In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplab as plt
from datetime import datetime
import seaborn as sns
from scipy import stats as st
```

In [2]:
```
#Importation des données pour 2013 et la population 2014
population2013 = pd.read_csv('population2013.csv', delimiter = ";")
population2014 = pd.read_csv('population2014.csv', delimiter = ";")
animaux = pd.read_csv('fr_animaux.csv', delimiter = ";")
vegetaux = pd.read_csv('fr_vegetaux.csv', delimiter = ";")
pib13 = pd.read_csv('PIB2013.csv', delimiter = ";")
#je supprime la chine car elle représentée deux fois
animaux.drop(animaux.index[animaux['Zone'] == "Chine"], inplace = True)
vegetaux.drop(vegetaux.index[vegetaux['Zone'] == "Chine"], inplace = True)
#Je selection les colonnes d'interet
veg = pd.DataFrame(data=vegetaux, columns = ["Zone", "Disponibilite alimentaire (Kcal/personne/jour)",
                                             "Disponibilite alimentaire en quantite (kg/personne/an)",
                                             "Disponibilite de proteines en quantite (g/personne/jour)",
                                             "Disponibilite interieure (kg/an)"])
anim = pd.DataFrame(data=animaux, columns = ["Zone", "Disponibilite alimentaire (Kcal/personne/jour)",
                                             "Disponibilite alimentaire en quantite (kg/personne/an)",
                                             "Disponibilite de proteines en quantite (g/personne/jour)",
                                             "Disponibilite interieure (kg/an)"])
```

In [3]:
```
#Pour le PIB je selectionne uniquement la colone indiquant le PIB
pib13.drop(pib13.index[pib13['Zone'] == "Chine"], inplace = True)
pib = pd.DataFrame(data=pib13, columns = ["Zone", "Valeur"])
pib["PIB"] = pib["Valeur"]*1000000  #multiplication par 1de6
pib1 = pd.DataFrame(data=pib, columns = ["Zone", "PIB"])
pib1.head()
```

Out[3]:

	Zone	PIB
0	Afghanistan	1.962180e+10
1	Afrique du Sud	3.666449e+11
2	Albanie	1.277628e+10
3	Algerie	2.097550e+11
4	Allemagne	3.732743e+12

In [4]:
```
#Preparation du donnée vegetaux pour la merge, je selctionne uniquement la colonne de la disponibilité des proteines
veg1 = pd.DataFrame(data = veg, columns = ["Zone", "Disponibilite de proteines en quantite (g/personne/jour)"])
veg1["Disponibilite de proteines en quantite (g/personne/jour)"]=veg1["Disponibilite de proteines en quantite (g/personne/jour)"]*365
veg2 = veg1.rename(columns={"Disponibilite de proteines en quantite (g/personne/jour)": "Disponibilite de proteines vegetale en quantite (kg/personne/an)"})
veg3 = veg2.groupby(["Zone"]).sum().reset_index()
veg3.head()
```

Out[4]:

	Zone	Disponibilite de proteines vegetale en quantite (kg/personne/an)
0	Afghanistan	46.05
1	Afrique du Sud	48.94
2	Albanie	51.96
3	Algerie	66.94
4	Allemagne	39.90

In [5]:
```
#preparation des données 'animaux',
#Harmonisation des unitées en kg et pour 1 année
anim["Disponibilite de proteines animale en quantite (kg/personne/an)"] = (anim["Disponibilite de proteines en quantite (g/personne/jour)"]/1000)*365
anim["Disponibilite alimentaire (Kcal/personne/an)"] = anim["Disponibilite alimentaire (Kcal/personne/jour)"]*365
#selection des colonnes d'interet
anim1 = pd.DataFrame(data = anim, columns = ["Zone", "Disponibilite alimentaire (Kcal/personne/an)",
                                             "Disponibilite alimentaire en quantite (kg/personne/an)",
                                             "Disponibilite de proteines animale en quantite (kg/personne/an)",
                                             "Disponibilite interieure (kg/an)"])
#merge des donnée animaux et vegetale
