DATA ANALYTICS MINI PROJECT

Reg No: 21112041

Class: 3BSc DS

Date: 11th December 2022

Webscraped the data from: https://oec.world/en

GETTING THE DATA

In [1]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
import warnings
```

In [2]:
```
warnings.filterwarnings("ignore")
```

In [3]:
```
countries = {
    'Japan': 'jpn',
    'United states' : 'usa',
    'China': 'chn',
    'Germany' : 'deu',
    'France' : 'fra',
    'Italy' : 'ita',
    'Canada' : 'can',
    'South Korea' : 'kor',
    'Russia' : 'rus',
    'Brazil' : 'bra',
    'Australia' : 'aus',
    'Spain' : 'esp',
    'Indonesia' : 'idn',
    'India' : 'ind'}
```

In [4]:
```
len(countries)
```

Out[4]: 14

In [5]:
```
def getdata(country_1,country_2):
    URL = f"https://oec.world/en/profile/bilateral-country/{countries[country_1]}/partner/{countries[country_2]}?measureE
    r = requests.get(URL)

    soup = BeautifulSoup(r.content, 'html5lib')

    table = soup.find('div',attrs = {'class':"cp-stat-group-wrapper cp-hero-stat-group-wrapper"})
    data = []
    for row in table.findAll('span',
                         attrs = {'class':'cp-stat-value-text heading length-sm'}):
                                data.append(row.text)

    ranks = []
    for row in table.findAll('span',
                         attrs = {'class':'cp-stat-subtitle heading length-md'}):
                                ranks.append(row.text)

    top_products = []
    ranks_2 = []

    for row in table.findAll('span',
                         attrs = {'class':'cp-stat-subtitle heading length-lg'}):
                                ranks_2.append(row.text)
    for row in table.findAll('span',
                         attrs = {'class':'cp-stat-subtitle heading length-xxl'}):
                                top_products.append(row.text)

    for row in table.findAll('span',
                         attrs = {'class':'cp-stat-subtitle heading length-sm'}):
                                top_products.append(row.text)

    for row in table.findAll('span',
                         attrs = {'class':'cp-stat-subtitle heading length-xl'}):
                                top_products.append(row.text)

    for i in ranks:
            if i[0:3]!='Rnk':
                top_products.append(i)
```

```
                ranks.remove(i)

    for i in ranks_2:
        if '$' in i:
            ranks.append(i.split("$")[1])
        else:
            top_products.append(i)

    dict_ = {}
    dict_1 = {}

    if len(top_products)!=0:

        dict_['country_1'] = country_1
        dict_['country_2'] = country_2
        dict_['exports'] = data[0]
        dict_['exports rank'] = ranks[0]
        dict_['top product exported'] = top_products[0]
        dict_['top product export'] = data[1]
        dict_['imports'] = data[2]
        dict_['imports rank'] = ranks[1]
        if len(top_products)>1:
            dict_['top product imported'] = top_products[1]
        else:
            dict_['top product imported'] = top_products[0]
        dict_['top product import'] = data[3]

        dict_1['country_name'] = country_1
        dict_1['ECI'] = data[4]
        dict_1['ECI rank'] = ranks[2]
        dict_1['GDP'] = data[6]
        dict_1['GDP rank'] = ranks[4]
        dict_1['GDP growth'] = data[7]
        dict_1['GDP growth rank'] = ranks[5]
        dict_1['GDP PC'] = data[10]
        dict_1['GDP PC rank'] = ranks[8]
        dict_1['GDP PC growth'] = data[11]
        dict_1['GDP PC growth rank'] = ranks[9]

        return dict_,dict_1
```

In [6]:

```
# countries_df = pd.DataFrame()
# trade_df = pd.DataFrame()
# for i in countries.keys():
```

```
#         for j in countries.keys():
#             if i!=j:
#                 a,b = getdata(i,j)
#                 print(i,j)
#                 countries_df = countries_df.append(b,ignore_index=True)
#                 trade_df = trade_df.append(a,ignore_index=True)
```

In [7]:
```
# countries_df = countries_df.drop_duplicates()
```

In [8]:
```
# countries_df.reset_index()
```

EXPORTING THE DATAFRAMES AS CSV

In [9]:
```
# countries_df.to_csv('countriesdf.csv', index=False)
```

In [10]:
```
# trade_df.to_csv('tradedf.csv', index=False)
```

IMPORTING THE CSVS USING PANDAS

In [11]:
```
countries_df = pd.read_csv("C:/Users/harshitha/Documents/SEM 3/Data analytics with Python/MINI PROJECT/countriesdf.csv")
```

In [12]:
```
trade_df = pd.read_csv("C:/Users/harshitha/Documents/SEM 3/Data analytics with Python/MINI PROJECT/tradedf.csv")
```

DESCRIBE THE DATASETS

In [13]:
```
countries_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   country_name        14 non-null     object
 1   ECI                 14 non-null     object
 2   ECI rank            14 non-null     object
 3   GDP                 14 non-null     object
```

```
 4   GDP rank            14 non-null     object
 5   GDP growth          14 non-null     object
 6   GDP growth rank     14 non-null     object
 7   GDP PC              14 non-null     object
 8   GDP PC rank         14 non-null     object
 9   GDP PC growth       14 non-null     object
 10  GDP PC growth rank  14 non-null     object
dtypes: object(11)
memory usage: 1.3+ KB
```

In [14]:
```
trade_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 182 entries, 0 to 181
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   country_1             182 non-null    object
 1   country_2             182 non-null    object
 2   exports               182 non-null    object
 3   exports rank          182 non-null    object
 4   top product exported  182 non-null    object
 5   imports               182 non-null    object
 6   imports rank          182 non-null    object
 7   top product imported  182 non-null    object
 8   top product export    182 non-null    object
 9   top product import    182 non-null    object
dtypes: object(10)
memory usage: 14.3+ KB
```

In [15]:
```
countries_df.describe()
```

Out[15]:

	country_name	ECI	ECI rank	GDP	GDP rank	GDP growth	GDP growth rank	GDP PC	GDP PC rank	GDP PC growth	GDP PC growth rank
count	14	14	14	14	14	14	14	14	14	14	14
unique	14	13	14	14	14	14	14	14	14	14	14
top	Japan	ECI 1.88	Rnk 1 / 127	$5.06T	Rnk 3 / 195	-12.2%	Rnk 180 / 195	$40,193	Rnk 25 / 195	-10.6%	Rnk 161 / 195
freq	1	2	1	1	1	1	1	1	1	1	1

In [16]:
```
trade_df.describe()
```

Out[16]:

	country_1	country_2	exports	exports rank	top product exported	imports	imports rank	top product imported	top product export	top product import
count	182	182	182	182	182	182	182	182	182	182
unique	14	14	174	147	38	174	147	44	171	170
top	Japan	United states	$112B	Rnk 1 / 218	Cars	$4.39B	Rnk 1 / 218	Cars	$2.75B	$2.75B
freq	13	13	2	4	41	2	4	23	3	3

DATA CLEANING AND TRANFORMATION - COUNTRIES_DF

MISSING OR NULL VALUES

In [17]:
```
countries_df.isnull().sum()
```

Out[17]:
```
country_name          0
ECI                   0
ECI rank              0
GDP                   0
GDP rank              0
GDP growth            0
GDP growth rank       0
GDP PC                0
GDP PC rank           0
GDP PC growth         0
GDP PC growth rank    0
dtype: int64
```

DUPLICATES

In [18]:
```
countries_df.duplicated()
```

Out[18]:
```
0     False
1     False
2     False
3     False
4     False
5     False
6     False
7     False
```

