TP : Utilisation de Python pour gérer des bases de données

Vous devez rendre des programmes commentés et expliquer vos choix de conception.

1. Bases de données sous Python

1.1 Instructions sous Python

La librairie qu'il faut utiliser est **sqlite3**
Cette librairie permet de créer et d'interroger des bases de données en utilisant le langage SQL

Référence : https://docs.python.org/2/library/sqlite3.html

Les bases créées avec cette librairie enregistrées sous forme de fichier .db

Un fichier .db s'ouvre en créer un objet de type *Connection*

```
import sqlite3
conn = sqlite3.connect('example.db')
```

Un *Curseur* permet ensuite d'accèder au contenu de l'objet *Connection* , soit pour insérer/supprimer des données (INSERT, DELETE FROM), soit pour interroger la base (SELECT)

```
cur = conn.cursor()
```

On se concentrera pour ce type à l'interrogation de données.

```
cur.execute("SELECT Nom,Prenom FROM Employe WHERE Nom='Abellard')");
```

Grâce à la méthode execture, le résultat de la requête est rangée dans un objet de type *Curseur*.
Pour pouvoir lire les lignes de la requête, deux méthodes possibles de l'objet *Curseur*

- `fetchone()` qui retourne le résultat d'une ligne de la requête dans une liste
- `fetchall()` qui retourne en une fois toutes les lignes résultants de la requête dans une « liste de listes ».

Dans les deux cas, une boucle sera nécessaire pour accéder à chaque donnée séparément.

Par exemple avec `fetchall()` :

```
data = cur.fetchall()
for ligne in data :
     print("l'enseignant ",ligne[0]," a pour prénom ",ligne[1])
```

Avec fetchone() :

```
while True:
     ligne = cur.fetchone()
     if row == None:
          break
     print("l'enseignant ",ligne[0]," a pour prénom ",ligne[1])
```

1.2 SQL : groupement et calcul

Soit le tableau suivant :

Nom	*Devoir*	*Note*
'Maxime'	'Python'	12.5
'Zoé'	'Python'	14.0
'Erwan'	'Automates'	11.5
'Zoé'	'Automates'	16.5
'Erwan'	'Python'	13.0
'Maxime'	'Automates'	11.5

Pour obtenir la moyenne par nom de personne, il faut regrouper les lignes pour lesquelles le nom est le même, puis calculer la moyenne pour chaque groupement. Ceci se fait avec l'expression GROUP BY

Ainsi,

SELECT Nom, AVG(Note) FROM Tableau GROUP BY Nom ;

affichera

'Maxime'	12.0
'Zoé'	15.25
'Erwan'	12.25

De même :
- SELECT Nom, AVG(Note) FROM Tableau GROUP BY Devoir; permet d'afficher les moyennes par devoir
- SELECT Nom, COUNT(*) FROM Tableau GROUP BY Nom ; permet d'afficher le nombre de notes par étudiant.

On peut ajouter des WHERE et ORDER BY en cas de nécessité.

2. Sujet à traiter

La base de données que nous allons étudier est dérivée du fichier JASON_PC_FORTEM_METEOTHYLUXV1_HSOL_20150412_151727.log (cf TP2) dont seules les lignes de données (« $ ») ont été conservées. Elle contient une seule table nommée JasonBase avec les colonnes suivantes :

Idx INT, Jour DATE, Heure TIME, upTime INT, tempExt REAL, humExt INT, humTemp INT, avgLight0 INT, avgLight1 INT, avgLight2 INT, avgLight3 INT, avgNight0 INT, avgNight1 INT, voltBat INT, microTemp REAL

Les données de type TIME s'écrivent 'HH:MM:SS' et les données de type DATE : "AAAA-MM-JJ'. Elles s'écrivent comme des chaînes de caractères, mais des recherches de minimum et de maximum sont possibles (les recherches de moyennes n'ayant guère de sens).

2.1. Requêtes simples

Créer un programme Python qui permet de saisir une lettre tant qu'on n'a pas saisi 'S'.
A chaque lettre, va correspondre une requête :

A : afficher les uptime et voltBat de toute la table

B : afficher le jour, l'heure et l'uptime pour l'index de valeur 100

C : afficher les heures, uptime, tempExt, microTemp et humExt pour le jour du '2015-04-15'

D : le programme demande la saisie d'un jour. Puis faire la même requête que pour C pour la date choisie

E : le programme demande la saisie d'un jour. Puis afficher le jour, l'uptime minimum et l'uptime maximum pour la date choisie.

F : le programme demande la saisie d'un jour. Puis afficher les valeurs moyennes de tempExt, avgLight0 et avgLight1 pour le jour choisi, pour des heures comprises entre 12:00:00 et 15:00:00

G : afficher les jours et heures pour lesquelles la différence entre microTemp et tempExt est supérieure à 5 degrés

H : afficher chaque jour, les tempExt minimal, maximal, et moyen par jour

I : afficher chaque jour, et pour chacun d'eux, l'heure minimum et l'heure maximum pour laquelle la valeur de AvgLight0 est positive (périodes de « jour »)

S : stop

Les affichages devront être « élégants », et non pas simplement consister en l'affichage brut d'une liste. Exemple pour B :« La ligne 100 du tableau a été mesurée le 2015-04-13 à 15:36:00, soit 92100s ».

2.2. Combinaison avec matplotlib

Aux requêtes A, D et H, ajouter la visualisation les résultats sous forme de graphiques :

- A : axe des ordonnées entre le min et le max de voltBat ;
- D : 3 courbes pour les tempExt, microTemp et humExt en fonction de upTime ;
- H : 3 courbes pour les TempExt max, min et moyen en fonction du n° du jour)

Les graphiques devront comporter un titre, les axes des abscisses et ordonnées légendées.