

Enseignant : Jaafar Chaaouri

Email: [Jaafar.chaaouri@fsm.rnu.tn](mailto:Jaafar.chaaouri@fsm.rnu.tn)

---

## ▼ TP2 : Pandas

### ▼ 1. Échauffement

Une simple exploration de la liste des participants.

Objectif : créer un petit DataFrame à partir d'un fichier csv et faire quelques comptages et visualisations simples.

Un jeu de données nettoyé `../data/liste_participants_clean.csv`. C'est un très petit jeu de données sans valeurs numériques.

#### ▼ 1.1 Importer les modules pandas, matplotlib.pyplot et seaborn avec les conventions habituelles de renommage

`# votre réponse`

`# commande magique pour que les graphiques s'affichent dans le notebook`  
`%matplotlib inline`

#### ▼ 1.2 Créer un DataFrame Pandas participants à partir du fichier `liste_participants_clean.csv`

`# votre réponse`

#### ▼ 1.3 Afficher les premières lignes et le nombre de participants

`# votre réponse`

Posons-nous des questions et essayons d'y répondre en interrogeant les données avec pandas :

- Les participants proviennent d'universités de quels pays ? Combien de pays représentés ?
- Combien de participants par pays ? Faire une visualisation.
- Compter le nombre de d'étudiants par niveau d'étude, faire une visualisation.

*(Remarques sur les bases de données et le langage de requêtes SQL)*

#### ▼ 1.4 Pays

`# votre réponse`

`# visualisation, votre solution :`

#### ▼ 1.5 Niveaux d'études

`# votre solution`

# visualisation, votre réponse

# Posez-vous d'autres questions et essayer d'y répondre

## ▼ 2. Exercice avec le jeu de données des prénoms américains

Tiré d'un exemple du livre *Python for Data Analysis* de Wes McKinney (O'Reilly's).

Nous allons explorer le jeu de données publiques des prénoms donnés à la naissance aux USA depuis 1880 (jusqu'à 2018)

Le jeu de données original se trouve là : <https://www.ssa.gov/oact/babynames/names.zip>, mais il est déjà téléchargé et décompressé dans ce dépôt et se trouve dans le répertoire `data/names/`.

### ▼ 2.1 Premier contact

Avec le navigateur de fichier de votre système ou celui de Jupyter inspecter le répertoire `data/names/`, lire la documentation (fichier pdf) et ouvrir un des fichier `txt`.

À l'aide de la fonction `read_csv` de Pandas, lire le fichier `yob1880.txt` et définir la variable de type `DataFrame` `names1880`. Définissez les noms des colonnes par: 'name', 'gender' et 'births'.

Afficher les premières lignes. Vous devez obtenir :

	name	gender	births
0	Mary	F	7065
1	Anna	F	2604
2	Emma	F	2003
3	Elizabeth	F	1939
4	Minnie	F	1746

# votre réponse

### ▼ 2.2 Aggrégation

Comptez le nombre de naissances de genre masculin et féminin respectivement. Vous pouvez utiliser une solution "pédestre" ou la méthode `groupby` s'appliquant au `DataFrame`.

# votre réponse

### ▼ 2.3 Concaténation

Jusqu'ici nous n'avons lu qu'un seul fichier correspondant aux noms de 1880. Maintenant nous voulons lire les données pour toutes les années et les regrouper dans un seul `DataFrame`.

#### ▼ 2.3.1 Écrire une fonction qui prend pour argument une année (int) et retourne un `DataFrame` des données correspondantes.

Comme ci-dessus mais ajoutez en plus la colonne 'year' au `DataFrame`.

```
# votre réponse
def load_year(year):
    """Retourne un DataFrame des noms avec les colonnes 'name', 'gender', 'births' et 'year'."""
    # complétez...
    return
```

#### ▼ 2.3.2 Construire une liste de `DataFrame` pour chaque année (utiliser une `list comprehension` ou une boucle ainsi que la fonction ci-dessus) puis utiliser la fonction `concat` de Pandas pour créer un seul `DataFrame` `names` à partir de cette liste.

Les années sont toutes représentées de 1880 à 2018 inclus.

# votre réponse

Afficher les premières lignes et le nombre de lignes. Vous devez trouver 1957046 lignes

# votre solution

### ▼ 2.3.3 Nombre de naissances par année

En utilisant à nouveau `groupby` faire un graphique du nombre de naissances par année. Faire une interprétation du graphique à l'aune de vos connaissances historiques.

# votre solution

### ▼ 2.3.4 Nombre de prénoms distincts donnés par année

Faire un graphique du nombre de prénoms distincts donnés par année

# votre solution

### ▼ 2.3.5 Évolution des noms, utilisations des tables pivot

Certains noms ont changé de genre majoritaire. Explorons par exemple *Allison* ou *Alison*.

la méthode `pivot_table` permet d'aggréger des données selon plusieurs dimensions à la fois.

```
names_to_check = ['Allison', 'Alison']
```

2.3.5.1 Filtrer les lignes de `names` pour ne garder que celles correspondant à Allison ou Alison. Vous pourrez utiliser la méthode `isin`.

# votre solution

```
births.head()
```



	name	gender	births	year
1753	Allison	M	6	1880
1532	Allison	M	9	1881
1679	Allison	M	9	1882
1582	Allison	M	12	1883
1940	Allison	M	7	1884

2.3.5.2 Avec la méthode `pivot_table` construire un DataFrame comptabilisant le nombre de naissances d'Allison et Alison pour chaque années (lignes) et pour chacun des genre F et M (colonnes).

La méthode `DataFrame.pivot_table` prends les arguments suivants :

- `values` = nom de la colonne des valeurs à agréger (le nombre de naissances),
- `index` = nom de la colonne dont les valeurs serviront d'aggrégateur par ligne (les années de naissance),
- `columns` = nom de la colonne dont les valeurs serviront d'aggrégateur par colonne (le genre),
- `aggfunc` = nom de la fonction d'aggrégation,
- `fill_value` = valeurs remplaçant les valeurs manquantes

# votre solution


```
births.head()
```



gender	F	M
year		
1880	0	6
1881	0	9
1882	0	9
1883	0	12

```
# normalisation
births = births.div(births.sum(axis=1), axis=0)
```

```
births.head()
```



gender	F	M
year		
1880	0.0	1.0
1881	0.0	1.0
1882	0.0	1.0
1883	0.0	1.0
1884	0.0	1.0

2.3.5.3 Afficher un graphique de la fréquence des Allison et Alison par année et pour chacun des genres F et M

```
# votre solution
```