I will be using plotly express which is a wrapper for plotly with cufflinks for creating interactive graphs

Installing and Importing the Necessary Libraries

In [1]:
```
#pip install cufflinks
```

In [2]:
```
#pip install pandas-profiling
```

In [3]:
```
import pandas as pd
import pandas_profiling
import numpy as np
import cufflinks as cf
import plotly
import plotly.express as px
import plotly.graph_objects as go
import math

%matplotlib inline
#magic function for showing the plots in the same code cell and for additional
facilities
```

For working in offline mode and saving the plotly plots in local machine

In [4]:
```
from plotly.offline import plot,iplot,download_plotlyjs,init_notebook_mode
init_notebook_mode(connected=True)
cf.go_offline()
```

Q.1.1 - For visualizing the Co2 emission by the countries overtime

Reading the DataFrame

In [5]:
```
co2_df = pd.read_csv("D:\\Downloads\\global_co2_emissions.csv", parse_dates =
['year'])
```

In [6]:
```
#co2_df.profile_report()
```

I will be making a copy of the original dataframe for easy handling and checking for missing values

In [7]:
```
co2_dfc = co2_df.copy()


co2_dfc.isna().sum()
```

Out[7]:
```
country                         0
iso_code                     3371
year                            0
Annual CO2 emissions (tonnes )  0
dtype: int64
```

In [8]:
```
#co2_dfc1 = co2_df.copy()
```

Creating a separate dataframe for the missing values and checking for the unique values

In [9]:
```
missing_vals = co2_dfc[co2_dfc.isna().any(axis=1)]
missing_vals['country'].unique()
```

Out[9]:
```
array(['Africa', 'Asia', 'Asia (excl. China & India)', 'Europe',
       'Europe (excl. EU-27)', 'Europe (excl. EU-28)',
       'European Union (27)', 'European Union (28)',
       'French Equatorial Africa', 'French West Africa',
       'High-income countries', 'International transport',
       'Kuwaiti Oil Fires', 'Leeward Islands', 'Low-income countries',
       'Lower-middle-income countries', 'North America',
       'North America (excl. USA)', 'Oceania', 'Panama Canal Zone',
       'Ryukyu Islands', 'South America', 'St. Kitts-Nevis-Anguilla',
       'Upper-middle-income countries'], dtype=object)
```

In [10]:
```
co2_dfc.describe()
```

Out[10]:

	Annual CO2 emissions (tonnes)
count	2.467000e+04
mean	3.266583e+08
std	1.677027e+09
min	3.400000e+01
25%	5.569280e+05
50%	5.332958e+06
75%	4.815309e+07
max	3.670250e+10

Filling in the missing iso codes for Kyrgystan and W&F Islands so that we can use the iso codes to remove the non countries

In [11]:

```
df1 = co2_dfc [co2_dfc ["country"].isin(["Kyrgysztan"])]
df1.fillna('KGZ', inplace = True)


df2 = co2_dfc [co2_dfc["country"].isin(["Wallis and Futuna Islands"])]
df2.fillna('WLF', inplace =True)
```

Removing the null values for the rows which are not countries

In [12]:

```
co2_dfc.dropna(inplace=True)
```

In [13]:

```
co2_dfc = pd.concat([co2_dfc,df1,df2])
co2_dfc
```

Out[13]:

	country	iso_code	year	Annual CO2 emissions (tonnes)
0	Afghanistan	AFG	1949-01-01	14656
1	Afghanistan	AFG	1950-01-01	84272
2	Afghanistan	AFG	1951-01-01	91600
3	Afghanistan	AFG	1952-01-01	91600
4	Afghanistan	AFG	1953-01-01	106256
...	...	...	...	...
24665	Zimbabwe	ZWE	2016-01-01	10737567
24666	Zimbabwe	ZWE	2017-01-01	9581633
24667	Zimbabwe	ZWE	2018-01-01	11854367
24668	Zimbabwe	ZWE	2019-01-01	10949084
24669	Zimbabwe	ZWE	2020-01-01	10531342

21299 rows × 4 columns

In [14]:

```
#co2_dfc1 = co2_dfc1[co2_dfc1['country'] == 'World']
#co2_dfc1.reset_index(level=0,inplace=True)
#del co2_dfc1["index"]
#co2_dfc1
```

Removing the rows which contain the entire worlds data which is not needed

In [15]:

```
co2_dfc = co2_dfc[co2_dfc['country'] != 'World']
co2_dfc
```

Out[15]:

	country	iso_code	year	Annual CO2 emissions (tonnes)
0	Afghanistan	AFG	1949-01-01	14656
1	Afghanistan	AFG	1950-01-01	84272
2	Afghanistan	AFG	1951-01-01	91600
3	Afghanistan	AFG	1952-01-01	91600
4	Afghanistan	AFG	1953-01-01	106256
...	...	...	...	...
24665	Zimbabwe	ZWE	2016-01-01	10737567
24666	Zimbabwe	ZWE	2017-01-01	9581633
24667	Zimbabwe	ZWE	2018-01-01	11854367
24668	Zimbabwe	ZWE	2019-01-01	10949084
24669	Zimbabwe	ZWE	2020-01-01	10531342

21028 rows × 4 columns

In [16]:

```
co2_dfc['Annual CO2 emissions (tonnes )'].nlargest(n=10)
```

Out[16]:

```
4536    10667887453
4535    10489988555
4534    10289989525
4530     9985583382
4529     9952743755
4533     9920459189
4531     9848419740
4528     9775621803
4532     9720444086
4527     9528555734
Name: Annual CO2 emissions (tonnes ), dtype: int64
```

Storing the list of countries

In [103…

```
list_countries = co2_dfc['country'].unique()
#for i in list_countries:
    #print(i)
```

Grouping the countries together for further processing

In [18]:

```
country_group = co2_dfc.groupby('country')


country_afghanistan = country_group.get_group('Afghanistan')
country_afghanistan['Annual CO2 emissions (tonnes )'].max()
```

Out[18]:
```
12160286
```

Using a for loop to get the maximum co2 emitted by a country over the course of years

In [19]:
```
# Making empty lists to store the maximum c02 of countries and storing the corresponding country name
list_max= []
list_country_name = []

#begining the for loop
for i in list_countries:
    #will fetch the i th group from the "country_group"
    country_name = country_group.get_group(i)

    # will  take out the maximum co2 present in that group
    max_emm = country_name['Annual CO2 emissions (tonnes )'].max()

    # storing the values in the empty list
    list_max.append(max_emm)
    list_country_name.append(i)

print(len(list_max))
print(list_country_name)
```

```
222
['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola', 'Anguilla', 'Antarctica',
'Antigua and Barbuda', 'Argentina', 'Armenia', 'Aruba', 'Australia', 'Austria', 'Azer
baijan', 'Bahamas', 'Bahrain', 'Bangladesh', 'Barbados', 'Belarus', 'Belgium', 'Beliz
e', 'Benin', 'Bermuda', 'Bhutan', 'Bolivia', 'Bonaire Sint Eustatius and Saba', 'Bosn
ia and Herzegovina', 'Botswana', 'Brazil', 'British Virgin Islands', 'Brunei', 'Bulga
ria', 'Burkina Faso', 'Burundi', 'Cambodia', 'Cameroon', 'Canada', 'Cape Verde', 'Cen
tral African Republic', 'Chad', 'Chile', 'China', 'Christmas Island', 'Colombia', 'Co
moros', 'Congo', 'Cook Islands', 'Costa Rica', "Cote d'Ivoire", 'Croatia', 'Cuba', 'C
uracao', 'Cyprus', 'Czechia', 'Democratic Republic of Congo', 'Denmark', 'Djibouti',
'Dominica', 'Dominican Republic', 'Ecuador', 'Egypt', 'El Salvador', 'Equatorial Guin
ea', 'Eritrea', 'Estonia', 'Eswatini', 'Ethiopia', 'Faeroe Islands', 'Fiji', 'Finlan
d', 'France', 'French Guiana', 'French Polynesia', 'Gabon', 'Gambia', 'Georgia', 'Ger
many', 'Ghana', 'Greece', 'Greenland', 'Grenada', 'Guadeloupe', 'Guatemala', 'Guine
a', 'Guinea-Bissau', 'Guyana', 'Haiti', 'Honduras', 'Hong Kong', 'Hungary', 'Icelan
d', 'India', 'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy', 'Jamaica', 'J
apan', 'Jordan', 'Kazakhstan', 'Kenya', 'Kiribati', 'Kosovo', 'Kuwait', 'Kyrgyzstan',
'Laos', 'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya', 'Liechtenstein', 'Lithuan
ia', 'Luxembourg', 'Macao', 'Madagascar', 'Malawi', 'Malaysia', 'Maldives', 'Mali',
'Malta', 'Marshall Islands', 'Martinique', 'Mauritania', 'Mauritius', 'Mayotte', 'Mex
ico', 'Micronesia (country)', 'Moldova', 'Mongolia', 'Montenegro', 'Montserrat', 'Mor
occo', 'Mozambique', 'Myanmar', 'Namibia', 'Nauru', 'Nepal', 'Netherlands', 'New Cale
donia', 'New Zealand', 'Nicaragua', 'Niger', 'Nigeria', 'Niue', 'North Korea', 'North
Macedonia', 'Norway', 'Oman', 'Pakistan', 'Palau', 'Palestine', 'Panama', 'Papua New
Guinea', 'Paraguay', 'Peru', 'Philippines', 'Poland', 'Portugal', 'Puerto Rico', 'Qat
ar', 'Reunion', 'Romania', 'Russia', 'Rwanda', 'Saint Helena', 'Saint Kitts and Nevi
s', 'Saint Lucia', 'Saint Pierre and Miquelon', 'Saint Vincent and the Grenadines',
'Samoa', 'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia', 'Seychelles',
'Sierra Leone', 'Singapore', 'Sint Maarten (Dutch part)', 'Slovakia', 'Slovenia', 'So
lomon Islands', 'Somalia', 'South Africa', 'South Korea', 'South Sudan', 'Spain', 'Sr
i Lanka', 'Sudan', 'Suriname', 'Sweden', 'Switzerland', 'Syria', 'Taiwan', 'Tajikista
n', 'Tanzania', 'Thailand', 'Timor', 'Togo', 'Tonga', 'Trinidad and Tobago', 'Tunisi
a', 'Turkey', 'Turkmenistan', 'Turks and Caicos Islands', 'Tuvalu', 'Uganda', 'Ukrain
e', 'United Arab Emirates', 'United Kingdom', 'United States', 'Uruguay', 'Uzbekista
n', 'Vanuatu', 'Venezuela', 'Vietnam', 'Wallis and Futuna', 'Yemen', 'Zambia', 'Zimba
bwe']
```

