**Week 2 - Lab Exercise: Pandas and DataFrames**

**📌 Instructions**

This lab will guide you through **15 questions** to practice Pandas operations.

**NOTE**: remember to activate your python environment (revise week 1 – lab exercise)

* The first **5 questions** use a simple DataFrame to build your understanding.
* The next **10 questions** use real-world datasets:
  + GlobalLandTemperaturesByCountry.csv
  + owid-co2-data.csv
  + owid-covid-data.csv

💡 **Important:**

* **For COVID-19 data**, focus on **2021** to avoid missing values.
* **For CO₂ and temperature data**, use **years before 2023** to get meaningful outputs.
* **Repeat the examples** with different values (e.g. different countries…etc)

**🔹 Part 1: Working with a Simple DataFrame (Q1 - Q5)**

**Q1: Create and Display a DataFrame**

Run the code below and observe the output.

python

import pandas as pd

# Sample DataFrame

df = pd.DataFrame({

'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva'],

'Age': [25, 30, 35, 40, 22],

'City': ['London', 'Paris', 'Berlin', 'Madrid', 'Rome']

})

print(df)

**📝 Question:** What are the column names? How are the rows indexed by default?

**Q2: Select Specific Columns and Rows**

python

# Select the 'Name' and 'Age' columns

df\_subset = df[['Name', 'Age']]

print(df\_subset)

# Select the row for 'Charlie' using .loc[]

charlie\_row = df.loc[2]

print(charlie\_row)

**📝 Question:** What is the difference between selecting with df[['Name', 'Age']] and df['Name']?

**Q3: Add and Delete a Row**

python

# Adding a new row

df.loc[5] = ['Frank', 28, 'Amsterdam']

print(df)

# Deleting Bob's row

df = df.drop(1)

print(df)

**📝 Question:** Why do we assign df = df.drop(1) instead of just using df.drop(1)?

**Q4: Filtering and Sorting**

python

# Select only people older than 30

df\_filtered = df[df['Age'] > 30]

print(df\_filtered)

# Sorting by Age in descending order

df\_sorted = df.sort\_values(by='Age', ascending=False)

print(df\_sorted)

**📝 Question:** What happens if you change ascending=False to ascending=True?

**Q5: Grouping and Aggregation**

python

# Grouping by City and calculating average Age

df\_grouped = df.groupby('City')['Age'].mean()

print(df\_grouped)

**📝 Question:** What does .groupby('City')['Age'].mean() do?

**🔹 Part 2: Working with Real-World Datasets (Q6 - Q15)**

**📌 Load the Datasets**

python

# Load datasets

df\_temp = pd.read\_csv('GlobalLandTemperaturesByCountry.csv')

df\_co2 = pd.read\_csv('owid-co2-data.csv')

df\_covid = pd.read\_csv('owid-covid-data.csv')

**Q6: Exploring the COVID-19 Data**

python

# Display dataset info

print(df\_covid.info())

# Select COVID-19 data for 2021

df\_covid = df\_covid[df\_covid['date'].str.startswith('2021')]

print(df\_covid.head())

**📝 Question:** What percentage of rows contain missing values?

**Q7: Filtering COVID-19 Data for a Specific Country**

python

# Select data for Vietnam in 2021

df\_vietnam = df\_covid[df\_covid['location'] == 'Vietnam']

print(df\_vietnam.tail())

**📝 Question:** What is the total number of COVID-19 cases recorded in Vietnam in 2021?

**Q8: Handling Missing Values in COVID-19 Data**

python

# Fill missing values in new\_cases with 0

df\_covid['new\_cases'] = df\_covid['new\_cases'].fillna(0)

# Drop rows where total\_cases is NaN

df\_covid = df\_covid.dropna(subset=['total\_cases'])

**📝 Question:** What impact does filling missing values with 0 have on the dataset?

**Q9: Exploring CO₂ Emissions Data**

python

# Select CO₂ data for 2021

df\_co2 = df\_co2[df\_co2['year'] == 2021]

# Display the top 5 rows

print(df\_co2.head())

**📝 Question:** What are the top 5 countries with the highest CO₂ emissions in 2021?

**Q10: Finding the Highest and Lowest CO₂ Emissions**

python

# Find the country with the highest CO₂ emissions

highest\_co2 = df\_co2[df\_co2['co2'] == df\_co2['co2'].max()]

print(highest\_co2)

**📝 Question:** What country had the lowest recorded CO₂ emissions in 2021?

**Q11: Exploring Temperature Data**

python

# Select only temperature data from before 2023

df\_temp = df\_temp[pd.to\_datetime(df\_temp['dt']).dt.year < 2023]

# Display dataset info

print(df\_temp.info())

**📝 Question:** What are the top 3 hottest recorded years?

**Q12: Handling Missing Temperature Data**

python

# Drop rows where AverageTemperature is NaN

df\_temp = df\_temp.dropna(subset=['AverageTemperature'])

print(df\_temp.head())

**📝 Question:** What happens to the dataset size after dropping NaN values?

**Q13: Grouping Temperature Data by Country**

python

# Find the average temperature for each country

df\_avg\_temp = df\_temp.groupby('Country')['AverageTemperature'].mean()

print(df\_avg\_temp.head())

**📝 Question:** What country had the highest average temperature?

**Q14: Sorting Temperature Data**

python

# Sorting by temperature in descending order

df\_sorted\_temp = df\_avg\_temp.sort\_values(ascending=False)

print(df\_sorted\_temp.head(10))

**📝 Question:** What are the top 10 hottest countries?

**Q15: Exporting Cleaned Data to CSV**

python

df\_co2.to\_csv('cleaned\_co2\_data.csv', index=False)

df\_covid.to\_csv('cleaned\_covid\_data.csv', index=False)

df\_temp.to\_csv('cleaned\_temp\_data.csv', index=False)

**📝 Question:** Why is it useful to export cleaned datasets?

**🔹 Part 3: Visualisation Using Pandas DataFrames (Q16 - Q20)**

Pandas allows us to generate basic visualisations directly using .plot(). In this section, you will **create visualisations** using Pandas and **avoid missing values** to ensure meaningful insights.

