In [1]:
```
import pandas as pd
import numpy as np
world_data=pd.read_csv('world_population.csv')
world_data.head()
```

Out[1]:

	Rank	CCA3	Country/Territory	Capital	Continent	2022 Population	2020 Population	2015 Population	2010 Population	F
0	36	AFG	Afghanistan	Kabul	Asia	41128771	38972230	33753499	28189672	
1	138	ALB	Albania	Tirana	Europe	2842321	2866849	2882481	2913399	
2	34	DZA	Algeria	Algiers	Africa	44903225	43451666	39543154	35856344	
3	213	ASM	American Samoa	Pago Pago	Oceania	44273	46189	51368	54849	
4	203	AND	Andorra	Andorra la Vella	Europe	79824	77700	71746	71519	

In [2]:
```
# Checking Missing Values
missing=world_data.isnull().sum()
missing
```

Out[2]:
```
Rank                          0
CCA3                          0
Country/Territory             0
Capital                       0
Continent                     0
2022 Population               0
2020 Population               0
2015 Population               0
2010 Population               0
2000 Population               0
1990 Population               0
1980 Population               0
1970 Population               0
Area (km²)                    0
Density (per km²)             0
Growth Rate                   0
World Population Percentage   0
dtype: int64
```

In [3]:
```
# Checking correct datatypes
world_data.dtypes
```

Out[3]:
```
Rank                             int64
CCA3                            object
Country/Territory               object
Capital                         object
Continent                       object
2022 Population                  int64
2020 Population                  int64
2015 Population                  int64
2010 Population                  int64
2000 Population                  int64
1990 Population                  int64
1980 Population                  int64
1970 Population                  int64
Area (km²)                       int64
Density (per km²)              float64
Growth Rate                    float64
World Population Percentage    float64
dtype: object
```

In [4]:
```
# Checking for duplicates
duplicate=world_data.duplicated()
world_data[duplicate]
```

Out[4]:

Rank	CCA3	Country/Territory	Capital	Continent	2022 Population	2020 Population	2015 Population	2010 Population	Po

Performing Descriptive Analysis

In [6]:
```
round(world_data.describe(),2)
```

Out[6]:

	Rank	2022 Population	2020 Population	2015 Population	2010 Population	2000 Population	1990 Population	P
count	234.00	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+02	2.34
mean	117.50	3.407441e+07	3.350107e+07	3.172996e+07	2.984524e+07	2.626947e+07	2.271022e+07	1.89
std	67.69	1.367664e+08	1.355899e+08	1.304050e+08	1.242185e+08	1.116982e+08	9.783217e+07	8.17
min	1.00	5.100000e+02	5.200000e+02	5.640000e+02	5.960000e+02	6.510000e+02	7.000000e+02	7.33
25%	59.25	4.197385e+05	4.152845e+05	4.046760e+05	3.931490e+05	3.272420e+05	2.641158e+05	2.29
50%	117.50	5.559944e+06	5.493074e+06	5.307400e+06	4.942770e+06	4.292907e+06	3.825410e+06	3.14
75%	175.75	2.247650e+07	2.144798e+07	1.973085e+07	1.915957e+07	1.576230e+07	1.186923e+07	9.82
max	234.00	1.425887e+09	1.424930e+09	1.393715e+09	1.348191e+09	1.264099e+09	1.153704e+09	9.82

In [7]:
```
# Renaming Columns containing years to simple form
world_data.columns = world_data.columns.str.replace(" Population","")
print(world_data.iloc[:,5:13].columns)
```

```
Index(['2022', '2020', '2015', '2010', '2000', '1990', '1980', '1970'], dtype='object')
```

In [8]:
```
round(world_data.describe(),2)
```

Out[8]:

	Rank	2022	2020	2015	2010	2000	1990	
count	234.00	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+02	2.340000e+02	2.34
mean	117.50	3.407441e+07	3.350107e+07	3.172996e+07	2.984524e+07	2.626947e+07	2.271022e+07	1.89
std	67.69	1.367664e+08	1.355899e+08	1.304050e+08	1.242185e+08	1.116982e+08	9.783217e+07	8.17
min	1.00	5.100000e+02	5.200000e+02	5.640000e+02	5.960000e+02	6.510000e+02	7.000000e+02	7.33
25%	59.25	4.197385e+05	4.152845e+05	4.046760e+05	3.931490e+05	3.272420e+05	2.641158e+05	2.29
50%	117.50	5.559944e+06	5.493074e+06	5.307400e+06	4.942770e+06	4.292907e+06	3.825410e+06	3.14
75%	175.75	2.247650e+07	2.144798e+07	1.973085e+07	1.915957e+07	1.576230e+07	1.186923e+07	9.82
max	234.00	1.425887e+09	1.424930e+09	1.393715e+09	1.348191e+09	1.264099e+09	1.153704e+09	9.82

In [9]:
```
# Converting Growth rate to percentage
world_data["Growth Rate"] = (world_data["Growth Rate"] - 1) * 100
world_data[["Country/Territory", "Growth Rate"]].head() # this will show only two columns
```

Out[9]:

	Country/Territory	Growth Rate
0	Afghanistan	2.57
1	Albania	-0.43
2	Algeria	1.64
3	American Samoa	-1.69
4	Andorra	1.00

Exploratory data analysis

In [11]:
```
# Total world population
Total_population= print("Total population of the world is:", world_data["2022"].sum())
```

```
Total population of the world is: 7973413042
```

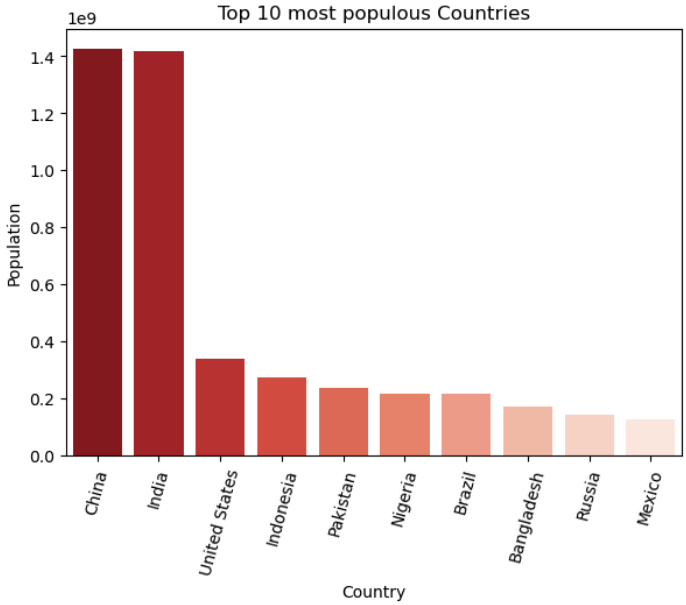
In [12]:
```
# Sort data in ascending order
population_sorted = world_data.sort_values("2022", ascending=False)
population_sorted
```

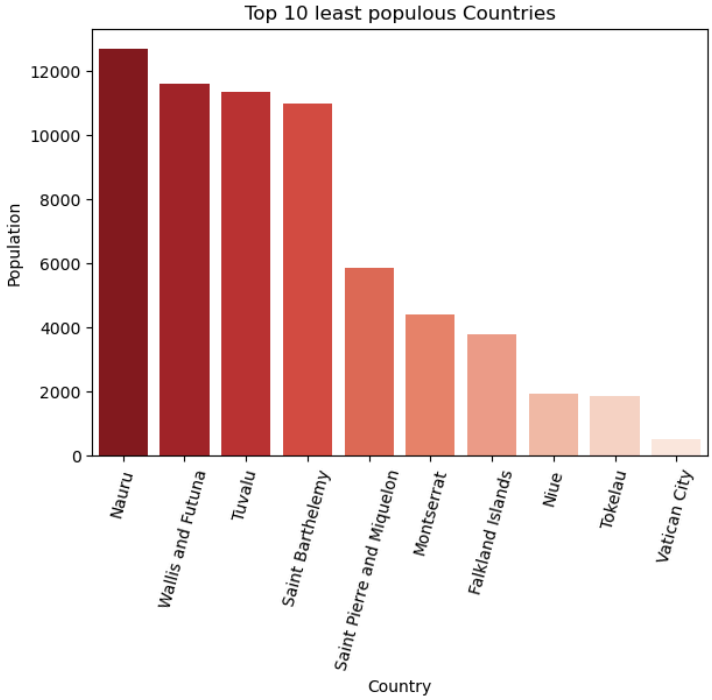
Out[12]:

	Rank	CCA3	Country/Territory	Capital	Continent	2022	2020	2015	
41	1	CHN	China	Beijing	Asia	1425887337	1424929781	1393715448	13481
92	2	IND	India	New Delhi	Asia	1417173173	1396387127	1322866505	12406
221	3	USA	United States	Washington, D.C.	North America	338289857	335942003	324607776	3111
93	4	IDN	Indonesia	Jakarta	Asia	275501339	271857970	259091970	2440
156	5	PAK	Pakistan	Islamabad	Asia	235824862	227196741	210969298	1944
...	...	...	...	...	...	...	...	...	
137	230	MSR	Montserrat	Brades	North America	4390	4500	5059	
64	231	FLK	Falkland Islands	Stanley	South America	3780	3747	3408	
150	232	NIU	Niue	Alofi	Oceania	1934	1942	1847	
209	233	TKL	Tokelau	Nukunonu	Oceania	1871	1827	1454	
226	234	VAT	Vatican City	Vatican City	Europe	510	520	564	

234 rows × 17 columns

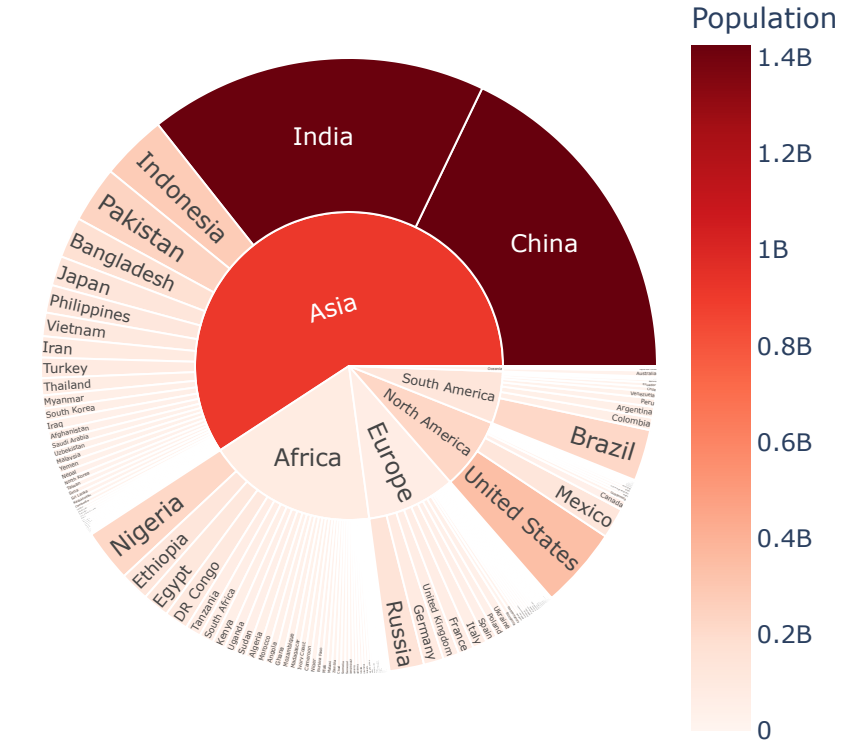
In [92]:
```
# Top 10 most populos countries
import matplotlib.pyplot as plt
import seaborn as sns

fig, ax = plt.subplots(1,2, figsize=(12,6),constrained_layout = True)
sns.barplot(data=population_sorted.head(10), x='Country/Territory', y='2022',palette="Reds_r",hu
ax[0].set_title("Top 10 most populous Countries")
ax[0].set_xlabel("Country")
ax[0].set_ylabel("Population")# Rotate x-axis labels by 45 degrees
ax[0].tick_params(axis='x', rotation=75)# Adjust layout to prevent labels from being cut off


sns.barplot(data=population_sorted.tail(10), x='Country/Territory', y='2022',palette="Reds_r",hu
ax[1].set_title("Top 10 least populous Countries")
ax[1].set_xlabel("Country")
ax[1].set_ylabel("Population")# Rotate x-axis labels by 45 degrees
ax[1].tick_params(axis='x', rotation=75)# Adjust layout to prevent labels from being cut off

plt.show()
```