```
8     False
9     False
10    False
11    False
12    False
13    False
dtype: bool
```

In [19]:
```
countries_df
```

Out[19]:

	country_name	ECI	ECI rank	GDP	GDP rank	GDP growth	GDP growth rank	GDP PC	GDP PC rank	GDP PC growth	GDP PC growth rank
0	Japan	ECI 2.19	Rnk 1 / 127	$5.06T	Rnk 3 / 195	-12.2%	Rnk 180 / 195	$40,193	Rnk 25 / 195	-10.6%	Rnk 161 / 195
1	United states	ECI 1.56	Rnk 9 / 127	$21T	Rnk 1 / 195	39.8%	Rnk 78 / 195	$63,593	Rnk 8 / 195	31.2%	Rnk 57 / 195
2	China	ECI 0.96	Rnk 28 / 127	$14.7T	Rnk 2 / 195	142%	Rnk 5 / 195	$10,435	Rnk 68 / 195	129%	Rnk 5 / 195
3	Germany	ECI 1.88	Rnk 4 / 127	$3.85T	Rnk 4 / 195	13.3%	Rnk 133 / 195	$46,208	Rnk 19 / 195	11.3%	Rnk 107 / 195
4	France	ECI 1.34	Rnk 15 / 127	$2.63T	Rnk 7 / 195	-0.47%	Rnk 160 / 195	$39,030	Rnk 27 / 195	-3.96%	Rnk 149 / 195
5	Italy	ECI 1.3	Rnk 19 / 127	$1.89T	Rnk 8 / 195	-11.5%	Rnk 179 / 195	$31,714	Rnk 31 / 195	-11.9%	Rnk 166 / 195
6	Canada	ECI 0.93	Rnk 29 / 127	$1.65T	Rnk 9 / 195	1.74%	Rnk 155 / 195	$43,295	Rnk 22 / 195	-8.97%	Rnk 157 / 195
7	South Korea	ECI 1.88	Rnk 5 / 127	$1.64T	Rnk 10 / 195	43.2%	Rnk 69 / 195	$31,631	Rnk 32 / 195	37%	Rnk 45 / 195
8	Russia	ECI 0.5	Rnk 43 / 127	$1.48T	Rnk 11 / 195	-2.72%	Rnk 164 / 195	$10,127	Rnk 70 / 195	-5.14%	Rnk 154 / 195
9	Brazil	ECI 0.44	Rnk 47 / 127	$1.44T	Rnk 12 / 195	-34.6%	Rnk 190 / 195	$6,797	Rnk 91 / 195	-39.8%	Rnk 187 / 195
10	Australia	ECI -0.31	Rnk 74 / 127	$1.33T	Rnk 13 / 195	15.7%	Rnk 128 / 195	$51,693	Rnk 14 / 195	-0.76%	Rnk 142 / 195
11	Spain	ECI 0.76	Rnk 34 / 127	$1.28T	Rnk 14 / 195	-9.8%	Rnk 174 / 195	$27,063	Rnk 36 / 195	-11.3%	Rnk 164 / 195

	country_name	ECI	ECI rank	GDP	GDP rank	GDP growth	GDP growth rank	GDP PC	GDP PC rank	GDP PC growth	GDP PC growth rank
12	Indonesia	ECI -0.093	Rnk 64 / 127	$1.06T	Rnk 16 / 195	40.2%	Rnk 76 / 195	$3,870	Rnk 121 / 195	23.9%	Rnk 78 / 195
13	India	ECI 0.56	Rnk 40 / 127	$2.66T	Rnk 6 / 195	58.8%	Rnk 46 / 195	$1,928	Rnk 150 / 195	42%	Rnk 38 / 195

CONVERTING ALL THE RANK COLUMNS INTO INT DTYPE

In [20]:
```
# ECI RANK - 127
# GDP/GDP GROWTH/GDP PC/GDP PC GROWTH RANK - 195

rank_cols = ['ECI rank','GDP rank','GDP growth rank','GDP PC rank','GDP PC growth rank']

for i in range(len(countries_df)):
    for j in rank_cols:
        rank = countries_df.iloc[i].loc[j]
        countries_df.iloc[i].loc[j] = rank[4:5]

for i in range(len(countries_df)):
        string = countries_df.iloc[i].loc['ECI']
        countries_df.iloc[i].loc['ECI'] = string[5:]

countries_df[rank_cols] = countries_df[rank_cols].apply(pd.to_numeric)
```

In [21]:
```
countries_df['GDP growth'] = countries_df['GDP growth'].str.rstrip("%").astype(float)/100
countries_df['GDP PC growth'] = countries_df['GDP PC growth'].str.rstrip("%").astype(float)/100
```

In [22]:
```
countries_df['GDP'] = countries_df['GDP'].str.lstrip('$').str.rstrip('T')
countries_df['GDP PC'] = countries_df['GDP PC'].str.lstrip('$').str.replace(",", "")
```

In [23]:
```
countries_df[['GDP','GDP PC']] = countries_df[['GDP','GDP PC']].apply(pd.to_numeric)
```

In [24]:
```
countries_df['ECI'] = countries_df['ECI'].astype(float)
```

In [25]:
```
countries_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   country_name        14 non-null     object 
 1   ECI                 14 non-null     float64
 2   ECI rank            14 non-null     int64  
 3   GDP                 14 non-null     float64
 4   GDP rank            14 non-null     int64  
 5   GDP growth          14 non-null     float64
 6   GDP growth rank     14 non-null     int64  
 7   GDP PC              14 non-null     int64  
 8   GDP PC rank         14 non-null     int64  
 9   GDP PC growth       14 non-null     float64
 10  GDP PC growth rank  14 non-null     int64  
dtypes: float64(4), int64(6), object(1)
memory usage: 1.3+ KB
```

ADDING A NEW COLUMN CALLED CODE

In [26]:
```
codes = list(countries.values())
```

In [27]:
```
for i in range(len(codes)):
    codes[i]= codes[i].upper()
```

In [28]:
```
countries_df['Codes']= codes
```

RENAMING COLUMNS

In [29]:
```
countries_df = countries_df.rename(columns = {'ECI rank':'ECI rank/127','GDP rank':'GDP rank/195','GDP growth rank':'GDP
```

In [30]:
```
countries_df.head()
```

Out[30]:

	country_name	ECI	ECI rank/127	GDP(T$)	GDP rank/195	GDP growth	GDP growth rank/195	GDP PC($)	GDP PC rank/195	GDP PC growth	GDP PC growth rank/195	Codes
0	Japan	2.19	1	5.06	3	-0.1220	1	40193	2	-0.1060	1	JPN
1	United states	1.56	9	21.00	1	0.3980	7	63593	8	0.3120	5	USA
2	China	0.96	2	14.70	2	1.4200	5	10435	6	1.2900	5	CHN
3	Germany	1.88	4	3.85	4	0.1330	1	46208	1	0.1130	1	DEU
4	France	1.34	1	2.63	7	-0.0047	1	39030	2	-0.0396	1	FRA

DATA CLEANING AND TRANFORMATION - TRADE_DF

MISSING OR NULL VALUES

In [31]:
```
trade_df.isnull().sum()
```

Out[31]:
```
country_1              0
country_2              0
exports                0
exports rank           0
top product exported   0
imports                0
imports rank           0
top product imported   0
top product export     0
top product import     0
dtype: int64
```

DUPLICATES

In [32]:
```
trade_df.duplicated().sum()
```

Out[32]:
```
0
```

In [33]:
```
trade_df.head()
```

Out[33]:

	country_1	country_2	exports	exports rank	top product exported	imports	imports rank	top product imported	top product export	top product import
0	Japan	United states	$112B	Rnk 2 / 213	Cars	$63.1B	Rnk 4 / 218	Petroleum Gas	$32.6B	$5.12B
1	Japan	China	$133B	Rnk 1 / 213	Cars	$151B	Rnk 3 / 209	Computers	$8.69B	$12.1B
2	Japan	Germany	$18.1B	Rnk 7 / 213	Cars	$19.7B	Rnk 19 / 220	Cars	$1.83B	$3.64B
3	Japan	France	$5.52B	Rnk 22 / 213	Cars	$6.66B	Rnk 13 / 217	Planes, Helicopters, and/or Spacecraft	$686M	$664M
4	Japan	Italy	$4.01B	Rnk 26 / 213	Cars	$8.35B	Rnk 13 / 213	Processed Tobacco	$735M	$1.43B

CHANGING SOME OF THE COLUMNS DTYPE INTO FLOAT

In [34]:

