

# Mini-projet : CDI



Le but de ce mini-projet est de créer un programme Python qui permettra d'effectuer différentes opérations sur la base de donnée du CDI déjà manipulée en cours et en exercices cette année.

La base de données CDI, déjà utilisée, est redonnée avec les documents de ce mini-projet, qui ne nécessite donc pas la création d'une nouvelle base.

On pourra revoir les documents la concernant pour se rappeler sa structure.

Pour importer le module et faire des requêtes, on peut procéder ainsi (cf. `exemple_sqlite3.py`) :

```
import sqlite3 as sgbd # import du module

with sgbd.connect('cdi.db') as cnx: # connexion à la base de données

    # création d'un curseur avec lequel on va exécuter les requêtes
    c = cnx.cursor()

    # Obtention et définition de paramètres à venir dans les requêtes
    texte = input ("Texte à rechercher dans le titre : ")
    motif = '%' + texte + '%'

    annee = 2005

    # Exécution d'une requête, avec deux paramètres :
    c.execute("SELECT titre, annee \
              FROM Livre \
              WHERE titre LIKE ? AND annee < ?" ,
              [motif, annee])

    for l in c.fetchall(): # parcours des lignes du résultat de la requête
        print(l) # affichage d'une ligne (qui est un tuple)

# quand on revient à l'indentation de base, la connexion est fermée
```

Exemple d'exécution :

```
Texte à rechercher dans le titre : ast
('Astérix chez les Belges', 1979)
('Astérix chez les Bretons', 2002)
('Astérix et Cléopâtre', 1999)
```

Dans les cas où la requête est en fait un ordre de modification de la base (**INSERT**, **DELETE**, etc.), il faudra, juste après l'obtention de la connexion avec la base de donnée, exécuter l'instruction suivante :

```
cnx.execute("PRAGMA foreign_keys = ON")
```

Et après l'exécution de la requête de modification, demander à appliquer les changements dans la base avec la ligne de code suivante :

```
cnx.commit()
```

Pour que les contraintes de clés étrangères soient vérifiées (lors de modifications de la base), il faut également,

Le fonctionnement du programme sera basé sur une fonction principale `main()` ayant la forme suivante (cf. `code.py`) :

```
def main():
    stop=False
    while not stop:
        print("", "Choix :",
              "emprunter (e)", "rendre (r)", "ajouter (a)",
              "supprimer (s)", "lister les prêts (l)",
              "rechercher un document (?)",
              "quitter (q)", sep="\n")
        choix=input("Quel est votre choix ? ")
        if choix=="e":
            emprunt()
        elif choix=="r":
            rendre()
        elif choix=="q":
            stop=True
        else:
            print("Je n'ai pas compris...")
    print("\nSortie du programme. Au revoir !")
```

Autrement dit, la fonction `main()`, exécutée au lancement du fichier, est une boucle infinie de laquelle on sort uniquement lorsque l'on décide de quitter.

Les différentes actions exécutées pourront être définies dans des fonctions pour alléger le code global.

Les choses suivantes devront être rendues possibles :

- **emprunter** : dans cette partie, on permettra l'ajout dans la base de l'emprunt d'un livre par un élève.  
Pour cela, on demandera le numéro de l'élève et celui livre emprunté (ISBN).
- **rendre** : dans cette partie, on permettra la suppression dans la base de l'emprunt d'un livre par un élève.  
Pour cela, on demandera le numéro du livre rendu (et éventuellement celui de l'élève).
- **ajouter** : ajouter un livre (dont on demandera les caractéristiques) à la base.
- **supprimer** : supprimer un livre (dont on demandera le numéro) de la base.  
Il faudra vérifier au préalable que le livre n'est pas emprunté.  
Facultatif : Si, ce faisant, son auteur n'apparaît plus comme auteur de livres du CDI, on pourra également supprimer cet auteur (en le précisant).
- **lister les prêts** : étant donné le numéro d'un élève, affiche la liste des prêts de l'élève. On précisera pour chaque emprunt le numéro ISBN et la date de retour.
- **rechercher un document** : dans cette partie, on pourra mettre en œuvre un sous-menu permettant de rechercher par **titre**, par **auteur** ou par **ISBN**.  
On devra préciser les informations sur le documents : titre, auteur, etc.

On donne l'exemple pour l'emprunt (cf. `code.py`) :

```

def emprunt():
    with sgbd.connect('cdi.db') as cnx: # connexion à la base de données
        # vérification des contraintes de clé étrangère
        cnx.execute('PRAGMA foreign_keys = ON')
        # création d'un curseur avec lequel on va exécuter les requêtes
        c = cnx.cursor()

        # Obtention et définition de paramètres à venir dans les requêtes
        isbn = input("ISBN du livre : ")
        eleve = input("numéro de l'élève : ")

        # Exécution d'une requête, avec deux paramètres :
        c.execute("INSERT INTO Emprunt \
                    VALUES (?, ?, date('now', '+28 days'))",
                    [isbn, int(eleve)])
        cnx.commit()
        print("La base de données a été mise à jour")

```

Le travail sera rendu avec un fichier texte indiquant des exemples d'utilisation de chacun des choix possibles.

Au cas où vous en auriez le temps, vous pouvez ajouter des difficultés supplémentaires, soit au niveau des requêtes à exécuter, soit au niveau de la robustesse (éviter un arrêt par erreur) ou tout autre élément d'extension envisageable (la base de données contient des auteurs, mais aussi des éditeurs...).