 # Final Project: Analyzing International Trade Data

You are provided with a dataset that contains export/import data over several years, broken down by country. Your task is to clean, analyze, and visualize this data to provide insights into global trade patterns.

Tasks:*

- **Data Cleaning and Preparation:**

- *Extract and Rename Columns:*

Identify the relevant columns that represent the actual data.

Rename the columns appropriately (e.g., Country, Year_2008_2009, Year_2009_2010, etc.).

After reading the file 'exportall 2009-2025)' and extract the information, we see there is 36 unnamed columns in our dataset, so we should rename these columns appropriately.

0	Export Import Data Bank	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Export :: Country-wise	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	S.No.	Country	2008-2009	%Share	2009-2010	%Share	2010-2011	%Share
3	1	AFGHANISTAN	1,82,344.16	0.2169	2,20,362.72	0.2606	1,92,084.44	0.1689
4	2	ALBANIA	5,589.70	0.0066	4,027.54	0.0048	5,227.48	0.0046

As seen in output of **df.info()** ,the names of row 2 are the real names which we can use for the names of columns. In addition, there is 2 fully empty rows which we should remove them.

df.columns=df.iloc[2]

df=df.drop(index=[0,1,2])

df.head()

S.No.	Country	2008-2009	%Share	2009-2010	%Share	2010-2011	%Share	2011-1
1	AFGHANISTAN	1,82,344.16	0.2169	2,20,362.72	0.2606	1,92,084.44	0.1689	2,42,91
2	ALBANIA	5,589.70	0.0066	4,027.54	0.0048	5,227.48	0.0046	6,06
3	ALGERIA	2,99,636.27	0.3564	2,73,710.63	0.3237	3,55,787.00	0.3129	3,99,22
4	AMERI SAMOA	58.26	0.0001	40.45	0	92.93	0.0001	97
5	ANDORRA	412.37	0.0005	96.49	0.0001	120.35	0.0001	12

ws × 36 columns

- *Handle Missing Values:*

Identify and appropriately handle any missing values in the dataset.

df.isna().sum()
df=df.dropna() #cleaning missing values from rows

missing_value=df.isna().sum() #check if there are missing values

- *Convert Data Types:*

Convert columns to appropriate data types, such as converting numeric columns from strings to floats or integers.

numeric_columns=df.select_dtypes(include=['int','float']).columns

df[numeric_columns]=df[numeric_columns].astype(float)

This line is used to select columns that contain numeric data types, specifically integers (int) and floating-point numbers (float). It ensures that only the

numeric columns are selected for further operations (e.g., calculating statistics or handling missing values).

We need also to clean and convert export values in a dataset, specifically handling cases where the values might be strings that include commas and need to be converted to numeric values (floats).

```
def clean_export_values(value):
  if isinstance(value, str):  # Check if the value is a string
    # Replace commas with nothing and convert to float
    try:
      value = value.replace(',', '')  # Remove thousands separators
      return float(value)  # Convert to float
    except ValueError:
      return None  # Handle cases where conversion might fail
  return value  # Return the value if it's already clean
```

value.replace(',', ''): Remove any commas (thousands separators) from the string. For example, "1,234" would become "1234".

And finally, for each existing year column, the apply() method applies the clean_export_values function to each cell (value) in that column. It cleans up the values by removing commas and converting them to floats.

```
year_columns=[f'{year}-{year+1}' for year in range(2008,2025)]
for col in year_columns:
 if col in df.columns:
  df[col]=df[col].apply(clean_export_values)
# Save the cleaned data back to a new CSV file
cleaned_file_path = '/content/exportall 2009 - 2025_cleaned.csv'  # Specify the new or original file path
```

```
df.to_csv(cleaned_file_path, index=False)  # Save without row index

print(f"Cleaned data saved to {cleaned_file_path}")

#df['2008-2009']=df['2008-2009'].apply(clean_export_values)
df=pd.read_csv('/content/exportall 2009 - 2025_cleaned.csv')
```

we create a list with format of our columns to hold years.

```
year_columns=[f'{year}-{year+1}' for year in range(2008,2025)]
```

```
['2008-2009', '2009-2010', '2010-2011', '2011-2012', '2012-2013', '2013-2014', '2014-2015'
```

Then we create an empty dictionary to hold summary statics.

