Exploratory Data Analysis on the World Bank Data

The objective of this project is to collect data from the World Bank Open APIs and prepare and analyse the data using Python.

In [1]:

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

from IPython.display import display
import matplotlib.pyplot as plt
import seaborn as sns
```

The purpose of this report is to first collect the data from the World Open Bank API and then prepare that dataset for analysis and visualization of some big countries among the world datasets. There are multiple indicators which has been chosen for the purpose of analysis. For example, Total Population, GDP ($), unemployment, electric power consumption, agriculture and some more. Following are the points of interest question for the purpose of analysis. From all the above indicators, some big countries which are selected for that are USA, Great Britain, China, Japan, Africa, Canada and India

Analyse and Summarise the dataset

Read the cleaned data from the CSV files

We can now use the cleaned dataset for analysis. Hence we first read the CSV files we created in the previous task.

In [16]:

```python
# read the cleaned data from every CSV
try:
    df_cleaned_us=pd.read_csv('USA.csv')
    df_cleaned_in=pd.read_csv('India.csv')
    df_cleaned_cn=pd.read_csv('China.csv')
    df_cleaned_jp=pd.read_csv('Japan.csv')
    df_cleaned_ca=pd.read_csv('Canada.csv')
    df_cleaned_gb=pd.read_csv('Great Britain.csv')
    df_cleaned_za=pd.read_csv('South Africa.csv')
    print("Successfully read all the files")
except:
    # handling I/O exceptions
    print("Unexpected error in reading a file. Check the file path and if a f

# display data of one country to check if the cleaned data is loaded
print("Displaying data for USA: ")
display(df_cleaned_us.head())

# create a list of clean dataframes for further analysis
list_cleaned_df=[df_cleaned_us, df_cleaned_in, df_cleaned_cn, df_cleaned_jp,
```

```
Successfully read all the files
Displaying data for USA:
```

	Total Population	Female Population	Male Population	Birth Rate	Death Rate	Compulsory Education Dur.	Employment in Industry(%)	Employm Agriculture
0	326838199.0	165118706.0	161719493.0	11.6	8.678	12.0	19.870001	
1	325122128.0	164262696.0	160859432.0	11.8	8.638	12.0	19.730000	
2	323071755.0	163245938.0	159825817.0	12.2	8.493	12.0	19.780001	
3	320738994.0	162095782.0	158643212.0	12.4	8.440	12.0	19.860001	
4	318386329.0	160946434.0	157439895.0	12.5	8.237	12.0	19.980000	

Prepare a combined DataFrame for further analysis

For each country, separate dataframes have been created. There may be cases when we may need the entire data across all countries in certain analysis. Hence it is a good idea to have a single combined dataframe ready.

```
In [17]:  combined_df=pd.concat(list_cleaned_df,sort=False)
          combined_df.head(200)
```

Out[17]:

	Total Population	Female Population	Male Population	Birth Rate	Death Rate	Compulsory Education Dur.	Employment in Industry(%)	Employ Agricultu
0	326838199.0	165118706.0	161719493.0	11.6	8.678	12.0	19.870001	
1	325122128.0	164262696.0	160859432.0	11.8	8.638	12.0	19.730000	
2	323071755.0	163245938.0	159825817.0	12.2	8.493	12.0	19.780001	
3	320738994.0	162095782.0	158643212.0	12.4	8.440	12.0	19.860001	
4	318386329.0	160946434.0	157439895.0	12.5	8.237	12.0	19.980000	
...	...	...	...	...	...	...	...	
18	126843000.0	64552936.0	62290064.0	9.4	7.700	9.0	31.420000	
19	126631000.0	64416945.0	62214055.0	9.3	7.800	9.0	31.860001	
20	126400000.0	64270952.0	62129048.0	9.6	7.500	9.0	32.220001	
21	126057000.0	64068675.0	61988325.0	9.5	7.300	9.0	33.259998	
22	125757000.0	63890191.0	61866809.0	9.6	7.100	9.0	33.430000	

200 rows × 20 columns

Descriptive Statistics

We can now analyse the data we have loaded. Descriptive statistics show the characteristics of numerical features. It gives information such as count, mean, mininum and maximum values etc. Hence we exclude the categorical features Year and Country from the combined DataFrame.

