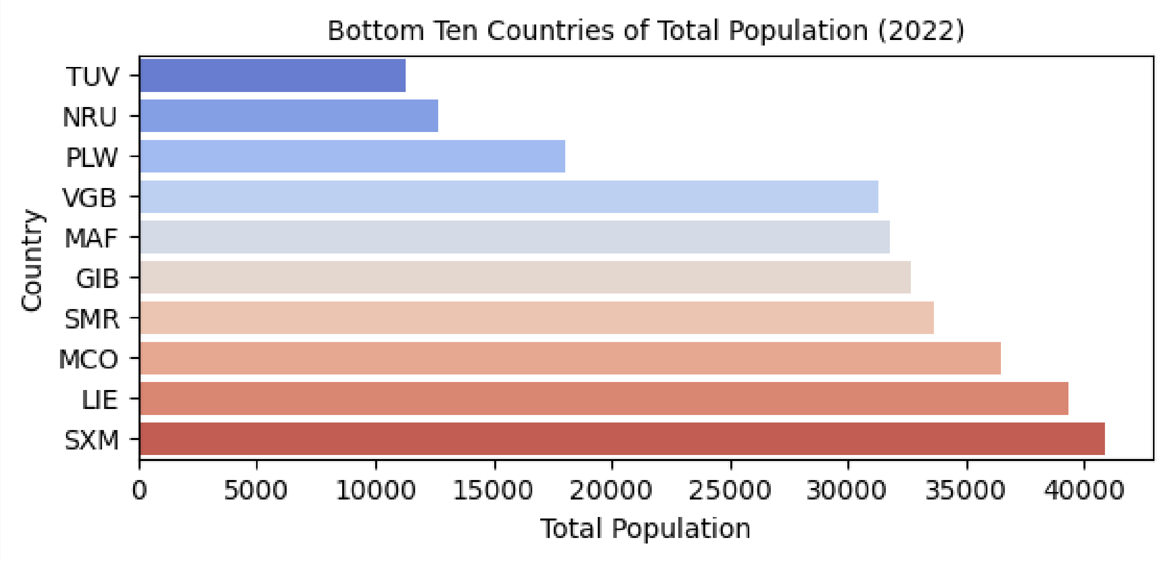
```
Bottom ten countries of total population

    Country Code
245          TUV
179          NRU
188          PLW
255          VGB
147          MAF
84           GIB
212          SMR
149          MCO
137          LIE
225          SXM
```

INPUT

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

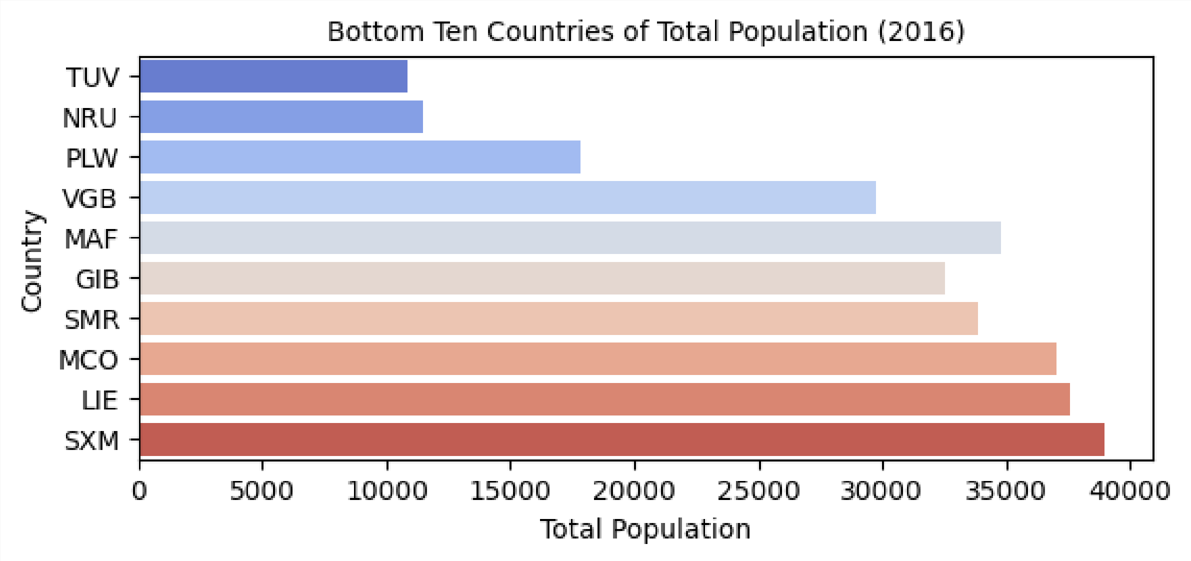
OUTPUT



INPUT

```
# Create the bar plot
plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2016", y="Country Code",
data=total_bottom_ten_countries,
palette="coolwarm")
plt.title("Bottom Ten Countries of Total
Population (2016)",fontsize=10)
plt.xlabel("Total Population",fontsize=10)
plt.ylabel("Country",fontsize=10)
plt.show()
```

