

dufour-david-1-notebook-042024

April 30, 2024

PROJET 4 DATA ANALYST

Réalisez une étude de santé publique avec R ou Python

1 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

Pour vous aider dans vos premiers pas à l'usage de Jupyter et de Python, nous avons rédigé ce notebook en vous indiquant les instructions à suivre.

Il vous suffit pour cela de saisir le code Python répondant à l'instruction donnée.

Vous verrez de temps à autre le code Python répondant à une instruction donnée mais cela est fait pour vous aider à comprendre la nature du travail qui vous est demandée.

Et garder à l'esprit, qu'il n'y a pas de solution unique pour résoudre un problème et qu'il y a autant de résolutions de problèmes que de développeurs ;)...

Note jeremy Est ce qu'il faut faire le calcul de la sous nutrition sur les pays qu'on a ? Est ce qu'il faut faire des graphiques ? Rajouter le soja La liste des céréales est difficile à trouver ...

Etape 1 - Importation des librairies et chargement des fichiers

1.1 - Importation des librairies

```
[1]: #Importation de la librairie Pandas
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1.2 - Chargement des fichiers Excel

```
[2]: #Importation du fichier population.csv
population = pd.read_csv('population.csv')

#Importation du fichier dispo_alimentaire.csv
dispo_alimentaire = pd.read_csv('dispo_alimentaire.csv')

#Importation du fichier aide_alimentaire.csv
aide_alimentaire = pd.read_csv('aide_alimentaire.csv')

#Importation du fichier sous_nutrition.csv
sous_nutrition = pd.read_csv('sous_nutrition.csv')
```

Etape 2 - Analyse exploratoire des fichiers

2.1 - Analyse exploratoire du fichier population

```
[3]: #Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".format(population.
    ↪shape[0]))
print("Le tableau comporte {} colonne(s)".format(population.shape[1]))
```

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

Le tableau comporte 3 colonne(s)

```
[4]: #Consulter le nombre de colonnes
#La nature des données dans chacune des colonnes
#Le nombre de valeurs présentes dans chacune des colonnes
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
```

```
[5]: #Affichage les 5 premières lignes de la table
population.head()
```

```
[5]:
```

	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

```
[6]: population.describe(include = 'all')
```

```
[6]:
```

	Zone	Année	Valeur
count	1416	1416.000000	1.416000e+03
unique	236	NaN	NaN
top	Afghanistan	NaN	NaN
freq	6	NaN	NaN
mean	NaN	2015.500000	3.144793e+04
std	NaN	1.708428	1.300812e+05
min	NaN	2013.000000	7.930000e-01
25%	NaN	2014.000000	3.783410e+02
50%	NaN	2015.500000	5.126480e+03
75%	NaN	2017.000000	1.930666e+04
max	NaN	2018.000000	1.427648e+06

```
[7]: #Nous allons harmoniser les unités. Pour cela, nous avons décidé de multiplier
      ↪ la population par 1000
      #Multiplication de la colonne valeur par 1000
      population['Valeur'] = population['Valeur']*1000
```

```
[8]: #changement du nom de la colonne Valeur par Population
      population.rename(columns = {'Valeur' : 'Population'}, inplace = True)
```

```
[9]: #Affichage les 5 premières lignes de la table pour voir les modifications
      population.head()
```

```
[9]:
```

	Zone	Année	Population
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

```
[10]: population_2017 = population.loc[population['Année'] == 2017, :].copy().
      ↪ reset_index()
      population_2017.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 236 entries, 0 to 235
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   index       236 non-null   int64
1   Zone        236 non-null   object
2   Année       236 non-null   int64
```

```

3   Population    236 non-null    float64
dtypes: float64(1), int64(2), object(1)
memory usage: 7.5+ KB

```

```

[11]: del population_2017['index']
      population_2017.head()

```

```

[11]:
      Zone  Année  Population
0   Afghanistan  2017  36296113.0
1  Afrique du Sud  2017  57009756.0
2      Albanie  2017   2884169.0
3      Algérie  2017  41389189.0
4   Allemagne  2017  82658409.0

```

```

[12]: pop = population_2017['Population'].sum()
      print(pop)

```

```

7548134111.0

```

2.2 - Analyse exploratoire du fichier disponibilité alimentaire

```

[13]: #Afficher les dimensions du dataset
      print("Le tableau comporte {} observation(s) ou article(s)".
            ↪format(dispo_alimentaire.shape[0]))
      print("Le tableau comporte {} colonne(s)".format(dispo_alimentaire.shape[1]))

```

```

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

```

```

[14]: #Consulter le nombre de colonnes
      dispo_alimentaire.info()

```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15605 entries, 0 to 15604
Data columns (total 18 columns):
 #   Column                                     Non-Null
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 non-
null  float64
4    Autres Utilisations                      5496 non-
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 non-
null float64
14  Production                                           9180 non-
null float64
15  Semences                                             2091 non-
null float64
16  Traitement                                           2292 non-
null float64
17  Variation de stock                                   6776 non-
null float64
dtypes: float64(15), object(3)
memory usage: 2.1+ MB

```

```

[15]: #Affichage les 5 premières lignes de la table
      dispo_alimentaire.head()

```

```

[15]:
      Zone      Produit  Origine  Aliments pour animaux \
0  Afghanistan  Abats Comestible  animale      NaN
1  Afghanistan  Agrumes, Autres  vegetale      NaN
2  Afghanistan  Aliments pour enfants  vegetale      NaN
3  Afghanistan      Ananas  vegetale      NaN
4  Afghanistan      Bananes  vegetale      NaN

      Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour) \
0      NaN      5.0
1      NaN      1.0
2      NaN      1.0
3      NaN      0.0
4      NaN      4.0

      Disponibilité alimentaire en quantité (kg/personne/an) \

```

0	1.72
1	1.29
2	0.06
3	0.00
4	2.70

	Disponibilité de matière grasse en quantité (g/personne/jour) \
0	0.20
1	0.01
2	0.01
3	NaN
4	0.02

	Disponibilité de protéines en quantité (g/personne/jour) \
0	0.77
1	0.02
2	0.03
3	NaN
4	0.05

	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité \
0	53.0	NaN	NaN
1	41.0	2.0	40.0
2	2.0	NaN	2.0
3	0.0	NaN	0.0
4	82.0	NaN	82.0

	Nourriture	Pertes	Production	Semences	Traitement	Variation de stock
0	53.0	NaN	53.0	NaN	NaN	NaN
1	39.0	2.0	3.0	NaN	NaN	NaN
2	2.0	NaN	NaN	NaN	NaN	NaN
3	0.0	NaN	NaN	NaN	NaN	NaN
4	82.0	NaN	NaN	NaN	NaN	NaN

```
[16]: dispo_alimentaire.describe(include='all')
```

```
[16]:
```

	Zone	Produit	Origine	Aliments pour animaux \
count	15605	15605	15605	2720.000000
unique	174	98	2	NaN
top	Inde	Abats Comestible	vegetale	NaN
freq	97	174	11896	NaN
mean	NaN	NaN	NaN	479.501838
std	NaN	NaN	NaN	4240.119637
min	NaN	NaN	NaN	0.000000
25%	NaN	NaN	NaN	0.000000
50%	NaN	NaN	NaN	4.000000
75%	NaN	NaN	NaN	74.000000

max	NaN	NaN	NaN	150000.000000
-----	-----	-----	-----	---------------

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	\
count	5496.000000	14241.000000	
unique	NaN	NaN	
top	NaN	NaN	
freq	NaN	NaN	
mean	157.391376	34.789832	
std	5076.785816	107.287655	
min	0.000000	-21.000000	
25%	0.000000	0.000000	
50%	0.000000	4.000000	
75%	4.000000	21.000000	
max	347309.000000	1711.000000	

	Disponibilité alimentaire en quantité (kg/personne/an)	\
count	14015.000000	
unique	NaN	
top	NaN	
freq	NaN	
mean	8.719368	
std	24.618223	
min	-1.930000	
25%	0.060000	
50%	0.830000	
75%	5.190000	
max	430.760000	

	Disponibilité de matière grasse en quantité (g/personne/jour)	\
count	11794.000000	
unique	NaN	
top	NaN	
freq	NaN	
mean	1.283111	
std	3.680399	
min	-0.030000	
25%	0.010000	
50%	0.080000	
75%	0.630000	
max	60.760000	

	Disponibilité de protéines en quantité (g/personne/jour)	\
count	11561.000000	
unique	NaN	
top	NaN	
freq	NaN	
mean	1.223608	

std	3.598686
min	-0.370000
25%	0.010000
50%	0.100000
75%	0.660000
max	54.970000

	Disponibilité intérieure	Exportations - Quantité \
count	15382.000000	12226.000000
unique	NaN	NaN
top	NaN	NaN
freq	NaN	NaN
mean	640.293460	110.596925
std	9067.267153	1053.318990
min	-3430.000000	-41.000000
25%	0.000000	0.000000
50%	7.000000	0.000000
75%	76.750000	9.000000
max	739267.000000	42797.000000

	Importations - Quantité	Nourriture	Pertes	Production \
count	14852.000000	14015.000000	4278.000000	9180.000000
unique	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN
mean	87.264543	347.931359	106.053763	1090.379085
std	717.372714	4475.704458	1113.100416	12067.344094
min	-201.000000	-246.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	2.000000
50%	2.000000	5.000000	4.000000	22.000000
75%	18.000000	52.000000	26.000000	191.250000
max	63381.000000	426850.000000	55047.000000	739267.000000

	Semences	Traitement	Variation de stock
count	2091.000000	2292.000000	6776.000000
unique	NaN	NaN	NaN
top	NaN	NaN	NaN
freq	NaN	NaN	NaN
mean	73.974653	961.905323	-15.407615
std	528.069224	10381.795904	549.834540
min	0.000000	-19.000000	-39863.000000
25%	0.000000	0.000000	0.000000
50%	2.000000	6.000000	0.000000
75%	17.000000	69.000000	0.000000
max	17060.000000	326711.000000	5284.000000


```
[17]: #remplacement des NaN dans le dataset par des 0
dispo_alimentaire1 = dispo_alimentaire.fillna(0).copy()
```

```
[18]: #multiplication de toutes les lignes contenant des milliers de tonnes en Kg
liste_col = ['Aliments pour animaux', 'Autres Utilisations', 'Disponibilité_
↳intérieure', 'Exportations - Quantité', 'Importations - Quantité',_
↳'Nourriture', 'Pertes', 'Production', 'Semences', 'Traitement']
for i in liste_col :
    dispo_alimentaire1[i] = dispo_alimentaire1[i]*1000000
```

```
[19]: #Affichage les 5 premières lignes de la table
dispo_alimentaire1.head()
```

```
[19]:
```

	Zone	Produit	Origine	Aliments pour animaux \
0	Afghanistan	Abats Comestible	animale	0.0
1	Afghanistan	Agrumes, Autres	vegetale	0.0
2	Afghanistan	Aliments pour enfants	vegetale	0.0
3	Afghanistan	Ananas	vegetale	0.0
4	Afghanistan	Bananes	vegetale	0.0

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour) \
0	0.0	5.0
1	0.0	1.0
2	0.0	1.0
3	0.0	0.0
4	0.0	4.0

	Disponibilité alimentaire en quantité (kg/personne/an) \
0	1.72
1	1.29
2	0.06
3	0.00
4	2.70