```
In [31]:  combined_df = pd.read_csv('worldbankindicatorsdatase.csv')
          combined_df
```

Out[31]:

	Unnamed: 0	Total Population	Female Population	Male Population	Birth Rate	Death Rate	Compulsory Education Dur.	Employ Industr
0	0	326,838,199.0	165,118,706.0	161,719,493.0	11.6	8.7	12.0	
1	1	325,122,128.0	164,262,696.0	160,859,432.0	11.8	8.6	12.0	
2	2	323,071,755.0	163,245,938.0	159,825,817.0	12.2	8.5	12.0	
3	3	320,738,994.0	162,095,782.0	158,643,212.0	12.4	8.4	12.0	
4	4	318,386,329.0	160,946,434.0	157,439,895.0	12.5	8.2	12.0	
...	...	...	...	...	...	...	...	
408	54	18,896,303.0	9,495,930.0	9,400,373.0	40.2	16.2	9.0	
409	55	18,423,157.0	9,254,686.0	9,168,471.0	40.4	16.5	9.0	
410	56	17,965,733.0	9,020,208.0	8,945,525.0	40.7	16.8	9.0	
411	57	17,524,533.0	8,793,188.0	8,731,345.0	40.9	17.1	9.0	
412	58	17,099,836.0	8,574,106.0	8,525,730.0	41.1	17.4	9.0	

413 rows × 21 columns

In [18]:
```python
# create a copy so that the original DF is not affected
# drop the columns year and country
df_copy=combined_df.drop(['Year', 'Country'], axis='columns')
df_copy.describe()
```

Out[18]:

	Total Population	Female Population	Male Population	Birth Rate	Death Rate	Compulsory Education Dur.	Empl Indu
count	3.540000e+02	4.130000e+02	4.130000e+02	413.000000	413.000000	413.000000	413.
mean	2.626257e+08	1.709377e+08	1.787918e+08	19.764932	9.427298	9.714286	23.
std	3.875364e+08	2.079419e+08	2.229416e+08	9.303838	3.052679	1.279303	3.
min	1.709984e+07	8.574106e+06	8.525730e+06	7.400000	5.900000	8.000000	15.
25%	3.686665e+07	2.397613e+07	2.331548e+07	12.800000	7.200000	9.000000	21.
50%	7.983817e+07	6.240214e+07	6.066686e+07	16.400000	8.600000	9.000000	23.
75%	2.462390e+08	2.933630e+08	3.154396e+08	23.499000	10.900000	11.000000	25.
max	1.402760e+09	6.828548e+08	7.199052e+08	43.370000	25.430000	12.000000	34.

We can oberve that there are total **413 records** across all the countries. The values for each of the countries are varied. For example, consider female employment percentage in agriculture. The minimum value is 0.5% and the maximum is 76%. This means that some countries have a very low percentage of females in agriculture whereas some other countries have a good number of females employed in this field.

Also, on observing the national income per capita, we can see that the lowest value is around -26. On observing the data carefully it was found that some countries do have a negative national income in some years. This may be probably because the country was in losses and there was no income as such.

Analysis Using Plots

Comparing Population of Countries in 2000 and 2018:

The difference in population has been shown using a **grouped bar** chart.

In [21]:

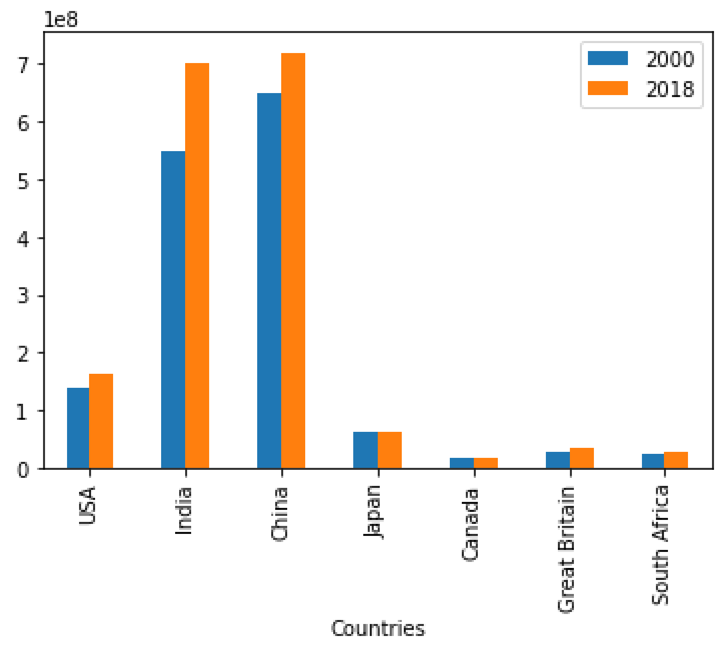
```
# refer to the list of countries
list_countries = COUNTRY_LIST
# intialise two dataframes df_00- year 2000, df_18 - year 2018
df_00 = pd.DataFrame()
df_18 = pd.DataFrame()

