Importing Libraries

```
import numpy as np
import pandas as pd
```

```
country = pd.read_csv("/content/Metadata_Country.csv")
population = pd.read_csv("/content/country_population.csv")
fertility = pd.read_csv("/content/fertility_rate.csv")
life_expectancy = pd.read_csv("/content/life_expectancy.csv")
```

Data cleaning on Country Dataset

```
country.head()
```

	Country Code	Region	IncomeGroup	SpecialNotes	TableName	Unnamed: 5
0	ABW	Latin America & Caribbean	High income	SNA data for 2000-2011 are updated from offici...	Aruba	NaN
1	AFG	South Asia	Low income	Fiscal year end: March 20; reporting period fo...	Afghanistan	NaN
2	AGO	Sub-Saharan Africa	Lower middle income	NaN	Angola	NaN
3	ALB	Europe & Central Asia	Upper middle income	NaN	Albania	NaN
4	AND	Europe & Central Asia	High income	WB-3 code changed from ADO to AND to align wit...	Andorra	NaN

```
country.tail()
```

	Country Code	Region	IncomeGroup	SpecialNotes	TableName	Unnamed: 5
258	XKX	Europe & Central Asia	Lower middle income	WB-3 code changed from KSV to XKX to align wit...	Kosovo	NaN
259	YEM	Middle East & North Africa	Lower middle income	Based on official government statistics and In...	Yemen, Rep.	NaN
260	ZAF	Sub-Saharan Africa	Upper middle income	Fiscal year end: March 31; reporting period fo...	South Africa	NaN
261	ZMB	Sub-Saharan Africa	Lower middle income	The base year is 2010. National accounts data ...	Zambia	NaN
262	ZWE	Sub-Saharan Africa	Low income	Fiscal year end: June 30; reporting period for...	Zimbabwe	NaN

```
# We want only 'Country code', 'Region' and 'TableName' columns.
country = country[['Country Code','Region','TableName']]
country.shape
```

```
(263, 3)
```

```
#Changing the 'TableName' to 'Country Name'
country.rename(columns={'TableName': 'Country Name'}, inplace=True)
```

```
#Checking data types
country.dtypes
```

```
Country Code    object
Region          object
Country Name    object
dtype: object
```

```
country.head()
```

	Country Code	Region	Country Name
0	ABW	Latin America & Caribbean	Aruba
1	AFG	South Asia	Afghanistan
2	AGO	Sub-Saharan Africa	Angola
3	ALB	Europe & Central Asia	Albania
4	AND	Europe & Central Asia	Andorra

```
country.isna().sum()
```

```
Country Code     0
Region          46
Country Name     0
dtype: int64
```

Data cleaning on fertillity dataset

```
1 fertility.head()
```

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964	1965	...	2007	2008	2009	2010	2011	2012
0	Aruba	ABW	Fertility rate, total (births per woman)	SP.DYN.TFRT.IN	4.820	4.655	4.471	4.271	4.059	3.842	...	1.763	1.764	1.769	1.776	1.783	1.791
1	Afghanistan	AFG	Fertility rate, total (births per woman)	SP.DYN.TFRT.IN	7.450	7.450	7.450	7.450	7.450	7.450	...	6.460	6.254	6.038	5.816	5.595	5.380

```
1 fertility.shape
```

```
(264, 61)
```

```
1 #We don't need 'Indicator Name' and 'Indiactor Code' column
2 fertility.drop(['Indicator Name','Indicator Code'],axis=1,inplace=True)
```

```
1 #Filling the empty values with the mean
2 fertility.fillna(fertility.mean(), inplace=True)
```

```
<ipython-input-13-47090a0846a8>:2: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a future ver
  fertility.fillna(fertility.mean(), inplace=True)
```

```
1 fertility.head()
```

	Country Name	Country Code	1960	1961	1962	1963	1964	1965	1966	1967	...	2007	2008	2009	2010	
0	Aruba	ABW	4.820000	4.655000	4.471000	4.271000	4.059000	3.842000	3.625000	3.4170	...	1.763	1.764	1.769	1.776	1.78
1	Afghanistan	AFG	7.450000	7.450000	7.450000	7.450000	7.450000	7.450000	7.450000	7.4500	...	6.460	6.254	6.038	5.816	5.59
2	Angola	AGO	7.478000	7.524000	7.563000	7.592000	7.611000	7.619000	7.618000	7.6130	...	6.368	6.307	6.238	6.162	6.08
3	Albania	ALB	6.489000	6.401000	6.282000	6.133000	5.960000	5.773000	5.581000	5.3940	...	1.668	1.650	1.646	1.653	1.66
4	Andorra	AND	5.508217	5.493573	5.495798	5.495507	5.458346	5.415664	5.365999	5.3277	...	1.180	1.250	1.190	1.270	2.87

```
1 fertility.tail()
```

	Country Name	Country Code	1960	1961	1962	1963	1964	1965	1966	1967	...	2007	2008	2009	2010	20
259	Kosovo	XKX	5.508217	5.493573	5.495798	5.495507	5.458346	5.415664	5.365999	5.3277	...	2.430	2.380	2.340	2.290	2.2
260	Yemen, Rep.	YEM	7.488000	7.531000	7.575000	7.621000	7.665000	7.705000	7.737000	7.7600	...	5.090	4.940	4.801	4.674	4.5
261	South Africa	ZAF	6.041000	6.028000	6.010000	5.986000	5.956000	5.920000	5.878000	5.8320	...	2.636	2.619	2.603	2.588	2.5
262	Zambia	ZMB	7.115000	7.169000	7.214000	7.249000	7.274000	7.291000	7.304000	7.3170	...	5.642	5.561	5.478	5.397	5.3