anim2 = anim1.groupby(["Zone"]).sum().reset_index()
alim = pd.merge(anim2, veg3, on = "Zone", how = "left")
#calcule des proteines totaux et du % de proteine animale
alim["Proteine totale"] = alim["Disponibilite de proteines animale en quantite (kg/personne/an)"]+alim["Disponibilite de proteines vegetale en quantite (kg/personne/an)"]
alim["%Proteine animale"] = (alim["Disponibilite de proteines animale en quantite (kg/personne/an)"] * 100)/alim["Proteine totale"]
alim.head()
```

Out[5]:

	Zone	Disponibilite alimentaire (Kcal/personne/an)	Disponibilite alimentaire en quantite (kg/personne/an)	Disponibilite de proteines animale en quantite (kg/personne/an)	Disponibilite interieure (kg/an)	Disponibilite de proteines vegetale en quantite (kg/personne/an)	Proteine totale	%Proteine animale
0	Afghanistan	78840.0	79.92	4.45665	2.631000e+09	46.05	50.50665	8.823888
1	Afrique du Sud	177755.0	139.63	13.28235	7.817000e+09	48.94	62.22235	21.346590
2	Albanie	359525.0	389.05	21.68465	1.432000e+09	51.96	73.64465	29.444977
3	Algerie	137970.0	176.78	9.11770	7.120000e+09	66.94	76.05770	11.987872
4	Allemagne	380330.0	388.70	22.44385	3.540900e+10	39.90	62.34385	36.000103

In [6]:
```
#preparation des données population
population2013.head()
```

Out[6]:

	Code Domaine	Domaine	Code zone	Zone	Code element	element	Code Produit	Produit	Code annee	Annee	Unite	Valeur	Symbole	Description du Symbole
0	FBSH	Bilans Alimentaire (Ancienne methodologie et p...	2	Afghanistan	511	Population totale	2501	Population	2013	2013	1000 personnes	30552	NaN	Donnee officielle
1	FBSH	Bilans Alimentaire (Ancienne methodologie et p...	202	Afrique du Sud	511	Population totale	2501	Population	2013	2013	1000 personnes	52776	NaN	Donnee officielle
2	FBSH	Bilans Alimentaire (Ancienne methodologie et p...	3	Albanie	511	Population totale	2501	Population	2013	2013	1000 personnes	3173	NaN	Donnee officielle
3	FBSH	Bilans Alimentaire (Ancienne methodologie et p...	4	Algerie	511	Population totale	2501	Population	2013	2013	1000 personnes	39208	NaN	Donnee officielle
4	FBSH	Bilans Alimentaire (Ancienne methodologie et p...	79	Allemagne	511	Population totale	2501	Population	2013	2013	1000 personnes	82727	NaN	Donnee officielle

In [7]:
```
population2014.head()
```

Out[7]:

	Code Domaine	Domaine	Code zone (FAO)	Zone	Code element	element	Code Produit	Produit	Code annee	Annee	Unite	Valeur	Symbole	Description du Symbole	Note
0	OA	Series temporelles annuelles	2	Afghanistan	511	Population totale	3010	Population-Estimations	2014	2014	1000 personnes	33370.794	X	Sources internationales series	NaN
1	OA	Series temporelles annuelles	202	Afrique du Sud	511	Population totale	3010	Population-Estimations	2014	2014	1000 personnes	54544.186	X	Sources internationales series	NaN
2	OA	Series temporelles annuelles	3	Albanie	511	Population totale	3010	Population-Estimations	2014	2014	1000 personnes	2896.305	X	Sources internationales series	NaN
3	OA	Series temporelles annuelles	4	Algerie	511	Population totale	3010	Population-Estimations	2014	2014	1000 personnes	38923.692	X	Sources internationales series	NaN
4	OA	Series temporelles annuelles	79	Allemagne	511	Population totale	3010	Population-Estimations	2014	2014	1000 personnes	81450.378	X	Sources internationales series	NaN

In [8]:
```
#Traitement de la table population

#Selection des variables pertinents pour la table population

population13 = population2013[["Zone", "Valeur" ]]
population14 = population2014[["Zone", "Valeur" ]]
#renommer certains colonnes

population13.rename(columns= {"Valeur":"Population2013"}, inplace=True)
population14.rename(columns= {"Valeur":"Population2014"}, inplace=True)