# for each dataframe in the list of cleaned dataframes
for i,df in enumerate(list_cleaned_df):
    # pick the value of Total Population for year 2000 and 2018
    df_00[list_countries[i]] = df[df['Year'] == 2000]["Male Population"]
    df_18[list_countries[i]] = df[df['Year'] == 2018]["Male Population"]

# The resulting dataframes have the countries as columns and the two rows eac
# To be able to draw a grouped bar plot we need the years as columns, hence w
df_00 = df_00.T
df_18 = df_18.T

pd.options.display.float_format = '{:,.1f}'.format  # set other global format

# rename the columns to the year
df_00 = df_00.rename(columns={18 : 2000})
df_18 = df_18.rename(columns={0 : 2018})

# join the dataframes for both the years
df_both_years= df_00.join(df_18)

# the index is the Country name, hence we add it as a column into the data fr
df_both_years['Countries'] = df_both_years.index

# drop the original index
df_both_years.reset_index(drop=True)

print("Data of Total Population for 2000 and 2018 for all countries: ")
display(df_both_years)
```

```
Data of Total Population for 2000 and 2018 for all countries:
```

	2000	2018	Countries
USA	138,983,981.0	161,719,493.0	USA
India	549,387,864.0	703,055,580.0	India
China	647,869,175.0	719,905,214.0	China
Japan	62,290,064.0	61,797,750.0	Japan
Canada	15,200,542.0	18,387,421.0	Canada
Great Britain	28,696,560.0	32,807,966.0	Great Britain
South Africa	22,171,606.0	28,495,093.0	South Africa

```
In [22]:  plt.figure(figsize=(7, 5))
          # plot the chart using matplotlib.pyplot library
          df_both_years.plot(kind='bar',x='Countries',y=[2000, 2018])
```

```
Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x27892343c10>

         <Figure size 504x360 with 0 Axes>
```



```
India and China have the biggest population in both years, as can be seen.
However, India's population growth in 2018 was higher than China's, bringing
their populations closer together. Canada has the smallest population of all the
countries, with only a slight increase. One remarkable finding for Japan is that
the population is the same in both 2000 and 2018, indicating that the country
has effective population control techniques.
```

Average Birth Rate and Death Rate for countries across all the years

```
In [23]:  def group_df(feature):
              # create a new dataframe
              df_grouped=pd.DataFrame()

              # find average for each country
              df_grouped['Avg. ' + feature]=combined_df.groupby('Country')[feature].mea

              # set the index as a column - countries
              df_grouped['Country']=df_grouped.index

              # drop the index
              df_grouped.reset_index(drop=True, inplace=True)

              # sort the rows based of Avg Birth rate
              df_grouped.sort_values('Avg. '+feature, inplace=True, ascending=False)

              print("Avg. " + feature)
              display(df_grouped)