OUTPUT



INPUT

```
#filter data for male population
male_population_data = df[df['Indicator
Code']=='SP.POP.TOTL']
male_population_sorted
=male_population_data.sort_values(by="2022"
,ascending=False)
male_top_ten_countries =
male_population_sorted .head(10)
print("Top ten countries of male
population")
print(male_top_ten_countries[['Country
Code']])
```

```
OUTPUT
```

```
Top ten countries of male population
    Country Code
259         WLD
103         IBT
140         LMY
156         MIC
102         IBD
62          EAR
139         LMC
249         UMC
63          EAS
142         LTE
```

INPUT

```
female_population_data = df[df['Indicator Code']=='SP.POP.TOTL']
female_population_sorted=female_population_data.sort_values(by="2022",ascending=False)
female_top_ten_countries=female_population_sorted.head(10)
print("Top ten countries of female population")
print(female_top_ten_countries[['Country Code']])
```

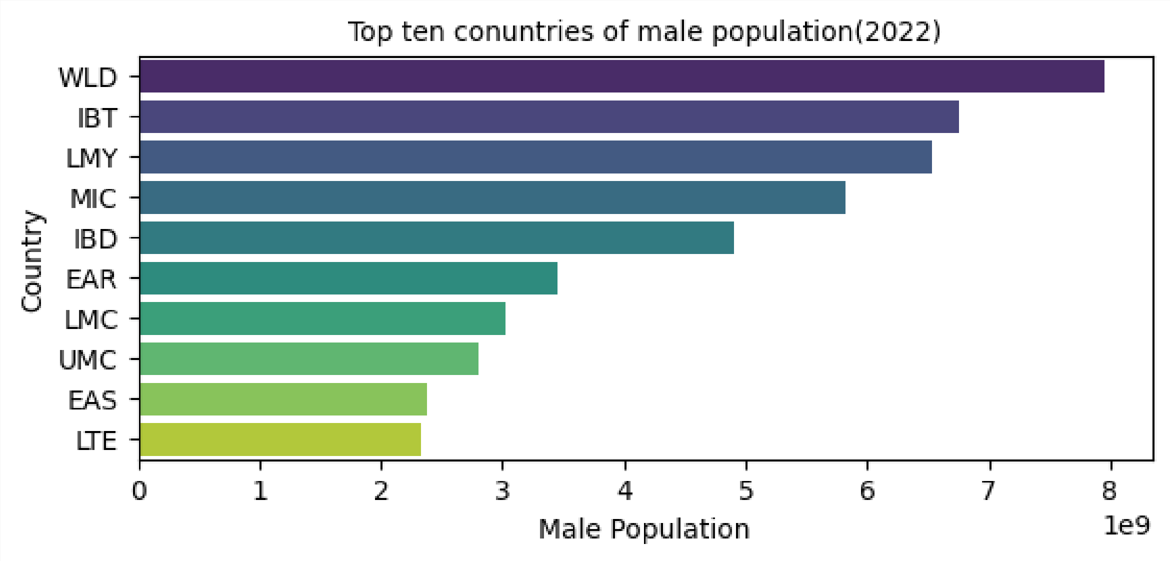
OUTPUT

```
Top ten countries of female population
    Country Code
259          WLD
103          IBT
140          LMY
156          MIC
102          IBD
62           EAR
139          LMC
249          UMC
63           EAS
142          LTE
```

INPUT

```
plt.figure(figsize=(15, 6))
plt.subplot(2,2,1)
sns.barplot(x="2022", y="Country Code",
data=male_top_ten_countries,
palette="viridis")
plt.title("Top ten conuntries of male
population(2022)",fontsize=10)
plt.xlabel("Male Population",fontsize=10)
plt.ylabel("Country",fontsize=10)
plt.show()
```

