BENMRAD_Justine_2_notebook-pdf_012024

March 6, 2024

1.1 - Importation des librairies

[1]:
```
#Importation de la librairie Pandas
import pandas as pd
```

1.2 - Chargement des fichiers Excel

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

[3]:
```
#Importation du fichier dispo_alimentaire.csv
dispo_alimentaire = pd.read_csv('dispo_alimentaire.csv')
```

[4]:
```
#Importation du fichier aide_alimentaire.csv
aide_alimentaire = pd.read_csv('aide_alimentaire.csv')
```

[5]:
```
#Importation du fichier sous_nutrition.csv
sous_nutrition = pd.read_csv('sous_nutrition.csv')
```

2.1 - Analyse exploratoire du fichier population

[6]:
```
# Affichage des dimensions du dataset du fichier population
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)
```

[7]:
```
# Consultation du nombre de colonnes du fichier population
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
```

[8]:
```
# Affichage de la nature des données dans chacune des colonnes
population.dtypes
```

[8]:
```
Zone       object
Année       int64
Valeur    float64
dtype: object
```

[9]:
```
# Affichage des 5 premières lignes de la table
display(population.head())
```

	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

[10]:
```
# Harmonisation des unités ; multiplication la population par 1000

# Multiplication de la colonne Valeur par 1000
population['Valeur'] = population['Valeur']*1000
```

[11]:
```
# Changement du nom de la colonne Valeur par Population
population.rename(columns={'Valeur': 'Population'}, inplace=True)
print(population.columns)
```

```
Index(['Zone', 'Année', 'Population'], dtype='object')
```

[12]:
```
# Affichage des 5 premières lignes de la table pour voir les modifications
display(population.head())
```

	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

2.2 - Analyse exploratoire du fichier disponibilité alimentaire

[13]:
```
# Affichage des dimensions du dataset dispo_alimentaire
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]:
```
# Consultation du 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 des 5 premières lignes de la table
display(dispo_alimentaire.head())
```

```
          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]:
```
# Remplacement des NaN dans le dataset par des 0
dispo_alimentaire.fillna(0, inplace=True)
```

[17]:
```
# Multiplication de toutes les lignes contenant des milliers de tonnes, en Kg

# Définition des lignes à multiplier
colonnes_tonnes = ['Aliments pour animaux', 'Autres Utilisations',␣
 ↪'Disponibilité intérieure', 'Exportations - Quantité', 'Importations -␣
 ↪Quantité', 'Nourriture', 'Pertes', 'Production', 'Semences', 'Traitement',␣
 ↪'Variation de stock']
# Multiplication des lignes * 1 000 000
dispo_alimentaire.loc[:, colonnes_tonnes] = dispo_alimentaire.loc[:,␣
 ↪colonnes_tonnes] * 1000000
```

[18]:
```
# Affichage des 5 premières lignes de la table
display(dispo_alimentaire.head())
```

```
          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

[19]:
```
# Affichage des dimensions du dataset aide_alimentaire
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)
```

[20]:
```
# Consultation du 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
```

[21]:
```
# Affichage des 5 premières lignes de la table
display(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
4	Afghanistan	2013	Céréales	40504

[22]:
```
# Changement du nom de la colonne Pays bénéficiaire par Zone
aide_alimentaire.rename(columns={'Pays bénéficiaire': 'Zone'}, inplace=True)
print(aide_alimentaire.columns)
```

```
Index(['Zone', 'Année', 'Produit', 'Valeur'], dtype='object')
```

[23]:
```
# Multiplication de la colonne Valeur qui contient des tonnes par 1 000 pour␣
↪avoir des kg
aide_alimentaire.loc[:, 'Valeur'] = aide_alimentaire.loc[:, 'Valeur'] * 1000
```

[24]:
```
#Affichage les 5 premières lignes de la table
display(aide_alimentaire.head())
```

	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

[25]:
```
# Affichage des dimensions du dataset sous_nutrition
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)
```

[26]:
```
# Consultation du 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
```

[27]:
```
# Affichage des 5 premières lignes de la table
display(sous_nutrition.head())
```

```
          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
```

[28]:
```
# Conversion de la colonne sous nutrition en numérique

# Remplacement de la valeur <0.1 par 0.001 pour avoir un format convertissable
sous_nutrition['Valeur'].replace('<0.1', 0.001, inplace=True)
# Conversion en numérique
sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'])
```

[29]:
```
# Autre possibilité de conversion en numérique :

# Conversion de la colonne (avec l'argument errors=coerce qui permet de␣
 ↪convertir automatiquement les lignes qui ne sont pas des nombres en NaN)
#sous_nutrition['Valeur'] = pd.to_numeric(sous_nutrition['Valeur'],␣
 ↪errors='coerce')

# Remplacement des NaN en 0
#sous_nutrition.fillna(0, inplace=True)
```

[30]:
```
# Changement du nom de la colonne Valeur par SousNutrition
sous_nutrition.rename(columns={'Valeur': 'SousNutrition'}, inplace=True)
print(sous_nutrition.columns)
```

```
Index(['Zone', 'Année', 'SousNutrition'], dtype='object')
```

[31]:
```
# Multiplication de la colonne SousNutrition par 1 000 000
sous_nutrition['SousNutrition'] = sous_nutrition['SousNutrition']*1000000
```

[32]:
```
# Affichage des 5 premières lignes de la table
display(sous_nutrition.head())
```

```
          Zone      Année  SousNutrition
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
```