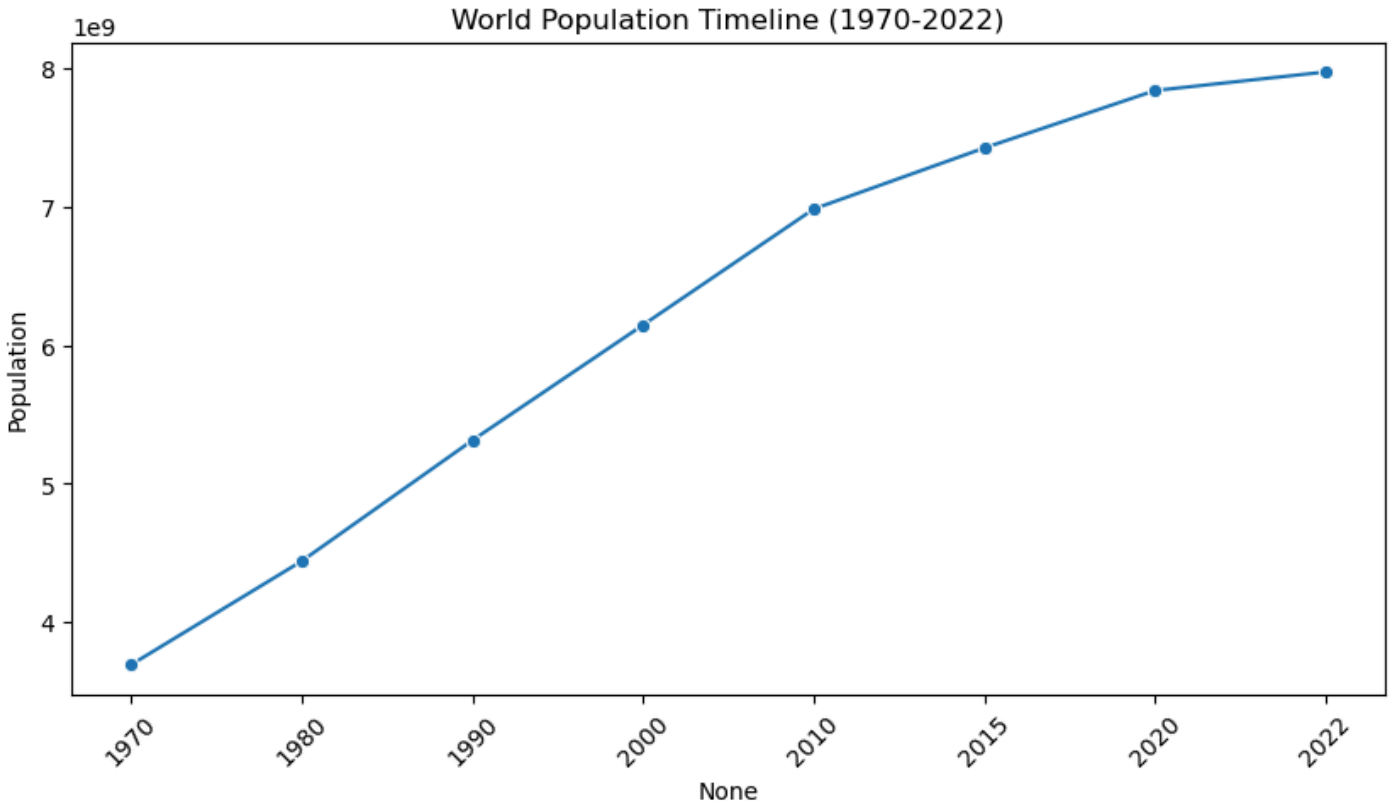


In [14]:
```python
# Population distribution by continent
import plotly.express as px
print("Population Distribution by Continent (2022):")
pop_cont = world_data.groupby("Continent")["2022"].sum()
pop_cont.sort_values(ascending=False, inplace=True)
print(pop_cont)

# Sunburst Chart
sb = px.sunburst(world_data, path=["Continent", "Country/Territory"],
                 values="2022", color="2022",
                 labels={"2022":"Population"},
