Le langage SQL - Activité d'introduction

Dernière mise à jour le : 16/12/2024

Introduction

Dans cette activité, vous allez découvrir comment effectuer des requêtes sur une table d'une base de données, en utilisant le langage SQL (pour *Structured Query Language*).

Pour cela, nous allons travailler avec une table déjà étudiée l'année dernière en classe de Première, ce sera l'occasion de faire le lien avec ce qui a été vu l'an dernier.

Voici un extrait de la table utilisée l'année dernière :

prénom	jour	mois	année	sexe	classe	projet
Naïma	7	9	2007	F	1G4	apprendre à piloter une Formule 1
Jade	5	7	2007	F	1G2	ouvrir un restaurant
Mathilda	9	7	2007	F	1G4	gagner au loto
Louenn	8	7	2007	G	1G3	devenir quelqu'un de célèbre
Enola	13	2	2007	F	1G3	avoir une bonne note au prochain devoir
Samuel	2	8	2007	G	1G4	manger des frites tous les jours à la cantine
Lucien	2	8	2007	G	1G4	marcher sur la lune
Matias	9	8	2007	G	1G3	dormir plus longtemps le matin
Nathan	3	9	2007	G	1G2	aider les autres
Théliau	7	3	2007	G	1G2	apprendre à piloter une Formule 1
...	...	...	...	...	...	...

Cette dernière était stockée dans un fichier `eleves.csv` que l'on pouvait importer en Python pour mémoriser son contenu dans un tableau de dictionnaires `eleves` :

```
import csv

fichier = open('eleves.csv', 'r', encoding = 'UTF-8')  # ouverture du fichier
# Lecture de son contenu avec la méthode DictReader qui renvoie une valeur spéciale représentant le fichier
table = csv.DictReader(fichier, delimiter=',')  # caractère de séparation : la virgule
eleves = [dict(ligne) for ligne in table]  # chaque élément est un dictionnaire
fichier.close()  # fermeture du fichier
```

```
>>> eleves
[{'prénom': 'Naïma', 'jour': '7', 'mois': '9', 'année': '2007', 'sexe': 'F', 'classe': '1G4', 'projet': 'app
 {'prénom': 'Jade', 'jour': '5', 'mois': '7', 'année': '2007', 'sexe': 'F', 'classe': '1G2', 'projet': 'ouvr
 {'prénom': 'Mathilda', 'jour': '9', 'mois': '7', 'année': '2007', 'sexe': 'F', 'classe': '1G4', 'projet': '
 {'prénom': 'Louenn', 'jour': '8', 'mois': '7', 'année': '2007', 'sexe': 'G', 'classe': '1G3', 'projet': 'de
 ...,
 {'prénom': 'Amaël', 'jour': '16', 'mois': '9', 'année': '2007', 'sexe': 'G', 'classe': '1G1', 'projet': 'de
```

Par défaut, toutes les valeurs sont mémorisées comme des chaînes de caractères, y compris le jour, le mois et l'année. On peut effectuer un prétraitement pour convertir en type `int` chaque valeur associée aux clés `'jour'`, `'mois'` et `'annee'` :

```
eleves = [
    {'prénom': e['prénom'],
```

```
    'jour': int(e['jour']),
    'mois': int(e['mois']),
    'année': int(e['année']),
    'sexe': e['sexe'],
    'classe': e['classe'],
    'projet': e['projet']} for e in eleves
]
```

```
>>> eleves
[{'prénom': 'Naïma', 'jour': 7, 'mois': 9, 'année': 2007, 'sexe': 'F', 'classe': '1G4', 'projet': 'apprendre
 {'prénom': 'Jade', 'jour': 5, 'mois': 7, 'année': 2007, 'sexe': 'F', 'classe': '1G2', 'projet': 'ouvrir un
 {'prénom': 'Mathilda', 'jour': 9, 'mois': 7, 'année': 2007, 'sexe': 'F', 'classe': '1G4', 'projet': 'gagner
 {'prénom': 'Louenn', 'jour': 8, 'mois': 7, 'année': 2007, 'sexe': 'G', 'classe': '1G3', 'projet': 'devenir
 ...,
 {'prénom': 'Amaël', 'jour': 16, 'mois': 9, 'année': 2007, 'sexe': 'G', 'classe': '1G1', 'projet': 'devenir
```

On a créé une base de données `eleves.db` qui ne contient que la *relation* `Eleve` suivante :

```
Eleve(
    id_eleve INT,
    prenom TEXT,
    jour INT,
    mois INT,
    annee INT,
    sexe TEXT,
    classe TEXT,
    projet TEXT
)
```

> On a ajouté un attribut `id_eleve`, pourquoi selon vous ?

Remarque : Le fichier `eleves.db` que nous utiliserons est accessible dans l'archive de ce chapitre.