3.1 - Proportion de personnes en sous nutrition

[33]:
```
# Réalisation d'une jointure entre la table population et la table sous␣
 ↪nutrition, en ciblant l'année 2017

# Remplacement des tranches d'années par l'année représentative
sous_nutrition['Année'].replace('2012-2014', 2013, inplace=True)
sous_nutrition['Année'].replace('2013-2015', 2014, inplace=True)
sous_nutrition['Année'].replace('2014-2016', 2015, inplace=True)
sous_nutrition['Année'].replace('2015-2017', 2016, inplace=True)
sous_nutrition['Année'].replace('2016-2018', 2017, inplace=True)
sous_nutrition['Année'].replace('2017-2019', 2018, inplace=True)

# Création de la jointure entre population et sous_nutrition
pop_sous_nut = pd.merge(population,sous_nutrition)

# Création d'un DF pop_sous_nut_2017 avec uniquement les données pour l'année␣
 ↪2017
pop_sous_nut_2017 = pop_sous_nut[pop_sous_nut['Année'] == 2017]

# Affichage du DF pop_sous_nut_2017
display(pop_sous_nut_2017.head())
```

	Zone	Année	Population	SousNutrition
4	Afghanistan	2017	36296113.0	10500000.0
10	Afrique du Sud	2017	57009756.0	3100000.0
16	Albanie	2017	2884169.0	100000.0
22	Algérie	2017	41389189.0	1300000.0
28	Allemagne	2017	82658409.0	NaN

[34]:
```
print("Le tableau comporte {} observation(s) ou article(s)".
 ↪format(pop_sous_nut_2017.shape[0]))
print("Le tableau comporte {} colonne(s)".format(pop_sous_nut_2017.shape[1]))
```

```
Le tableau comporte 203 observation(s) ou article(s)
Le tableau comporte 4 colonne(s)
```

[35]:
```
# Calcul et affichage du nombre de personnes en état de sous nutrition

# Calcul de la population mondiale totale = somme Population
total_population = pop_sous_nut_2017['Population'].sum()

# Calcul de la population mondiale en sous nutrition totale = somme␣
 ↪SousNutrition
total_sous_nutrition = pop_sous_nut_2017['SousNutrition'].sum()
```

```
# Calcul de la part de personnes en état de sous nutrition dans la population␣
 ↪mondiale = SousNutriton / Population
pourcentage_sous_nutrition = (total_sous_nutrition / total_population) * 100

# Affichage des résultats
print('La population totale en 2017 est de', total_population)
print('Le nombre de personnes en état sous nutrition en 2017 est de',␣
 ↪total_sous_nutrition)
print("Le pourcentage de personnes en état sous-nutrition par rapport à la␣
 ↪population totale en 2017 est de:", pourcentage_sous_nutrition, "%")
```

```
La population totale en 2017 est de 7543798779.0
Le nombre de personnes en état sous nutrition en 2017 est de 535720000.0
Le pourcentage de personnes en état sous-nutrition par rapport à la population
totale en 2017 est de: 7.101461951653682 %
```

3.2 - Nombre théorique de personne qui pourrait être nourries

[36]:
```
# Selon l'OMS, un être humain mange en moyenne 2 000 calories par jour (et ~␣
 ↪900 kg / an / pers)
```

[37]:
```
# Réalisation d'une jointure entre population et dispo_alimentaire, pour␣
 ↪ajouter la colonne Population dans dispo_alimentaire

# Création du DF population_2017 avec uniquement les données 2017
population_2017 = population.loc[population['Année'] == 2017][['Zone',␣
 ↪'Population']]

# Création de la jointure
dispo_ali_pop_2017 = pd.merge(dispo_alimentaire,population_2017)
```

[38]:
```
# Affichage du nouveau DF
display(dispo_ali_pop_2017.head())
```

	Zone	Produit	Origine	Aliments pour animaux	Autres Utilisations	Disponibilité alimentaire (Kcal/personne/jour)
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

```
   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  \
0  53000000.0        0.0  53000000.0       0.0         0.0
1  39000000.0  2000000.0   3000000.0       0.0         0.0
2   2000000.0        0.0         0.0       0.0         0.0
3         0.0        0.0         0.0       0.0         0.0
4  82000000.0        0.0         0.0       0.0         0.0

   Variation de stock  Population
0                 0.0  36296113.0
1                 0.0  36296113.0
2                 0.0  36296113.0
3                 0.0  36296113.0
4                 0.0  36296113.0
```

[39]:
```
# Création de la colonne dispo_kcal avec calcul des kcal disponibles␣
  ↪mondialement (par jour) = Disponibilité alimentaire * Population
dispo_ali_pop_2017['dispo_kcal'] = dispo_ali_pop_2017['Disponibilité␣
  ↪alimentaire (Kcal/personne/jour)'] * dispo_ali_pop_2017['Population']
```

```
# Calcul du nombre de calories disponibles mondialement par jour
somme_kcal = dispo_ali_pop_2017['dispo_kcal'].sum()

# Affichage du résultat
print("Le nombre de calories disponibles mondialement par jour en 2017 est de",␣
 ↪somme_kcal, "kg")
```

```
Le nombre de calories disponibles mondialement par jour en 2017 est de
20918984627331.0 kg
```

