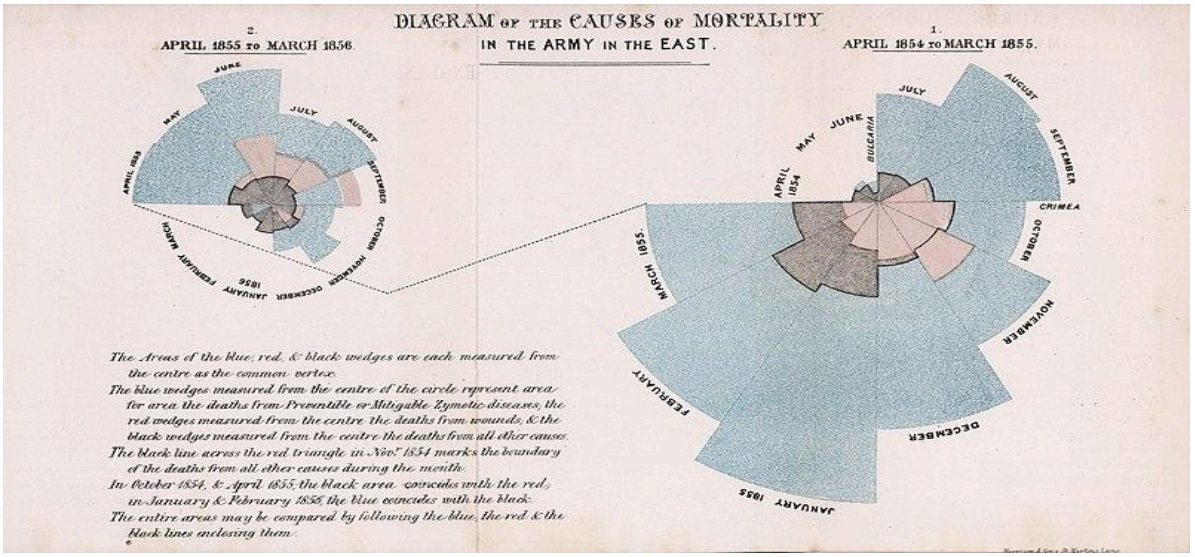
# REPORT LAB 2.1 - AI1703

## Activity 1:

1. Discuss the importance of data visualization in data analysis and decision making.
    - Simplify complex data: Visualization distills this data into easier-to-understand formats -> key insights are more easily grasped by stakeholders.
    - Pattern recognition: identify trends, patterns and outliers that may not be obvious in raw data -> more accurate predictions and insights.
    - Enhanced communication: Visualization is a powerful tool for communicating data insights clearly and effectively to diverse audiences, including those who may not have a diverse background. Strong in data analysis.
    - Informed decision making: By presenting data in a visual format, decision makers can quickly understand the meaning of the data, compare alternatives, and make evidence-based decisions proof.
    - Increase engagement: Visual data is often more engaging and memorable than text-based data, which can help maintain stakeholder attention and facilitate good discussions than.
2. Discuss the principles of effective data visualization, including clarity, precision, and storytelling.
    - Clear: Visual images must be easy to understand at a glance. This involves using appropriate chart types, labels, and annotations to ensure that the meaning of the data is clear.
    - Accuracy: Data must be presented honestly, without distortions that cause misunderstanding. This includes maintaining balance in the chart and avoiding visual manipulations that might confuse the viewer.
    - Storytelling: Good visualization will tell a story. It should have a clear story that guides viewers through the data, highlights the most important insights, and provides context.
3. Provide examples of well-designed visualizations and discuss their impact.
    + Example :  Florence Nightingale’s Coxcomb Chart