Data cleaning on Population Dataset

```
1 population.head()
```

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964	1965	...	2007	
0	Aruba	ABW	Population, total	SP.POP.TOTL	54211.0	55438.0	56225.0	56695.0	57032.0	57360.0	...	101220.0	10
1	Afghanistan	AFG	Population, total	SP.POP.TOTL	8996351.0	9166764.0	9345868.0	9533954.0	9731361.0	9938414.0	...	26616792.0	2729
2	Angola	AGO	Population, total	SP.POP.TOTL	5643182.0	5753024.0	5866061.0	5980417.0	6093321.0	6203299.0	...	20997687.0	2175
3	Albania	ALB	Population, total	SP.POP.TOTL	1608800.0	1659800.0	1711319.0	1762621.0	1814135.0	1864791.0	...	2970017.0	294
4	Andorra	AND	Population, total	SP.POP.TOTL	13411.0	14375.0	15370.0	16412.0	17469.0	18549.0	...	82683.0	8

5 rows × 61 columns

```
population.shape
```

```
(264, 61)
```

```
#We don't need 'Indicator Name' and 'Indiactor Code' column
population.drop(['Indicator Name','Indicator Code'],axis=1,inplace=True)
```

```
population.dropna(axis=0, inplace=True)
```

```
population.shape
```

```
(258, 59)
```

```
population = population.round(decimals=0)
```

```
population.head()
```

	Country Name	Country Code	1960	1961	1962	1963	1964	1965	1966	1967	...	2007	
0	Aruba	ABW	54211.0	55438.0	56225.0	56695.0	57032.0	57360.0	57715.0	58055.0	...	101220.0	1013
1	Afghanistan	AFG	8996351.0	9166764.0	9345868.0	9533954.0	9731361.0	9938414.0	10152331.0	10372630.0	...	26616792.0	27294
2	Angola	AGO	5643182.0	5753024.0	5866061.0	5980417.0	6093321.0	6203299.0	6309770.0	6414995.0	...	20997687.0	21759
3	Albania	ALB	1608800.0	1659800.0	1711319.0	1762621.0	1814135.0	1864791.0	1914573.0	1965598.0	...	2970017.0	2947
4	Andorra	AND	13411.0	14375.0	15370.0	16412.0	17469.0	18549.0	19647.0	20758.0	...	82683.0	838

5 rows × 59 columns

Data cleaning on Life Expectancy Dataset

```
life_expectancy.head()
```

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	196
0	Aruba	ABW	Life expectancy at birth, total (years)	SP.DYN.LE00.IN	65.662	66.074	66.444	66.787	67.11
1	Afghanistan	AFG	Life expectancy at birth,	SP.DYN.LE00.IN	32.292	32.742	33.185	33.624	34.06

```
life_expectancy.shape
```

```
(264, 61)
```

```
#We don't need 'Indicator Name' and 'Indiactor Code' column
life_expectancy.drop(['Indicator Name','Indicator Code'],axis=1,inplace=True)
```

```
life_expectancy = life_expectancy.fillna(life_expectancy.mean()).round(decimals=0)
```

```
<ipython-input-26-55c4c1a08fc7>:1: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a future ver
  life_expectancy = life_expectancy.fillna(life_expectancy.mean()).round(decimals=0)
```

```
life_expectancy.dropna(axis=0, inplace=True)
```

```
life_expectancy.head()
```

	Country Name	Country Code	1960	1961	1962	1963	1964	1965	1966	1967	...	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
0	Aruba	ABW	66.0	66.0	66.0	67.0	67.0	67.0	68.0	68.0	...	75.0	75.0	75.0	75.0	75.0	75.0	75.0	76.0	76.0	76.0
1	Afghanistan	AFG	32.0	33.0	33.0	34.0	34.0	34.0	35.0	35.0	...	60.0	60.0	61.0	61.0	62.0	62.0	62.0	63.0	63.0	64.0
2	Angola	AGO	33.0	34.0	34.0	34.0	35.0	35.0	35.0	36.0	...	55.0	56.0	57.0	58.0	59.0	60.0	60.0	61.0	61.0	62.0
3	Albania	ALB	62.0	63.0	64.0	65.0	65.0	66.0	66.0	66.0	...	76.0	76.0	76.0	77.0	77.0	77.0	78.0	78.0	78.0	78.0
4	Andorra	AND	53.0	54.0	54.0	55.0	55.0	56.0	56.0	57.0	...	69.0	70.0	70.0	70.0	71.0	71.0	71.0	72.0	72.0	72.0

Melt and Merge Function of Python

```
#Years present in the data are present in row wise format we want it in column format
years = [str(i) for i in range(1960,2017)]
print(years)
```

```
['1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968', '1969', '1970', '1971', '1972', '1973', '1974', '1975', '19
```

```
# Melt function on population data
df_population = pd.melt(population,
        id_vars='Country Name',
        value_vars=years,
        var_name='Year',
        value_name='Population')
df_population
```

	Country Name	Year	Population
0	Aruba	1960	54211.0
1	Afghanistan	1960	8996351.0
2	Angola	1960	5643182.0
3	Albania	1960	1608800.0
4	Andorra	1960	13411.0
...	...	...	...
14701	Kosovo	2016	1816200.0
14702	Yemen, Rep.	2016	27584213.0
14703	South Africa	2016	56015473.0
14704	Zambia	2016	16591390.0
14705	Zimbabwe	2016	16150362.0