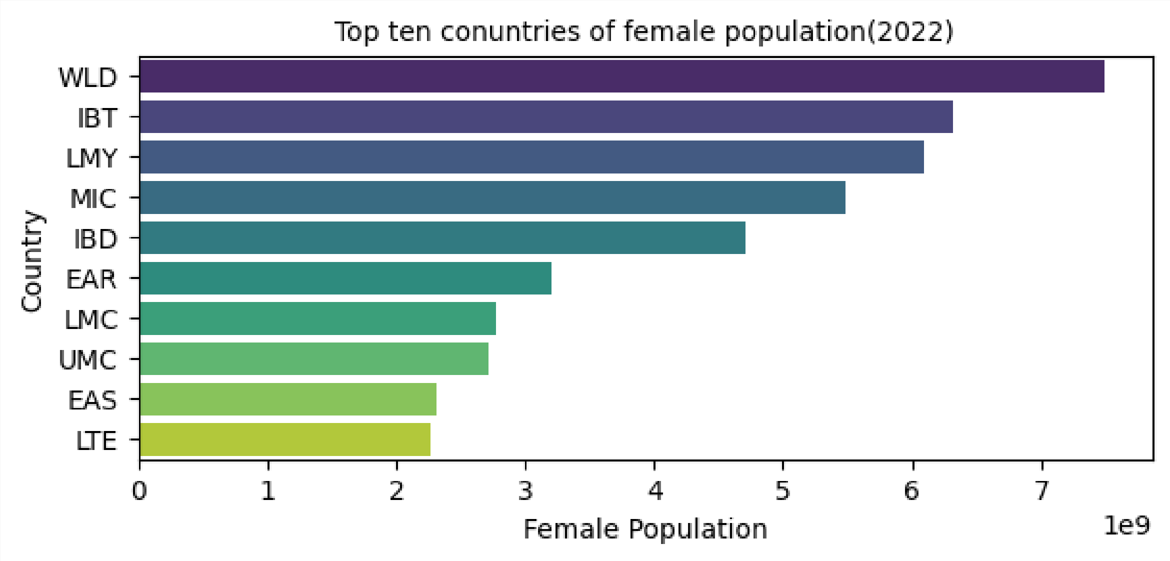
OUTPUT



INPUT

```
plt.figure(figsize=(15, 6))
plt.subplot(2,2,2)
sns.barplot(x="2016", y="Country Code",
data=female_top_ten_countries,
palette="viridis")
plt.title("Top ten conuntries of female
population(2022)",fontsize=10)
plt.xlabel("Female Population",fontsize=10)
plt.ylabel("Country",fontsize=10)
plt.show()
```

OUTPUT



INPUT

```
merge_data=pd.merge(male_population_data,female_population_data,on="Country Code",suffixes=("_male","_female"))
```

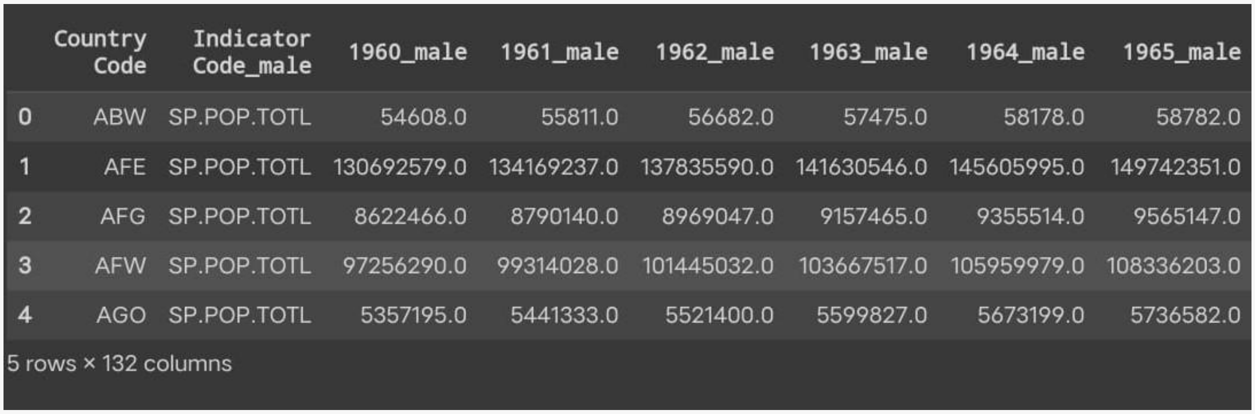
INPUT

```
merge_data["Total population"] = merge_data["2022_male"] + merge_data["2022_female"]
```