	Disponibilité de matière grasse en quantité (g/personne/jour) \
0	0.20
1	0.01
2	0.01
3	0.00
4	0.02

	Disponibilité de protéines en quantité (g/personne/jour) \
0	0.77
1	0.02
2	0.03
3	0.00
4	0.05

	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité \
0	53000000.0	0.0	0.0
1	41000000.0	2000000.0	40000000.0
2	2000000.0	0.0	2000000.0
3	0.0	0.0	0.0
4	82000000.0	0.0	82000000.0

	Nourriture	Pertes	Production	Semences	Traitement	Variation de stock
0	53000000.0	0.0	53000000.0	0.0	0.0	0.0
1	39000000.0	2000000.0	3000000.0	0.0	0.0	0.0
2	2000000.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0
4	82000000.0	0.0	0.0	0.0	0.0	0.0

2.3 - Analyse exploratoire du fichier aide alimentaire

```
[20]: #Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".
      format(aide_alimentaire.shape[0]))
print("Le tableau comporte {} colonne(s)".format(aide_alimentaire.shape[1]))
```

Le tableau comporte 1475 observation(s) ou article(s)

Le tableau comporte 4 colonne(s)

```
[21]: #Consulter le nombre de colonnes
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
```

```
[22]: #Affichage les 5 premières lignes de la table
aide_alimentaire.head()
```

	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

```
[23]: #changement du nom de la colonne Pays bénéficiaire par Zone
aide_alimentaire.rename(columns = {'Pays bénéficiaire' : 'Zone'}, inplace =
↳True)
```

```
[24]: #Multiplication de la colonne Aide_alimentaire qui contient des tonnes par 1000
↳pour avoir des kg
aide_alimentaire['Valeur'] = aide_alimentaire['Valeur']*1000
```

```
[25]: #Affichage les 5 premières lignes de la table
aide_alimentaire.head()
```

```
[25]:
```

	Zone	Année	Produit	Valeur
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

2.3 - Analyse exploratoire du fichier sous nutrition

```
[26]: #Afficher les dimensions du dataset
print("Le tableau comporte {} observation(s) ou article(s)".
↳format(sous_nutrition.shape[0]))
print("Le tableau comporte {} colonne(s)".format(sous_nutrition.shape[1]))
```

Le tableau comporte 1218 observation(s) ou article(s)

Le tableau comporte 3 colonne(s)

```
[27]: #Consulter le nombre de colonnes
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
```

```
[28]: #Afficher les 5 premières lignes de la table
sous_nutrition.head()
```

```
[28]:
```

	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

```
[29]: #Conversion de la colonne sous nutrition en numérique
sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'])
```

```
-----
ValueError                                Traceback (most recent call last)
File lib.pyx:2374, in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string "<0.1"

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
Cell In[29], line 2
      1 #Conversion de la colonne sous nutrition en numérique
----> 2 sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'])

File ~\anaconda3\Lib\site-packages\pandas\core\tools\numeric.py:222, in
->to_numeric(arg, errors, downcast, dtype_backend)
    220 coerce_numeric = errors not in ("ignore", "raise")
    221 try:
--> 222     values, new_mask = lib.maybe_convert_numeric( # type:
->ignore[call-overload] # noqa: E501
    223         values,
    224         set(),
    225         coerce_numeric=coerce_numeric,
    226         convert_to_masked_nullable=dtype_backend is not lib.no_default
    227         or isinstance(values_dtype, StringDtype)
    228         and not values_dtype.storage == "pyarrow_numpy",
    229     )
    230 except (ValueError, TypeError):
    231     if errors == "raise":

File lib.pyx:2416, in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string "<0.1" at position 60
```

```
[30]: #Conversion de la colonne (avec l'argument errors=coerce qui permet de
->convertir automatiquement les lignes qui ne sont pas des nombres en NaN)
#Puis remplacement des NaN en 0
```

```
sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'],
↳errors='coerce')
sous_nutrition = sous_nutrition.fillna(0)
```

```
[31]: #changement du nom de la colonne Valeur par sous_nutrition
sous_nutrition.rename(columns={'Valeur' : 'sous_nutrition'}, inplace=True)
```

```
[32]: #Multiplication de la colonne sous_nutrition par 1000000
sous_nutrition['sous_nutrition'] = sous_nutrition['sous_nutrition']*1000000
```

```
[33]: #Afficher les 5 premières lignes de la table
sous_nutrition.head()
```

```
[33]:
```

	Zone	Année	sous_nutrition
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

```
[34]: sous_nutrition['Année'].unique()
```

```
[34]: array(['2012-2014', '2013-2015', '2014-2016', '2015-2017', '2016-2018',
'2017-2019'], dtype=object)
```

```
[35]: sous_nutrition_2017 = sous_nutrition.loc[sous_nutrition['Année'] ==
↳'2016-2018', :].reset_index()
```

```
[36]: sous_nutrition_2017.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 203 entries, 0 to 202
Data columns (total 4 columns):
#   Column          Non-Null Count  Dtype
---  -
0   index           203 non-null   int64
1   Zone            203 non-null   object
2   Année           203 non-null   object
3   sous_nutrition  203 non-null   float64
dtypes: float64(1), int64(1), object(2)
memory usage: 6.5+ KB
```

```
[37]: sous_nutrition_2017.head()
```

```
[37]:
```

	index	Zone	Année	sous_nutrition
0	4	Afghanistan	2016-2018	10500000.0
1	10	Afrique du Sud	2016-2018	3100000.0

2	16	Albanie	2016-2018	100000.0
3	22	Algérie	2016-2018	1300000.0
4	28	Allemagne	2016-2018	0.0

```
[38]: sous_nutrition_2017 = sous_nutrition_2017.replace('2016-2018', '2017').copy()
```

```
[39]: del sous_nutrition_2017['index']
sous_nutrition_2017.head()
```

```
[39]:
```

	Zone	Année	sous_nutrition
0	Afghanistan	2017	10500000.0
1	Afrique du Sud	2017	3100000.0
2	Albanie	2017	100000.0
3	Algérie	2017	1300000.0
4	Allemagne	2017	0.0

3.1 - Proportion de personnes en sous nutrition

```
[40]: # Il faut tout d'abord faire une jointure entre la table population et la table
      ↪ sous nutrition, en ciblant l'année 2017
proportion_sous_nutrition = pd.merge(population_2017,
      ↪ sous_nutrition_2017[['Zone', 'sous_nutrition']], on='Zone', how='inner')
```

```
[41]: #Affichage du dataset
proportion_sous_nutrition.head()
```

```
[41]:
```

	Zone	Année	Population	sous_nutrition
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

```
[42]: #Calcul et affichage du nombre de personnes en état de sous nutrition
nb_personnes_sous_nutrition = proportion_sous_nutrition['sous_nutrition'].sum()
print("Le nombre des personnes en état de sous-nutrition est de : {} personnes.
      ↪ ".format(round(nb_personnes_sous_nutrition)))
```

Le nombre des personnes en état de sous-nutrition est de : 535700000 personnes.

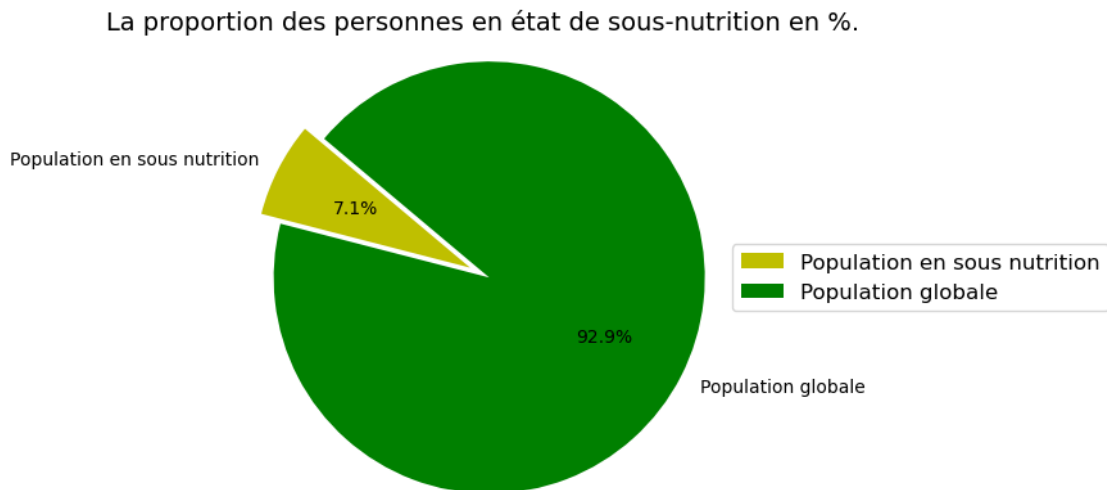
```
[43]: #proportion a calculer = valeur
proportion_personne_sous_nutrition = nb_personnes_sous_nutrition*100/
      ↪ proportion_sous_nutrition['Population'].sum()
population = proportion_sous_nutrition['Population'].sum()
print("La proportion des personnes en état de sous-nutrition est de {}% de la
      ↪ population globale.".format(round(proportion_personne_sous_nutrition)))
```

La proportion des personnes en état de sous-nutrition est de 7% de la population globale.

```
[130]: # Data to plot
labels = ['Population en sous nutrition', 'Population globale']
sizes = [proportion_personne_sous_nutrition,
        ↪100-proportion_personne_sous_nutrition]
colors = ['y', 'g']
explode = (0.1, 0) # explode 1st slice

# Plot
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.
        ↪1f%%', startangle=140)

plt.axis('equal')
plt.title("La proportion des personnes en état de sous-nutrition en %".
        ↪format(round(proportion_personne_sous_nutrition,2)), fontsize=14)
plt.legend(loc='center right', fontsize=12, bbox_to_anchor=(1.5,0.5))
plt.show()
```



```
[45]: sous_nutrition.head()
```

```
[45]:
```

	Zone	Année	sous_nutrition
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

```
[46]: sous_nutrition_evol = sous_nutrition.groupby('Année').sum().reset_index()
sous_nutrition_evol.head()
```

```
[46]:
```

	Année	Zone \
0	2012-2014	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
1	2013-2015	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
2	2014-2016	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
3	2015-2017	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
4	2016-2018	AfghanistanAfrique du SudAlbanieAlgérieAllemag...


```

sous_nutrition
0    528100000.0
1    523500000.0
2    524700000.0
3    528600000.0
4    535700000.0

```

```
[47]: sous_nutrition_evol['An'] = ['2013', '2014', '2015', '2016', '2017', '2018']
sous_nutrition_evol.head()
```

```
[47]:
```

	Année	Zone \
0	2012-2014	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
1	2013-2015	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
2	2014-2016	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
3	2015-2017	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
4	2016-2018	AfghanistanAfrique du SudAlbanieAlgérieAllemag...