14706 rows × 3 columns

```
df_population.isna().sum()
```

```
Country Name    0
Year            0
Population      0
dtype: int64
```

Merging the Country and Population data

```
country_and_population = pd.merge(country,df_population,how='left',on='Country Name')
```

```
country_and_population
```

	Country Code	Region	Country Name	Year	Population
0	ABW	Latin America & Caribbean	Aruba	1960	54211.0
1	ABW	Latin America & Caribbean	Aruba	1961	55438.0
2	ABW	Latin America & Caribbean	Aruba	1962	56225.0
3	ABW	Latin America & Caribbean	Aruba	1963	56695.0
4	ABW	Latin America & Caribbean	Aruba	1964	57032.0
...	...	...	...	...	...
14202	ZWE	Sub-Saharan Africa	Zimbabwe	2012	14710826.0
14203	ZWE	Sub-Saharan Africa	Zimbabwe	2013	15054506.0
14204	ZWE	Sub-Saharan Africa	Zimbabwe	2014	15411675.0
14205	ZWE	Sub-Saharan Africa	Zimbabwe	2015	15777451.0
14206	ZWE	Sub-Saharan Africa	Zimbabwe	2016	16150362.0

14207 rows × 5 columns

```
# Melt function on fertility data
df_fertility = pd.melt(fertility,
        id_vars='Country Code',
        value_vars=years,
        var_name='Year',
        value_name='Fertility Rate')
df_fertility
```

	Country Code	Year	Fertility Rate
0	ABW	1960	4.820000
1	AFG	1960	7.450000
2	AGO	1960	7.478000
3	ALB	1960	6.489000
4	AND	1960	5.508217
...	...	...	...
15043	XKX	2016	2.060000
15044	YEM	2016	3.995000
15045	ZAF	2016	2.458000
15046	ZMB	2016	4.981000
15047	ZWE	2016	3.760000

15048 rows × 3 columns

```
df_fertility.isna().sum()
```

```
Country Code      0
Year              0
Fertility Rate    0
dtype: int64
```

```
# Melt function on life_expectancy data
df_life_expectancy = pd.melt(life_expectancy,
        id_vars='Country Code',
        value_vars=years,
        var_name='Year',
        value_name='Life Expectancy')
df_life_expectancy
```

	Country Code	Year	Life Expectancy
0	ABW	1960	66.0
1	AFG	1960	32.0
2	AGO	1960	33.0
3	ALB	1960	62.0
4	AND	1960	53.0
...	...	...	...
15043	XKX	2016	72.0
15044	YEM	2016	65.0
15045	ZAF	2016	63.0
15046	ZMB	2016	62.0
15047	ZWE	2016	61.0

15048 rows × 3 columns

```
df_life_expectancy.isna().sum()
```

```
Country Code       0
Year               0
Life Expectancy    0
dtype: int64
```

Merging Fertility and Life expectancy data

```
fertility_and_life_expectancy=pd.merge(df_fertility, df_life_expectancy, how='left', on=['Country Code', 'Year'])
fertility_and_life_expectancy
```

	Country Code	Year	Fertility Rate	Life Expectancy
0	ABW	1960	4.820000	66.0
1	AFG	1960	7.450000	32.0
2	AGO	1960	7.478000	33.0
3	ALB	1960	6.489000	62.0
4	AND	1960	5.508217	53.0
...	...	...	...	...
15043	XKX	2016	2.060000	72.0
15044	YEM	2016	3.995000	65.0
15045	ZAF	2016	2.458000	63.0
15046	ZMB	2016	4.981000	62.0
15047	ZWE	2016	3.760000	61.0

15048 rows × 4 columns

Merging all four dataset i.e country, population, fertility, life_expectancy

```
# Merge country_and_population and fertility_and_life_expectancy
df2 = pd.merge(country_and_population, fertility_and_life_expectancy, how='left', on=['Country Code', 'Year'])
df2 = df2[['Country Name', 'Country Code', 'Region', 'Year', 'Population', 'Fertility Rate', 'Life Expectancy']]
df2
```

	Country Name	Country Code	Region	Year	Population	Fertility Rate	Life Expectancy
0	Aruba	ABW	Latin America & Caribbean	1960	54211.0	4.820	66.0
1	Aruba	ABW	Latin America & Caribbean	1961	55438.0	4.655	66.0
2	Aruba	ABW	Latin America & Caribbean	1962	56225.0	4.471	66.0
3	Aruba	ABW	Latin America & Caribbean	1963	56695.0	4.271	67.0
4	Aruba	ABW	Latin America & Caribbean	1964	57032.0	4.059	67.0
...	...	...	...	...	...	...	...
14202	Zimbabwe	ZWE	Sub-Saharan Africa	2012	14710826.0	3.996	57.0
14203	Zimbabwe	ZWE	Sub-Saharan Africa	2013	15054506.0	3.957	58.0
14204	Zimbabwe	ZWE	Sub-Saharan Africa	2014	15411675.0	3.903	59.0
14205	Zimbabwe	ZWE	Sub-Saharan Africa	2015	15777451.0	3.836	60.0
14206	Zimbabwe	ZWE	Sub-Saharan Africa	2016	16150362.0	3.760	61.0

14207 rows × 7 columns

```
df2.dtypes
```

```
Country Name        object
Country Code        object
Region              object
Year                object
Population         float64
Fertility Rate     float64
Life Expectancy    float64
dtype: object
```

```
df2.isna().sum()
```

```
Country Name          0
Country Code          0
Region             2342
Year                 14
Population           14
Fertility Rate       14
Life Expectancy      14
dtype: int64
```

```
df2.dropna(axis=0, inplace=True)
```

```
df2.describe()
```