[40]:
```
# Calcul du nombre d'humains pouvant être nourris = somme des calories / 2000
humains_nourris = somme_kcal / 2000

# Affichage du résultat
print("Le nombre d'humains pouvant être nourris en 2017 est de",␣
 ↪humains_nourris)
```

```
Le nombre d'humains pouvant être nourris en 2017 est de 10459492313.6655
```

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

[41]:
```
# Création d'un nouveau DF avec les données provenant uniquement des produits␣
 ↪d'orignie vegetale : df_vegetaux
df_vegetaux = dispo_ali_pop_2017[dispo_ali_pop_2017["Origine"] == "vegetale"]

# Affichage du nouveau DF df_vegetaux
display(df_vegetaux.head())
```

```
          Zone                Produit   Origine  Aliments pour animaux  \
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
6  Afghanistan                  Bière  vegetale                    0.0

   Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
1                  0.0                                             1.0
2                  0.0                                             1.0
3                  0.0                                             0.0
4                  0.0                                             4.0
6                  0.0                                             0.0

   Disponibilité alimentaire en quantité (kg/personne/an)  \
1                                                1.29
2                                                0.06
3                                                0.00
4                                                2.70
6                                                0.09
```

```
   Disponibilité de matière grasse en quantité (g/personne/jour)  \
1                                                   0.01
2                                                   0.01
3                                                   0.00
4                                                   0.02
6                                                   0.00

   Disponibilité de protéines en quantité (g/personne/jour)  \
1                                                   0.02
2                                                   0.03
3                                                   0.00
4                                                   0.05
6                                                   0.00

   Disponibilité intérieure  Exportations - Quantité  Importations - Quantité  \
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
6                 3000000.0                      0.0                 3000000.0

   Nourriture     Pertes  Production  Semences  Traitement  \
1  39000000.0  2000000.0   3000000.0       0.0         0.0
2   2000000.0        0.0         0.0       0.0         0.0
3         0.0        0.0         0.0       0.0         0.0
4  82000000.0        0.0         0.0       0.0         0.0
6   3000000.0        0.0         0.0       0.0         0.0

   Variation de stock  Population   dispo_kcal
1                 0.0  36296113.0   36296113.0
2                 0.0  36296113.0   36296113.0
3                 0.0  36296113.0          0.0
4                 0.0  36296113.0  145184452.0
6                 0.0  36296113.0          0.0
```

[42]:
```
# Calcul du nombre de kcal disponible pour les végétaux = somme de la colonne dispo_kcal
kcal_vegetaux = df_vegetaux["dispo_kcal"].sum()

# Affichage du résultat
print("Le nombre de calories disponibles pour les végétaux en 2017 est de", kcal_vegetaux)
```

```
Le nombre de calories disponibles pour les végétaux en 2017 est de
17260764211501.0
```

[43]:
```
#Calcul du nombre d'humains pouvant être nourris avec les végétaux = calories vegetaux / 2 000
```

```
vegetaux_nourriture = kcal_vegetaux / 2000

# Affichage du résultat
print("Le nombre d'humain pouvant être nourris par jour avec les végétaux en␣
 ↪2017 est de", vegetaux_nourriture)
```

```
Le nombre d'humain pouvant être nourris par jour avec les végétaux en 2017 est
de 8630382105.7505
```

3.4 - Utilisation de la disponibilité intérieure

[44]:
```
# Calcul de la disponibilité intérieure totale
dispo_totale = dispo_ali_pop_2017 ['Disponibilité intérieure'].sum()

# Affichage du résultat
print("La disponibilité intérieure totale en 2017 est de", dispo_totale, "kg")
```

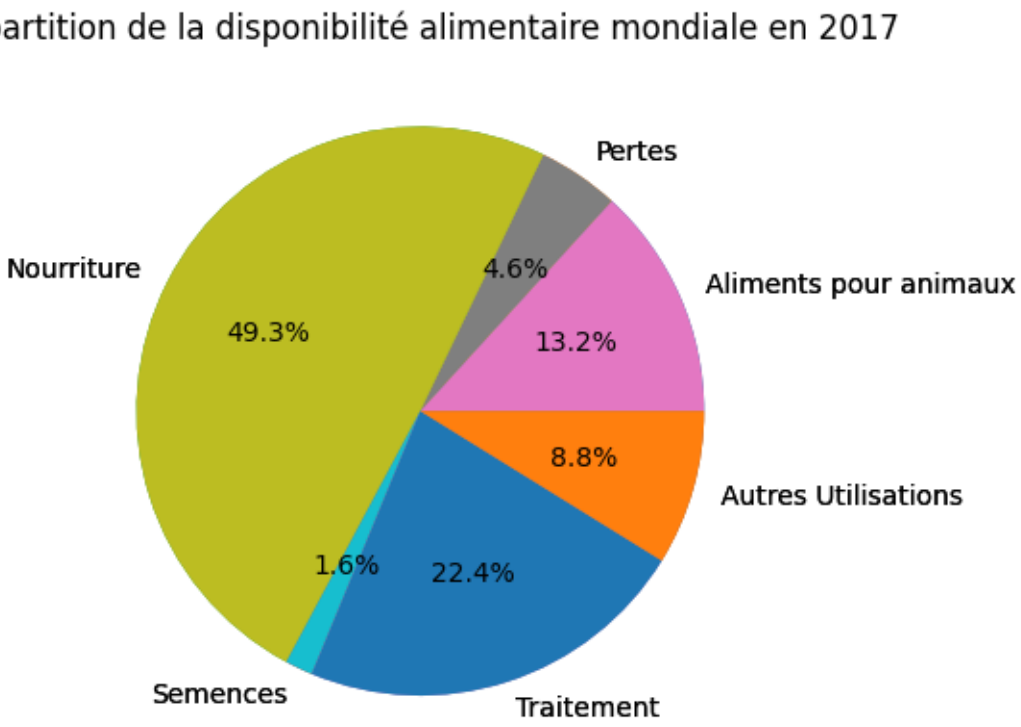
```
La disponibilité intérieure totale en 2017 est de 9733927000000.0 kg
```