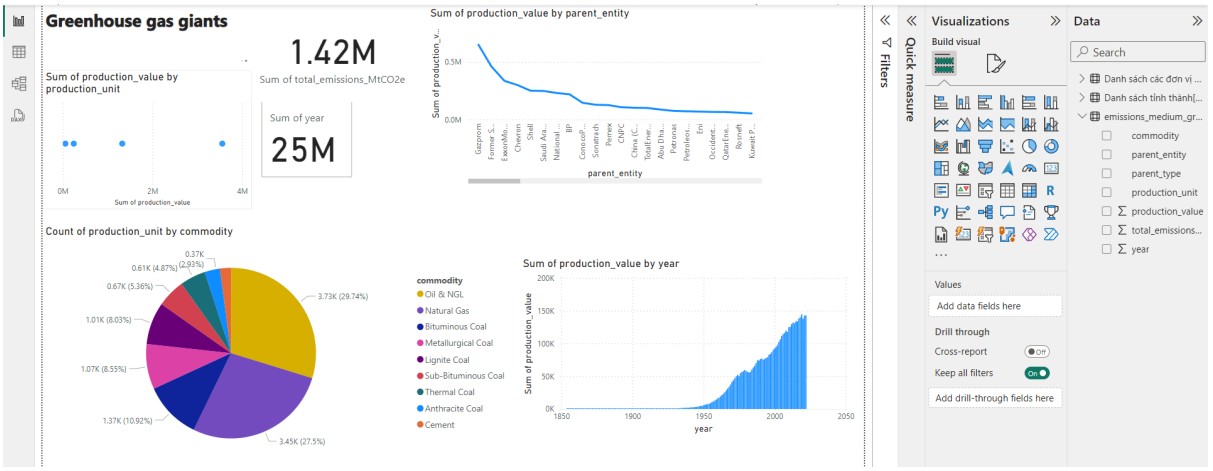
Visualization: Polar area diagram showing the causes of mortality in the Crimean War.
Impact: This historical visualization by Florence Nightingale highlighted the preventable deaths due to unsanitary conditions in hospitals. It was instrumental in driving health care reforms and improving sanitary practices in medical facilities.




# Activity 2:

1. matplotlib (Python library):
    - Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It is extensively used for creating static, animated, and interactive visualisations in Python.
    - It is highly customizable, allowing for a wide range of chart types including line plots, scatter plots, bar charts, and histograms.
2. Power BI:
    - Power BI is a business analytics service by Microsoft that provides interactive visualizations and business intelligence capabilities with a simple interface.
    - The interface integrates well with other Microsoft services. It offers features such as data connectivity, custom visualizations, and easy sharing of reports and dashboards.

Example :




# Activity 3:

The "Greenhouse Gas Giants" dataset comprises three CSV files (Emission High, Medium, and Low granularity) sourced from the project website Carbon Majors and made available on Kaggle through the following link: Greenhouse Gas Giants on Kaggle.

Our focus is on the Medium Granularity dataset, which contains 12,551 rows and 7 features. These features are:

1. **year**: The year of the recorded data.
2. **parent_entity**: The name of the parent entity responsible for the emissions.
3. **parent_type**: The type of the parent entity (e.g., state-owned, investor-owned, nation state).
4. **commodity**: The type of commodity produced by the entity.
5. **production_value**: The value of commodity production.
6. **production_unit**: The unit of measurement for the production value.
7. **total_emissions_MtCO2e**: The total emissions measured in million tonnes of CO2 equivalent (MtCO2e).

| | year | parent_entity | parent_type | commodity | production_value | production_unit | total_emissions_MtCO2e |
|---|---|---|---|---|---|---|---|
| 0 | 1962 | Abu Dhabi National Oil Company | State-owned Entity | Oil & NGL | 0.91250 | Million bbl/yr | 0.363885 |
| 1 | 1962 | Abu Dhabi National Oil Company | State-owned Entity | Natural Gas | 1.84325 | Bcf/yr | 0.134355 |
| 2 | 1963 | Abu Dhabi National Oil Company | State-owned Entity | Oil & NGL | 1.82500 | Million bbl/yr | 0.727770 |
| 3 | 1963 | Abu Dhabi National Oil Company | State-owned Entity | Natural Gas | 4.42380 | Bcf/yr | 0.322453 |
| 4 | 1964 | Abu Dhabi National Oil Company | State-owned Entity | Oil & NGL | 7.30000 | Million bbl/yr | 2.911079 |

