

# PROJET 4 DATA ANALYST

Réalisez une étude de santé publique avec R ou Python

## OBJECTIF DE CE NOTEBOOK

Bienvenue dans l'outil plébiscité par les analystes de données Jupyter.

Il s'agit d'un outil permettant de mixer et d'alterner codes, textes et graphique.

Cet outil est formidable pour plusieurs raisons:

- il permet de tester des lignes de codes au fur et à mesure de votre rédaction, de constater immédiatement le résultat d'une instruction, de la corriger si nécessaire.
- De rédiger du texte pour expliquer l'approche suivie ou les résultats d'une analyse et de le mettre en forme grâce à du code html ou plus simple avec **Markdown**
- d'agrémenter de graphiques

## Etape 1 - Importation des librairies et chargement des fichiers

### 1.1 - Importation des librairies

- Importation de la librairie Pandas

Entrée [294]:

```
import pandas as pd
```

- Importation de la librairie NumPy

Entrée [295]:

```
import numpy as np
```

- Importation de la librairie Matplotlib.pyplot

Entrée [296]:

```
import matplotlib.pyplot as plt
```

## 1.2 - Chargement des fichiers Excel

- Importation du fichier population.csv

Entrée [297]:

```
population = pd.read_csv('population.csv')
```

Entrée [298]:

```
type(population)
```

Out[298]:

```
pandas.core.frame.DataFrame
```

- Importation du fichier dispo\_alimentaire.csv

Entrée [299]:

```
dispo_alimentaire = pd.read_csv('dispo_alimentaire.csv')
```

Entrée [300]:

```
type(dispo_alimentaire)
```

Out[300]:

```
pandas.core.frame.DataFrame
```

- Importation du fichier aide\_alimentaire.csv

Entrée [301]:

```
aide_alimentaire = pd.read_csv('aide_alimentaire.csv')
```

Entrée [302]:

```
type(aide_alimentaire)
```

Out[302]:

```
pandas.core.frame.DataFrame
```

- Importation du fichier sous\_nutrition.csv

Entrée [303]:

```
sous_nutrition = pd.read_csv('sous_nutrition.csv')
```

Entrée [304]:

```
type(sous_nutrition)
```

Out[304]:

```
pandas.core.frame.DataFrame
```

## Etape 2 - Analyse exploratoire des fichiers

### 2.1 - Analyse exploratoire du fichier population

- Afficher les dimensions du dataset

Entrée [305]:

```
print("Le tableau population comporte {} observation(s) ou article(s)".format(population.shape[0]))  
print("Le tableau population comporte {} colonne(s)".format(population.shape[1]))
```

Le tableau population comporte 1416 observation(s) ou article(s)

Le tableau population comporte 3 colonne(s)

Entrée [306]:

```
population.shape
```

Out[306]:

```
(1416, 3)
```

- Consulter le nombre de colonnes

Entrée [307]:

```
population.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1416 entries, 0 to 1415  
Data columns (total 3 columns):  
 #   Column      Non-Null Count  Dtype    
---  -  
 0   Zone        1416 non-null   object   
 1   Année       1416 non-null   int64    
 2   Valeur      1416 non-null   float64  
dtypes: float64(1), int64(1), object(1)  
memory usage: 33.3+ KB
```

- La nature des données dans chacune des colonnes

Entrée [308]:

```
population.dtypes
```

Out[308]:

```
Zone      object
Année      int64
Valeur    float64
dtype: object
```

- Le nombre de valeurs présentes dans chacune des colonnes

Entrée [309]:

```
population.count()
```

Out[309]:

```
Zone      1416
Année      1416
Valeur     1416
dtype: int64
```

Entrée [310]:

```
population.nunique()
```

Out[310]:

```
Zone      236
Année       6
Valeur    1413
dtype: int64
```

Entrée [ ]:

- Affichage les 5 premières lignes de la table

Entrée [311]:

```
population.head()
```

Out[311]:

	Zone	Année	Valeur
0	Afghanistan	2013	32269.589
1	Afghanistan	2014	33370.794
2	Afghanistan	2015	34413.603
3	Afghanistan	2016	35383.032
4	Afghanistan	2017	36296.113

- Nous allons harmoniser les unités. Pour cela, nous avons décidé de multiplier la population par 1000
- Multiplication de la colonne valeur par 1000

Entrée [312]:

```
population['Valeur']=population['Valeur']*1000
```

- changement du nom de la colonne Valeur par Population

Entrée [313]:

```
population.rename(columns={"Valeur": "Pop"}, inplace=True)
```

- Affichage les 5 premières lignes de la table pour voir les modifications

Entrée [314]:

```
population.head()
```

Out[314]:

	Zone	Année	Pop
0	Afghanistan	2013	32269589.0
1	Afghanistan	2014	33370794.0
2	Afghanistan	2015	34413603.0
3	Afghanistan	2016	35383032.0
4	Afghanistan	2017	36296113.0

## 2.2 - Analyse exploratoire du fichier disponibilité alimentaire

- Afficher les dimensions du dataset

Entrée [315]:

```
print("Le tableau dispo_alimentaire comporte {} observation(s) ou article(s)".format(dispo_alimentaire.shape[0]))
print("Le tableau dispo_alimentaire comporte {} colonne(s)".format(dispo_alimentaire.shape[1]))
```

Le tableau dispo\_alimentaire comporte 15605 observation(s) ou article(s)  
 Le tableau dispo\_alimentaire comporte 18 colonne(s)

- Consulter le nombre de colonnes

Entrée [316]:

```
dispo_alimentaire.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15605 entries, 0 to 15604
Data columns (total 18 columns):
#   Column                                     Non-Nu
ll Count  Dtype
---  -
-----
0   Zone                                     15605
non-null object
1   Produit                                15605
non-null object
2   Origine                                15605
non-null object
3   Aliments pour animaux                 2720 n
on-null float64
4   Autres Utilisations                   5496 n
on-null float64
5   Disponibilité alimentaire (Kcal/personne/jour) 14241
non-null float64
6   Disponibilité alimentaire en quantité (kg/personne/an) 14015
non-null float64
7   Disponibilité de matière grasse en quantité (g/personne/jour) 11794
non-null float64
8   Disponibilité de protéines en quantité (g/personne/jour) 11561
non-null float64
9   Disponibilité intérieure                15382
non-null float64
10  Exportations - Quantité                12226
non-null float64
11  Importations - Quantité                14852
non-null float64
12  Nourriture                             14015
non-null float64
13  Pertes                                 4278 n
on-null float64
14  Production                             9180 n
on-null float64
15  Semences                              2091 n
on-null float64
16  Traitement                             2292 n
on-null float64
17  Variation de stock                     6776 n
on-null float64
dtypes: float64(15), object(3)
memory usage: 2.1+ MB
```

- Affichage les 5 premières lignes de la table

Entrée [317]:

```
dispo_alimentaire.head()
```

Out[317]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire (kg/personne/an)
0	Afghanistan	Abats Comestible	animale	NaN	NaN	5.0	1.0
1	Afghanistan	Agrumes, Autres	vegetale	NaN	NaN	1.0	1.0
2	Afghanistan	Aliments pour enfants	vegetale	NaN	NaN	1.0	0.0
3	Afghanistan	Ananas	vegetale	NaN	NaN	0.0	0.0
4	Afghanistan	Bananes	vegetale	NaN	NaN	4.0	2.0

- Remplacement des NaN dans le dataset par des 0

Entrée [318]:

```
dispo_alimentaire.fillna(0, inplace=True)
```

Entrée [319]:

```
dispo_alimentaire.head()
```

Out[319]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire (kg/personne/an)
0	Afghanistan	Abats Comestible	animale	0.0	0.0	5.0	1.0
1	Afghanistan	Agrumes, Autres	vegetale	0.0	0.0	1.0	1.0
2	Afghanistan	Aliments pour enfants	vegetale	0.0	0.0	1.0	0.0
3	Afghanistan	Ananas	vegetale	0.0	0.0	0.0	0.0
4	Afghanistan	Bananes	vegetale	0.0	0.0	4.0	2.0

- Multiplication de toutes les lignes contenant des milliers de tonnes en Kg

Entrée [320]:

```
conversion_tonnes_kg = ['Aliments pour animaux', 'Disponibilité intérieure', 'Exportation',  
                        'Importations - Quantité', 'Nourriture', 'Pertes', 'Production',  
                        'Semences', 'Traitement', 'Variation de stock', 'Autres Utilisati  
  
for x in conversion_tonnes_kg:  
    dispo_alimentaire[x] *= 1000000
```

- Affichage les 5 premières lignes de la table

Entrée [321]:

```
dispo_alimentaire.head()
```

Out[321]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibili alimentaire (kg/personne/a
0	Afghanistan	Abats Comestible	animale	0.0	0.0	5.0	1.0
1	Afghanistan	Agrumes, Autres	vegetale	0.0	0.0	1.0	1.0
2	Afghanistan	Aliments pour enfants	vegetale	0.0	0.0	1.0	0.0
3	Afghanistan	Ananas	vegetale	0.0	0.0	0.0	0.0
4	Afghanistan	Bananes	vegetale	0.0	0.0	4.0	2.0

## 2.3 - Analyse exploratoire du fichier aide alimentaire

- Afficher les dimensions du dataset

Entrée [322]:

```
print("Le tableau aide_alimentaire comporte {} observation(s) ou article(s)".format(aide_  
print("Le tableau aide_alimentaire comporte {} colonne(s)".format(aide_alimentaire.shape[
```

Le tableau aide\_alimentaire comporte 1475 observation(s) ou article(s)  
Le tableau aide\_alimentaire comporte 4 colonne(s)