Before starting, install **Matplotlib** if needed:

bash

pip install matplotlib

Import Matplotlib at the start of your script:

python

import matplotlib.pyplot as plt

**Q16: Plot CO₂ Emissions Over Time for a Specific Country**

Use the owid-co2-data.csv file and filter data for a country of your choice (e.g., United Kingdom).

python

# Load CO₂ data

df\_co2 = pd.read\_csv('owid-co2-data.csv')

# Select data for a specific country and filter years before 2023

df\_uk\_co2 = df\_co2[(df\_co2['country'] == 'United Kingdom') & (df\_co2['year'] < 2023)]

# Drop rows with missing CO₂ values

df\_uk\_co2 = df\_uk\_co2.dropna(subset=['co2'])

# Plot CO₂ emissions over time

df\_uk\_co2.plot(x='year', y='co2', kind='line', title='CO₂ Emissions in the UK Over Time')

plt.xlabel('Year')

plt.ylabel('CO₂ Emissions (Million Tonnes)')

plt.show()

**📝 Question:** What trends do you notice in CO₂ emissions?

**Q17: Bar Chart - Top 10 CO₂ Emitting Countries (2021)**

python

# Select only 2021 data

df\_co2\_2021 = df\_co2[df\_co2['year'] == 2021]

# Drop NaN values in CO₂ emissions

df\_co2\_2021 = df\_co2\_2021.dropna(subset=['co2'])

# Get top 10 emitting countries

df\_top10\_co2 = df\_co2\_2021.sort\_values(by='co2', ascending=False).head(10)

# Plot a bar chart

df\_top10\_co2.plot(x='country', y='co2', kind='bar', title='Top 10 CO₂ Emitting Countries in 2021')

plt.xlabel('Country')

plt.ylabel('CO₂ Emissions (Million Tonnes)')

plt.xticks(rotation=45)

plt.show()

**📝 Question:** Which countries contributed the most to CO₂ emissions in 2021?

**Q18: Temperature Trends Over Time for a Country**

Use the GlobalLandTemperaturesByCountry.csv dataset.

python

# Load temperature data

df\_temp = pd.read\_csv('GlobalLandTemperaturesByCountry.csv')

# Convert date column to datetime

df\_temp['dt'] = pd.to\_datetime(df\_temp['dt'])

# Filter for a specific country and years before 2023

df\_uk\_temp = df\_temp[(df\_temp['Country'] == 'United Kingdom') & (df\_temp['dt'].dt.year < 2023)]

# Drop NaN values

df\_uk\_temp = df\_uk\_temp.dropna(subset=['AverageTemperature'])

# Plot temperature trends

df\_uk\_temp.plot(x='dt', y='AverageTemperature', kind='line', title='Temperature Trends in the UK')

plt.xlabel('Year')

plt.ylabel('Average Temperature (°C)')

plt.show()

**📝 Question:** Can you identify warming trends in the UK?

**Q19: COVID-19 Cases vs. Deaths (2021) - Scatter Plot**

Use the owid-covid-data.csv dataset and focus on 2021.

python

# Load COVID-19 data

df\_covid = pd.read\_csv('owid-covid-data.csv')

# Select only 2021 data and drop NaNs

df\_covid\_2021 = df\_covid[df\_covid['date'].str.startswith('2021')].dropna(subset=['total\_cases', 'total\_deaths'])

# Plot scatter plot

df\_covid\_2021.plot(x='total\_cases', y='total\_deaths', kind='scatter', title='COVID-19 Cases vs. Deaths (2021)')

plt.xlabel('Total Cases')

plt.ylabel('Total Deaths')

plt.show()

**📝 Question:** Is there a correlation between total cases and total deaths?

**Q20: Comparing CO₂ Emissions and Temperature Change for a Country**

This combines data from **two datasets (CO₂ and Temperature)** to explore their relationship.

python

# Merge CO₂ and temperature data on year

df\_uk\_co2['year'] = df\_uk\_co2['year'].astype(int) # Ensure year is an integer

df\_uk\_temp['year'] = df\_uk\_temp['dt'].dt.year

# Group temperature data by year

df\_uk\_avg\_temp = df\_uk\_temp.groupby('year')['AverageTemperature'].mean().reset\_index()

# Merge datasets

df\_combined = pd.merge(df\_uk\_co2, df\_uk\_avg\_temp, on='year', how='inner')

# Drop NaN values

df\_combined = df\_combined.dropna(subset=['co2', 'AverageTemperature'])

# Plot CO₂ vs Temperature

df\_combined.plot(x='co2', y='AverageTemperature', kind='scatter', title='CO₂ vs Temperature in the UK')

plt.xlabel('CO₂ Emissions (Million Tonnes)')

plt.ylabel('Average Temperature (°C)')

plt.show()

**📝 Question:** Does increasing CO₂ correlate with rising temperatures?

✅ **Lab complete! If you haven’t changed the code by trying different examples, please do so. If you have done that, now move to the Lab Assignment.**