#Conversion de la population a 1'unité habitants

population13["Population2013"] *= 1000
population14["Population2014"] *= 1000
#la Chine est représentée deux fois donc je supprime la 'Chine totale' et je garde la Chine representé par les
#la chine continentale, Macao et hong Kong
population13.drop(population13.index[population13['Zone'] == "Chine"], inplace = True)
population14.drop(population14.index[population14['Zone'] == "Chine"], inplace = True)
population13.head()
```

```
/opt/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py:4296: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().rename(
<ipython-input-8-db05fdee966>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  population13["Population2013"] *= 1000
<ipython-input-8-db05fdee966>:14: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  population14["Population2014"] *= 1000
/opt/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py:4163: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().drop(
```

Out[8]:

	Zone	Population2013
0	Afghanistan	30552000
1	Afrique du Sud	52776000
2	Albanie	3173000
3	Algerie	39208000
4	Allemagne	82727000

In [ ]:
```

```

In [9]:
```
population14.head()
```

Out[9]:

	Zone	Population2014
0	Afghanistan	33370794.0
1	Afrique du Sud	54544186.0
2	Albanie	2896305.0
3	Algerie	38923692.0
4	Allemagne	81450378.0

In [10]:
```
#merge de deux années de population afin de faire le calcule de l'evolution
population = pd.merge(population13, population14, on = "Zone", how = "left")
population.head()
```

Out[10]:

	Zone	Population2013	Population2014
0	Afghanistan	30552000	33370794.0
1	Afrique du Sud	52776000	54544186.0
2	Albanie	3173000	2896305.0
3	Algerie	39208000	38923692.0
4	Allemagne	82727000	81450378.0

In [11]:
```
#Calcule de l'evolution de la population de 2013 a 2014 en %
population["% Evolution13_14"] = -(population["Population2014"] *100)/population["Population2013"] - 100
population1 = pd.DataFrame(data=population, columns = ["Zone", "% Evolution13_14"])
population1.head()
```

Out[11]:

	Zone	% Evolution13_14
0	Afghanistan	9.226218
1	Afrique du Sud	3.350360
2	Albanie	-8.720296
3	Algerie	-0.725128
4	Allemagne	-1.543175

In [12]:
```
#obtention du tableau avec les donnée animale, vegetale et de la evoltion de la population
df = pd.merge(alim, population1, on = "Zone", how = 'left')
df.fillna(value='0', inplace=True)
df.head()
```

Out[12]:

	Zone	Disponibilite alimentaire (Kcal/personne/an)	Disponibilite alimentaire en quantite (kg/personne/an)	Disponibilite de proteines animale en quantite (kg/personne/an)	Disponibilite interieure (kg/an)	Disponibilite de proteines vegetale en quantite (kg/personne/an)	Proteine totale	%Proteine animale	% Evolution13_14
0	Afghanistan	78840.0	79.92	4.45665	2.631000e+09	46.05	50.50665	8.823888	9.22622
1	Afrique du Sud	177755.0	139.63	13.28235	7.817000e+09	48.94	62.22235	21.346590	3.35036
2	Albanie	359525.0	389.05	21.68465	1.432000e+09	51.96	73.64465	29.444977	-8.7203
3	Algerie	137970.0	176.78	9.11770	7.120000e+09	66.94	76.05770	11.987872	-0.725128
4	Allemagne	380330.0	388.70	22.44385	3.540900e+10	39.90	62.34385	36.000103	-1.54317

In [13]:
```
#obtention du tableau definitif avec les données animale, vegetale, l'evoltion de la population et du PIB
df1 = pd.merge(df, pib1, on = "Zone", how = 'left')
df1.fillna(value='0', inplace=True)
df1.head()
```

Out[13]:

	Zone	Disponibilite alimentaire (Kcal/personne/an)	Disponibilite alimentaire en quantite (kg/personne/an)	Disponibilite de proteines animale en quantite (kg/personne/an)	Disponibilite interieure (kg/an)	Disponibilite de proteines vegetale en quantite (kg/personne/an)	Proteine totale	%Proteine animale	% Evolution13_14	PIB