Utilisation d'un SGBD

Seul un logiciel de type SGBD (Système de Gestion de Bases de Données) peut manipuler les données présentes dans une base de données.

Vous pouvez utiliser :

- **DB Browser for SQLite** qui est installé sur vos ordinateurs :
  - (ce logiciel est téléchargeable à l'adresse : https://sqlitebrowser.org/)
  - Lancez le logiciel
  - Ouvrez le fichier `eleves.db` en le sélectionnant après avoir cliqué sur "Ouvrir une base de données"
- **sqliteonline** accessible en ligne à l'adresse https://sqliteonline.com/.
  - Rendez-vous à l'adresse https://sqliteonline.com/
  - Cliquez sur *File* puis sur *Open DB* et sélectionnez le fichier `eleves.db`

Première requête SQL

Commençons par une requête permettant d'afficher tout le contenu de la table `Eleve`.

```sql
SELECT * FROM Eleve;
```

Analyse :

- En SQL, chaque requête contient au moins les clauses `SELECT` et `FROM` et **se termine par un point-virgule**.
- Le mot-clé `SELECT` demande au SGBD d'afficher ce que contient une table.
- Après `SELECT`, il faut indiquer quels champs de la table, le SGBD doit récupérer dans celle-ci. Ici le caractère « `*` » indique qu'il faut récupérer **tous** les champs de la table.
- Après le `FROM` (de l'anglais « de ») on indique la table dans laquelle on veut récupérer des informations (ici `Eleve`)
- Bilan : cette requête se traduit par « prend tout ce qu'il y a dans la table `Eleve`, sous-entendu prend tous les champs de cette table.

Q1 : Écrivez cette requête dans le SGBD puis exécutez-la. Vous devriez voir apparaître la table `Eleve` complète comme ci-dessous.

id_eleve	prenom	jour	mois	annee	sexe	classe	projet
1	Naïma	7	9	2007	F	1G4	apprendre à piloter une Formule 1
2	Jade	5	7	2007	F	1G2	ouvrir un restaurant
3	Mathilda	9	7	2007	F	1G4	gagner au loto
4	Louenn	8	7	2007	G	1G3	devenir quelqu'un de célèbre
5	Enola	13	2	2007	F	1G3	avoir une bonne note au prochain devoir
6	Samuel	26	4	2007	G	1G4	manger des frites tous les jours à la cantine
7	Lucien	2	8	2007	G	1G4	marcher sur la lune
8	Matias	9	8	2007	G	1G3	dormir plus longtemps le matin
9	Nathan	3	9	2007	G	1G2	aider les autres
10	Théliau	7	3	2007	G	1G2	devenir président
11	Ali	21	11	2007	G	1G2	devenir un joueur professionnel de billard
12	Anastasia	24	9	2006	F	1G4	créer un jeu vidéo
13	Yohann	22	7	2007	G	1G2	gérer sa propre entreprise
14	Bastien	9	3	2007	G	1G2	jouer dans le prochain James Bond
15	Noe	30	12	2007	G	1G3	monter le tapis rouge à Cannes
16	Lilian	27	12	2007	G	1G2	devenir champion du monde de bras de fer
17	Simon	6	12	2007	G	1G1	aller à Poudlard
18	Eliott	12	11	2007	G	1G4	avoir son émission de TV
19	Amaël	16	9	2007	G	1G1	devenir une rockstar

Sélection de colonnes en SQL

Rappel : en classe de Première, on a effectué des projections sur les colonnes de la table. L'instruction suivante permet de créer une table `t` contenant les attributs `prenom` et `annee` de la table `eleves`. On dit que l'on fait ainsi une projection sur ces deux colonnes.

```
>>> t = [{'prénom': e['prénom'], 'année': e['année']} for e in eleves]
>>> t
[{'prénom': 'Naïma', 'année': 2007},
 {'prénom': 'Jade', 'année': 2007},
 {'prénom': 'Mathilda', 'année': 2007},
 {'prénom': 'Louenn', 'année': 2007},
 {'prénom': 'Enola', 'année': 2007},
 {'prénom': 'Samuel', 'année': 2007},
 {'prénom': 'Lucien', 'année': 2007},
 {'prénom': 'Matias', 'année': 2007},
 {'prénom': 'Nathan', 'année': 2007},
 {'prénom': 'Théliau', 'année': 2007},
 {'prénom': 'Ali', 'année': 2007},
 {'prénom': 'Anastasia', 'année': 2006},
 {'prénom': 'Yohann', 'année': 2007},
```

```
 {'prénom': 'Bastien', 'année': 2007},
 {'prénom': 'Noe', 'année': 2007},
 {'prénom': 'Lilian', 'année': 2007},
 {'prénom': 'Simon', 'année': 2007},
 {'prénom': 'Eliott', 'année': 2007},
 {'prénom': 'Amaël', 'année': 2007}]
```

En SQL, il est possible de sélectionner certaines colonnes de la table simplement en indiquant après le `SELECT`, les noms des attributs à conserver.

