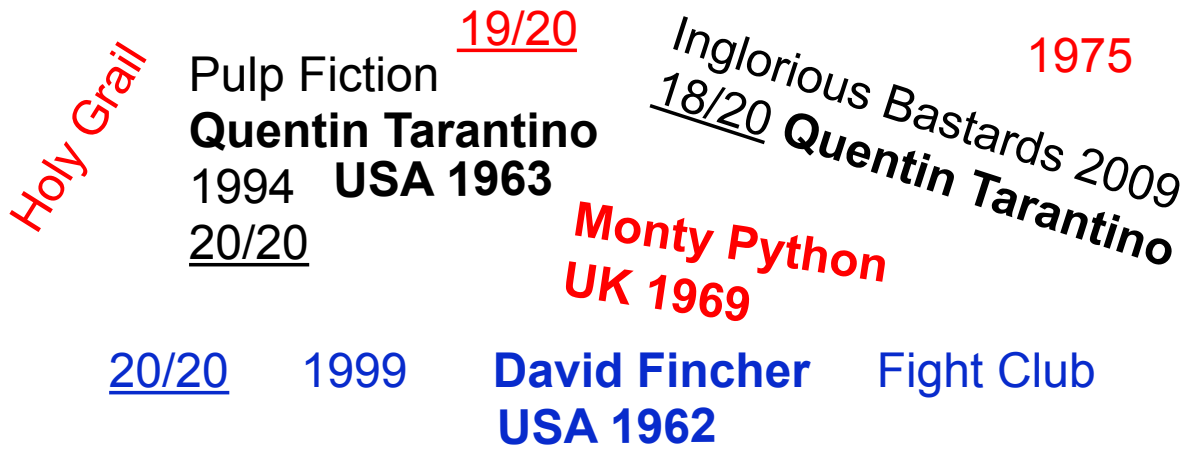
Chapitre 9 : Traitement de données

I. Introduction

1) Petit effort d'imagination et d'écriture

Imaginez que vous disposiez d'un certain nombre de données :



On vous demande d'organiser ces données de manière lisible afin de pouvoir les échanger avec une personne située à l'autre bout du monde.

Comment feriez-vous ? *Quelles informations enverriez-vous ? Comment organiseriez-vous ces informations ?*

2) Tableaux et bases de données

Il est clair que l'on va mettre ces données sous forme de tableau(x) avec des en-têtes claires et concises et, si possible de proposer un format **standardisé** contenant le moins de **répétition**. De cette manière, n'importe qui pourra lire ces deux tableaux que je lui envoie :

Clé	Titre	Réalisateur	Année de sortie	Note
1	Pulp Fiction	Quentin Tarantino	1994	20
2	Inglorious Bastards	Quentin Tarantino	2009	18
3	Holy Grail	Monty Python	1975	19
4	Fight Club	David Fincher	1999	20
5	Life of Brian	Monty Python	1979	17

Clé	Réalisateur	Origine	Date de naissance
1	Quentin Tarantino	USA	1962
2	Monty Python	UK	1969
3	David Fincher	USA	1963

Définitions importantes :

- Un **enregistrement** est une structure de données à laquelle on accède grâce à un nom ou à un identificateur.
- Un **descripteur** est le nom d'un en-tête.
- Une clé **primaire** est un identificateur unique à chaque enregistrement de la table ;
- Une clé **étrangère** est un identificateur qui **relie** un enregistrement à une table connexe.

Exemple :

- Un enregistrement s'écrit :
  { 'clé': 4, 'titre': 'Fight Club', 'réalisateur': 'David Fincher', 'année': 1999, 'note': 20}
- 3 est la clé primaire de l'enregistrement
  { 'clé': 3, 'titre': 'Holy Grail', 'réalisateur': 'Monty Python', 'année': 1975, 'note': 19}

❖ Il n'existe pas de clé étrangère ici car les deux tableaux ne sont pas reliés entre eux. Pour en introduire une, il faut donc changer le premier tableau :

Clé	Titre	Réalisateur	Année de sortie	Note
1	Pulp Fiction	1	1994	20
2	Inglorious Bastards	1	2009	18
3	Holy Grail	2	1975	19
4	Fight Club	3	1999	20
5	Life of Brian	2	1979	17

2 est ici la clé étrangère des enregistrements 3 et 5.