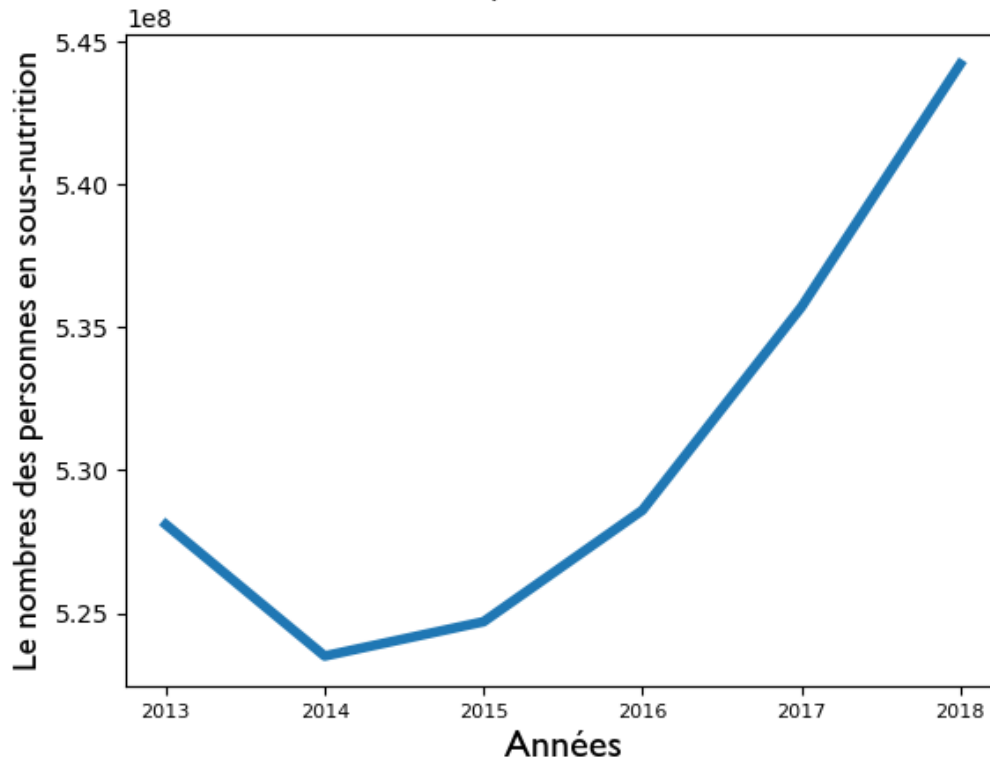

```

sous_nutrition  An
0    528100000.0  2013
1    523500000.0  2014
2    524700000.0  2015
3    528600000.0  2016
4    535700000.0  2017

```

```
[48]: plt.plot(sous_nutrition_evol['An'], sous_nutrition_evol['sous_nutrition'],
↳linewidth=4)
plt.xlabel("Années", fontname='Gill Sans MT', fontsize=16)
plt.ylabel("Le nombres des personnes en sous-nutrition", fontname='Gill Sans
↳MT', fontsize=14)
plt.xticks(fontsize=8)
plt.title("L'évolution du nombre des personnes en état de sous-nutrition",
↳fontsize=14)
plt.show()
```


L'évolution du nombre des personnes en état de sous-nutrition



3.2 - Nombre théorique de personne qui pourrait être nourries

[49]: *#Combien mange en moyenne un être humain ? Source => <https://www.cerin.org/articles/references-nutritionnelles-pour-les-adultes/>
 ↳ un homme dépense par jour 2600 kcal et une femme 2100 kcal je vais donc
 ↳ prendre une moyenne de 2350 kcal/jour comme besoin journalier*
 besoin_journalier = 2350

[50]: `dispo_alimentaire_group = dispo_alimentaire1.groupby('Zone').sum().reset_index()
 dispo_alimentaire_group.head()`

[50]:

	Zone	Produit \
0	Afghanistan	Abats ComestibleAgrumes, AutresAliments pour e...
1	Afrique du Sud	Abats ComestibleAgrumes, AutresAlcool, non Com...
2	Albanie	Abats ComestibleAgrumes, AutresAlcool, non Com...
3	Algérie	Abats ComestibleAgrumes, AutresAlcool, non Com...
4	Allemagne	Abats ComestibleAgrumes, AutresAlcool, non Com...

	Origine	Aliments pour animaux \
0	animalevegetalevegetalevegetalevegetaleanimale...	7.680000e+08
1	animalevegetalevegetalevegetalevegetaleanimale...	5.309000e+09

2	animalevegetalevegetalevegetalevegetaleanimale...	6.600000e+08
3	animalevegetalevegetalevegetalevegetaleanimale...	4.352000e+09
4	animalevegetalevegetalevegetalevegetaleanimale...	3.020900e+10

Autres Utilisations Disponibilité alimentaire (Kcal/personne/jour) \		
0	4.150000e+08	2087.0
1	8.760000e+08	3020.0
2	1.740000e+08	3188.0
3	1.707000e+09	3293.0
4	7.711000e+09	3503.0

Disponibilité alimentaire en quantité (kg/personne/an) \	
0	351.41
1	564.80
2	1095.54
3	809.13
4	957.52

Disponibilité de matière grasse en quantité (g/personne/jour) \	
0	33.50
1	83.58
2	106.48
3	76.48
4	141.62

Disponibilité de protéines en quantité (g/personne/jour) \	
0	58.26
1	85.33
2	111.37
3	91.92
4	101.39

Disponibilité intérieure	Exportations - Quantité	Importations - Quantité \
0 1.351500e+10	2.780000e+08	2.988000e+09
1 6.125600e+10	1.167400e+10	7.517000e+09
2 4.758000e+09	9.400000e+07	9.330000e+08
3 4.263000e+10	6.550000e+08	1.740300e+10
4 1.622750e+11	5.743100e+10	6.535300e+10

Nourriture	Pertes	Production	Semences	Traitement \
0 1.073500e+10	1.135000e+09	1.117100e+10	3.950000e+08	6.300000e+07
1 2.981200e+10	2.193000e+09	6.326300e+10	2.530000e+08	2.281900e+10
2 3.476000e+09	2.760000e+08	3.964000e+09	5.500000e+07	1.130000e+08
3 3.172900e+10	3.753000e+09	2.635900e+10	4.490000e+08	6.370000e+08
4 7.923800e+10	3.781000e+09	1.545470e+11	1.551000e+09	3.984200e+10

Variation de stock

```

0          -368.0
1          2153.0
2           -45.0
3         -479.0
4         -192.0

```

```

[51]: #On commence par faire une jointure entre le data frame population et
      ↪Dispo_alimentaire afin d'ajouter dans ce dernier la population
dispo_alimentaire_population = pd.merge(dispo_alimentaire_group,
      ↪population_2017[['Zone', 'Population']], on = 'Zone', how = 'left')

```

```

[52]: #Affichage du nouveau dataframe
dispo_alimentaire_population.head()

```

```

[52]:
      Zone
0  Afghanistan  Abats ComestibleAgrumes, AutresAliments pour e...
1  Afrique du Sud  Abats ComestibleAgrumes, AutresAlcool, non Com...
2      Albanie  Abats ComestibleAgrumes, AutresAlcool, non Com...
3      Algérie  Abats ComestibleAgrumes, AutresAlcool, non Com...
4  Allemagne  Abats ComestibleAgrumes, AutresAlcool, non Com...

      Origine  Aliments pour animaux \
0  animalevegetalevegetalevegetalevegetaleanimale...  7.680000e+08
1  animalevegetalevegetalevegetalevegetaleanimale...  5.309000e+09
2  animalevegetalevegetalevegetalevegetaleanimale...  6.600000e+08
3  animalevegetalevegetalevegetalevegetaleanimale...  4.352000e+09
4  animalevegetalevegetalevegetalevegetaleanimale...  3.020900e+10

      Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour) \
0      4.150000e+08  2087.0
1      8.760000e+08  3020.0
2      1.740000e+08  3188.0
3      1.707000e+09  3293.0
4      7.711000e+09  3503.0

      Disponibilité alimentaire en quantité (kg/personne/an) \
0      351.41
1      564.80
2     1095.54
3      809.13
4      957.52

      Disponibilité de matière grasse en quantité (g/personne/jour) \
0      33.50
1      83.58
2     106.48
3      76.48

```

4 141.62

	Disponibilité de protéines en quantité (g/personne/jour) \
0	58.26
1	85.33
2	111.37
3	91.92
4	101.39

	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité \
0	1.351500e+10	2.780000e+08	2.988000e+09
1	6.125600e+10	1.167400e+10	7.517000e+09
2	4.758000e+09	9.400000e+07	9.330000e+08
3	4.263000e+10	6.550000e+08	1.740300e+10
4	1.622750e+11	5.743100e+10	6.535300e+10

	Nourriture	Pertes	Production	Semences	Traitement \
0	1.073500e+10	1.135000e+09	1.117100e+10	3.950000e+08	6.300000e+07
1	2.981200e+10	2.193000e+09	6.326300e+10	2.530000e+08	2.281900e+10
2	3.476000e+09	2.760000e+08	3.964000e+09	5.500000e+07	1.130000e+08
3	3.172900e+10	3.753000e+09	2.635900e+10	4.490000e+08	6.370000e+08
4	7.923800e+10	3.781000e+09	1.545470e+11	1.551000e+09	3.984200e+10

	Variation de stock	Population
0	-368.0	36296113.0
1	2153.0	57009756.0
2	-45.0	2884169.0
3	-479.0	41389189.0
4	-192.0	82658409.0

```
[53]: #Création de la colonne dispo_kcal avec calcul des kcal disponibles mondialement
dispo_alimentaire_population['Disponibilité annuelle'] =_
↳(dispo_alimentaire_population['Disponibilité alimentaire (Kcal/personne/
↳jour)'] * dispo_alimentaire_population['Population']*365)/1000
dispo_alimentaire_population.head()
```

	Zone	Produit \
0	Afghanistan	Abats ComestibleAgrumes, AutresAliments pour e...
1	Afrique du Sud	Abats ComestibleAgrumes, AutresAlcool, non Com...
2	Albanie	Abats ComestibleAgrumes, AutresAlcool, non Com...
3	Algérie	Abats ComestibleAgrumes, AutresAlcool, non Com...
4	Allemagne	Abats ComestibleAgrumes, AutresAlcool, non Com...