                 color_continuous_scale="Reds",
                 width=500)
sb.update_layout(title=dict(text="Population Distribution by Continent (2022)",x=0.5, y=0.95))
```

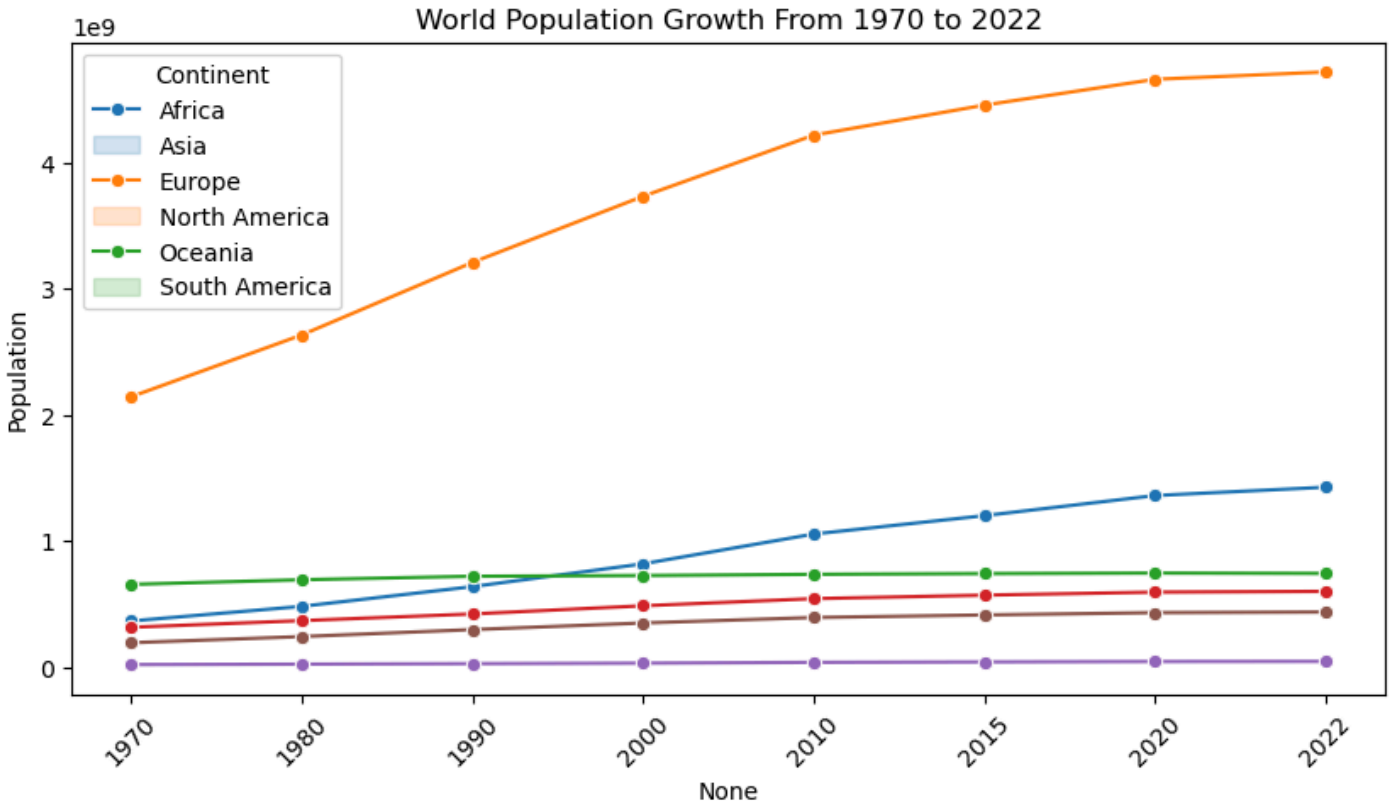
```
Population Distribution by Continent (2022):
Continent
Asia             4721383274
Africa           1426730932
Europe            743147538
North America     600296136
South America     436816608
Oceania            45038554
Name: 2022, dtype: int64
```