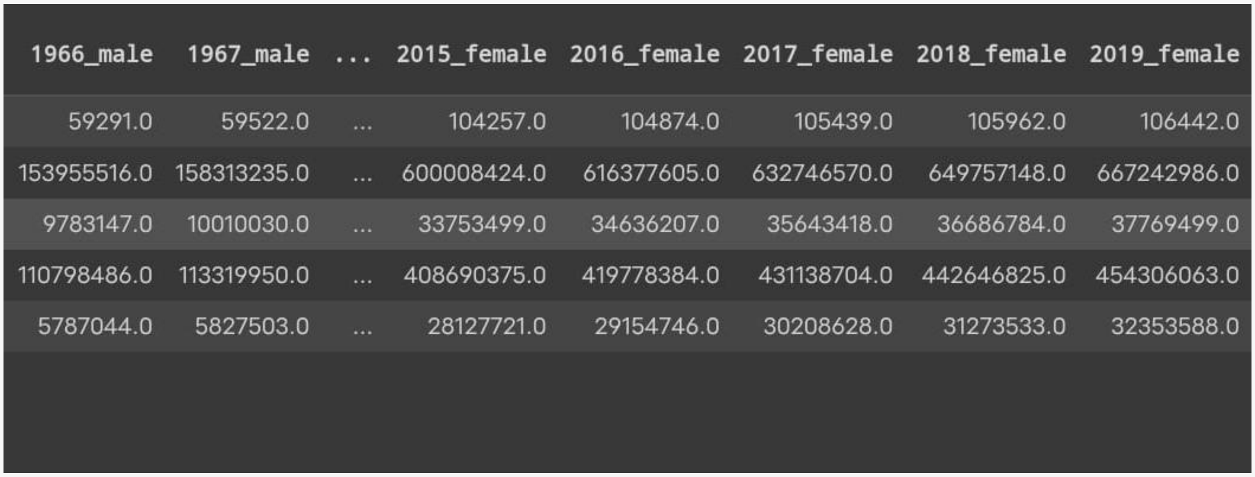
INPUT

```
merge_data.head()
```

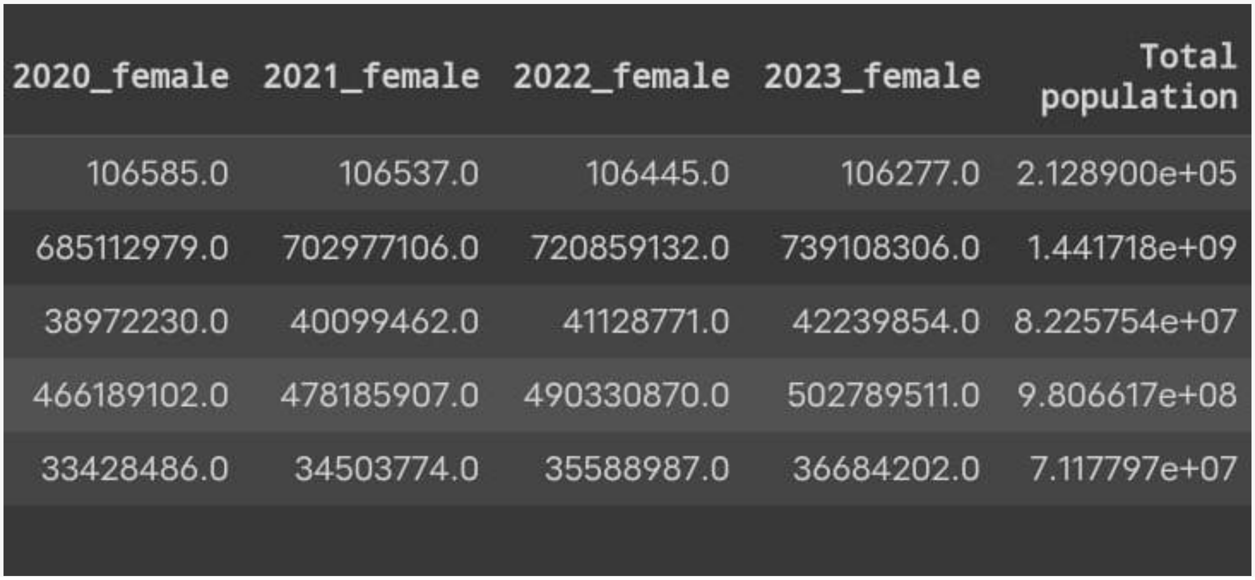
OUTPUT



```
   Country  Indicator
      Code  Code_male    1960_male    1961_male    1962_male    1963_male    1964_male    1965_male
0      ABW  SP.POP.TOTL      54608.0      55811.0      56682.0      57475.0      58178.0      58782.0
1      AFE  SP.POP.TOTL  130692579.0  134169237.0  137835590.0  141630546.0  145605995.0  149742351.0
2      AFG  SP.POP.TOTL    8622466.0    8790140.0    8969047.0    9157465.0    9355514.0    9565147.0
3      AFW  SP.POP.TOTL   97256290.0   99314028.0  101445032.0  103667517.0  105959979.0  108336203.0
4      AGO  SP.POP.TOTL    5357195.0    5441333.0    5521400.0    5599827.0    5673199.0    5736582.0
5 rows × 132 columns
```



```
  1966_male    1967_male  ...  2015_female  2016_female  2017_female  2018_female  2019_female
    59291.0      59522.0  ...     104257.0     104874.0     105439.0     105962.0     106442.0
153955516.0  158313235.0  ...  600008424.0  616377605.0  632746570.0  649757148.0  667242986.0
  9783147.0   10010030.0  ...   33753499.0   34636207.0   35643418.0   36686784.0   37769499.0
110798486.0  113319950.0  ...  408690375.0  419778384.0  431138704.0  442646825.0  454306063.0
  5787044.0    5827503.0  ...   28127721.0   29154746.0   30208628.0   31273533.0   32353588.0
```



```
                                                                    Total
2020_female  2021_female  2022_female  2023_female             population
   106585.0     106537.0     106445.0     106277.0           2.128900e+05
685112979.0  702977106.0  720859132.0  739108306.0           1.441718e+09
 38972230.0   40099462.0   41128771.0   42239854.0           8.225754e+07
466189102.0  478185907.0  490330870.0  502789511.0           9.806617e+08
 33428486.0   34503774.0   35588987.0   36684202.0           7.117797e+07
```