- **Exploratory Data Analysis (EDA):**

- *Summary Statistics:*

Generate summary statistics for the dataset, such as mean, median, and total exports for each year.

```
# Dictionary to hold summary statistics for each year
summary_statics={}
for col in year_columns:
  if col in df.columns:
    summary_statics[col]={
```

```
      'mean':df[col].mean(),
      'median':df[col].median(),
      'total_export':df[col].sum()
   }
disply staticts for every year:
summary_statics={}
for col in year_columns:
  if col in df.columns:
    summary_statics[col]={
      'mean':df[col].mean(),
      'median':df[col].median(),
      'total_export':df[col].sum()
   }
#disply staticts for every year:
for year in summary_statics.items():
  print(f'summary for {year}:')
```

- *Top 10 Exporting Countries:*

Identify the top 10 exporting countries for each year based on total export values.

we create a subset of the DataFrame df that includes only the Country column and the current col year column. The result is a smaller DataFrame with two columns: one for the country names and one for the export values for that year.

The next step is createing a dictionary holder as top exporters:

```
top_exporters={}

for col in year_columns:

  if col in df.columns:

    top_countries=df[['Country',col]].sort_values(by=col).head(10)
```

now,displaying the name of 10 top exporter:

```
for year, countries in top_exporters.items():

    print(f"Top 10 exporting countries for {year}:")

    print(countries.to_string(index=False))

    print("\n")
```

?

- *Growth Rate Analysis:*

Calculate the year-over-year growth rate for each country and identify the countries with the highest growth.

```
growth_rates = df[['Country']].copy()  # Copy country names

for i in range(len(year_columns)):

    current_year=year_columns[i]

    perevious_year=year_columns[i-1]
```

```
   if current_year in df.columns and perevious_year in df.columns:
     growth_rate_column=f'{current_year}_growth_rate'
                                  growth_rates[growth_rate_column]=((df[current_year]-
df[perevious_year])/df[perevious_year])*100
  #display
top_growth_countries={}
for i in range(1,len(year_columns)):
  current_year=year_columns[i]
```

A new column name is created in growth_rates for the growth rate of that year:

```
  growth_rate_column=f'{current_year}_growth_rate'
  if growth_rate_column in growth_rates.columns:
    top_growth=growth_rates[['Country',growth_rate_column]].sort_values(by=g
rowth_rate_column).head(10)
    top_growth_countries[current_year]=top_growth
for year, countries in top_growth_countries.items():
  print(f'top countries with  highest grows for {year}:')
  print(countries.to_string(index=False))
  print("\n")
```

- **Data Visualization:**

- *Bar Charts:*

Create bar charts to visualize the top 10 exporting countries for a selected year.

```
# Identify top 10 countries based on growth rate for each year

top_growth_countries = {}

for i in range(1, len(year_columns)):

 current_year = year_columns[i]

 growth_rate_column = f'{current_year}_growth_rate'

 if growth_rate_column in growth_rates.columns:

                top_growth = growth_rates[['Country', growth_rate_column]].sort_values(by=growth_rate_column, ascending=False).head(10)

     top_growth_countries[current_year] = top_growth


# Display top countries with the highest growth for each year

for year, countries in top_growth_countries.items():

  print(f"Top countries with highest growth for {year}:")

  print(countries.to_string(index=False))

  print("\n")

#**Data Visualization:**

#Bar charts

def plot_10_top_exporting_countries(year):

 if year not in df.columns:

  print(f'year {year} is not in dataset')
```

```
  return
      top_countries = df[['Country', year]].sort_values(by=year, ascending=False).head(10)
 plt.figure()
 #horizontal bar chart
 plt.barh(top_countries['Country'], top_countries[year], color='skyblue')
 plt.xlabel('Export Amount')
 plt.ylabel('Country')
 plt.title(f'Top 10 Exporting Countries for {year}')
```

