TP1
Gestion des effectifs d'une formation universitaire
1ère partie
Sujet proposé par L. Marchal , ENS Lyon

BDD
BUT 3 GEII
P.-A. Hébert
2023–2024

Objectif

L'objectif de cette 1ère partie est d'implanter une base de données dans le SGBD MariaDB (appelé ci-dessous MySQL, dont il est dérivé), et de la remplir avec des données fournies sous la forme de fichiers csv.

1 Présentation de la base de données

1.1 Description du problème

Les étudiants d'une université suivent différents cours. Certains cours nécessitent un prérequis (cours précédemment validé). Les cours ont lieu dans des salles de classe sur des créneaux hebdomadaires invariables.

Un étudiant est décrit par son état civil (réduit à nom, prénom, ville) et reçoit un numéro d'étudiant (unique).

À l'issue de chaque module, l'étudiant est évalué et reçoit une note. Lorsque la note manque (« valeur NULL »), cela signifie que l'étudiant est actuellement inscrit à ce cours (l'évaluation n'a pas encore eu lieu).

Un cours se déroule toujours dans la même salle, mais une salle de classe peut accueillir plusieurs cours.

1.2 Description de la base de données relationnelles

La figure 1 présente la structure logique retenue pour cette base de données.

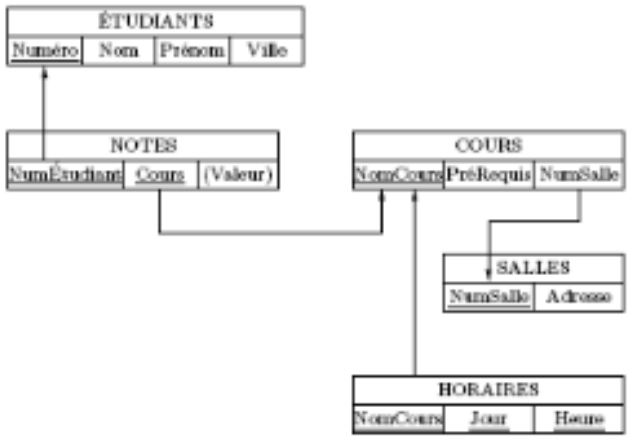

FIGURE 1 – Schéma relationnel de la base de la formation

La figure 2 qui suit montre quelques extraits de données des différentes tables.

SALLES

NumSalle	Adresse
1	Ampère sud
2	Lagrange nord
3	Ampère nord

NOTES

NumÉtudiant	Cours	Valeur
12024	analyse 1	17
12024	analyse 2	15
12024	analyse 3	

COURS

NomCours	PréRequis	NumSalle
analyse 1	aucun	1
analyse 2	analyse 1	1
analyse 3	analyse 2	1

HORAIRES

NomCours	Jour	Heure
analyse 1	lundi	8
analyse 1	mardi	8
analyse 1	jeudi	10

ETUDIANTS

Numéro	Nom	Prénom	Ville
12024	RAVEL	Maurice	SAINT ETIENNE
12042	MALRAUX	Andre	GRENOBLE
12057	FRANK	Cesar	PARIS

FIGURE 2 – Extraits des données

1.3 Brève description du logiciel MySQL

Cette base doit être programmée grâce au logiciel MySQL. C'est un SGBD libre, multi-plateformes, qui permet une intégration aisée dans des applications C, C++, Java, PHP, Python... Il se décompose en deux produits :

- le serveur mysql : il gère physiquement la base, et a pour rôle d'exécuter les requêtes SQL (LDD, LMD ou LCD), tout en respectant les droits des utilisateurs. Il doit pouvoir accéder directement aux tables des bases de données (stockées sous forme de données) ;
- le client mysql : c'est un logiciel léger qui communique avec le serveur mysql ; les commandes SQL qu'il permet de recueillir sont transmises au serveur, qui se charge seul de les exécuter, pour finalement renvoyer les résultats au client. Le client n'est pas nécessairement lancé sur la même machine que le serveur : il exploite les fonctionnalités réseaux d'un système pour communiquer avec le serveur.

L'utilisation d'une base de données se déroule suivant ces différentes étapes :

1. lancement du client console mysql ;
2. connexion et authentification sur le serveur mysql ;
3. sélection de la base ;
4. envoi de requêtes SQL.

Dans ce TP, vous lancerez le client et le serveur sur une même machine (la vôtre !).

Plutôt que de lancer le client mysql dans la console, il est possible d'utiliser un autre client : l'interface web *phpMyAdmin*.

2 Construction de la base

2.1 Création de sa structure

Il s'agit tout d'abord de construire la structure relationnelle de la base de données.

Après avoir lancé le serveur mysql (se renseigner sur UwAmp), vous lancerez le client de votre choix, soit *phpMyAdmin* dans votre navigateur web, soit le client console mysql, en précisant votre statut de « super-utilisateur » du SGBD de votre machine (`mysql -u root`).

Une fois connecté vous devez :

1. créer la base :

```
CREATE DATABASE univ;
```

2. en faire la base de travail (celle qui sera utilisée à moins qu'une autre base soit nommée) :

```
USE univ;
```

3. créer sa structure : soit l'ensemble de ses tables, avec dans chacune un maximum de contraintes (domaine, intégrité).

Vous allez ainsi créer toutes les tables, c'est la tâche la plus longue.

N'oubliez pas de bien réfléchir à l'ordre de création des tables, de manière à éviter de déclarer une clé étrangère faisant référence à une table non encore créée : les instructions SQL étant interprétées dans l'ordre dans lequel elles sont données, une instruction ne peut référencer un élément (table, champ) que si cet élément a déjà été défini au-dessus.

Il est conseillé de rédiger l'ensemble des instructions SQL dans un fichier texte d'extension .sql *: ainsi vous gardez trace de votre travail, et vous avez à tout moment la possibilité de corriger une instruction. Le copier/coller dans la console fonctionne.*

Attention : aussitôt après avoir créer une table, il est fortement conseillé de vous rendre à l'étape suivante, pour la remplir *avec les données fournies : ce sera un moyen de vérifier que la table a été correctement créée.*

2.2 Remplissage de la base

La structure de la base étant définie, il s'agit maintenant d'alimenter ses tables en données. Celles-ci sont fournies dans des fichiers « texte », correspondant chacun à une table. Ces fichiers sont formatés CSV : les champs sont séparés par des tabulations, les lignes par des sauts, et les valeurs NULL sont spécifiées explicitement (NULL).

Pour intégrer ces données dans la base, suivez la procédure suivante :

1. localisez les fichiers texte sur votre machine ;
2. pour chacun des fichiers, lancer la commande suivante :

```
LOAD DATA LOCAL INFILE 'cheminDuFichier' INTO TABLE nomDeLaTable;
```

2.3 Extraction d'informations en ligne de commandes

Nous travaillons ici avec un client en ligne de commandes. Il faut donc directement taper les instructions à l'invite « `mysql>` » :

— pour lister les bases de données disponibles :

```
SHOW DATABASES;
```

— pour voir les tables de la base de données courante :

```
SHOW TABLES;
```

— pour voir la description d'une table :

```
DESCRIBE nomDeLaTable;
```

— pour quitter le client MySQL :

```
QUIT;
```