	Origine	Aliments pour animaux \
0	animalevegetalevegetalevegetalevegetaleanimale...	7.680000e+08
1	animalevegetalevegetalevegetalevegetaleanimale...	5.309000e+09
2	animalevegetalevegetalevegetalevegetaleanimale...	6.600000e+08

3	animalevegetalevegetalevegetalevegetaleanimale...	4.352000e+09
4	animalevegetalevegetalevegetalevegetaleanimale...	3.020900e+10

Autres Utilisations		Disponibilité alimentaire (Kcal/personne/jour)	\
0	4.150000e+08	2087.0	
1	8.760000e+08	3020.0	
2	1.740000e+08	3188.0	
3	1.707000e+09	3293.0	
4	7.711000e+09	3503.0	

Disponibilité alimentaire en quantité (kg/personne/an)		\
0	351.41	
1	564.80	
2	1095.54	
3	809.13	
4	957.52	

Disponibilité de matière grasse en quantité (g/personne/jour)		\
0	33.50	
1	83.58	
2	106.48	
3	76.48	
4	141.62	

Disponibilité de protéines en quantité (g/personne/jour)		\
0	58.26	
1	85.33	
2	111.37	
3	91.92	
4	101.39	

Disponibilité intérieure	Exportations - Quantité	Importations - Quantité	\
0 1.351500e+10	2.780000e+08	2.988000e+09	
1 6.125600e+10	1.167400e+10	7.517000e+09	
2 4.758000e+09	9.400000e+07	9.330000e+08	
3 4.263000e+10	6.550000e+08	1.740300e+10	
4 1.622750e+11	5.743100e+10	6.535300e+10	

Nourriture	Pertes	Production	Semences	Traitement	\
0 1.073500e+10	1.135000e+09	1.117100e+10	3.950000e+08	6.300000e+07	
1 2.981200e+10	2.193000e+09	6.326300e+10	2.530000e+08	2.281900e+10	
2 3.476000e+09	2.760000e+08	3.964000e+09	5.500000e+07	1.130000e+08	
3 3.172900e+10	3.753000e+09	2.635900e+10	4.490000e+08	6.370000e+08	
4 7.923800e+10	3.781000e+09	1.545470e+11	1.551000e+09	3.984200e+10	

Variation de stock	Population	Disponibilité annuelle
0 -368.0	36296113.0	2.764875e+10

1	2153.0	57009756.0	6.284185e+10
2	-45.0	2884169.0	3.356077e+09
3	-479.0	41389189.0	4.974753e+10
4	-192.0	82658409.0	1.056866e+11

```
[54]: #Calcul du nombre d'humains pouvant être nourris
nombres_humains_nourris = dispo_alimentaire_population['Disponibilité_
↪annuelle'].sum()*1000/(besoin_journalier*365)
print("Le nombre de personnes pouvant être nourris avec la disponibilité_
↪alimentaire totales est de", round(nombres_humains_nourris), "personnes.")
```

Le nombre de personnes pouvant être nourris avec la disponibilité alimentaire totales est de 8901695586 personnes.

```
[55]: Proportion_globale = nombres_humains_nourris*100/population
print("La proportion totale de la population qui peut être nourris avec la_
↪disponibilité alimentaire est de :", round(Proportion_globale), "%")
```

La proportion totale de la population qui peut être nourris avec la disponibilité alimentaire est de : 118 %

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

```
[56]: #Transfert des données avec les végétaux dans un nouveau dataframe
dispo_alimentaire_group = dispo_alimentaire1.groupby(['Zone', 'Origine']).sum().
↪reset_index()
dispo_alimentaire_group_vegetale = dispo_alimentaire_group.
↪loc[dispo_alimentaire_group['Origine'] == 'vegetale', :].reset_index()
```

```
[57]: dispo_alimentaire_group_vegetale[['Zone', 'Origine', 'Disponibilité alimentaire_
↪(Kcal/personne/jour)']].head()
```

```
[57]:
```

	Zone	Origine	Disponibilité alimentaire (Kcal/personne/jour)
0	Afghanistan	vegetale	1871.0
1	Afrique du Sud	vegetale	2533.0
2	Albanie	vegetale	2203.0
3	Algérie	vegetale	2915.0
4	Allemagne	vegetale	2461.0

```
[58]: #Jointure entre le data frame population et produits_vegetaux afin d'ajouter_
↪dans ce dernier la population
produits_vegetaux_population2 = pd.merge(dispo_alimentaire_group_vegetale,
↪population_2017[['Zone', 'Population']], on = 'Zone', how = 'left')
produits_vegetaux_population2[['Zone', 'Origine', 'Disponibilité alimentaire_
↪(Kcal/personne/jour)', 'Population']].head()
```

```
[58]:
```

	Zone	Origine	Disponibilité alimentaire (Kcal/personne/jour)	\
0	Afghanistan	vegetale	1871.0	
1	Afrique du Sud	vegetale	2533.0	
2	Albanie	vegetale	2203.0	
3	Algérie	vegetale	2915.0	
4	Allemagne	vegetale	2461.0	

	Population
0	36296113.0
1	57009756.0
2	2884169.0
3	41389189.0
4	82658409.0

```
[59]: produits_vegetaux_population2['Total disponibilité produits vegetaux'] =
↳ produits_vegetaux_population2['Disponibilité alimentaire (Kcal/personne/
↳ jour)'] * produits_vegetaux_population2['Population']*365
produits_vegetaux_population2[['Zone', 'Origine', 'Disponibilité alimentaire_
↳ (Kcal/personne/jour)', 'Population', 'Total disponibilité produits_
↳ vegetaux']].head()
```

```
[59]:
```

	Zone	Origine	Disponibilité alimentaire (Kcal/personne/jour)	\
0	Afghanistan	vegetale	1871.0	
1	Afrique du Sud	vegetale	2533.0	
2	Albanie	vegetale	2203.0	
3	Algérie	vegetale	2915.0	
4	Allemagne	vegetale	2461.0	

	Population	Total disponibilité produits vegetaux
0	36296113.0	2.478716e+13
1	57009756.0	5.270808e+13
2	2884169.0	2.319146e+12
3	41389189.0	4.403706e+13
4	82658409.0	7.424916e+13

```
[60]: #Calcul du nombre de kcal disponible pour les végétaux
nombre_kcal_disponible_vegetaux = produits_vegetaux_population2['Total_
↳ disponibilité produits vegetaux'].sum()
print("Le nombre de kcal disponible pour les végétaux :",
↳ nombre_kcal_disponible_vegetaux)
```

Le nombre de kcal disponible pour les végétaux : 6300178937197865.0

```
[61]: #Calcul du nombre d'humains pouvant être nourris avec les végétaux
Le_nombre_humain_nourris_vegetaux = nombre_kcal_disponible_vegetaux/
↳ (besoin_journalier*365)
```

```
print("Le nombre de personnes pouvant être nourris avec les végétaux est de :",  
      ↪round(Le_nombre_humain_nourris_vegetaux))
```

Le nombre de personnes pouvant être nourris avec les végétaux est de :
7345006047

```
[62]: print("La proportion de la population pouvant être nourris uniquement avec les  
      ↪produits vegetaux est de :", round(Le_nombre_humain_nourris_vegetaux*100/  
      ↪population),"%")
```

La proportion de la population pouvant être nourris uniquement avec les produits
vegetaux est de : 97 %

3.4 - Utilisation de la disponibilité intérieure

```
[63]: #Calcul de la disponibilité totale  
Disponibilite_totale = dispo_alimentaire_population['Disponibilité intérieure'].  
      ↪sum()  
print("La Disponibilité tototale est de :", Disponibilite_totale)
```

La Disponibilité tototale est de : 9848994000000.0

```
[64]: #création d'une boucle for pour afficher les différentes valeurs en fonction  
      ↪des colonnes aliments pour animaux, pertes, nourritures,  
liste2 = ['Aliments pour animaux', 'Pertes', 'Nourriture', 'Semences',  
      ↪'Traitement', 'Autres Utilisations']  
liste4 = []  
for i in liste2 :  
    print(i, dispo_alimentaire_population[i].sum())  
    liste4.append(dispo_alimentaire_population[i].sum()*100/  
      ↪dispo_alimentaire_population['Disponibilité intérieure'].sum())
```

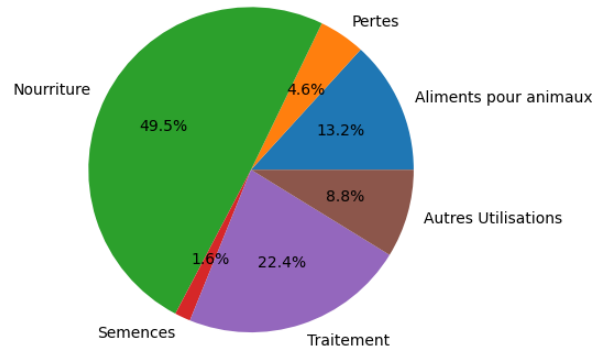
Aliments pour animaux 1304245000000.0
Pertes 453698000000.0
Nourriture 4876258000000.0
Semences 154681000000.0
Traitement 2204687000000.0
Autres Utilisations 865023000000.0

```
[65]: print(liste4)
```

[13.242418464261426, 4.606541541196999, 49.51021393657058, 1.5705258831511117,
22.384895350733284, 8.782856401374596]

```
[66]: plt.pie(liste4, labels=liste2, autopct='%1.1f%%')  
plt.title("Disponibilité totale : Aliments pour animaux, Pertes, Alimentation  
      ↪humaine, Semences, Traitement et Autres utilisations", fontsize=12)  
plt.show()
```


Disponibilité totale : Aliments pour animaux, Pertes, Alimentation humaine, Semences, Traitement et Autres utilisations



3.5 - Utilisation des céréales

```
[67]: #Création d'une liste avec toutes les variables
liste_variables_produit = ['Abats Comestible', 'Agrumes, Autres', 'Aliments_
    pour enfants',
    'Ananas', 'Bananes', 'Beurre, Ghee', 'Bière', 'Blé',
    'Boissons Alcooliques', 'Café', 'Coco (Incl Coprah)', 'Crème',
    'Céréales, Autres', 'Dattes', 'Edulcorants Autres',
    'Fève de Cacao', 'Fruits, Autres', 'Graines de coton',
    'Graines de tournesol', 'Graisses Animales Crue',
    'Huile Plantes Oleif Autr', 'Huile Graines de Coton',
    'Huile d'Arachide', 'Huile d'Olive', 'Huile de Colza&Moutarde',
    'Huile de Palme', 'Huile de Soja', 'Huile de Sésame',
    'Huile de Tournesol', 'Lait - Excl Beurre', 'Légumes, Autres',
    'Légumineuses Autres', 'Maïs', 'Miel', 'Millet', 'Miscellanees',
    'Noix', 'Oeufs', 'Olives', 'Oranges, Mandarines', 'Orge',
    'Plantes Oleiferes, Autre', 'Poissons Eau Douce', 'Poivre',
    'Pommes', 'Pommes de Terre', 'Raisin', 'Riz (Eq Blanchi)',
    'Sucre Eq Brut', 'Sucre, betterave', 'Sucre, canne', 'Sésame',
    'Thé', 'Tomates', 'Viande d'Ovins/Caprins', 'Viande de Bovins',
    'Viande de Volailles', 'Viande, Autre', 'Vin', 'Épices, Autres',
    'Alcool, non Comestible', 'Animaux Aquatiques Autre',
    'Arachides Decortiquees', 'Avoine', 'Bananes plantains',
    'Boissons Fermentés', 'Cephalopodes', 'Citrons & Limes',
    'Crustacés', 'Girofles', 'Graines Colza/Moutarde', 'Haricots',
    'Huile de Coco', 'Huile de Germe de Maïs', 'Huile de Palmistes',
    'Huiles de Foie de Poisso', 'Huiles de Poissons', 'Ignames',
    'Manioc', 'Mollusques, Autres', 'Oignons', 'Palmistes',
    'Pamplemousse', 'Patates douces', 'Perciform', 'Piments',
    'Plantes Aquatiques', 'Pois', 'Poissons Marins, Autres',
    'Poissons Pelagiques', 'Racines nda', 'Seigle', 'Soja', 'Sorgho',
    'Viande de Suides', 'Huile de Son de Riz', 'Sucre non centrifugé',
```

```
'Viande de Anim Aquatiq']
```

```
[68]: liste_cereales = ['Blé', 'Avoine', 'Céréales, Autres', 'Maïs', 'Millet',  
↳ 'Orge', 'Riz (Eq Blanchi)', 'Seigle', 'Sorgho', 'Soja']
```

```
[69]: #Création d'un dataframe avec les informations uniquement pour ces céréales  
dispo_alimentaire_cereales = dispo_alimentaire1.  
↳ loc[dispo_alimentaire1['Produit'].isin(liste_cereales),:]  
dispo_alimentaire_cereales.head()
```

```
[69]:
```