              return df_grouped

          def plot_bar(df, x_feature, y_feature):
              # bar plot
              plt.figure(figsize=(8, 5))
              sns.set(style="whitegrid")
              ax = sns.barplot(
                  data= df,
                  x= x_feature,
                  y= "Avg. " + y_feature)
```

In [24]:

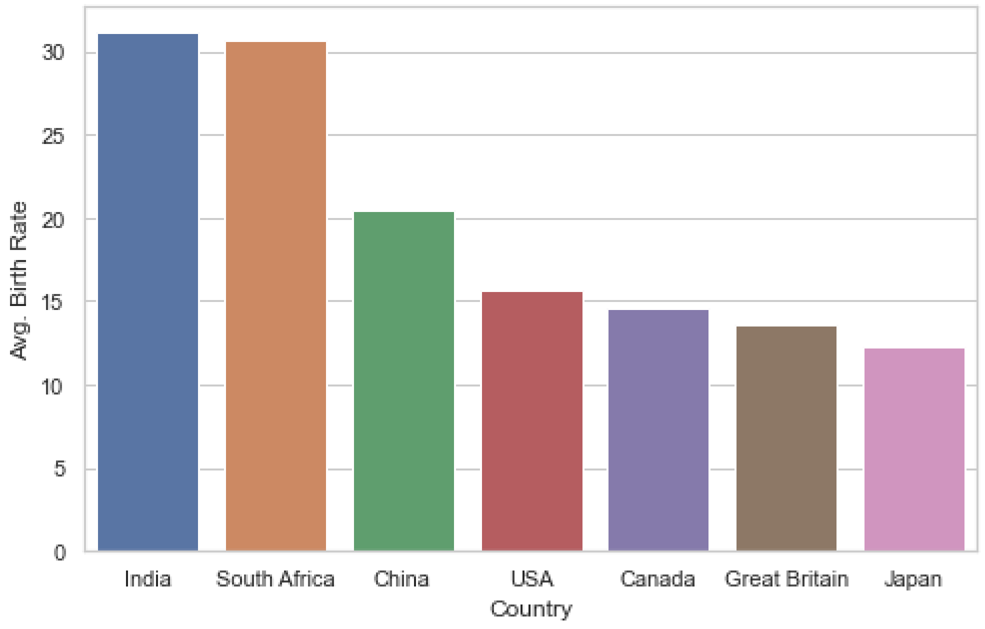
```
df_birth=group_df('Birth Rate')
plot_bar(df_birth, 'Country', 'Birth Rate')

print("==========================================================")
df_death=group_df('Death Rate')
plot_bar(df_death, 'Country', 'Death Rate')
```

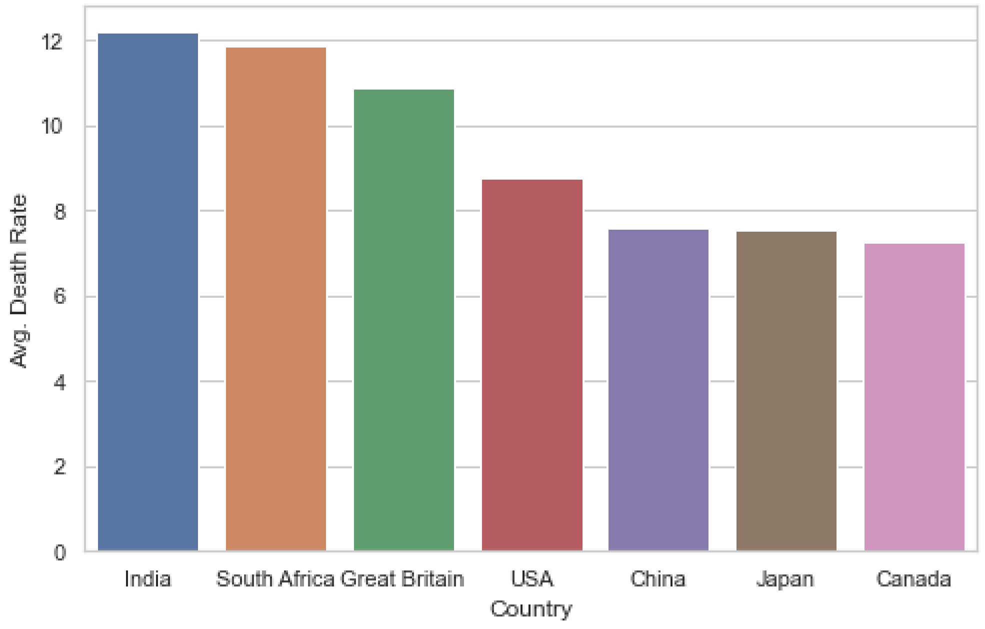
```
Avg. Birth Rate
```

	Avg. Birth Rate	Country
3	31.1	India
5	30.6	South Africa
1	20.5	China
6	15.6	USA
0	14.6	Canada
2	13.6	Great Britain
4	12.3	Japan

