Python – SQL

Préparation

L'objectif de ce TP est de réaliser une interface graphique pour un Pokédex. Toute la partie concernant l'interface est déjà faite. Il faut "juste" réaliser les fonctions faisant les requêtes à la base de données.

Le gros bouton bleu en haut à gauche permet de réinitialiser la liste des Pokémons. Cette liste permet de choisir le Pokémon à afficher soit en double-cliquant dessus, soit en le sélectionnant et en cliquant sur le bouton vert. Les deux autres boutons permettent d'accéder au Pokémon dont il est l'évolution, s'il y en a un, et à ses évolutions s'il en a. Un dernier bouton permet d'entendre le cri du Pokémon.
Enfin, le nombre de Pokémons dans le Pokédex est affiché en bas de la partie de gauche.
Les fichiers nécessaires se trouvent dans le dossier `pokedex` dans `Echange`.

Exercice 1 :

- Copier le dossier `pokedex` dans vos documents.
- Ouvrir `pokedex.py` dans `Thonny` et l'exécuter pour voir l'affichage du Pokédex. Vous pourrez remarquer que les boutons ne font rien pour l'instant. Il n'y a rien à changer dans ce fichier.
- Ouvrir `pokemons.py`, qui est le fichier qu'il faudra modifier.

Utilisation de SQL dans Python

Afin d'utiliser une base de donnée, il faut :

- charger le module `sqlite3` ;
- créer une connexion à une base de donnée ;
- créer un curseur qui permettra de faire les requêtes ;
- faire une requête ;
- récupérer les résultats.

Voici un exemple :

```
>>> import sqlite3
>>> bdd = sqlite3.connect("pokemon.db")
>>> curseur = bdd.cursor()
>>> curseur.execute("""SELECT * FROM pokemon WHERE nom = "Pikachu";""")
<sqlite3.Cursor object at 0x7f2497e97960>
>>> curseur.fetchone()     # on récupère le premier résultat
(25, 'pikachu', 'Pikachu', 35, 55, 40, 50, 50, 90, 172, 'Il lui arrive... ')
>>> curseur.fetchone()     # il n'y a plus rien
>>>
```

La méthode `curseur.fetchone()` renvoie les résultats un par un à chaque fois qu'il est appelé. On pourra l'utiliser lorsqu'il n'y en a qu'un qui est attendu.
À contrario, `curseur.fetchall()` renvoie tous les résultats dans une liste.

```
>>> curseur.execute("""SELECT pokemon.nom FROM pokemon
                    JOIN est_de_type ON pokemon.id = est_de_type.id_pokemon
                    JOIN type ON type.id = est_de_type.id_type
                    WHERE type.nom = "Feu"
                    ORDER BY pokemon.nom;""")
<sqlite3.Cursor object at 0x7f2497e97960>
>>> curseur.fetchall()
[('Aflamanoir',), ('Arcanin',), ('Boumata',), ('Braisillon',), ('Braségali',),
 ('Camérupt',), ('Caninos',), ('Chamallot',), ('Chartor',), ('Chimpenfeu',),
 ('Darumacho',), ('Darumarond',), ('Dracaufeu',), ('Démolosse',), ('Entei',),
 ..., ('Volcanion',), ('Volcaropod',)]
```

Pour pouvoir facilement écrire les requêtes, il est possible d'utiliser """ pour ouvrir et fermer les requêtes. Il est même possible de faire des f-strings de cette manière :

```
>>> pika = "Pikachu"
>>> curseur.execute(f"""SELECT vitesse FROM pokemon WHERE nom = "{pika}";""")
<sqlite3.Cursor object at 0x7f2497e97960>
>>> curseur.fetchone()
(90,)
```

On peut remarquer au passage que le résultat n'est pas juste un nombre, mais un tuple de longueur 1. Il faut faire attention pour récupérer la valeur.

```
>>> curseur.execute(f"""SELECT vitesse FROM pokemon WHERE nom = "{pika}";""")
<sqlite3.Cursor object at 0x7f2497e97960>
>>> reponse = curseur.fetchone()
>>> vitesse = reponse[0]
>>> curseur.execute(f"""SELECT nom FROM pokemon WHERE vitesse>{vitesse};""")
<sqlite3.Cursor object at 0x7f2497e97960>
>>> [rep[0] for rep in curseur.fetchall()]
['Dracaufeu', 'Roucarnage', 'Rattatac', 'Rapasdepic', 'Raichu', 'Feunard',
 'Taupiqueur', 'Triopikeur', 'Persian', 'Colossinge', 'Arcanin', ...]
```

L'usage de variables permet de faire plusieurs requêtes simples au lieu d'une seule, plus complexe.

Réalisation du Pokédex

La base de données `pokemons.db` est constituée des 3 tables suivantes :

- **pokemon**(id, name, nom, pv, attaque, defense, attaque_speciale, defense_speciale, vitesse, evolution_de, description)
- **type**(id, nom, name)
- **est_de_type**(id_pokemon, id_type)

Vous pouvez ouvrir la base avec un logiciel pour pouvoir l'inspecter davantage. Les attributs "nom" sont en français et "name" en anglais. L'attribut "evolution_de" est le numéro (id) du Pokémon dont il est l'évolution.
Afin de réaliser les fonctions nécessaires au projet, vous pouvez utiliser les fonctions `obtenir_une_reponse(requete)` qui renvoie la première réponse à la requête et `obtenir_toutes_les_reponses(requete)` qui renvoie la liste de toutes les réponses à la requête.

Exercice 2 : Compléter la fonction `nombre_pokemons()` qui renvoie le nombre de Pokémons présents dans la base.

Exercice 3 : Compléter la fonction `recherche_pokemons()` qui renvoie la liste des noms de Pokémons en français. Attention, certains n'ont pas de noms en français et ne doivent pas être affichés. Ces Pokémons ont un nom français qui est `'NULL'`.

Exercice 4 : Compléter la fonction `recherche_attributs_pokemon(nom_pokemon)` qui renvoie un objet de type `Pokemon` correspondant au nom donné. La liste de valeurs à passer au constructeur est dans le même ordre que les attributs de la table `pokemon`.

Exercice 5 : Compléter la méthode `trouve_evolution_de` qui associe le nom du Pokémon dont celui étudié est l'évolution, s'il y en a un, et `'---'` sinon.

Exercice 6 : Compléter la fonction `recherche_evolutions(nom_pokemon)` qui renvoie la liste des noms des évolutions du Pokemon dont le nom est donné en paramètre. S'il n'y en a pas, la liste est vide.

Exercice 7 : Compléter la fonction `recherche_types(nom_pokemon)` qui renvoie la liste des noms de types du Pokémon donné.

Exercice 8 : Le Pokédex est maintenant complet et vous pouvez l'utiliser.

D'après une activité de S. Chanthery