	Zone	Produit	Origine	Aliments pour animaux	\
7	Afghanistan	Blé	vegetale	0.0	
12	Afghanistan	Céréales, Autres	vegetale	0.0	
32	Afghanistan	Maïs	vegetale	200000000.0	
34	Afghanistan	Millet	vegetale	0.0	
40	Afghanistan	Orge	vegetale	360000000.0	

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	\
7	0.0	1369.0	
12	0.0	0.0	
32	0.0	21.0	
34	0.0	3.0	
40	0.0	26.0	

	Disponibilité alimentaire en quantité (kg/personne/an)	\
7	160.23	
12	0.00	
32	2.50	
34	0.40	
40	2.92	

	Disponibilité de matière grasse en quantité (g/personne/jour)	\
7	4.69	
12	0.00	
32	0.30	
34	0.02	
40	0.24	

	Disponibilité de protéines en quantité (g/personne/jour)	\
7	36.91	
12	0.00	
32	0.56	
34	0.08	
40	0.79	

	Disponibilité intérieure	Exportations - Quantité	\
7	5.992000e+09	0.0	

12	0.000000e+00	0.0
32	3.130000e+08	0.0
34	1.300000e+07	0.0
40	5.240000e+08	0.0

	Importations - Quantité	Nourriture	Pertes	Production \
7	1.173000e+09	4.895000e+09	775000000.0	5.169000e+09
12	0.000000e+00	0.000000e+00	0.0	0.000000e+00
32	1.000000e+06	7.600000e+07	31000000.0	3.120000e+08
34	0.000000e+00	1.200000e+07	1000000.0	1.300000e+07
40	1.000000e+07	8.900000e+07	52000000.0	5.140000e+08

	Semences	Traitement	Variation de stock
7	322000000.0	0.0	-350.0
12	0.0	0.0	0.0
32	5000000.0	0.0	0.0
34	0.0	0.0	0.0
40	22000000.0	0.0	0.0

```
[70]: #Affichage de la proportion d'alimentation animale
dispo_alimentaire_animales = dispo_alimentaire_cereales['Aliments pour_
↳animaux'].sum()*100/dispo_alimentaire_cereales['Disponibilité intérieure'].
↳sum()
print("La proportion d'alimentation animale sur la disponibilité totale est de :
↳", round(dispo_alimentaire_animales), "%")
```

La proportion d'alimentation animale sur la disponibilité totale est de : 33 %

```
[71]: #Affichage de la proportion d'alimentation humaine
dispo_alimentaire_nourriture = dispo_alimentaire_cereales['Nourriture'].
↳sum()*100/dispo_alimentaire_cereales['Disponibilité intérieure'].sum()
print("La proportion d'alimentation humaine sur la disponibilité totale est de :
↳", round(dispo_alimentaire_nourriture), "%")
```

La proportion d'alimentation humaine sur la disponibilité totale est de : 39 %

```
[72]: dispo_alimentaire_group_cereales = dispo_alimentaire_cereales.
↳groupby('Produit').sum().reset_index()
dispo_alimentaire_group_cereales.head(15)
```

	Produit	Zone \
0	Avoine	Afrique du SudAlbanieAlgérieAllemagneAngolaAnt...
1	Blé	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
2	Céréales, Autres	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
3	Maïs	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
4	Millet	AfghanistanAfrique du SudAlbanieAlgérieAllemag...
5	Orge	AfghanistanAfrique du SudAlbanieAlgérieAllemag...

6	Riz (Eq Blanchi)	Afghanistan	Afrique du Sud	Albanie	Algérie	Allemagne	...
7	Seigle	Afrique du Sud	Albanie	Algérie	Allemagne	Antigua-e...	...
8	Soja	Afrique du Sud	Albanie	Algérie	Allemagne	Angola	Ant...
9	Sorgho	Afrique du Sud	Albanie	Algérie	Allemagne	Angola	Ant...

	Origine	Aliments pour animaux \
0	vegetalevegetalevegetalevegetalevegetalevegeta...	1.625100e+10
1	vegetalevegetalevegetalevegetalevegetalevegeta...	1.296680e+11
2	vegetalevegetalevegetalevegetalevegetalevegeta...	1.903500e+10
3	vegetalevegetalevegetalevegetalevegetalevegeta...	5.461160e+11
4	vegetalevegetalevegetalevegetalevegetalevegeta...	3.306000e+09
5	vegetalevegetalevegetalevegetalevegetalevegeta...	9.265800e+10
6	vegetalevegetalevegetalevegetalevegetalevegeta...	3.359400e+10
7	vegetalevegetalevegetalevegetalevegetalevegeta...	8.099000e+09
8	vegetalevegetalevegetalevegetalevegetalevegeta...	1.747600e+10
9	vegetalevegetalevegetalevegetalevegetalevegeta...	2.480800e+10

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour) \
0	2.060000e+08	880.0
1	2.242400e+10	96132.0
2	1.388000e+09	1483.0
3	1.893860e+11	30903.0
4	1.960000e+08	3966.0
5	6.540000e+08	1814.0
6	1.846400e+10	53024.0
7	2.000000e+07	1967.0
8	6.380000e+08	1086.0
9	2.049000e+09	4952.0

	Disponibilité alimentaire en quantité (kg/personne/an) \
0	151.43
1	12339.51
2	193.74
3	3690.37
4	513.09
5	263.60
6	5384.46
7	263.36
8	127.12
9	614.02

	Disponibilité de matière grasse en quantité (g/personne/jour) \
0	15.29
1	548.76
2	5.46
3	271.50
4	37.27

5	7.35
6	123.95
7	8.62
8	47.72
9	48.34

	Disponibilité de protéines en quantité (g/personne/jour) \
0	35.70
1	2809.21
2	40.52
3	771.12
4	96.11
5	52.28
6	1027.25
7	51.56
8	109.71
9	146.16

	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité \
0	2.340700e+10	3.531000e+09	3.708000e+09
1	6.794980e+11	1.983800e+11	1.853770e+11
2	2.748500e+10	2.540000e+09	2.882000e+09
3	9.557990e+11	1.287870e+11	1.226910e+11
4	2.991100e+10	2.760000e+08	3.430000e+08
5	1.404390e+11	4.158500e+10	3.868900e+10
6	4.756560e+11	4.144500e+10	3.761800e+10
7	1.656700e+10	2.372000e+09	2.251000e+09
8	2.674450e+11	1.069120e+11	1.020820e+11
9	5.823700e+10	6.388000e+09	6.418000e+09

	Nourriture	Pertes	Production	Semences	Traitement \
0	3.903000e+09	5.140000e+08	2.381600e+10	2.485000e+09	3.600000e+07
1	4.578240e+11	2.753000e+10	7.084430e+11	3.428300e+10	7.877000e+09
2	5.324000e+09	7.850000e+08	2.841200e+10	1.203000e+09	1.600000e+08
3	1.251840e+11	4.070100e+10	1.017029e+12	6.837000e+09	4.764800e+10
4	2.304000e+10	2.276000e+09	2.890900e+10	6.820000e+08	4.030000e+08
5	6.794000e+09	4.628000e+09	1.424620e+11	8.804000e+09	2.691000e+10
6	3.772860e+11	2.737800e+10	4.954470e+11	1.219400e+10	6.740000e+09
7	5.502000e+09	5.620000e+08	1.671100e+10	1.262000e+09	1.120000e+09
8	1.064800e+10	3.970000e+09	2.783610e+11	7.401000e+09	2.273140e+11
9	2.415300e+10	2.746000e+09	6.211600e+10	7.880000e+08	3.695000e+09

	Variation de stock
0	-577.0
1	-15948.0
2	-1267.0
3	-55136.0

```

4          933.0
5          876.0
6        -15958.0
7          -25.0
8        -6082.0
9        -3913.0

```

```

[73]: animaux_nourriture = dispo_alimentaire_group_cereales[['Produit', 'Aliments_
    ↪pour animaux', 'Nourriture']]
    animaux_nourriture.head(10)

```

```

[73]:
      Produit  Aliments pour animaux  Nourriture
0      Avoine      1.625100e+10  3.903000e+09
1         Blé      1.296680e+11  4.578240e+11
2  Céréales, Autres      1.903500e+10  5.324000e+09
3         Maïs      5.461160e+11  1.251840e+11
4        Millet      3.306000e+09  2.304000e+10
5         Orge      9.265800e+10  6.794000e+09
6  Riz (Eq Blanchi)      3.359400e+10  3.772860e+11
7         Seigle      8.099000e+09  5.502000e+09
8         Soja      1.747600e+10  1.064800e+10
9        Sorgho      2.480800e+10  2.415300e+10

```

```

[74]: import warnings
    warnings.filterwarnings('ignore')
    total_cereales = animaux_nourriture['Aliments pour animaux'].sum() +
    ↪animaux_nourriture['Nourriture'].sum()
    animaux_nourriture['Proportion Aliments pour animaux'] =
    ↪round((animaux_nourriture['Aliments pour animaux']*100)/total_cereales, 2)
    animaux_nourriture['Proportion Nourriture'] =
    ↪round((animaux_nourriture['Nourriture']*100)/total_cereales, 2)
    dispo_alimentaire_population['Disponibilité annuelle'] =
    ↪(dispo_alimentaire_population['Disponibilité alimentaire (Kcal/personne/
    ↪jour)'] * dispo_alimentaire_population['Population']*365)/1000
    animaux_nourriture[['Produit', 'Proportion Aliments pour animaux', 'Proportion_
    ↪Nourriture']].head(10)

```

```

[74]:
      Produit  Proportion Aliments pour animaux  Proportion Nourriture
0      Avoine                0.84                0.20
1         Blé                6.72               23.71
2  Céréales, Autres                0.99                0.28
3         Maïs               28.29                6.48
4        Millet                0.17                1.19
5         Orge                4.80                0.35
6  Riz (Eq Blanchi)                1.74               19.54
7         Seigle                0.42                0.28
8         Soja                0.91                0.55

```

```
[75]: animaux_nourriture['Proportion Aliments pour animaux'].sum()
```

```
[75]: 46.16
```

```
[76]: animaux_nourriture['Proportion Nourriture'].sum()
```

```
[76]: 53.83
```