Zipping the lists in a dataframe

In [104…

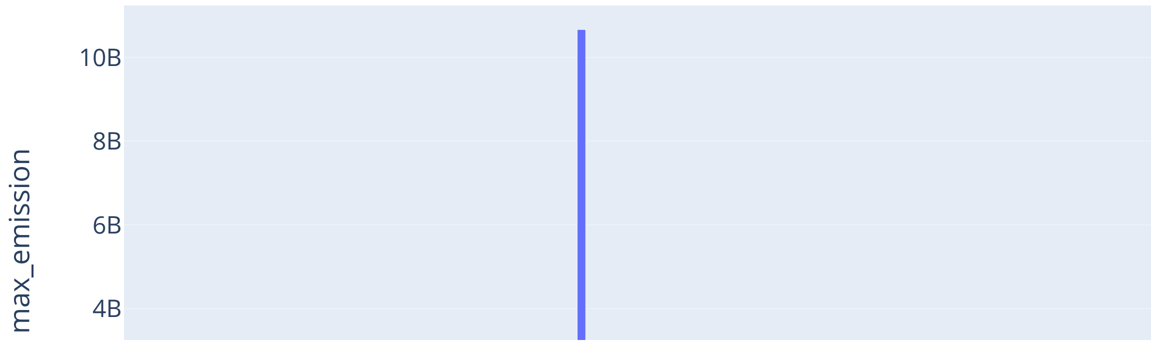
```python
max_emm_df = pd.DataFrame(list(zip(list_country_name,list_max)),columns=
['country','max_emission'])

# for printing the entire dataframe in a single cell output
#with pd.option_context('display.max_rows', None, 'display.max_columns', None):

print(max_emm_df)
```

```
              country  max_emission
0         Afghanistan      12160286
1             Albania       8976800
2             Algeria     166641950
3             Andorra        575248
4              Angola      33800624
..                ...           ...
217           Vietnam     260312093
218  Wallis and Futuna         29312
219             Yemen      24976297
220            Zambia       7313113
221          Zimbabwe      17393590

[222 rows x 2 columns]
```

For showing the barplot of emissions by top 10 countries

In [94]:

```
fig = px.bar(max_emm_df,x="country",y="max_emission")
fig.show()

#fig.write_html("D:\Downloads\GHG emissions HTML Plots\CO2 emissions of Bar plot of all the countries.html")
```



The Graphs have been distorted after converting to the pdf file. Please access the "HTML plots" .zip file and refer to the plot numbers.

Plot 1

The Above Graph shows the maximum co2 emitted by them in the entire history. If zoomed in to the graph we can see that china has emitted the maximum amount of co2 in history.

Taking the top 10 countries with the highest emission and storing it in top_10_list

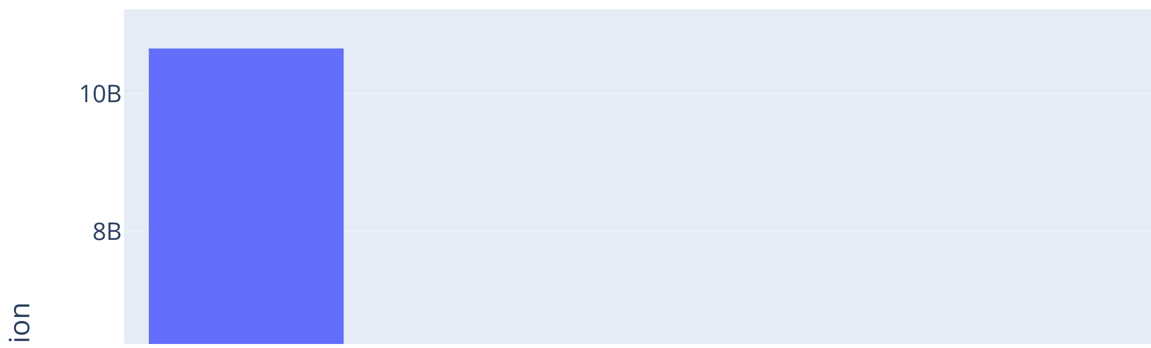
In [22]:
```
max_10 =max_emm_df.nlargest(10,'max_emission')
top_10_list = list(max_10['country'].unique())
top_10_list
```

Out[22]:
```
['China',
 'United States',
 'India',
 'Russia',
 'Japan',
 'Germany',
 'Iran',
 'Ukraine',
 'Saudi Arabia',
 'South Korea']
```

Here we can see the top 10 countries who emit the maximum amount of co2

In [95]:

```
fig2 = px.bar(max_10,x="country",y="max_emission")
fig2.show()

#fig.write_html("D:\Downloads\GHG emissions HTML Plots\CO2 emissions of TOP 10 countries.html")
```



Plot 2

Here the similar kind of graph is generated which also indicates China as the biggest emitter of co2 in history. Then US and India follow afterwards on the 2nd and 3rd positions. One of the reasons behind china's insanely high usage of co2 can be coupled with the booming industries in china which are mostly focused on making goods which are accountable in various sectors.

Now we can select these 10 countries and plot the co2 emitted by them over time

In [24]:

```
#Making an empty dataframe
top_10_country = pd.DataFrame()

for x in top_10_list:
```

```
    # Storing only those values in dataframe which come in top 10 countries over the years
    top_10 = co2_dfc[co2_dfc['country'] == x]
    top_10_country = top_10_country.append(top_10)
```

In [25]:

```
# resetting the index to the default values
top_10_country.reset_index(level=0,inplace=True)
del top_10_country["index"]
top_10_country
```

Out[25]:

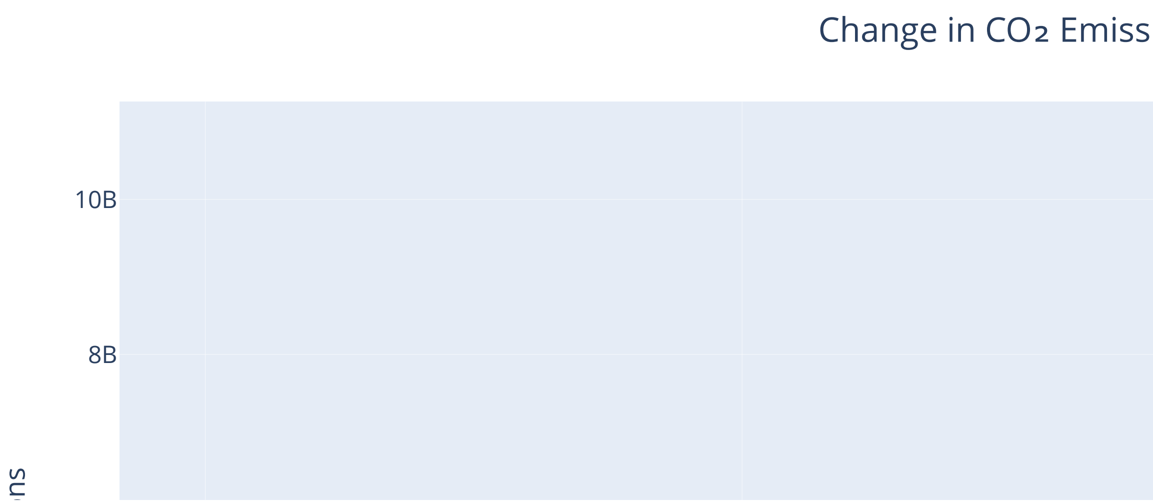
	country	iso_code	year	Annual CO2 emissions (tonnes)
0	China	CHN	1899-01-01	95264
1	China	CHN	1902-01-01	95264
2	China	CHN	1903-01-01	1963904
3	China	CHN	1904-01-01	2088480
4	China	CHN	1905-01-01	2297328
...	...	...	...	...
1514	South Korea	KOR	2016-01-01	639258641
1515	South Korea	KOR	2017-01-01	655747114
1516	South Korea	KOR	2018-01-01	671630709
1517	South Korea	KOR	2019-01-01	648024558
1518	South Korea	KOR	2020-01-01	597605055

1519 rows × 4 columns

Making the trend line plot for the top 10 emitter of co2 over the years

In [26]:

```
fig3 = px.line(top_10_country, x = "year", y = "Annual CO2 emissions (tonnes )", hover_name='country', hover_data= ['country','Annual CO2 emissions (tonnes )'],color='country', labels = {'country':'Country','Annual CO2 emissions (tonnes )': 'Co2 Emissions'}, height=600)
fig3.update_layout(title="Change in CO₂ Emission Between Years 1750 and 2020 - Countries",title_x=0.50)
fig3.update_layout(showlegend = False)
fig3.update(layout_coloraxis_showscale = True)
fig3.show()
```



The trend line shows the behaviour of top 10 countries in the matter of co2 emissions and we can see a certain spike in china's emissions whereas US had a consistent rise in co2 since the industrial age and a downward dip in the recent years due to the increase in the use of sustainable and renewable sources of energy. However for India there is a spike in recent years due to the booming population and exhaustive use of fossil fuels.