Par exemple, la requête

```
SELECT prenom, annee FROM Eleve;
```

permet de ne sélectionner que les attributs `prenom` et `annee` de la table `Eleve` (on fait une projection sur ces deux attributs). Elle produit le résultat :

prenom	annee
Naïma	2007
Jade	2007
Mathilda	2007
Louenn	2007
Enola	2007
Samuel	2007
Lucien	2007
Matias	2007
Nathan	2007
Théliau	2007
Ali	2007
Anastasia	2006
Yohann	2007
Bastien	2007
Noe	2007
Lilian	2007
Simon	2007
Eliott	2007
Amaël	2007

Q2 : Écrivez et testez une requête SQL permettant de sélectionner uniquement les attributs `prenom`, `sexe` et `projet`.

Réponse :

Sélectionner de lignes en SQL

Rappel : en classe de Première, on a sélectionné des lignes de la table vérifiant une certaine condition. L'instruction suivante permettait de créer une table `nes_en_septembre` contenant les lignes des élèves nés en janvier.

```
>>> nes_en_septembre = [eleve for eleve in eleves if eleve['mois'] == 9]
>>> nes_en_septembre
[{'prénom': 'Naïma',
  'jour': 7,
```

```
  'mois': 9,
  'année': 2007,
  'sexe': 'F',
  'classe': '1G4',
  'projet': 'apprendre à piloter une Formule 1'},
 {'prénom': 'Nathan',
  'jour': 3,
  'mois': 9,
  'année': 2007,
  'sexe': 'G',
  'classe': '1G2',
  'projet': 'aider les autres'},
 {'prénom': 'Anastasia',
  'jour': 24,
  'mois': 9,
  'année': 2006,
  'sexe': 'F',
  'classe': '1G4',
  'projet': 'créer un jeu vidéo'},
 {'prénom': 'Amaël',
  'jour': 16,
  'mois': 9,
  'année': 2007,
  'sexe': 'G',
  'classe': '1G1',
  'projet': 'devenir une rockstar'}]
```

En SQL, il est possible de préciser la condition de sélection en l'écrivant juste après le mot clé `WHERE` comme ci-dessous.

```
SELECT * FROM Eleve WHERE mois = 9;
```

id_eleve	prenom	jour	mois	annee	sexe	classe	projet
1	Naïma	7	9	2007	F	1G4	apprendre à piloter une Formule 1
9	Nathan	3	9	2007	G	1G2	aider les autres
12	Anastasia	24	9	2006	F	1G4	créer un jeu vidéo
19	Amaël	16	9	2007	G	1G1	devenir une rockstar

Q3 : Écrivez et testez une requête SQL permettant d'afficher les élèves issus de la classe 1G2.

Réponse :

Q4 : Écrivez et testez une requête SQL permettant d'afficher les élèves de 1G4 nés en septembre 2007.

> On peut utiliser les opérateurs `AND` ou `OR` pour combiner les conditions de sélection.

Réponse :

Sélectionner des lignes et des colonnes

Rappel : L'instruction suivante permettait de créer une table `prenoms_projets_des_filles` contenant les prénoms et les projets des filles de la classe.

```
>>> prenoms_projets_des_filles = [
    {'prénom': e['prénom'], 'projet': e['projet']} for e in eleves if e['sexe'] == 'F'
]
```

```
>>> prenoms_projets_des_filles
[{'prénom': 'Naïma', 'projet': 'apprendre à piloter une Formule 1'},
 {'prénom': 'Jade', 'projet': 'ouvrir un restaurant'},
 {'prénom': 'Mathilda', 'projet': 'gagner au loto'},
 {'prénom': 'Enola', 'projet': 'avoir une bonne note au prochain devoir'},
 {'prénom': 'Anastasia', 'projet': 'créer un jeu vidéo'}]
```

En SQL, pour combiner les les sélections sur les lignes et les projections sur les colonnes, il suffit de préciser les attributs de projection après le `SELECT` et préciser la condition de sélection après le `WHERE`.

Par exemple, la requête

```
SELECT prenom, projet FROM Eleve WHERE sexe="F";
```

permet d'afficher une table contenant les prénoms et les projets des filles de la classe :

prenom	projet
Naïma	apprendre à piloter une Formule 1
Jade	ouvrir un restaurant
Mathilda	gagner au loto
Enola	avoir une bonne note au prochain devoir
Anastasia	créer un jeu vidéo

Q5 : Écrivez et testez une requête SQL permettant d'afficher les prénoms (et uniquement les prénoms) des élèves nés en août 2007.