	Population	Fertility Rate	Life Expectancy
count	1.185600e+04	11856.000000	11856.000000
mean	2.442365e+07	4.049458	63.815115
std	1.014951e+08	1.955316	11.040307
min	4.279000e+03	0.827000	19.000000
25%	4.774162e+05	2.243000	57.000000
50%	4.140050e+06	3.758000	66.000000
75%	1.308559e+07	5.804000	72.000000
max	1.378665e+09	8.866000	85.000000

Final Data

```
#Final Data
print("Final data after cleaning and Merging")
df2
```

Final data after cleaning and Merging

	Country Name	Country Code	Region	Year	Population	Fertility Rate	Life Expectancy
0	Aruba	ABW	Latin America & Caribbean	1960	54211.0	4.820	66.0
1	Aruba	ABW	Latin America & Caribbean	1961	55438.0	4.655	66.0
2	Aruba	ABW	Latin America & Caribbean	1962	56225.0	4.471	66.0
3	Aruba	ABW	Latin America & Caribbean	1963	56695.0	4.271	67.0
4	Aruba	ABW	Latin America & Caribbean	1964	57032.0	4.059	67.0
...	...	...	...	...	...	...	...
14202	Zimbabwe	ZWE	Sub-Saharan Africa	2012	14710826.0	3.996	57.0
14203	Zimbabwe	ZWE	Sub-Saharan Africa	2013	15054506.0	3.957	58.0
14204	Zimbabwe	ZWE	Sub-Saharan Africa	2014	15411675.0	3.903	59.0
14205	Zimbabwe	ZWE	Sub-Saharan Africa	2015	15777451.0	3.836	60.0
14206	Zimbabwe	ZWE	Sub-Saharan Africa	2016	16150362.0	3.760	61.0

11856 rows × 7 columns

```
df2.shape
```

```
(11856, 7)
```

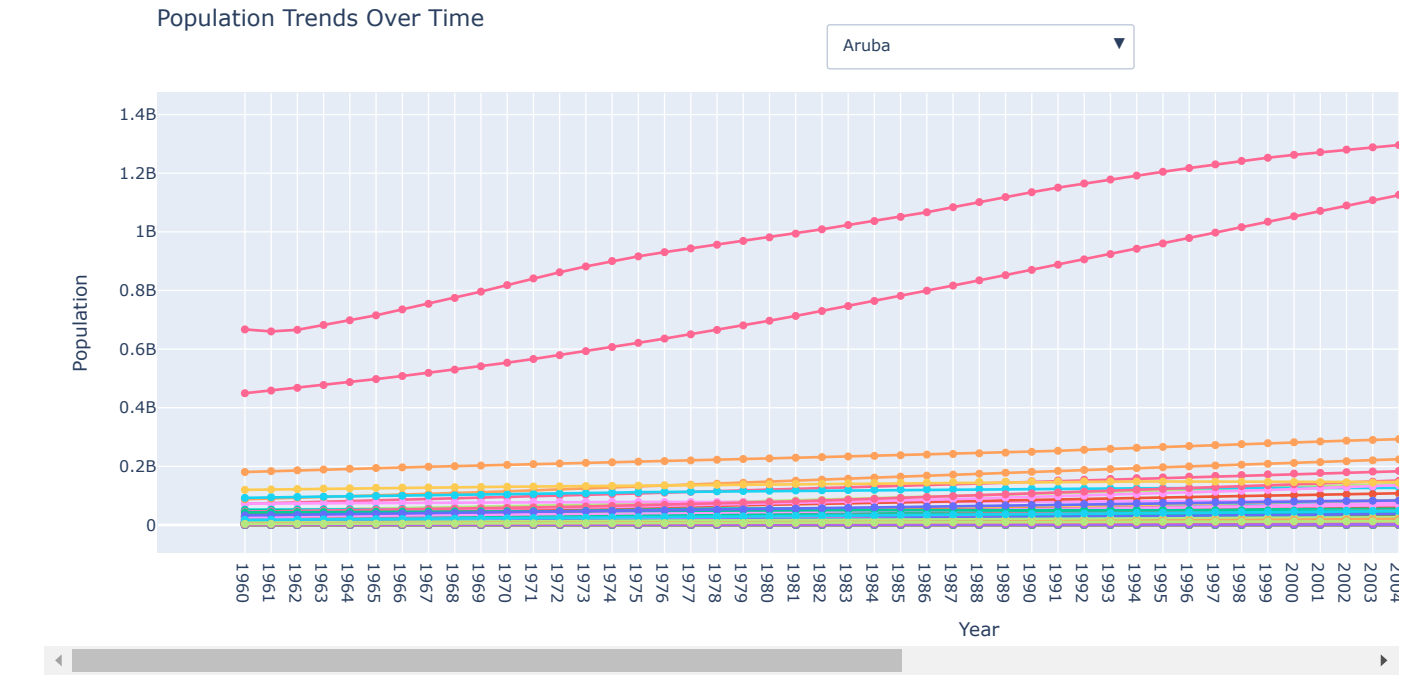
```
df2.to_csv('Merged_data.csv', index=False)
```

Data Visualization

```
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from ipywidgets import interact
```