Saving the plot to an html file

In [27]:
```
#fig3.write_html("D:/Downloads/file_name.html")
```

Q.1.2 - For finding out the Co2 emissions per capita

In [28]:

```python
global_co2 = pd.read_csv("D:\\Downloads\\global-co2-data.csv")
```

In [29]:

```python
per_capita = global_co2[['country','iso_code','year','co2_per_capita']]
per_capita
```

Out[29]:

	country	iso_code	year	co2_per_capita
0	Afghanistan	AFG	1949	0.002
1	Afghanistan	AFG	1950	0.011
2	Afghanistan	AFG	1951	0.012
3	Afghanistan	AFG	1952	0.012
4	Afghanistan	AFG	1953	0.013
...	...	...	...	...
23703	Zimbabwe	ZWE	2015	0.881
23704	Zimbabwe	ZWE	2016	0.771
23705	Zimbabwe	ZWE	2017	0.720
23706	Zimbabwe	ZWE	2018	0.785
23707	Zimbabwe	ZWE	2019	0.708

23708 rows × 4 columns

In [30]:

```python
per_capita.isna().sum()
```

Out[30]:

```
country              0
iso_code          2778
year                 0
co2_per_capita    1328
dtype: int64
```

In [31]:

```python
per_capita['co2_per_capita'].mean()
```

Out[31]:

```
4.059418990169768
```

We see that the mean is around 4. But this is for global average and hence it will be not a good practice to input all the missing values with the global mean. Hence I decided to make a function which can take the mean of co2 per capita for each country over the years and input the missing values present in that country with its mean itself.

In [32]:

```python
# demo function
emp = []
for i in per_capita['co2_per_capita'].isna():
```

```
    if i == True:
        emp.append(i)
len(emp)
```

Out[32]:
```
1328
```

In [33]:
```
null_vals = per_capita[per_capita.isna().any(axis=1)]
null_vals
```

Out[33]:

	country	iso_code	year	co2_per_capita
71	Africa	NaN	1884	0.000
72	Africa	NaN	1885	0.000
73	Africa	NaN	1886	0.000
74	Africa	NaN	1887	0.000
75	Africa	NaN	1888	0.001
...	...	...	...	...
23326	World	OWID_WRL	1895	NaN
23327	World	OWID_WRL	1896	NaN
23328	World	OWID_WRL	1897	NaN
23329	World	OWID_WRL	1898	NaN
23330	World	OWID_WRL	1899	NaN

3672 rows × 4 columns

In [105…
```
#taking the unique country values from the per_capita dataframe
list_country = per_capita['country'].unique()

#grouping the countries
capita_country_grp = per_capita.groupby('country')

#making an empty dataframe
per_capita_not_null = pd.DataFrame()

for i in list_country:
    # getting the i th group - lets suppose its afganistan, then it will fetch the group of afganistan
    countries = capita_country_grp.get_group(i)
```

```
    #checking whether the values are missing in the group of afganistan
    for x in countries['co2_per_capita'].isna():
        if x == True:
            # if x is true i.e the value is missing then, it will take the mean of co2 per
capita and replace it in the missing positions
            countries['co2_per_capita'].fillna(countries['co2_per_capita'].mean(),inplace =
True)
    per_capita_not_null = per_capita_not_null.append(countries)

# diplaying the entire dataframe
#with pd.option_context('display.max_rows', None, 'display.max_columns', None):
print(per_capita_not_null)

per_capita_not_null.isna().sum()
#per_capita_not_null.to_csv("D:\\Downloads\\per.csv")
```

```
           country iso_code  year  co2_per_capita
0      Afghanistan      AFG  1949           0.002
1      Afghanistan      AFG  1950           0.011
2      Afghanistan      AFG  1951           0.012
3      Afghanistan      AFG  1952           0.012
4      Afghanistan      AFG  1953           0.013
...            ...      ...   ...             ...
23703     Zimbabwe      ZWE  2015           0.881
23704     Zimbabwe      ZWE  2016           0.771
23705     Zimbabwe      ZWE  2017           0.720
23706     Zimbabwe      ZWE  2018           0.785
23707     Zimbabwe      ZWE  2019           0.708

[23708 rows x 4 columns]
```

Out[105]:

```
country              0
iso_code          2778
year                 0
co2_per_capita     369
dtype: int64
```

Now there will still be missing values in the co2 per capita column as there are countries for which the data is not present over the entire time range. So the mean will not be calculated for these countries and it will not be filled up.So we can just drop these countries.

In [35]:

```
per_capita_not_null.dropna(inplace=True)
```

In [36]:

```
kryg = per_capita_not_null[per_capita_not_null["country"].isin(["Kyrgysztan"])]
kryg.fillna('KGZ', inplace = True)

per_capita_clean = pd.concat([per_capita_not_null,kryg])

# dropping the countries which have outliers in their  co2 per capita column
per_capita_clean = per_capita_clean[per_capita_clean['country'] != 'World']
per_capita_clean = per_capita_clean[per_capita_clean['country'] != 'Sint Maarten
(Dutch part)']
per_capita_clean = per_capita_clean[per_capita_clean['country'] != 'Brunei']
per_capita_clean.isna().sum()
per_capita_clean
```

Out[36]:

	country	iso_code	year	co2_per_capita
0	Afghanistan	AFG	1949	0.002
1	Afghanistan	AFG	1950	0.011
2	Afghanistan	AFG	1951	0.012
3	Afghanistan	AFG	1952	0.012
4	Afghanistan	AFG	1953	0.013
...	...	...	...	...
23703	Zimbabwe	ZWE	2015	0.881
23704	Zimbabwe	ZWE	2016	0.771
23705	Zimbabwe	ZWE	2017	0.720
23706	Zimbabwe	ZWE	2018	0.785
23707	Zimbabwe	ZWE	2019	0.708

20311 rows × 4 columns

In [37]:

```
fig4 = px.line (per_capita_clean, x = "year", y = "co2_per_capita",
hover_name='country',
            hover_data= ['country' , 'co2_per_capita'],color='country',
            labels = {'country':'Country','co2_per_capita': 'Co2 Per Capita'},
            height=600)

fig4.update_layout ( title="Co2 per Capita of Countries",title_x=0.50)
fig4.update_layout (showlegend = False)
```

Out[38]:

	country	iso_code	year	co2_per_capita
0	Afghanistan	AFG	1949	0.002
1	Afghanistan	AFG	1950	0.011
2	Afghanistan	AFG	1951	0.012
3	Afghanistan	AFG	1952	0.012
4	Afghanistan	AFG	1953	0.013
...	...	...	...	...
23703	Zimbabwe	ZWE	2015	0.881
23704	Zimbabwe	ZWE	2016	0.771
23705	Zimbabwe	ZWE	2017	0.720
23706	Zimbabwe	ZWE	2018	0.785
23707	Zimbabwe	ZWE	2019	0.708

20311 rows × 4 columns

Now we will be using the same function approach for choosing the top 10 countries which have the highest c02 per capita as we cant just use the top 10 list obtained before as the countries who emit co2 over time will be different than the countries who have highest c02 per capita as per capita calculation depends on the population of the country

In [39]:

```
capita_countries = per_capita_clean['country'].unique()


group_countries = per_capita_clean.groupby('country')


list_capitas= []
list_capita_country = []
for k in capita_countries:
    name_of_country = group_countries.get_group(k)
    max_cap = name_of_country['co2_per_capita'].max()
    list_capitas.append(max_cap)
    list_capita_country.append(k)


print(list_capitas)
```

```
[0.402, 2.885, 3.988, 8.061, 1.642, 11.556, 19.637, 4.693, 5.824, 27.933, 19.276, 9.5
94, 8.327, 49.283, 40.348, 0.627, 5.758, 12.311, 14.262, 2.129, 0.678, 12.794, 2.237,
1.968, 88.84, 8.065, 3.279, 2.584, 7.806, 10.186, 0.212, 0.051, 0.972, 0.693, 18.705,
1.234, 0.115, 0.09, 4.649, 7.096, 2.092, 0.297, 1.088, 4.303, 1.807, 5.696, 3.463, 8.
047, 18.269, 0.251, 14.24, 1.132, 2.58, 2.549, 2.709, 2.595, 1.121, 10.486, 0.392, 2
3.583, 0.148, 16.215, 2.536, 13.9, 9.957, 3.342, 10.917, 0.253, 5.668, 14.252, 0.581,
10.305, 14.436, 2.816, 1.167, 0.249, 0.242, 3.053, 0.303, 1.121, 6.524, 8.542, 12.29
9, 1.916, 2.282, 9.402, 5.632, 12.391, 10.398, 8.622, 4.327, 10.246, 3.544, 17.448,
0.379, 0.702, 74.98, 5.649, 4.577, 7.321, 4.409, 1.208, 1.107, 17.379, 7.168, 10.238,
41.105, 0.224, 0.135, 8.073, 3.14, 0.178, 7.793, 2.764, 1.725, 3.869, 5.961, 6.891, 2
0.35, 4.195, 13.19, 1.972, 0.391, 0.486, 1.749, 17.913, 0.486, 13.292, 29.864, 9.086,
0.949, 0.147, 1.009, 5.496, 10.657, 7.206, 9.797, 16.778, 1.166, 0.766, 2.964, 0.898,
1.174, 1.969, 1.334, 13.037, 6.697, 98.928, 9.103, 17.117, 0.122, 2.198, 4.789, 2.29
9, 17.099, 2.884, 1.449, 0.604, 20.348, 0.675, 6.654, 8.465, 0.403, 18.139, 12.248,
9.006, 0.655, 0.166, 9.95, 12.408, 0.181, 8.394, 1.165, 0.537, 6.681, 11.457, 7.342,
3.334, 11.939, 2.536, 0.214, 4.212, 0.437, 0.523, 1.686, 35.36, 2.652, 5.243, 14.414,
6.255, 1.099, 0.153, 14.144, 101.022, 11.846, 22.133, 2.543, 6.763, 1.067, 17.031, 2.
568, 1.939, 1.635, 1.895]
```