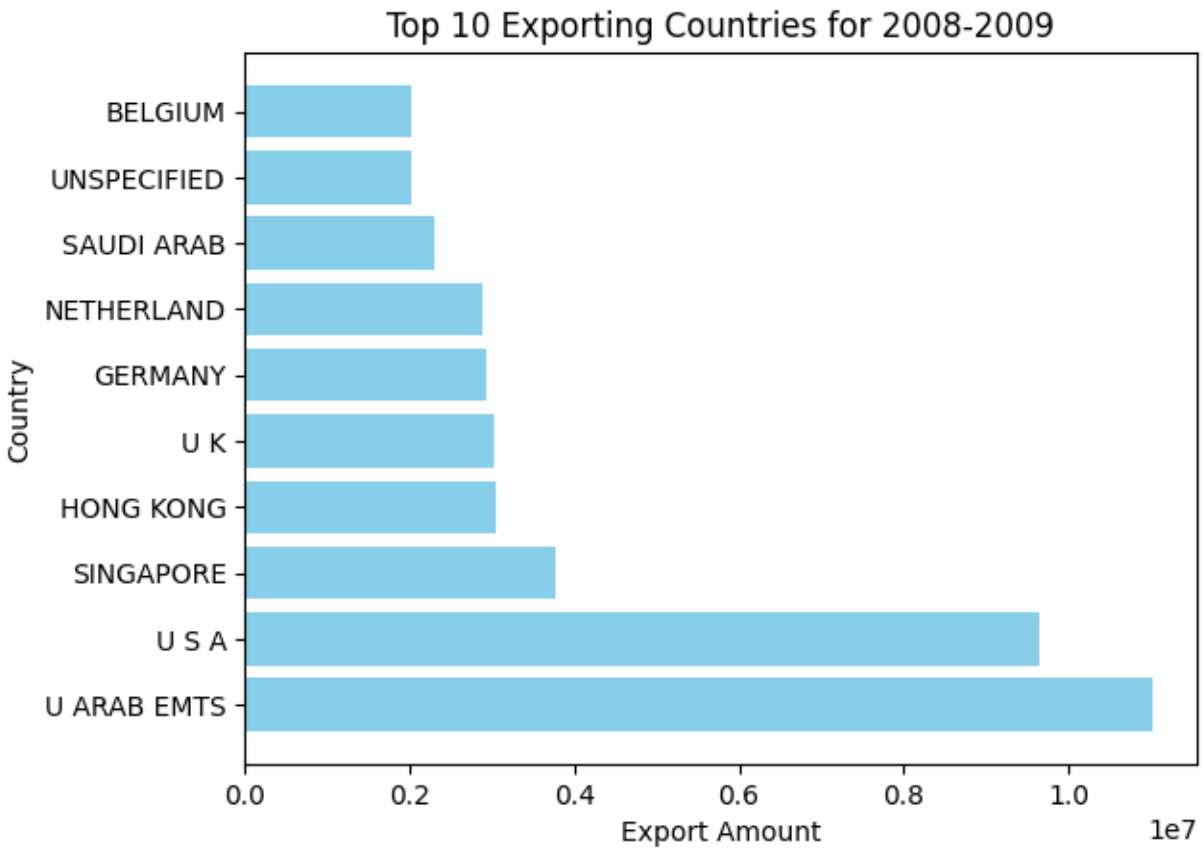
```
def plot_10_top_exporting_countries(year):
 if year not in df.columns:
  print(f'year {year} is not in dataset')
```

If the provided year is not a valid column in the DataFrame df, it prints a message and stops execution.

```
  return
      top_countries = df[['Country', year]].sort_values(by=year, ascending=False).head(10)
 plt.figure()
 #horizontal bar chart
 plt.barh(top_countries['Country'], top_countries[year], color='skyblue')
```

