RD06 TRAITEMENT DE DONNÉES EN TABLE

Définition 1 : Bases de données

- Une **base de données** (BD) peut contenir une ou plusieurs **tables**.
- Une **table** est un ensemble d'objets appelés **enregistrements**.
- Un **enregistrement** possède différentes **valeurs** associées à des **attributs** (descripteurs).
- Toutes les **valeurs** que peut prendre un attribut s'appelle son **domaine de valeurs** et sont toutes du même type.

voiture nom de la table

marque	couleur	plaque
Renault	bleu	1233 DC 81
BMW	rouge	1213 DC 95
Audi	orange	2342 AC 66
Mercedes	argent	1234 CD 88

attributs ou descripteurs (colonnes)

enregistrements avec leurs valeurs

Le format CSV

Le format CSV (pour comma separated values, valeurs séparées par des virgules) est un fichier texte où chaque ligne représente un enregistrement avec différents champs séparés par une virgule. La première ligne d'un fichier CSV est généralement utilisée pour indiquer le nom des différents champs que l'on appelle les descripteurs.

- Le caractère de séparation peut aussi être un point-virgule, une tabulation ou deux points.
- Un fichier CSV peut être lu par un éditeur de texte mais aussi un tableur.

Dans la suite du cours, nous utiliserons comme BD, le fichier `countries.csv` issues du site `www.geonames.org` qui listes les pays du monde entier. Les descripteurs sont assez clairs, mis à part les premiers qui correspondent à des codes associés aux pays selon la norme ISO-3166. En voici les premières lignes :

```
ISO-alpha2;ISO-alpha3;ISO-numeric;fips;Country;Capital;Area;Population;Continent
AD;AND;20;AN;Andorra;Andorra la Vella;468.0;84,000;EU
AE;ARE;784;AE;United Arab Emirates;Abu Dhabi;82,880.0;4,975,593;AS
AF;AFG;4;AF;Afghanistan;Kabul;647,500.0;29,121,286;AS
AG;ATG;28;AC;Antigua and Barbuda;St. John's;443.0;86,754;NA
AI;AIA;660;AV;Anguilla;The Valley;102.0;13,254;NA
AL;ALB;8;AL;Albania;Tirana;28,748.0;2,986,952;EU
...
```

I. Importation d'une table depuis un fichier CSV

Le module natif csv de Python permet d'importer les données depuis un fichier CSV. Voici un script permettant de charger le fichier `countries.csv` avec des points-virgules comme délimitations.

</> Code Python

```
import csv

with open('countries.csv', newline='', encoding='utf-8') as csvfile:
    pays = list(csv.reader(csvfile, delimiter=';'))
```

Exercice 1 : Importation d'une table depuis un fichier CSV

1. Récupérer le fichier countries.csv depuis le répertoire de travail et le sauvegarder dans le répertoire de travail.
2. L'explorer avec un éditeur de texte et identifier chaque descripteur.
3. Créer un fichier python dans le même répertoire et y copier le script ci-dessus.
4. Exécuter le script ci-dessus (il devra être placé dans le même dossier que countries.csv).
5. Qu'affiche l'instruction `pays[0]` ?
6. Qu'affiche l'instruction `pays[1]` ?

Remarque

On s'aperçoit que la première ligne (des descripteurs) est traitée comme les autres (les enregistrements). Les valeurs du 1re enregistrement (concernant l'Andorre) ne sont pas associées directement à leur descripteur. Pour y remédier et pouvoir faciliter le traitement des données, il est préférable que celles-ci soient sous forme d'une liste de dictionnaire, plutôt qu'un simple tableau (liste de listes) comme dans l'exemple suivant :

</> Code Python

```
import csv

with open('countries.csv', newline='', encoding='utf-8') as csvfile:
    pays = list(csv.DictReader(csvfile, delimiter=';'))
```

Exercice 2 : Importation d'une table depuis un fichier CSV avec DictReader

1. Modifier votre code python pour utiliser la classe `DictReader` du module csv.
2. Qu'affiche l'instruction `pays[0]` ?
3. Qu'affiche l'instruction `pays[1]` ?

Remarque

La variable `pays` est donc une liste d'objets ressemblant à des dictionnaires. Chaque élément de cette liste correspond à une ligne du fichier `countries.csv` (sauf la première qui servira d'attribut).