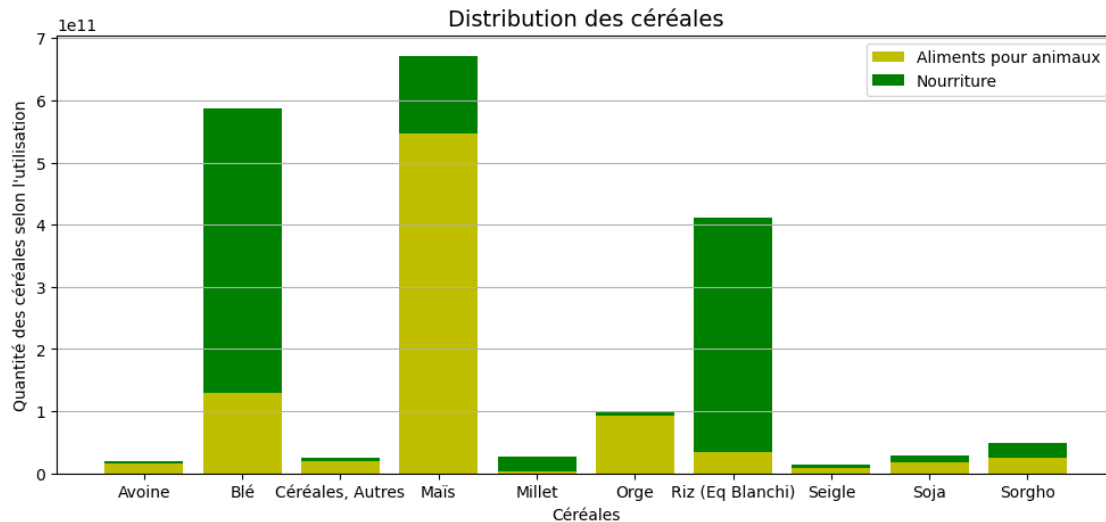
```
[77]: import matplotlib.pyplot as plt

# Données d'exemple
produits = animaux_nourriture['Produit']
aliments_pour_animaux = animaux_nourriture['Aliments pour animaux']
nourriture = animaux_nourriture['Nourriture']

# Créer le graphique à barres empilées
plt.figure(figsize=(12,5))
r = range(len(produits))
barWidth = 0.8
plt.bar(r, aliments_pour_animaux, color='y', width=barWidth, label='Aliments_
↳pour animaux')
plt.bar(r, nourriture, bottom=aliments_pour_animaux, color='g', width=barWidth,
↳label='Nourriture')
plt.xticks(r, produits)
plt.ylabel('Quantité des céréales selon l\'utilisation')
plt.grid(axis='y')
plt.xlabel('Céréales')
plt.title('Distribution des céréales', fontsize=14)
plt.legend(loc='upper right')

# Ajouter des étiquettes aux barres
#for i in range(len(produits)):
#    plt.text(i, aliments_pour_animaux[i]/2, aliments_pour_animaux[i],
↳ha='center', va='center')
#    plt.text(i, aliments_pour_animaux[i]+nourriture[i]/2, nourriture[i],
↳ha='center', va='center')

plt.show()
```



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

```
[78]: #Création de la colonne proportion par pays
proportion_sous_nutrition['Proportion sous nutrition'] =
    round((proportion_sous_nutrition['sous_nutrition']*100)/
    proportion_sous_nutrition['Population'],2)
proportion_sous_nutrition.head()
```

```
[78]:
```

	Zone	Année	Population	sous_nutrition \
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

```
Proportion sous nutrition
0      28.93
1       5.44
2       3.47
3       3.14
4       0.00
```

```
[79]: #affichage après trie des 10 pires pays
les_dix_pays_proportion_sous_nutrition = proportion_sous_nutrition.
    sort_values(by = 'Proportion sous nutrition', ascending = False).head(10).
    reset_index()
les_dix_pays_proportion_sous_nutrition.head(10)
```



```
[79]:
```

	index		Zone	Année	Population	\
0	78		Haïti	2017	10982366.0	
1	157	République populaire démocratique de Corée		2017	25429825.0	
2	108		Madagascar	2017	25570512.0	
3	103		Libéria	2017	4702226.0	
4	100		Lesotho	2017	2091534.0	
5	183		Tchad	2017	15016753.0	
6	161		Rwanda	2017	11980961.0	
7	121		Mozambique	2017	28649018.0	
8	186		Timor-Leste	2017	1243258.0	
9	0		Afghanistan	2017	36296113.0	

	sous_nutrition	Proportion sous nutrition
0	5300000.0	48.26
1	12000000.0	47.19
2	10500000.0	41.06
3	1800000.0	38.28
4	800000.0	38.25
5	5700000.0	37.96
6	4200000.0	35.06
7	9400000.0	32.81
8	400000.0	32.17
9	10500000.0	28.93

```
[110]: les_dix_pays_proportion_sous_nutrition.loc[1, 'Zone'] = 'RPD Corée'
les_dix_pays_proportion_sous_nutrition.head(10)
```

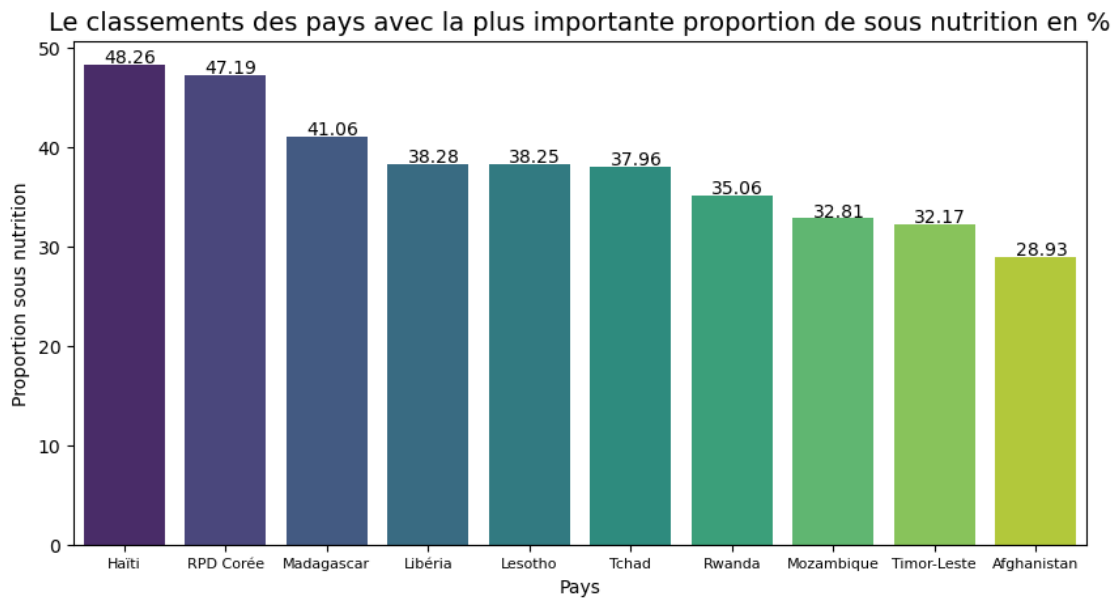
```
[110]:
```

	index		Zone	Année	Population	sous_nutrition	\
0	78		Haïti	2017	10982366.0	5300000.0	
1	157	RPD Corée		2017	25429825.0	12000000.0	
2	108	Madagascar		2017	25570512.0	10500000.0	
3	103	Libéria		2017	4702226.0	1800000.0	
4	100	Lesotho		2017	2091534.0	800000.0	
5	183	Tchad		2017	15016753.0	5700000.0	
6	161	Rwanda		2017	11980961.0	4200000.0	
7	121	Mozambique		2017	28649018.0	9400000.0	
8	186	Timor-Leste		2017	1243258.0	400000.0	
9	0	Afghanistan		2017	36296113.0	10500000.0	

	Proportion sous nutrition
0	48.26
1	47.19
2	41.06
3	38.28
4	38.25
5	37.96
6	35.06

7	32.81
8	32.17
9	28.93

```
[141]: plt.figure(figsize=(10,5))
sns.barplot(data=les_dix_pays_proportion_sous_nutrition, x='Zone',
            y='Proportion sous nutrition', palette="viridis")
plt.xlabel("Pays")
plt.xticks(fontsize=8)
plt.title("Le classements des pays avec la plus importante proportion de sous_
            nutrition en %", fontsize=14)
va = les_dix_pays_proportion_sous_nutrition['Proportion sous nutrition'].
            tolist()
for i in range(10):
    plt.text(i-0.2, va[i]+0.20, va[i])
plt.show()
```



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

```
[82]: #calcul du total de l'aide alimentaire par pays
aide_alimentaire_group = aide_alimentaire.groupby('Zone').sum().reset_index()
aide_alimentaire_group[['Zone', 'Valeur']].head()
```

```
[82]:
```

	Zone	Valeur
0	Afghanistan	185452000
1	Algérie	81114000
2	Angola	5014000

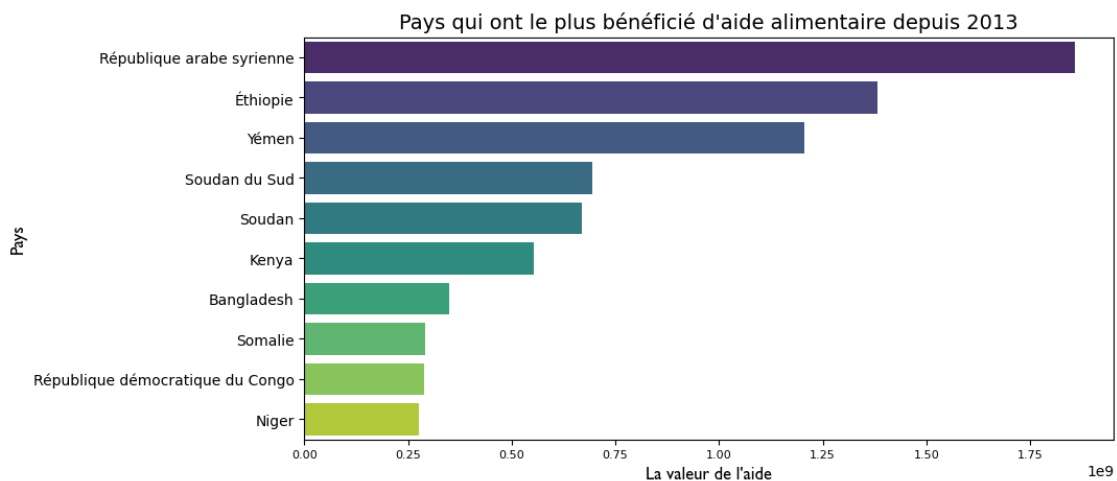
```
3 Bangladesh 348188000
4 Bhoutan 2666000
```

```
[83]: #affichage après trie des 10 pays qui ont bénéficié le plus de l'aide_
      ↪ alimentaire
dix_pays_aide_alimentaire = aide_alimentaire_group[['Zone', 'Valeur']].
      ↪ sort_values(by = 'Valeur', ascending = False).reset_index().head(10)
dix_pays_graphique = dix_pays_aide_alimentaire[['Zone', 'Valeur']]
dix_pays_graphique.head(10)
```

```
[83]:
```

	Zone	Valeur
0	République arabe syrienne	1858943000
1	Éthiopie	1381294000
2	Yémen	1206484000
3	Soudan du Sud	695248000
4	Soudan	669784000
5	Kenya	552836000
6	Bangladesh	348188000
7	Somalie	292678000
8	République démocratique du Congo	288502000
9	Niger	276344000

```
[84]: plt.figure(figsize=(10,5))
sns.barplot(data=dix_pays_graphique, x='Valeur', y='Zone', palette="viridis")
plt.xlabel("La valeur de l'aide", fontname='Gill Sans MT', fontsize=12)
plt.ylabel("Pays", fontname='Gill Sans MT', fontsize=12)
plt.xticks(fontsize=8)
plt.title("Pays qui ont le plus bénéficié d'aide alimentaire depuis 2013",
      ↪ fontsize=14)
plt.show()
```



```
[85]: aide_alimentaire.head()
```

```
[85]:
```