In [15]:
```
# Plotting Number of World Population Timeline
plt.subplots(figsize=(10,5))
growth = world_data.iloc[:,5:13].sum()[::-1]# taking index of the column with data and [::-1] th
sns.lineplot(x=growth.index, y=growth.values, marker="o")
plt.xticks(rotation=45)
plt.ylabel("Population")
plt.title("World Population Timeline (1970-2022)")
plt.show()
```



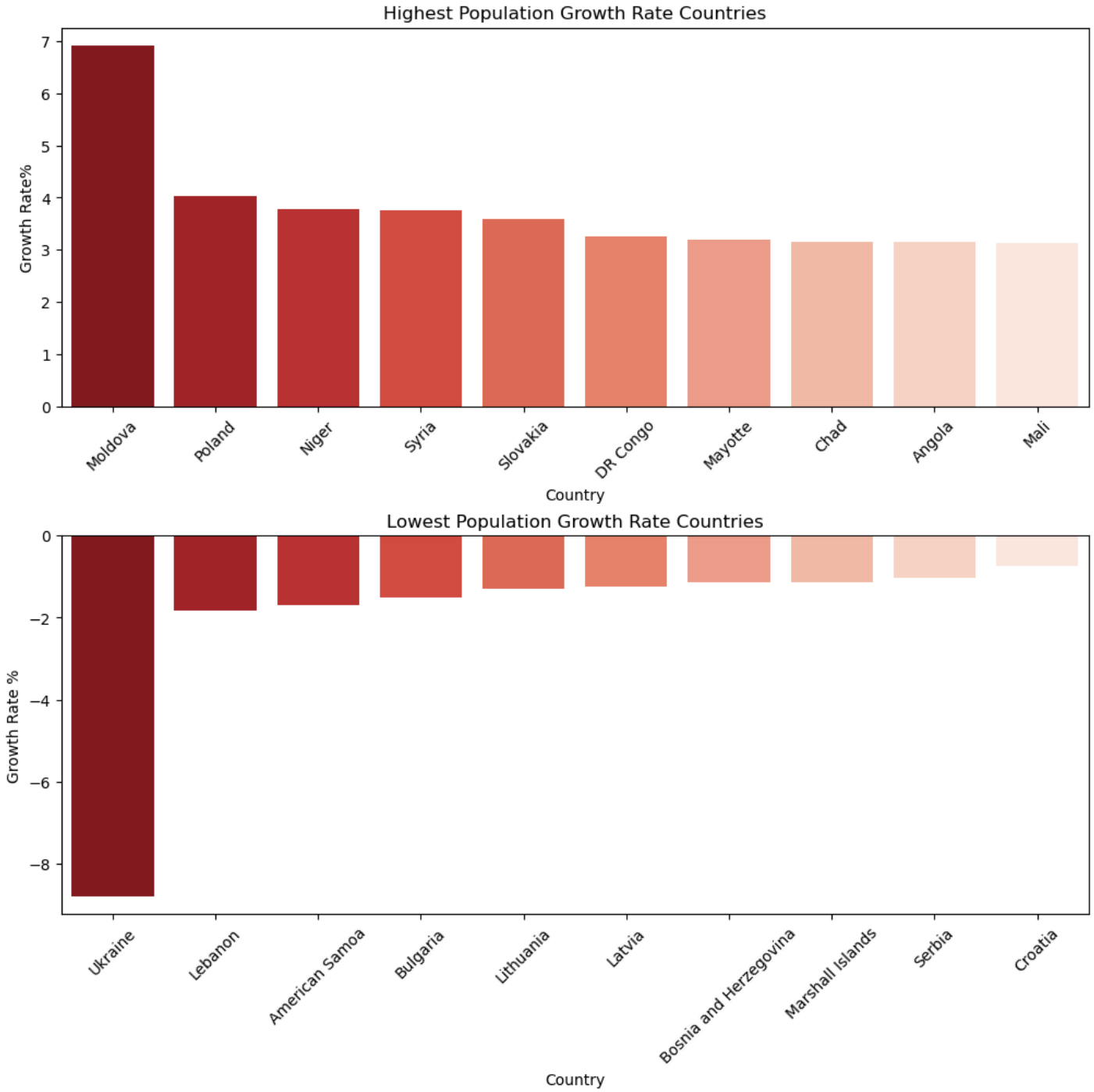
In [16]:
```
# Get total population of each continent
continent_pop = world_data.copy()
continent_pop = continent_pop.groupby("Continent").sum().iloc[:,4:12] # get 1970-2022 column [:,
continent_pop = continent_pop.iloc[:,::-1] # Reverse the years
plt.subplots(figsize=(10,5))
for continent in continent_pop.index:
    sns.lineplot(x=continent_pop.T.index, y=continent_pop.T[continent], marker="o")
plt.xticks(rotation=45)
plt.ylabel("Population")
plt.title("World Population Growth From 1970 to 2022")
plt.legend(continent_pop.index, title="Continent")
plt.show()
```



