EXERCICE 3 (8 points)

Cet exercice porte sur les dictionnaires et leurs algorithmes associés, le traitement de données en table, la sécurisation des communications et la programmation en général.

Lorsque l'énoncé demande la manipulation de la structure de données abstraites `liste`, on utilisera les `list` en Python avec la méthode `append`.

Un club de judo souhaite développer un système d'informations pour faciliter les traitements administratifs en cours d'année (inscriptions, communication, compétitions…).

Cet exercice comporte 2 parties indépendantes.

Partie A

On pourra utiliser les mots du langage SQL suivants : `SELECT, FROM, WHERE, JOIN ON, INSERT INTO, VALUES, COUNT, DELETE.`

Le club de Judo souhaite proposer à ses adhérents une location de kimonos. Pour cela, il décide de mettre en place une base de données contenant les relations `adherent`, `kimono` et `location`.

Le schéma relationnel, où les clés primaires sont soulignées et les clés étrangères sont précédées du symbole #, est le suivant :

`adherent(` <u>`numero-licence`</u> `, taille-adherent, nom, prenom)`

`kimono(` <u>`id-kimono`</u> `, taille-kimono)`

`location(` <u>`#numero-licence`</u> `,` <u>`#id-kimono`</u> `, debut, fin)`

L'attribut `id-kimono` est un nombre entier. Les attributs `taille-adherent` et `taille-kimono` sont des nombres entiers dont l'unité est le centimètre et qui sont tous multiples de 10 (100, 110, 120, 130,…).

Les attributs `debut` et `fin` sont des dates au format chaîne de caractères 'AAAA-MM-JJ'. Tous les attributs doivent être renseignés et valides mais l'attribut `fin` peut éventuellement être égal à la chaîne de caractères vide `''` pour les kimonos en cours de location.

On rappelle qu'en langage SQL la fonction d'agrégation `COUNT` permet de compter un nombre d'enregistrements. Par exemple, pour déterminer le nombre d'adhérents du club, on peut utiliser la requête suivante :

```
SELECT COUNT(numero-licence) FROM adherent
```

1. Écrire une requête SQL permettant de connaître le numéro des kimonos **en cours de location**.

2. Écrire une requête SQL permettant de connaître le nombre de kimonos de taille 130 cm possédés par le club.

3. Écrire la requête SQL permettant de connaître le nom et le prénom de l'adhérent qui possède le kimono 42 uniquement si ce kimono est en cours de location. Cette requête ne doit renvoyer que le nom et le prénom de l'adhérent à qui il est actuellement loué et pas celui de tous ceux qui l'ont éventuellement loué par le passé.

Dans une requête SQL, il est possible de modifier un attribut entier avec une expression du type `a = a + 5`.

4. À la fin de l'année, pour anticiper les locations de l'année à venir, le club de judo modifie arbitrairement la taille de tous ses adhérents mesurant strictement moins de 160 cm en leur rajoutant 10 cm. Écrire une requête SQL permettant de réaliser cette opération.

5. Le kimono numéro 25 a été déchiré lors d'un combat. Écrire les requêtes SQL permettant de le supprimer de la base de données.

Partie B

Un adhérent du club de judo est décrit par les descripteurs :

`nom` : nom de famille de l'adhérent, la donnée est une chaîne de caractères (on supposera dans l'exercice que tous les noms de famille font plus de cinq caractères) ;

`prenom` : prénom de l'adhérent, la donnée est une chaîne de caractères ;

`annee` : année de naissance de l'adhérent, la donnée est une chaîne composée de 4 caractères ;

`mois` : mois de naissance de l'adhérent, la donnée est une chaîne de deux caractères parmi '0123456789' : '01' (pour janvier) et '12' (pour décembre) ;

`jour` : jour de naissance de l'adhérent, la donnée est une chaîne de deux caractères parmi '0123456789' : '01' (si l'adhérent est né le premier jour du mois) et '31' (si l'adhérent est né le 31 du mois) ;

`sexe` : sexe de l'adhérent, la donnée est un caractère valant soit 'F' si l'adhérent est de sexe féminin, soit 'M' s'il est de sexe masculin.

Pour différencier les pratiquants du judo en France, la fédération française de judo, attribue à chaque pratiquant un « numéro de licence ». Il s'agit d'une chaîne de 16 caractères dont :

- le premier caractère vaut soit 'F' soit 'M' selon le sexe de l'adhérent ;
- les huit caractères suivants correspondent à la date de naissance au format 'JJMMAAAA' ;

- les cinq caractères suivants correspondent aux cinq premières lettres du nom de famille de l'adhérent en majuscules ;
- les deux derniers caractères correspondent à un nombre entre 01 et 99. Ils permettent à la fédération française de judo de différencier des pratiquants, dans le cas rare où ils ont le même sexe, la même date de naissance et les cinq premières lettres de leur nom identiques.