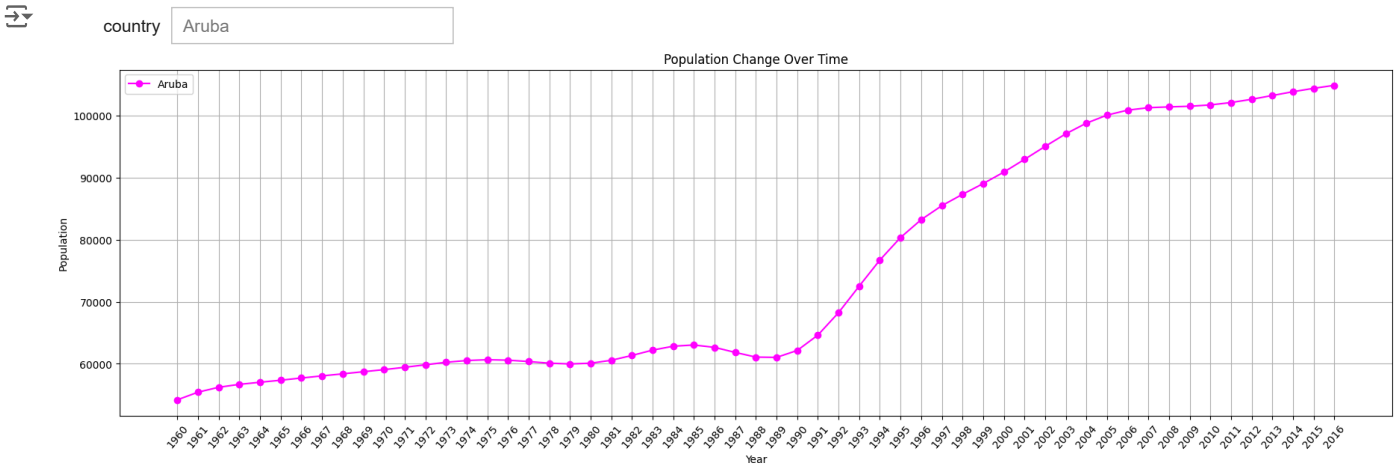
1.Creating line chart to show population trends over time w.r.to Country

```
countries = df2['Country Name'].unique()

# Create an empty figure
fig = go.Figure()

# Add traces for each country
for country in countries:
    country_data = df2[df2['Country Name'] == country]
    fig.add_trace(go.Scatter(x=country_data['Year'], y=country_data['Population'], mode='lines+markers', name=countr

fig.update_layout(
    title='Population Trends Over Time',
    xaxis=dict(title='Year'),
    yaxis=dict(title='Population'),
    showlegend=True,
    updatemenus=[
        {
            'buttons': [
                {
                    'method': 'update',
                    'label': country,
                    'args': [{'y': [df2[df2['Country Name'] == country]['Population']], 'name': country}]
                } for country in countries
            ],
            'direction': 'down',
            'showactive': True,
            'x': 0.5,
            'xanchor': 'center',
            'y': 1.15,
            'yanchor': 'top'
        }
    ]
)

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



Created filter on Country to show population trends over time w.r.to Country

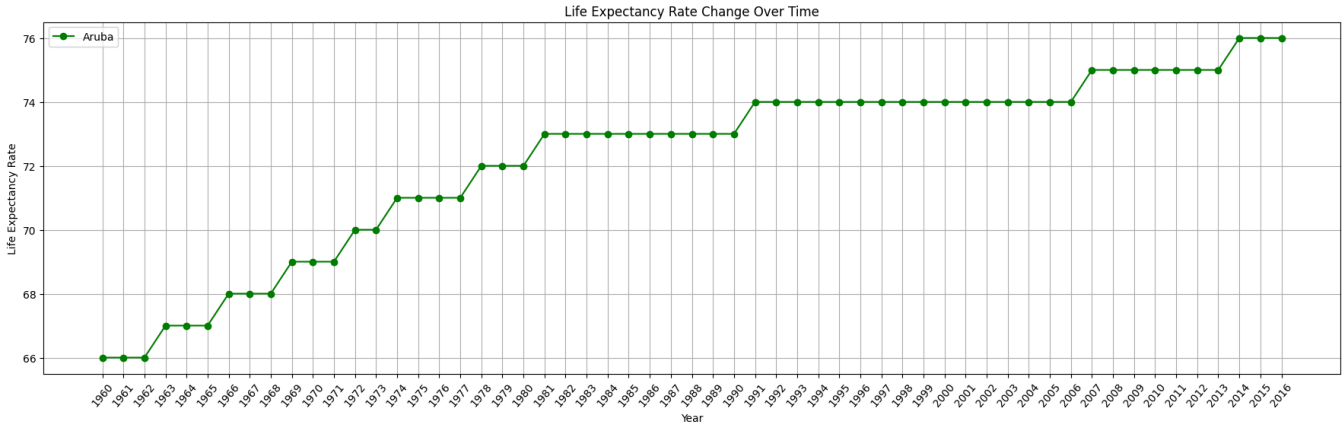
```
def plot_population(country):
    country_data = df2[df2['Country Name'] == country]
    plt.figure(figsize=(22, 6))  # Adjust figure size to accommodate the longer x-axis labels
    plt.plot(country_data['Year'], country_data['Population'],color='magenta', marker='o', linestyle='-', label=cour
    plt.title('Population Change Over Time')
    plt.xlabel('Year')
    plt.ylabel('Population')
    plt.legend(loc='upper left')  # Change the legend position if needed
    plt.xticks(rotation=50)  # Rotate x-axis labels for better readability
    plt.grid(True)
    plt.show()

# Get unique countries
countries = df2['Country Name'].unique()

# Create interactive plot using ipywidgets
interact(plot_population, country=countries);
```



Created filter on Country to show Life Expectancy trends over time w.r.to Country

