Traitement de données Excel sous Python

1 Introduction et prérecquis

Nous allons utiliser deux modules Python : *Openpyxl* et *Pandas*. Nous pouvons les installer en utilisant les deux lignes de commande suivantes dans le prompt actif :

```
pip install openpyxl
pip install pandas
```

Notons également que le fichier Python de traitement de données et le fichier contenant les données Excel à traiter doivent être dans le même dans dossier, sans quoi Python ne trouvera pas le fichier de données.

Pensons également en début de code à importer les modules dont nous allons nous servir :

```
import pandas as pd
```

Remarque : Certaines fonctions de *Pandas* dépendent d'*Openpyxl*. L'installation des deux modules est nécéssaire. Si l'on ne fait pas appel à des fonctions propres à *Openpyxl*, son importation n'est pas nécéssaire.

2 Création d'un DataFrame

La structure essentielle pour traiter des données avec le module *Pandas* est le *DataFrame*. Chaque fois qu'on importe un fichier excel dans Python via *Pandas*, il le converti en DataFrame. Cela se fait au moyen de la commande `.read_excel()`.

```
DataFrame = pd.read_excel("donnees.xlsx"))
```

La structure des DataFrames est très similaire à celle d'un fichier SQL. Chaque ligne correspond à une entrée dans la base de donnée, chaque colonne une propriété de l'entrée (p.ex. bookID, title, authors, etc...). La première ligne de chaque colonne du fichier Excel est interprétée comme le nom de la propriété.

Pour avoir un aperçu du DataFrame créé, il suffit d'utiliser la commande `print()`.

```
print(DataFrame)
```

2.1 Choix des pages lues

Par défaut, Pandas lit la première page du document excel fourni. La fonction `.read_excel()` permet de choisir les pages lues par Panda avec l'argument (optionnel) `sheet_name`.

bookID	title	authors	...	text_reviews_count	publication_date	publisher
1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPrÃ©	...	27591	9/16/2006	Scholastic Inc.
2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPrÃ©	...	29221	2004-01-09 00:00:00	Scholastic Inc.
3	Harry Potter and the Chamber of Secrets (Harry...	J.K. Rowling	...	244	2003-01-11 00:00:00	Scholastic
4	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPrÃ©	...	36325	2004-01-05 00:00:00	Scholastic Inc.
5	Harry Potter Boxed Set Books 1-5 (Harry Potte...	J.K. Rowling/Mary GrandPrÃ©	...	164	9/13/2004	Scholastic
6	Unauthorized Harry Potter Book Seven News: "Ha...	W. Frederick Zimmerman	...	1	4/26/2005	Nimble Books
7	Harry Potter Collection (Harry Potter #1-6)	J.K. Rowling	...	808	2005-12-09 00:00:00	Scholastic
8	The Ultimate Hitchhiker's Guide: Five Complete...	Douglas Adams	...	254	2005-01-11 00:00:00	Gramercy Books
9	The Ultimate Hitchhiker's Guide to the Galaxy ...	Douglas Adams	...	4080	4/30/2002	Del Rey Books
10	The Hitchhiker's Guide to the Galaxy (Hitchhik...	Douglas Adams	...	460	2004-03-08 00:00:00	Crown

FIGURE 1 – Liste de livres sur Excel converti en DataFrame

Lecture d'une seule page

On peut sélectionner la page par son indice (entier de type `int`) ou par son nom directement.

```
DataFrame = pd.read_excel("donnees.xlsx", sheet_name = 0))
DataFrame = pd.read_excel("donnees.xlsx", sheet_name = "Nom"))
```

Lecture de plusieurs pages

Dans la même logique que la lecture d'une seule page, on peut sélectionner plusieurs pages par leur indice ou leur nom. Si l'on écrit `None`, Pandas sélectionnera toutes les pages.

```
DataFrame = pd.read_excel("donnees.xlsx", sheet_name = [0,1]))
DataFrame = pd.read_excel("donnees.xlsx", sheet_name = ["Page_1", "Page_2"]))
```

Remarque : Il faudra ensuite spécifier la page que l'on souhaite manipuler en utilisant les commandes de sélection de données. Par exemple, on devra écrire : `DataFrame["Page_1"].head(n)` au lieu d'écrire `DataFrame.head(n)`.

2.2 Choix des colonnes et lignes lues

Pour ne pas alourdir le DataFrame inutilement, on peut choisir les colonnes du fichier Excel que l'on souhaite retrouver dans le DataFrame. On utilise pour cela le paramètre `usecols`. On référence les colonnes que l'on souhaite utiliser par leur nom (la première ligne), le tout dans une liste.

```
DataFrame = pd.read_excel("donnees.xlsx", usecols = ["col_1","col_2"}))
```

2.3 Conversion en différents formats de données

Si nécéssaire, Pandas permet de convertir les DataFrame en autre formats de données : dictionnaires , json, etc... On utilise pour cela les commandes `.to_dict()`, `.to_json()`, etc...

3 Analyse de données

3.1 Sélection de données à partir du DataFrame directement

3.1.1 Sélection de lignes

```
Selection = DataFrame.head(n)
```

`.head(n)` permet de sélectionner les n premières lignes *de données* du DataFrame.

Remarque : La ligne des noms de colonnes ne compte pas comme une ligne de donnée

```
Selection = DataFrame.tail(n)
```

`.tail(n)` permet de sélectionner les n dernières lignes *de données* du DataFrame.

```
Selection = DataFrame.at(n)
```

`.at(n)` permet de sélectionner la n-ième ligne *de données* du DataFrame.

```
Selection = DataFrame.loc(i:j)
```

`.loc(n)` permet de sélectionner de la i-ième à la j-ième ligne *de données* du DataFrame.

3.1.2 Sélection de colonnes

Sélectionner une seule colonne

```
Selection = DataFrame["col_1"]
```

Sélectionner plusieurs colonnes

```
Selection = DataFrame[ ["col_1","col_2"] ]
```

Remarque !!! Doubles crochets !!! On donne une liste des noms de colonnes voulues `["col_1","col_2"]` , pas seulement un enchainement de strings séparés par des virgules `"col_1","col_2"` !

3.1.3 Selection d'éléments spécifiques

Combinaison de commandes

On peut combiner les deux types de commandes. Par exemple :

```
Selection = DataFrame["col_1"].head(n)
```

Commandes `.at()` et `.loc()`

```
Selection = DataFrame.at(n, "col")
```

La commande `.at()` permet de sélectionner un élément spécifique du DataFrame, avec en premier argument la ligne `n` et en second argument la colonne `"col"`.

```
Selection = DataFrame.at(i:j, ["col_1" , "col_3"])
Selection = DataFrame.at(i:j,  "col_1" : "col_3" )
```

La commande `.at()` permet de sélectionner un élément spécifique du DataFrame, avec en premier argument les lignes `i` à `j` et en second argument les colonnes souhaitées ; `["col_1", "col_3"]` si l'on veut les colonnes 1 et 3 `"col_1" : "col_3"` si l'on veut les colonnes 1 à 3.

3.2 Autres commandes

```
DataFrame.columns
```

`.columns` permet de sélectionner l'ensemble des noms des colonnes.

```
DataFrame.dtypes
```

`.dtypes` permet de sélectionner le type de données de chaque colonne.