In [106…

```python
max_cap_df = pd.DataFrame(list(zip(list_capita_country,list_capitas)),columns=
['country','co2_per_capita'])


#with pd.option_context('display.max_rows', None, 'display.max_columns', None):
print(max_cap_df)
```

```
        country  co2_per_capita
0    Afghanistan           0.402
1        Albania           2.885
2        Algeria           3.988
3        Andorra           8.061
4         Angola           1.642
..           ...             ...
199    Venezuela          17.031
200      Vietnam           2.568
201        Yemen           1.939
202       Zambia           1.635
203     Zimbabwe           1.895

[204 rows x 2 columns]
```

In [41]:

```python
max_capita_10 =max_cap_df.nlargest(10,'co2_per_capita')
top_10_capitas = list(max_capita_10['country'].unique())
top_10_capitas
```

Out[41]:
```
['United Arab Emirates',
 'Qatar',
 'Bonaire Sint Eustatius and Saba',
 'Kuwait',
 'Bahamas',
 'Luxembourg',
 'Bahrain',
 'Trinidad and Tobago',
 'New Caledonia',
 'Aruba']
```

In [42]:
```
capitas_10_countries = pd.DataFrame()


for j in top_10_capitas:
    capitas= per_capita_clean[per_capita_clean['country'] == j]
    capitas_10_countries = capitas_10_countries.append(capitas)
```

In [43]:
```
capitas_10_countries.reset_index(level=0,inplace=True)
del capitas_10_countries["index"]
capitas_10_countries
```

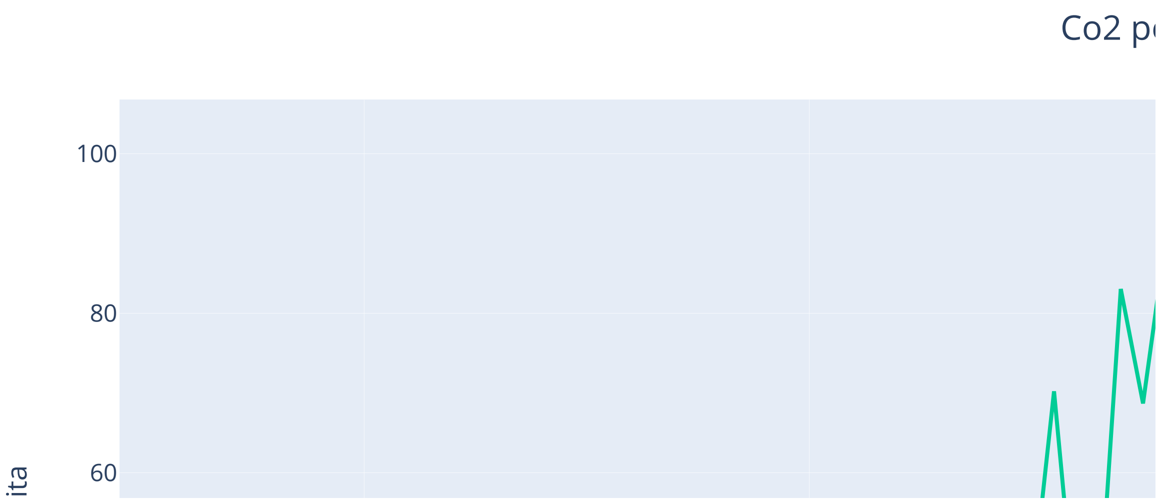
Out[43]:

	country	iso_code	year	co2_per_capita
0	United Arab Emirates	ARE	1959	0.126
1	United Arab Emirates	ARE	1960	0.119
2	United Arab Emirates	ARE	1961	0.109
3	United Arab Emirates	ARE	1962	0.164
4	United Arab Emirates	ARE	1963	0.176
...	...	...	...	...
802	Aruba	ABW	2015	8.632
803	Aruba	ABW	2016	8.410
804	Aruba	ABW	2017	8.724
805	Aruba	ABW	2018	8.898
806	Aruba	ABW	2019	8.666

807 rows × 4 columns

In [96]:
```
fig5 = px.line(capitas_10_countries, x = "year", y = "co2_per_capita",
hover_name='country',
            hover_data= ['country','co2_per_capita'],color='country',
            labels = {'country':'Country','co2_per_capita': 'Co2 per capita'}, height=600)
```

```
fig5.update_layout(title="Co2 per Capita for Top 10 Countries",title_x=0.50)
fig5.update_layout(showlegend = False)
fig5.update(layout_coloraxis_showscale = True)

fig5.show()

#fig5.write_html("D:\Downloads\GHG emissions HTML Plots\CO2 per Capita of TOP 10 countries.html")
```



Plot 4

In the above graph we see that the major countries that have the highest co2 per capita are the countries that produce the most amount of fossil fuels such as crude oil. And these include the countries which come in the Middle East and they have very low population as compared with the amount of fossil fuels they extract. In 1969 UAE had the highest co2 per capita with 101 tonnes per person. And the highest recorded co2 per capita for Qatar was in 1963.

In [45]:
```
top_10_country['year'] = top_10_country['year'].astype(str)

pd.to_datetime(top_10_country['year'],errors='ignore')

top_10_country = top_10_country.loc[top_10_country['year'] > '1900-01-01' ]
```

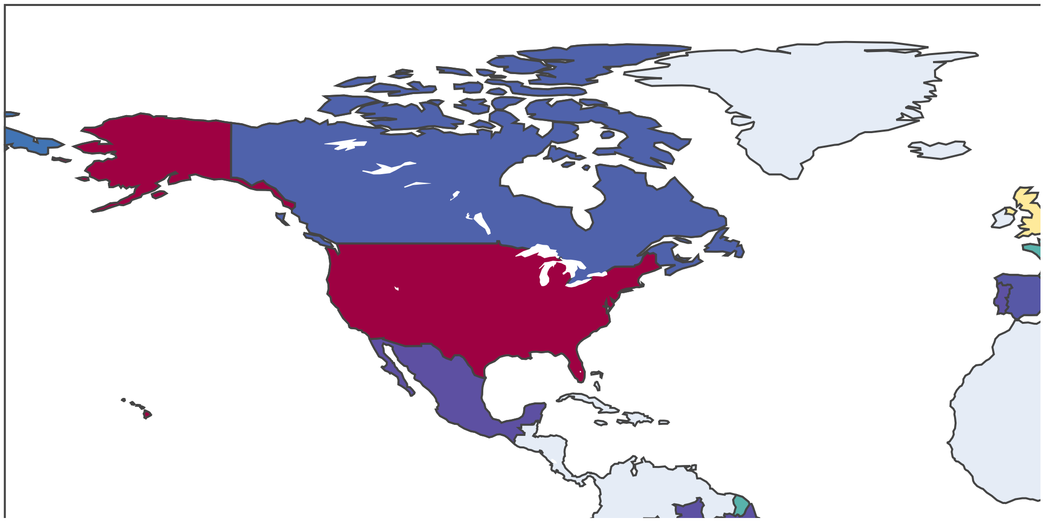
Q.1.3 For visualizing how much they have emitted overtime

Filtering the timelime to be later than the year 1900

In [46]:
```
co2_dfc['year'] = co2_dfc['year'].astype(str)

pd.to_datetime(co2_dfc['year'],errors='ignore')

co2_dfc3 = co2_dfc.loc[co2_dfc['year'] > '1900-01-01' ]
```

In [97]:
```
fig6 = px.choropleth(co2_dfc3.groupby(['country' , 'year'])['Annual CO2 emissions (tonnes )'].sum().reset_index().sort_values(by=['year'],ascending = True),
                    locations = 'country',
                    locationmode='country names',
                    color = 'Annual CO2 emissions (tonnes )',
                    color_continuous_scale='Spectral_r',
                    height=800,
                    animation_frame='year',
                    animation_group='country')

fig6.update_layout(title = ' Co2 emissions by all the countries overtime')
fig6.show()

#fig6.write_html("D:\Downloads\GHG emissions HTML Plots\CO2 emmisions animation map.html")
```

