Comment se connecter à une base de données SQL à l'aide de python

Pour effectuer cette connexion, on peut utiliser SQLITE3 déjà installé sur Python. Donc, pour cela il faut importer la commandante **sqlite3** sur un éditeur Python. Dans notre cas, nous utilisons **Google Colab** pour sa simplicité et sa praticabilité.

Une fois que la librairie **sqlite3** est importé, on peut utiliser la fonction '**connect()'** de cette librairie. Elle requiert un seul paramètre, et ce paramètre est le fichier de la base de données dont nous souhaitons nous connecter. Cette fonction permet de maintenir la connexion à la base de données. Cela nous permettra de réaliser des opérations sur cette base de données via Python.

Pour se connecter à une base de données sur Python, on utilise la commande ci-dessous :

$$\boldsymbol{conn = sqlite3.connect('name\_data')}$$

Pour contrôler une requête sur SQL, il faut utiliser un curseur de la classe curseur. Ce curseur nous permettra d'exécuter une requête au près de la base de données. Il va aussi nous permettre d'analyser les résultats de la base de données et nous permettra de convertir les résultats en objets Python. Et enfin il nous permettra de stocker les résultats dans une variable locale. Donc, après avoir exécuter la requête et convertit les résultats en liste de tuples. L'instance du curseur stocke la liste en tant que variable locale.

Un tuple c'est une structure que Python utilise pour représenter une séquence de valeurs un peu comme une liste. Les elements d'un tuple sont non modifiables, en revanche ceux d'une liste sont modifiables (ex : tuple ()). Les tuples sont plus rapides que les listes ce qui est très utile quand on utilise des grandes bases de données.

Pour se connecter à un curseur on utilise :

$$\boldsymbol{cursor = connexion.cursor()}$$

Il faut enregistrer ou stocker notre première requête sur python dans une variable **qery** en tant que chaine de caractères**.** Par exemple on a :

$$query = "select * from recent\_grads;"$$

Par la suite on va juste exécuter notre variable **query**

$$cursor.execute(query)$$

Maintenant ce qu'il nous reste à faire, c'est de récupérer les résultats de cette variable locale en utilisant la commande ci-dessous :

$$results = cursor.fetchall()$$

Maintenait on va afficher la variable **results**

La méthode $fetchone()$ va chercher un élément de résultats, par contre la méthode $fetchall()$ va chercher tous les résultats. En outre, $fetchmany(n)$ permet de chercher le nombre **n** précisé par avance.

NB : la méthode $fetchone()$ si on l'utilise plusieurs, elle retournera à chaque fois le tuple suivant et ainsi de suite.

SQLITE restreint la connexion à la base de données, c'est la raison pour laquelle nous devons stopper la connexion à la base de données quand on a fini de travailler. Donc, ça permettra à une autre personne de se connecter à la base de données.

Pour se connecter à **SQLITE** depuis une terminale, veuillez cliquer **:**

$$sqlite3\ name\_table$$

Une fois que nous avons récupéré la table on peut effectuer des manipulations dessus avec différentes requêtes.

. $\boldsymbol{help}$: il permet d'avoir plusieurs commandes SQL

. $\boldsymbol{tables}$: il permet d'afficher le contenu de la base de données

. $\boldsymbol{header\ on}$: il permet d'avoir accès à l'entête des colonnes

. $\boldsymbol{header\ off}$: cette commande supprime les entêtes de colonnes

Pour lancer un fichier Python depuis le terminal veuillez taper la commande suivante :

$$\boldsymbol{python\ name\_fichier}$$

Pour lire une table SQL avec pandas, nous utiliserons la fonction suivante :

$$\boldsymbol{pd.read\_sql\_query(\text{'}requ\text{ê}te\ SQL\text{'}, objet\ de\ connexion)}$$

C'est une fonction de la librairie pandas qui nous permettra de lire nos données dans un DataFrame. Cette fonction va prendre deux paramètres : le premier c'est la requête SQL et le deuxième c'est l'objet de connexion à la base de données.

$\boldsymbol{SUM}$: c'est une fonction qui permettra d'additionner toutes les valeurs d'une colonne