- Consulter le nombre de colonnes



Entrée [323]:

```
aide_alimentaire.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1475 entries, 0 to 1474
Data columns (total 4 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Pays bénéficiaire     1475 non-null   object
1   Année                 1475 non-null   int64
2   Produit               1475 non-null   object
3   Valeur                1475 non-null   int64
dtypes: int64(2), object(2)
memory usage: 46.2+ KB
```

- Affichage les 5 premières lignes de la table

Entrée [324]:

```
aide_alimentaire.head()
```

Out[324]:

	Pays bénéficiaire	Année	Produit	Valeur
0	Afghanistan	2013	Autres non-céréales	682
1	Afghanistan	2014	Autres non-céréales	335
2	Afghanistan	2013	Blé et Farin	39224
3	Afghanistan	2014	Blé et Farin	15160
4	Afghanistan	2013	Céréales	40504

- Changement du nom de la colonne Pays bénéficiaire par Zone et Valeur par quantiteReçu

Entrée [325]:

```
aide_alimentaire.rename(columns={"Pays bénéficiaire": "Zone", "Valeur": "quantiteReçu"}, inplace=True)
```

- Multiplication de la colonne Aide\_alimentaire qui contient des tonnes par 1000 pour avoir des kg

Entrée [326]:

```
aide_alimentaire['quantiteReçu'] = aide_alimentaire['quantiteReçu'] * 1000
```

- Affichage les 5 premières lignes de la table

Entrée [327]:

```
aide_alimentaire.head()
```

Out[327]:

	Zone	Année	Produit	quantiteReçu
0	Afghanistan	2013	Autres non-céréales	682000
1	Afghanistan	2014	Autres non-céréales	335000
2	Afghanistan	2013	Blé et Farin	39224000
3	Afghanistan	2014	Blé et Farin	15160000
4	Afghanistan	2013	Céréales	40504000

Entrée [328]:

```
aide_alimentaire.dtypes
```

Out[328]:

```
Zone          object
Année         int64
Produit       object
quantiteReçu  int64
dtype: object
```

## 2.3 - Analyse exploratoire du fichier sous nutrition

- Afficher les dimensions du dataset

Entrée [329]:

```
print("Le tableau sous_nutrition comporte {} observation(s) ou article(s)".format(sous_nu
print("Le tableau sous_nutrition comporte {} colonne(s)".format(sous_nutrition.shape[1]))
```

```
Le tableau sous_nutrition comporte 1218 observation(s) ou article(s)
Le tableau sous_nutrition comporte 3 colonne(s)
```

- Consulter le nombre de colonnes

Entrée [330]:

```
sous_nutrition.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1218 entries, 0 to 1217
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  -
 0   Zone      1218 non-null   object
 1   Année     1218 non-null   object
 2   Valeur    624 non-null    object
dtypes: object(3)
memory usage: 28.7+ KB
```

- Afficher les 5 premières lignes de la table

Entrée [331]:

```
sous_nutrition.head()
```

Out[331]:

	Zone	Année	Valeur
0	Afghanistan	2012-2014	8.6
1	Afghanistan	2013-2015	8.8
2	Afghanistan	2014-2016	8.9
3	Afghanistan	2015-2017	9.7
4	Afghanistan	2016-2018	10.5

- Vérification des types de nos variables

Entrée [332]:

```
sous_nutrition.dtypes
```

Out[332]:

```
Zone      object
Année     object
Valeur    object
dtype: object
```

- Conversion de la colonne sous nutrition en numérique avec l'argument errors=coerce qui permet de convertir automatiquement les lignes qui ne sont pas des nombres en NaN)

Entrée [333]:

```
sous_nutrition['Valeur']=pd.to_numeric(sous_nutrition['Valeur'], errors='coerce')
```

- Remplacement des NaN en 0

Entrée [334]:

```
sous_nutrition['Valeur'] = sous_nutrition['Valeur'].fillna(0)
```

- Changement du nom de la colonne Valeur par sous\_nutrition

Entrée [335]:

```
sous_nutrition.rename(columns={"Valeur":"popSNut"},inplace=True)
```

- Multiplication de la colonne sous\_nutrition par 1000000

Entrée [336]:

```
sous_nutrition['popSNut']=sous_nutrition['popSNut']*1000000
```

- Afficher les 5 premières lignes de la table

Entrée [337]:

```
sous_nutrition.head()
```

Out[337]:

	Zone	Année	popSNut
0	Afghanistan	2012-2014	8600000.0
1	Afghanistan	2013-2015	8800000.0
2	Afghanistan	2014-2016	8900000.0
3	Afghanistan	2015-2017	9700000.0
4	Afghanistan	2016-2018	10500000.0

- Remplacement des intervalles d'année en leur moyenne. Exemple la valeur '2012-2014' est remplacée par sa moyenne : '2013'

Entrée [338]:

```
sous_nutrition['Année'] = sous_nutrition['Année'].str.replace('2012-2014', '2013')
sous_nutrition['Année'] = sous_nutrition['Année'].str.replace('2013-2015', '2014')
sous_nutrition['Année'] = sous_nutrition['Année'].str.replace('2014-2016', '2015')
sous_nutrition['Année'] = sous_nutrition['Année'].str.replace('2015-2017', '2016')
sous_nutrition['Année'] = sous_nutrition['Année'].str.replace('2016-2018', '2017')
sous_nutrition['Année'] = sous_nutrition['Année'].str.replace('2017-2019', '2018')
```

- Afficher les 5 premières lignes de la table

Entrée [339]:

```
sous_nutrition.head()
```

Out[339]:

	Zone	Année	popSNut
0	Afghanistan	2013	8600000.0
1	Afghanistan	2014	8800000.0
2	Afghanistan	2015	8900000.0
3	Afghanistan	2016	9700000.0
4	Afghanistan	2017	10500000.0

## Etape 3 - Analyses réalisées

### 3.1 - Proportion de personnes en sous nutrition

- Changement du type de la serie Année dans le dataframe sous\_nutrition

Entrée [340]:

```
sous_nutrition['Année'] = sous_nutrition['Année'].astype(np.int64)  
sous_nutrition.dtypes
```

Out[340]:

```
Zone          object  
Année          int64  
popSNut      float64  
dtype: object
```

- Il faut tout d'abord faire une jointure entre la table population et la table sous nutrition, en ciblant l'année 2017

Entrée [341]:

```
df_popSN17 = pd.merge(population.loc[population['Année'] == 2017], sous_nutrition.loc[sou
```

- Affichage du dataset

Entrée [435]:

```
df_popSN17
```

Out[435]:

	Zone	Année	Pop	popSNut	Proportion_par_pays
0	Afghanistan	2017	36296113.0	10500000.0	28.93
1	Afrique du Sud	2017	57009756.0	3100000.0	5.44
2	Albanie	2017	2884169.0	100000.0	3.47
3	Algérie	2017	41389189.0	1300000.0	3.14
4	Allemagne	2017	82658409.0	0.0	0.00
...	...	...	...	...	...
198	Venezuela (République bolivarienne du)	2017	29402484.0	8000000.0	27.21
199	Viet Nam	2017	94600648.0	6500000.0	6.87
200	Yémen	2017	27834819.0	0.0	0.00
201	Zambie	2017	16853599.0	0.0	0.00
202	Zimbabwe	2017	14236595.0	0.0	0.00

203 rows × 5 columns

Entrée [343]:

```
print("La population mondiale en 2017 est de: {:.0f} personnes".format(df_popSN17['Pop']).
```

La population mondiale en 2017 est de: 7543798779 personnes

Entrée [344]:

```
print("La population totale en état de sous nutrition en 2017 est de: {:.0f} personnes".f
```

La population totale en état de sous nutrition en 2017 est de: 535700000 p  
ersonnes

- Calcul et affichage du nombre de personnes en état de sous nutrition

Entrée [345]:

```
print("La population mondiale en 2017 est de: {:.0f} personnes".format(df_popSN17['Pop']).
```

La population mondiale en 2017 est de: 7543798779 personnes

Entrée [346]:

```
print("La population totale en état de sous nutrition en 2017 est de: {:.0f} personnes".f
```

La population totale en état de sous nutrition en 2017 est de: 535700000 p  
ersonnes

Entrée [347]:

```
print("La proportion des personnes en état de sous-nutrition est de: {:.2f}%".format(df_p
```

La proportion des personnes en état de sous-nutrition est de: 7.10%

### 3.2 - Nombre théorique de personne qui pourrait être nourries

- Combien mange en moyenne un être humain ? Source =>

Un être humain a besoin en moyenne de 2500 calories par jour selon l'Organisation mondiale de la santé (OMS). Une estimation générale basée sur les besoins énergétiques moyens d'un homme adulte.

