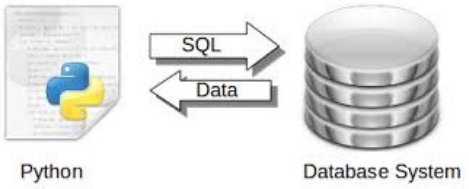
TNSI	Bases de données Bases de données et python	TP – Utilisation de bases de données en python

Objectifs :

- ⇨ Installer et utiliser une bibliothèque python
- ⇨ Découvrir comment manipuler une base de données avec python
- ⇨ Consolider ses connaissances en SQL



I – Mise en place

Dans ce TP nous allons utiliser le module `pymysql` qui permet d'interfacer python avec le SGBD mysql qui est déjà installé sur les ordinateurs du lycée.

1) Connection à mysql

Question 1 :

1) A l'aide du gestionnaire de plugins de Thonny, installer le module nommé « pymysql ».
2) Consulter la documentation du module disponible sur : https://pymysql.readthedocs.io/en/latest/user/index.html.
3) Commencer un nouveau programme python et recopier l'exemple proposé dans la documentation (la partie python, pas SQL). Essayer ensuite sur une base de données sur laquelle vous avez déjà travaillée avec mysql (par exemple la base « pronote »).
4) D'après vous, à quoi sert le « `with` » utilisé dans ce programme d'exemple ?
5) Pourquoi le « `;` » qui doit toujours terminer une requête SQL n'est-il pas utile ici ?

Remarque :

Il existe de nombreuses bibliothèques python pour faire l'interface avec un SGBD (au moins une par nom de SGBD) qui fonctionnent toutes sur le même principe (création d'un objet `connection` pour se connecter au SGBD puis d'un objet `cursor` pour exécuter des requêtes). Ainsi il est très facile de migrer d'un SGBD à un autre en changeant simplement le nom de l'import au début du programme.

Nous avons maintenant la possibilité de nous connecter à la base de données et d'effectuer des commandes SQL.

Question 2 :

En modifiant légèrement votre programme effectuer une requête sur la base pour afficher l'intégralité de sa plus grande table. Par exemple on pourra faire « `SELECT * FROM eleve` ».
Afficher ensuite le résultat dans la console à l'aide d'un « `print` ». Le résultat est-il satisfaisant ?

2) Amélioration de l'ergonomie

On va chercher maintenant à améliorer la lisibilité de nos résultats.

Question 3 :

1) Ecrire une procédure `affiche_table` qui prend en argument une liste de dictionnaires comme celle renvoyée par `pymysql` et affiche son contenu de manière plus propre. Pour les plus avancés, on peut essayer de faire un affichage avec des colonnes de largeur fixée en fonction du contenu de la table comme dans l'exemple ci-dessous.

```
| id | nom         | prenom  | classe | naissance  | entree     | sortie     |
-------------------------------------------------------------------------------
| 1  | Zeblouze    | Agathe  | TG3    | 2004-11-03 | 2021-09-03 | None       |
| 2  | Desartistes | André   | 1G5    | 2005-08-27 | 2021-09-03 | 2021-10-22 |
| 3  | Zetaufré    | Mélanie | 2GT6   | 2006-04-06 | 2021-09-03 | None       |
| 7  | Lovelace    | Ada     | 1G2    | 2005-06-20 | 2021-09-03 | None       |
| 8  | Dubois      | Alix    | TG1    | 2004-01-05 | 2021-09-03 | None       |
```

2) Utiliser votre procédure pour écrire une procédure `execute_et_affiche` qui prend en argument une commande SQL (une chaîne de caractère) et affiche le résultat sous forme de table.

Comme notre logiciel va permettre d'effectuer plusieurs opérations sur la base, on va créer un menu très simple en mode texte.

Question 4 :

1) Ecrire une fonction `menu` qui affiche pour l'instant uniquement 2 options :

```
1) Afficher les informations d'un élève
0) Quitter
```

et renvoie un entier correspondant au choix effectué par l'utilisateur. La fonction doit s'assurer que le choix de l'utilisateur est valide.

2) Ecrire le programme principal qui crée la connexion à la base de données « `pronote` » et affiche en boucle le menu jusqu' à ce que l'utilisateur choisisse l'option 0 (auquel cas il sort du programme en fermant la base de données).

II - Exploitation

Pour cette partie, on peut éventuellement repartir du fichier « TP_Python&mysql_Questions1-4.py ».

1) Afficher les informations d'un élève

Question 5 :

1) Quelle requête SQL permet d'afficher toutes les informations de la table `eleve` concernant un élève de nom `N` et de prénom `P` ?
2) Ecrire une procédure `affiche_info_eleve` qui demande le nom et le prénom d'un élève, crée une chaîne de caractère contenant la requête SQL pour interroger la base (voir question précédente) puis exécute la requête et affiche le résultat en utilisant la procédure `execute_et_affiche` programmée précédemment.
3) Modifier le programme principal pour que l'option de menu 1 exécute la procédure `affiche_info_eleve`. Vérifier sur quelques exemples que votre procédure fonctionne bien.
4) Exécuter votre programme, choisir l'option 1 pour afficher les informations d'un élève et utiliser comme nom « `' UNION (SELECT * FROM eleve) ; --` » et comme prénom n'importe quoi. Que se passe-t-il ?

