INFORMATIQUE PSI P2B

Année 2025/26

TP n° 5

Bases de données

Table des matières

Vincent Rohart

http://vrohart.e-monsite.com/

1 Création de tables et requêtes SQL

Le module à importer est `sqlite3`.

Supposons que l'on veuille créer une table `ELEVES` dont le schéma est

```
ELEVES(id : INT, prénom : STR(15), nom : STR(15), age : INT, classe : STR(10), redoubl : BOOL, sexe : STR(1)).
```

La marche à suivre est alors
— de créer une *connexion* avec la table que l'on va construire.
— de créer un *curseur* qui permettra d'enregistrer des nouvelles entrées dans cette table, mais aussi d'exécuter des requête SQL au moyen de la méthode `execute`.

```
import sqlite3

connexion = sqlite3.connect(':memory:') #L'option memory est importante
curseur = connexion.cursor()
```

Pour créer la table `ELEVES` dont les attributs sont donnés ci-dessus, on écrira :

```
curseur.execute("""CREATE TABLE ELEVES(
      id  INT,
      prenom  str,
      nom str,
      age  int,
      classe  str,
      redoubl  bool,
      sexe str)""")
```

Pour enregistrer une entrée dans cette table, actuellement vide, on écrit :

```
e = (5 ,"Charles", "Bosco", "2006", "P2B", False,"M") #Ceci est une entrée

curseur.execute("INSERT INTO ELEVES VALUES (?,?,?,?,?,?,?)", e)
```

Les symboles `?` vont alors se substituer aux valeurs présentes dans l'entrée `e`. Si l'on veut enregistrer un ensemble d'entrées, on crée la liste de ces entrées, puis on fait appel à la méthode `executemany` :

```
Liste =  [(5 ,"Charles", "Bosco", "2006","P2B", False,"M"),
          (22 ,"Yann", "Mescam", "2006","P2B", False,"M"),
          (24 ,"Gabriel", "Noel", "2006","P2B", False,"M"),
          (16 ,"Marie-Jeanne", "Goirand", "2006","P2B", False,"F")]

curseur.executemany("INSERT INTO ELEVES VALUES (?,?,?,?,?,?,?)", Liste)
```

La table étant créée, des requêtes SQL sont alors possibles. Il faut alors demander de traiter toutes les sorties à l'aide de la méthode `fetchall`, et ne pas oublier d'afficher (avec `print`) lesdites sorties.

```
curseur.execute("SELECT nom, age  FROM ELEVES WHERE age >= 20")
resultat=curseur.fetchall()
for i in resultat :
    print(i)
```

L'opérateur `LIKE` permet d'affiner les conditions présentes après `WHERE`. Par exemple

```
curseur.execute("SELECT nom, age  FROM ELEVES WHERE nom LIKE 'S%' ")
resultat=curseur.fetchall()
for i in resultat :
    print(i)
```

affichera tous les noms des élèves dont le nom commence par un S. L'option `'% S'` recherche ceux dont le nom finit par un S, etc.

Questions.

1. Implémenter les tables `ELEVES` et `NOTES` fournies avec ce TP.
2. Écrire une requête SQL permettant d'afficher le prénom et le nom de tous les élèves né(e)s en 2007. Présenter les résultats par ordre alphabétique.
3. Afficher toutes les notes données par Mme Bourouina.
4. Écrire une requête SQL permettant d'afficher le nom des élèves ayant eu plus que 10/20 à la khôlle n° 19 en mathématiques. *On devra faire une jointure.*
5. Afficher le nombre total d'élèves en P2B. *Cf. cours en ligne.*
6. Afficher la moyenne de la khôlle n° 19 en maths des élèves de P2B. *Cf. cours en ligne.*
7. Y a-t-il des élèves dont le prénom commence par M ?
8. Y a-t-il des élèves dont le nom de famille comporte deux Y consécutifs ?
9. Afficher la liste de tous les élèves dont le nom de famille est composé de plus d'un mot.