- On commence par faire une jointure entre le data frame population et Dispo\_alimentaire afin d'ajouter dans ce dernier la population

Entrée [348]:

```
dispo_alimentaire
```

Out[348]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Dispo aliment q (kg/persor
0	Afghanistan	Abats Comestible	animale	0.0	0.0	5.0	
1	Afghanistan	Agrumes, Autres	vegetale	0.0	0.0	1.0	
2	Afghanistan	Aliments pour enfants	vegetale	0.0	0.0	1.0	
3	Afghanistan	Ananas	vegetale	0.0	0.0	0.0	
4	Afghanistan	Bananes	vegetale	0.0	0.0	4.0	
...	...	...	...	...	...	...	
15600	Îles Salomon	Viande de Suides	animale	0.0	0.0	45.0	
15601	Îles Salomon	Viande de Volailles	animale	0.0	0.0	11.0	
15602	Îles Salomon	Viande, Autre	animale	0.0	0.0	0.0	
15603	Îles Salomon	Vin	vegetale	0.0	0.0	0.0	
15604	Îles Salomon	Épices, Autres	vegetale	0.0	0.0	4.0	

15605 rows × 18 columns



Entrée [349]:

```
df_popDA17 = pd.merge(population.loc[population['Année'] == 2017], dispo_alimentaire, on=
```

- Affichage du nouveau dataframe



Entrée [436]:

```
df_popDA17
```

Out[436]:

	Zone	Année	Pop	Produit	Origine	Aliments pour animaux	Autres Utilisations	Dis al (Kcal/persc
0	Afghanistan	2017	36296113.0	Abats Comestible	animale	0.0	0.0	
1	Afghanistan	2017	36296113.0	Agrumes, Autres	vegetale	0.0	0.0	
2	Afghanistan	2017	36296113.0	Aliments pour enfants	vegetale	0.0	0.0	
3	Afghanistan	2017	36296113.0	Ananas	vegetale	0.0	0.0	
4	Afghanistan	2017	36296113.0	Bananes	vegetale	0.0	0.0	
...	...	...	...	...	...	...	...	
15411	Zimbabwe	2017	14236595.0	Viande de Suides	animale	0.0	0.0	
15412	Zimbabwe	2017	14236595.0	Viande de Volailles	animale	0.0	0.0	
15413	Zimbabwe	2017	14236595.0	Viande, Autre	animale	0.0	1000000.0	
15414	Zimbabwe	2017	14236595.0	Vin	vegetale	0.0	0.0	
15415	Zimbabwe	2017	14236595.0	Épices, Autres	vegetale	0.0	0.0	

15416 rows × 21 columns



- Création de la colonne dispo\_kcal avec calcul des kcal disponibles mondialement

Entrée [351]:

```
pd.options.mode.chained_assignment = None # default='warn'
```

Entrée [352]:

```
df_popDA17['Dispo_kcal'] = df_popDA17['Disponibilité alimentaire (Kcal/personne/jour)'] *
```

Entrée [353]:

```
df_popDA17.head()
```

Out[353]:

	Zone	Année	Pop	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponil alimen (Kcal/personne/
0	Afghanistan	2017	36296113.0	Abats Comestible	animale	0.0	0.0	
1	Afghanistan	2017	36296113.0	Agrumes, Autres	vegetale	0.0	0.0	
2	Afghanistan	2017	36296113.0	Aliments pour enfants	vegetale	0.0	0.0	
3	Afghanistan	2017	36296113.0	Ananas	vegetale	0.0	0.0	
4	Afghanistan	2017	36296113.0	Bananes	vegetale	0.0	0.0	

5 rows × 21 columns

Entrée [354]:

```
print("La disponibilité alimentaire en kcal mondiale est de: {:.1f} kcal".format(df_popDA
```

La disponibilité alimentaire en kcal mondiale est de: 7635429388975815.0 kcal

- Calcul du nombre d'humains pouvant être nourris

Entrée [355]:

```
Total_kcal=round(df_popDA17['Dispo_kcal'].sum()/(2500*365))  
print("Total d'humains pouvant etre nourris est de :", Total_kcal)
```

Total d'humains pouvant etre nourris est de : 8367593851

Entrée [356]:

```
print("La proportion d'humains pouvant etre nourris est de :", "{:.2f}".format(Total_kcal
```

La proportion d'humains pouvant etre nourris est de : 110.86 %

### 3.3 - Nombre théorique de personne qui pourrait être nourrie avec les produits végétaux

- Transfert des données avec les végétaux dans un nouveau dataframe

Entrée [357]:

```
df_popDA17V = df_popDA17.loc[(df_popDA17['Origine'] == 'vegetale')]
```

Entrée [358]:

```
df_popDA17V.head()
```

Out[358]:

	Zone	Année	Pop	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibi alimenta (Kcal/personne/jo
1	Afghanistan	2017	36296113.0	Agrumes, Autres	vegetale	0.0	0.0	
2	Afghanistan	2017	36296113.0	Aliments pour enfants	vegetale	0.0	0.0	
3	Afghanistan	2017	36296113.0	Ananas	vegetale	0.0	0.0	
4	Afghanistan	2017	36296113.0	Bananes	vegetale	0.0	0.0	
6	Afghanistan	2017	36296113.0	Bière	vegetale	0.0	0.0	

5 rows × 21 columns



- Calcul du nombre d'humains pouvant être nourris avec les végétaux

Entrée [359]:

```
df_popDA17V['Dispo_kcalV']=df_popDA17V['Disponibilité alimentaire (Kcal/personne/jour)']*  
print("La disponibilité alimentaire en kcal mondiale des produits végétaux est de: {:.1f}")
```



La disponibilité alimentaire en kcal mondiale des produits végétaux est de: 6300178937197865.0 kcal

Entrée [360]:

```
Total_kcalV=round(df_popDA17V['Dispo_kcalV'].sum()/(2500*365))  
print("Total d'humains pouvant etre nourris avec les végétaux est de:", Total_kcalV)
```

Total d'humains pouvant etre nourris avec les végétaux est de: 6904305685

Entrée [361]:

```
print("La proportion d'humains pouvant etre nourris avec les végétaux est de:", "{:.2f}%".format(
```



La proportion d'humains pouvant etre nourris avec les végétaux est de: 91.47 %

### 3.4 - Utilisation de la disponibilité intérieure

Disponibilité intérieure = Semences + Pertes + Nourriture + Aliments pour animaux + Traitement + Autres utilisations

Disponibilité intérieure = Production + Variation de stock + Importation - Exportation.

- Calcul de la disponibilité totale

Entrée [362]:

```
dispo_int_totale = df_popDA17['Disponibilité intérieure'].sum()  
print("La disponibilité intérieure totale est de:", dispo_int_totale)
```

La disponibilité intérieure totale est de: 9733927000000.0

- Création d'une boucle for pour afficher les différentes valeurs en fonction des colonnes aliments pour animaux, pertes, nourritures,

On va procéder par le calcul de la part en pourcentage de la disponibilité alimentaire des produits aliments pour animaux, pertes, nourritures par rapport à la disponibilité intérieure totale.

Entrée [363]:

```
colonnes=['Aliments pour animaux', 'Pertes', 'Nourriture']  
for x in colonnes:  
    print("Proportion de", x, ":", "{:.2f}".format(dispo_alimentaire[x].sum()*100/dispo_i
```

Proportion de Aliments pour animaux : 13.40 %

Proportion de Pertes : 4.66 %

Proportion de Nourriture : 50.10 %

- Création d'une boucle for pour afficher le pourcentage de la disponibilité alimentaire des produits aliments pour animaux, pertes, nourritures, semences, traitement, autres utilisations par rapport à la disponibilité intérieure totale.

Entrée [364]:

```
colonnes=['Aliments pour animaux', 'Pertes', 'Nourriture', 'Semences', 'Traitement', 'Autres']  
for x in colonnes:  
    print(" proportion de", x, ":", "{:.2f}".format(dispo_alimentaire[x].sum()*100/dispo_
```

proportion de Aliments pour animaux : 13.40 %

proportion de Pertes : 4.66 %

proportion de Nourriture : 50.10 %

proportion de Semences : 1.59 %

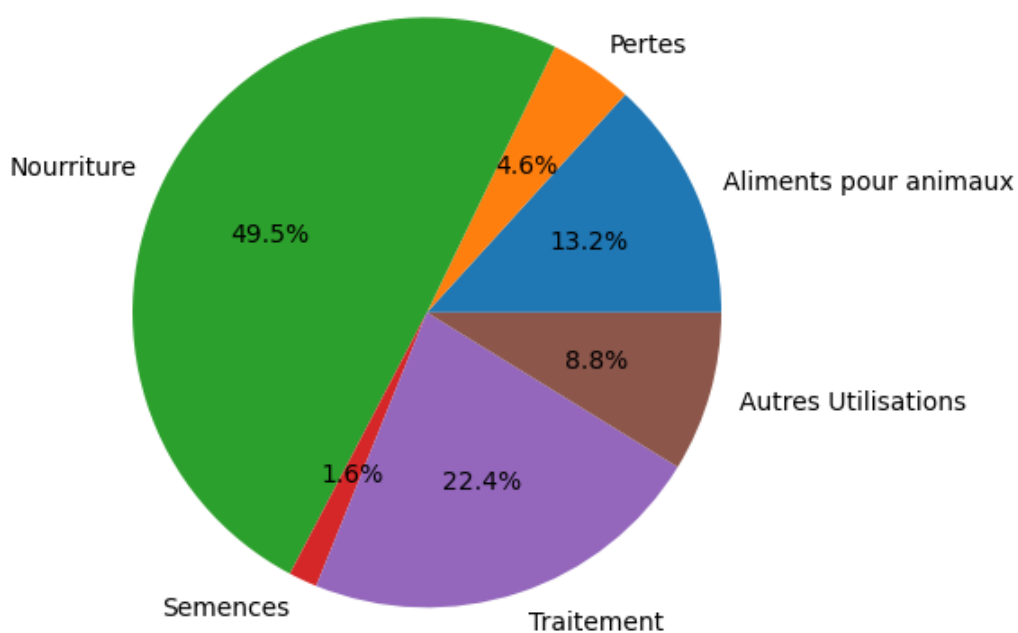
proportion de Traitement : 22.65 %

proportion de Autres Utilisations : 8.89 %

Entrée [365]:

```
colonnes=['Aliments pour animaux', 'Pertes', 'Nourriture','Semences','Traitement','Autres  
proportions = []  
for x in colonnes:  
    proportion = dispo_alimentaire[x].sum() * 100 / dispo_int_totale  
    proportions.append(proportion)  
  
plt.pie(proportions, labels=colonnes, autopct='%1.1f%%')  
plt.axis('equal')  
plt.title('Répartition de la disponibilité intérieure de produits alimentaires en %')  
  
plt.show()
```