Ce type d'organisation s'appelle "base de données relationnelles" et sera vue en Terminale.

Voir aussi : Exercice 1 et 2

"Ok, non mais c'est cool. Mais moi, j'ai **jamais envoyé de tableaux à personne**."

Réponse : Déjà, ce n'est pas tout à fait vrai. Vous avez sans doute déjà envoyé des centaines — voire des milliers — de tableaux, sans même le savoir. Et, puis, même si vous n'avez jamais envoyé de tableaux, vous en avez déjà reçu beaucoup !

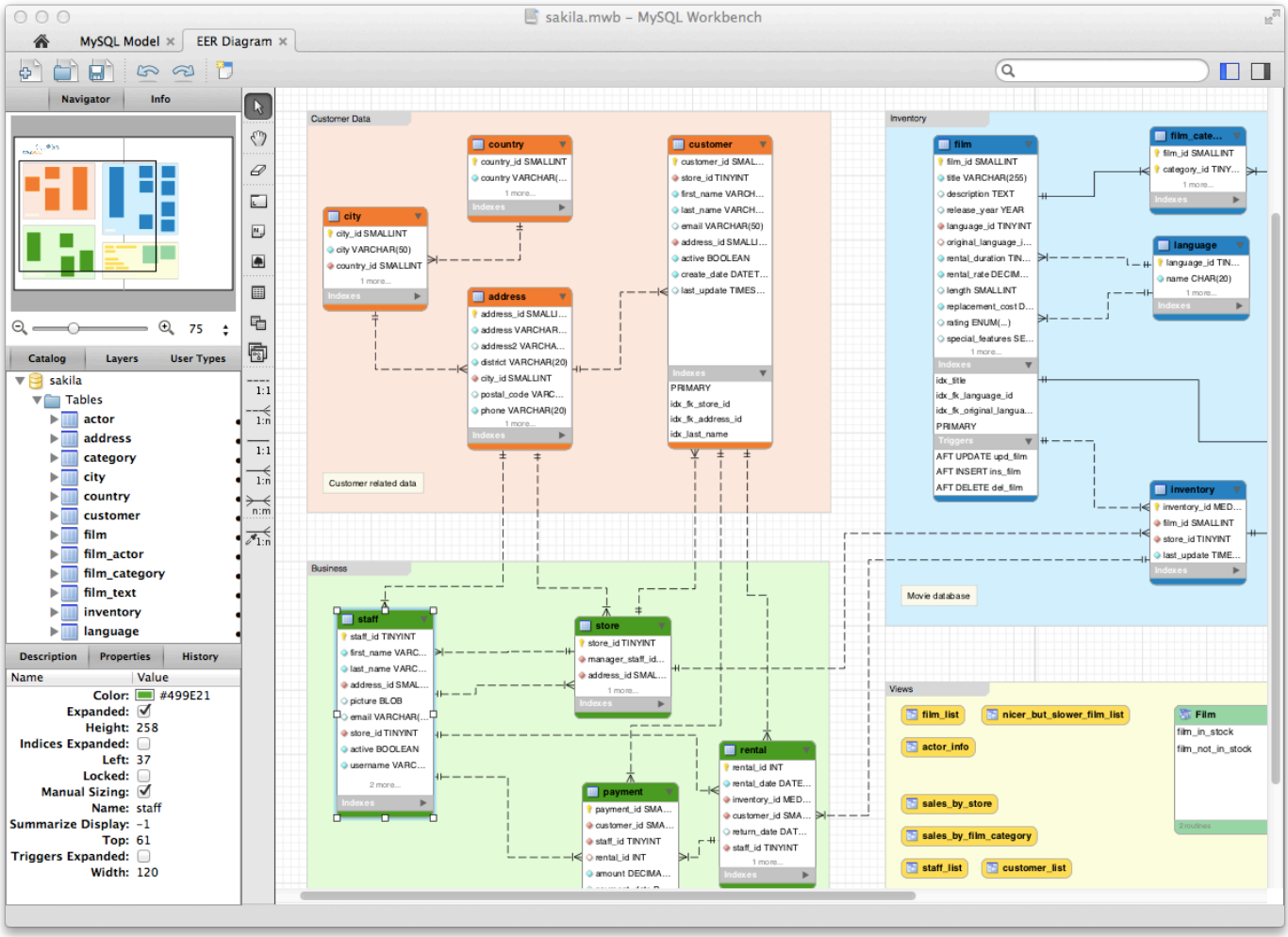
En effet, dans le chapitre précédent, nous avons dit que les serveurs web étaient pratiquement toujours clients de serveurs gérant uniquement des bases de données et stockaient leurs informations importantes dans ces bases. Ces bases de données contiennent ni plus ni moins que de gigantesques tableaux standardisés bourrés d'informations…