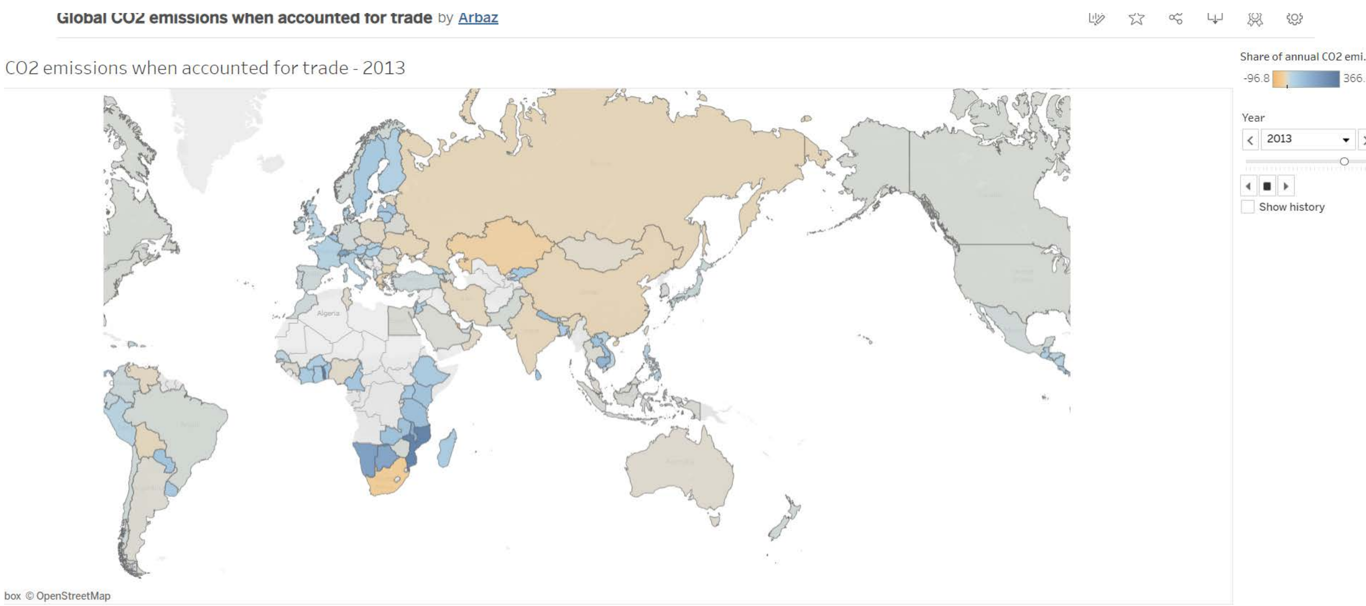
Co2 emissions by all the countries overtime



Plot 5

The animated map depicts the chann co2 in tonnes over time from the year 1900 to 2020. The countries which are in blue show a relatively low emission of co2 in comparison to the countries in red such as USA.

Q.1.4 How do emissions compare when we correct for trade?
Done on Tableau

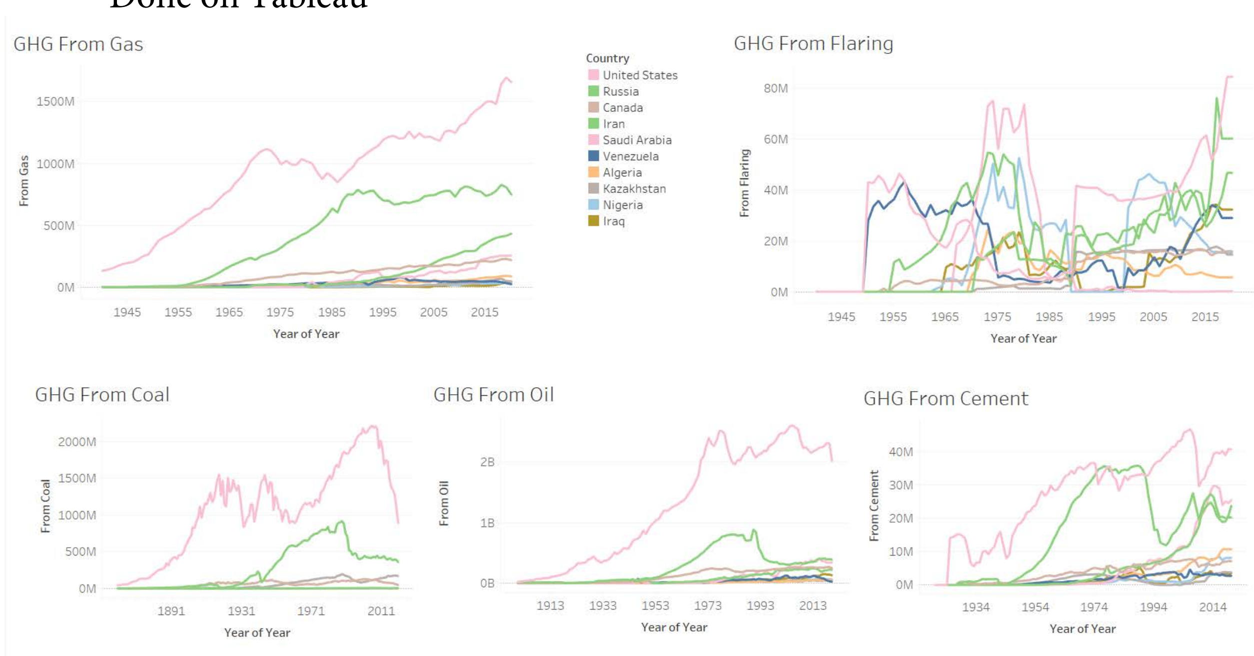


Click here for tableau link

Countries having higher positive values are net importers of co2 i.e they import more co2 than they export
Countries having higher negative values are net exporters of co2, they export more co2 than they import

The interactive map shows that US is a net importer of co2 and china is the net exporter of co2

Q.2.1 How much CO2 comes from coal, oil, gas, and flaring or cement production?
Done on Tableau

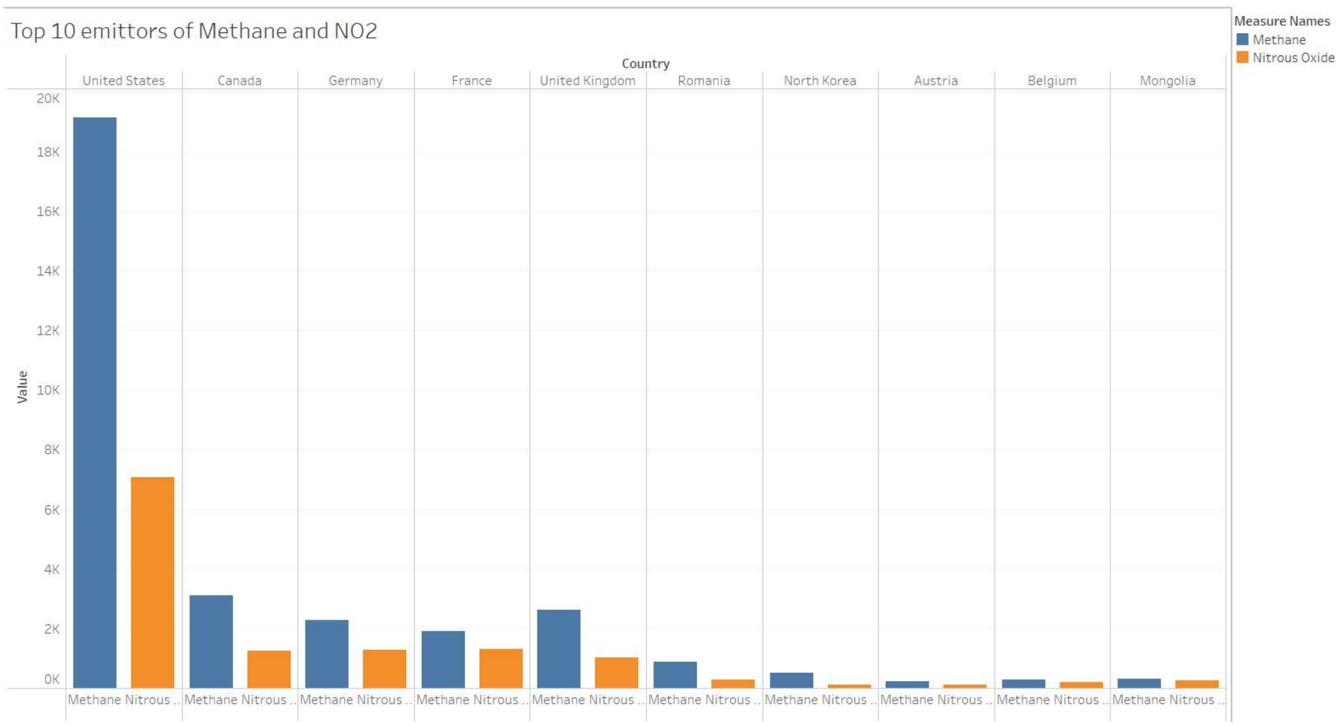


Click here for Tableau link

There are many sources from where greenhouse gases can arise. The interactive graphs of the top 10 countries depicts the amount of co2 emitted in the atmosphere over time. The industrial revolution kickstarted the increase in the amount of co2 from coal and oil. Even till now, many industries rely heavily on these resources of energy.
Cement and Flaring also contribute significantly to the co2 emissions and african countries are the major stakeholders in these domains.
Talking about the more recent past, all of the sources of co2 contribute fairly equally to the global co2 amount.

Q.3.2 How much methane and nitrous oxide is emitted?

Done on tableau



Click here for tableau link

This is a comparative bar graph of Methane vs NO2 for the top 10 emitting cointries. It is seen as America comes as the highest emitter of these GHGs with around 19000 tonnes of Methane and 7000 tonnes of NO2. One of the big reasons is the high beef consumption in the USA. The livestock such as cows can produce upto 30 tonnes in their lifetime and due to the vast amount of cows in many livestock farms around the country. And NO2 comes from factories and the usage of transportation vehicles in the USA.