Répartition de la disponibilité intérieure de produits alimentaires en %



### 3.5 - Utilisation des céréales

- Création d'une liste avec toutes les variables

Entrée [366]:

```
liste_cereales = ['Blé', 'Riz (Eq Blanchi)', 'Orge', 'Maïs', 'Seigle', 'Avoine', 'Millet', 'Sor
```

- Création d'un dataframe avec les informations uniquement pour ces céréales

Entrée [367]:

```
df_cereales=dispo_alimentaire.loc[dispo_alimentaire['Produit'].isin(liste_cereales),:]
```

Entrée [368]:

```
df_cereales.head(10)
```

Out[368]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Dispor alimenta qu
7	Afghanistan	Blé	vegetale	0.000000e+00	0.0	1369.0	
32	Afghanistan	Maïs	vegetale	2.000000e+08	0.0	21.0	
34	Afghanistan	Millet	vegetale	0.000000e+00	0.0	3.0	
40	Afghanistan	Orge	vegetale	3.600000e+08	0.0	26.0	
47	Afghanistan	Riz (Eq Blanchi)	vegetale	0.000000e+00	0.0	141.0	
67	Afrique du Sud	Avoine	vegetale	8.000000e+06	0.0	5.0	
72	Afrique du Sud	Blé	vegetale	3.700000e+07	0.0	492.0	
111	Afrique du Sud	Maïs	vegetale	4.715000e+09	0.0	858.0	
113	Afrique du Sud	Millet	vegetale	8.000000e+06	0.0	1.0	
121	Afrique du Sud	Orge	vegetale	1.900000e+07	0.0	1.0	

- Affichage de la proportion d'alimentation animale

Entrée [369]:

```
print("La proportion d'alimentation animale est de :", "{:.0f}".format(df_cereales['Alime
```

La proportion d'alimentation animale est de : 36 %

- Affichage de la proportion d'alimentation humaine

Entrée [370]:

```
print("La proportion d'alimentation humaine est de :", "{:.0f}".format(df_cereales['Nourr
```

La proportion d'alimentation humaine est de : 43 %

### 3.6 - Pays avec la proportion de personnes sous-alimentée la plus forte en 2017

- Création de la colonne proportion par pays

Entrée [371]:

```
df_popSN17.head()
```

Out[371]:

	Zone	Année	Pop	popSNut
0	Afghanistan	2017	36296113.0	10500000.0
1	Afrique du Sud	2017	57009756.0	3100000.0
2	Albanie	2017	2884169.0	100000.0
3	Algérie	2017	41389189.0	1300000.0
4	Allemagne	2017	82658409.0	0.0

Entrée [372]:

```
df_popSN17['Proportion_par_pays']=round(df_popSN17['popSNut']/df_popSN17['Pop']*100,2)  
df_popSN17.head()
```

Out[372]:

	Zone	Année	Pop	popSNut	Proportion_par_pays
0	Afghanistan	2017	36296113.0	10500000.0	28.93
1	Afrique du Sud	2017	57009756.0	3100000.0	5.44
2	Albanie	2017	2884169.0	100000.0	3.47
3	Algérie	2017	41389189.0	1300000.0	3.14
4	Allemagne	2017	82658409.0	0.0	0.00

- Affichage après trié des 10 pires pays

Entrée [373]:

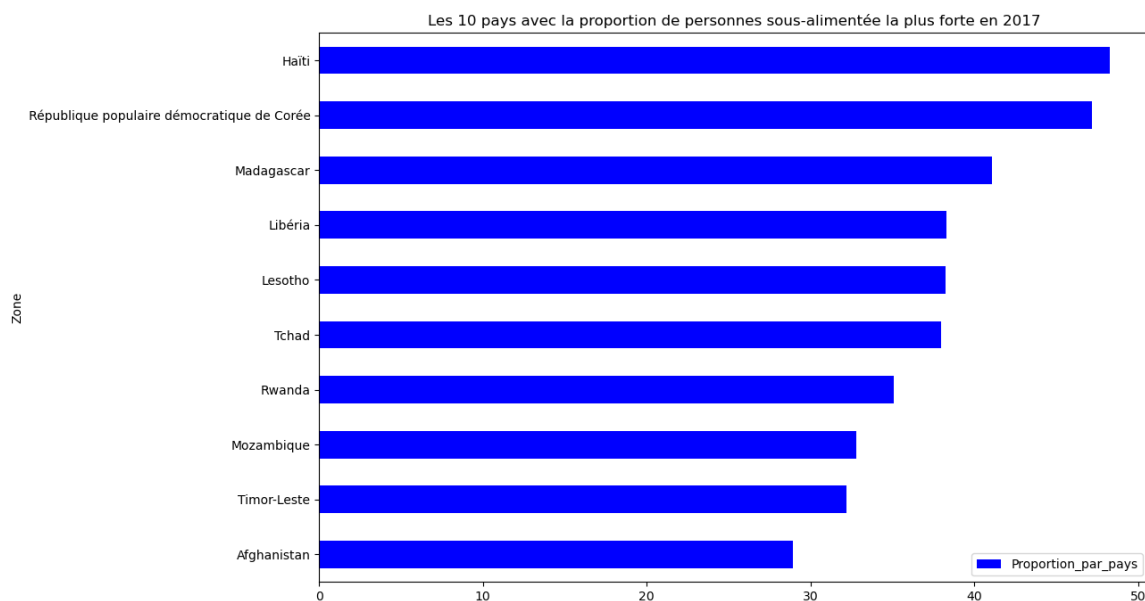
```
df_popSN17_max = df_popSN17.sort_values('Proportion_par_pays', ascending=False).head(10)
df_popSN17_max
```

Out[373]:

	Zone	Année	Pop	popSNut	Proportion_par_pays
78	Haïti	2017	10982366.0	5300000.0	48.26
157	République populaire démocratique de Corée	2017	25429825.0	12000000.0	47.19
108	Madagascar	2017	25570512.0	10500000.0	41.06
103	Libéria	2017	4702226.0	1800000.0	38.28
100	Lesotho	2017	2091534.0	800000.0	38.25
183	Tchad	2017	15016753.0	5700000.0	37.96
161	Rwanda	2017	11980961.0	4200000.0	35.06
121	Mozambique	2017	28649018.0	9400000.0	32.81
186	Timor-Leste	2017	1243258.0	400000.0	32.17
0	Afghanistan	2017	36296113.0	10500000.0	28.93

Entrée [374]:

```
df_popSN17_max.head(10).groupby("Zone")["Proportion_par_pays"].sum().sort_values().plot.bar(figsize=(12,8), legend=True, title='Les 10 pays avec la proportion de personnes sous-alime
```



### 3.7 - Pays qui ont le plus bénéficié d'aide alimentaire depuis 2013

- Calcul du total de l'aide alimentaire par pays



Entrée [375]:

```
aide_alimentaire.head()
```

Out[375]:

	Zone	Année	Produit	quantiteReçu
0	Afghanistan	2013	Autres non-céréales	682000
1	Afghanistan	2014	Autres non-céréales	335000
2	Afghanistan	2013	Blé et Farin	39224000
3	Afghanistan	2014	Blé et Farin	15160000
4	Afghanistan	2013	Céréales	40504000

Entrée [376]:

```
aide_alim_totale= aide_alimentaire[['Zone','quantiteReçu']].groupby("Zone").sum()  
aide_alim_totale.head()
```

Out[376]:

	quantiteReçu
Zone	
Afghanistan	185452000
Algérie	81114000
Angola	5014000
Bangladesh	348188000
Bhoutan	2666000

- Affichage après trie des 10 pays qui ont bénéficié le plus de l'aide alimentaire

Entrée [377]:

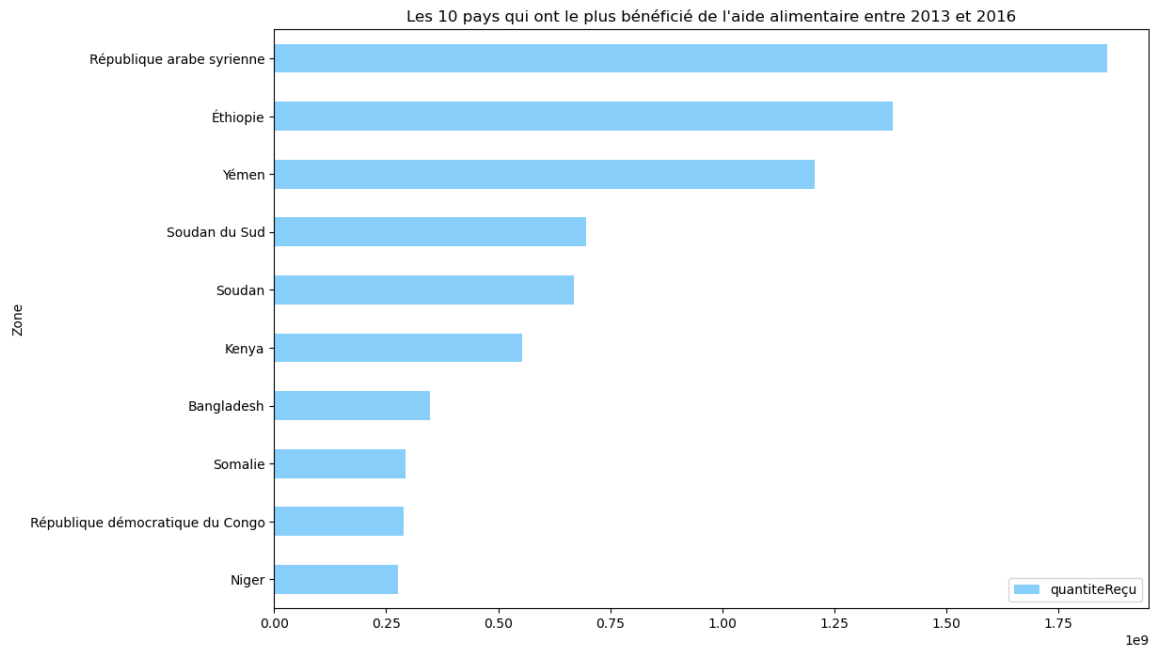
```
aide_alim_max = aide_alim_totale.sort_values('quantiteReçu',ascending=False)
aide_alim_max.head(10)
```