Les grandes bases de données sont organisées grâce à des langages de gestion de base de données (comme MySQL ou PostgreSQL). Grâce à l'utilisation de nombreuses clés étrangères, il devient possible de retrouver très rapidement un enregistrement donné.

Toutefois, très souvent, pour afficher une page dans un navigateur, il suffit de charger une toute petite partie de la base de données. Le serveur web vous transmet donc des **petits tableaux** qui sont construits **sous la forme de fichiers texte** simplifié : les deux standards dominant à l'heure actuelle sont le format JSON (JavaScript Object Notation) et le CSV (Comma Separated Value).

Exemple de base MySQL :

On remarque l'extrême importance des clés secondaires et les nombreuses relations entre tableaux.



Cette année, nous allons nous intéresser tout particulièrement au format CSV.

3) Fichiers CSV

Définition :

Le format CSV est couramment utilisé pour importer ou exporter des données d'une feuilles de calcul d'un tableur. C'est un fichier texte dans lequel **chaque ligne correspond à une ligne du tableau**, les colonnes étant séparées par des virgules (ou des points-virgules pour les francophones).

Exemple :

prenom, nom, date_naissance
Poussin, Piou, 2012
Crazy, Frog, 2005
René, LaTaupe, 2009

constitue un tableau au format csv . Pour vous en convaincre, suivez les étapes ci-dessous :

- Copier/coller les quatre lignes ci-dessus dans un **éditeur de texte** (Geany, VSCode ou Spyder)
- Enregistrer ce texte sous le nom : testCSV.csv.
- Ouvrez testCSV.csv avec Excel ou LibreOffice.

Vous devriez vous apparaitre la belle feuille de calcul ci-contre :

prenom	nom	date_naissance
Poussin	Piou	2012
Crazy	Frog	2005
René	LaTaupe	2009

Voir aussi : Exercice 3

Comment faire pour représenter un fichier CSV en Python ?