[45]:
```
# Création d'une boucle for pour afficher les différentes valeurs en fonction␣
 ↪des colonnes

sommes = {
    "Aliments pour animaux": 0,
    "Pertes": 0,
    "Nourriture": 0,
    "Semences": 0,
    "Traitement": 0,
    "Autres Utilisations": 0,
}

# Parcours des lignes du DF
for index, row in dispo_ali_pop_2017.iterrows():
    # Ajout des valeurs des colonnes spécifiées aux variables de somme
    for col in sommes.keys():
        valeur = row.get(col)
        if valeur is not None:
            sommes[col] += valeur

# Affichage des résultats bruts
for col, somme in sommes.items():
    print(f"Somme de {col}: {somme}")

# Affichage du résultat (indem ceullule ci dessus)
print("La disponibilité intérieure totale en 2017 est de", dispo_totale)

# Calcul du pourcentage de chaque colonne
pourcentages = {}
```

```
for col, somme in sommes.items():
    pourcentages[col] = (somme / dispo_totale) * 100

# Affichage des résultats en pourcentage
for col, pourcentage in pourcentages.items():
    print(f"Pourcentage de {col}: {pourcentage:.2f}%")
```

```
Somme de Aliments pour animaux: 1288002000000.0
Somme de Pertes: 452283000000.0
Somme de Nourriture: 4805525000000.0
Somme de Semences: 153317000000.0
Somme de Traitement: 2185641000000.0
Somme de Autres Utilisations: 858771000000.0
La disponibilité intérieure totale en 2017 est de 9733927000000.0
Pourcentage de Aliments pour animaux: 13.23%
Pourcentage de Pertes: 4.65%
Pourcentage de Nourriture: 49.37%
Pourcentage de Semences: 1.58%
Pourcentage de Traitement: 22.45%
Pourcentage de Autres Utilisations: 8.82%
```

[46]:
```
import matplotlib.pyplot as plt

# Sélection des colonnes pour le camembert
colonnes_camembert = ["Aliments pour animaux", "Pertes", "Nourriture",␣
 ↪"Semences", "Traitement", "Autres Utilisations"]

# Extraction des valeurs des colonnes sélectionnées
valeurs_camembert = [pourcentages[col] for col in colonnes_camembert]

# Définition des labels
labels_camembert = colonnes_camembert

# Création du graphique camembert
plt.pie(valeurs_camembert, labels=labels_camembert)

# Ajout du pourcentage dans chaque secteur
plt.pie(valeurs_camembert, labels=labels_camembert, autopct='%1.1f%%')

# Affichage du titre
plt.title("Répartition de la disponibilité alimentaire mondiale en 2017")

# Affichage du graphique
plt.show()
```



3.5 - Utilisation des céréales

[47]:
```
# Création d'une liste avec toutes les variables

# Création d'une liste avec les valeurs de la colonne produit
produits_uniques = set(dispo_ali_pop_2017["Produit"])

# Trie des valeurs uniques et par ordre alphabétique
produits_triés = sorted(produits_uniques)
for produit in produits_triés:
    print(produit)
```

```
Abats Comestible
Agrumes, Autres
Alcool, non Comestible
Aliments pour enfants
Ananas
Animaux Aquatiques Autre
Arachides Decortiquees
Avoine
Bananes
Bananes plantains
Beurre, Ghee
```

Bière
Blé
Boissons Alcooliques
Boissons Fermentés
Café
Cephalopodes
Citrons & Limes
Coco (Incl Coprah)
Crustacés
Crème
Céréales, Autres
Dattes
Edulcorants Autres
Feve de Cacao
Fruits, Autres
Girofles
Graines Colza/Moutarde
Graines de coton
Graines de tournesol
Graisses Animales Crue
Haricots
Huil Plantes Oleif Autr
Huile Graines de Coton
Huile d'Arachide
Huile d'Olive
Huile de Coco
Huile de Colza&Moutarde
Huile de Germe de Maïs
Huile de Palme
Huile de Palmistes
Huile de Soja
Huile de Son de Riz
Huile de Sésame
Huile de Tournesol
Huiles de Foie de Poisso
Huiles de Poissons
Ignames
Lait - Excl Beurre
Légumes, Autres
Légumineuses Autres
Manioc
Maïs
Miel
Millet
Miscellanees
Mollusques, Autres
Noix
Oeufs

```
Oignons
Olives
Oranges, Mandarines
Orge
Palmistes
Pamplemousse
Patates douces
Perciform
Piments
Plantes Aquatiques
Plantes Oleiferes, Autre
Pois
Poissons Eau Douce
Poissons Marins, Autres
Poissons Pelagiques
Poivre
Pommes
Pommes de Terre
Racines nda
Raisin
Riz (Eq Blanchi)
Seigle
Soja
Sorgho
Sucre Eq Brut
Sucre non centrifugé
Sucre, betterave
Sucre, canne
Sésame
Thé
Tomates
Viande d'Ovins/Caprins
Viande de Anim Aquatiq
Viande de Bovins
Viande de Suides
Viande de Volailles
Viande, Autre
Vin
Épices, Autres
```