**Figure 1: The Dataset of "Greenhouse Gas Giants"**

The dataset is comprehensive and does not contain any missing values, providing a solid foundation for analysis. So, we can skip that processing to step-by-step guide for performing Exploratory Data Analysis (EDA)

```
data.info()
✓ 0.0s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12551 entries, 0 to 12550
Data columns (total 7 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   year                   12551 non-null  int64
 1   parent_entity          12551 non-null  object
 2   parent_type            12551 non-null  object
 3   commodity              12551 non-null  object
 4   production_value       12551 non-null  float64
 5   production_unit        12551 non-null  object
 6   total_emissions_MtCO2e 12551 non-null  float64
dtypes: float64(2), int64(1), object(4)
memory usage: 686.5+ KB
```

**Figure 2 : The information of dataset**

```
# Show the amount of unique value in each column
dataset_without_total_column = raw.drop(columns=["total_emissions_MtCO2e","production_value"])
for column in dataset_without_total_column.columns:
    unique_value = dataset_without_total_column[column].unique()
    num_unique = len(unique_value)
    print(f"{column}: {num_unique} unique values")
    if num_unique < 10:
        print(unique_value)
    print("\n")
```

[29]

```
year: 169 unique values


parent_entity: 122 unique values


parent_type: 3 unique values
['State-owned Entity' 'Investor-owned Company' 'Nation State']


commodity: 9 unique values
['Oil & NGL' 'Natural Gas' 'Sub-Bituminous Coal' 'Metallurgical Coal'
 'Bituminous Coal' 'Thermal Coal' 'Anthracite Coal' 'Cement'
 'Lignite Coal']


production_unit: 4 unique values
['Million bbl/yr' 'Bcf/yr' 'Million tonnes/yr' 'Million Tonnes CO2']
```

**Figure 3 : The amount of unique value in each column**

The objective of this analysis is to determine the number of unique values in each column of a dataset, excluding the columns "total_emissions_MtCO2e" and "production_value".

## Activity 4,5,6:

Create, customising visulisation and presenting insights

1. Bar chart

```
# Bar chart of total production value of commodities
group_data_by_commodity = raw.groupby(raw['commodity'])['production_value'].sum().reset_index()
print(group_data_by_commodity)
```

✓ 0.0s

| | commodity | production_value |
|---|---|---|
| 0 | Anthracite Coal | 1.627872e+04 |
| 1 | Bituminous Coal | 1.218675e+05 |
| 2 | Cement | 5.663804e+04 |
| 3 | Lignite Coal | 3.187816e+04 |
| 4 | Metallurgical Coal | 2.951517e+04 |
| 5 | Natural Gas | 3.555407e+06 |
| 6 | Oil & NGL | 1.324404e+06 |
| 7 | Sub-Bituminous Coal | 2.466162e+04 |
| 8 | Thermal Coal | 1.930153e+04 |

**Figure 4 : Raw the bar chart of total production of commodities**

In this bar chart we group the raw DataFrame data by the value in the 'commodity' column then we sum the values of the 'production_value' column in each group created and convert the data into a regular dataframe. This helps to aggregate production data by commodity, making it easy to analyze which commodity has the highest, lowest total production value, or how production is distributed among different commodities.