II. Dictionnaires Python

1) Introduction

Utiliser des listes semble être une bonne idée mais, très rapidement, on risque de se retrouver avec des données comme cela (cf Pronote) :

```
[['Prénom;Né;Sexe;Classe;Entrée;Sortie;Option 1;Option 2;Option
3;Option 4;Option 5'], ['Eloise;2A03;G;1G03;02/09/2019;;ANGLAIS
LV1;ITALIEN LV2;MATHEMATIQUES;Numérique et sciences informatiques;
PHYSIQUE-CHIMIE'], ['Valentin;2003;G;1G03;02/09/2019;;ANGLAIS
LV1;ITALIEN LV2;MATHEMATIQUES;Numérique et sciences informatiques;
PHYSIQUE-CHIMIE'], ['Léo;2002;G;1G 08;02/09/2019;;ANGLAIS
LV1;ALLEMAND LV2;Numérique et sciences informatiques;LITT. ANGLAIS;
MATHEMATIQUES'], ['Thibaud;2003;G;1G03;02/09/2019;;ANGLAIS
LV1;ESPAGNOL LV2;Numérique et sciences informatiques;MATHEMATIQUES;
PHYSIQUE-CHIMIE'], ['Julia;2003;F;1G11;02/09/2019 ;;ANGLAIS
LV1;ESPAGNOL LV2;LITT. ANGLAIS;Numérique et sciences informatiques;
PHYSIQUE- CHIMIE'], ['Julien;2002;G;1G06;02/09/2019;;ANGLAIS
LV1;ESPAGNOL LV2;Numérique et sciences informatiques;MATHEMATIQUES;
LITT. ANGLAIS'], ['Sylvain;2003;G;1G09;02/09/2019;;ANGLAIS
LV1;ESPAGNOL LV2;Numérique et sciences informatiques;LI……..]
```

Imaginons : on souhaite accéder à l'Option 3 des élèves ayant anglais comme Option 1 …
Je vais devoir dire : "quand le 6ème élément de la sous-liste est égal à 'ANGLAIS', lis le 8ème élément STP" ➠ **Ignoble**.
Il serait tellement plus facile de dire : "quand l'Option 1 est égal à ANGLAIS, lis l'Option 3".

L'inconvénient des listes dans ce cas est qu'il est pénible d'y rechercher des éléments. En effet, elles sont basées sur la notion **d'indice** et non sur la notion de **contenu** alors que les bases de données sont précisément basées sur la notion de contenu.

Pour faire une chose pareille, il faut nous appuyer sur une autre structure de données disponible en Python et basée sur les **contenus** : les dictionnaires.

2) Manipulation de dictionnaires

Un dictionnaire Python est composé de deux parties : la clé et la valeur. Étudions un exemple de dictionnaire pour mieux comprendre :

```
monDico = {"nom": "LaTaupe", "prenom": "René", "naissance": "2009"}
```

Syntaxe :
- ❖ un dictionnaire se caractérise par des accolades {} ;
- ❖ les clés sont ici "nom", "prenom" et "naissance". Les clés sont de types **`str`** ou **`int`**.
- ❖ la valeur associée à la clé "nom" est "LaTaupe", à la clé "prenom" est "René" et à la clé "naissance" "2009". Les valeurs peuvent être de n'importe quel type (y compris liste ou dictionnaire !)

Initialisation :
Il est possible d'initialiser un dictionnaire vide puis de le remplir ainsi :

```
monDico = {}     # monDico = dict() est une autre syntaxe possible.
monDico["nom"] = "LaTaupe"
monDico["prenom"] = "René"
monDico["naissance"] = "2009"
```

❐ — À faire vous-même **1** —

Dans Geany, VSCode ou Spyder, tapez le code suivant et exécutez-le :

```
monDico = {"nom": "LaTaupe", "prenom": "René", "naissance": "2009"}
print("Le type de mon Dico est ", type(monDico))
print(monDico)
```

Quel est le type affiché ? _____

❐ — À faire vous-même **2** —

Dans Geany, VSCode ou Spyder, créez un fichier appelé **dicoTest.py**, tapez le code suivant puis exécutez-le :

```
monDico = {"nom": "LaTaupe", "prenom": "René", "naissance": "2009"}

print( 'Bonjour je suis {0} {1} !'.format(monDico["prenom"],
monDico["nom"]) )
```

Quel est le résultat attendu après l'exécution de ce programme ?

_____

_____

Vérifiez votre réponse.

- En réutilisant la même syntaxe, ajoutez la phrase : "Je suis né en ……" .
- Testez votre programme en changeant le nom de René LaTaupe en Sergent LaTaupe.

Les dictionnaires sont des objets mutables (=pouvant être modifiés).

Rappel :

- les listes sont aussi des objets mutables et sont obtenus avec des crochets [] ;
- les tuples sont des objets immutables obtenus à l'aide de parenthèses ().