[48]:
```
# Création d'un DF avec les informations uniquement pour les céréales, à partir
 ↪de dispo_alimentaire
# Céréales = blé maïs orge avoine seigle sorgho millet riz autres

# Création du df_cereales
```

```
df_cereales = dispo_ali_pop_2017[dispo_ali_pop_2017["Produit"].isin(["Avoine",␣
 ↪"Blé", "Céréales, Autres", "Maïs", "Millet", "Orge", "Riz (Eq Blanchi)",␣
 ↪"Seigle", "Sorgho"])]

# Affichage du df_cereales
display(df_cereales.head())
```

```
          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  Population    dispo_kcal
7   322000000.0         0.0        -350000000.0  36296113.0  4.968938e+10
12          0.0         0.0                 0.0  36296113.0  0.000000e+00
32    5000000.0         0.0                 0.0  36296113.0  7.622184e+08
34          0.0         0.0                 0.0  36296113.0  1.088883e+08
40   22000000.0         0.0                 0.0  36296113.0  9.436989e+08
```

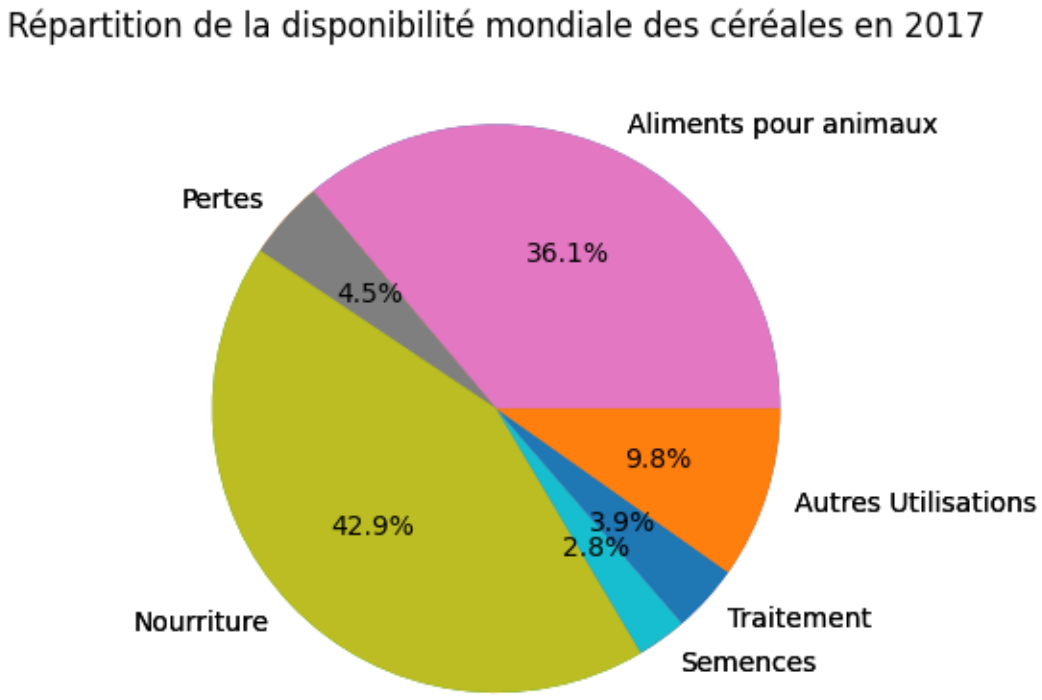
[49]:
```python
# Affichage de la proportion d'alimentation animale
# Je me permets d'étendre la question afin de calculer la proportion de chacune des valeurs dans la disponibilité intéreure totale des céréales.
# Disponibilité intérieure céréales = semences + pertes + nourriture + aliments pour animaux + traitement + autres utilisations

# Total de la disponibilité intérieure pour les céréales
dispo_cereales = df_cereales["Disponibilité intérieure"].sum()

# Création d'une boucle for pour afficher les différentes valeurs en fonction des colonnes

sommes = {
    "Aliments pour animaux": 0,
    "Pertes": 0,
    "Nourriture": 0,
    "Semences": 0,
    "Traitement": 0,
    "Autres Utilisations": 0,
}

# Parcours des lignes du DF
for index, row in df_cereales.iterrows():
    # Ajout des valeurs des colonnes spécifiées aux variables de somme
    for col in sommes.keys():
        valeur = row.get(col)
        if valeur is not None:
            sommes[col] += valeur

# Affichage des résultats bruts
for col, somme in sommes.items():
    print(f"Somme de {col}: {somme}")
```

```
# Affichage du résultat (indem ceullule ci dessus)
print("La disponibilité intérieure des céréales en 2017 est de", dispo_cereales)

# Calcul du pourcentage de chaque colonne
pourcentages = {}
for col, somme in sommes.items():
    pourcentages[col] = (somme / dispo_cereales) * 100

# Affichage des résultats en pourcentage
for col, pourcentage in pourcentages.items():
    print(f"Pourcentage de {col}: {pourcentage:.2f}%")
```

```
Somme de Aliments pour animaux: 859615000000.0
Somme de Pertes: 106706000000.0
Somme de Nourriture: 1020464000000.0
Somme de Semences: 67719000000.0
Somme de Traitement: 91781000000.0
Somme de Autres Utilisations: 232672000000.0
La disponibilité intérieure des céréales en 2017 est de 2378371000000.0
Pourcentage de Aliments pour animaux: 36.14%
Pourcentage de Pertes: 4.49%
Pourcentage de Nourriture: 42.91%
Pourcentage de Semences: 2.85%
Pourcentage de Traitement: 3.86%
Pourcentage de Autres Utilisations: 9.78%
```