```
==========================================================
Avg. Death Rate
```

	Avg. Death Rate	Country
3	12.2	India
5	11.9	South Africa
2	10.9	Great Britain
6	8.7	USA
1	7.6	China
4	7.5	Japan
0	7.2	Canada





We can see from the above two plots that India has the highest average birth and death rates, which is interesting given that it had the second greatest population in 2018. The trend in South Africa is similar, which is unexpected given the country's small population. China's birth rate is higher than its death rate, resulting in the world's largest population. The death rate in the United Kingdom is substantially greater than the average birth rate, resulting in a small population

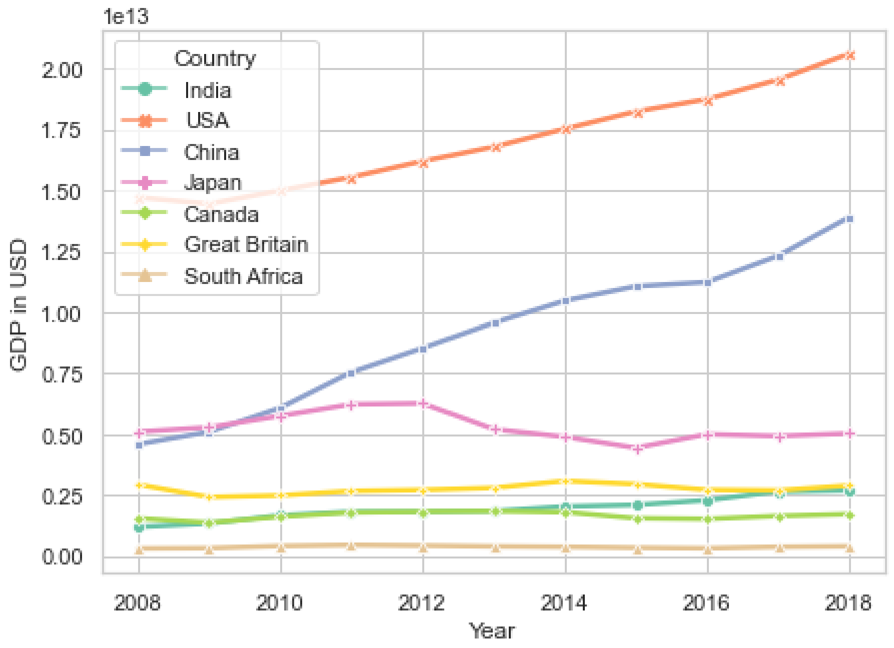
### GDP for all countries in the last 10 years

In [25]:

```
# function to to form a dataframe with Year, GDP and Country
def extract_columns(df_cleaned):
    df=pd.DataFrame()
    # pick data for the recent 10 years, note that the data sorted in descend
    df['Year']=df_cleaned.loc[:10, 'Year']
    df['GDP in USD']=df_cleaned.loc[:10, 'GDP in USD']
    df['Country']=df_cleaned.loc[:10, 'Country']
    return df

# function to fetch a single dataframe with 3 features from each country
def form_gdp_df():
    # function call to extract_columns()
    indf=extract_columns(df_cleaned_in)
    usdf=extract_columns(df_cleaned_us)
    cndf=extract_columns(df_cleaned_cn)
    jpdf=extract_columns(df_cleaned_jp)
    cadf=extract_columns(df_cleaned_ca)
    gbdf=extract_columns(df_cleaned_gb)
    zadf=extract_columns(df_cleaned_za)
    # combine the 7 dfs into a single df with 3 columns
    # we ignore the original index
    gdp_df=pd.concat([indf, usdf, cndf, jpdf, cadf, gbdf, zadf], ignore_index
    return gdp_df

# get the combined DF
gdp_df=form_gdp_df()

print("Few records from the Dataframe containing Year, GDP and Country:")
display(gdp_df.head())

# set figure size
plt.figure(figsize=(7, 5))
sns.set(style="whitegrid")
# plot using seaborn library
ax=sns.lineplot(x='Year', y='GDP in USD', hue='Country', style="Country",pale
```