❐ — À faire vous-même **3** —

Nous allons modifier le fichier **dicoTest.py.**
En suivant la méthode d'initialisation proposée page 6, rajoutez à votre dictionnaire monDico une clé "lieu_naissance" à laquelle on affectera la valeur "Copacabana".

Le code ressemblera donc à :

```
monDico = {"nom": "LaTaupe", "prenom": "René", "naissance": "2009"}
# instruction à rajouter :
# …… …… …… …… …… …… …… …… ……
print( 'Bonjour je suis {0} {1} ! Je suis né à
{2}'.format(monDico["prenom"], monDico["nom"],
monDico["lieu_naissance"]) )
```

Remarque : Il est maintenant clair qu'on peut accéder au contenu d'un dictionnaire en tapant : nomDuDictionnaire["nomDuContenu"] . C'est exactement ce que l'on souhaite faire pour lire du CSV.

❐ — À faire vous-même **4** —

Nous allons encore modifier le fichier **dicoTest.py.**

Créez un nouveau dictionnaire qui va s'appeler mesFruits :

```
mesFruits = {"poire": 3, "pomme": 4, "orange": 2}
```

Les dictionnaires sont mutables, donc modifiables.

❖ J'aime bien les oranges. Soustraire 2 oranges à mesFruits en utilisant :

```
mesFruits["orange"] = mesFruits["orange"] – 2
```

❖ Raccourcir cette instruction en évitant la répétition de `mesFruits["orange"]`

❖ Afficher ensuite le nombre d'orange contenu dans mesFruits :

```
print("Il y a {0} oranges dans le bol!".format(……)
```

❖ Normalement, vous avez du trouvé qu'il n'y a plus d'oranges dans le bol… Supprimons donc l'entrée `orange`. Pour faire cela, nous utilisons l'instruction **`del`** :

```
del mesFruits["orange"]
```

Il est possible de parcourir un dictionnaire à l'aide d'une boucle **for**. Par rapport à une liste, ce parcours peut se faire selon les **clés** ou les **valeurs**.

Commençons par parcourir les **clés** à l'aide de la méthode **`keys()`**.

❐ — À faire vous-même **5** —

❖ Rajoutez dans le fichier **dicoTest.py :**

```
monBol = {"poire": 3, "pomme": 4, "orange": 2}
print("liste des fruits :")
for fruit in monBol.keys():
   print(fruit)
```

❖ Modifier la boucle for afin d'afficher à chaque passage dans la boucle la clé et la valeur associée. On doit obtenir :

```
poire 3
pomme 4
orange 2
```

❖ Essayer de supprimer la méthode **`keys()`** . Que remarquez-vous ?

_____

Remarque : La méthode **`keys()`** est donc assez inutile pour parcourir les clés…

Les **valeurs** d'un dictionnaire peuvent être parcourues en utilisant la méthode **`values()`**.

❐ — À faire vous-même **6** —

❖ À la suite de votre travail précédent, dans le fichier **dicoTest.py**, ajoutez :

```
print("quantité de fruits :")
for nombre in monBol.values():
   print(nombre)
```

❖ Essayez de modifier la boucle afin d'obtenir le même affichage que dans le **"À faire vous-même 5"**. Est-ce possible ?

_____

Enfin, les **clés** et les **valeurs** (appelées **champs**) peuvent être parcourues simultanément en utilisant la méthode **`items()`**.

❐ — À faire vous-même **6** —

❖ Finalement, dans le fichier **dicoTest.py**, rajoutez à la suite de votre travail **:**

```
print("liste des fruits et quantité :")
for (fruit, nombre) in monBol.items():
   print(fruit, nombre)
```

❖ Remarquez la syntaxe de la boucle for. Avez-vous déjà vu cette syntaxe ? Rappelez dans quel contexte.

_____

_____

Remarque importante : un dictionnaire Python a exactement le format JSON proposé en Javascript.