Dans notre cas, pour observer la variable `pays[0]` comme un dictionnaire, il faudra faire :

</> Code Python

```
>>> dict(pays[0])
{'ISO-alpha2': 'AD', 'ISO-alpha3': 'AND', 'ISO-numeric': '20', 'fips': 'AN', 'country': 'Andorra',
 'capital': 'Andorra la Vella', 'area': '468.0', 'population': '84', 'continent': 'EU'}
```

Ainsi pour accéder à la capitale de l'Andorre, il faudra faire :

</> Code Python

```
>>> pays[0]['capital']
'Andorra la Vella'
```

II. Rechercher dans une table

Définition 1 : Recherche dans une table

Une **recherche** vise à extraire une partie des données d'une table vérifiant certains **critères**.
Un critère est une condition appliquée à la **valeur** d'un **attribut** d'un **enregistrement**.

Exercice 3 : Quels sont les pays de l'union européenne?

Intuitivement, on peut faire une recherche par itération. Les pays seront rassemblés dans une liste `resul`.

</> Code Python

```
# Recherche des pays de l'union européenne
resul = []
for enreg in donnees:
    # A compléter




assert resul == ['Andorra', 'Albania', ...]
```

Exercice 4 : Quels sont les différents continents présents dans la base?

Remarque : Elimination des doublons

Pour éliminer les éventuels doublons dans la liste, on utilise la fonction `set()`.

</> Code Python

```
>>> set([3, 2, 1, 3, 7])
{3, 2, 1, 7}
```

Exercice 5 : Déterminer le nom du pays le plus peuplé d'Europe

III. Trier une table

Le tri d'une table présente deux intérêts : il permet d'établir des classements et accélère la recherche de l'information. En effet, il est beaucoup plus efficace (rapide) d'effectuer une recherche dans une liste triée. Cela vient du fait qu'il est alors possible d'effectuer une recherche par dichotomie.

Tri selon un unique critère

On ne peut pas directement trier la liste pays car cela ne veut rien dire. Il faut indiquer selon quels critères (population, superficie etc.) on veut effectuer ce tri.

Pour cela, on appelle la fonction `sorted(list)` qui retourne une copie de la liste triée ou la méthode `list.sort()` qui trie la liste sur place. Ces deux fonctions ont deux paramètres possibles :

- `key` qui est la fonction permettant d'évaluer les éléments de la liste afin de les classer selon leur score
- `reverse` qui est un boolean (par défaut `False`) permettant de trie selon l'ordre décroissant (`True`).

Par exemple, si l'on veut trier les pays selon leur superficie : il faut d'abord définir une fonction qui permet d'accéder à la valeur de l'attribut `'area'` d'un enregistrement puis retourne le score de l'enregistrement afin qu'il puisse être comparé aux autres et ainsi trier la liste entière.

</> Code Python

```python
def cle_superficie(p):
    return p['area']

pays.sort(key=cle_superficie, reverse=True) # tri

top5 = []
for p in pays[:5] : # extraction
    top5.append((p['country'], p['area']))
```

Exercice 6 :

1. Ajouter le script ci-dessus à la suite de l'importation des données (script précédent) et l'exécuter.
2. Afficher les 5 pays les plus peuplés d'Europe.
3. Les 5 pays affichés ne sont pas ceux ayant la plus grande superficie ? Donner une explication au problème.
4. Proposer une solution pour régler ce problème.

Remarque

- la fonction `float` permet de transformer une chaîne de caractère en nombre à virgule flottante (nombre décimal)
- la méthode `mot.replace(",","")` permet de supprimer les virgules d'une chaîne de caractère (utile pour les nombres avec des milliers séparés par des virgules)

Tri selon plusieurs critères

Supposons maintenant que l'on veut trier les pays selon deux critères : d'abord par continent, puis par population.

</> Code Python

```python
def cle_population(p):
    return int(p['population'].replace(",", ""))

def cle_continent(p):
    return p['continent']

pays.sort(key=cle_population, reverse=True)  # tri par population
pays.sort(key=cle_continent)                 # tri par continent
resultat = []
for p in pays[:5] : # extraction
    resultat.append((p['country'], p['population'], p['continent']))
for p in resultat : # affichage
    print(p)
```

IV. Ecrire dans un fichier CSV

Pour écrire dans un fichier CSV, on peut utiliser la fonction `csv.writer` du module csv. Voici un script permettant d'écrire une liste de listes dans un fichier CSV avec des points-virgules comme délimiteurs.

</> Code Python

```python
import csv
with open('nom_fichier.csv', 'w', newline='', encoding='utf-8') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=liste_de_dico[0].keys(), delimiter=';')
    writer.writeheader()
    writer.writerows(liste_de_dico)
```