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,6))
sns.barplot(x='commodity', y='production_value', data=group_data_by_commodity)
plt.title('Total amount of production_value of commodity')
plt.xlabel('Commodity')
plt.ylabel('Total product value')
plt.show()
```

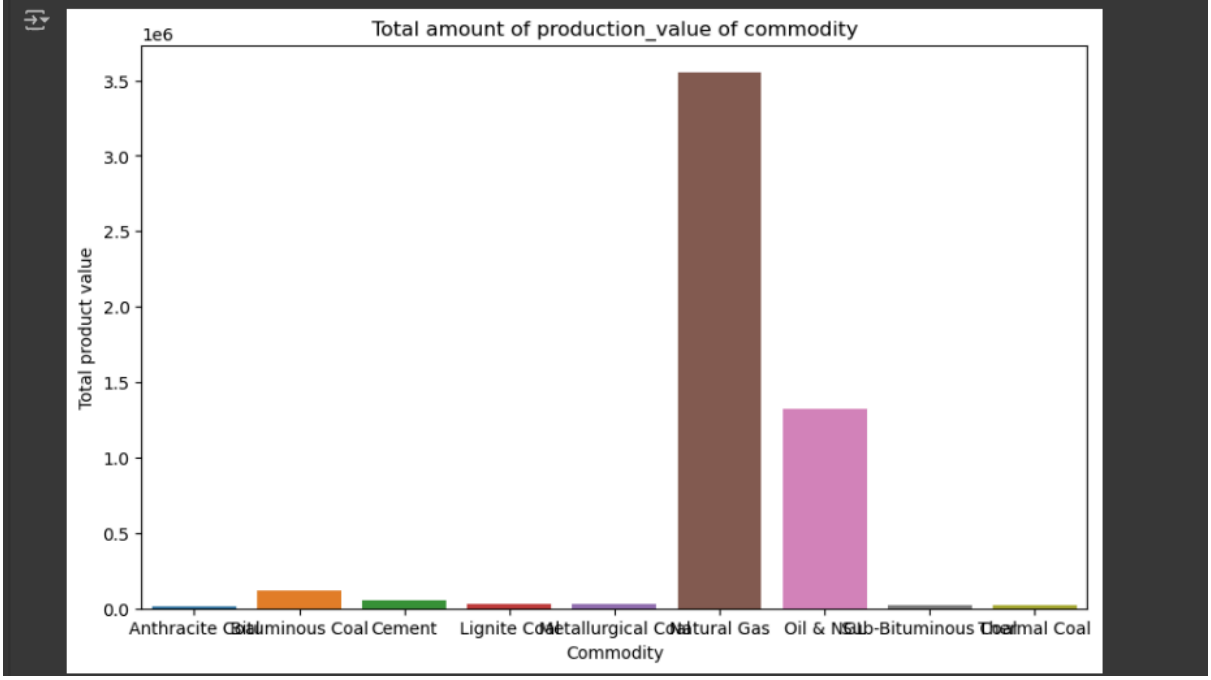



**Figure 5 : Bar chart of total production of commodities**

The bar chart for total production of commodities shows the total production value by product type, in which natural gas and oil & NGL have the highest quantity, however the chart has many limitations and is difficult to see.

```python
import matplotlib.pyplot as plt
import seaborn as sns
data_sorted = group_data_by_commodity.sort_values(by="production_value", ascending=True) # Sắp xếp giảm dần theo production_value

plt.figure(figsize=(10,6))
sns.barplot(x='commodity', y='production_value', data=data_sorted, palette='viridis')

plt.yscale('log') # Log scale

plt.title('Total amount of production_value of commodity')
plt.xlabel('Commodity')
plt.ylabel('Total product value (log scale)')