Linear Regression Analysis

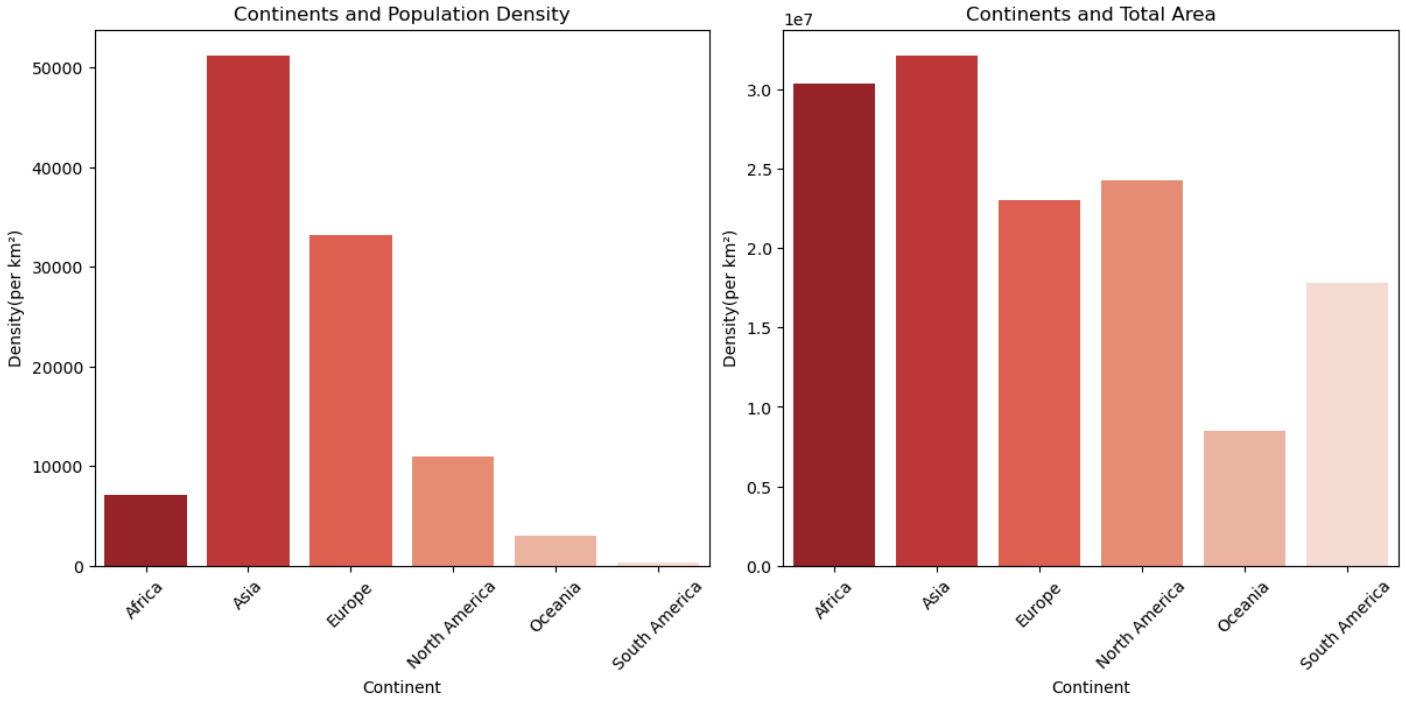
In [68]:

```
# # Countries with highest and lowest population growth
growth_sort=world_data.sort_values("Growth Rate", ascending=False)

fig, ax = plt.subplots(2,1, figsize=(10,10),constrained_layout = True)
# Highest population growth countries
sns.barplot(data=growth_sort.head(10), x='Country/Territory', y='Growth Rate',palette="Reds_r",a
ax[0].set_title("Highest Population Growth Rate Countries")
ax[0].set_xlabel("Country")
ax[0].set_ylabel("Growth Rate%")
ax[0].tick_params(axis='x', rotation=45)

# Lowest population growth countries
sns.barplot(data=growth_sort.tail(10)[::-1], x='Country/Territory', y='Growth Rate',palette="Red
ax[1].set_title("Lowest Population Growth Rate Countries")
ax[1].set_xlabel("Country")
ax[1].set_ylabel("Growth Rate %")
ax[1].tick_params(axis='x', rotation=45)

plt.show()
```



In [58]:
```
# Top 10 most populos countries
import matplotlib.pyplot as plt
import seaborn as sns

fig, ax = plt.subplots(1,2, figsize=(12,6),constrained_layout = True) # Subplot

density = world_data.groupby("Continent")["Density (per km²)"].sum()# This group function will c

"""density.index will take category value and density.values will return respected value of each
sns.barplot(x=density.index, y=density.values, palette="Reds_r",hue=density.index,ax=ax[0])

ax[0].set_title("Continents and Population Density")
ax[0].set_xlabel("Continent")
ax[0].set_ylabel("Density(per km²)")
ax[0].tick_params(axis='x', rotation=45)
#ax[0].set_xticklabels(ax[0].get_xticklabels(), rotation=45)# Rotate x-axis labels by 45 degrees#

area = world_data.groupby("Continent")["Area (km²)"].sum()# This group function will caluculate
```

```
"""area.index will take category value and area.values will return respected value of each catego
sns.barplot(x=area.index, y=area.values, palette="Reds_r",hue=area.index,ax=ax[1])

ax[1].set_title("Continents and Total Area")
ax[1].set_xlabel("Continent")
ax[1].set_ylabel("Density(per km²)")
ax[1].tick_params(axis='x', rotation=45)# Rotate x-axis labels by 45 degrees# Adjust layout to pr
plt.show() # For visualization
```



In [ ]: