Serveur de base de données MySQL

Présentation

Le *serveur de base de données* est un outil indépendant d'un serveur Web.

Son rôle est de permettre le stockage, l'exploitation et la sécurisation de données structurées en bases de données (un nom pour un **domaine de gestion**) au sein desquelles seront visibles les **tables** et **occurrences**.

Le serveur MySql est un outil adapté à une gestion de **bases de données relationnelles**, permettant d'assurer de la redondance de serveur, de l'exécution de **procédures stockées** ou de **déclencheurs**.

Procédures

Installation du service

1. Installer le paquetage *mysql-server*
2. Au cours de l'installation, penser à donner un **mot de passe** à l'utilisateur *root* (et penser à s'en souvenir 😉 !). Pour certaines installations, la mise en place d'un mot de passe root passe par l'exécution de la commande ***mysql-secure-installation*** après l'installation
3. Vérifier la connexion : ***mysql -u <nomUtilisateur> [-p[<mot_passe>]] [<nomBaseDonnees>]***
   (*-p* permet de saisir un mot de passe à la connexion ou de le fournir directement, on peut se connecter directement à la base de données voulue)

Administration par PhpMyAdmin

PHPMyAdmin est une surcouche graphique permettant d'administrer la base de données MySQL par des pages Web (en PHP). On procèdera d'abord à l'installation d'Apache et PHP conformément à la procédure ici.

On pourra aussi utiliser *phpmyadmin* pour administrer depuis un point unique plusieurs bases de données : voir ici.

Gestion de MySql

Il ne s'agit pas ici de faire un cours sur le langage SQL mais de présenter les principales commandes d'utilisation en dehors de l'interface graphique *PhpMyAdmin*.

MYSQL sans interface graphique

Après l'installation du service, on intervient dans MySql par la console en ligne de commande.

Se connecter à MySQL

La commande ***mysql*** tente une connexion avec le compte *root* sans mot de passe. Elle devrait normalement déboucher sur un échec.

Syntaxe

Connexion avec demande du mot de passe

```
mysql -u <nomUtil> [-p] [<nom_base_de_donnees>]
/* Permet la connexion sous un nom d'utilisateur, avec demande du mot de passe.
On peut aussi directement utiliser une base de données.*/
```

Connexion en fournissant le mot de passe

```
mysql -u <nomUtil> [-p<motPasse>]
[<nom_base_de_donnees>]
/* Permet la connexion sous un nom
d'utilisateur, avec fourniture du mot de
passe.
On peut aussi directement utiliser une base
de données.*/
```

Attention : avec cet syntaxe, on peut retrouver le mot de passe dans l'historique des commandes

Connexion à une machine distante

```
mysql -u <nomUtil> [-p<motPasse>] -h <nom_ou_adresse_serveur_distant>
[<nom_base_de_donnees>]
```

Remarque : Pour des raisons de sécurité, le compte *root* de *MySql* n'est pas autorisé à établir des connexions à la base depuis d'autres machines que *localhost*.

On devra donc procéder comme suit sur le SGBD pour permettre une connexion distante avec un autre compte :

1. Créer un utilisateur *MySQL* avec mot de passe

```
CREATE USER <nomutil> IDENTIFIED BY '<motPasse>' ;
```

2. Lui donner le droit de se connecter depuis toute machine

```
GRANT usage ON *.* TO '<nomutil>'@'%';
```

3. Lui donner éventuellement les droits nécessaires sur la base de données particulière

Exemple

```
mysql -u uGSB -pbazGSB;
```

Pour utiliser une base particulière, on a recours à la commande ***use***

```
USE <nom_base_de_donnees>;
```

Modifier le mot de passe d'un compte

Les comptes de MySql sont stockés dans la base ***mysql*** créée lors de l'installation. Pour modifier le mot de passe et l'exiger à chaque connexion, on sélectionnera cette base et on modifiera le contenu de la table *user* :

```
SET PASSWORD FOR 'Utilisateur'@'localhost' = PASSWORD("MotDePasse");

FLUSH privileges;

ALTER USER 'Utilisateur'@'localhost' IDENTIFIED BY 'MotDePasse';

FLUSH privileges;
```

Autoriser l'interrogation distante de MySql

Par défaut, l'installation de MySql n'autorise les connexions que depuis *localhost* ou *127.0.0.1*.