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

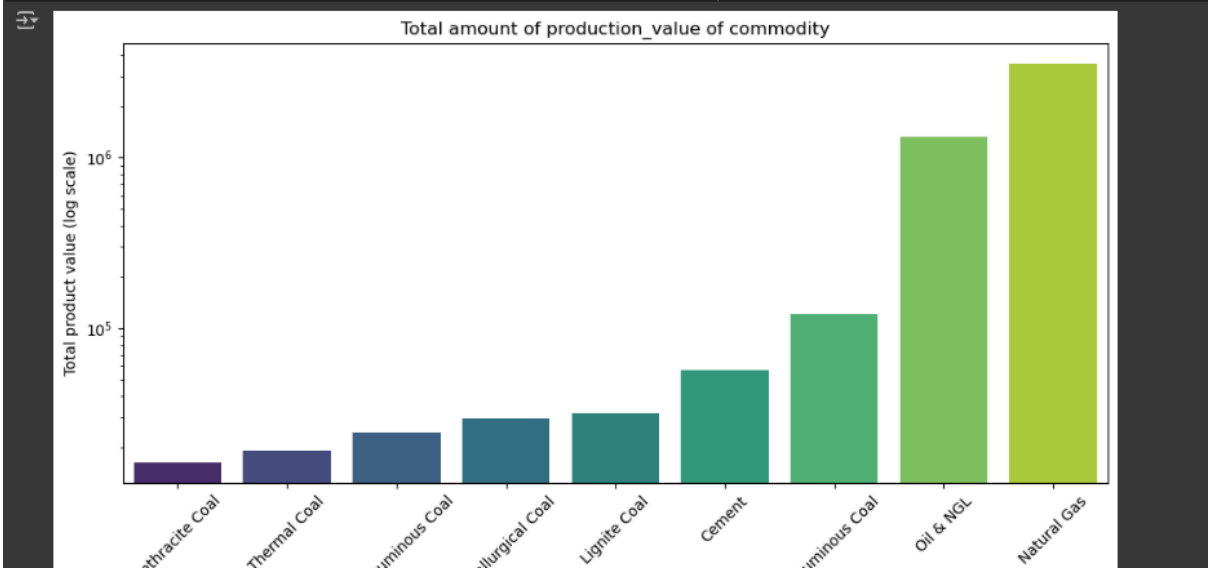



**Figure 6 : Bar chart of total production of commodities use logarithmic**

In this chart we have used a logarithmic scale to make it easier to see and we have customised the colours to clearly show the difference between the products. As shown above, we can see that Natural Gas has the highest quantity and Anthracite Coal has the lowest quantity.

```
data_sorted = group_data_by_commodity.sort_values(by="production_value", ascending=False)
plt.figure(figsize=(10, 6))
sns.barplot(x='production_value', y='commodity', data=data_sorted, palette='viridis')

plt.xscale('log')

plt.xlabel('Production Value (log scale)')
plt.ylabel('Commodity')
plt.title('Production Value by Commodity')

for index, value in enumerate(data_sorted['production_value']):
    plt.text(value, index, f'{value:.0f}', va='center')

plt.tight_layout()
plt.show()
```

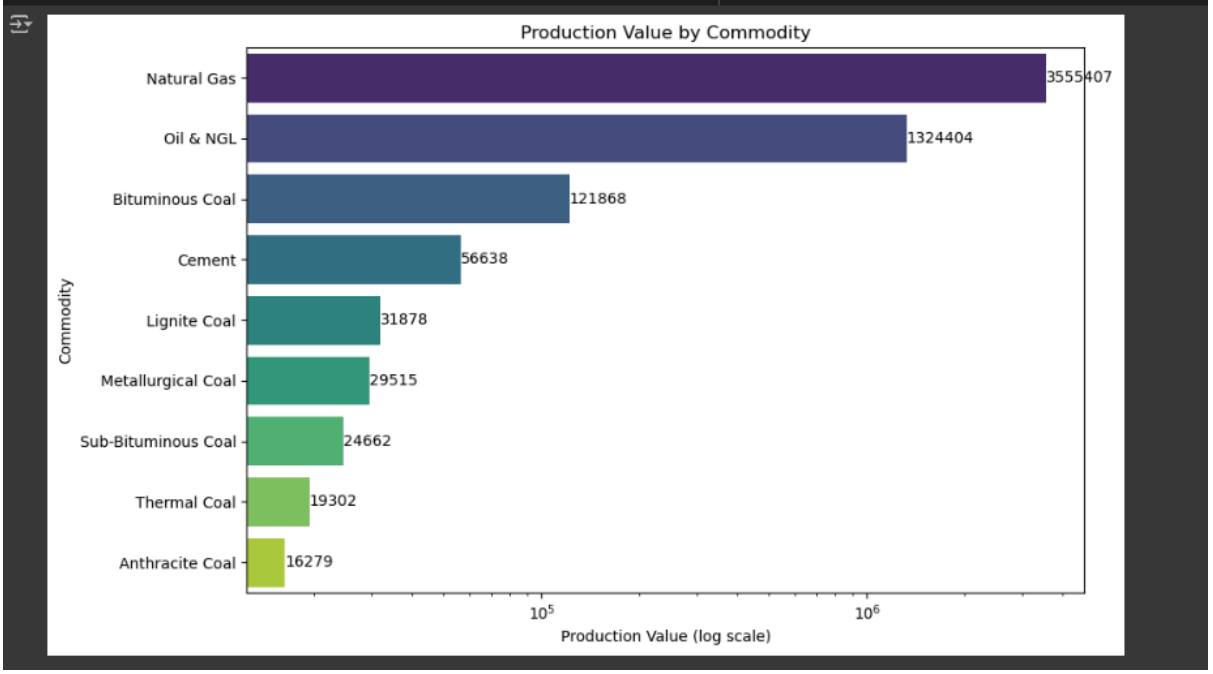



**Figure 7 : Bar chart shows specific data of total goods output**

To show the reality here, we have included specific data to easily visualize the difference between each product.

