Bases de données
Gestion des utilisateurs et des droits
Dictionnaires : BD mysql, BD information_schema

Bertrand LIAUDET

SOMMAIRE

Dernière édition : octobre 2016

GESTION DES UTILISATEURS
ET DE LEURS PRIVILEGES (DROITS)

Gestion des utilisateurs et sécurité

Présentation

La gestion des utilisateurs relève globalement de la **politique de sécurité**.

La politique de sécurité se développe sur plusieurs plans :

- le plan SE
- le plan SGBD
- le plan application.

La base de la protection des données au plan SGBD est fondée sur :

- L'accès au serveur : ça relève de la gestion des utilisateurs.
- L'accès aux données : ça relève de la gestion des privilèges.

Les règles élémentaires de sécurité au niveau SE

Principe

Protéger l'environnement du serveur.

Ca relève des compétences de l'administrateur système.

Quelques règles

- **Protéger l'hôte du serveur tout entier** et pas seulement le serveur MySQL (mysqld) contre tout type d'attaque : écoute, altération, déni de service, etc.
- **Protéger l'hôte du serveur avec un pare-feu** et placer MySQL derrière le pare-feu.
- **Protéger le port utilisé par MySQL** (par défaut le port 3306) : il ne doit pas être accessible par des hôtes qui ne sont pas de confiance.

Pour connaître l'hôte serveur :

```
shell> hostname
```

Pour vérifier la situation du port MySQL :

```
shell> telnet hôte_serveur 3306
```

Si telnet permet la connexion et renvoie des caractères bizarres, le port est ouvert et doit être fermé sur le pare-feu ou le routeur.

Si telnet se suspend ou si la connexion est refusée : le port est bloqué.

- **Eviter d'exécuter le serveur MySQL (mysqld) en tant que root SE**. En effet, un utilisateur MySQL ayant le privilège FILE pourrait alors créer des fichiers en tant que root. mysql_safe permet de préciser l'utilisateur SE qui exécute mysqld.
- **S'assurer que le seul utilisateur SE avec des privilèges en lecture et en écriture dans le DATADIR soit l'utilisateur qui exécute le serveur.** De même pour les fichiers de configuration et pour les programmes et les scripts.
- **S'assurer de la confidentialité des données qui circulent** : à part le mot de passe, toutes les données qui circulent sont transférées sous forme de texte. Toute personne capable de surveiller la connexion peut donc lire ces données.

A tester :

```
shell> tcpdump –l –i eth0 –w – src or dst port 3306 | strings
```

Cette commande, sous linux, doit permettre de vérifier si les flux de données de MySQL sont cryptés. Si des données en texte clair apparaissent, c'est que les données ne sont pas cryptées. L'inverse n'est pas certain.

On peut utiliser les protocoles cryptés SSL ou SSH pour crypter les données qui circulent sur internet.

Les règles élémentaires de sécurité au niveau SGBD

Principe

Protéger l'utilisation faite du serveur.

Ca concerne essentiellement la gestion des droits des utilisateurs.

Ca relève des compétences de l'administrateur système et aussi du développeur.

Quelques règles

- Tous les utilisateurs (clients, applications) doivent avoir un mot de passe.
- Contrôler les droits de tous les utilisateurs. Utiliser la commande « show grants ».
- Le compte root doit avoir un mot de passe ! Essayer mysql –u root : si ça passe, il y a un problème !
- Il faut éviter de créer plusieurs comptes avec des droits d'administrateur.
- Ne stocker aucun mot de passe en clair. Utilisez toujours une fonction de cryptage ou de hachage à sens unique : password().
- La table user de la database mysql ne doit être accessible que par root ! La table user contient les mots de passe. Ils doivent être cryptés. Le mot de passe crypté est le véritable mot de passe dans MySQL. Un « drop database mysql » doit être impossible par quiconque n'est pas root !
- Choisir des mots de passe assez compliqués et éviter de les perdre ! Trouver des systèmes mnémotechniques.
- Les privilèges PROCESS, SUPER et FILE ne doivent être accordés qu'aux administrateurs.
  - ✓ PROCESS permet aux commandes : « mysqladmin processlist » ou à « show full processlist » d'accéder en clair à toutes les requêtes en cours d'exécution de tous les utilisateurs, donc un SET PASSWORD !!!
  - ✓ SUPER autorise une connexion unique même si max_connections est atteint ; il permet aussi de terminer des connexions clientes.
  - ✓ FILE donne le droit d'écrire dans le SE avec les privilèges du propriétaire de mysqld.

Les règles élémentaires au niveau du développeur

- **Ne pas faire confiance aux données entrées par les utilisateurs !** Par exemple : si on attend un ID pour un select et que l'utilisateur entre « 234 or 1=1 », et que l'application envoie ça au serveur, alors toute la table sera sélectionnée ! La solution simple est de mettre ce qui est saisie entre quottes simples : '234 or 1=1'. Ainsi, le résultat sera nul.

  Le problème posé par de tel comportement est triple :

  - ✓ Il permet un accès à des données éventuellement confidentielles : déni de sécurité
  - ✓ Il engendre une surcharger du serveur : déni de service.
  - ✓ Il permet d'afficher des informations non contrôlées : déni de service.

  C'est la notion d'**injection SQL**

  Pour se protéger des injections SQL, on peut :

  - ✓ Mettre la saisie entre quottes simples
  - ✓ Utiliser des procédures stockées
  - ✓ Limiter les droits des utilisateurs

- **Limiter les usages des utilisateurs**

- **Limiter les droits des utilisateurs**

Options de démarrage de mysqld relatives à la sécurité

http://dev.mysql.com/doc/refman/5.0/fr/privileges-options.html

--skip-grant-tables

Cette option force le serveur à ne pas utiliser le système de privilèges (donc la table des droits). Cette option donne donc tous les droits à tout le monde sur le serveur ! Elle est utile en cas de perte de mot de passe root ! En général, on l'associe à un skip-networking.

--skip-net-working

Cette option interdit toute connexion au serveur autrement que depuis l'hôte local. Quand on démarre le serveur en skip-grant-tables, on ajoute skip-net-working !

--skip-show-database

Avec cette option, l'instruction SHOW DATABASES n'est autorisée que pour les utilisateurs qui possèdent le privilèges SHOW DATABASES. L'instruction affiche tous les noms de base de données. Sans cette option, SHOW DATABASES est autorisée pour tous les utilisateurs, mais elle n'affiche les noms des BD que si l'utilisateur possède des privilèges pour la BD.

Protéger le mot de passe

http://dev.mysql.com/doc/refman/5.0/fr/password-security.html

Eviter de taper :

```
shell> mysql –uroot -pmdpRoot
```

Faire plutôt :

```
shell> mysql –uroot –p
Enter password: *********
```

On peut aussi mettre le mot de passe dans le fichier .my.cnf

```
[client]
password=mdpRoot
```

avec en plus :

```
shell> chmod 600 .my.cnf
```

Pour rendre le fichier .my.cnf innaccessible.

Ces deux techniques sont correctement sécurisées.

Gestion des utilisateurs et de leurs mots de passe

Résumé

Le DCL : data control language

gestion des utilisateurs : **Create user, Drop user, Set password, Grant, Revoke**

Définition d'un utilisateur

Un utilisateur MySQL est caractérisé par son nom et par son hôte (nom ou adresse IP de la machine) : **user, host**

Création d'un utilisateur

```
CREATE USER nomUtilisateur[@nomHote] [IDENTIFIED BY motDePasse]
```

Connexion au client mysql

```
shell> mysql –unomUtilisateur –hnomHote
```

Consultation de la liste des utilisateurs

```
mysql> select host, user, password from mysql.user;
```

Connaître l'identité effective : user()

```
mysql> select user() ;
```

Création d'un utilisateur avec un GRANT

La commande GRANT appliquée à un utilisateur n'existant pas crée cet utilisateur.

```
GRANT privilege [,privilege] ON composant TO nomUtilisateur
IDENTIFIED BY motDePasse [WITH GRANT OPTION]
```

- ***Exemple de création d'un utilisateur sans droits***

```
mysql> GRANT USAGE ON *.* TO toto@localhost IDENTIFIED BY 'mdptoto';
```

L'utilisateur toto peut se connecter sur n'importe quelle machine et n'a aucun droit. Il accède uniquement à la BD information_schema.

On donne d'abord l'USAGE, puis des droits particulier.

ON *.* veut dire : toutes les BD . toutes les tables

- ***Exemple de création d'un utilisateur qui peut tout consulter***

```
mysql> GRANT SELECT ON *.* TO toto@localhost IDENTIFIED BY 'mdptoto';
```

- ***Exemple de création d'un super-utilisateur avec tous les droits***

```
mysql> GRANT ALL PRIVILEGES ON *.* TO admin@localhost IDENTIFIED
BY 'mdpAdmin' WITH GRANT OPTION;
```

Suppression d'un utilisateur

```
DROP USER nomUtilisateur[@nomHote]
```

Modification du password

```
mysql> SET PASSWORD [FOR nomUtilisateur ] =
    PASSWORD('motDePasse')
```

La fonction PASSWORD( ) permet de crypter le mot de passe.

Suppression du password

```
mysql> SET PASSWORD FOR nomUtilisateur = '' ;
```

Consulter les passwords cryptés dans la table des utilisateurs

```
mysql> select host, user, password from mysql.user;
```

Présentation

Le DCL

Les instructions du DCL (data control language) du SQL permettent la gestion des utilisateurs :

Create user, Rename user, Drop user, Set password, Grant, Revoke

Définition d'un utilisateur

Un utilisateur MySQL est caractérisé par son nom et par son hôte (nom ou adresse IP de la machine) :

user, host

Les privilèges concernent ce couple. Un même utilisateur (même nom) peut donc avoir des privilèges différents selon l'hôte où il se situe.

Précisions sur les hôtes

- ***L'hôte c'est le nom ou l'adresse IP de la machine.***

  « localhost » c'est le nom de l'hôte du serveur. Un utilisateur dont l'hôte est localhost ne peut se connecter qu'à partir de la machine du serveur. localhost peut être remplacé par 127.0.0.1

  % dans le nom de l'hôte est remplacé par n'importe quelle chaîne. On peut aussi écrire : @'%.insia.com'

  user@'%' est équivalent à user tout seul : ça permet la connexion de user à partir de tous les hôtes. user@'' permet aussi la connexion de user à partir de tous les hôtes.

- ***Accès à l'hôte et l'IP sous XP :***

```
C:>ipconfig /all
```

Dans un script .bat :

```
cmd /k ipconfig /all
```

ou

```
ipconfig /all
pause
```

Précisions sur les utilisateurs

- ***Tout le monde, de partout***

  L'utilisateur permettant à n'importe qui de se connecter, c'est : '' : à éviter !!!

  ''@'%' (équivalent à '' sans hôte) ou ''@'' : à éviter !!!!

- ***L'utilisateur anonyme : %, par convention***

  Le % dans un nom d'utilisateur est un caractère comme un autre.

  % comme nom d'utilisateur, c'est, par convention, l'utilisateur anonyme : ce n'est pas remplaçable par n'importe quel utilisateur.

Consultation de la liste des utilisateurs

Consultation directe de la table des USER dans la BD mysql :

```
mysql> select host, user, password from mysql.user;
```

Création d'un utilisateur – 1 : CREATE USER

Création d'un utilisateur

http://dev.mysql.com/doc/refman/5.0/fr/create-user.html

La commande CREATE USER crée un utilisateur qui n'a aucun droit. Il ne peut donc rien faire avec la BD. Il faudra ensuite donner des droits avec un GRANT.

```
CREATE USER nomUtilisateur[@nomHote] [IDENTIFIED BY motDePasse]
```

Pour créer un utilisateur, il faut être un utilisateur qui a le droit de créer des utilisateurs (privilège GRANT).

- ***Précisions sur les password***

  IDENFIED BY 'motDePasse' : permet de donner un password. Le password proposé sera ensuite crypté par MySQL avec la fonction PASSWORD( ) ;

Connexion au client mysql

- ***Syntaxe***

```
shell> mysql –u nomUtilisateur –h nomHote
```

ou

```
shell> mysql –unomUtilisateur –hnomHote
```

- ***Paramétrage***

  Dans le fichier de configuration my.ini, la variable 'bind-address » ne doit pas avoir de valeur. Si elle existe, il faut la mettre en commentaire : # au début de la ligne.

- ***Exemple***

Avec

```
mysql> select host, user from mysql.user;
+-----------+------+
| host      | user |
+-----------+------+
| localhost | root |
+-----------+------+
```

On peut se connecter, à partir de la machine du serveur :

```
C:\WINDOWS>mysql –u root –p
```

Ou

```
C:\WINDOWS>mysql –u root –h localhost –p
```

Ou

```
C:\WINDOWS>mysql –u root –h 127.0.0.1 -p
```

Consultation de la liste des utilisateurs

- ***Consultation directe de la table des USER***

```
mysql> select host, user, password from mysql.user;
```

Profil de connexion et identité effective

- ***Définitions***

  On distingue entre **profil de connexion** et **identité effective** : l'identité effective instancie le profil de connexion : elle instancie l'hôte et l'utilisateur.

  Les profils sont des classes d'utilisateurs provenant d'une création pour tous les utilisateurs et/ou pour un ensemble d'hôtes (avec un % dans le nom de l'hôte).

- ***Connaître le profil de connexion : current_user()***

```
mysql> select current_user() ;
```

  Le résultat peut être : moi@%, moi@localhost, moi@maMachine

- ***Connaître l'identité effective : user()***

```
mysql> select user() ;
```

  Le résultat peut être : moi@localhost, moi@maMachine

  Il ne sera pas : moi@%

Choix du profil de connexion à la connexion

- ***Principe***
  - Quand on se connecte, le SGBD reçoit un nom et un hôte concret. Le SGBD va donc définir le profil de connexion correspondant à cet utilisateur. Le principe est que le SGBD choisit le nom et l'hôte le plus spécifique.

- ***Règles concrètes***
  - MySQL choisit d'abord l'hôte le plus spécifique (localhost de préférence à %, par exemple).
  - MySQL choisit ensuite le nom d'utilisateur le plus spécifique ('toto' de préférence à '', par exemple).
  - L'utilisateur % reste % puisque % est un caractère comme un autre pour le nom d'utilisateur.
  - L'utilisateur sans nom devient 'ODBC'
  - L'hôte sans nom ou % devient le nom de la machine hôte ou 'localhost' si la machine hôte est celle du serveur.
  - Tous les autres profils d'hôte deviennent celui de la machine hôte.

Création d'un utilisateur – 2 : GRANT

GRANT

La commande GRANT appliquée à un utilisateur n'existant pas crée cet utilisateur.

```
GRANT privilege [,privilege] ON composant TO nomUtilisateur
IDENTIFIED BY motDePasse [WITH GRANT OPTION]
```

- ***Exemple de création d'un utilisateur sans droits***

```
mysql> GRANT USAGE ON *.* TO toto@localhost IDENTIFIED BY 'mdptoto';
```

L'utilisateur toto peut se connecter sur n'importe quelle machine et n'a aucun droit. Il accède uniquement à la BD information_schema.

On donne d'abord l'USAGE, puis des droits particulier.

ON *.* veut dire : toutes les BD . toutes les tables

- ***Exemple de création d'un utilisateur qui peut tout consulter***

```
mysql> GRANT SELECT ON *.* TO toto@localhost IDENTIFIED BY 'mdptoto';
```

- ***Exemple de création d'un super-utilisateur***

```
mysql> GRANT ALL PRIVILEGES ON *.* TO admin@localhost IDENTIFIED
BY 'mdpAdmin' WITH GRANT OPTION;
```

L'utilisateur admin2 peut se connecter sur la machine du serveur et a les mêmes droits que l'administrateur root.

Il faut éviter de multiplier les administrateurs !!!

Création d'un utilisateur – 3 : mysql .user

table des USER

```
mysql> use mysql
mysql> insert into user (host, user, ssl_cipher, x509_issuer,
x509_subject) values ('%', 'toto', '', '','');
mysql> flush privileges ;// pour prendre en compte la modification
```

Les attributs ssl_cipher, x509_issuer, x509_subject doivent être renseignés. Une chaîne vide suffit.

La commande précédente est équivalente à :

```
mysql> create user toto;
```

Il faut éviter de passer directement par les tables de la BD mysql ! C'est toutefois parfois utile pour des commandes de masse (surtout en modification).

Modification d'un utilisateur

Par la commande RENAME USER

```
RENAME USER nomUtilisateur[@nomHote]
```

Renommer un utilisateur permet de changer son nom et son hôte en gardant le mot de passe et les droits.

Par la modification directe de la table des USER

On peut faire des update directement dans la BD mysql, mais il faut aussi modifier tous les autres objets qui y étaient attachés.

Il faut éviter de passer directement par les tables de la BD mysql ! C'est toutefois parfois utile pour des commandes de masse (surtout en modification).

Suppression d'un utilisateur

Par la commande DROP USER

```
DROP USER nomUtilisateur[@nomHote]
```

La suppression d'un utilisateur supprime aussi toutes les BD, toutes les tables et tous les autres objets qui y étaient attachés (cf. schéma de la BD mysql : logique de « on delete cascade »).

Par la modification directe de la table des USER

On peut faire des delete directement dans la BD mysql, mais il faut aussi supprimer tous les autres objets qui y était attachés.

Il faut éviter de passer directement par les tables de la BD mysql ! C'est toutefois parfois utile pour des commandes de masse (surtout en modification).

Modification des mots de passe

Par la commande SET PASSWORD FOR

```
mysql> SET PASSWORD [FOR nomUtilisateur ] =
    PASSWORD('motDePasse')
```

La fonction PASSWORD( ) permet de crypter le mot de passe.

Par la modification directe de la table des USER

```
shell> mysql uroot –p
mysql> UPDATE mysql.user
          SET password = PASSWORD('motDePasse')
          WHERE User='root';
mysql> FLUSH PRIVILEGES; // pour prendre en compte la modification
```

Il faut éviter de passer directement par les tables de la BD mysql ! C'est toutefois parfois utile pour des commandes de masse (surtout en modification).

Suppression des mots de passe

Par la commande SET PASSWORD FOR

```
mysql> SET PASSWORD FOR nomUtilisateur = '' ;
```

La fonction PASSWORD( ) permet de crypter le mot de passe.

Par la modification directe de la table des USER

```
shell> mysql uroot –p
mysql> UPDATE mysql.user
          SET password = ''
          WHERE User='root';
mysql> FLUSH PRIVILEGES; // pour prendre en compte la modification
```

Il faut éviter de passer directement par les tables de la BD mysql ! C'est toutefois parfois utile pour des commandes de masse (surtout en modification).

Gestion MySQL des mots de passe

En consultant la liste des utilisateurs, on voit les mots de passe cryptés apparaître :

```
mysql> select host, user, password from mysql.user;
```

```
mysql> create user moi@localhost identified by 'monMotDePasse';

mysql> select host, user, password from mysql.user;
+-----------+------+-------------------------------------------+
| host      | user | password                                  |
+-----------+------+-------------------------------------------+
| localhost | moi  | *7800185BB24C689804A2E0BDDC5940E68D70735A |
```

```
+-----------+------+-------------------------------------------+
```

Le password est une chaîne suivie de 40 chiffres hexadécimaux

La fonction de cryptage est déterministe : un même mot de passe non crypté produit toujours le même mot de passe crypté, mais irréversible : on ne peut pas retrouver le mot de passe non crypté à partir du mot de passe crypté.

Création d'un utilisateur avec password crypté

```
CREATE USER re-moi@localhost
IDENTIFIED BY PASSWORD
    '*7800185BB24C689804A2E0BDDC5940E68D70735A'
```

Attention à distinguer le mot clé PASSWORD de la fonction PASSWORD !

Insertion ou modification directement dans la table des USER

Exemple :

```
shell> mysql uroot –p
mysql> UPDATE mysql.user
          SET password =
    '*7800185BB24C689804A2E0BDDC5940E68D70735A'
          WHERE User='moi';
mysql> FLUSH PRIVILEGES; // pour prendre en compte la modification
```

Gestion des privilèges (privilège = droit)

http://dev.mysql.com/doc/refman/5.0/fr/grant.html

Résumé

Principes

La notion de privilège est identique à celle de droit.

Les droits s'appliquent aux utilisateurs selon leur hôte (selon la machine d'où ils se connectent).

Consulter les privileges d'un utilisateur

Pour n'importe quel utilisateur :

```
SHOW GRANTS [FOR user];
```

Pour soi-même :

```
mysql> SHOW GRANTS;
```

Liste des privilèges

```
SHOW PRIVILEGES ;
```

GRANT : pour donner des privileges

- ***Syntaxe***

```
GRANT privilege [,privilege] ON composant TO user[@host]
[IDENTIFIED BY motDePasse] [WITH GRANT OPTION]
```

- ***Création d'un utilisateur avec tous les droits :***

. : toutes les BD.toutes les tables

```
mysql> GRANT ALL PRIVILEGES ON *.* TO admin2@localhost
IDENTIFIED BY 'mdpAdmin' WITH GRANT OPTION;
```

- ***Donner des droits de consultation sur certaines colonnes :***

```
mysql> GRANT SELECT (ne, nom) ON empdept.emp TO toto@'%';
```

REVOKE : pour retirer des privilèges

- ***Syntaxe***

```
REVOKE privilege [,privilege] ON composant FROM user[@host]
```

- ***Suppression d'un droit***

```
mysql> REVOKE select (nom) on empdept.emp from toto@'%';
```

- ***Suppression de tous les droits sauf le grant option:***

```
mysql> REVOKE all on *.* from admin2@localhost;
```

Limiter les ressources d'un utilisateur

```
mysql> GRANT USAGE ON *.* TO 'toto'@'localhost'
```

```
WITH MAX_QUERIES_PER_HOUR 20
MAX_UPDATES_PER_HOUR 10
MAX_CONNECTIONS_PER_HOUR 5
MAX_USER_CONNECTIONS 2;
```

Privilège et droit

La notion de privilège est identique à celle de droit.

Standard SQL

La gestion des privilèges MySQL suit en partie la norme SQL.

Il lui manque la gestion des rôles qu'on trouve sous ORACLE (un rôle est une sorte d'utilisateur abstrait avec des droits. Un utilisateur concret peut alors avoir certains rôles).

Elle distingue les utilisateurs selon leur hôte : la machine d'où ils se connectent.

Donner des privilèges : GRANT … ON … TO

Syntaxe simplifiée

```
GRANT privilege [,privilege] ON composant TO user[@host]
[IDENTIFIED BY motDePasse] [WITH GRANT OPTION]
```

Syntaxe complète

```
GRANT priv_type [(column_list)] [, priv_type [(column_list)]] ...
ON {tbl_name | * | *.* | db_name.*}
TO user [IDENTIFIED BY [PASSWORD] 'password']
    [, user [IDENTIFIED BY [PASSWORD] 'password']] ...
[REQUIRE NONE
    |[{SSL| X509}]
    [CIPHER cipher [AND]]
    [ISSUER issuer [AND]]
    [SUBJECT subject]]
[WITH [GRANT OPTION | MAX_QUERIES_PER_HOUR count
                | MAX_UPDATES_PER_HOUR count
                | MAX_CONNECTIONS_PER_HOUR count]]
```

Création d'utilisateur avec GRANT

Le GRANT s'applique à un utilisateur existant déjà. S'il n'existe pas, le GRANT fait office de CREATE USER. C'est à cela que sert le IDENTIFIED BY…

Tous les privilèges

Mot clé : ALL ou ALL PRIVILEGES à la place d'une liste de privilèges particuliers.

Exemples

Création d'un utilisateur avec tous les droits :

```
mysql> GRANT ALL PRIVILEGES ON *.* TO admin2@localhost
IDENTIFIED BY 'mdpAdmin' WITH GRANT OPTION;
```

Donner des droits de consultation sur certaines colonnes :

```
mysql> GRANT SELECT (ne, nom) ON empdept.emp TO toto@'%';
```

Consulter les privileges : SHOW GRANTS FOR

Syntaxe

```
SHOW GRANTS [FOR user];
```

Exemple

Consultation des droits d'un utilisateur: pour pouvoir voir les privilèges, il faut avoir l'accès à la BD mysql.

```
mysql> SHOW GRANTS FOR admin2@'localhost';
+-----------------------------------------------------------------+
| Grants for admin2@localhost |
+-----------------------------------------------------------------+
| GRANT ALL PRIVILEGES ON *.* TO 'admin2'@'localhost' IDENTIFIED BY
PASSWORD '*65D70AAC97E173AC4521B06CED3F332D9CB6E2E4' WITH
      GRANT
OPTION |
+-----------------------------------------------------------------+
1 row in set (0.00 sec)
```

Consultation de ses propres droits : tout utilisateur, quels que soient ses droits, peut les consulter

```
mysql> SHOW GRANTS;
```

Retirer des privileges : REVOKE … ON … FROM …

Syntaxe simplifiée

```
REVOKE privilege [,privilege] ON composant FROM user[@host]
```

Syntaxe complète n°1

```
REVOKE privilege  [(column_list)] [* , privilege  [(column_list)]]
   ON [composant] priv_level FROM user[@host] [*, user[@host]]
```

Syntaxe complète n°2

Pour supprimer tous les droits et le GRANT OPTION

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM user[@host] [*,
    user[@host]]
```

Exemples

- ***Suppression d'un droit***

```
mysql> REVOKE select (nom) on empdept.emp from toto@'%';
```

- ***Suppression de tous les droits sauf le grant option:***

```
mysql> REVOKE all on *.* from admin2@localhost;
```

- ***Suppression du droit de grant :***

```
mysql> REVOKE grant option on *.* from admin2@localhost;
```

- ***Suppression de tous les droits et du grant option :***

```
mysql> revoke all, grant option from admin2@localhost;
```

A noter qu'il n'y a plus de « on *.* »

Transitivité du grant option

Sous Oracle, si un utilisateur u1 donne des droits à un utilisateur u2 sur une BD1, que l'utilisateur u2 donne des droit à u3 sur BD1 et que finalement u1 retire les droits à u2 sur BD1, alors, automatiquement, u3 perd ses droits sur BD1. Ce n'est pas le cas avec MySQL.

Limiter les ressources des comptes

http://dev.mysql.com/doc/refman/5.0/fr/user-resources.html

Limiter les ressources globalement

Variable système :

- max_connections : vaut 100 par défaut.
- max_user_connections. Vaut 0 par défaut (il n'y a alors pas de limites).

Limiter les ressources d'un utilisateur

Il y a 4 possibilités de limitation des ressources au niveau des droits :

- Nombre de connexions
- Nombre de connexions par heure
- Nombre de requête par heure
- Nombre de modifications par heure

Ce sont des privilèges client de niveau user (globaux).

Ils mettent à jour les attributs max_user_connections, max_connections (par heure) max_questions et max_updates dans mysql.user.

Exemples

➢ ***Création des limites***

```
mysql> GRANT USAGE ON *.* TO 'toto'@'localhost'
WITH MAX_QUERIES_PER_HOUR 20
MAX_UPDATES_PER_HOUR 10
MAX_CONNECTIONS_PER_HOUR 5
MAX_USER_CONNECTIONS 2;
```

Les types de limite peuvent être appelés dans n'importe quel ordre.

Si la commande GRANT n'a pas de clause WITH, les limites valent alors 0, c'est à dire qu'il n'y a pas de limite.

➢ ***Modification de limites***

```
mysql> GRANT USAGE ON *.* TO 'toto'@'%'
WITH MAX_QUERIES_PER_HOUR 100;
```

➢ ***Suppression de limites***

Pour supprimer une limite existante, on lui donne la valeur 0.

```
mysql> GRANT USAGE ON *.* TO 'toto'@'%'
WITH MAX_QUERIES_PER_HOUR 0;
```

Le compteur d'utilisation des ressources se met en marche dès que la limite n'est pas nulle.

Remise à zéro des comptes

L'administrateur peut remettre les compteurs à zéro pour tous les comptes (ce n'est pas la même chose que d'annuler les limites !)

```
mysql> flush user_resources;
```

ou bien

```
mysql> flush privileges ;
```

ou bien

```
shell> mysqladmin reload ;
```

Pour remettre à zéro les compteurs d'un seul compte, il suffit de refaire un grant usage en redonnant la valeur des limites.

Effectivité des privilèges

Les privilèges de niveau utilisateur seront effectifs immédiatement pour toute nouvelle connexion mais ne sont pas effectifs pour les connexions en cours.

Les privilèges de niveau BD seront effectifs immédiatement pour toute nouvelle utilisation d'une BD (use database) mais ne sont pas effectifs pour la BD en cours d'utilisation.

Les privilèges de niveau tables et colonnes seront effectifs immédiatement pour toute nouvelle utilisation de la table et de la colonne.

Démarrer le serveur sans les privilèges

```
shell> bin/mysql_safe–-user=mysql –skip-grant-tables
shell> bin/mysql mysql
```

Ensuite, on peut ajouter les privilèges avec des GRANT

Redémarrer le serveur sans privilèges permet de se connecter root si on a perdu le mot de passe.

Organisation des privilèges

2 classes de privilèges : serveur et client

Les classes de privilèges concernent les utilisateurs des privilèges :

- **Les privilèges serveur** : concernent l'administrateur et les développeurs d'applications.
- **Les privilèges client** : concernent les applications.

4 niveaux de privilèges

Les niveaux de privilèges concernent l'objet de la BD qui porte le privilège.

On distingue 4 niveaux hiérarchiques d'intervention: utilisateur, BD, table et attribut.

- ***Niveau 1 : utilisateur ou global***

  Les droits globaux s'appliquent à un utilisateur.

  Ils concernent toutes les bases de données d'un serveur.

  Ces droits sont stockés dans la table **mysql.user**.

- ***Niveau 2 : BD***

  Les droits de niveau BD s'appliquent à toutes les tables d'une base de données.

  Ces droits sont stockés dans la table **mysql.db**

- ***Niveau 3 : table et procédure***

  Les droits de niveau table s'appliquent à toutes les colonnes et à tous les tuples d'une table.

  Les droits de niveau procédure s'appliquent aux procédures d'une BD.

  Ces droits sont stockés dans la table **mysql.tables_priv et mysql.procs_priv**.

- ***Niveau 4 : colonne***

  Les droits de niveau colonne s'appliquent à des colonnes dans une table.

  Ces droits sont stockés dans la table **mysql.columns_priv**.

Héritage des privilèges

Un privilège de niveau N est transmis au niveau N+1 (inférieurs) : un privilège de niveau « user » (1) existe au niveau « colonne » (4)

Liste des privilèges : SHOW PRIVILEGES

Nom	Classe	Niveau	Principe
ALL [PRIVILEGES]	Serveur	1 : User	Tous les privilèges, sauf WITH GRANT OPTION
CREATE USER	Serveur	1 : User	Autorise l'utilisation de CREATE USER
PROCESS	Serveur	1 : User	Autorise l'utilisation de SHOW FULL ROCESSLIST.
RELOAD	Serveur	1 : User	Autorise l'utilisation de FLUSH.
SHUTDOWN	Serveur	1 : User	Autorise l'utilisation de mysqladmin shutdown.
SUPER	Serveur	1 : User	Autorise une connexion unique même si max_connections est atteint, et l'exécution des commandes CHANGE MASTER, KILL thread, mysqladmin debug, PURGE MASTER LOGS et ET GLOBAL.
USAGE	Serveur	1 : User	Pour créer un utilisateur sans droits.
FILE	Client	1 : User	Autorise l'utilisation de SELECT ... INTO OUTFILE et LOAD DATA INFILE.
SHOW DATABASES	Client	1 : User	Autorise l'utilisation de SHOW DATABASES.

Nom	Classe	Niveau	Principe
CREATE ROUTINE	Serveur	2 : BD	Autorise l'utilisation de CREATE ROUTINE
LOCK TABLES	Serveur	2: BD	Autorise l'utilisation de LOCK TABLES sur les tables pour lesquelles l'utilisateur a les droits de SELECT.

Nom	Classe	Niveau	Principe
ALTER	Serveur	3 : Table	Modification (tous les ALTER possibles)
CREATE	Client	3 : Table	Autorise l'utilisation de CREATE.
CREATE VIEW	Client	3 : Table	Autorise l'utilisation de CREATE VIEW.
DELETE	Client	3 : Table	Autorise l'utilisation de DELETE.
DROP	Serveur	3 : Table	Destruction de bases ou de tables
INDEX	Serveur	3 : Table	Création ou suppression d'index sur une table
SHOW VIEW	Client	3 : Table	Autorise l'utilisation de SHOW CREATE VIEW
GRANT OPTION	Serveur	3 : Table et proc	Autorise l'utilisation de GRANT
Nom	**Classe**	**Niveau**	**Principe**
ALTER ROUTINE	Client	3 : Proc	Autorise l'utilisation de ALTER ROUTINE
EXECUTE	Client	3 : Proc	Autorise l'utilisateur à exécuter des procédures stockées (pour MySQL 5.0).
Nom	**Classe**	**Niveau**	**Principe**
INSERT	Client	4 : Col	Autorise l'utilisation de INSERT.
SELECT	Client	4 : Col	Autorise l'utilisation de SELECT.
UPDATE	Client	4 : Col	Autorise l'utilisation de UPDATE.

La BD mysql

http://dev.mysql.com/doc/refman/5.0/fr/information-schema.html

Présentéation

MySQL contient deux bases de données par défaut pour gérer les données du SGBD :

- **la BD MySQL** : première version du dictionnaire des données, principalement pour gérer les droits.
- **la BD information_shéma** : deuxième version du dictionnaire des données, plus standard

Schéma de la BD mysql

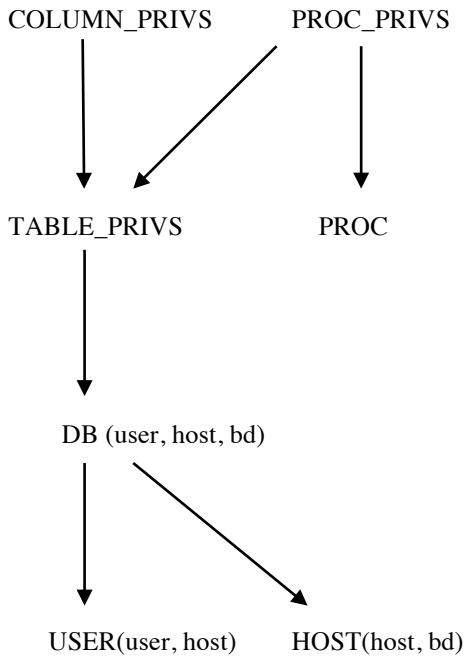
Liste des tables

On peut voir les tables de la BD mysql en faisant un show tables