```
Few records from the Dataframe containing Year, GDP and Country:
```

	Year	GDP in USD	Country
0	2018	2,701,111,782,775.0	India
1	2017	2,651,472,946,374.9	India
2	2016	2,294,797,980,509.0	India
3	2015	2,103,587,813,812.8	India
4	2014	2,039,127,446,298.6	India



A line plot clearly depicts trends over time and can also be used to compare the patterns of several categories. We can see that the United States has the greatest GDP of all countries, which has been consistent over time. China's GDP was low in 2008 and has increased significantly since then, owing to their growth in numerous industries such as manufacturing; but, in comparison to the United States, it is still quite low.

The GDP of Japan shows a modest increase up until 2012, which could be attributable to Japan's growth plan, which was implemented to lift the country out of deflation, but then it drops.

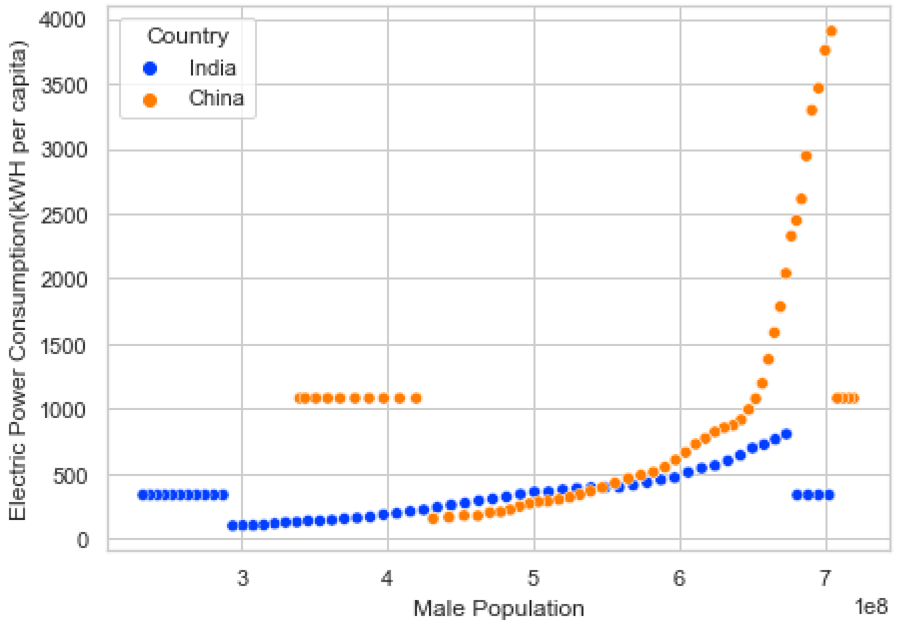
Electric Power Consumption vs Population for India and China

In [27]:

```
# function to extract specific columns from the DFs for India and China
def form_in_cn_df():
    # for India
    indf=df_cleaned_in[['Male Population', 'Electric Power Consumption(kWH pe
    # for China
    cndf=df_cleaned_cn[['Male Population', 'Electric Power Consumption(kWH pe
    # combine the two dataframes
    in_cn_df=pd.concat([indf, cndf])
    return in_cn_df

# get the desired data
in_cn_df=form_in_cn_df()
print("Few records from the selected features: ")
display(in_cn_df.head())

# scatter plot
plt.figure(figsize=(7, 5))
sns.set(style="whitegrid")
ax=sns.scatterplot(x='Male Population', y='Electric Power Consumption(kWH per
```

```
Few records from the selected features:
```

	Male Population	Electric Power Consumption(kWH per capita)	Country
0	703,055,580.0	335.2	India
1	695,880,522.0	335.2	India
2	688,604,687.0	335.2	India
3	681,223,332.0	335.2	India
4	673,747,770.0	804.5	India



It is evident from the accompanying graph that as the population grows, so will the need for electricity. The dramatic increase in China's electric power consumption demonstrates this. There has been a gradual growth in consumption in India as the population has grown. The mean consumption is indicated by the constant numbers in both lines.

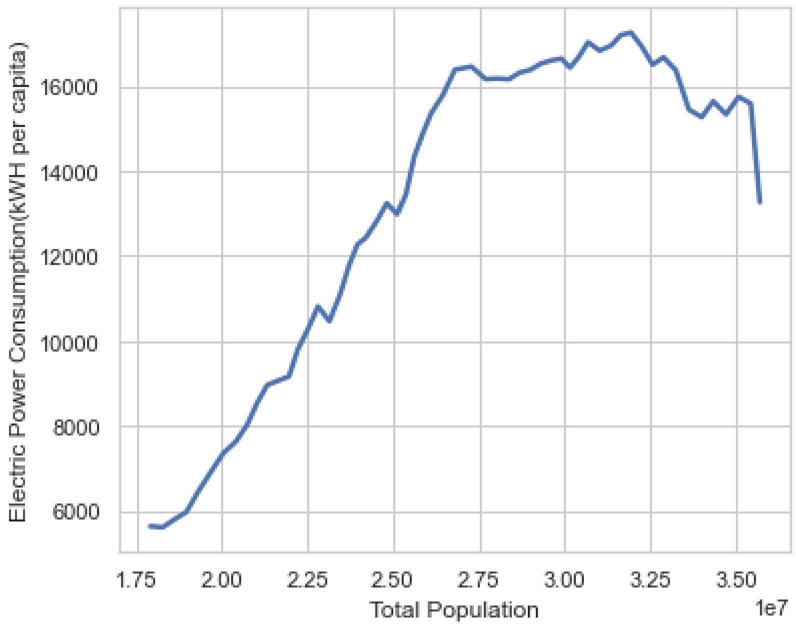
Total Population vs Electric Power consumption for Canada upto 2015

In [28]:
```
# read the columns from the df for Canada
df=df_cleaned_ca.loc[3:, ['Electric Power Consumption(kWH per capita)','Total