```
for k in ['exports','top product export','imports','top product import']:
        indexes = []
        for i in range(len(trade_df)):
               if trade_df[k][i][-1]=='B':
                           indexes.append(i)
        trade_df[k] = (trade_df[k].str.strip().str.lstrip('$'))
        trade_df[k] = trade_df[k].str.rstrip('B')
        trade_df[k] = trade_df[k].str.rstrip('M')
        trade_df[k] = trade_df[k].astype(float)
        for i in range(len(trade_df)):
            if i in indexes:
                trade_df.loc[i,k] = trade_df.loc[i,k]*1000000000
            else:
                trade_df.loc[i,k]=  trade_df.loc[i,k]*1000000
```

In [35]:

```
trade_df.head()
```

Out[35]:

	country_1	country_2	exports	exports rank	top product exported	imports	imports rank	top product imported	top product export	top product import
0	Japan	United states	1.120000e+11	Rnk 2 / 213	Cars	6.310000e+10	Rnk 4 / 218	Petroleum Gas	3.260000e+10	5.120000e+09

	country_1	country_2	exports	exports rank	top product exported	imports	imports rank	top product imported	top product export	top product import
1	Japan	China	1.330000e+11	Rnk 1 / 213	Cars	1.510000e+11	Rnk 3 / 209	Computers	8.690000e+09	1.210000e+10
2	Japan	Germany	1.810000e+10	Rnk 7 / 213	Cars	1.970000e+10	Rnk 19 / 220	Cars	1.830000e+09	3.640000e+09
3	Japan	France	5.520000e+09	Rnk 22 / 213	Cars	6.660000e+09	Rnk 13 / 217	Planes, Helicopters, and/or Spacecraft	6.860000e+08	6.640000e+08
4	Japan	Italy	4.010000e+09	Rnk 26 / 213	Cars	8.350000e+09	Rnk 13 / 213	Processed Tobacco	7.350000e+08	1.430000e+09

In [36]:
```
trade_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 182 entries, 0 to 181
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   country_1             182 non-null    object 
 1   country_2             182 non-null    object 
 2   exports               182 non-null    float64
 3   exports rank          182 non-null    object 
 4   top product exported  182 non-null    object 
 5   imports               182 non-null    float64
 6   imports rank          182 non-null    object 
 7   top product imported  182 non-null    object 
 8   top product export    182 non-null    float64
 9   top product import    182 non-null    float64
dtypes: float64(4), object(6)
memory usage: 14.3+ KB
```

RENAMING THE COLUMNS

In [37]:
```
trade_df = trade_df.rename(columns = {'exports':'exports($)','top product export':'top product export($)','top product im
```

CORRECTING TWO COLUMNS - TOP PRODUCT EXPORTED AND TOP PRODUCT IMPORTED

In [38]:
```
for i in countries.keys():
        for j in countries.keys():
            if i!=j:
```

```
                df_1=trade_df.loc[(trade_df['country_1']==i) & (trade_df['country_2']==j)]
                df_2=trade_df.loc[(trade_df['country_1']==j) & (trade_df['country_2']==i)]
                if (list(df_1['top product exported']))==(list(df_2['top product exported'])):
                            print(i,j)
                            trade_df.loc[(trade_df['country_1']==j) & (trade_df['country_2']==i), "top product exported"]
                            trade_df.loc[(trade_df['country_1']==j) & (trade_df['country_2']==i), "top product imported"]
```

```
Japan United states
Japan Germany
Japan France
Japan Italy
Japan Russia
Japan Brazil
Japan Australia
Japan Indonesia
United states Germany
United states Canada
United states South Korea
United states Russia
United states Australia
China Germany
China France
China Canada
China South Korea
China Russia
China Spain
China Indonesia
Germany Japan
Germany United states
Germany France
Germany Italy
Germany Canada
Germany South Korea
Germany Brazil
Germany Australia
Germany Spain
Germany India
France Germany
France Italy
France Canada
France South Korea
France Russia
France Brazil
France Spain
France Indonesia
France India
Italy France
Italy Canada
```

```
Italy South Korea
Italy Russia
Italy Brazil
Italy Spain
Italy Indonesia
Italy India
Canada South Korea
Canada Russia
Canada Brazil
Canada Australia
Canada Spain
South Korea China
South Korea Russia
Russia Brazil
Russia Australia
Russia Spain
Russia Indonesia
Brazil Australia
Brazil India
Australia Spain
Australia Indonesia
Spain Germany
Spain Indonesia
Spain India
Indonesia India
```

In [39]:
```
trade_df.head()
```

Out[39]:

	country_1	country_2	exports($)	exports rank	top product exported	imports($)	imports rank	top product imported	top product export($)	top product import($)
0	Japan	United states	1.120000e+11	Rnk 2 / 213	Cars	6.310000e+10	Rnk 4 / 218	Petroleum Gas	3.260000e+10	5.120000e+09
1	Japan	China	1.330000e+11	Rnk 1 / 213	Cars	1.510000e+11	Rnk 3 / 209	Computers	8.690000e+09	1.210000e+10
2	Japan	Germany	1.810000e+10	Rnk 7 / 213	Cars	1.970000e+10	Rnk 19 / 220	Cars	1.830000e+09	3.640000e+09
3	Japan	France	5.520000e+09	Rnk 22 / 213	Cars	6.660000e+09	Rnk 13 / 217	Planes, Helicopters, and/or Spacecraft	6.860000e+08	6.640000e+08
4	Japan	Italy	4.010000e+09	Rnk 26 / 213	Cars	8.350000e+09	Rnk 13 / 213	Processed Tobacco	7.350000e+08	1.430000e+09

EDA

In [40]:
```
countries_df.head()
```

Out[40]:

	country_name	ECI	ECI rank/127	GDP(T$)	GDP rank/195	GDP growth	GDP growth rank/195	GDP PC($)	GDP PC rank/195	GDP PC growth	GDP PC growth rank/195	Codes
0	Japan	2.19	1	5.06	3	-0.1220	1	40193	2	-0.1060	1	JPN
1	United states	1.56	9	21.00	1	0.3980	7	63593	8	0.3120	5	USA
2	China	0.96	2	14.70	2	1.4200	5	10435	6	1.2900	5	CHN
3	Germany	1.88	4	3.85	4	0.1330	1	46208	1	0.1130	1	DEU
4	France	1.34	1	2.63	7	-0.0047	1	39030	2	-0.0396	1	FRA

COUNTRIES WITH THEIR GDP

In [104…
```
import plotly.graph_objects as go

fig = go.Figure(data=go.Choropleth(
    locations = countries_df['Codes'],
    z = countries_df['GDP(T$)'],
    text = countries_df['country_name'],
    colorscale = 'Blues',
    autocolorscale=False,
    reversescale=True,
    marker_line_color='darkgray',
    marker_line_width=0.5,
    colorbar_tickprefix = '$',
    colorbar_title = 'GDP<br>Trillions US$',
))

fig.update_layout(
    title_text='Countries GDP',
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='equirectangular'
    )
)
```