Pour permettre une connexion depuis une autre machine (un serveur Web, par exemple) :

- dans le fichier ***/etc/mysql/mariadb.conf.d/50-server.cnf***

```
nano  /etc/mysql/mariadb.conf.d/50-server.cnf
```

  - modifier la valeur ***bind-address*** qui indique les adresses du serveur MySql sur lesquelles il est en écoute, et y mettre la valeur **0.0.0.0** (en écoute sur tous les ports) :

```
bind-address=<ip_d_ecoute>    #par défaut, la valeur est limitée à 127.0.0.1 ou localhost
                              #la valeur 0.0.0.0 autorise la connexion depuis toutes les IP du serveur MySQL/MariaDB
```

On redémarrera alors le service Mysql :

```
service mysql-server restart
//ou
service mysql restart
```

Objets

Pour connaître/créer les objets d'un serveur MySQL, on a recours aux commandes suivantes :

Commande	Explication
SHOW DATABASES	Affiche la liste des bases de données
CREATE DATABASE nomBase	Crée un conteneur de base de données (il n'y a pas de tables)
SHOW TABLES	Liste les tables contenues dans une base de données
DESC nom_table	Présente la structure de la table nom_table
CREATE TABLE nom_table	Crée une table (il faudra aussi décrire ses champs et leurs propriétés, ainsi que les contraintes)
DROP TABLE nom_table	Détruit la table (structure et contenu)
ALTER TABLE nom_table	Modifie la structure de la table (type d'un champ, ajout de colonne, ajout de contrainte d'intégrité, etc).

Gestion des droits

Les clauses SQL de base pour la création des comptes utilisateur et la gestion des droits sont les suivantes :

Clause	Explication	Exemple
CREATE USER	Crée un compte utilisateur	`CREATE USER compta IDENTIFIED BY 'mpcompta'`

GRANT	Attribue des privilèges à un compte	`GRANT USAGE ON bddComptes.* TO compta@'nom_machine'`
		`/* donne des droits d'utilisation sur toutes les tables de la base de données bddComptes`
		`au compte compta depuis la machine nom_machine */`

REVOKE	Retire des privilèges à un compte	`REVOKE SELECT ON bdGest.employes FROM compta`
		`/* retire le droit de lire (select) la table employes`
		`de la base bdGest au compte compta */`

SHOW GRANTS	Voir les droits d'utilisateur	`SHOW GRANTS FOR <USER>`

Sauvegarde et restauration : mysqldump

La sauvegarde et la restauration des bases de données passent par la commande ***mysqldump*** (utilisée par l'interface *PhpMyAdmin* .

On trouvera la syntaxe complète ici.

Prérequis

- Pour effectuer une sauvegarde, il faut disposer d'un compte autorisé à lire les données de la base ciblée (ou de toutes les bases si on sauvegarde un serveur complet) et à verrouiller les tables (*LOCK_TABLES*).
- Si la base est sur un autre serveur, il faudra autoriser la connexion distante au serveur cible depuis la machine de sauvegarde (fichier my.cnf) et autoriser le compte à agir à distance (privilèges).
- Pour récupérer le *dump* généré par la commande, on redirigera le résultat vers un fichier accessible en écriture (existant ou créé lors de l'exécution).
- Pour restaurer une base, il faut disposer des droits correspondant aux clauses contenues dans le fichier *dump* (Create, insert, etc).