In [81]:
```
ghg_emissions = pd.read_csv("D:\\Downloads\\global-ghg-data.csv",parse_dates=
['year'])
```

In [82]:
```
ghg_emissions
ghg_emissions['land-use-change-forestry'].dtype
```

Out[82]:
```
dtype('float64')
```

Substituting the values which are lesser than zero by zero as we are only focussing on the contribution of these sectors to emissions.

In [107…
```
num = ghg_emissions._get_numeric_data()
num[num<0] = 0

#with pd.option_context('display.max_rows', None, 'display.max_columns', None):
print(ghg_emissions)
```

Q.4.1 Which sector contributes most to the GHG emissions

```
          country iso_code       year  agriculture  land-use-change-forestry  \
0     Afghanistan      AFG 1990-01-01    8070000.0                       0.0   
1     Afghanistan      AFG 1991-01-01    8400000.0                       0.0   
2     Afghanistan      AFG 1992-01-01    8410000.0                       0.0   
3     Afghanistan      AFG 1993-01-01    8490000.0                       0.0   
4     Afghanistan      AFG 1994-01-01    8520000.0                       0.0   
...           ...      ...        ...          ...                       ...   
5650     Zimbabwe      ZWE 2014-01-01   10190000.0                11490000.0   
5651     Zimbabwe      ZWE 2015-01-01   11470000.0                11610000.0   
5652     Zimbabwe      ZWE 2016-01-01   10540000.0                87400000.0   
5653     Zimbabwe      ZWE 2017-01-01   10780000.0                87290000.0   
5654     Zimbabwe      ZWE 2018-01-01   11150000.0                87380000.0   

          waste  industry  manufacturing-and-construction  transport  \
0     1230000.0     50000                        570000.0  1670000.0   
1     1320000.0     50000                        530000.0  1550000.0   
2     1400000.0     60000                        390000.0   770000.0   
3     1490000.0     60000                        380000.0   740000.0   
4     1580000.0     70000                        360000.0   710000.0   
...         ...       ...                             ...        ...   
5650  2380000.0   1530000                       1070000.0  2640000.0   
5651  2430000.0   1580000                       1090000.0  2570000.0   
5652  2480000.0   1720000                       1090000.0  2180000.0   
5653  2540000.0   1790000                       1120000.0  2240000.0   
5654  2590000.0   1850000                       1190000.0  2870000.0   

      electricity  buildings  fugitive-emissions  other-fuel  \
0        270000.0    80000.0            610000.0   2630000.0   
1        270000.0    70000.0            520000.0   2400000.0   
2        160000.0    30000.0            220000.0   2180000.0   
3        160000.0    30000.0            160000.0   1950000.0   
4        160000.0    20000.0            120000.0   1720000.0   
...           ...        ...                 ...         ...   
5650    6850000.0   190000.0            600000.0   3890000.0   
5651    7210000.0   220000.0            660000.0   4060000.0   
5652    6220000.0   230000.0            680000.0   3980000.0   
5653    5410000.0   220000.0            700000.0   4000000.0   
5654    6600000.0   230000.0            710000.0   4190000.0   

      aviation-and-shipping  
0                   20000.0  
1                   20000.0  
2                   20000.0  
3                   20000.0  
4                   20000.0  
...                     ...  
5650                50000.0  
5651               100000.0  
5652               150000.0  
5653               180000.0  
5654               220000.0  
```

```
[5655 rows x 14 columns]
```

Choosing the data of the entire world as our focus point as we want to see the cummulative contribution of each sector to emissions

In [85]:
```
ghg_world= ghg_emissions[ghg_emissions['country'] == 'World']
ghg_world.reset_index(level=0,inplace=True)
del ghg_world["index"]
ghg_world
```

Out[85]:

	country	iso_code	year	agriculture	land-use-change-forestry	waste	industry	manufacturir ar constructi
0	World	OWID_WRL	1990-01-01	4.997830e+09	1.909290e+09	1.364400e+09	1010440000	3.955390e+
1	World	OWID_WRL	1991-01-01	4.988460e+09	1.909290e+09	1.395180e+09	1014310000	3.875910e+
2	World	OWID_WRL	1992-01-01	4.966820e+09	1.909290e+09	1.418280e+09	1030400000	3.743480e+
3	World	OWID_WRL	1993-01-01	4.936150e+09	1.909290e+09	1.444390e+09	1044440000	3.694800e+
4	World	OWID_WRL	1994-01-01	4.981350e+09	1.909320e+09	1.470960e+09	1151620000	3.711410e+
5	World	OWID_WRL	1995-01-01	5.038180e+09	1.915150e+09	1.476510e+09	1225100000	3.937600e+
6	World	OWID_WRL	1996-01-01	5.057350e+09	1.711370e+09	1.478310e+09	1277110000	3.827310e+
7	World	OWID_WRL	1997-01-01	4.986720e+09	2.681330e+09	1.474360e+09	1319750000	3.852360e+
8	World	OWID_WRL	1998-01-01	5.042290e+09	2.011150e+09	1.466280e+09	1323480000	3.854060e+
9	World	OWID_WRL	1999-01-01	5.098820e+09	1.819850e+09	1.464470e+09	1329140000	3.694740e+
10	World	OWID_WRL	2000-01-01	5.094120e+09	1.670960e+09	1.466760e+09	1388470000	3.874930e+
11	World	OWID_WRL	2001-01-01	5.105040e+09	1.338220e+09	1.452910e+09	1413160000	3.897350e+
12	World	OWID_WRL	2002-01-01	5.164050e+09	1.853110e+09	1.447970e+09	1479760000	3.896220e+
13	World	OWID_WRL	2003-01-01	5.158070e+09	1.545550e+09	1.443660e+09	1555420000	4.083280e+
14	World	OWID_WRL	2004-01-01	5.271690e+09	1.909320e+09	1.431880e+09	1654810000	4.515140e+
15	World	OWID_WRL	2005-01-01	5.307630e+09	1.631340e+09	1.422660e+09	1737330000	4.927560e+
16	World	OWID_WRL	2006-01-01	5.365470e+09	2.005130e+09	1.433850e+09	1868580000	5.176290e+
17	World	OWID_WRL	2007-01-01	5.450030e+09	1.471650e+09	1.441860e+09	1993860000	5.450180e+
18	World	OWID_WRL	2008-01-01	5.460100e+09	1.432930e+09	1.446930e+09	2039580000	5.563140e+
19	World	OWID_WRL	2009-01-01	5.455290e+09	1.825130e+09	1.454290e+09	2089210000	5.547260e+

	country	iso_code	year	agriculture	land-use-change-forestry	waste	industry	manufacturing-and-constructi
20	World	OWID_WRL	2010-01-01	5.515230e+09	1.490210e+09	1.465130e+09	2223080000	6.087720e+
21	World	OWID_WRL	2011-01-01	5.649500e+09	4.046800e+08	1.467310e+09	2371510000	6.313540e+
22	World	OWID_WRL	2012-01-01	5.677850e+09	4.326400e+08	1.476630e+09	2450410000	6.332910e+
23	World	OWID_WRL	2013-01-01	5.621260e+09	3.882600e+08	1.484040e+09	2556160000	6.324160e+
24	World	OWID_WRL	2014-01-01	5.669750e+09	7.379400e+08	1.514260e+09	2671420000	6.360380e+
25	World	OWID_WRL	2015-01-01	5.691560e+09	7.864600e+08	1.543590e+09	2678620000	6.315810e+
26	World	OWID_WRL	2016-01-01	5.737280e+09	1.267610e+09	1.560850e+09	2768080000	6.188610e+
27	World	OWID_WRL	2017-01-01	5.821070e+09	1.220050e+09	1.583860e+09	2825880000	6.174410e+
28	World	OWID_WRL	2018-01-01	5.817650e+09	1.387560e+09	1.606860e+09	2902680000	6.158320e+

In [54]:

```
ghg_world.isna().sum()
```

Out[54]:

```
country                         0
iso_code                        0
year                            0
agriculture                     0
land-use-change-forestry        0
waste                           0
industry                        0
manufacturing-and-construction  0
transport                       0
electricity                     0
buildings                       0
fugitive-emissions              0
other-fuel                      0
aviation-and-shipping           0
dtype: int64
```

In [98]:

```
fig_world = px.line(ghg_world, x = 'year', y = ['land-use-change-forestry' , 'waste' ,
'industry' , 'transport' , 'manufacturing-and-construction' ,
                                              'electricity', 'buildings', 'fugitive-emissions', 'other-fuel' ,
'aviation-and-shipping' ],
             hover_data= ['value'],
```

```
                labels = {'value': 'GHG emissions'}, height=600)

fig_world.update_layout(title="Sector wise contribution to the Global GHG
Emissions",yaxis_title = "GHG emissions in Tonnes",
                   title_x=0.50)
fig_world.update_layout(showlegend = False)
fig_world.update(layout_coloraxis_showscale = True)

fig_world.show()

#fig_world.write_html("D:\Downloads\GHG emissions HTML Plots\Sector Wise
Distribution of GHG Emissions.html")
```

Plot 6

This graph is very essential as it indcates from where most of the ghg emissions come from. We can generate the same graph for a particular country if a country wants to suppress the co2 coming from a particular sector but it is more sensible to see the sector wise behaviour from a global perspective. We see that heat and electricity are the largest contributors to the emissions. The least amount of co2 comes from aviation and shipping. The contributions from a particular country may change the behaviour of this graph as some countries specialize in a particular sector than others. Which can be shown as:

In [89]:

```
ghg_china= ghg_emissions[ghg_emissions['country'] == 'China']
ghg_china.reset_index(level=0,inplace=True)
del ghg_china["index"]
ghg_china
```

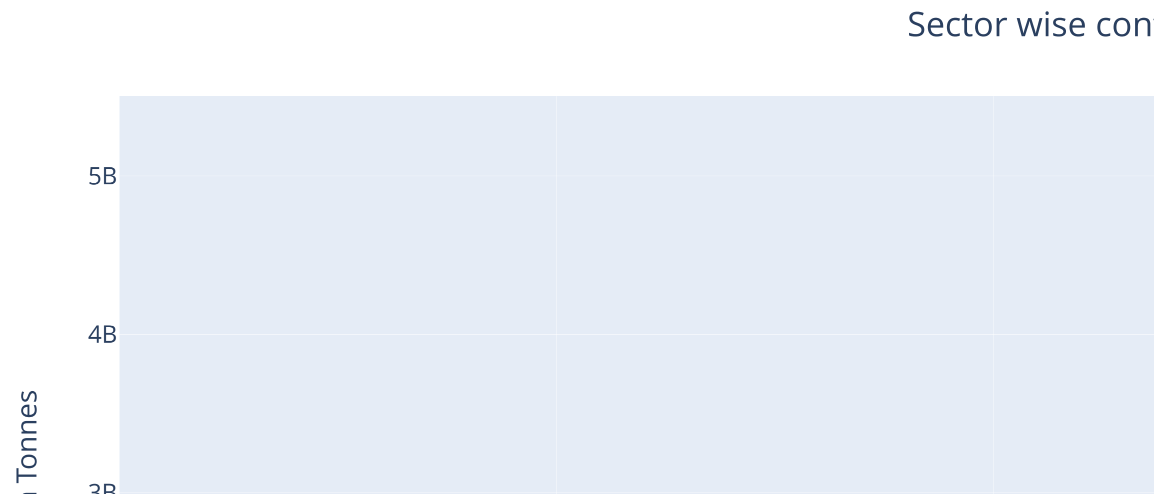
Out[89]:

	country	iso_code	year	agriculture	land-use-change-forestry	waste	industry	manufacturing-and-construction	tra
0	China	CHN	1990-01-01	590560000.0	0.0	194710000.0	94350000	7.452000e+08	9415
1	China	CHN	1991-01-01	600810000.0	0.0	199460000.0	112600000	7.788300e+08	10062
2	China	CHN	1992-01-01	605370000.0	0.0	204210000.0	135160000	8.078000e+08	1111(
3	China	CHN	1993-01-01	592670000.0	0.0	208960000.0	157020000	8.615400e+08	1257(
4	China	CHN	1994-01-01	612720000.0	0.0	213710000.0	180080000	9.033800e+08	11635
5	China	CHN	1995-01-01	671870000.0	0.0	205550000.0	204110000	1.070170e+09	12737
6	China	CHN	1996-01-01	714640000.0	0.0	197380000.0	219900000	9.686600e+08	16985
7	China	CHN	1997-01-01	646420000.0	0.0	189200000.0	237400000	9.611300e+08	14613
8	China	CHN	1998-01-01	669110000.0	0.0	181030000.0	253390000	1.023820e+09	14029
9	China	CHN	1999-01-01	689040000.0	0.0	172860000.0	277780000	8.714900e+08	15736
10	China	CHN	2000-01-01	677060000.0	0.0	164690000.0	301370000	9.063800e+08	24848
11	China	CHN	2001-01-01	666040000.0	0.0	156220000.0	344630000	9.613300e+08	25425
12	China	CHN	2002-01-01	671530000.0	0.0	147740000.0	387290000	9.995400e+08	27644
13	China	CHN	2003-01-01	660990000.0	0.0	139270000.0	452750000	1.138750e+09	31314
14	China	CHN	2004-01-01	679510000.0	0.0	130790000.0	504420000	1.500190e+09	37114
15	China	CHN	2005-01-01	687280000.0	0.0	122320000.0	552880000	1.935870e+09	39731
16	China	CHN	2006-01-01	689390000.0	0.0	128340000.0	632320000	2.083320e+09	43481
17	China	CHN	2007-01-01	680840000.0	0.0	134370000.0	698160000	2.274350e+09	46858
18	China	CHN	2008-01-01	693160000.0	0.0	140390000.0	730000000	2.409860e+09	50707
19	China	CHN	2009-01-01	694650000.0	0.0	146410000.0	808640000	2.601830e+09	51708

	country	iso_code	year	agriculture	land-use-change-forestry	waste	industry	manufacturing-and-construction	tra
20	China	CHN	2010-01-01	695410000.0	0.0	152440000.0	885580000	2.844450e+09	56878
21	China	CHN	2011-01-01	689640000.0	0.0	158460000.0	975360000	3.005390e+09	62189
22	China	CHN	2012-01-01	691170000.0	0.0	164480000.0	1002330000	3.038400e+09	68613
23	China	CHN	2013-01-01	690840000.0	0.0	169550000.0	1058980000	3.029290e+09	74109
24	China	CHN	2014-01-01	691800000.0	0.0	174620000.0	1112430000	3.022680e+09	77056
25	China	CHN	2015-01-01	698730000.0	0.0	179680000.0	1090680000	2.972980e+09	82714
26	China	CHN	2016-01-01	699820000.0	0.0	185650000.0	1122480000	2.846380e+09	84342
27	China	CHN	2017-01-01	684300000.0	0.0	191610000.0	1144490000	2.734180e+09	88090
28	China	CHN	2018-01-01	672870000.0	0.0	197570000.0	1166290000	2.667430e+09	91702

In [99]:

```
fig_china = px.line(ghg_china, x = 'year', y = ['land-use-change-forestry' , 'waste' ,
'industry' , 'transport' , 'manufacturing-and-construction' ,
                                     'electricity', 'buildings', 'fugitive-emissions', 'other-fuel' ,
'aviation-and-shipping' ],
                hover_data= ['value'],
                labels = {'value': 'GHG emissions'}, height=600)

fig_china.update_layout(title="Sector wise contribution to the Chinese GHG
Emissions",yaxis_title = "GHG emissions in Tonnes",
                   title_x=0.50)
fig_china.update_layout(showlegend = False)
fig_china.update(layout_coloraxis_showscale = True)

fig_china.show()

#fig_china.write_html("D:\Downloads\GHG emissions HTML Plots\Sectorwise
distribution of GHG in China.html")
```



Plot 7

This shows the sector wise contribution of ghg emissions in China. Here, contrary to the world graph, electricty and manufactoring sectors dominate the ghg emissions.

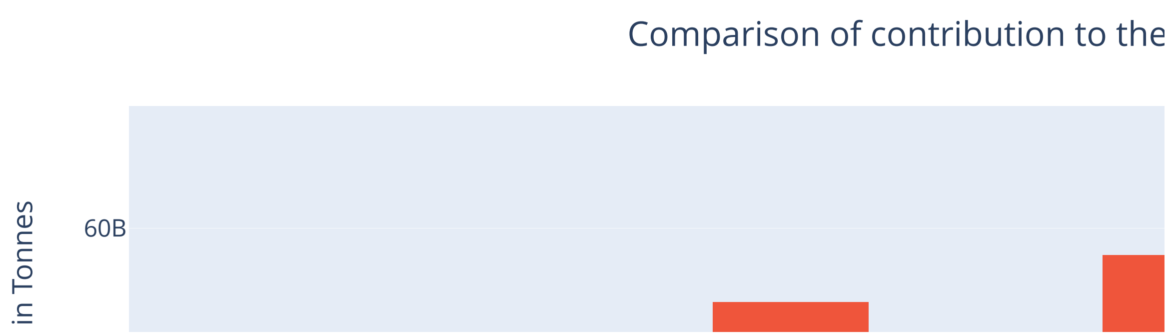
Q.4.2 Does transport contribute more or less than electricity

To show the contribution of transport and electricity we can plot a comparative histogram of them

In [100…

```
fig_comp = px.histogram(ghg_world, x="year", y=['transport','electricity'],
                         hover_data= ['value'],
                        labels = {'value': 'GHG emissions'},
                        barmode='group',
```

```
                    height=400)

fig_comp.update_layout(title="Comparison of contribution to the Global GHG emissions by Electricity and Transport Sector",
                   yaxis_title = "GHG emissions in Tonnes",
                    title_x=0.50)
fig_comp.update(layout_coloraxis_showscale = True)

fig_comp.show()

#fig_comp.write_html("D:\Downloads\GHG emissions HTML Plots\Electricity vs Transport.html")
```



Plot 8

We see that electricity always accounts for more co2 than the transport sector over the years.