```
fig.show()
```

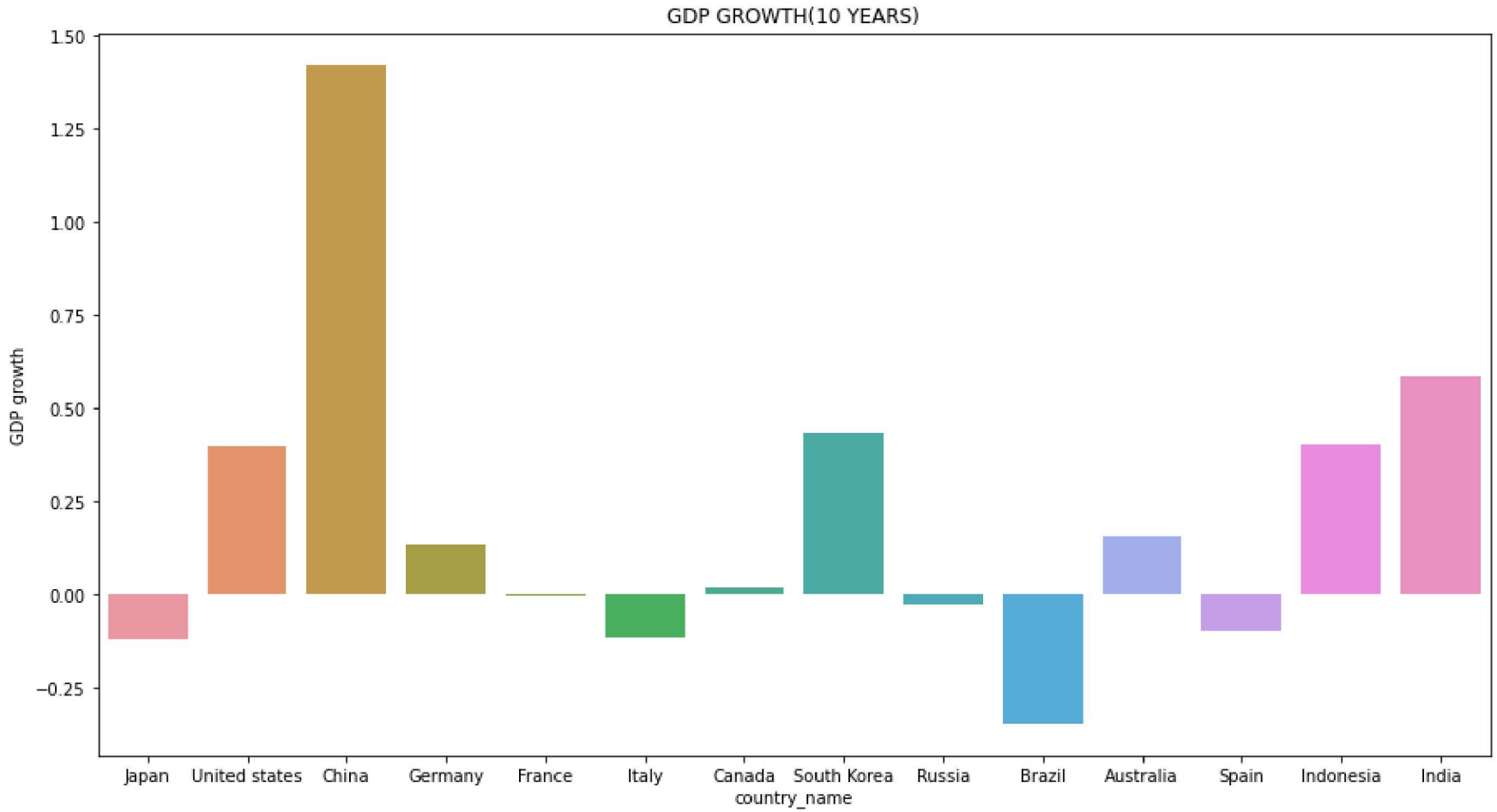
In [42]:

```
with open('plotly_graph.html', 'w') as f:
    f.write(fig.to_html(include_plotlyjs='cdn'))
```

COUNTRIES AND THEIR GDP GROWTH OVER THE LAST 10 YEARS

In [43]:
```
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(15,8))
sns.barplot(data=countries_df, x="country_name", y="GDP growth").set(title='GDP GROWTH(10 YEARS)')
```

Out[43]: [Text(0.5, 1.0, 'GDP GROWTH(10 YEARS)')]



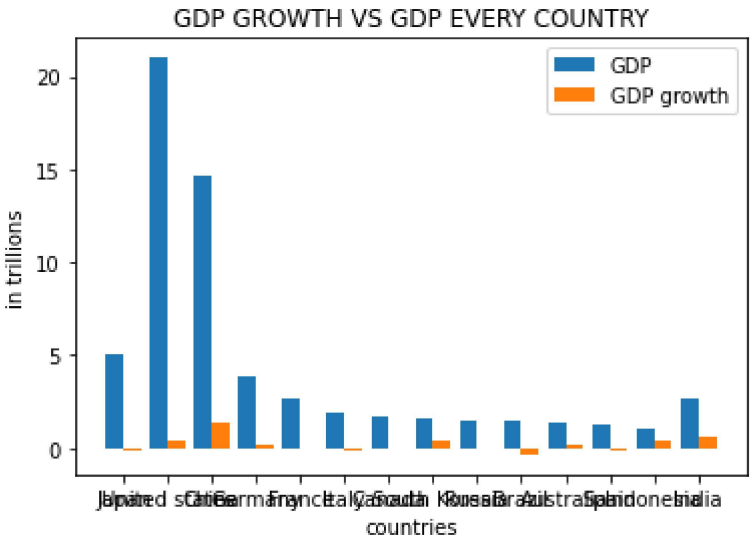
In [44]:
```
import numpy as np

X = list(countries_df['country_name'].unique())
GDP = list(countries_df['GDP(T$)'])
GDP_growth = list(countries_df['GDP growth'])
```

```
X_axis = np.arange(len(X))

plt.bar(X_axis - 0.2, GDP, 0.4, label = 'GDP')
plt.bar(X_axis + 0.2, GDP_growth, 0.4, label = 'GDP growth')

plt.xticks(X_axis, X)
plt.xlabel("countries")
plt.ylabel("in trillions")
plt.title("GDP GROWTH VS GDP EVERY COUNTRY")
plt.legend()
plt.show()
```



COUNTRY VS EXPORTS AND IMPORTS

In [45]:

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
country = input("Enter the country name :")
ie = input("exports or imports :")
if ie=="exports":
    col = 'top product exported'
    o_col = 'exports($)'
else:
    col = 'top product imported'
    o_col = 'imports($)'
df = trade_df[trade_df['country_1']==country]
```

```
values = list(df[col].value_counts())
names = list(df[col].unique())
fig = px.pie(df, values=values,names = names, title=country)
fig.show()
```

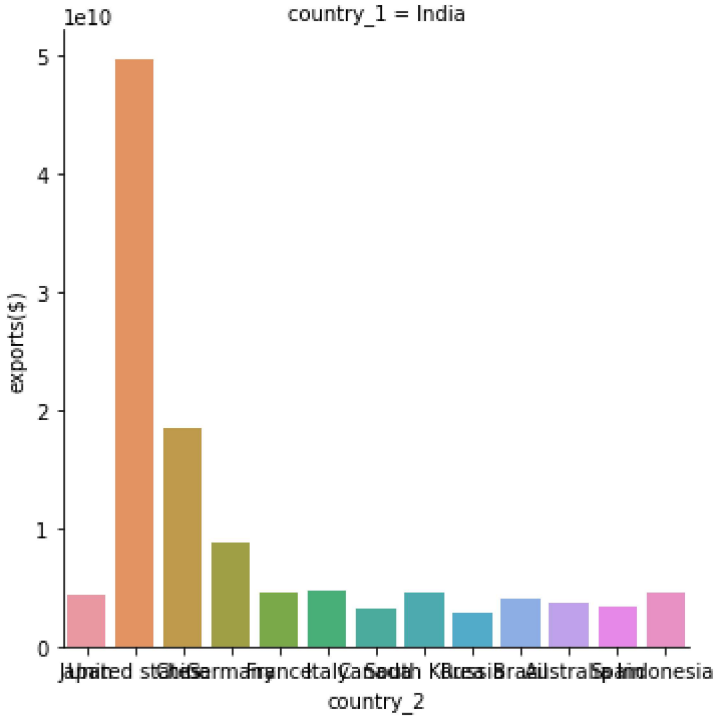
```
Enter the country name :India
exports or imports :exports
```

In [46]:

```
plt.figure(figsize=(500,200))
sns.catplot(
    data=df, x="country_2", y=o_col, col="country_1",
    kind="bar")
```

Out[46]: <seaborn.axisgrid.FacetGrid at 0x1d90f3f2070>

<Figure size 36000x14400 with 0 Axes>



In [47]:
```
import plotly.express as px
fig = px.bar(df, x="country_1", y=o_col, color="country_2")
fig.show()
```

ADDING COUNTRY'S GDP AND GDP GROWTH TO TRADE_DF

In [48]:
```
trade_df['1_gdp'] = 0
trade_df['2_gdp'] = 0
trade_df['1_gdpgrowth'] = 0
trade_df['2_gdpgrowth'] = 0
```