Syntaxes

Sauvegarde des bases

- Sauvegarde de toutes les bases d'un serveur :

```
mysqldump [-u <compte> [-p<motPasse>]] --all-databases >
fichierSortie.sql
/* Attention : le mot de passe est collé au "-p" */
```

- Sauvegarde d'une liste de bases de données :

```
mysqldump [-u <compte> [-p<motPasse>]] --database <base1> [<base2>]
[<base3>] [...] > fichierSortie.sql
```

- Sauvegarde d'une base de données :

```
mysqldump [-u <compte> [-p<motPasse>]] <base> > fichierSortie.sql
```

- Sauvegarde d'une partie de base de données :

```
mysqldump [-u <compte> [-p<motPasse>]] <base> [<table1>] [<table2>] [-w
"<condition where>"] > fichierSortie.sql
```

Restauration

La restauration consiste à faire exécuter un script SQL au serveur MySql.

- Restaurer à partir d'un fichier :

```
mysql [-u <compte> [-p<motPasse>]] [-h <serveurDistant>]
[<baseExistante>] < <fichierBackup.sql>
/* On peut exécuter le script sur une base existante ou faire la
création de la base dans le script */
```

Migration de bases de données

Principes

La migration d'une base de données consiste à prendre un contenu (structure et données) et à le transférer dans un environnement différent : par exemple une base existant sous Acces que l'on souhaite transférer vers un environnement Serveur améliorant les performances et les possibilités (procédures stockées ou accès parallèles).

On peut le faire de plusieurs façons. En voici quelques unes.

Remarque : Le code SQL proposé est à adapter aux possibilités de syntaxe de l'environnement.

Deux bases en parallèle, recopie à l'identique : les deux bases doivent disposer d'un environnement graphique sous Windows permettant l'utilisation d'une source de données ***ODBC (Open Database Connectivity)***.

1. Créer une base N dans le nouvel environnement,
2. Connecter l'ancienne et la nouvelle par un lien *ODBC* et des tables liées
3. Procéder par SQL :

```
CREATE TABLE <maNvlleBase>.<maTable> AS SELECT * FROM
<monAncienneBase>.<maTable>
    /* les tables source et destination seront exactement identiques
(Structures / Données) */
```

4. Créer les requêtes permettant de re-générer les contraintes d'intégrité :

```
ALTER TABLE <nomTable> ADD CONSTRAINT <typeContrainte>
<DetailContrainte>
```

Deux bases en parallèles, structures différentes, insertion de données : L'environnement de l'ancienne base doit disposer d'un environnement permettant la connexion *ODBC* ou équivalente, il doit exister un pilote *ODBC* pour accéder à l'ancien SGBD.

1. Faire du ***reverse engeneering*** sur l'ancienne base
2. Apporter les modifications à la structure de la base
3. Générer le ***script*** de création dans le nouvel environnement
4. Implanter la base (*structure*) dans le nouvel environnement grâce au script
5. Les **contraintes d'intégrité** seront existantes dans la nouvelle base
6. Connecter l'ancienne et la nouvelle
7. Procéder par SQL, en pensant à l'ordre de réalisation des insertions du fait des contraintes d'intégrité :

```
INSERT INTO <maNvlleBase>.<maTable> (<liste_des_champs>)
  AS SELECT <liste_des_champs> FROM <monAncienneBase>.<maTable>
```

Script d'insertion

1. Faire du reverse engeneering sur l'ancienne base, adapter éventuellement la structure, générer le script pour le nouvel environnement, l'implanter
2. Sur l'ancienne base, créer un fichier qui contiendra les données à insérer (ordres SQL, données au format XML, fichier CSV, ...). On vérifiera dans le script que l'ordre de réalisation des insertions respecte les contraintes d'intégrité

   ```
   INSERT INTO (<liste_champs>) VALUES (<liste_valeurs>)
   ```

   → le script peut être généré par des fonctions d'export du SGBD, ou par une programmation dans un langage (VBA, PHP, autre)
   → Access ou MySQL ont des fonctions permettant de générer automatiquement ce script (en SQL, en XML ou d'autres formats)

3. Exécuter le script ou importer le fichier de données dans la nouvelle base.

Des outils existent pour faire des conversions/migrations entre différents environnements.