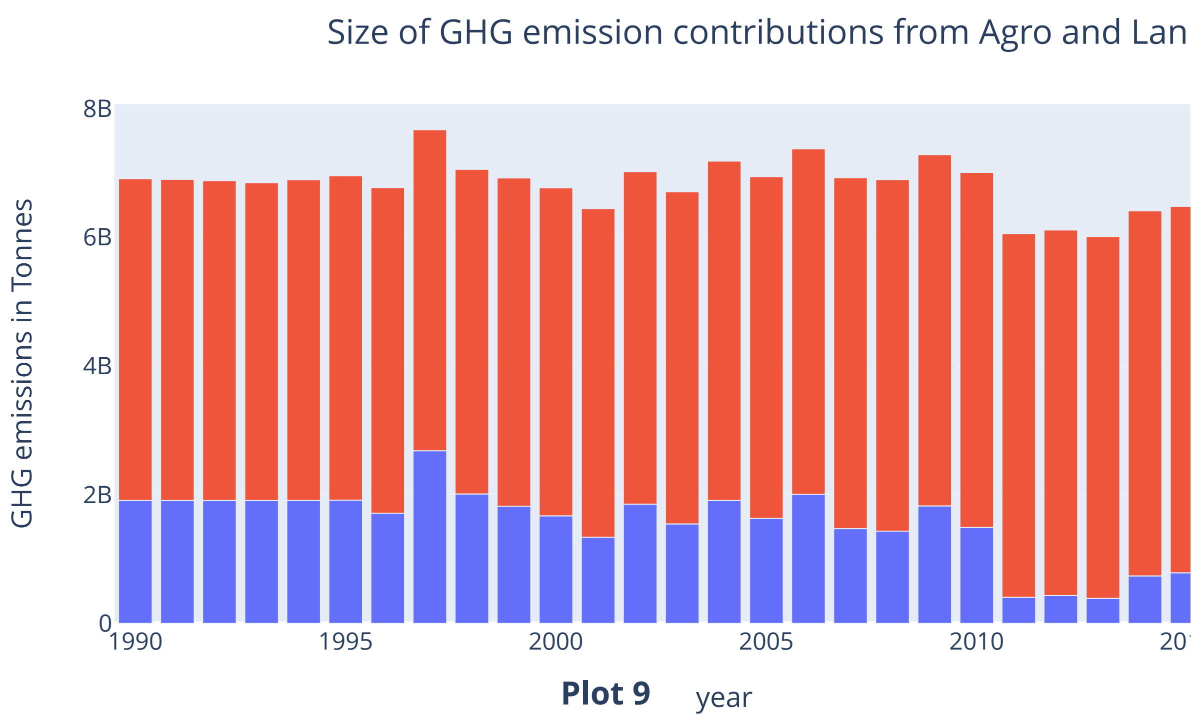
Q.4.3 How large are agriculture and land use emissions

For answering this question, we plot a stacked bar plot of land use and agriculture

In [101…

```
fig_stack = px.bar(ghg_world, x="year", y=['land-use-change-forestry','agriculture'],
```

```
                    hover_data= ['value'],
                   labels = {'value': 'GHG emissions'},
                   height=400, width=900)

fig_stack.update_layout(title="Size of GHG emission contributions from Agro and
Land use sector",
                  yaxis_title = "GHG emissions in Tonnes",
                   title_x=0.50)
fig_stack.update(layout_coloraxis_showscale = True)

fig_stack.show()

#fig_stack.write_html("D:\Downloads\GHG emissions HTML Plots\Agro and land
use.html")
```



Plot 9

The highest emission by the agriculture and land use sector was in 1997 which stands as a staggering 4.98 Billion tonnes and 2.6 Billion tonnes respectively.

Q.5.1 and 5.2

How much energy do we use per unit of GDP?

How much carbon do we emit per unit of energy?

In [58]:
```
gdp_df = pd.read_csv("D:\\Downloads\\global-co2-data.csv",parse_dates=['year'])
gdp_df
```

Out[58]:

	iso_code	country	year	co2	co2_growth_prct	co2_growth_abs	consumption_co2	tra
0	AFG	Afghanistan	1949-01-01	0.015	NaN	NaN	NaN	
1	AFG	Afghanistan	1950-01-01	0.084	475.000	0.070	NaN	
2	AFG	Afghanistan	1951-01-01	0.092	8.696	0.007	NaN	
3	AFG	Afghanistan	1952-01-01	0.092	NaN	NaN	NaN	
4	AFG	Afghanistan	1953-01-01	0.106	16.000	0.015	NaN	
...	...	...	...	...	...	...	...	
23703	ZWE	Zimbabwe	2015-01-01	12.170	1.653	0.198	13.308	
23704	ZWE	Zimbabwe	2016-01-01	10.815	-11.139	-1.356	12.171	
23705	ZWE	Zimbabwe	2017-01-01	10.247	-5.251	-0.568	11.774	
23706	ZWE	Zimbabwe	2018-01-01	11.341	10.674	1.094	12.815	
23707	ZWE	Zimbabwe	2019-01-01	10.374	-8.521	-0.966	NaN	

23708 rows × 55 columns

Focusing on India for this particular question

In [59]:
```
gdp_df= gdp_df[gdp_df['country'] == 'India']
gdp_df
```

Out[59]:

	iso_code	country	year	co2	co2_growth_prct	co2_growth_abs	consumption_co2	trad
10133	IND	India	1858-01-01	0.395	NaN	NaN	NaN	
10134	IND	India	1859-01-01	0.638	61.344	0.242	NaN	
10135	IND	India	1860-01-01	0.644	1.042	0.007	NaN	
10136	IND	India	1861-01-01	0.498	-22.680	-0.146	NaN	
10137	IND	India	1862-01-01	0.551	10.667	0.053	NaN	
...	...	...	...	...	...	...	...	
10279	IND	India	2015-01-01	2253.429	3.159	68.997	2067.349	-18
10280	IND	India	2016-01-01	2392.360	6.165	138.931	2180.245	-21
10281	IND	India	2017-01-01	2456.848	2.696	64.488	2252.484	-20
10282	IND	India	2018-01-01	2591.324	5.474	134.476	2354.795	-23
10283	IND	India	2019-01-01	2616.449	0.970	25.125	NaN	

151 rows × 55 columns

In [108…

```
gdp_df = gdp_df.loc[gdp_df['year'] > '1964-01-01' ]
```

Selecting the useful columns

In [61]:

```
gdp_df = gdp_df[['country' , 'year' , 'co2_per_unit_energy','energy_per_gdp']]


gdp_df.reset_index(level=0,inplace=True)
del gdp_df["index"]
gdp_df
```

Out[61]:

	country	year	co2_per_unit_energy	energy_per_gdp
0	India	1965-01-01	0.251	0.991
1	India	1966-01-01	0.253	1.036
2	India	1967-01-01	0.245	1.011
3	India	1968-01-01	0.251	1.089
4	India	1969-01-01	0.230	1.177
5	India	1970-01-01	0.241	1.128
6	India	1971-01-01	0.245	1.144
7	India	1972-01-01	0.246	1.175
8	India	1973-01-01	0.249	1.115
9	India	1974-01-01	0.242	1.190
10	India	1975-01-01	0.245	1.159
11	India	1976-01-01	0.244	1.244
12	India	1977-01-01	0.244	1.308
13	India	1978-01-01	0.239	1.374
14	India	1979-01-01	0.237	1.551
15	India	1980-01-01	0.243	1.546
16	India	1981-01-01	0.237	1.733
17	India	1982-01-01	0.247	1.748
18	India	1983-01-01	0.254	1.813
19	India	1984-01-01	0.244	1.938
20	India	1985-01-01	0.254	2.032
21	India	1986-01-01	0.254	2.063
22	India	1987-01-01	0.255	2.072
23	India	1988-01-01	0.253	2.067
24	India	1989-01-01	0.255	2.101
25	India	1990-01-01	0.254	2.109
26	India	1991-01-01	0.257	2.163
27	India	1992-01-01	0.260	2.141
28	India	1993-01-01	0.262	2.039
29	India	1994-01-01	0.263	2.035
30	India	1995-01-01	0.260	2.037
31	India	1996-01-01	0.271	1.945
32	India	1997-01-01	0.267	1.930

	country	year	co2_per_unit_energy	energy_per_gdp
33	India	1998-01-01	0.257	1.898
34	India	1999-01-01	0.271	1.827
35	India	2000-01-01	0.265	1.828
36	India	2001-01-01	0.267	1.736
37	India	2002-01-01	0.263	1.713
38	India	2003-01-01	0.262	1.622
39	India	2004-01-01	0.263	1.590
40	India	2005-01-01	0.258	1.507
41	India	2006-01-01	0.261	1.425
42	India	2007-01-01	0.258	1.401
43	India	2008-01-01	0.263	1.295
44	India	2009-01-01	0.270	1.312
45	India	2010-01-01	0.268	1.227
46	India	2011-01-01	0.266	1.170
47	India	2012-01-01	0.278	1.165
48	India	2013-01-01	0.280	1.177
49	India	2014-01-01	0.281	1.182
50	India	2015-01-01	0.281	1.135
51	India	2016-01-01	0.286	1.106
52	India	2017-01-01	NaN	NaN
53	India	2018-01-01	NaN	NaN
54	India	2019-01-01	NaN	NaN

Performing the multiple y axis plots for co2 per unit energy and energy per gdp

In [102…

```
from plotly.subplots import make_subplots

#fig_co2 = px.line(gdp_df, x = "year", y = "co2_per_unit_energy", height=600)

# Create figure with secondary y-axis
fig_co2 = make_subplots(specs=[[{"secondary_y": True}]])

# Add traces
fig_co2.add_trace(
```

```
    go.Scatter( x=gdp_df['year'], y=gdp_df['co2_per_unit_energy'], name="Co2 per unit energy"),
    secondary_y=False,
)

fig_co2.add_trace(
    go.Scatter(x=gdp_df['year'], y=gdp_df['energy_per_gdp'], name="Energy per unit gdp"),
    secondary_y=True,
)

fig_co2.update_layout(title="Energy per GDP and Co2 per unit energy for India",title_x=0.50)
fig_co2.update(layout_coloraxis_showscale = True)

fig_co2.show()

#fig_co2.write_html("D:\Downloads\GHG emissions HTML Plots\Energy per gdp and co2 per unit energy.html")
```

Plot 10

This plot shows the energy per gdp with reference to the right y axis and co2 per unit energy with reference to the left y axis with their corresponding scales.