Librairies importation

```
import pandas as pd
import matplotlib.pyplot as plt
```

Dataset importation after being downloaded from World Bank group

```
#data = pd.read_excel("API_DZA_DS2_en_excel_v2_21660.xlsx" , index_col = 0 , sheet_name = "Data")
#data = pd.read_excel("API_MAR_DS2_en_excel_v2_41281.xlsx" , index_col = 0 , sheet_name = "Data")
data = pd.read_excel("API_TUN_DS2_en_excel_v2_21951.xlsx" , index_col = 0 , sheet_name = "Data")
```

```
# add index
data = data.reset_index()
```

```
# view the first 5 rows
data.head()
```

```
# use of melt() function to resize the new dataset and convert some columns to rows
data = pd.melt(data , id_vars = ['Indicator Name' , 'Country Name' , 'Country Code' , 'Indicator Code'] , var_name = 'Year' , value_
```

```
# view data after be melted
data.head()
```

```
# define a list of macroeconomic indicators
indicator = ["GNI (constant 2015 US$)" , "Gross capital formation (constant 2015 US$)"
            , "Population, total" , "Official exchange rate (LCU per US$, period average)" , "GNI per capita (constant 2015 US$)"]
```

```
# change data to keep just the selected indicators
data = data[data["Indicator Name"].isin(indicator)]
```

```
# reset index
data = data.reset_index(drop = True)
```

```
# view the new data
data.head()
```

```
# view the data information to observe column type and null values
data.info()
```

```
# change the data type of "Year" column
data["Year"] = pd.to_datetime(data["Year"], format="%Y")
```

```
# count of the NaN values (that must be repere after)
data[data["value"].isna()].loc[: , "Year"].count()
```

```
15
```

Replacing the missing values

Outlier check

```
# empty dictionnarie
d = {}
```

```
# Searching of outliers, if they existe then put them in (d) dictionnarie
for x in indicator:
    df = data[data["Indicator Name"] == x].reset_index(drop = True)
    Q1 = df["value"].quantile(0.25)
    Q3 = df["value"].quantile(0.75)
    IQR = Q3 - Q1
    sup = Q3 + IQR
    inf = Q1 - IQR
    outliers = df[(df["value"] < inf) | (df["value"] > sup)].reset_index(drop = True)
    d[x] = outliers
```

Replace outlier with the mean

```
# empty dictionnary
Mean = {}
```

```
# Creation of a dictionary of means after removing outliers
for x in indicator :
    if len(d[x]) > 0:
       df = data[data["Indicator Name"] == x].reset_index(drop = True)
       df["is_outlier"] = df.index.isin(d[x].index)
       df = df[df["is_outlier"] == False]
       Mean[x] = df.value.mean()
```

```
# Creating another emty dictionnary
Mean2 = {}
```

```
# store the mean value of all idicators in Mean2 (rather if they contain outliers or not)
for x in indicator :
    df = data[data["Indicator Name"] == x].reset_index(drop = True)
    Mean2[x] = df.value.mean()
```

```
# Merging the two dictionaries to create a final mean (without outliers)
Mean = {**Mean , **Mean2}
```

```
# Filling empty values with the corresponding mean values
for el in data["Indicator Name"].unique():
    mask = data["Indicator Name"] == el
    if data.loc[mask, "value"].isna().any():
        data.loc[mask, "value"] = data.loc[mask, "value"].fillna(Mean[el])
```

```
# Ensuring that ther is no more empty value in the data
data.info()
```

```
# store the final data into an excel file
data.to_excel("algeria.xlsx")
#data.to_excel("tunisia.xlsx")
#data.to_excel("morocco.xlsx")
```

Merge datasets

```
# import all datasets of the three countries
al = pd.read_excel("al.xlsx" , index_col = 0)
tu = pd.read_excel("tu.xlsx" , index_col = 0)
ma = pd.read_excel("ma.xlsx" , index_col = 0)
```

```
# Use the concat function to concatinate all the three datasets together in one dataset
dataset = pd.concat([al , tu , ma] , ignore_index = True)
```

```
# reindexing the new dataset
dataset = dataset.reset_index(drop = True)
```

```
# Cheking the dataset
dataset.info()
```

```
# Saving the dataset into an excel file
dataset.to_excel("maghreb_macro_economic.xlsx")
```

Creating dimension tables (Indicators , Countries)

```
# Import the last dataset
data = pd.read_excel("maghreb_macro_economic.xlsx" , index_col = 0)
```

```
# Extracting the Indicator Name and the Indicator Code from the dataset then droping the duplicates values
indicators = data.loc[: , ["Indicator Name" , "Indicator Code"]].drop_duplicates()
```

```
countries = data.loc[: , ["Country Name" , "Country Code"]].drop_duplicates().reset_index(drop = True)
```

```
# Make a simple rectification on the name of an indicator
indicators = indicators.replace("Population, total" , "Population")
data = data.replace("Population, total" , "Population")
```

```
# Save indicators into an excel file
indicators.to_excel("Indicators.xlsx")
```

```
# Save countries into an excel file
countries.to_excel("Countries.xlsx")
```