	Zone	Année	Produit	Valeur
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

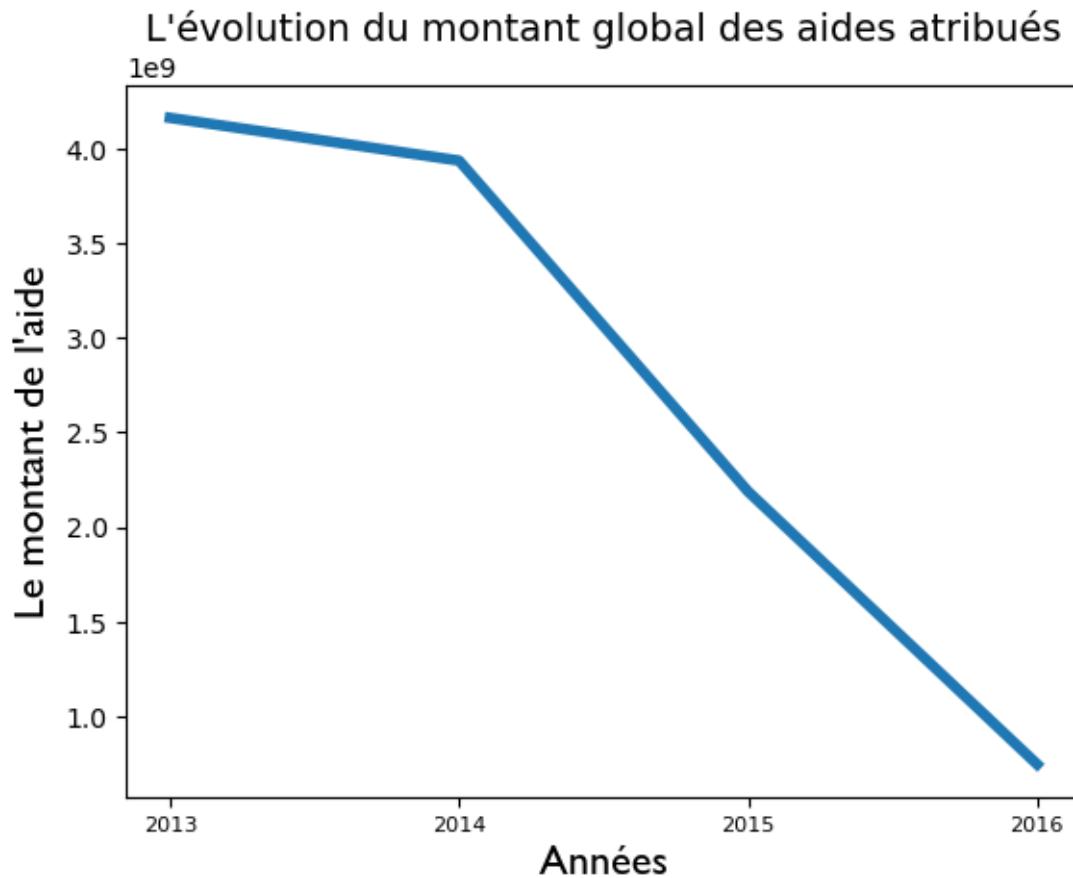
```
[86]: aide_alimentaire_group_année = aide_alimentaire.groupby('Année').sum().
      ↪reset_index()
      aide_alimentaire_group_année.head()
```

```
[86]:
```

	Année	Zone \
0	2013	AfghanistanAfghanistanAfghanistanAfghanistanAf...
1	2014	AfghanistanAfghanistanAfghanistanAfghanistanAf...
2	2015	AlgérieAlgérieAlgérieAlgérieAlgérieAlgérieAlgé...
3	2016	AlgérieAlgérieAlgérieAlgérieAlgérieAlgérieAlgé...

	Produit	Valeur
0	Autres non-céréalesBlé et FarinCéréalesFruits ...	4165674000
1	Autres non-céréalesBlé et FarinCéréalesHuiles ...	3939152000
2	Autres non-céréalesCéréalesCéréales Secondaire...	2187507000
3	CéréalesCéréales SecondairesFruits secs, total...	743568000

```
[144]: plt.plot(aide_alimentaire_group_année['Année'],
      ↪aide_alimentaire_group_année['Valeur'], linewidth=4)
      plt.xlabel("Années", fontname='Gill Sans MT', fontsize=16)
      plt.ylabel("Le montant de l'aide", fontname='Gill Sans MT', fontsize=16)
      plt.xticks(aide_alimentaire_group_année['Année'].astype(int), fontsize=8)
      plt.title("L'évolution du montant global des aides attribués", fontsize=14)
      plt.show()
```



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

```
[88]: #Création d'un dataframe avec la zone, l'année et l'aide alimentaire puis
      ↪groupby sur zone et année
aide_alimentaire_ZAV = aide_alimentaire[['Zone', 'Année', 'Valeur']].copy()
aide_alimentaire_ZAV.head()
```

```
[88]:
```

	Zone	Année	Valeur
0	Afghanistan	2013	682000
1	Afghanistan	2014	335000
2	Afghanistan	2013	39224000
3	Afghanistan	2014	15160000
4	Afghanistan	2013	40504000

```
[89]: aide_alimentaire_group = aide_alimentaire_ZAV.groupby(['Zone', 'Année']).sum().
      ↪reset_index()
aide_alimentaire_group.head()
```

```
[89]:
```

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

```
[90]: #Création d'une liste contenant les 5 pays qui ont le plus bénéficiées de
      ↪ l'aide alimentaire
liste = ['République arabe syrienne', 'Éthiopie', 'Yémen', 'Soudan du Sud',
      ↪ 'Soudan']
```

```
[91]: #On filtre sur le dataframe avec notre liste
aide_alim_graph = aide_alimentaire_group.loc[aide_alimentaire_group['Zone']
      ↪ .isin(liste),:]
```

```
[92]: # Affichage des pays avec l'aide alimentaire par année
aide_alim_graph.head(20)
```

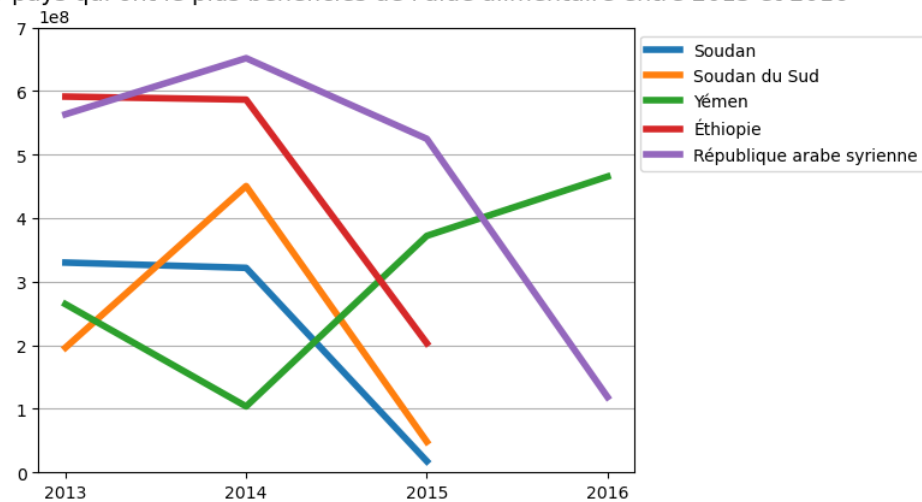
```
[92]:
```

	Zone	Année	Valeur
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	176500000
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

```
[93]: Soudan = aide_alim_graph.loc[aide_alim_graph['Zone'] == 'Soudan', :]
SoudanSud = aide_alim_graph.loc[aide_alim_graph['Zone'] == 'Soudan du Sud', :]
Yemen = aide_alim_graph.loc[aide_alim_graph['Zone'] == 'Yémen', :]
Etiophie = aide_alim_graph.loc[aide_alim_graph['Zone'] == 'Éthiopie', :]
Syrie = aide_alim_graph.loc[aide_alim_graph['Zone'] == 'République arabe
      ↪ syrienne', :]
plt.plot(Soudan['Année'], Soudan['Valeur'], label='Soudan', linewidth=4)
plt.plot(SoudanSud['Année'], SoudanSud['Valeur'], label='Soudan du Sud',
      ↪ linewidth=4)
```

```
plt.plot(Yemen['Année'], Yemen['Valeur'], label = 'Yémen', linewidth=4)
plt.plot(Etiopie['Année'], Etiopie['Valeur'], label = 'Éthiopie', linewidth=4)
plt.plot(Syrie['Année'], Syrie['Valeur'], label = 'République arabe syrienne',
↪linewidth=4)
plt.title("Evolution des 5 pays qui ont le plus bénéficiés de l'aide
↪alimentaire entre 2013 et 2016", fontsize=14)
plt.legend(loc='best', bbox_to_anchor=(1, 0.5, 0.5, 0.5))
plt.grid(axis='y')
plt.ylim(0, 7000000000)
plt.xticks([2013, 2014, 2015, 2016])
plt.show()
```

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



3.9 - Pays avec le moins de disponibilité par habitant

```
[94]: #Calcul de la disponibilité en kcal par personne par jour par pays
dispo_alimentaire_population.head()
```

```
[94]:
```

	Zone	Produit \
0	Afghanistan	Abats ComestibleAgrumes, AutresAliments pour e...
1	Afrique du Sud	Abats ComestibleAgrumes, AutresAlcool, non Com...
2	Albanie	Abats ComestibleAgrumes, AutresAlcool, non Com...
3	Algérie	Abats ComestibleAgrumes, AutresAlcool, non Com...
4	Allemagne	Abats ComestibleAgrumes, AutresAlcool, non Com...

	Origine	Aliments pour animaux \
0	animalevegetalevegetalevegetalevegetaleanimale...	7.680000e+08
1	animalevegetalevegetalevegetalevegetaleanimale...	5.309000e+09
2	animalevegetalevegetalevegetalevegetaleanimale...	6.600000e+08
3	animalevegetalevegetalevegetalevegetaleanimale...	4.352000e+09

4 animalevegetalevegetalevegetalevegetaleanimale... 3.020900e+10

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	\
0	4.150000e+08	2087.0	
1	8.760000e+08	3020.0	
2	1.740000e+08	3188.0	
3	1.707000e+09	3293.0	
4	7.711000e+09	3503.0	

	Disponibilité alimentaire en quantité (kg/personne/an)	\
0	351.41	
1	564.80	
2	1095.54	
3	809.13	
4	957.52	

	Disponibilité de matière grasse en quantité (g/personne/jour)	\
0	33.50	
1	83.58	
2	106.48	
3	76.48	
4	141.62	

	Disponibilité de protéines en quantité (g/personne/jour)	\
0	58.26	
1	85.33	
2	111.37	
3	91.92	
4	101.39	

	Disponibilité intérieure	Exportations - Quantité	Importations - Quantité	\
0	1.351500e+10	2.780000e+08	2.988000e+09	
1	6.125600e+10	1.167400e+10	7.517000e+09	
2	4.758000e+09	9.400000e+07	9.330000e+08	
3	4.263000e+10	6.550000e+08	1.740300e+10	
4	1.622750e+11	5.743100e+10	6.535300e+10	

	Nourriture	Pertes	Production	Semences	Traitement	\
0	1.073500e+10	1.135000e+09	1.117100e+10	3.950000e+08	6.300000e+07	
1	2.981200e+10	2.193000e+09	6.326300e+10	2.530000e+08	2.281900e+10	
2	3.476000e+09	2.760000e+08	3.964000e+09	5.500000e+07	1.130000e+08	
3	3.172900e+10	3.753000e+09	2.635900e+10	4.490000e+08	6.370000e+08	
4	7.923800e+10	3.781000e+09	1.545470e+11	1.551000e+09	3.984200e+10	

	Variation de stock	Population	Disponibilité annuelle
0	-368.0	36296113.0	2.764875e+10
1	2153.0	57009756.0	6.284185e+10

