Chargement des données avec pandas
pd.read_csv

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

Convertir les types de données d'un dataFrame
astype() - ex astype(int)

```
population['Population'] = population['Population'].astype(int)
```

Supprimer une colonne
drop()

```
population.drop(columns=['Valeur'], inplace=True)
```

inplace = True → modifie directement dans le dataFrame au lieu de faire une copie

Remplacer NaN par 0
fillna(0)

Conversion d'une colonne en numérique

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

Lorsque vous utilisez `errors='coerce'` avec certaines opérations sur un DataFrame pandas, cela signifie que les valeurs qui ne peuvent pas être converties en un type compatible sont remplacées par NaN (Not a Number).

Renommer une colonne

rename()

```
sousNutrition = sousNutrition.rename(columns={'Valeur': 'sous_nutrition'})
```

Exporter un fichier au format excel
dataFrame.to_excel

```
sousNutrition.to_excel('sous_nutrition.xlsx', index=False)
```

Diviser une chaîne de caractères en tenant en compte un caractère

```
str.split('-')
```

Utiliser une fonction

```
apply(create_year_list)
```

Déplier une liste d'une colonne pour que chaque valeur soit sur une ligne

```
explode('Année')
```

Réinitialiser les index

```
reset_index(drop=True)
```

Fusionner deux dataFrame comme une jointure SQL (par défaut inner join)

```
pd.merge(population, sousNutrition, on=['Zone', 'Année'])
```

```
jointure = pd.merge(population, sousNutrition, on=['Zone', 'Année'], how='outer')
```

Grouper en fonction des valeurs d'une ou plusieurs colonnes

```
groupby('Produit')
```

Faire une moyenne sur une colonne

```
mean()
disponibilite_moyenne_par_personne = dispoAlimentaire[['Produit', 'Disponibilité alimentaire en quantité (kg/personne/an)']].groupby('Produit').mean()
print(disponibilite_moyenne_par_personne)
```

Trouver le nombre d'éléments unique dans un dataFrame

```
nombre_pays = population_2017['Zone'].nunique()
```

Transformer le résultat d'un group by et d'une agregation dans un tableau avec un n° d'index pour chaque ligne

reset_index()

```
Nbre_cal_journaliere_par_personne_disponibles = dispoAlimentaire.groupby('Zone')['Disponibilité alimentaire (Kcal/personne/jour)'].sum().reset_index()
```

Supprimer colonne ou ligne d'un DataFrame

- `axis=0` : Supprime les lignes. Cet argument indique que l'étiquette donnée se réfère à l'index (les lignes).
- `axis=1` : Supprime les colonnes. Cet argument indique que l'étiquette donnée se réfère aux colonnes.

```
# Supprimer la colonne 'Nbre_cal_journaliere_par_personne_disponibles' du DataFrame dispoAlimentaire_vegetaux car elle regroupe toutes les origines
dispoAlimentaire_vegetaux = dispoAlimentaire_vegetaux.drop('Nbre_cal_journaliere_par_personne_disponibles', axis=1)
```

Formater un nombre

- La chaîne `"La disponibilité intérieure mondiale est de : {:.2f} kilos"` contient du texte et une spécification de format pour la valeur numérique à insérer.
- `:.2f` est une spécification de format qui signifie :
  - `:` : Indique le début de la spécification de format.
  - `.2` : Précise que deux décimales doivent être affichées.
  - `f` : Indique que la valeur est un nombre à virgule flottante (float).

```
print("Le nombre total d'humains pouvant être nourris avec les produits végétaux est d'environ : {:.2f} milliards d'humains".format(nombre_total_humains_nourris_vegetaux / 1e9))
```

- 1e9 = 10 exp 9 soit 1 milliard

Aller chercher ou vérifier si des élements sont présents dans une liste de valeurs

- Lorsque vous utilisez `isin()` sur une Series, vous vérifiez si les éléments de la Series sont présents dans une liste de valeurs.

```
cereales = dispoAlimentaire[dispoAlimentaire['Produit'].isin(['Blé', 'Riz (Eq Blanchi)',
'Orge', 'Maïs', 'Millet', 'Seigle', 'Avoine', 'Sorgho', 'Céréales, Autres'])]
```

Méthode pour modifier en chaine de caractère en float

- Utilise la méthode `str.replace('%', '')` pour supprimer le symbole `%` des chaînes de caractères dans la colonne.
- `str` est un accès aux méthodes de manipulation de chaînes pour les Series de pandas.
- `replace('%', '')` remplace toutes les occurrences du symbole `%` par une chaîne vide, supprimant ainsi le symbole `%`.

```
resultats_2017['proportion_sous_nutrition'] =
resultats_2017['proportion_sous_nutrition'].str.replace('%', '').astype(float)
```