Exemple : Clémence et Stéphanie Dupond, sœurs jumelles nées le 03/07/1997, se voient attribuées les numéros de licences respectifs 'F03071997DUPON01' et 'F03071997DUPON02'

6. Déterminer un numéro de licence possible pour Eddie Nirrer né le 12/10/2021.
7. Un adhérent a le numéro de licence 'M23091974MARTI01'. Donner sa date de naissance et un nom de famille possible.

Pour manipuler efficacement sa base d'adhérents, le club de judo implémente une table par un tableau nommé `tab_adherents` contenant des dictionnaires en langage Python. Chaque dictionnaire correspond à un adhérent du club.

Les clés des dictionnaires, communes à tous les dictionnaires, correspondent aux descripteurs utilisés pour cette table et sont `nom`, `prenom`, `annee`, `mois`, `jour`, `sexe`, `numero-licence`.

On donne en illustration les deux premiers éléments de ce tableau de dictionnaires,

```
1 tab_adherents= [
2 {'nom': 'DUPOND', 'prenom' : 'CLEMENCE', 'annee' : '1997',
3  'mois' : '07', 'jour' : '03', 'sexe' : 'F',
4  'numero-licence' : 'F03071997DUPON01'},
5 {'nom': 'DUPOND', 'prenom' : 'STEPHANIE', 'annee' : '1997',
6  'mois' : '07', 'jour' : '03', 'sexe' : 'F',
7  'numero-licence' : 'F03071997DUPON02'},
8 ...]
```

8. Donner, sans justifier, la valeur à laquelle on accède avec l'instruction `tab_adherents[1]['prenom']`.
9. Écrire l'instruction permettant d'obtenir la valeur 'F03071997DUPON01'.

La direction du club souhaite connaître le nombre de ses adhérents nés une année donnée. Elle demande d'écrire une fonction `nombre_adherents` qui prend en entrée un tableau de dictionnaires `table` correspondant aux adhérents et une chaîne de 4 caractères `annee` correspondant à une année et qui renvoie en sortie l'entier correspondant au nombre d'adhérents nés cette année.

10. Recopier et compléter la fonction suivante pour permettre de répondre à cette problématique.

```
1 def nombre_adherents(table, annee):
2     compteur = ...
3     for adherent in table:
4         if ...:
5             ...
6     ...
```

La direction souhaite également connaître les adhérents les plus âgés du club. Elle nous demande d'écrire une fonction `adherent_plus_age` qui prend en entrée un tableau de dictionnaires `table` correspondant aux adhérents et qui renvoie en sortie une liste de dictionnaires correspondant aux adhérents les plus âgés du club. La comparaison se restreindra uniquement à l'année de naissance. Il peut donc y avoir un ou plusieurs adhérents qui sont nés la même année et qui sont donc les adhérents les plus âgés. On pourra supposer qu'aucun adhérent n'est né après l'année `'2024'` et qu'il y a toujours au moins un adhérent dans le club. On admet que l'on peut comparer en Python deux chaînes de caractères représentant deux années et que le résultat est le même que si ces années étaient des entiers. Par exemple, on a `tab_adherents[0]['annee'] < '2025'`, car la première adhérente est née en 1997.

11. Écrire en Python cette fonction `adherent_plus_age`.

Pour tester la cohérence de sa base et des erreurs causées par un mauvais enregistrement de numéro de licence, le club souhaite programmer une fonction `verification_licence`. Cette fonction prend en paramètre un dictionnaire `adherent` correspondant à un adhérent du club et renvoie `True` si le numéro de licence de l'adhérent est conforme à son sexe, sa date de naissance et son nom de famille et `False` sinon. On pourra supposer que l'on dispose d'une fonction `extraire` qui prend en paramètre une chaîne de caractères `s` et deux indices `i` et `j` avec `0 <= i <= j <= len(s)` et qui renvoie la sous-chaîne de `s` entre les indices `i` inclus et `j` exclu. Par exemple :

```
>>> no = 'F03071997DUPON01'
>>> extraire(no, 0, len(no))
'F03071997DUPON01'
>>> extraire(no, 5, 9)
'1997'
```

12. Écrire la fonction `verification_licence` en Python.