Commandes de base en SQL : créations de tables et types de données

Une utilisation efficace des bases de données relationnelles n'est réalisable qu'avec un SGBDR (Système de Gestion de Bases d Données Relationnelles), c'esst-à-dire un logiciel offrant :

- la définition des données sous forme de relations ;
- la manipulation des données par un langage déclaratif ;
- l'administration des données.

Dans ce chapitre, nous nous contenterons d'utiliser de manière sommaire des fonctionnalités de SGBDR offertes :

- soit par les notebooks Capytale ;
- soit par le logiciel portable DB Browser for SQLite ;
- soit par l'intermédiaire de modules python.

Nous verrons dans un chapitre post-épreuve les SGBDR un peu plus en détail. Pour l'instant nous nous contenterons des fonctionnalités de base du langage SQL

1. Du modèle relationnel à la base de donnée : une première approche

SQL (*Structured Query Language*) est directement inspiré du modèle relationnel défini dans la partie précédente. Ce langage est standardisé par la norme `ISO/IEC 9075`, dont la dernière version date de 2016.

La syntaxe du SQL est volontairement *verbeuse* et proche de l'anglais standard. Nous allons montrer dans les exemples suivants la création d'une table, l'insertion d'éléments, et quelques requêtes simples sur cette base de donnée.

Exemple : Création de la table `usager`

Les lignes de code SQL suivantes permettent la création d'une **table** `usager`, qui correspond globalement à la **relation** `usager` telle que définie par le schéma :

usager(nom *String*, prénom *String*, email *String*, cp *String*, adresse *String*, inscription *Date*, code_barre *String*)

```
CREATE TABLE usager (nom VARCHAR(90),
                     prenom VARCHAR(90),
                     email VARCHAR(60),
                     cp VARCHAR(5),
                     adresse VARCHAR(90),
                     inscription DATE,
                     code_barre CHAR(15) PRIMARY KEY);
```

On envoie ici au SGBD un ordre SQL de création de table, par l'intermédiare de la commande `CREATE TABLE`. Le nom de la table sera `usager`. Cette table contiendra des **colonnes** (ou **champs** qui correspondent aux **attributs** de la relation. La première colonne `nom` est définie comme étant du type `VARCHAR`, c'est-à-dire chaîne de caractère, avec une contrainte de longueur maximale de 90 caractères. Il en est de même pour les 5 premières colonnes, avec différentes contraintes de longueur. La colonne `inscription` est définie comme étant du type `DATE`. La dernière colonne `code_barre` est définie comme étant du type `VARCHAR` de longueur maximale 15, auquel on adjoint la contrainte `PRIMARY KEY`, qui indique simplement qu'il s'agit bien de la **clé primaire** de la relation.

L'ordre se termine par un `;`.

SQL et case des caractères

SQL est **insensible à la casse**. Il aurait été tout aussi efficace d'écrire `create table` ou `CrEaTE tABlE`.

De fait certaines règles de bonnes pratiques sont à suivre :

- les mots réservés de SQL sont écrits en **majuscule** ;
- les attributs sont écrits en **minuscules**, ainsi que les noms de tables[1] ;
- les noms d'attributs et de tables ne pouvant contenir d'espaces, ceux-ci sont remplacés par le caractère `_` (*underscore*).
- les noms de table sont écrits au singulier.

Exemple : insertions de lignes

Les lignes SQL suivantes permettent l'insertion de trois **lignes** dans la table `usager`, qui correspondent à trois **entités** de la relation `Usager`.

```
INSERT INTO usager VALUES
('Pavie', 'Auguste', 'auguste.pavie@ggp.fr', '22200', '13 rue Anatole Le Braz, Guingamp', '2021-11-09',
'012345678910111'),
('Prevert', 'Jacques', 'jacques.prevert@ggp.fr', '22200', '58 Rue de la Trinité, Guingamp', '2021-11-10',
'012345678910112'),
('Camus', 'Albert', 'albert.camus@grace', '22205', '2 Rue de Kerpaour, Grâces', '2021-10-09',
'012345678910113')
;
```

Chacune des **lignes** est un tuple, chaque composante correspondant à la **colonne** définie dans l'ordre de création de table, **dans l'ordre de définition** (il existe un moyen d'être plus explicite et de s'affranchir de l'ordre, que nous verrons plus tard). Vous pouvez constater que dans la troisième ligne, l'email n'est pas correctement écrit. Nous n'avons pour l'instant pas mis de contraintes supplémentaires sur le **champ** `email`, donc le SGBD acceptera cette entrée comme correcte.

Exemple : Première requête

Nous allons maintenant interroger cette table par l'intermédiaire d'une **requête** (*query* en anglais) :

```
SELECT
nom, prenom
FROM
usager;
```

Cette requête renvoie le résultat suivant :

nom	prenom
Pavie	Auguste
Prevert	Jacques
Camus	Albert

c'est-à-dire une **table** avec les attributs `nom` et `prenom`.

Différences entre modèle relationnel et SQL

Formellement, SQL n'est pas aussi strict que le modèle relationnel. En effet une table ne doit pas obligatoirement posséder une **clé primaire**.

La conséquence directe de ce choix est une violation de la contrainte de relation, et il est tout à fait possible d'avoir des doublons dans une table, ce qui sera toléré par SQL. Mais c'est une mauvaise pratique !

Manipuler SQL

Rendez-vous dans le bac-à-sable SQL sur Capytale.

1. Exécutez les 4 premières cellules.
2. Exécutez la cinquième. Quel est le résultat fournit par cette requête ?
3. Exécutez la sixième. Quel est le résultat fournit par cette requête ?
4. Ajoutez à la base de données l'entité suivante : `('Brochen', 'Charles', 'charles.brochen@pontrieux.fr', '22260', '13 Rue de Pen Fantan, Pontrieux', '2021-10-11', '012345678910113')` Que se passe-t-il ? Pourquoi ?
5. Ajoutez à la base de données l'entité suivante : `('Pavie', 'Auguste', 'auguste.pavie@ggp.fr', '22200', '13 rue Anatole Le Braz, Guingamp', '2021-11-09', '012345678910110')`
   Que se passe-t-il ? Pourquoi ?
6. Exécutez maintenant une requête afin de récupérer une table contenant les emails et code postaux des usagers dont la date d'inscription est le 10 Novembre 2021.

2. Types de données en SQL

3. Créations et suppressions de tables

4. Insertions de données dans une table existante.

1. tout le monde n'est pas forcément d'accord sur ce point. Certains mettent une majuscule à la première lettre du nom de la table... ↩