In [49]:
```
for i in range(len(countries_df)):
    for j in range(len(trade_df)):
        if trade_df['country_1'][j]==countries_df['country_name'][i]:
            trade_df['1_gdp'][j]=countries_df['GDP(T$)'][i]
            trade_df['1_gdpgrowth'][j]=countries_df['GDP growth'][i]
        elif trade_df['country_2'][j]==countries_df['country_name'][i]:
            trade_df['2_gdp'][j]=countries_df['GDP(T$)'][i]
            trade_df['2_gdpgrowth'][j]=countries_df['GDP growth'][i]
```

In [50]:
```
trade_df['exports($)']=trade_df['exports($)']/1000000
trade_df['imports($)']=trade_df['imports($)']/1000000
trade_df['top product export($)'] = trade_df['top product export($)']/1000000
trade_df['top product import($)'] = trade_df['top product import($)']/1000000
```

In [51]:
```
trade_df
```

Out[51]:

	country_1	country_2	exports($)	exports rank	top product exported	imports($)	imports rank	top product imported	top product export($)	top product import($)	1_gdp	2_gdp	1_gdpgrow
0	Japan	United states	112000.0	Rnk 2 / 213	Cars	63100.0	Rnk 4 / 218	Petroleum Gas	32600.0	5120.0	5.06	21.00	-0.1
1	Japan	China	133000.0	Rnk 1 / 213	Cars	151000.0	Rnk 3 / 209	Computers	8690.0	12100.0	5.06	14.70	-0.1
2	Japan	Germany	18100.0	Rnk 7 / 213	Cars	19700.0	Rnk 19 / 220	Cars	1830.0	3640.0	5.06	3.85	-0.1
3	Japan	France	5520.0	Rnk 22 / 213	Cars	6660.0	Rnk 13 / 217	Planes, Helicopters, and/or Spacecraft	686.0	664.0	5.06	2.63	-0.1
4	Japan	Italy	4010.0	Rnk 26 / 213	Cars	8350.0	Rnk 13 / 213	Processed Tobacco	735.0	1430.0	5.06	1.89	-0.1
...	...	...	...	...	...	...	...	...	...	...	...	...	
177	India	Russia	2870.0	Rnk 30 / 218	Packaged Medicaments	5930.0	Rnk 15 / 187	Coal Briquettes	444.0	923.0	2.66	1.48	0.5
178	India	Brazil	4130.0	Rnk 21 / 218	Crude Petroleum	3010.0	Rnk 14 / 223	Pesticides	621.0	1220.0	2.66	1.44	0.5
179	India	Australia	3740.0	Rnk 23 / 218	Refined Petroleum	11300.0	Rnk 5 / 211	Coal Briquettes	913.0	9120.0	2.66	1.33	0.5
180	India	Spain	3320.0	Rnk 28 / 218	Special Purpose Ships	1400.0	Rnk 37 / 218	Nitrogen Heterocyclic Compounds	148.0	112.0	2.66	1.28	0.5
181	India	Indonesia	4550.0	Rnk 18 / 218	Special Purpose Ships	11000.0	Rnk 5 / 213	Coal Briquettes	772.0	3800.0	2.66	1.06	0.5

182 rows × 14 columns

TRENDS IN THE FINAL TRADE DATASET

OUTLIERS IN EXPORTS

In [52]:
```
import plotly.express as px

fig = px.box(trade_df, x="country_1", y="exports($)",title = "Outliers in exports", hover_name="country_2", hover_data=['
fig.update_layout(yaxis_range=[100,450000])
fig.show()
```

OUTLIERS IN IMPORTS

In [53]:
```
fig_1 = px.box(trade_df, x="country_1", y="imports($)",title = "Outliers in imports", hover_name="country_2", hover_data=
fig_1.update_layout(yaxis_range=[157,500000])
fig_1.show()
```

In [54]:
```
fig_2 = px.treemap(trade_df, path=[px.Constant("country"),'country_1','country_2'], values='exports($)',title = "Country
fig_2.update_layout(yaxis_range=[0,6710])
fig_2.show()
```

In [55]:

```
fig_2 = px.treemap(trade_df, path=[px.Constant("country"),'country_1','country_2'], values='imports($)',title = "Country
fig_2.update_layout(yaxis_range=[0,6710])
fig_2.show()
```

In [56]:

```
fig_3 = px.icicle(trade_df, path=['country_1','country_2'], values='imports($)',title = "Country vs imports from other co
fig_3.update_layout(yaxis_range=[0,6710])
fig_3.update_layout(
    autosize=False,
    width=800,
    height=1000,)
fig_3.show()
```

In [57]:
```
trade_df.head()
```

Out[57]:

	country_1	country_2	exports($)	exports rank	top product exported	imports($)	imports rank	top product imported	top product export($)	top product import($)	1_gdp	2_gdp	1_gdpgrowth	2_
0	Japan	United states	112000.0	Rnk 2 / 213	Cars	63100.0	Rnk 4 / 218	Petroleum Gas	32600.0	5120.0	5.06	21.00	-0.122	
1	Japan	China	133000.0	Rnk 1 / 213	Cars	151000.0	Rnk 3 / 209	Computers	8690.0	12100.0	5.06	14.70	-0.122	
2	Japan	Germany	18100.0	Rnk 7 / 213	Cars	19700.0	Rnk 19 / 220	Cars	1830.0	3640.0	5.06	3.85	-0.122	
3	Japan	France	5520.0	Rnk 22 / 213	Cars	6660.0	Rnk 13 / 217	Planes, Helicopters, and/or Spacecraft	686.0	664.0	5.06	2.63	-0.122	
4	Japan	Italy	4010.0	Rnk 26 / 213	Cars	8350.0	Rnk 13 / 213	Processed Tobacco	735.0	1430.0	5.06	1.89	-0.122	

TOP PRODUCT EXPORTED VS COUNTRIES

In [58]:
```
h = pd.DataFrame(trade_df.groupby(['country_1','top product exported']).count())
```

In [59]:
```
h = h.sort_index()
```

In [60]:
```
h
```

Out[60]:

country_1	top product exported	country_2	exports($)	exports rank	imports($)	imports rank	top product imported	top product export($)	top product import($)	1_gdp	2_gdp	1_gdpgrowth	2_(
Australia	Cars	1	1	1	1	1	1	1	1	1	1	1	
	Coal Briquettes	3	3	3	3	3	3	3	3	3	3	3	
	Gold	1	1	1	1	1	1	1	1	1	1	1	
	Iron Ore	2	2	2	2	2	2	2	2	2	2	2	
	Large Construction Vehicles	1	1	1	1	1	1	1	1	1	1	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
United states	Gold	1	1	1	1	1	1	1	1	1	1	1	
	Packaged Medicaments	1	1	1	1	1	1	1	1	1	1	1	
	Petroleum Gas	2	2	2	2	2	2	2	2	2	2	2	
	Refined Petroleum	1	1	1	1	1	1	1	1	1	1	1	
	Soybeans	1	1	1	1	1	1	1	1	1	1	1	

119 rows × 12 columns

In [61]:

```
i = pd.DataFrame(data = list(h.index))
i['count'] = list(h['country_2'])
i
```

Out[61]:

	0	1	count
0	Australia	Cars	1

	0	1	count
1	Australia	Coal Briquettes	3
2	Australia	Gold	1
3	Australia	Iron Ore	2
4	Australia	Large Construction Vehicles	1
...	...	...	...
114	United states	Gold	1
115	United states	Packaged Medicaments	1
116	United states	Petroleum Gas	2
117	United states	Refined Petroleum	1
118	United states	Soybeans	1

119 rows × 3 columns

In [62]:

```
fig = px.sunburst(i, path=[1,0], values='count',title = 'Top products exported vs Country')
fig.update_layout(
    autosize=False,
    width=600,
    height=500,)
fig.show()
```

In [63]:

```
fig_1 = px.sunburst(i, path=[0,1], values='count',title = 'Country vs Top products exported')
fig_1.update_layout(
    autosize=False,
    width=600,
    height=500,)
fig_1.show()
```

TOP PRODUCT IMPORTED VS COUNTRIES

In [64]:
```
imp = pd.DataFrame(trade_df.groupby(['country_1','top product imported']).count())
```

In [65]:
```
imp = imp.sort_index()
```

In [66]:
```
imp_count = pd.DataFrame(data = list(imp.index))
imp_count['count'] = list(imp['country_2'])
imp_count
```

Out[66]:

	0	1	count
0	Australia	Aluminium Oxide	1
1	Australia	Cars	2
2	Australia	Coal Briquettes	2
3	Australia	Computers	1
4	Australia	Gold	1
...	...	...	...
121	United states	Crustaceans	1
122	United states	Packaged Medicaments	3
123	United states	Planes, Helicopters, and/or Spacecraft	1
124	United states	Refined Petroleum	2

	0	1	count
125	United states	Semi-Finished Iron	1

126 rows × 3 columns

In [67]:

```
fig = px.bar(imp_count, x=1, y='count', barmode="group",facet_col=0,title = 'Each Country vs Top product imported')
fig.update_layout(
    autosize=False,
    width=6500,
    height=500,)
fig.show()
```

In [68]:

```
fig = px.sunburst(imp_count, path=[1,0], values='count',title = 'Top products imported vs Country')
fig.update_layout(
    autosize=False,
    width=600,
    height=600,)
fig.show()
```

In [69]:
```
fig = px.scatter(trade_df, x="country_1", y="exports($)",size="top product export($)",
                 hover_name="country_2",title = 'Exports vs exports of top product')
fig.show()
```

In [70]:
```
fig = px.scatter(trade_df, x="imports($)", y="exports($)", color="country_1",size = '1_gdp',title='Exports and Imports vs
fig.show()
```

In [71]:

```
fig = px.line(trade_df, x='exports($)', y='1_gdp', color='country_1',title="Exports vs GDP")
fig.show()
```

NO.OF CARS EXPORTED OR IMPORTED VS GDP

In [72]:
```
cars_exports  = i[i[1]=='Cars']
```

In [73]:
```
cars_exports = cars_exports.reset_index()
```

In [74]:
```
cars_exports
```

Out[74]:

	index	0	1	count
0	0	Australia	Cars	1
1	22	Canada	Cars	1
2	33	China	Cars	1
3	42	France	Cars	2

	index	0	1	count
4	50	Germany	Cars	5
5	74	Italy	Cars	2
6	82	Japan	Cars	9
7	95	South Korea	Cars	4
8	102	Spain	Cars	3
9	111	United states	Cars	4

In [75]:
```
countries_df['cars_exported'] = 0
country_names = list(countries_df['country_name'])
for i in range(len(country_names)):
    for j in range(len(cars_exports)):
        if cars_exports[0][j]== countries_df['country_name'][i]:
            countries_df['cars_exported'][i]=cars_exports['count'][j]
```

In [76]:
```
countries_df
```

Out[76]:

	country_name	ECI	ECI rank/127	GDP(T$)	GDP rank/195	GDP growth	GDP growth rank/195	GDP PC($)	GDP PC rank/195	GDP PC growth	GDP PC growth rank/195	Codes	cars_exported
0	Japan	2.190	1	5.06	3	-0.1220	1	40193	2	-0.1060	1	JPN	9
1	United states	1.560	9	21.00	1	0.3980	7	63593	8	0.3120	5	USA	4
2	China	0.960	2	14.70	2	1.4200	5	10435	6	1.2900	5	CHN	1
3	Germany	1.880	4	3.85	4	0.1330	1	46208	1	0.1130	1	DEU	5
4	France	1.340	1	2.63	7	-0.0047	1	39030	2	-0.0396	1	FRA	2
5	Italy	1.300	1	1.89	8	-0.1150	1	31714	3	-0.1190	1	ITA	2
6	Canada	0.930	2	1.65	9	0.0174	1	43295	2	-0.0897	1	CAN	1
7	South Korea	1.880	5	1.64	1	0.4320	6	31631	3	0.3700	4	KOR	4
8	Russia	0.500	4	1.48	1	-0.0272	1	10127	7	-0.0514	1	RUS	0
9	Brazil	0.440	4	1.44	1	-0.3460	1	6797	9	-0.3980	1	BRA	0

	country_name	ECI	ECI rank/127	GDP(T$)	GDP rank/195	GDP growth	GDP growth rank/195	GDP PC($)	GDP PC rank/195	GDP PC growth	GDP PC growth rank/195	Codes	cars_exported
10	Australia	-0.310	7	1.33	1	0.1570	1	51693	1	-0.0076	1	AUS	1
11	Spain	0.760	3	1.28	1	-0.0980	1	27063	3	-0.1130	1	ESP	3
12	Indonesia	-0.093	6	1.06	1	0.4020	7	3870	1	0.2390	7	IDN	0
13	India	0.560	4	2.66	6	0.5880	4	1928	1	0.4200	3	IND	0

In [77]:

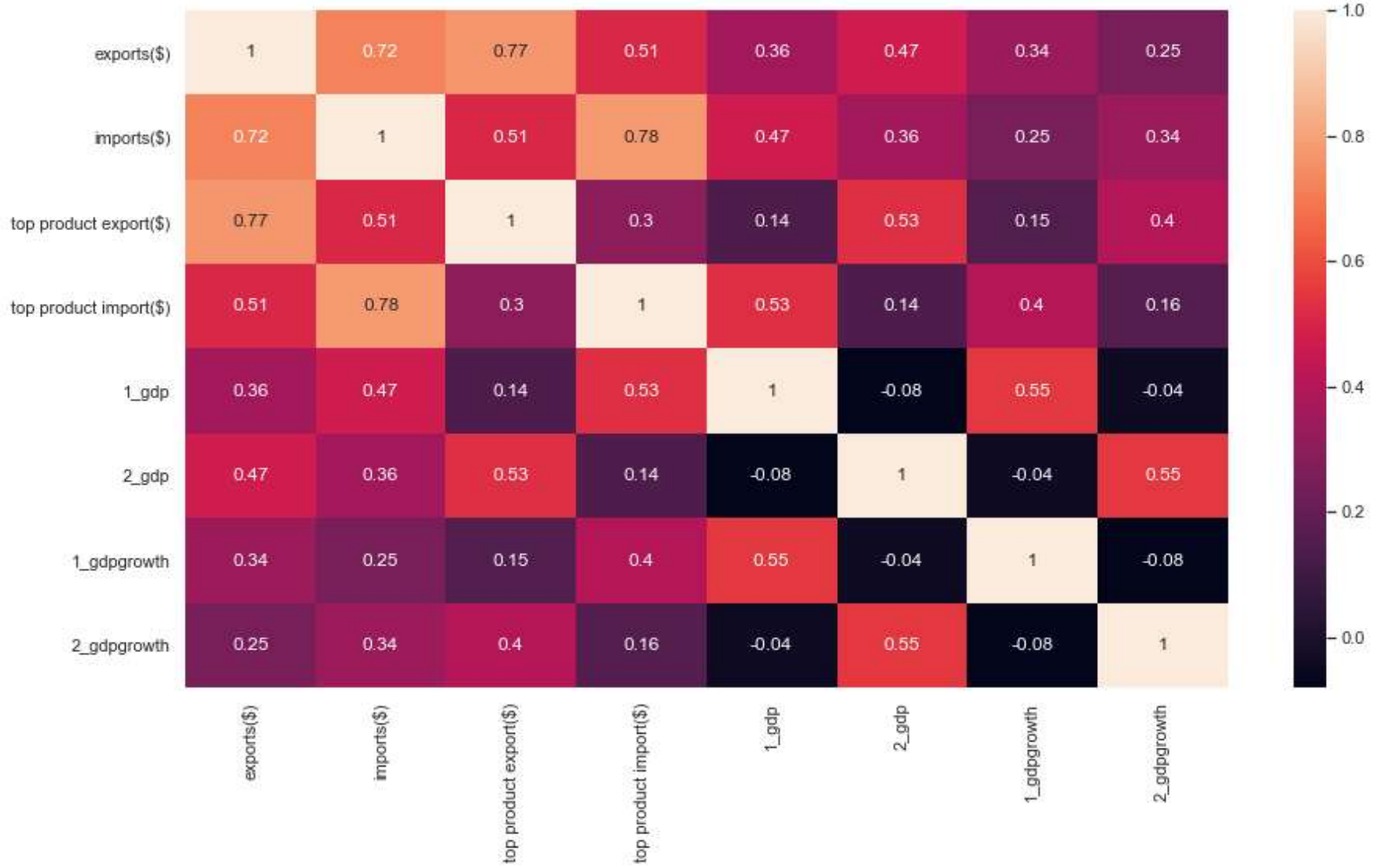
```
fig = px.scatter(countries_df, x="country_name", y="GDP(T$)",size="cars_exported",
                 title = 'Country GDP vs No.of times cars was exported as the top product',color = "cars_exported")
fig.show()
```