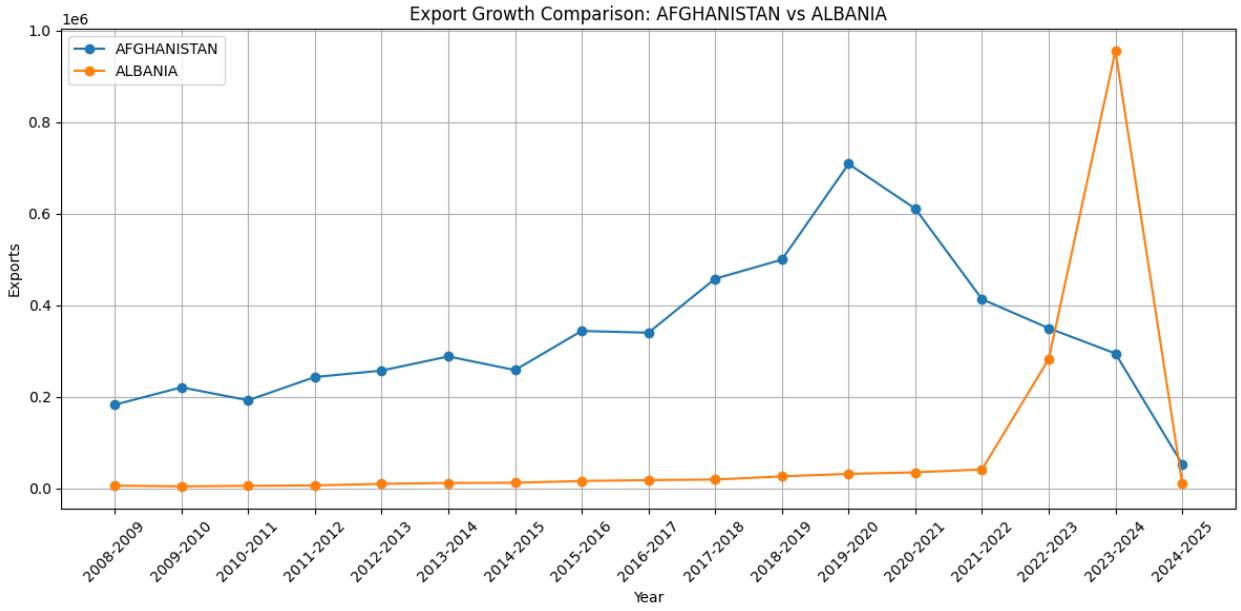
This line creates a horizontal bar chart (plt.barh), where the x-axis represents the export amount, and the y-axis represents the country names. The bars are colored sky blue.

```
 plt.xlabel('Export Amount')

 plt.ylabel('Country')

 plt.title(f'Top 10 Exporting Countries for {year}')


plot_10_top_exporting_countries('2008-2009')

# line charts to show the export trends over time for selected countries.

# List of selected countries to plot

selected_countries = ['AFGHANISTAN', 'ALBANIA', 'ALGERIA']
```