[50]:
```
import matplotlib.pyplot as plt

# Sélection des colonnes pour le camembert
colonnes_cereales = ["Aliments pour animaux", "Pertes", "Nourriture",␣
 ↪"Semences", "Traitement", "Autres Utilisations"]

# Extraction des valeurs des colonnes sélectionnées
valeurs_cereales = [pourcentages[col] for col in colonnes_cereales]

# Définition des labels
labels_cereales = colonnes_cereales

# Création du graphique camembert
plt.pie(valeurs_cereales, labels=labels_cereales)

# Ajout du pourcentage dans chaque secteur
plt.pie(valeurs_cereales, labels=labels_cereales, autopct='%1.1f%%')

# Affichage du titre
plt.title("Répartition de la disponibilité mondiale des céréales en 2017")
```

```
# Affichage du graphique
plt.show()
```



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

[51]:
```
# Création de la colonne proportion par pays

# Utilisation de .loc pour créer une nouvelle colonne proportion par pays dans␣
 ↪le DF pop_sous_nut_2017 = (SousNutrition / Population) * 100
pop_sous_nut_2017.loc[:,"Proportion par pays"] =␣
 ↪(pop_sous_nut_2017["SousNutrition"] / pop_sous_nut_2017["Population"]) * 100
```

```
C:\Users\Yassine\AppData\Local\Temp\ipykernel_6840\576008377.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  pop_sous_nut_2017.loc[:,"Proportion par pays"] =
(pop_sous_nut_2017["SousNutrition"] / pop_sous_nut_2017["Population"]) * 100
```

[52]:
```
# Affichage, après trie, des 10 pays où la proportion de personnes en état de sous-nutrition est la plus forte en 2017

# Tri décroissant du DF pop_sous_nut_2017 par la colonne Proportion par pays
pop_sous_nut_2017_flop = pop_sous_nut_2017.sort_values("Proportion par pays",ascending=False)

# Affichage des 10 premières lignes du DF
display(pop_sous_nut_2017_flop.head(10))
```

	Zone	Année	Population	SousNutrition	Proportion par pays
472	Haïti	2017	10982366.0	5300000.0	48.259182
946	République populaire démocratique de Corée	2017	25429825.0	12000000.0	47.188685
652	Madagascar	2017	25570512.0	10500000.0	41.062924
622	Libéria	2017	4702226.0	1800000.0	38.279742
604	Lesotho	2017	2091534.0	800000.0	38.249438
1102	Tchad	2017	15016753.0	5700000.0	37.957606
970	Rwanda	2017	11980961.0	4200000.0	35.055619
730	Mozambique	2017	28649018.0	9400000.0	32.810898
1120	Timor-Leste	2017	1243258.0	400000.0	32.173531
4	Afghanistan	2017	36296113.0	10500000.0	28.928718

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

[53]:
```
# Calcul, par pays, du total de l'aide alimentaire

# Regroupement du DF aide_alimentaire par Zone (pays) et calcul de la somme de Valeur
aide_alimentaire_total = aide_alimentaire.groupby("Zone")["Valeur"].sum()
```

[54]:
```
# Affichage, après trie, des 10 pays qui ont le plus bénéficié de l'aide alimentaire

# Tri du DF aide_alimentaire_totale par la colonne valeur par ordre décroissant
aide_alimentaire_total_top = aide_alimentaire_total.sort_values(ascending=False)
```

```
# Affichage des résultats triés
display(aide_alimentaire_total_top.head(10))
```

```
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
Name: Valeur, dtype: int64
```

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

[55]:
```
# Création d'un dataframe avec la zone, l'année et l'aide alimentaire puis groupby sur zone et année

# Création du DF aide_ali avec les colonnes Zone, Année et Valeur du DF aide_alimentaire
aide_ali = aide_alimentaire[["Zone", "Année", "Valeur"]]

# Regroupement du DF aide_ali par Zone puis par Année, puis addition des valeurs
aide_ali_groupe = aide_ali.groupby(["Zone", "Année"])["Valeur"].sum()

# Affichage du résultat
display(aide_ali_groupe.head())
```

```
Zone         Année
Afghanistan  2013     128238000
             2014      57214000
Algérie      2013      35234000
             2014      18980000
             2015      17424000
Name: Valeur, dtype: int64
```

[56]:
```
#Création d'une liste contenant les 5 pays qui ont le plus bénéficié de l'aide alimentaire

# Grouper par Zone et additionner les Valeur
aide_ali_zone = aide_ali.groupby("Zone")["Valeur"].sum()

# Trier par valeur dans l'ordre décroissant
aide_ali_zone_desc = aide_ali_zone.sort_values(ascending=False)
```

```
# Affichage des 5 zones avec la valeur la plus élevée et leur valeur respective
top_5_zones = aide_ali_zone_desc.to_frame().reset_index().head(5)

print(top_5_zones)
```

```
                         Zone      Valeur
0  République arabe syrienne  1858943000
1                   Éthiopie  1381294000
2                      Yémen  1206484000
3               Soudan du Sud   695248000
4                      Soudan   669784000
```

