T3.1 Tables de données

3.1.1 Données tabulées

Lorsqu'on souhaite manipuler des données, il est très fréquent de les présenter sous forme de tableaux: extrait de compte bancaire, bulletin scolaire, statistiques sportives, etc. On parle alors de *données tabulées*.

{: .center}

Ces tables sont le principal moyen de stocker des données structurées, et il est très important de savoir traiter efficacement ces données pour en tirer de l'information.

!!! abstract "Vocabulaire" - une table est une collection d'**enregistrements**; - chaque enregistrement contient des valeurs correspondant aux **champs** ou **attributs** de la table (des enregistrements).

```
| Nom | Classe | LV1 | Spés |
|:-:|:-:|:-:|:-:|
|Safia|1G1|Anglais|Maths,Physique-Chimie|
|Dawson|1G1|Anglais|Maths,Physique-Chimie|
|Nathan|1G5|Anglais|LLCE,Physique-Chimie|
|Mélodie|1G1|Anglais|Maths,Physique-Chimie|
|Manon|1G2|Anglais|HGGSP,LLCE|

Dans la table ci-dessus, les champs/attributs sont `'Nom'`, `'Classe'`, `'LV1'` et `'Spés'`
```

!!! code "Représentation des tables en Python" Puisque chaque enregistrement contient une valeur pour chaque champ de la table, il paraît naturel de représenter chaque enregistrement par un **dictionnaire**.

```
Par exemple, le premier enregistrement de la table précédente est
```python
{'Nom': 'Safia', 'Classe': '1G1', 'LV1': 'Anglais', 'Spés': 'Maths,Physique-Chimie'}
```


Une table sera donc représentée par une **liste de dictionnaires**.
```

3.1.2 Fichiers CSV

Les tables de données que nous manipulerons sont représentées par des fichiers au format `csv` (*comma separated values*) qui est un format fréquemment utilisé pour importer ou exporter des données tabulées.

C'est un simple fichier texte dans lequel la première ligne contient les champs, et chaque ligne suivante les enregistrements. Les champs et valeurs sont séparées par un **délimiteur**, généralement la **virgule** (*comma* in english).

!!! warning “Virgule ou point-virgule?” Il peut être nécessaire (comme dans l'exemple de la table précédente) de conserver la virgule pour séparer une valeur composite. On utilise alors parfois le point-virgule pour séparer les champs et valeurs... Lors de l'import d'un fichier `csv` avec un tableur, il faut préciser à l'ouverture quel délimiteur il faut utiliser.

!!! example “Exercice” - Télécharger ce fichier{:target="_blank"} et l'ouvrir avec LibreOffice Calc. Observer l'importance de préciser les délimiteurs. - L'ouvrir ensuite avec l'éditeur de texte (`gedit`) pour observer sa construction.

!!! code “Charger un fichier CSV avec Python” Avec une ouverture classique de fichier texte, les données importées sont sous forme d'une liste, dont la première valeur contient les champs, et les autres les enregistrements:

```
```python
>>> data = open('pokedex.csv').read().splitlines()
>>> data[:5]
['No;Nom;Type;PV;Attaque;Défense;Vitesse;ASpé;DSpé;Talent;Nom US;code', '1;Bulbizarre;plante
```
Pour obtenir une liste de dictionnaires, il faudrait donc retravailler ces données (c'est un

On va donc plutôt utiliser un module dédié à la manipulation de fichiers au format `csv`, le

```python linenums='1' title="Code-type d'importation d'un fichier CSV"
import csv

with open('pokedex.csv') as f:       # (1)
    table = list(csv.DictReader(f, delimiter=';')) # (2)
```

1. On charge le fichier dans une variable `f`
2. On applique la fonction `DictReader` du module `csv` au fichier, **en précisant le délimi
```

!!! example “Exercice” Exécuter le code précédent et contrôler que la variable `table` est bien une liste de dictionnaires avec l'inspecteur d'objet de Thonny.

3.1.3 Export en csv

!!! code “Écrire un fichier CSV” Le module `csv` permet également d'exporter une table (une liste de dictionnaires, donc) vers un fichier CSV, avec la fonction `DictWriter`.

```
```python linenums='1' title="Code-type d'exportation d'une table vers un fichier CSV"
def export_csv(nomfichier: str, table: list, champs: list) -> None:
    '''
    Écrit dans un fichier le contenu d'une table connaissant ses champs.

    - nomfichier: le nom du fichier à créer, d'extension csv. S'il existe, il sera écrasé,
```
```

```
    sinon il sera créé.
    - table: la table à exporter dans le fichier.
    - champs: une liste de str contenant les intitulés des champs de la table.
    '''
    with open(nomfichier, "w") as sortie:
        w = csv.DictWriter(sortie, champs)
        w.writeheader()
        w.writerows(table)
```