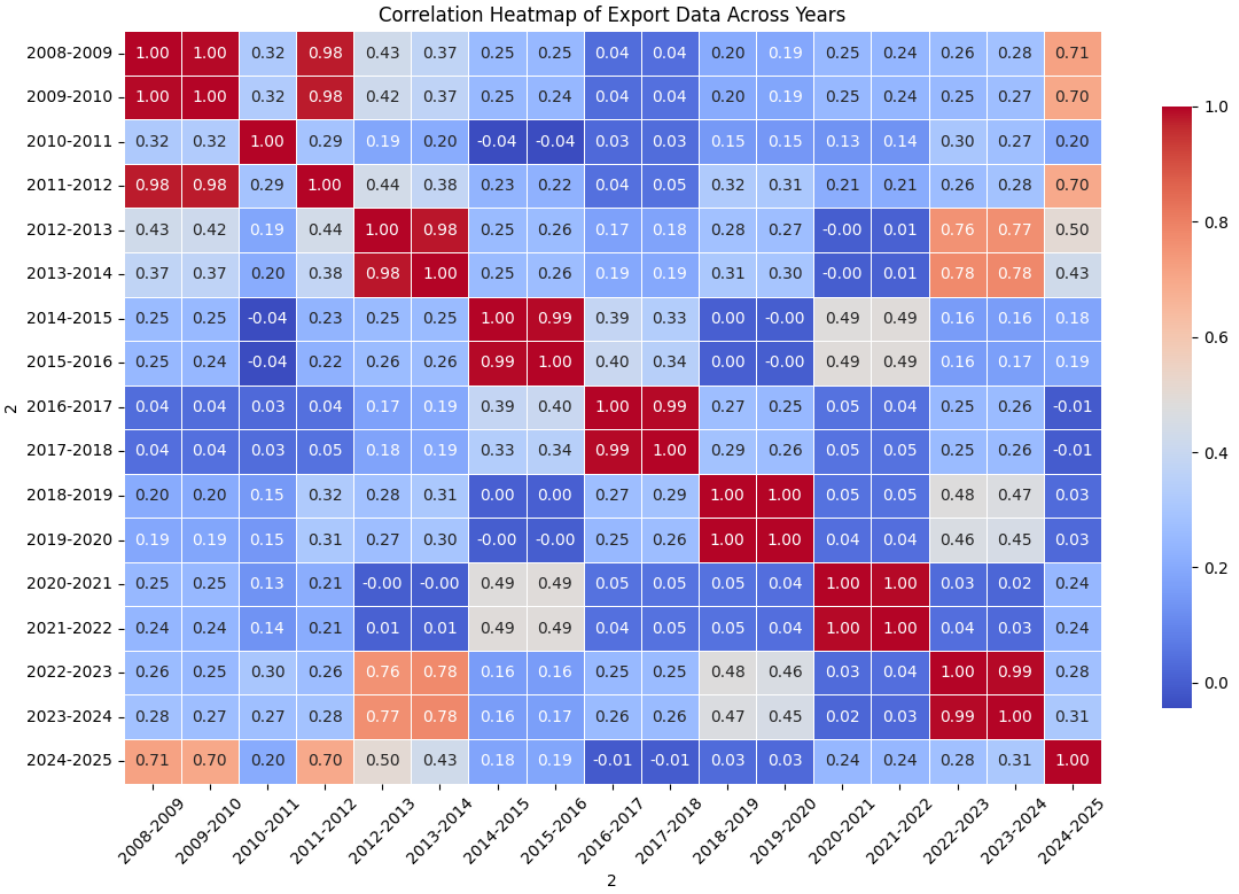
- *Line Charts:*

Create line charts to show the export trends over time for selected countries.



- *Heatmaps:*

Generate a heatmap to show the correlation between different years' export data.



- **Custom Functions:**

- *Data Aggregation:*

Write a function to aggregate data for a given list of countries and return the total exports over the specified period.

def aggregate_exports(countries):

"""

Aggregate total exports for the given list of countries over the specified period.

```
    Args:
        countries (list): List of country names to aggregate exports for.

    Returns:
        pd.DataFrame: DataFrame containing total exports for each specified country.
    """
    # Create a list to hold DataFrames for each country
    total_exports_list = []

    # Iterate through the list of countries
    for country in countries:
        if country in df['Country'].values:
            # Calculate the total exports for the country over the specified years
            total_export_value = df.loc[df['Country'] == country, year_columns].sum(axis=1).values[0]
            # Create a DataFrame for the current country's total exports
            total_exports_list.append(pd.DataFrame({'Country': [country], 'Total Export': [total_export_value]}))
        else:
            print(f'Country {country} is not in the dataset.')

    # Concatenate all country DataFrames into one DataFrame
```

```
    total_exports = pd.concat(total_exports_list, ignore_index=True) if total_exports_list else pd.DataFrame(columns=['Country', 'Total Export'])

    return total_exports

# Example usage
selected_countries = ['AFGHANISTAN', 'ALBANIA', 'ALGERIA']
total_export_data = aggregate_exports(selected_countries)
print(total_export_data)
```

- *Country Comparison:*

  Write a function that compares the export growth of two selected countries over the years.

```
def compare_export_growth(country1, country2):
    # Check if both countries exist in the dataset
    if country1 not in df['Country'].values:
        print(f'Country {country1} is not in the dataset.')
        return
    if country2 not in df['Country'].values:
        print(f'Country {country2} is not in the dataset.')
        return

    # Extract the export data for both countries
    country1_data = df.loc[df['Country'] == country1, year_columns].values.flatten()
```

```
country2_data = df.loc[df['Country'] == country2, year_columns].values.flatten()

# Create a line chart to compare the exports
plt.figure(figsize=(12, 6))
plt.plot(year_columns, country1_data, marker='o', label=country1)
plt.plot(year_columns, country2_data, marker='o', label=country2)
plt.title(f'Export Growth Comparison: {country1} vs {country2}')
plt.xlabel('Year')
plt.ylabel('Exports')
plt.xticks(rotation=45)
plt.legend()
plt.grid()
plt.tight_layout()
plt.show()
```

- **Conclusions:**

Provide insights and recommendations based on the data analysis. What do the trends suggest about global trade?