INPUT

```
sorted_data=merge_data.sort_values(by="Total population", ascending=False)
```

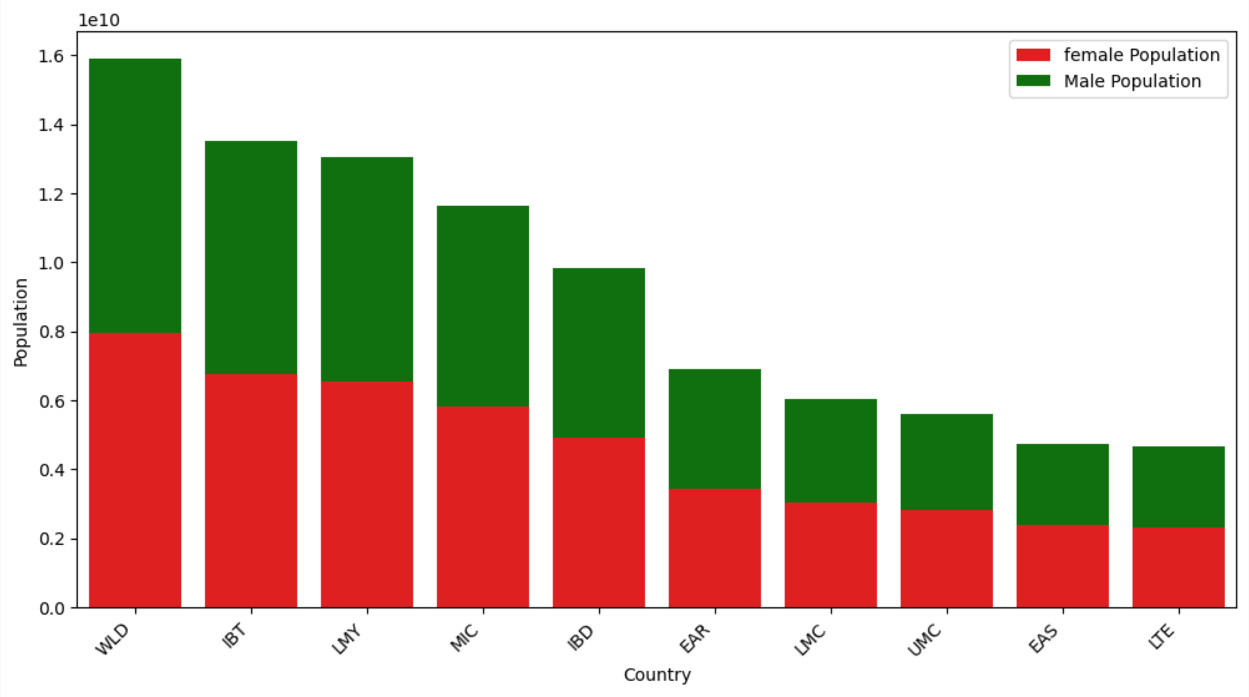
INPUT

```
top_10_countries = sorted_data.head(10)
```