print("First few records of the data: ")
display(df.head())

# line plot
plt.figure(figsize=(6, 5))
sns.set(style="whitegrid")
sns.lineplot(x='Total Population', y='Electric Power Consumption(kWH per capi
```

```
First few records of the data:
```

	Electric Power Consumption(kWH per capita)	Total Population	Year
3	13,266.0	35,702,908.0	2015
4	15,588.5	35,437,435.0	2014
5	15,750.8	35,082,954.0	2013
6	15,336.6	34,714,222.0	2012
7	15,644.5	34,339,328.0	2011

Out[28]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2789262cd60>
```

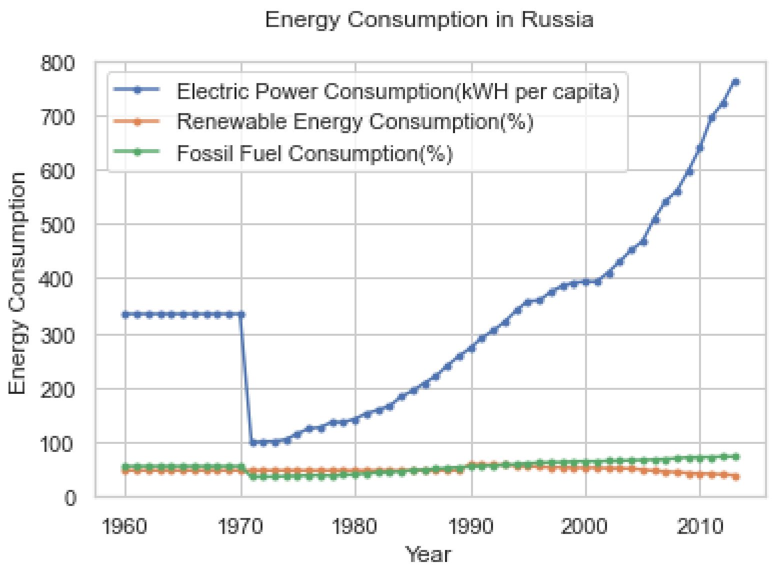


We can see from the graph above that Canada's electric power usage has risen in lockstep with the country's population growth. However, after a certain point, it remained constant and then decreased. Despite having the smallest population, Canada has the highest electric power usage, which may be attributable to the country's unusually cold climate.

Variation in different Energy Consumption over the years for Russia

For this analysis, I have chosen the energy consumption data for Russia over the years upto 2010 and plotted a multi-line chart to observe the trend.

In [29]:
```
# Pick the columns Year, and 3 different power consumptions from the datafram
plt.plot(df_cleaned_in.loc[5:, ['Year']],df_cleaned_in.loc[5:, ['Electric Pow
plt.plot(df_cleaned_in.loc[5:, ['Year']],df_cleaned_in.loc[5:, ['Renewable En
plt.plot(df_cleaned_in.loc[5:, ['Year']],df_cleaned_in.loc[5:, ['Fossil Fuel 

plt.legend(['Electric Power Consumption(kWH per capita)', 'Renewable Energy C
plt.title("Energy Consumption in Russia\n")
plt.xlabel('Year')
plt.ylabel('Energy Consumption')
plt.show()
```



From this plot, it can be observed that The Electric power consumption has increased significantly over time. On the other hand, the consumption for Fossil Fuels and Renewable Energy is low as compared to electrical energy. Fossil Fuel consumption seems to have increased between 2000 and 2010.