CORRELATION HEATMAP

In [78]:
```
sns.set(rc = {'figure.figsize':(15,8)})
```

In [79]:
```
correlation_matrix = trade_df.corr().round(2)
# annot = True to print the values inside the square
sns.heatmap(data=correlation_matrix, annot=True)
```

Out[79]: <AxesSubplot: >



ML MODELING

DATA TRANSFORMATION

LABEL ENCODING

In [80]:
```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
trade_df['top product exported'] = le.fit_transform(trade_df['top product exported'])
trade_df['top product imported'] = le.fit_transform(trade_df['top product imported'])
```

In [81]:
```
trade_df.drop(['exports rank','imports rank'], axis=1)
```

Out[81]:

	country_1	country_2	exports($)	top product exported	imports($)	top product imported	top product export($)	top product import($)	1_gdp	2_gdp	1_gdpgrowth	2_gdpgrowth
0	Japan	United states	112000.0	2	63100.0	28	32600.0	5120.0	5.06	21.00	-0.122	0.3980
1	Japan	China	133000.0	2	151000.0	6	8690.0	12100.0	5.06	14.70	-0.122	1.4200
2	Japan	Germany	18100.0	2	19700.0	2	1830.0	3640.0	5.06	3.85	-0.122	0.1330
3	Japan	France	5520.0	2	6660.0	31	686.0	664.0	5.06	2.63	-0.122	-0.0047
4	Japan	Italy	4010.0	2	8350.0	36	735.0	1430.0	5.06	1.89	-0.122	-0.1150
...	...	...	...	...	...	...	...	...	...	...	...	...
177	India	Russia	2870.0	25	5930.0	3	444.0	923.0	2.66	1.48	0.588	-0.0272
178	India	Brazil	4130.0	7	3010.0	27	621.0	1220.0	2.66	1.44	0.588	-0.3460
179	India	Australia	3740.0	40	11300.0	3	913.0	9120.0	2.66	1.33	0.588	0.1570
180	India	Spain	3320.0	45	1400.0	20	148.0	112.0	2.66	1.28	0.588	-0.0980
181	India	Indonesia	4550.0	45	11000.0	3	772.0	3800.0	2.66	1.06	0.588	0.4020

182 rows × 12 columns

SCALING

In [82]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
trade_df[['exports($)']] = scaler.fit_transform(trade_df[['exports($)']])
trade_df[['imports($)']] = scaler.fit_transform(trade_df[['imports($)']])
```

```
trade_df[['top product export($)']] = scaler.fit_transform(trade_df[['top product export($)']])
```

In [83]:
```
trade_df.head()
```

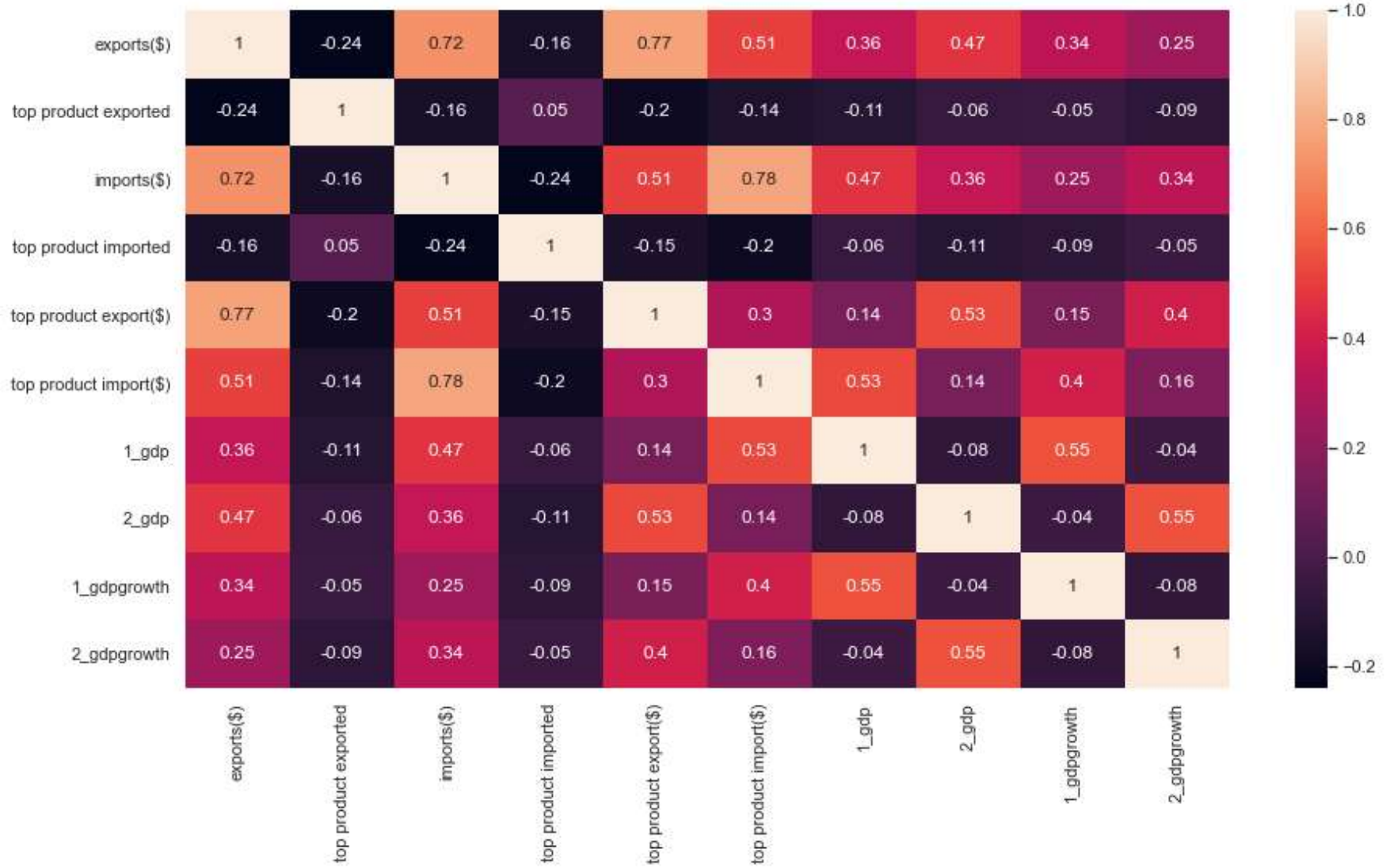
Out[83]:

	country_1	country_2	exports($)	exports rank	top product exported	imports($)	imports rank	top product imported	top product export($)	top product import($)	1_gdp	2_gdp	1_gdpgrowth	2_g
0	Japan	United states	1.771224	Rnk 2 / 213	2	0.768945	Rnk 4 / 218	28	3.473682	0.156114	5.06	21.00	-0.122	
1	Japan	China	2.201651	Rnk 1 / 213	2	2.570588	Rnk 3 / 209	6	0.578531	1.004693	5.06	14.70	-0.122	
2	Japan	Germany	-0.153398	Rnk 7 / 213	2	-0.120603	Rnk 19 / 220	2	-0.252114	-0.023814	5.06	3.85	-0.122	
3	Japan	France	-0.411244	Rnk 22 / 213	2	-0.387878	Rnk 13 / 217	31	-0.390636	-0.385616	5.06	2.63	-0.122	
4	Japan	Italy	-0.442193	Rnk 26 / 213	2	-0.353238	Rnk 13 / 213	36	-0.384702	-0.292491	5.06	1.89	-0.122	

FEATURE SELECTION USING CORRELATION COEFFICENT

In [84]:
```
correlation_matrix = trade_df.corr().round(2)
# annot = True to print the values inside the square
sns.heatmap(data=correlation_matrix, annot=True)
```

Out[84]:
```
<AxesSubplot: >
```



In [85]:
```
cor = trade_df.corr()
cor_target = abs(cor["exports($)"])
relevant_features = cor_target[cor_target > 0.1]
relevant_features.index
```

Out[85]:
```
Index(['exports($)', 'top product exported', 'imports($)',
       'top product imported', 'top product export($)',
       'top product import($)', '1_gdp', '2_gdp', '1_gdpgrowth',
```