## 2. Scatter plot

```
data_pvt = raw.pivot_table(values = ["production_value", "total_emissions_MtCO2e"], index = "year", aggfunc = "sum").reset_index()
data_pvt.head()
```

| | year | production_value | total_emissions_MtCO2e |
|---|---|---|---|
| 0 | 1854 | 0.051005 | 0.099198 |
| 1 | 1855 | 0.066326 | 0.128996 |
| 2 | 1856 | 0.081647 | 0.158793 |
| 3 | 1857 | 0.094906 | 0.184580 |
| 4 | 1858 | 0.108165 | 0.210367 |

**Figure 8 : Create a Pivot Table total production value of each type of goods for the scatter plot**

Like the bar chart above, we create a pivot table to group "production value" and "total_MtCO2e_emissions" and rearrange the data into a summary table that we then "sum" to use to calculate the price. total value for each year. This table will have the number of years of production and show the total production value and total emissions per year.

The chart illustrates the correlation between the amount of ONGL products and the amount of CO2 emissions released into the environment.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(10,6))
plt.scatter(data_pvt['production_value'],data_pvt['total_emissions_MtCO2e'])

plt.title("The relationship between the amount of ONGL production and total emissions of CO2")
plt.xlabel("The amount of ONGL production")
plt.ylabel("total emissions of CO2 (Million Tonnes)")

plt.show()
```

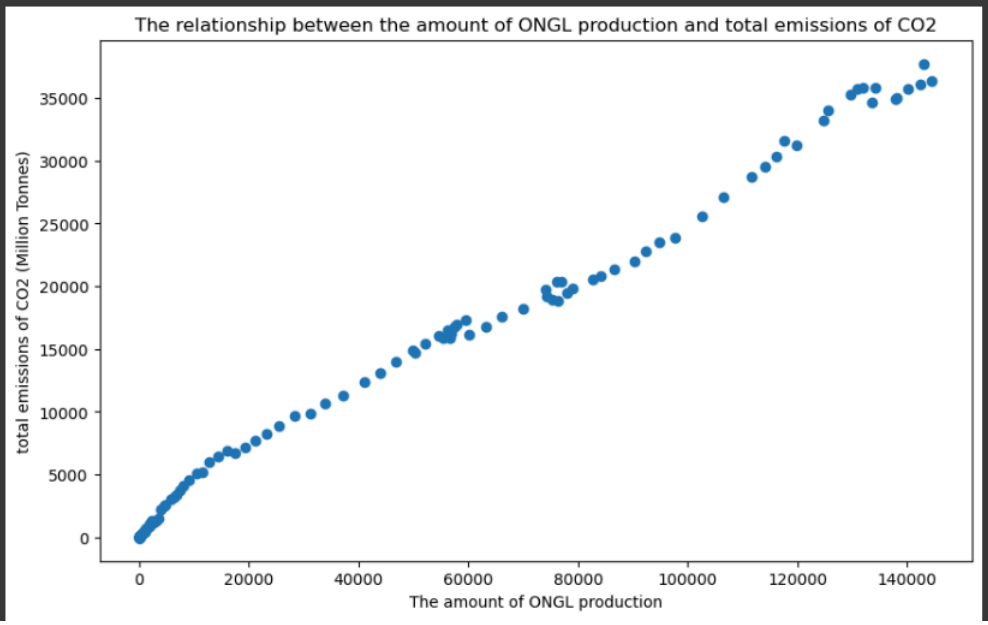



**Figure 9 : The relationship between the amount of ONGL production and total emissions of CO2**

We use a scatter plot to effectively visualize the relationship between ONGL production and CO2 emissions, showing a strong positive correlation. We do see more and more CO2 emissions in the production of ONGL. Additionally, the model highlights the environmental challenges posed by increased ONGL production and the need for strategic interventions to manage and reduce CO2 emissions.

## 3. Line chart

```
[ ] # Create pivot mean table
    data_pvt = raw.pivot_table(values = ["production_value", "total_emissions_MtCO2e"], index = "year", aggfunc = "mean").reset_index()
    data_pvt.head()
```

| | year | production_value | total_emissions_MtCO2e |
|---|---|---|---|
| 0 | 1854 | 0.017002 | 0.033066 |
| 1 | 1855 | 0.022109 | 0.042999 |
| 2 | 1856 | 0.027216 | 0.052931 |
| 3 | 1857 | 0.031635 | 0.061527 |
| 4 | 1858 | 0.036055 | 0.070122 |

**Figure 10 : Create a Pivot mean Table total production value of each type of goods for the Line Chart**

Like the chart above, we group these code snippets and rearrange the data into a summary table. The difference compared to the above charts is that we use mean to calculate the average. Therefore, the result in the pivot table will be the average value of "production_value" and "total_emissions_MtCO2e" for each year.