Out[377]:

quantiteReçu	
Zone	
République arabe syrienne	1858943000
Éthiopie	1381294000
Yémen	1206484000
Soudan du Sud	695248000
Soudan	669784000
Kenya	552836000
Bangladesh	348188000
Somalie	292678000
République démocratique du Congo	288502000
Niger	276344000

Entrée [378]:

```
aide_alim_max.head(10).groupby("Zone")["quantiteReçu"].sum().sort_values().plot.barh(
    color="lightskyblue", figsize=(12,8), legend=True,title="Les 10 pays qui ont le plus
```



### 3.8 - Evolution des 5 pays qui ont le plus bénéficiés de l'aide alimentaire entre 2013 et 2016

- Création d'un dataframe avec la zone, l'année et l'aide alimentaire puis groupby sur zone et année

Entrée [379]:

```
aide_alimentaire.head()
```

Out[379]:

	Zone	Année	Produit	quantiteReçu
0	Afghanistan	2013	Autres non-céréales	682000
1	Afghanistan	2014	Autres non-céréales	335000
2	Afghanistan	2013	Blé et Farin	39224000
3	Afghanistan	2014	Blé et Farin	15160000
4	Afghanistan	2013	Céréales	40504000

Entrée [380]:

```
df_Aide = aide_alimentaire[['Zone', 'Année', 'quantiteReçu']]  
df_Aide = df_Aide.groupby(["Zone", "Année"]).sum().reset_index()
```

Entrée [381]:

```
df_Aide.head()
```

Out[381]:

	Zone	Année	quantiteReçu
0	Afghanistan	2013	128238000
1	Afghanistan	2014	57214000
2	Algérie	2013	35234000
3	Algérie	2014	18980000
4	Algérie	2015	17424000

- Création d'une liste contenant les 5 pays qui ont le plus bénéficiés de l'aide alimentaire

Entrée [382]:

```
liste_aide5Pays=['République arabe syrienne','Éthiopie','Yémen','Soudan du Sud','Soudan']
```

- On filtre sur le dataframe avec notre liste

Entrée [383]:

```
df_Aide=df_Aide.loc[df_Aide['Zone'].isin(liste_aide5Pays),:]
```

- Affichage des pays avec l'aide alimentaire par année

Entrée [384]:

```
df_Aide
```

Out[384]:

	Zone	Année	quantiteReçu
157	République arabe syrienne	2013	563566000
158	République arabe syrienne	2014	651870000
159	République arabe syrienne	2015	524949000
160	République arabe syrienne	2016	118558000
189	Soudan	2013	330230000
190	Soudan	2014	321904000
191	Soudan	2015	17650000
192	Soudan du Sud	2013	196330000
193	Soudan du Sud	2014	450610000
194	Soudan du Sud	2015	48308000
214	Yémen	2013	264764000
215	Yémen	2014	103840000
216	Yémen	2015	372306000
217	Yémen	2016	465574000
225	Éthiopie	2013	591404000
226	Éthiopie	2014	586624000
227	Éthiopie	2015	203266000

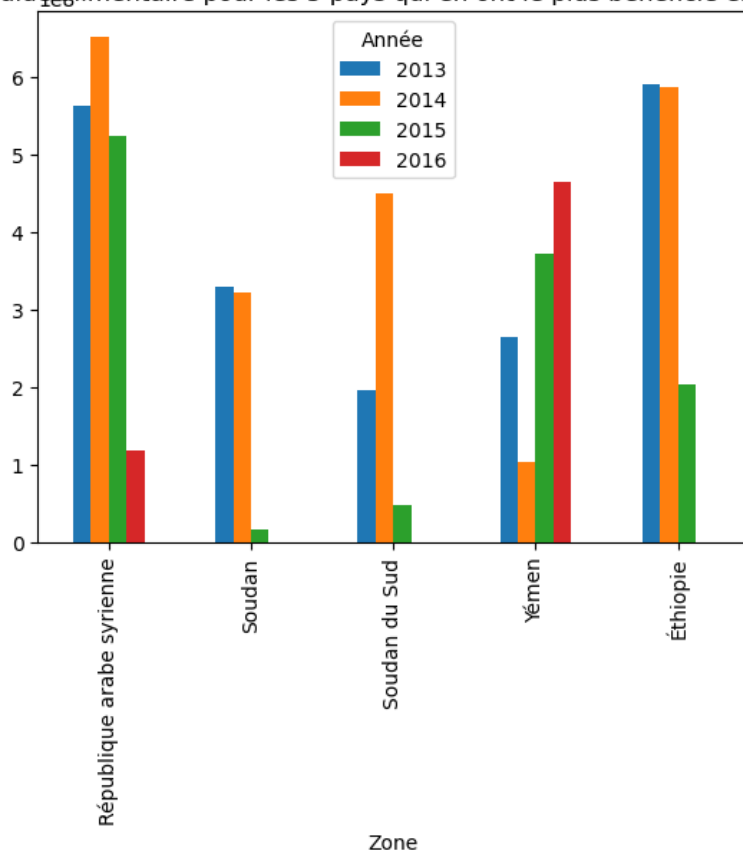
Entrée [385]:

```
df1 = df_Aide.pivot(index='Zone', columns='Année', values='quantiteReçu')
```

Entrée [386]:

```
df1.plot.bar(title="Évolution de l'aide alimentaire pour les 5 pays qui en ont le plus bé  
plt.show()
```

Évolution de l'aide alimentaire pour les 5 pays qui en ont le plus bénéficié entre 2013 et 2016



### 3.9 - Pays avec le moins de disponibilité par habitant

- Calcul de la disponibilité en kcal par personne par jour par pays

Entrée [387]:

```
dispo_alimentaire.head()
```

Out[387]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibili alimentaire (kg/personne/an)
0	Afghanistan	Abats Comestible	animale	0.0	0.0	5.0	1.0
1	Afghanistan	Agrumes, Autres	vegetale	0.0	0.0	1.0	1.0
2	Afghanistan	Aliments pour enfants	vegetale	0.0	0.0	1.0	0.0
3	Afghanistan	Ananas	vegetale	0.0	0.0	0.0	0.0
4	Afghanistan	Bananes	vegetale	0.0	0.0	4.0	2.0

Entrée [388]:

```
dispo_kcal_pers_jr_pays=dispo_alimentaire[['Zone', 'Disponibilité alimentaire (Kcal/personne/jour)']]
```

Entrée [389]:

```
dispo_kcal_pers_jr_pays.head()
```

Out[389]:

Disponibilité alimentaire (Kcal/personne/jour)	
Zone	
Afghanistan	2087.0
Afrique du Sud	3020.0
Albanie	3188.0
Algérie	3293.0
Allemagne	3503.0

- Affichage des 10 pays qui ont le moins de dispo alimentaire par personne

Entrée [390]:

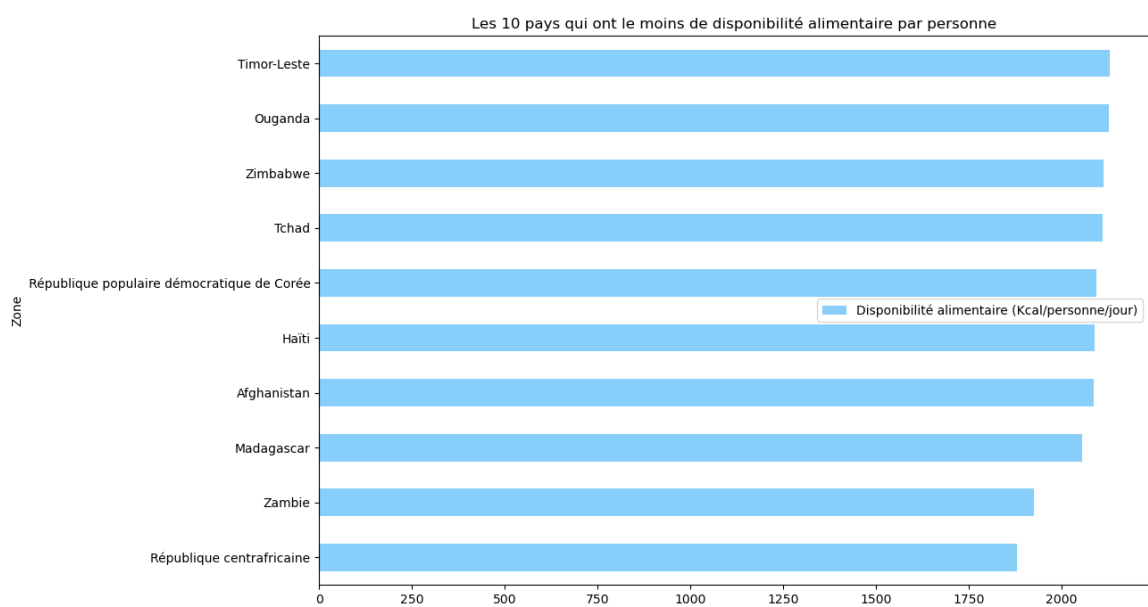
```
dispo_kcal_pers_jr_pays_min=dispo_kcal_pers_jr_pays.sort_values('Disponibilité alimentaire')
dispo_kcal_pers_jr_pays_min.head(10)
```