2 Un exemple : sujet 2020 Centrale-Supélec

Une base de données répertorie des photographies. Son modèle relationnel est donné par la figure ci-dessous.

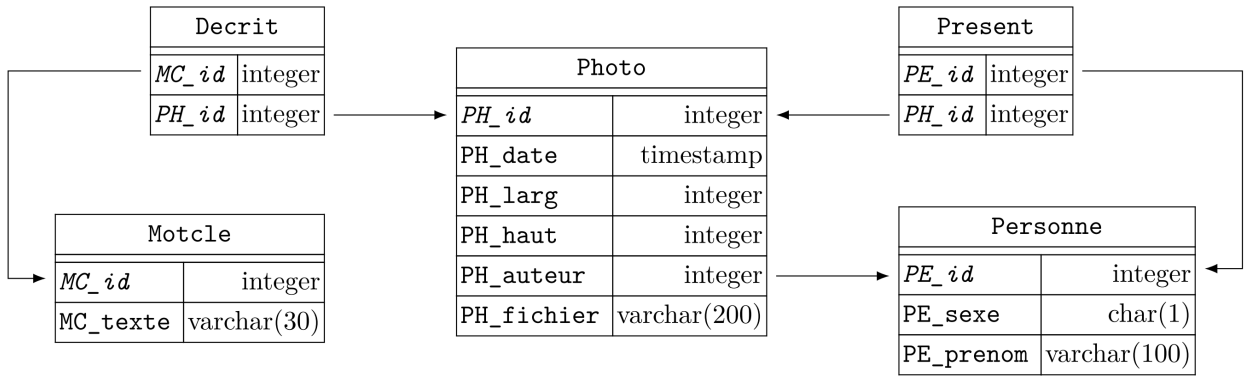

Figure 4 Structure physique de la base de données de photographies.

Cette base comporte les cinq tables listées ci-dessous avec la description de leurs colonnes :

— la table `Photo` répertorie les photographies
- `PH_id` : identifiant (entier arbitraire) de la photographie (clé primaire)
- `PH_date` : date et heure de la prise de vue
- `PH_larg`, `PH_haut` : largeur et hauteur de la photographie en pixels
- `PH_auteur` : identifiant de l'auteur de la photographie
- `PH_fichier` : nom du fichier contenant la photographie

— la table `Personne` des modèles et des photographes
- `PE_id` : identifiant (entier arbitraire) de la personne (clé primaire)
- `PE_sexe` : sexe de la personne ('M' ou 'F')
- `PE_prenom` : prénom de la personne

— la table `Motcle` des mots-clés utilisés pour décrire une photographie
- `MC_id identifiant` : (entier arbitraire) du mot-clé (clé primaire)

- `MC_texte` : le mot-clé lui-même

— la table `Decrit` fait le lien entre les photographies et les mots-clés qui les décrivent, ses deux colonnes constituent sa clé primaire
  - `MC_id` : identifiant du mot-clé (décrivant la photographie)
  - `PH_id` : identifiant de la photographie (décrite par le mot-clé)

— la table `Present` fait le lien entre les photographies et les personnes qui y figurent, ses deux colonnes constituent sa clé primaire
  - `PE_id` : identifiant de la personne (figurant sur la photographie)
  - `PH_id` : identifiant de la photographie (représentant la personne)

Questions.

1. Écrire une requête SQL donnant les identifiants de toutes les photographies au ratio 4/3 c'est-à-dire dont le rapport largeur sur hauteur est 4/3. *Indication : éviter les fractions.*
2. Écrire une requête qui compte le nombre de photos prises par Alice ou Bernard.
3. Écrire une requête qui fournit l'identifiant et la date des photographies prises avant 2006 et associées au mot-clé « surf ».
4. Écrire une requête qui donne le prénom de l'auteur et l'identifiant de tous les selfies, c'est-à-dire des photographies sur lesquelles l'auteur est présent.
5. Écrire une requête qui sélectionne toutes les photographies sur lesquelles sont présents Alice et Bernard à l'exception de tout autre personne.