```
[ ] # Plot data
    fig,ax = plt.subplots(figsize=(12,8))

    chart = sns.lineplot(x='year',
                         y='value',
                         hue='variable',
                         data=pd.melt(data_pvt,['year']),
                         color=sns.color_palette()[0],)

    chart.set(xlabel='Year', ylabel='Mean Value')
    # ONGL (oil and natural gas liquids)
    plt.title('Avg. global ONGL production & co2 emissions over the years')
    plt.grid(True)
    plt.show()
```

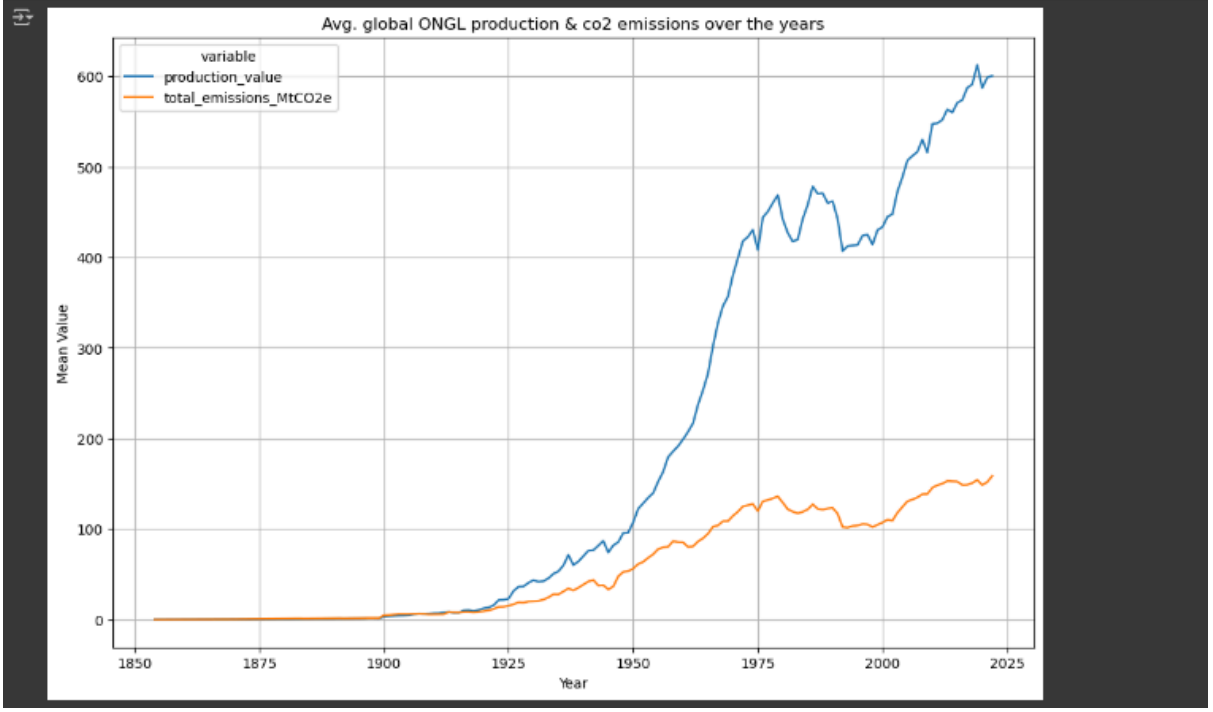



**Figure 11 : Avg. global ONGL production & co2 emissions over the years**

The total production of ONGL is directly proportional to the amount of CO2 emissions, but the correlation between these two factors varies from year to year. Specifically, ONGL production increases rapidly over time, while CO2 emissions increase but at a much slower pace.