Out[390]:

Disponibilité alimentaire (Kcal/personne/jour)	
Zone	
République centrafricaine	1879.0
Zambie	1924.0
Madagascar	2056.0
Afghanistan	2087.0
Haïti	2089.0
République populaire démocratique de Corée	2093.0
Tchad	2109.0
Zimbabwe	2113.0
Ouganda	2126.0
Timor-Leste	2129.0

Entrée [391]:

```
dispo_kcal_pers_jr_pays_min.head(10).groupby("Zone")["Disponibilité alimentaire (Kcal/per
color="lightskyblue", figsize=(12,8), legend=True,title='Les 10 pays qui ont le moins de
```



### 3.10 - Pays avec le plus de disponibilité par habitant

- Affichage des 10 pays qui ont le plus de dispo alimentaire par personne

Entrée [392]:

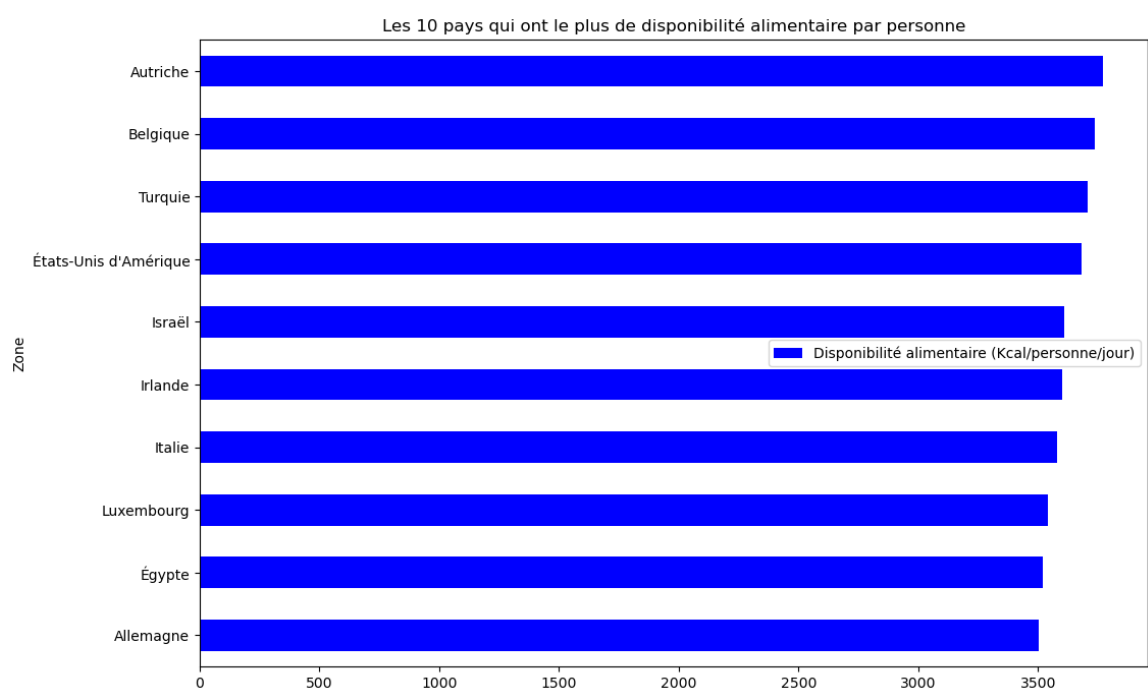
```
dispo_kcal_pers_jr_pays_max=dispo_kcal_pers_jr_pays.sort_values('Disponibilité alimentaire')
dispo_kcal_pers_jr_pays_max.head(10)
```

Out[392]:

Disponibilité alimentaire (Kcal/personne/jour)	
Zone	
Autriche	3770.0
Belgique	3737.0
Turquie	3708.0
États-Unis d'Amérique	3682.0
Israël	3610.0
Irlande	3602.0
Italie	3578.0
Luxembourg	3540.0
Égypte	3518.0
Allemagne	3503.0

Entrée [393]:

```
dispo_kcal_pers_jr_pays_max.head(10).groupby("Zone")["Disponibilité alimentaire (Kcal/per
color="blue", figsize=(12,8), legend=True,title='Les 10 pays qui ont le plus de dispo
```





### 3.11 - Exemple de la Thaïlande pour le Manioc

- Création d'un dataframe avec uniquement la Thaïlande

Entrée [394]:

```
df_Thaïlande=sous_nutrition.loc[sous_nutrition['Zone']=='Thaïlande',:]
```

Entrée [395]:

```
df_Thaïlande.head()
```

Out[395]:

	Zone	Année	popSNut
1110	Thaïlande	2013	6200000.0
1111	Thaïlande	2014	6000000.0
1112	Thaïlande	2015	5900000.0
1113	Thaïlande	2016	6000000.0
1114	Thaïlande	2017	6200000.0

Entrée [396]:

```
print("La population totale en état de sous nutrition en 2017 en Thaïlande est de: {:.0f}
```

La population totale en état de sous nutrition en 2017 en Thaïlande est de: 36800000 personnes

- Calcul de la sous nutrition en Thaïlande

Entrée [397]:

```
df_Thaïlande1=pd.merge(df_Thaïlande,population,on=['Zone','Année'])  
df_Thaïlande1.head()
```

Out[397]:

	Zone	Année	popSNut	Pop
0	Thaïlande	2013	6200000.0	68144518.0
1	Thaïlande	2014	6000000.0	68438746.0
2	Thaïlande	2015	5900000.0	68714511.0
3	Thaïlande	2016	6000000.0	68971308.0
4	Thaïlande	2017	6200000.0	69209810.0

Entrée [398]:

```
sous_nutrition_Thaïlande = round(df_Thaïlande1['popSNut'].sum() / df_Thaïlande1['Pop'].sum())
print("La proportion de sous nutrition en Thaïlande est de : {}".format(sous_nutrition_Thaïlande))
```

La proportion de sous nutrition en Thaïlande est de : 8.91%

- On calcule la proportion exportée en fonction de la production

Entrée [399]:

```
dispo_alim_maniocTh = dispo_alimentaire.loc[(dispo_alimentaire['Produit'] == 'Manioc') & (dispo_alim_maniocTh.head())
```

Out[399]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Dispo alime (kg/pers)
13809	Thaïlande	Manioc	vegetale	1.800000e+09	2.081000e+09	40.0	

Entrée [400]:

```
proportion_exportée = round(dispo_alim_maniocTh['Exportations - Quantité'].sum() / dispo_alim_maniocTh['Exportations - Quantité'].sum())
print("La proportion de manioc exportée Thaïlande en fonction de la production est de : {}".format(proportion_exportée))
```

La proportion de manioc exportée Thaïlande en fonction de la production est de : 83.41%

- Quelle est la disponibilité par habitant pour la Thaïlande

Entrée [401]:

```
dispo_kcal_pers_jr_Thai = dispo_alimentaire.loc[dispo_alimentaire['Zone'] == 'Thaïlande',]
print("La disponibilité alimentaire par habitant pour la Thaïlande est de : {} calories".format(dispo_kcal_pers_jr_Thai['Disponibilité alimentaire (Kcal/personne/jour)'].sum() / dispo_kcal_pers_jr_Thai['Population'].sum()))
```

La disponibilité alimentaire par habitant pour la Thaïlande est de : 2785.0 calories