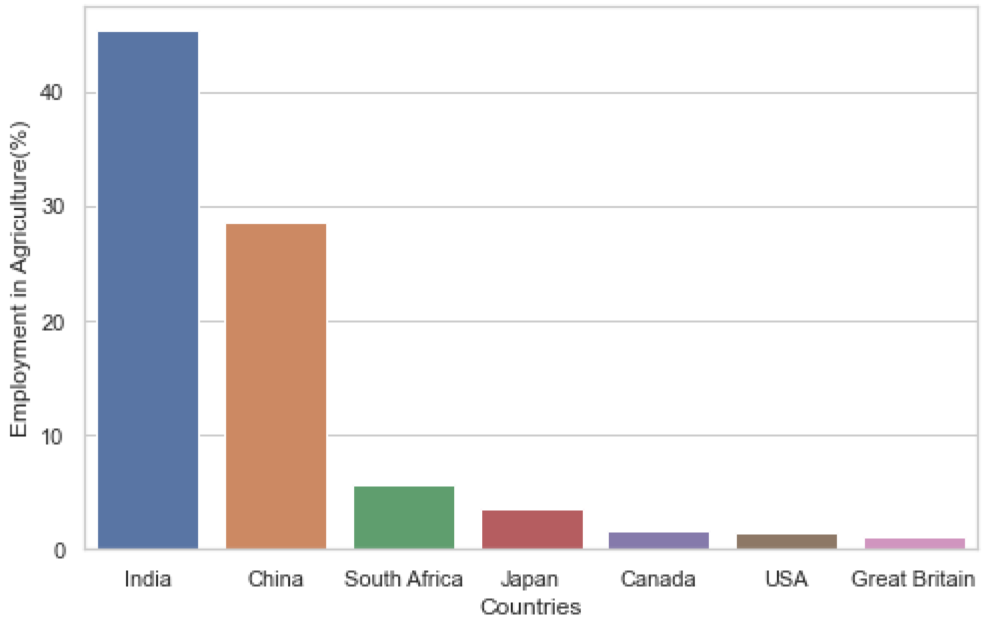
Employment in the Agricultural sector in Countries in the year 2015

In [30]:

```
# get the list of countries
list_countries=COUNTRY_LIST

df_agr_emp=pd.DataFrame()

# iterate over the dataframe
for i,df in enumerate(list_cleaned_df):
    # store the % Employment in Agriculture for each country
    df_agr_emp[list_countries[i]]=df[df['Year'] == 2015]["Employment in Agric

# take the transpose
df_agr_emp=df_agr_emp.T

pd.options.display.float_format = '{:,.1f}'.format  # set other global format

# since we took a transpose to get countries as the rows, the row index shows
# Add it as a column in the DF
df_agr_emp['Countries'] = df_agr_emp.index
# drop the index
df_agr_emp.reset_index(drop=True, inplace=True)
# The index for Employment % in agriculture needs to be renamed to the actual
df_agr_emp=df_agr_emp.rename(columns={3: "Employment in Agriculture(%)"})

# sort the rows based in decreasing order of % Employment
df_agr_emp.sort_values('Employment in Agriculture(%)', inplace=True, ascendir

print("Employment in Agriculture data for 2015: ")
display(df_agr_emp)

# bar plot
plt.figure(figsize=(8, 5))
sns.set(style="whitegrid")
ax = sns.barplot(
    data= df_agr_emp,
    x= 'Countries',
    y= 'Employment in Agriculture(%)')
```

```
Employment in Agriculture data for 2015:
```

	Employment in Agriculture(%)	Countries
1	45.3	India
2	28.6	China
6	5.6	South Africa
3	3.6	Japan
4	1.6	Canada
0	1.4	USA
5	1.1	Great Britain



From the graph above, India has the biggest percentage of people employed in agriculture out of all countries. This reflects the fact that India is a predominantly agricultural country with numerous food products grown on its soil. India is followed by China. Due to a scarcity of arable land in China, farming is particularly labor intensive. As a result, agriculture employs many people. Agriculture employs a very small percentage of the population in the remainder of the world. The United Kingdom has the lowest percentage in agriculture, which is since it produces less than 60% of the food consumed.

Summary and Conclusion

A number of features in the World Bank dataset can be leveraged to build intriguing patterns in the data. Population counts, birth rate, and death rate were all analysed, and several fascinating patterns about how the birth rate and death rate affect the overall population were discovered. Similarly, it was discovered that India has the highest percentage of employment in agriculture, depending on the nature of the countries. Other studies of the relationship between power consumption and total population found that electrical power consumption is the highest in most countries when compared to their population.

```
In [ ]:
```