2	-45.0	2884169.0	3.356077e+09
3	-479.0	41389189.0	4.974753e+10
4	-192.0	82658409.0	1.056866e+11

```
[95]: #Affichage des 10 pays qui ont le moins de dispo alimentaire par personne
liste_pays_moins_dispo_alim = dispo_alimentaire_population.sort_values(by =
↳ 'Disponibilité alimentaire (Kcal/personne/jour)').head(10)
```

```
[96]: liste_pays_moins_dispo_alim[['Zone', 'Disponibilité alimentaire (Kcal/personne/
↳ jour)']].head(10)
```

```
[96]:
```

	Zone \	
128	République centrafricaine	
166	Zambie	
91	Madagascar	
0	Afghanistan	
65	Haïti	
133	République populaire démocratique de Corée	
151	Tchad	
167	Zimbabwe	
114	Ouganda	
154	Timor-Leste	

	Disponibilité alimentaire (Kcal/personne/jour)
128	1879.0
166	1924.0
91	2056.0
0	2087.0
65	2089.0
133	2093.0
151	2109.0
167	2113.0
114	2126.0
154	2129.0

3.10 - Pays avec le plus de disponibilité par habitant

```
[97]: #Affichage des 10 pays qui ont le plus de dispo alimentaire par personne
liste_pays_plus_dispo_alim = dispo_alimentaire_population.sort_values(by =
↳ 'Disponibilité alimentaire (Kcal/personne/jour)', ascending = False).head(10)
```

```
[98]: liste_pays_plus_dispo_alim[['Zone', 'Disponibilité alimentaire (Kcal/personne/
↳ jour)']].head(10)
```

```
[98]:
```

	Zone	Disponibilité alimentaire (Kcal/personne/jour)
11	Autriche	3770.0
16	Belgique	3737.0

159	Turquie	3708.0
171	États-Unis d'Amérique	3682.0
74	Israël	3610.0
72	Irlande	3602.0
75	Italie	3578.0
89	Luxembourg	3540.0
168	Égypte	3518.0
4	Allemagne	3503.0

3.11 - Exemple de la Thaïlande pour le Manioc

```
[99]: #création d'un dataframe avec uniquement la Thaïlande
dispo_alimentaire_group_Thaïlande = dispo_alimentaire1.
↳ loc[dispo_alimentaire1['Zone'] == 'Thaïlande', :]
dispo_alimentaire_group_Thaïlande.head()
```

```
[99]:
```

	Zone	Produit	Origine	Aliments pour animaux	\
13759	Thaïlande	Abats Comestible	animale	0.0	
13760	Thaïlande	Agrumes, Autres	vegetale	0.0	
13761	Thaïlande	Alcool, non Comestible	vegetale	0.0	
13762	Thaïlande	Aliments pour enfants	vegetale	0.0	
13763	Thaïlande	Ananas	vegetale	0.0	

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	\
13759	0.0	3.0	
13760	0.0	0.0	
13761	358000000.0	0.0	
13762	0.0	2.0	
13763	0.0	10.0	

	Disponibilité alimentaire en quantité (kg/personne/an)	\
13759	1.11	
13760	0.09	
13761	0.00	
13762	0.18	
13763	10.02	

	Disponibilité de matière grasse en quantité (g/personne/jour)	\
13759	0.09	
13760	0.00	
13761	0.00	
13762	0.01	
13763	0.04	

	Disponibilité de protéines en quantité (g/personne/jour)	\
13759	0.56	
13760	0.00	

13761	0.00
13762	0.08
13763	0.08

	Disponibilité intérieure	Exportations - Quantité \
13759	74000000.0	5.000000e+06
13760	8000000.0	6.000000e+06
13761	358000000.0	1.100000e+08
13762	12000000.0	7.000000e+06
13763	782000000.0	1.449000e+09

	Importations - Quantité	Nourriture	Pertes	Production \
13759	33000000.0	75000000.0	0.0	4.500000e+07
13760	2000000.0	6000000.0	0.0	1.200000e+07
13761	21000000.0	0.0	0.0	4.470000e+08
13762	19000000.0	12000000.0	0.0	0.000000e+00
13763	9000000.0	671000000.0	110000000.0	2.209000e+09

	Semences	Traitement	Variation de stock
13759	0.0	0.0	0.0
13760	0.0	2000000.0	0.0
13761	0.0	0.0	0.0
13762	0.0	0.0	0.0
13763	0.0	0.0	13.0

```
[100]: dispo_alimentaire_Thaïlande = dispo_alimentaire_group_Thaïlande.
        ↳groupby(["Zone"]).sum()
        dispo_alimentaire_Thaïlande.head()
```

```
[100]:
```

	Produit \
--	-----------

Zone

Thaïlande Abats ComestibleAgrumes, AutresAlcool, non Com...

Origine \

Zone

Thaïlande animalevegetalevegetalevegetalevegetaleanimale...

	Aliments pour animaux	Autres Utilisations \
--	-----------------------	-----------------------

Zone

Thaïlande	9.561000e+09	4.210000e+09
-----------	--------------	--------------

	Disponibilité alimentaire (Kcal/personne/jour) \
--	--

Zone

Thaïlande	2785.0
-----------	--------

	Disponibilité alimentaire en quantité (kg/personne/an) \
--	--

Zone

Thaïlande	586.23
Disponibilité de matière grasse en quantité (g/personne/jour) \	
Zone	
Thaïlande	58.57
Disponibilité de protéines en quantité (g/personne/jour) \	
Zone	
Thaïlande	60.85
Disponibilité intérieure Exportations - Quantité \	
Zone	
Thaïlande	1.581300e+11 5.043000e+10
Importations - Quantité Nourriture Pertes Production \	
Zone	
Thaïlande	1.133500e+10 3.928300e+10 5.749000e+09 2.017640e+11
Semences Traitement Variation de stock	
Zone	
Thaïlande	823000000.0 9.917000e+10 -4534.0

```
[101]: dispo_alimentaire_group_Thaïlande_manioc = dispo_alimentaire_group_Thaïlande.
        loc[dispo_alimentaire_group_Thaïlande['Produit'] == 'Manioc', :]
        dispo_alimentaire_group_Thaïlande_manioc.head()
```

```
[101]:
```

	Zone	Produit	Origine	Aliments pour animaux	\
13809	Thaïlande	Manioc	vegetale	1.800000e+09	
Autres Utilisations Disponibilité alimentaire (Kcal/personne/jour) \					
13809		2.081000e+09			40.0
Disponibilité alimentaire en quantité (kg/personne/an) \					
13809					13.0
Disponibilité de matière grasse en quantité (g/personne/jour) \					
13809					0.05
Disponibilité de protéines en quantité (g/personne/jour) \					
13809					0.14
Disponibilité intérieure Exportations - Quantité \					
13809		6.264000e+09		2.521400e+10	
Importations - Quantité Nourriture Pertes Production \					
13809		1.250000e+09	871000000.0	1.511000e+09	3.022800e+10

	Semences	Traitement	Variation de stock
13809	0.0	0.0	0.0

```
[102]: #Calcul de la sous nutrition en Thaïlande
proportion_sous_nutrition_Thaïlande = proportion_sous_nutrition.
↳loc[proportion_sous_nutrition['Zone'] == 'Thaïlande', :]
proportion_sous_nutrition_Thaïlande.head()
```

```
[102]:
```

	Zone	Année	Population	sous_nutrition	Proportion sous nutrition
185	Thaïlande	2017	69209810.0	6200000.0	8.96

```
[103]: # On calcule la proportion exportée en fonction de la proportion
manioc_proportion_export=round(dispo_alimentaire_group_Thaïlande_manioc['Exportations_
↳- Quantité']*100/dispo_alimentaire_group_Thaïlande_manioc['Production']).
↳item()
print("Le proportion exporté est de {} % de la production de Manioc. ").
↳format(round(manioc_proportion_export)))
```

Le proportion exporté est de 83 % de la production de Manioc.

Etape 6 - Analyse Complémentaires République Centrafricaine

```
[104]: #Rajouter 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"
```

```
[105]: #afficher la sou-nutrition en République centrafricaine
proportion_sous_nutrition.loc[proportion_sous_nutrition['Zone'] == 'République_
↳centrafricaine', :].head()
```

```
[105]:
```

	Zone	Année	Population	sous_nutrition \
151	République centrafricaine	2017	4596023.0	0.0

	Proportion sous nutrition
151	0.0

```
[106]: #afficher la Disponibilité alimentaire en République centrafricaine
dispo_alimentaire_group.loc[dispo_alimentaire_group['Zone'] == 'République_
↳centrafricaine', :]
```

```
[106]:
```

	Zone	Origine \
256	République centrafricaine	animale
257	République centrafricaine	vegetale

	Produit	Aliments pour animaux \
256	Abats ComestibleBeurre, GheeCephalopodesCrusta...	0.0

257	Alcool, non Comestible	Aliments pour enfants	Ana...	1000000.0
-----	------------------------	-----------------------	--------	-----------

	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)	\
256		1000000.0	206.0
257		26000000.0	1673.0

	Disponibilité alimentaire en quantité (kg/personne/an)	\
256		55.41
257		398.32

	Disponibilité de matière grasse en quantité (g/personne/jour)	\
256		13.94
257		45.47

	Disponibilité de protéines en quantité (g/personne/jour)	\
256		17.12
257		28.92

	Disponibilité intérieure	Exportations - Quantité	\
256		2.610000e+08	0.0
257		2.321000e+09	0.0

	Importations - Quantité	Nourriture	Pertes	Production	\
256		12000000.0	2.550000e+08	3000000.0	2.490000e+08
257		64000000.0	1.837000e+09	119000000.0	2.254000e+09

	Semences	Traitement	Variation de stock
256	1000000.0	0.0	0.0
257	62000000.0	278000000.0	3.0

```
[107]: # afficher l'aide et la disponibilité alimentaire
liste_pays_moins_dispo_alim[['Zone', 'Disponibilité alimentaire (Kcal/personne/
↪jour)', 'Disponibilité intérieure', 'Exportations - Quantité', 'Importations_
↪Quantité']].head(1)
```

```
[107]: Zone \
128 République centrafricaine

Disponibilité alimentaire (Kcal/personne/jour) Disponibilité intérieure \
128 1879.0 2.582000e+09

Exportations - Quantité Importations - Quantité
128 0.0 76000000.0
```

```
[108]: #aide alimentaire par année
aide_alimentaire_group = aide_alimentaire.groupby(['Zone', 'Année']).sum().
↪reset_index()
```

```

aide_alimentaire_group.head()
aide_alimentaire_centrafrique = aide_alimentaire_group.
↳loc[aide_alimentaire_group['Zone'] == 'République centrafricaine', :]
aide_alimentaire_centrafrique.head()

```

```

[108]:
           Zone  Année \
161  République centrafricaine  2013
162  République centrafricaine  2014
163  République centrafricaine  2015

           Produit  Valeur
161  Autres non-céréalesCéréalesCéréales Secondaire...  17156000
162  Autres non-céréalesCéréalesCéréales Secondaire...  27418000
163  Autres non-céréalesCéréalesCéréales Secondaire...  22036000

```

```

[109]: #aide alimentaire total
aide_alimentaire_centrafricaine = aide_alimentaire_centrafrique['Valeur'].sum()
print("L'aide alimentaire total reçu par la République centrafricaine entre_
↳2013 et 2015 est de ", aide_alimentaire_centrafricaine, "€")

```

L'aide alimentaire total reçu par la République centrafricaine entre 2013 et 2015 est de 66610000 €