Réponse :

Q6: Écrivez et testez une requête SQL permettant d'afficher les id, prénoms et projets des élèves nés les six derniers mois de l'année 2007.

Réponse :

Trier les résultats avec ORDER BY

La commande `ORDER BY` permet de trier les résultats d'une requête en précisant juste après sur quel attribut le tri doit se faire.

Par exemple, la requête

```
SELECT prenom, annee FROM Eleve ORDER BY prenom ASC;
```

permet de trier le résultat de la requête `SELECT prenom, annee FROM Eleves` selon les prénoms des élèves, ici selon l'ordre alphabétique grâce à `ASC` (« ascendant », du plus petit au plus grand).

> Il suffirait de remplacer `ASC` par `DESC` (« descendant ») pour que le tri se fasse dans le sens inverse.

Q7 Écrivez et testez une requête SQL permettant d'afficher les prénoms et l'année de naissance des élèves, le résultat étant trié par ordre croissant d'année de naissance.

Réponse :

Q8 : Écrivez et testez une requête SQL permettant d'afficher, du plus jeune au moins jeune, les prénoms des élèves nés en septembre 2007.

> On peut combiner les clauses `WHERE` et `ORDER BY`

Réponse :

Quelques fonctions de calculs

Il est possible d'utiliser des fonctions de calculs en SQL. En voici quelques unes.

Compter le nombre de tuples : `COUNT(*)`

Donnons tout de suite deux exemples.

La requête SQL suivante permet de compter le nombre d'élèves de Première NSI.

```
SELECT COUNT(*) AS nbEleves FROM Eleve;
```

nbEleves
19

La requête SQL suivante permet de compter le nombre de garçons de la classe.

```
SELECT COUNT(*) AS nbGarcons FROM Eleve WHERE sexe = "G";
```

nbGarcons
14

Analyse de la deuxième requête :

- On sélectionne les tuples de la table `Eleve` correspondant à des garçons grâce à l'instruction `FROM Eleves WHERE sexe="G"`
- L'instruction `COUNT(*)` permet de compter *tous* les tuples sélectionnés
- L'ajout de `AS nbGarcons` permet de nommer le résultat. Cette astuce est optionnelle mais est particulièrement utile pour faciliter la lecture des requêtes.

Q9 : Écrivez une requête SQL permettant de compter le nombre d'élèves de 1G2.

Réponse :

Q10 : Écrivez une requête SQL permettant de compter le nombre d'élèves nés en 2007.

Réponse :

Q11 : Écrivez une requête SQL permettant de compter le nombre d'élèves nés en février.

Réponse :

Compter le nombre de valeurs distinctes d'un attribut

La requête

```sql
SELECT annee FROM Eleve;
```

va afficher les années de naissance de chacun des 16 élèves (à savoir 2007, 2007, ..., 2007, 2006, 2007 ..., 2007)

L'utilisation d'un `DISTINCT` permet d'éviter les redondances dans les résultats affichées. Ainsi, l'instruction

```sql
SELECT DISTINCT annee FROM Eleve;
```

ne va afficher que les années de naissance distinctes des élèves (à savoir 2006, 2007). N'hésitez pas à tester ces deux requêtes.

Pour compter le nombre d'années de naissance distinctes des élèves de la classe, il suffit d'ajouter un `COUNT` comme suit :

```sql
SELECT COUNT(DISTINCT annee) AS nbAnneesDistinctes FROM Eleve;
```

Q12 : Écrivez une requête SQL permettant de compter le nombre de mois de naissances différents des élèves de la classe.

Réponse :

La fonction `AVG()`

La fonction d'agrégation `AVG()` (de l'anglais, *average* signifiant "moyenne") dans le langage SQL permet de calculer une valeur moyenne d'un ensemble de valeurs numériques.

Par exemple, la requête

```sql
SELECT AVG(annee) AS anneeMoyenne FROM Eleve;
```

permet de calculer l'année moyenne de naissance des élèves de la classe.

anneeMoyenne
2006.9473684210527

Q13 : Écrivez une requête SQL permettant de calculer l'âge moyen des élèves de la classe (en 2024). Ce résultat sera nommé `ageMoyen`.

Réponse :

Bilan

- Pour interroger une base de données, on doit écrire des requêtes dans le langage SQL via un logiciel de type SGBD.
- Nous avions ici une base de données formée d'une seule table, ce qui n'est pas souvent le cas. Ainsi, vous découvrirez d'autres fonctionnalités du langage SQL dans la suite de ce chapitre où nous travaillerons aussi sur des bases de données formées de plusieurs tables.

Germain BECKER, Sébastien Point, Lycée Mounier, ANGERS

Voir en ligne : info-mounier.fr/terminale_nsi/base_de_donnees/langage-sql-activite-intro.php