```
def life_expectancy_rate(country):
    country_data = df2[df2['Country Name'] == country]
    plt.figure(figsize=(22, 6))
    plt.plot(country_data['Year'], country_data['Life Expectancy'],color='green', marker='o', linestyle='-', label=(
    plt.title('Life Expectancy Rate Change Over Time')
    plt.xlabel('Year')
    plt.ylabel('Life Expectancy Rate')
    plt.legend(loc='upper left')  # Change the legend position if needed
    plt.xticks(rotation=50)  # Rotate x-axis labels for better readability
    plt.grid(True)
    plt.show()

# Get unique countries
countries = df2['Country Name'].unique()

# Create interactive plot using ipywidgets
interact(life_expectancy_rate, country=countries);
```

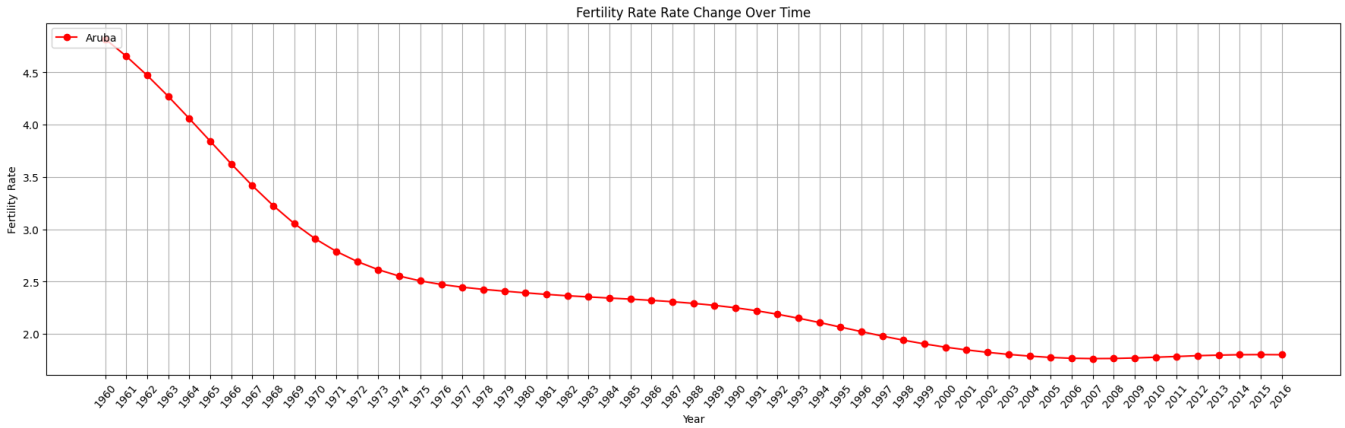
country Aruba



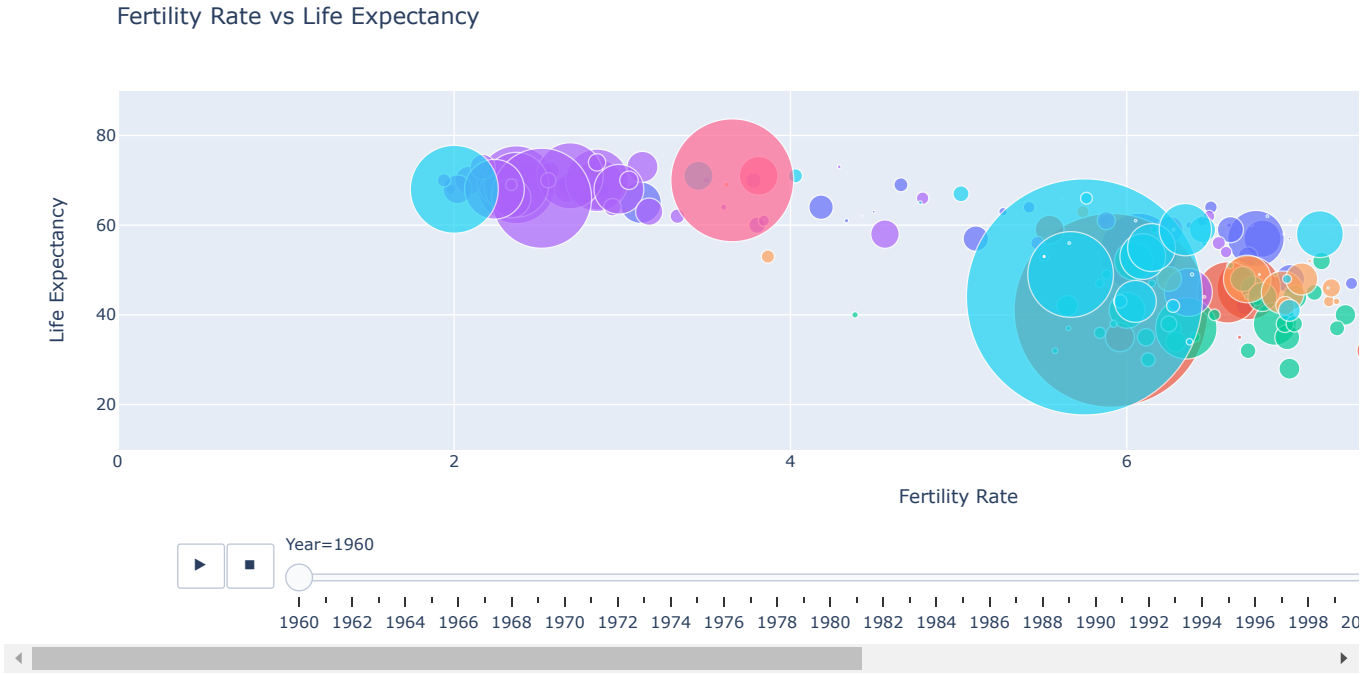
Created filter on Country to show Fertility Rate trends over time w.r.to Country