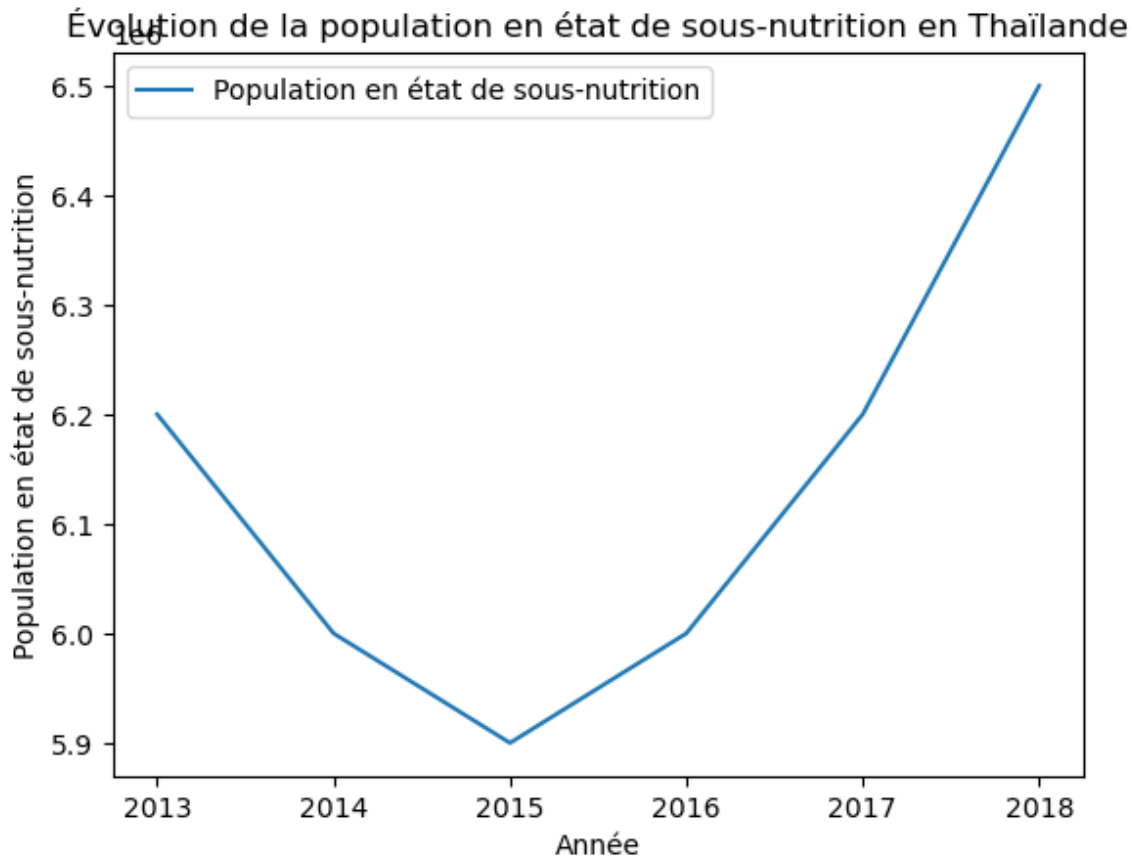
## Etape 4 - Analyse complémentaires

Ajouter en dessous toutes les analyses complémentaires suite à la demande de mélanie : "et toutes les infos que tu trouverais utiles pour mettre en relief les pays qui semblent être le plus en difficulté au niveau alimentaire"

- Évolution de la population en état de sous-nutrition en Thaïlande

Entrée [402]:

```
plt.plot(df_Thaïlande['Année'], df_Thaïlande['popSNut'])
plt.title("Évolution de la population en état de sous-nutrition en Thaïlande")
plt.xlabel("Année")
plt.ylabel("Population en état de sous-nutrition")
plt.legend(["Population en état de sous-nutrition"])
plt.show()
```



- Calcul de la balance commerciale de la Thaïlande en 2017

Entrée [403]:

```
dispo_alim_Th=df_popDA17.loc[df_popDA17['Zone'] == 'Thaïlande'].groupby('Produit').sum()
```

Entrée [404]:

```
dispo_alim_Th=df_popDA17[['Produit', 'Année', 'Pop', 'Disponibilité alimentaire (Kcal/person
```

Entrée [405]:

```
dispo_alim_Th
```

Out[405]:

	Année	Pop	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité
Produit						
Abats Comestible	2017	69209810.0	3.0	74000000.0	5.000000e+06	33000000.0
Agrumes, Autres	2017	69209810.0	0.0	8000000.0	6.000000e+06	2000000.0
Alcool, non Comestible	2017	69209810.0	0.0	358000000.0	1.100000e+08	21000000.0
Aliments pour enfants	2017	69209810.0	2.0	12000000.0	7.000000e+06	19000000.0
Ananas	2017	69209810.0	10.0	782000000.0	1.449000e+09	9000000.0
...	...	...	...	...	...	...
Viande de Suides	2017	69209810.0	124.0	871000000.0	2.200000e+07	1000000.0
Viande de Volailles	2017	69209810.0	52.0	945000000.0	5.360000e+08	11000000.0
Viande, Autre	2017	69209810.0	0.0	-92000000.0	9.600000e+07	4000000.0
Vin	2017	69209810.0	0.0	8000000.0	8.000000e+06	16000000.0
Épices, Autres	2017	69209810.0	16.0	114000000.0	4.200000e+07	13000000.0

95 rows × 8 columns

Entrée [406]:

```
balance_commerciale_Thai=dispo_alim_Th['Exportations - Quantité'].sum()-dispo_alim_Th['Im  
print("La balance commerciale de la Thaïlande en 2017 est de :",balance_commerciale_Thai)
```

La balance commerciale de la Thaïlande en 2017 est de : 39095000000.0

- Calcul de la proportion de personnes que la Thaïlande peut nourrir avec sa disponibilité intérieure en 2017

Entrée [407]:

```
population_totale_thai=69209810.0
```

Entrée [408]:

```
dispo_int_par_personne=dispo_alim_Th['Disponibilité intérieure'].sum()/population_totale_
```

Entrée [409]:

```
dispo_int_par_personne
```

Out[409]:

2284.7917079963086

Entrée [410]:

```
nb_personnes_nourries=dispo_int_par_personne/2500*100
print("La proportion de personnes que la Thaïlande peut nourrir avec sa disponibilité int
```

La proportion de personnes que la Thaïlande peut nourrir avec sa disponibi  
lité intérieure en 2017 est de: 91.39 %

- Liste des 10 produits les plus produits en Thaïlande

Entrée [411]:

```
production_thai_max=dispo_alim_Th.sort_values('Production',ascending=False).head(10)
production_thai_max
```

Out[411]:

	Année	Pop	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité
Produit						
Sucre, canne	2017	69209810.0	49.0	1.000960e+11	0.000000e+00	0.000000e+00
Manioc	2017	69209810.0	40.0	6.264000e+09	2.521400e+10	1.250000e+09
Riz (Eq Blanchi)	2017	69209810.0	1139.0	1.360500e+10	6.860000e+09	5.000000e+07
Sucre Eq Brut	2017	69209810.0	368.0	2.527000e+09	6.462000e+09	2.500000e+07
Fruits, Autres	2017	69209810.0	63.0	3.791000e+09	2.671000e+09	2.610000e+08
Maïs	2017	69209810.0	95.0	4.678000e+09	5.910000e+08	2.060000e+08
Légumes, Autres	2017	69209810.0	36.0	3.557000e+09	4.640000e+08	3.780000e+08
Bière	2017	69209810.0	41.0	2.040000e+09	2.490000e+08	1.700000e+07
Ananas	2017	69209810.0	10.0	7.820000e+08	1.449000e+09	9.000000e+06
Huile de Palme	2017	69209810.0	67.0	1.367000e+09	7.130000e+08	1.110000e+08

- Liste des 10 produits les plus exportés en Thaïlande

Entrée [412]:

```
export_thai_max=dispo_alim_Th.sort_values('Exportations - Quantité',ascending=False).head
```

Entrée [413]:

```
export_thai_max
```

Out[413]:

	Année	Pop	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importation - Quantit
Produit						
Manioc	2017	69209810.0	40.0	6.264000e+09	2.521400e+10	1.250000e+0
Riz (Eq Blanchi)	2017	69209810.0	1139.0	1.360500e+10	6.860000e+09	5.000000e+0
Sucre Eq Brut	2017	69209810.0	368.0	2.527000e+09	6.462000e+09	2.500000e+0
Fruits, Autres	2017	69209810.0	63.0	3.791000e+09	2.671000e+09	2.610000e+0
Ananas	2017	69209810.0	10.0	7.820000e+08	1.449000e+09	9.000000e+0
Poissons Pelagiques	2017	69209810.0	30.0	6.650000e+08	1.390000e+09	1.460000e+0
Huile de Palme	2017	69209810.0	67.0	1.367000e+09	7.130000e+08	1.110000e+0
Crustacés	2017	69209810.0	2.0	9.600000e+07	6.780000e+08	5.500000e+0
Maïs	2017	69209810.0	95.0	4.678000e+09	5.910000e+08	2.060000e+0
Viande de Volailles	2017	69209810.0	52.0	9.450000e+08	5.360000e+08	1.100000e+0

- Liste des 10 produits les plus importés en Thaïlande

Entrée [414]:

```
import_thai_max=dispo_alim_Th.sort_values('Importations - Quantité',ascending=False).head
```

Entrée [415]:

import\_thai\_max

	Annee	Pop	alimentaire (Kcal/personne/jour)	intérieure	- Quantité	- Quantité	Nourritui
Produit							
Blé	2017	69209810.0	76.0	1.882000e+09	2.370000e+08	2.118000e+09	7.330000e+C
Soja	2017	69209810.0	22.0	1.860000e+09	1.200000e+07	1.682000e+09	1.430000e+C
Poissons Pelagiques	2017	69209810.0	30.0	6.650000e+08	1.390000e+09	1.460000e+09	6.650000e+C
Manioc	2017	69209810.0	40.0	6.264000e+09	2.521400e+10	1.250000e+09	8.710000e+C
Lait - Excl Beurre	2017	69209810.0	45.0	2.147000e+09	2.000000e+08	1.241000e+09	1.967000e+C
Légumes, Autres	2017	69209810.0	36.0	3.557000e+09	4.640000e+08	3.780000e+08	3.207000e+C
Orge	2017	69209810.0	0.0	4.410000e+08	0.000000e+00	3.350000e+08	0.000000e+C
Fruits, Autres	2017	69209810.0	63.0	3.791000e+09	2.671000e+09	2.610000e+08	3.857000e+C

- Liste des 10 produits avec la plus forte disponibilité intérieure en Thaïlande

Entrée [416]:

disp\_int\_par\_produitmax=dispo\_alim\_Th.sort\_values('Disponibilité intérieure',ascending=Fa

Entrée [417]:

disp\_int\_par\_produitmax

Out[417]:

	Année	Pop	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité
Produit						
Sucre, canne	2017	69209810.0	49.0	1.000960e+11	0.000000e+00	0.000000e+00
Riz (Eq Blanchi)	2017	69209810.0	1139.0	1.360500e+10	6.860000e+09	5.000000e+07
Manioc	2017	69209810.0	40.0	6.264000e+09	2.521400e+10	1.250000e+09
Maïs	2017	69209810.0	95.0	4.678000e+09	5.910000e+08	2.060000e+08
Fruits, Autres	2017	69209810.0	63.0	3.791000e+09	2.671000e+09	2.610000e+08
Légumes, Autres	2017	69209810.0	36.0	3.557000e+09	4.640000e+08	3.780000e+08
Sucre Eq Brut	2017	69209810.0	368.0	2.527000e+09	6.462000e+09	2.500000e+07
Lait - Excl Beurre	2017	69209810.0	45.0	2.147000e+09	2.000000e+08	1.241000e+09
Bière	2017	69209810.0	41.0	2.040000e+09	2.490000e+08	1.700000e+07
Blé	2017	69209810.0	76.0	1.882000e+09	2.370000e+08	2.118000e+09

- Cration d'un dataframe avec uniquement le manioc



Entrée [418]:

```
DF=dispo_alimentaire.loc[dispo_alimentaire['Produit'] == 'Manioc']
DF
```

Out[418]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Dispo alime (kg/pers
110	Afrique du Sud	Manioc	vegetale	0.000000e+00	57000000.0	0.0	
203	Albanie	Manioc	vegetale	0.000000e+00	0.0	0.0	
295	Algérie	Manioc	vegetale	0.000000e+00	0.0	0.0	
390	Allemagne	Manioc	vegetale	1.000000e+06	26000000.0	0.0	
475	Angola	Manioc	vegetale	8.880000e+09	0.0	560.0	
...	...	...	...	...	...	...	
15108	Égypte	Manioc	vegetale	0.000000e+00	0.0	0.0	
15201	Émirats arabes unis	Manioc	vegetale	0.000000e+00	7000000.0	3.0	
15294	Équateur	Manioc	vegetale	0.000000e+00	1000000.0	6.0	
15390	États-Unis d'Amérique	Manioc	vegetale	2.880000e+08	245000000.0	0.0	
15567	Îles Salomon	Manioc	vegetale	0.000000e+00	0.0	15.0	

167 rows × 18 columns



- Liste des 10 premiers pays producteurs de manioc

Entrée [419]:

```
production_manioc=DF.sort_values('Production',ascending=False).head(10)
production_manioc
```

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	(g
9830	Nigéria	Manioc	vegetale	2.628800e+10	0.000000e+00	262.0	118.86	
13809	Thaïlande	Manioc	vegetale	1.800000e+09	2.081000e+09	40.0	13.00	
6334	Indonésie	Manioc	vegetale	4.800000e+08	8.894000e+09	132.0	46.97	
2129	Brésil	Manioc	vegetale	1.068400e+10	1.196000e+09	85.0	36.31	
475	Angola	Manioc	vegetale	8.880000e+09	0.000000e+00	560.0	197.90	
5255	Ghana	Manioc	vegetale	4.020000e+09	1.566000e+09	642.0	216.10	
9376	Mozambique	Manioc	vegetale	1.350000e+09	2.100000e+09	675.0	226.45	
14743	Viet Nam	Manioc	vegetale	6.980000e+08	0.000000e+00	22.0	8.09	
2665	Cambodge	Manioc	vegetale	0.000000e+00	6.562000e+09	70.0	25.33	

- Liste des 10 premiers pays exportateurs de manioc

Entrée [420]:

```
export_manioc=DF.sort_values('Exportations - Quantité',ascending=False).head(10)
export_manioc
```

Out[420]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Di alin (kg/pe
13809	Thaïlande	Manioc	vegetale	1.800000e+09	2.081000e+09	40.0	
14743	Viet Nam	Manioc	vegetale	6.980000e+08	0.000000e+00	22.0	
6334	Indonésie	Manioc	vegetale	4.800000e+08	8.894000e+09	132.0	
2665	Cambodge	Manioc	vegetale	0.000000e+00	6.562000e+09	70.0	
10752	Paraguay	Manioc	vegetale	1.376000e+09	1.150000e+08	289.0	
3698	Costa Rica	Manioc	vegetale	0.000000e+00	3.000000e+06	12.0	
3228	Chine, Taiwan Province de	Manioc	vegetale	1.000000e+06	1.317000e+09	0.0	
3043	Chine - RAS de Hong-Kong	Manioc	vegetale	4.000000e+06	0.000000e+00	3.0	
10386	Ouganda	Manioc	vegetale	2.100000e+07	0.000000e+00	227.0	
3323	Chine, continentale	Manioc	vegetale	2.286800e+10	7.940000e+09	6.0	

- Liste des 10 premiers pays importateurs de manioc

Entrée [421]:

```
import_manioc=DF.sort_values('Importations - Quantité',ascending=False).head(10)
import_manioc
```

Out[421]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Di alin (kg/pe
3323	Chine, continentale	Manioc	vegetale	2.286800e+10	7.940000e+09	6.0	
11769	République de Corée	Manioc	vegetale	0.000000e+00	1.230000e+08	0.0	
3228	Chine, Taiwan Province de	Manioc	vegetale	1.000000e+06	1.317000e+09	0.0	
13809	Thaïlande	Manioc	vegetale	1.800000e+09	2.081000e+09	40.0	
6334	Indonésie	Manioc	vegetale	4.800000e+08	8.894000e+09	132.0	
8378	Malaisie	Manioc	vegetale	3.000000e+06	8.360000e+08	6.0	
7073	Japon	Manioc	vegetale	0.000000e+00	0.000000e+00	0.0	
15390	États-Unis d'Amérique	Manioc	vegetale	2.880000e+08	2.450000e+08	0.0	
10940	Philippines	Manioc	vegetale	1.420000e+08	2.010000e+08	64.0	
1303	Bangladesh	Manioc	vegetale	0.000000e+00	1.240000e+08	0.0	

- Calcul de la balance commerciale des 5 pays les plus aidé

Entrée [422]:

```
liste_pays=['République arabe syrienne','Éthiopie','Yémen','Soudan du Sud','Soudan']
```

Entrée [423]:

```
dispo_Rep_Syr=dispo_alimentaire.loc[dispo_alimentaire['Zone'] == 'République arabe syrien']
dispo_Rep_Syr.head()
```

Out[423]:

Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	Dispo mati e (g/pers
------	---------	---------	-----------------------	---------------------	--	--	----------------------

Entrée [424]:

```
balance_Rep_Syr=dispo_Rep_Syr['Exportations - Quantité'].sum()-dispo_Rep_Syr['Importation  
print("La balance commerciale de la République arabe syrienne est de :",balance_Rep_Syr)
```

La balance commerciale de la République arabe syrienne est de : 0.0

Entrée [425]:

```
dispo_E=dispo_alimentaire.loc[dispo_alimentaire['Zone'] == 'Ethiopie']  
dispo_E.head()
```

Out[425]:

Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	Dispo mati e (g/pers

Entrée [426]:

```
balance_E=dispo_E['Exportations - Quantité'].sum()-dispo_E['Importations - Quantité'].sum  
print("La balance commerciale de l'Ethiopie est de :",balance_E)
```

La balance commerciale de l'Ethiopie est de : 0.0

Entrée [427]:

```
dispo_Y=dispo_alimentaire.loc[dispo_alimentaire['Zone'] == 'Yémen']  
dispo_Y.head()
```

Out[427]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibil alimentaire quant (kg/personne/a
14781	Yémen	Abats Comestible	animale	0.0	0.0	6.0	1.
14782	Yémen	Agrumes, Autres	vegetale	0.0	0.0	0.0	0.
14783	Yémen	Alcool, non Comestible	vegetale	0.0	0.0	0.0	0.
14784	Yémen	Aliments pour enfants	vegetale	0.0	0.0	2.0	0.
14785	Yémen	Ananas	vegetale	0.0	0.0	1.0	0.

Entrée [428]:


```
balance_y=dispo_Y['Exportations - Quantité'].sum()-dispo_Y['Importations - Quantité'].sum  
print("La balance commerciale du Yémen est de :",balance_y)
```

La balance commerciale du Yémen est de : -6067000000.0

Entrée [429]:

```
dispo_SoudanS=dispo_alimentaire.loc[dispo_alimentaire['Zone'] == 'Soudan du Sud']
dispo_SoudanS.head()
```

Out[429]:

Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibilité alimentaire en quantité (kg/personne/an)	Dispo mati e (g/pers
							

Entrée [430]:

```
balance_soudanS=dispo_SoudanS['Exportations - Quantité'].sum()-dispo_SoudanS['Importation']
print("La balance commerciale du Soudan du Sud est de :",balance_soudanS)
```

La balance commerciale du Soudan du Sud est de : 0.0


Entrée [431]:

```
liste_pays=['République arabe syrienne','Éthiopie','Yémen','Soudan du Sud','Soudan']
```

Entrée [432]:

```
dispo_Soudan=dispo_alimentaire.loc[dispo_alimentaire['Zone'] == 'Soudan']
dispo_Soudan.head()
```

Out[432]:

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	Disponibi alimentaire quan! (kg/personne/i
12980	Soudan	Abats Comestible	animale	0.0	0.0	15.0	4
12981	Soudan	Agrumes, Autres	vegetale	0.0	0.0	0.0	0
12982	Soudan	Alcool, non Comestible	vegetale	0.0	0.0	0.0	0
12983	Soudan	Aliments pour enfants	vegetale	0.0	0.0	0.0	0
12984	Soudan	Ananas	vegetale	0.0	0.0	0.0	0
							

Entrée [433]:

```
balance_soudan=dispo_Soudan['Exportations - Quantité'].sum()-dispo_Soudan['Importations - Quantité']
print("La balance commerciale du Soudan est de :",balance_soudan)
```

La balance commerciale du Soudan est de : -1777000000.0