[57]:
```
# Filtrage sur le DF avec la liste
# Affichage des pays avec l'aide alimentaire par année

# Filtrage des données pour les 5 zones sélectionnées
zone_select = ["République arabe syrienne", "Éthiopie", "Yémen", "Soudan du␣
 ↪Sud", "Soudan"]
aide_ali_filtre = aide_ali[aide_ali["Zone"].isin(zone_select)]

# Groupement par Zone et Année et additionner les Valeur
aide_ali_zone_annee = aide_ali_filtre.groupby(["Zone", "Année"])["Valeur"].sum()

# Afficher le résultat
print(aide_ali_zone_annee)
```

```
Zone                         Année
République arabe syrienne    2013     563566000
                             2014     651870000
                             2015     524949000
                             2016     118558000
Soudan                       2013     330230000
                             2014     321904000
                             2015      17650000
Soudan du Sud                2013     196330000
                             2014     450610000
                             2015      48308000
Yémen                        2013     264764000
                             2014     103840000
                             2015     372306000
                             2016     465574000
Éthiopie                     2013     591404000
                             2014     586624000
                             2015     203266000
Name: Valeur, dtype: int64
```

3.9 - Pays avec le moins de disponibilité par habitant

[58]:
```
#Calcul de la disponibilité en kcal par personne par jour par pays

# Regroupement du DF dispo_alimentaire par Zone
dispo_alimentaire_zone = dispo_alimentaire.groupby("Zone")

# Calcul de la somme de la disponibilité alimentaire pour chaque Zone
dispo_alimentaire_jour = dispo_alimentaire_zone["Disponibilité alimentaire␣
↪(Kcal/personne/jour)"].sum()
```

[59]:
```
# Affichage des 10 pays qui ont le moins de disponibilité alimentaire par␣
↪personne

# Tri du DF par la colonne disponibilité alimentaire par ordre croissant
dispo_alimentaire_asc = dispo_alimentaire_jour.sort_values()

# Affichage des 10 premières lignes
display(dispo_alimentaire_asc.head(10))
```

```
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
Name: Disponibilité alimentaire (Kcal/personne/jour), dtype: float64
```

3.10 - Pays avec le plus de disponibilité par habitant

[60]:
```
# Affichage des 10 pays qui ont le plus de disponibilité alimentaire par␣
↪personne

# Tri du DF par la colonne disponibilité alimentaire en ordre décroissant
dispo_alimentaire_desc = dispo_alimentaire_jour.sort_values(ascending=False)

# Affichage des 10 premières lignes
display(dispo_alimentaire_desc.head(10))
```

```
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
Name: Disponibilité alimentaire (Kcal/personne/jour), dtype: float64
```

3.11 - Exemple de la Thaïlande pour le Manioc

[61]:
```
# Création d'un DF avec uniquement la zone Thaïlande

# Filtrage du nouveau DF sur la zone "Thaïlande"
df_thailande = dispo_ali_pop_2017[dispo_ali_pop_2017["Zone"] == "Thaïlande"]

# Affichage du nouveau DF
display(df_thailande)
```

```
           Zone                 Produit   Origine  Aliments pour animaux  \
13570  Thaïlande        Abats Comestible   animale                    0.0
13571  Thaïlande         Agrumes, Autres  vegetale                    0.0
13572  Thaïlande  Alcool, non Comestible  vegetale                    0.0
13573  Thaïlande   Aliments pour enfants  vegetale                    0.0
13574  Thaïlande                 Ananas  vegetale                    0.0
…          …                       …       …                        …
13660  Thaïlande        Viande de Suides   animale                    0.0
13661  Thaïlande     Viande de Volailles   animale                    0.0
13662  Thaïlande           Viande, Autre   animale                    0.0
13663  Thaïlande                     Vin  vegetale                    0.0
13664  Thaïlande          Épices, Autres  vegetale                    0.0

       Autres Utilisations  Disponibilité alimentaire (Kcal/personne/jour)  \
13570                  0.0                                             3.0
13571                  0.0                                             0.0
13572          358000000.0                                             0.0
13573                  0.0                                             2.0
13574                  0.0                                            10.0
…                      …                                               …
13660                  0.0                                           124.0
13661                  0.0                                            52.0
13662                  0.0                                             0.0
13663                  0.0                                             0.0
13664                  0.0                                            16.0

       Disponibilité alimentaire en quantité (kg/personne/an)  \
13570                                               1.11
13571                                               0.09
13572                                               0.00
13573                                               0.18
13574                                              10.02
…                                                  …
```

```
13660                                                 13.00
13661                                                 13.69
13662                                                  0.03
13663                                                  0.12
13664                                                  1.70

       Disponibilité de matière grasse en quantité (g/personne/jour)  \
13570                                                  0.09
13571                                                  0.00
13572                                                  0.00
13573                                                  0.01
13574                                                  0.04
…                                                      …
13660                                                 11.83
13661                                                  3.62
13662                                                  0.01
13663                                                  0.00
13664                                                  0.30

       Disponibilité de protéines en quantité (g/personne/jour)  \
13570                                                  0.56
13571                                                  0.00
13572                                                  0.00
13573                                                  0.08
13574                                                  0.08
…                                                      …
13660                                                  3.92
13661                                                  4.49
13662                                                  0.02
13663                                                  0.00
13664                                                  0.43

       Disponibilité intérieure  Exportations - Quantité  \
13570                74000000.0              5.000000e+06
13571                 8000000.0              6.000000e+06
13572               358000000.0              1.100000e+08
13573                12000000.0              7.000000e+06
13574               782000000.0              1.449000e+09
…                         …                        …
13660               871000000.0              2.200000e+07
13661               945000000.0              5.360000e+08
13662               -92000000.0              9.600000e+07
13663                 8000000.0              8.000000e+06
13664               114000000.0              4.200000e+07

       Importations - Quantité   Nourriture       Pertes    Production  \
13570               33000000.0   75000000.0          0.0  4.500000e+07
13571                2000000.0    6000000.0          0.0  1.200000e+07
```

```
13572               21000000.0          0.0          0.0  4.470000e+08
13573               19000000.0   12000000.0          0.0  0.000000e+00
13574                9000000.0  671000000.0  110000000.0  2.209000e+09
…                          …          …            …           …
13660                1000000.0  871000000.0          0.0  8.910000e+08
13661               11000000.0  917000000.0   28000000.0  1.470000e+09
13662                4000000.0    2000000.0          0.0  0.000000e+00
13663               16000000.0    8000000.0          0.0  0.000000e+00
13664               13000000.0  114000000.0          0.0  1.430000e+08

       Semences  Traitement  Variation de stock  Population    dispo_kcal
13570       0.0         0.0                 0.0  69209810.0  2.076294e+08
13571       0.0   2000000.0                 0.0  69209810.0  0.000000e+00
13572       0.0         0.0                 0.0  69209810.0  0.000000e+00
13573       0.0         0.0                 0.0  69209810.0  1.384196e+08
13574       0.0         0.0          13000000.0  69209810.0  6.920981e+08
…           …           …                   …           …           …
13660       0.0         0.0                 0.0  69209810.0  8.582016e+09
13661       0.0         0.0                 0.0  69209810.0  3.598910e+09
13662       0.0         0.0                 0.0  69209810.0  0.000000e+00
13663       0.0         0.0                 0.0  69209810.0  0.000000e+00
13664       0.0         0.0                 0.0  69209810.0  1.107357e+09

[95 rows x 20 columns]
```