```
def fertility_rate(country):
    country_data = df2[df2['Country Name'] == country]
    plt.figure(figsize=(22, 6))
    plt.plot(country_data['Year'], country_data['Fertility Rate'],color='red', marker='o', linestyle='-', label=cour
    plt.title('Fertility Rate Rate Change Over Time')
    plt.xlabel('Year')
    plt.ylabel('Fertility Rate')
    plt.legend(loc='upper left')  # Change the legend position if needed
    plt.xticks(rotation=50)  # Rotate x-axis labels for better readability
    plt.grid(True)
    plt.show()

# Get unique countries
countries = df2['Country Name'].unique()

# Create interactive plot using ipywidgets
interact(fertility_rate, country=countries);
```

country Aruba

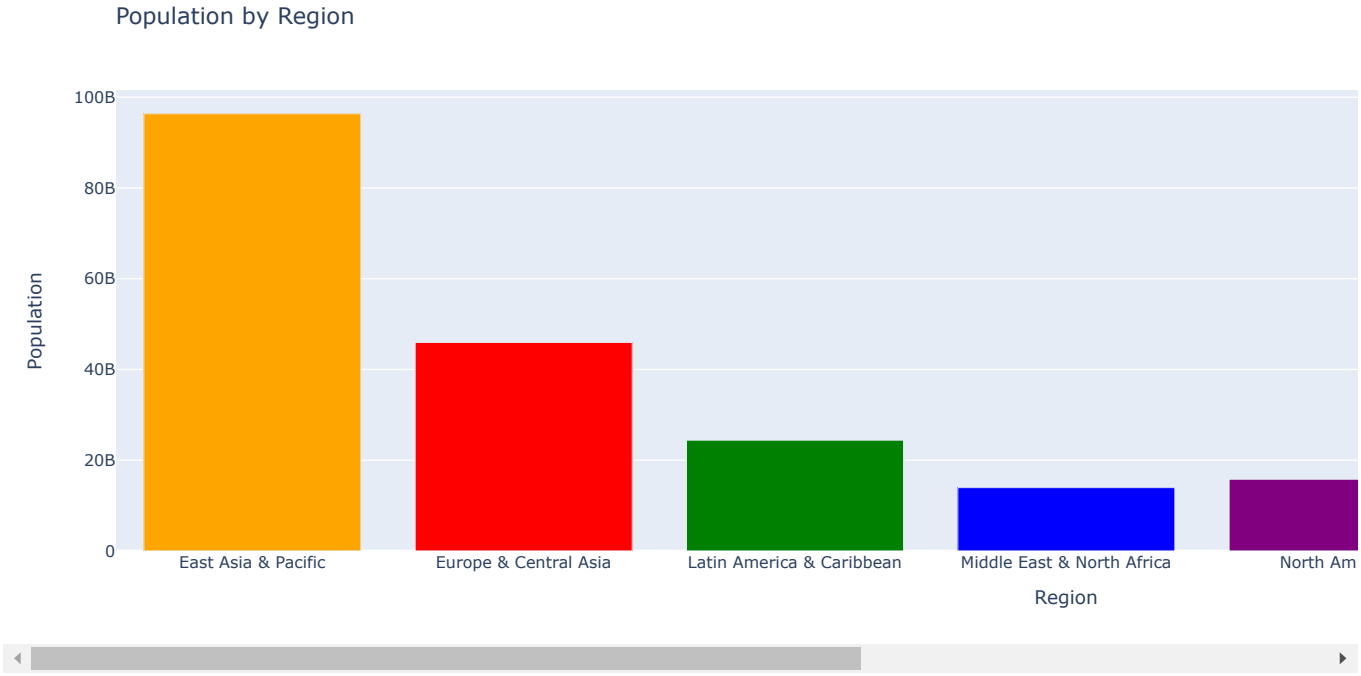