```
       '2_gdpgrowth'],
      dtype='object')
```

SPLITTING THE DATA INTO TRAIN AND TEST

In [86]:
```
from sklearn.model_selection import train_test_split
```

In [87]:
```
x = ['imports($)','top product export($)',
       'top product import($)', '1_gdp', '2_gdp', '1_gdpgrowth',
       '2_gdpgrowth']
X = trade_df[x]
y = trade_df['exports($)']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.4)
X_train
```

Out[87]:

	imports($)	top product export($)	top product import($)	1_gdp	2_gdp	1_gdpgrowth	2_gdpgrowth
10	-0.464944	-0.362060	-0.404703	5.06	1.28	-0.1220	-0.0980
79	3.943854	5.023574	0.822333	1.65	21.00	0.0174	0.3980
89	-0.501223	-0.410857	-0.450536	1.65	1.06	0.0174	0.4020
13	1.771224	0.146257	3.496939	21.00	5.06	0.3980	-0.1220
174	-0.452442	-0.444640	-0.453453	2.66	1.89	0.5880	-0.1150
...	...	...	...	...	...	...	...
51	-0.344015	-0.177041	-0.418684	3.85	2.66	0.1330	0.5880
73	-0.462895	-0.436527	-0.407864	1.89	1.44	-0.1150	-0.3460
138	-0.521166	-0.422844	-0.462985	1.33	1.48	0.1570	-0.0272
42	0.811988	0.526465	0.107484	3.85	2.63	0.1330	-0.0047
22	-0.286625	-0.289651	-0.132014	21.00	1.33	0.3980	0.1570

109 rows × 7 columns

TRAINING THE LINEAR REGRESSION MODEL

In [88]:
```
from sklearn.linear_model import LinearRegression
```

In [89]:
```
linear_regressor = LinearRegression()
# Fitting the Data
linear_regressor.fit(X_train, y_train)

# Predicting the Target Variable
y_pred = linear_regressor.predict(X_test)
```

In [90]:
```
# Chcking the accuracy of the model
print("The accuracy of the model is", linear_regressor.score(X_test, y_test))
```

```
The accuracy of the model is 0.7326824029595089
```

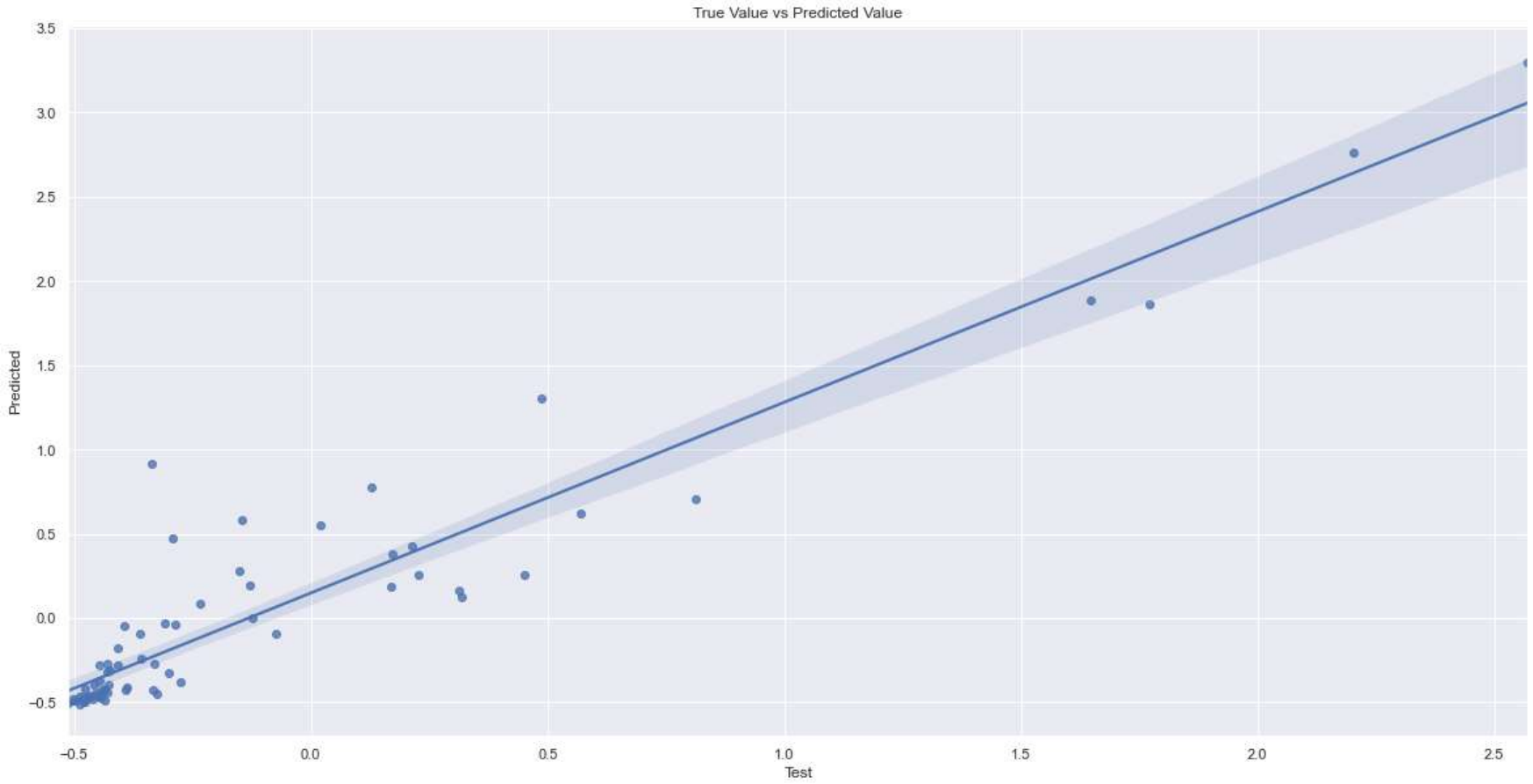
TRAINING THE RANDOM FOREST MODEL

In [91]:
```
from sklearn.ensemble import RandomForestRegressor
```

In [92]:
```
# Fitting the Data
forest = RandomForestRegressor(n_estimators=100)
forest.fit(X_train, y_train)

# Predicting the Target Variable
y_pred4 = forest.predict(X_test)
```

In [93]:
```
# Plotting the true and predicted value
fig = plt.figure(figsize =(20, 10))
sns.regplot(y_test, y_pred4)
plt.xlabel("Test")
plt.ylabel("Predicted")
plt.title("True Value vs Predicted Value")
plt.show()
```



In [94]:
```
# Chcking the accuracy of the model
print("The accuracy of the model is", forest.score(X_test, y_test))
```

```
The accuracy of the model is 0.7752407017507252
```

In [95]:
```
import math
from sklearn.metrics import mean_squared_error
```

In [96]:
```
print("The Mean Squared Error of the model is", math.sqrt(mean_squared_error(y_test, y_pred4)))
```

```
The Mean Squared Error of the model is 0.2892617621396154
```

TRAINING THE DECISION TREE MODEL

In [97]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [98]:
```
# Fitting the Data
tree = DecisionTreeRegressor(max_depth=4)
tree.fit(X_train, y_train)

# Predicting the Target Variable
y_pred3 = tree.predict(X_test)
```

In [99]:
```
print("The accuracy of the model is", tree.score(X_test, y_test))
```

```
The accuracy of the model is 0.7284062744541904
```

EXPORING THE RANDOM FOREST REGRESSOR AS PICKLE FILE

In [100…
```
import pickle
```

In [101…
```
#pickle.dump(forest, open('picklefile.pkl', 'wb'))
```

In [102…
```
# pickle.dump(scaler, open('ss.pkl', 'wb'))
```

In [105…
```
trade_df.to_csv('finaldataset.csv', index=False)
```