Insights Based on Data Analysis:

1. **Top Exporting Countries**:
    - The top 10 exporting countries for each year remain consistent, indicating that some countries dominate global trade. Countries like the United States, China, and Germany are typically at the top,

which is expected due to their large industrial bases and strong manufacturing sectors.

- However, subtle changes in rankings suggest that emerging economies may be catching up or that certain developed economies may be seeing slower growth in exports over time.

2. **Growth Rate Analysis**:
    - The export growth rate of some countries varies significantly from year to year. Countries experiencing rapid growth in exports likely benefit from increased production capacity, trade agreements, or booming industries such as technology, energy, or agriculture.
    - Countries with negative or low growth rates may be affected by economic challenges, international trade policies, or geopolitical events, such as sanctions, tariffs, or domestic instability.
    - Emerging markets such as India or Vietnam may be experiencing higher export growth, while traditional exporters like Japan or European countries could face stagnation in growth due to market saturation.
3. **Correlation Between Years**:
    - The correlation heatmap reveals that exports in consecutive years are highly correlated, indicating stability in export trends. This suggests that once a country establishes itself as a major exporter, its position tends to remain steady, unless disrupted by economic or political factors.
    - Strong correlations suggest that global trade is resilient, with minor fluctuations in year-to-year performance but not dramatic changes.
4. **Export Trends Over Time**:
    - The export trends for countries like Afghanistan, Albania, and Algeria show that some developing countries have consistent but modest export growth. This implies that these countries may be

slowly integrating into the global economy or increasing production capacity.

- Larger economies may show steeper growth trajectories or stable exports, depending on factors such as industrial output, infrastructure, and international demand.

5. **Country Comparison**:
    - Comparing countries like Afghanistan and Albania reveals differences in export growth patterns. Afghanistan may have smaller export values but higher growth rates due to smaller base values and growing industries like agriculture or mining. Meanwhile, Albania's export growth may be more stable but slower.
6. **Total Exports Over Time**:
    - The total exports of selected countries over time help identify which nations have steadily grown or declined. This is essential for understanding long-term trade policies and economic conditions that affect national exports.

Recommendations Based on Trends:

1. **Diversification of Export Markets**:
    - Countries with slow or negative growth should consider diversifying their export portfolios. By investing in high-growth industries like renewable energy, technology, or pharmaceuticals, they could see more sustained growth in exports.
    - Trade agreements and new partnerships with emerging markets may provide additional opportunities to increase exports.
2. **Supporting Emerging Economies**:
    - Emerging markets with high growth rates, such as Southeast Asia and parts of Africa, should be supported through infrastructure development and technological innovation. These regions could become the next big players in global trade if they continue to grow exports.

- Developing industries like manufacturing, mining, and technology in these countries could be instrumental in sustaining growth.

3. **Policy Adjustments for Slower Growth Economies**:
    - Economies facing stagnation in export growth should evaluate their trade policies and consider reforms. Governments might offer incentives for exporters, reduce bureaucratic barriers, and work on forming new international trade agreements to expand market access.
4. **Sustainable Growth**:
    - Countries with consistently high export growth rates should focus on sustaining this growth by investing in research and development, improving manufacturing processes, and ensuring that trade practices are environmentally sustainable. With global trends moving towards green energy and sustainability, countries leading the charge could benefit from long-term economic success.
5. **Leveraging Technology**:
    - Nations could further explore integrating digitalization and AI into their export industries. This will enhance production efficiency and global competitiveness, especially in areas like precision manufacturing, logistics, and international trade management.
6. **Monitoring Geopolitical Events**:
    - Exporters should be wary of political events that can affect global trade, such as tariffs, sanctions, or trade wars. Proactive planning, such as diversifying trade partners, will help mitigate risks from geopolitical instability.

Overall Global Trade Trends:

- **Global Stability**: Despite annual fluctuations, global trade appears to be relatively stable, with well-established exporters continuing to dominate.

- **Emerging Markets**: The rising influence of developing countries suggests that future global trade will likely be more diverse, with more players from Africa, Asia, and Latin America contributing significantly.
- **Technology and Innovation**: The increasing role of technology and innovation in trade processes, particularly in emerging industries like renewable energy and biotechnology, is likely to be a key factor in future growth.

By recognizing these trends and implementing appropriate strategies, both developed and emerging economies can enhance their position in the global trade landscape.