

TP - SQLite

- **import de la bibliothèque sqlite de python :**
`import sqlite3 as sq`
- **connexion avec la base (création et ouverture) :**
`ma_base=sq.connect("BASE_Cours_Etud_Notes.sqlite")`
Cette commande permet d'ouvrir le fichier 'BASE_Cours_Etud_Notes.sqlite'. Si le fichier n'existe pas, il sera créé.
NB : `ma_base` représente le nom logique de la base. Tandis que "BASE_Cours_Etud_Notes.sqlite" est le nom physique de la base.
- **Création d'un curseur où sera stockée une version temporaire du travail :** curseur est un objet de type cursor qui permettra d'effectuer des requêtes SQL depuis python.
`curseur = ma_base.cursor()`
- **Fermeture de la base (déconnexion) :**
`curseur.close()`
`ma_base.close()`
- **Création des tables (schémas des table ainsi que les contraintes d'intégrités :**
 - `curseur.execute("create table if not exists Etudiants(IdEtudiant integer primary key,Nom text,Prenom text,Adresse text)")`
 - `curseur.execute("create table if not exists Cours(IdCours text,Jour text,Heure text, primary key(IdCours,Jour)")`
 - `curseur.execute("create table if not exists SallesCours(IdCours text,IdSalle text, primary key (IdCours,IdSalle)")`
 - `curseur.execute("create table if not exists Notes(IdCours text,IdEtudiant integer>Note text, primary key(IdCours,IdEtudiant), foreign key (IdCours) references Cours(IdCours), foreign key (IdEtudiant) references Etudiant(IdEtudiant)")`
- **Validation de toutes les transactions (opérations) :**
`ma_base.commit()`
- **Insertion d'enregistrements dans une table**
 1. Insertion enregistrement par enregistrement
 - `curseur.execute("insert into Etudiants (IdEtudiant,Nom,Prenom,Adresse) values (100,'Toto','prenomToto', 'Tunis')")`
 - `curseur.execute("insert into Etudiants values (100,'Toto','prenomToto', 'Tunis')")`
 - `curseur.execute("insert into Etudiants values (?, ?, ?, ?)", (100,'Toto','prenomToto', 'Tunis'))`

Dans l'exemple ci-dessous, la valeur de l'attribut Adresse sera manquante :

- `curseur.execute("insert into Etudiants (IdEtudiant,Nom,Prenom) values (100,'Toto','prenomToto')")`
- 2. Insertion de plusieurs enregistrements en même temps (à partir d'une liste) :


```
L=list()
L.append((100,'Toto','prenomToto', 'Tunis'))
L.append((200,'Tata','prenomTata', 'La Manouba'))
L.append((300,'Titi','prenomTiti', 'Kairouan'))
curseur.executemany("insert into Etudiants values ( ?, ?, ?, ?)",L)
```
- **Modification d'enregistrement(s) :**

```
curseur.execute("update Etudiants set Adresse='Sousse' where idEtudiant=100 ")
```
- **Suppression d'enregistrement(s) :**

```
curseur.execute("delete from Etudiants where idEtudiant=200 ")
```
- **Modification du nom d'une table :**

```
curseur.execute("Alter table Etudiants rename to Etud")
```
- **Suppression d'une table :**

```
curseur.execute("drop table Notes")
```
- **Interrogation (Recherche) des données :**

```
curseur.execute("select * from Etudiants")
```
- Récupérer le résultat d'une recherche
 1. Manipuler directement le curseur.

```
for i in curseur:
    print (i)
```

NB : `i` est un tuple et représente une ligne du résultat de la requête.

2. Manipuler les lignes du curseur ligne par ligne avec `fetchone()`.

```
while True:
    T=curseur.fetchone()
    if T!=None:
        print(T)
    else:
        break
```

NB : De même, la méthode `fetchone()` retourne un tuple qui représente une ligne du résultat de la requête.

3. Manipuler le résultat du curseur à travers une liste avec `fetchall()`.

```
L=curseur.fetchall()
for i in L:
    print(i)
```

NB : La méthode `fetchall()` retourne une liste comportant toutes les lignes (sous forme de tuples) résultat de la requête.