[62]:
```
# Calcul de la sous nutrition en Thaïlande (cette donnée est déjà dans le DF␣
 ↪pop_sous_nut_2017)

# Filtrage du DF pop_sous_nut_2017 sur la zone Thaïlande
df_thailande_sn = pop_sous_nut_2017[pop_sous_nut_2017["Zone"] == "Thaïlande"]

# Affichage du DF
display(df_thailande_sn)
```

```
          Zone  Année  Population  SousNutrition  Proportion par pays
1114  Thaïlande   2017  69209810.0      6200000.0             8.958268
```

[63]:
```
# Calcul de la proportion des exportations de manioc en fonction de la␣
 ↪production

# Filtrage du DF sur le produit "manioc"
df_manioc = df_thailande[df_thailande["Produit"] == "Manioc"]

# Calcul de la proportion des exportations par rapport à la production =␣
 ↪(exportation / production)*100
df_manioc_proportion = (df_manioc["Exportations - Quantité"] /␣
 ↪df_manioc["Production"]) * 100
```

```
# Affichage du résultat
proportion = df_manioc_proportion.iloc[0]
print(f"Le taux d'exportation de manioc par rapport à sa production en␣
 ↪Thaïlande est de {proportion:.2f}%")
```

```
Le taux d'exportation de manioc par rapport à sa production en Thaïlande est de
83.41%
```

[64]:
```
# Total de la disponibilité intérieure pour la Thaïlande

# Groupement de la zone thailande ; somme de la colonne disponibilité intérieure
dispo_int_thai = df_thailande[df_thailande["Zone"] == "Thaïlande"].
 ↪groupby("Zone")["Disponibilité intérieure"].sum()

# division du résultat par la valeur de la colonne population
dispo_int_thai = dispo_int_thai / df_thailande[df_thailande["Zone"] ==␣
 ↪"Thaïlande"]["Population"].values[0]

# Afficher le résultat
print(dispo_int_thai)
```

```
Zone
Thaïlande    2284.791708
Name: Disponibilité intérieure, dtype: float64
```

[65]:
```
# Calcul du nombre théorique de thaïlandais pouvant être nourris

kcal_thai_pers = (df_thailande["dispo_kcal"].sum()) / 2000


# Affichage du résultat
print("Le nombre théorique de thaïlandais pouvant être nourris en 2017 est de",␣
 ↪kcal_thai_pers)
```

```
Le nombre théorique de thaïlandais pouvant être nourris en 2017 est de
96374660.425
```

[66]:
```
# Calcul de la balance commerciale de la Thaïlande = exportations - importations

# Calculer la somme de la colonne Exportation
balance_commerciale = (df_thailande["Exportations - Quantité"].sum()) -␣
 ↪(df_thailande["Importations - Quantité"].sum())


# Afficher le résultat
print("La balance commerciale de la Thaïlande en 2017 est de",␣
 ↪balance_commerciale)
```

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

Etape 6 - Analyse complémentaires

[67]:
```
# Rajouter en dessous, toutes les analyses complémtaires 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"
# Balance commercaile de la Thailande (voir cellule ci dessus)
```