```
# Fertility rate vs life expectancy
px.scatter(df2,
           x="Fertility Rate",
           y="Life Expectancy",
           animation_frame="Year",
           animation_group="Country Code",
           size="Population",
           size_max=float("180"),
           hover_name="Country Name",
           color="Region",
           template="plotly",
           labels={'Region'},
           range_x=[0,10],
           range_y=[10,90],
           title='Fertility Rate vs Life Expectancy')
```



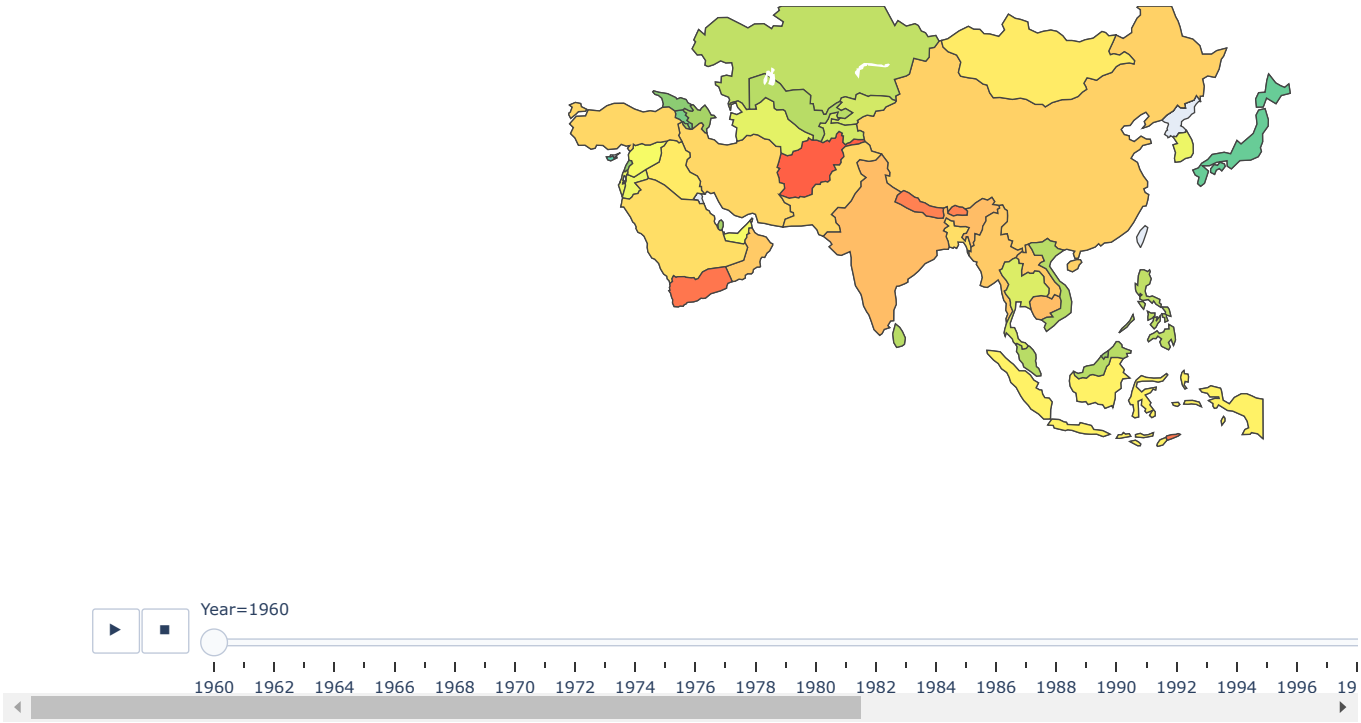
Bar plot showing region wise population

```
region_population = df2.groupby('Region')['Population'].sum().reset_index()

# Define colors for each region
colors = ["orange", "red", "green", "blue", "purple",'magenta','cyan']  # Example colors, you can define your own

# Create a Plotly bar plot with customized colors
fig = go.Figure(go.Bar(
    x=region_population['Region'],
    y=region_population['Population'],
    marker_color=colors
))

# Update layout
fig.update_layout(
    title='Population by Region',
    xaxis=dict(title='Region'),
    yaxis=dict(title='Population'),
    showlegend=False
)

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



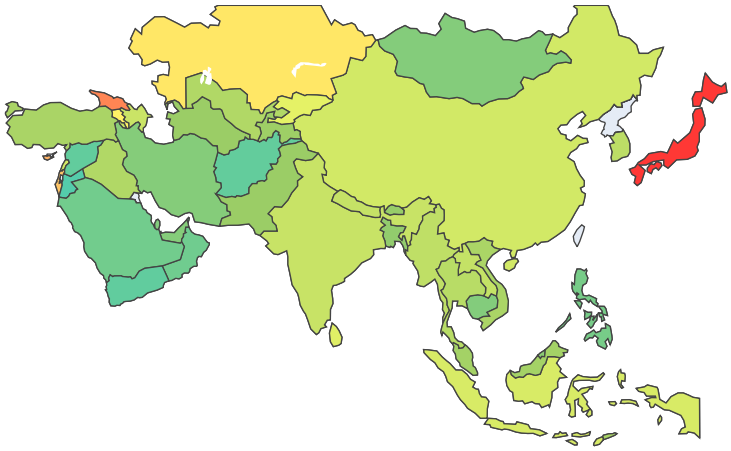
Country wise Life expectancy over the years

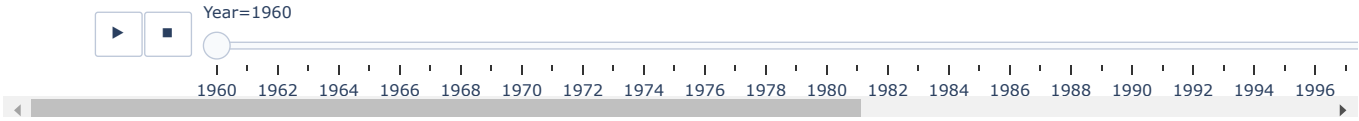
```
# map graph for country code and life expectancy
colorscale =["rgb(255, 51, 51)",   # Red
    "rgb(255, 179, 102)", # Orange
    "rgb(255, 255, 102)", # Yellow
    "rgb(153, 204, 102)", # Green
    "rgb(51, 204, 204)" ]
fig=px.choropleth(df2, locations="Country Code", color="Life Expectancy",color_continuous_scale=colorscale,
                    scope="asia", hover_name="Country Name", animation_frame="Year", animation_group="Country Code"
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



Country wise Fertility Rate over the years

```
fig=px.choropleth(df2, locations="Country Code", color="Fertility Rate",color_continuous_scale=colorscale,
                    scope="asia", hover_name="Country Name", animation_frame="Year", animation_group="Country Code"
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```






```
1 #scatter graph for population and fertility rate
2 px.scatter(df2, x="Year", y=["Population", "Fertility Rate"], color="Region", color_continuous_scale='bluered', titl
```