0	Afghanistan	78840.0	79.92	4.45665	2.631000e+09	46.05	50.50665	8.823888	9.22622	1.962180e+10
1	Afrique du Sud	177755.0	139.63	13.28235	7.817000e+09	48.94	62.22235	21.346590	3.35036	3.666454e+11
2	Albanie	359525.0	389.05	21.68465	1.432000e+09	51.96	73.64465	29.444977	-8.7203	1.277626e+10
3	Algerie	137970.0	176.78	9.11770	7.120000e+09	66.94	76.05770	11.987872	-0.725128	2.097550e+11
4	Allemagne	380330.0	388.70	22.44385	3.540900e+10	39.90	62.34385	36.000103	-1.54317	3.732743e+12

In [14]:
```
df1.dtypes
```

Out[14]:
```
Zone                                                 object
Disponibilite alimentaire (Kcal/personne/an)         float64
Disponibilite alimentaire en quantite (kg/personne/an)    float64
Disponibilite de proteines animale en quantite (kg/personne/an)    float64
Disponibilite interieure (kg/an)                     float64
Disponibilite de proteines vegetale en quantite (kg/personne/an)    float64
Proteine totale                                      float64
%Proteine animale                                    float64
% Evolution13_14                                     object
PIB                                                  object
dtype: object
```

In [15]:
```
df1[["% Evolution13_14", "PIB"]] = df1[["% Evolution13_14", "PIB"]].apply(pd.to_numeric)
```

In [16]:
```
isna = df.isna().any()
isnul = ~ df.isnull().any()
dup = df.duplicated().any()
isna, isnull, dup
```

Out[16]:
```
(Zone                                                 False
 Disponibilite alimentaire (Kcal/personne/an)         False
 Disponibilite alimentaire en quantite (kg/personne/an)    False
 Disponibilite de proteines animale en quantite (kg/personne/an)    False
 Disponibilite interieure (kg/an)                     False
 Disponibilite de proteines vegetale en quantite (kg/personne/an)    False
 Proteine totale                                      False
 %Proteine animale                                    False
 % Evolution13_14                                     False
 dtype: bool,
 Zone                                                 False
 Disponibilite alimentaire (Kcal/personne/an)         False
 Disponibilite alimentaire en quantite (kg/personne/an)    False
 Disponibilite de proteines animale en quantite (kg/personne/an)    False
 Disponibilite interieure (kg/an)                     False
 Disponibilite de proteines vegetale en quantite (kg/personne/an)    False
 Proteine totale                                      False
 %Proteine animale                                    False
 % Evolution13_14                                     False
 dtype: bool,
 False)
```

In [65]:
```
#l'exporte le fichier df1
df1.to_csv('df1.csv', index=False)
```

In [68]:
```
df1.head()
```

Out[68]:

	Zone	Disponibilite alimentaire (Kcal/personne/an)	Disponibilite alimentaire en quantite (kg/personne/an)	Disponibilite de proteines animale en quantite (kg/personne/an)	Disponibilite interieure (kg/an)	Disponibilite de proteines vegetale en quantite (kg/personne/an)	Proteine totale	%Proteine animale	% Evolution13_14	PIB
0	Afghanistan	78840.0	79.92	4.45665	2.631000e+09	46.05	50.50665	8.823888	9.22622	1.962180e+10
1	Afrique du Sud	177755.0	139.63	13.28235	7.817000e+09	48.94	62.22235	21.346590	3.35036	3.666454e+11
2	Albanie	359525.0	389.05	21.68465	1.432000e+09	51.96	73.64465	29.444977	-8.7203	1.277626e+10
3	Algerie	137970.0	176.78	9.11770	7.120000e+09	66.94	76.05770	11.987872	-0.725128	2.097550e+11
4	Allemagne	380330.0	388.70	22.44385	3.540900e+10	39.90	62.34385	36.000103	-1.54317	3.732743e+12

In [69]:
```
df1.to_csv('df2.csv', index=False)
```

In [ ]:
```

```