Ce qu'on vient de faire à la dernière question s'appelle une **injection SQL** (*SQL injection* en anglais). Elle permet à un pirate d'accéder de manière dérobée à la base de données. Ici on a affiché la liste des élèves ce qui nous donne des informations que nous ne devrions potentiellement pas voir, mais on aurait aussi bien pu supprimer la base de données en mettant comme nom « `' ; DROP DATABASE ;--` » ou encore plus sournoisement altérer quelques données de la base avec un `UPDATE` bien choisi.
C'est une attaque très classique et qui peut être redoutable pour des sites ou logiciels mal conçus.

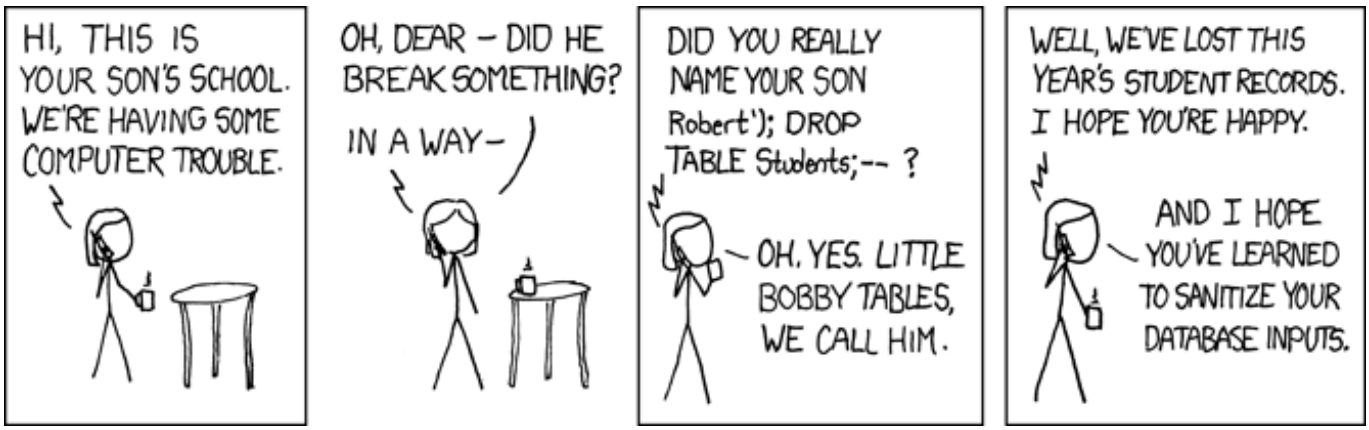
Pour s'en protéger on doit filtrer ou « assainir » (*sanitize*) toutes les données qui proviennent de l'utilisateur comme ici le nom ou le prénom avant de les insérer dans nos requêtes SQL.
La méthode `execute` de la classe `Cursor` fournit justement un, mécanisme d'échappement [1]pour assainir les chaînes à insérer dans les requêtes SQL.

Il est ainsi possible d'écrire « `%(nom)` » dans la chaîne de la requête SQL à chaque valeur que l'on veut insérer et de fournir en deuxième argument de la méthode `execute` un dictionnaire de toutes les valeurs à remplacer.
Exemple :

```
curseur.execute("INSERT INTO commande VALUES (%(id_cmd),%(id_client))",
{'id_cmd':'4','id_client':'23'})
```

On peut également écrire `%s` dans la requête SQL pour chaque valeur à remplacer et donner dans les arguments de SQL une liste de toutes les valeurs dans l'ordre dans lesquelles elles doivent remplacer les `%s`.
Exemple :

```
curseur.execute("INSERT INTO commande VALUES (%s,%s,%s)", [4,23,'fnac'])
```



Question 6 :
Modifier la procédure `affiche_info_eleve` pour qu'elle utilise le mécanisme d'assainissement des requêtes SQL proposé par la méthode SQL (on ne peut plus ici utiliser telle-quelle la procédure `execute_et_affiche` mais on peut toujours utiliser `affiche_table` avec le résultat de la requête).

[1] L'échappement consiste à exprimer certains caractères sensibles comme les guillemets ou les « \ » en python de manière à ce qu'ils puissent faire partie de la chaîne de caractère sans être interprétés. En python le caractère « \ » peut être échappé en le doublant. Ainsi `"saut de ligne : \\n"` correspond à la chaîne « `saut de ligne : \n` »

2) Ajouter un élève à la base

Question 7 :
Modifier la procédure `menu` pour faire apparaître l'option « `2) Ajouter un élève à la base` » et écrire une procédure `ajoute_eleve` qui demande toutes les informations utiles et insère un nouvel élève dans la base.

3) Afficher les notes d'un élève

Question 8 :
1) Quelle requête SQL permet d'afficher toutes les notes d'un élève de nom `N` et de prénom `P` ?
2) Ecrire une procédure `affiche_notes_eleve` qui demande le nom et le prénom d'un élève et affiche toutes les notes qu'il a eu. On modifiera également le menu pour faire apparaître une option « `3) Afficher les notes d'un élève` ». Pour les plus avancés, on pourra faire figurer dans l'affichage la matière, l'intitulé de l'évaluation et sa date en groupant les évaluations par matière et en triant par dates croissantes des évaluations.
3) A la question précédente, rajouter une colonne « Moyenne classe » qui affiche à côté de la note de l'élève la moyenne des notes pour la classe à la même évaluation. Pour cette question, on peut écrire une requête complexe (avec sous-requête) pour que le SGBD nous donne directement l'information demandée ou faire plusieurs requêtes et faire faire le calcul de moyenne à python.