INPUT

```
plt.figure(figsize=(12,6))
sns.barplot(x="CountryCode",y="2022_female",data=top_10_countries,color="red",label="female Population")
sns.barplot(x="CountryCode",y="2022_male",data=top_10_countries,bottom=top_10_countries["2022_female"],color="green",label="Male Population")
plt.xlabel("Country")
plt.ylabel("Population")
plt.legend()
plt.xticks(rotation=45,ha="right")
plt.show()
```

OUTPUT



INPUT

```
bottom_10_countries = sorted_data.tail(10)
```

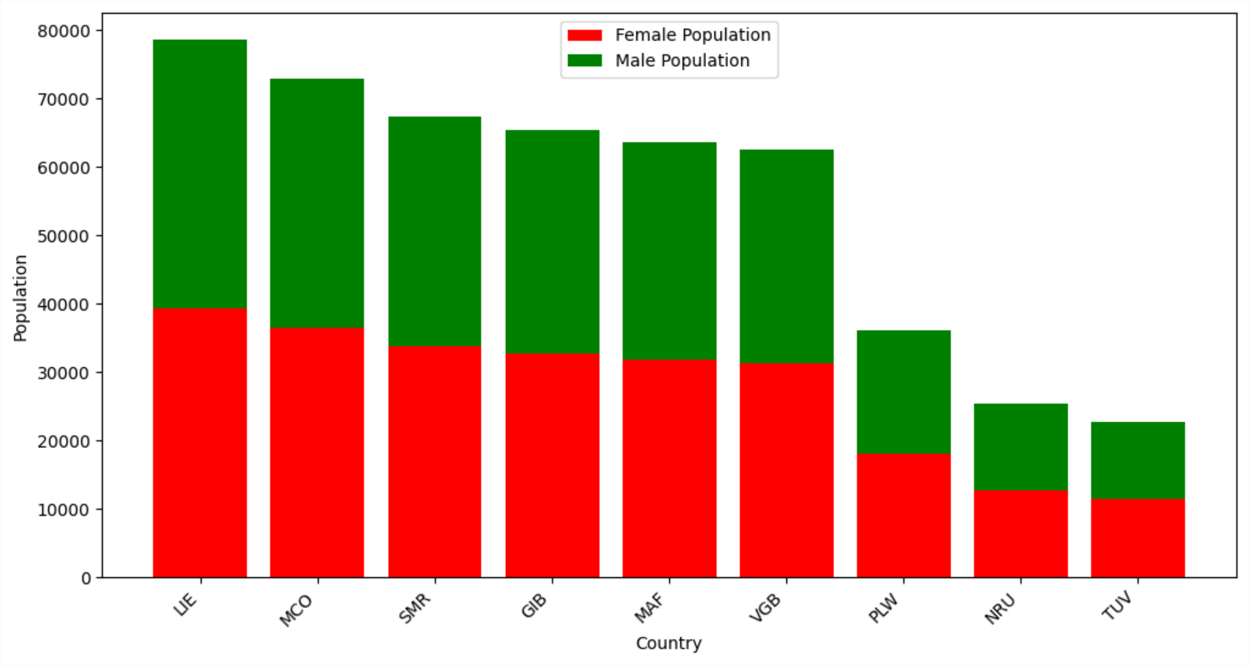
INPUT

```
plt.figure(figsize=(12, 6))
plt.bar(x=bottom_10_countries["Country Code"],height=bottom_10_countries["2022_female"], color="red", label="Female Population")
plt.bar(x=bottom_10_countries["Country Code"],height=bottom_10_countries["2022_male"],bottom=bottom_10_countries["2022_female"], color="green", label="Male Population")
plt.xlabel("Country")
plt.ylabel("Population")
plt.legend()

plt.xticks(rotation=45, ha="right")

plt.show()
```

OUTPUT


🟥 Female Population
🟩 Male Population
Population
LIE
MCO
SMR
GIB
MAF
VGB
PLW
NRU
TUV
Country