```
mysql> show tables from mysql;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| user                      | 6 tables de droits
| host                      |
| db                        |
| tables_priv               |
| columns_priv              |
| procs_priv                |

| proc                      | 2 tables de définition des procédures
| func                      | et des fonctions

| help_category             | 4 tables pour les infos de
| help_keyword              | la commande help
| help_relation             |
| help_topic                |

| time_zone                 | 5 tables pour les fuseaux horaires
| time_zone_leap_second     | vides par défaut
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type
|
| event                     | 7 tables ajoutées depuis la
| general_log               | version 5.1
| ndb_binlog_index          |
| plugin                    |
| proxies_priv              |
| servers                   |
| slow_log                  |
+---------------------------+
24 rows in set (0.01 sec)
```

Graphe des tables de droits

La colonne vertébrale du graphe est : TABLE ⟶ DB ⟶USER



A noter que :

L'utilisateur est caractérisé par son nom d'utilisateur et son hôte. La BD appartient à un utilisateur. Elle fait donc référence à un utilisateur.

MySQL permet aussi de définir des droits pour une BD et pour un hôte donné, quel que soit l'utilisateur.

Précisions sur le contenu des tables de droit

Les 6 tables de droits de la BD mysql	
Nom de la table	**Usage**
User	**Quid :** Droits d'un utilisateur pour une machine donnée. Droits généraux de l'utilisateur sur toutes les tables. root a tous les droits à ce niveau. En général, un utilisateur lambda n'a aucun droit à ce niveau. Eventuellement, un utilisateur lambda peut avoir des droits de consultation sur toutes les tables (select). **CP** : host, user host **:** nom (ou adresse IP) de la machine depuis laquelle un utilisateur se connecte **Attributs :** Password : le mot de passe Droits généraux de l'utilisateur (booléen) : select, update, etc. Limite des ressources de l'utilisateur : maximum. **Exemple** : grant LOCK_TABLES on *.* to toto@localhost Avec le *.*, toutes les BD sont concernées : on est au niveau global
Host	**Quid :** Droits d'une machine pour une BD donnée **CP** : host, db **Attributs :** droits généraux de l'hôte (booléen) : select, update, etc. **Exemple**
Db	**Quid :** Droits des utilisateurs chez un hôte pour une BD. **CP** : # (host, user), db **Attributs :** droits de l'utilisateur sur la BD (booléen) : select, update, etc. **Exemple** : grant all on empdept.* to toto@localhost Avec le * derrière le nom de bd (empdept), seule la bd est concernée.
Tables_priv	**Quid :** Droits des utilisateurs chez un hôte pour une table d'une BD.. **CP** : # (host, user, db), table_name **Attributs :** table_name : la table à laquelle les privilèges sont accordés grantor : celui qui a donné le droit, sous la forme : user@host table_priv : la liste des privilèges (select, insert, update, etc.) **Exemple** : grant all on empdept.emp to toto@localhost
Columns_priv	**Quid :** droits des utilisateurs chez un hôte pour un attribut d'une table d'une BD. **CP** : # (host, user, db, table_name), column_name **Attributs :** column_name : la colonne à laquelle les privilèges sont accordés grantor : celui qui a donné le droit, sous la forme : user@host column_priv : la liste des privilèges : select, insert, update et référence. **Exemple** : grant all on empdept.emp.ne to toto@localhost
Proc_priv	**Quid :** Droits des utilisateurs chez un hôte pour une procédure d'une BD.. **CP** : # (host, user, db), routine_name **Attributs :** routine_name : la routine à laquelle les privilèges sont accordés grantor : celui qui a donné le droit, sous la forme : user@host proc_priv : la liste des privilèges (alter, execute, grant) **Exemple** :

Description des attributs : Organisation des privilèges dans les tables de mysql

User	Db	Procs_priv	Tables_priv		Columns_priv
Host	**Host**	**Host**	**Host**		**Host**
User	**User**	**User**	**User**		**User**
	Db	**Db**	**Db**		**Db**
Password		**Routine_name**	**Table_name**		**Table_name**
max_questions		Routine_type			**Column_name**
max_updates		Grantor	Grantor		
max_connections					
max_user_connections					
		Proc_priv	Table_priv	Column_priv	Column_priv
Alter_priv	Alter_priv		'Alter'		
Alter_routine_priv	Alter_routine_priv	'Alter Routine',			
Create_priv	Create_priv		'Create'		
Create_routine_priv	Create_routine_priv				
Create_tmp_table_priv	Create_tmp_table_priv				
Create_user_priv					
Create_view_priv	Create_view_priv		'Create View'		
Delete_priv	Delete_priv		'Delete'		
Drop_priv	Drop_priv		'Drop'		
Execute_priv	Execute_priv	'Execute',			
File_priv					
Grant_priv	Grant_priv	'Grant'	'Grant'		
Index_priv	Index_priv		'Index'		
Insert_priv	Insert_priv		'Insert'	'Insert',	'Insert',
Lock_tables_priv	Lock_tables_priv				
Process_priv					
References_priv	References_priv		'References'	'References'	'References'
Reload_priv					
Repl_client_priv					
Repl_slave_priv					
Select_priv	Select_priv		'Select'	'Select',	'Select',
Show_db_priv					
Show_view_priv	Show_view_priv		'Show view'		
Shutdown_priv					
Super_priv					
Update_priv	Update_priv		'Update'	'Update',	'Update',

Manipulation des tables de la BD mysql

Les tables de la BD mysql sont modifiées par :

- **Les ordres DCL** (data control language) : CREATE USER, RENAME USER, DROP USER, SET PASSWORD, GRANT, REVOKE.
- **Des ordres DML** (data manipulation language) : INSERT, UPDATE, DELETE travaillant **directement dans les tables de la BD mysql.** C'est utile pour faire des modifications de masse et résoudre certains problèmes particuliers non gérés par le DCL.

Travailler directement sur les tables de la BD mysql n'est pas recommandé.

C'est parfois utile pour faire des modifications particulières.

C'est nécessaire pour la table host qui n'est pas modifiable avec les ordres du DCL.

Exemples

```
Shell> mysql –uroot -pmotDePasse
Mysql> Create user toto ;
```

La commande ajoute un utilisateur dans la table user.

Cet utilisateur peut ouvrir une calculette mysql :

```
Shell> mysql –utoto
mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
+--------------------+
1 row in set (0.00 sec)
```

Il n'a accès qu'à une database : information_schéma qui est le dictionnaire des données.

Il n'a accès qu'aux tables de informations schéma. Mais toutes les tables sont vides car elles décrivent les BD et les tables auxquelles l'utilisateur a accès.

Il n'a aucun droits. Il ne peut pas créer de database.

```
mysql> create database dbtoto;
ERROR 1044 (42000): Access denied for user 'toto'@'%' to database
'dbtoto'
```

Root doit créer des droits à l'utilisateur toto:

```
Shell> mysql –uroot -pmotDePasse
Mysql> Grant all on bdtoto.* to toto;
```

Ca permet à l'utisateur toto de faire tout ce qu'il veut sur la BD bdtoto.

La bdtoto est visible par toto, si elle existe. Si elle n'existe pas, toto peut la créer :

```
Shell> Mysql –utoto
Mysql> Create database bdtoto
```

La database bdtoto est créée :

```
mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
| bdtoto |
+--------------------+
1 row in set (0.00 sec)
```

Root peut donner des droits à sur une BD existant déjà :

```
Shell> mysql –uroot -pmotDePasse
Mysql> Grant select on biblio.* to toto;
```

Root peut donner des droits à une table existant déjà :

```
Shell> mysql –uroot -pmotDePasse
Mysql> Grant select on empdept.emp to toto;
```

Root peut donner des droits à certains attributs d'une table existant déjà :

```
Shell> mysql –uroot -pmotDePasse
Mysql> Grant select (NET, nom) on ecoling.etudiants to toto;
```

Droits d'une BD pour un hote : table mysql.host

La table **mysql.host** permet d'associer les privilèges d'une BD à un hôte.

C'est une spécificité MySQL.

Elle n'est pas mise à jour avec les instructions du DCL. Elle ne peut être mise à jour que directement, par des INSERT, UPDATE, DELETE.

Ainsi, root peut donner des droits sur une BD pour un hôte donné. Cette modification n'est pas accessible par un GRANT. On peut l'effectuer directement dans la table host :

```
mysql> insert into host (host, db, select_priv, insert_priv) values
('%','biblio', 'Y', 'Y');
Mysql> flush privileges;
```

Le "flush privileges" permet que la modification de la table des droits soit bien prise en compte.

Actualisation de la BD mysql : flush privileges

Au démarrage, le serveur charge en mémoire vive l'ensemble de la BD mysql.

Les modifications effectuées par les ordres du DCL sont répercutées aussi en mémoire vive.

Les modifications effectuées directement dans la BD mysql par des ordre du DML ne sont pas répercutées en mémoire vive.

Pour actualiser la mémoire vive :

```
mysql> flush privileges
```

Initialisation et réinitialisation de la BD mysql : mysql_install_db

Présentation

C'est le programme mysql_install_db qui initialise les tables de droits (la BD mysql). En général, le programme d'installation se charge de lancer mysql_install_db.

Normalement, mysql_install_db est exécuté uniquement la première fois qu'on installe MySQL. Le script de mysql_install_db crée :

- le répertoire des données
- la BD mysql qui contient tous les privilèges de la BD (avec les tables user, db, host, tables_priv, column_priv, func et éventuellement d'autres).
- la BD test pour tester mysql
- les entrées de la table de privilèges pour les comptes root et des comptes d'utilisateurs anonymes. **ATTENTION** : ces comptes n'ont pas de mot de passe !!! Le bilan est que l'utilisateur root peut faire ce qu'il veut et que les autres utilisateurs peuvent faire ce qu'ils veulent avec la BD test ou des BD commençant par test_

Recréer les tables de droits

Pour recréer les tables de droits, il faut commencer par supprimer tous les fichiers .frm, .MYI et .MYD du répertoire mysql de la BD mysql.

Ensuite on réexécute le fichier mysql_install_db :

```
shell > cd /usr/bin
shell > mysql_install_db --user=mysql
```

ou

```
shell > cd /usr/scripts
shell > mysql_install_db --user=mysql
```

Recharger les tables de droits

A tout moment, on peut recharger la table des droits pour que les dernières modifications prennent effet :

```
shell> mysqladmin flush-privileges
```

```
shell> mysqladmin reload
```

```
mysql> flush privileges
```

Création de rôles sous MySQL

Un rôle est un profil type d'utilisateur qu'on peut ensuite affecter à un utilisateur : par exemple : lecteur, opérateur de saisie, administrateur du SI, etc.

MySQL ne permet pas de créer de rôle.

Toutefois, on peut contourner cette impossibilité en créant des utilisateurs servant de modèle.

Ces utilisateurs seront créés sur « localhost » avec un mot de passe arbitraire.

Ces utilisateurs seront définis avec les autorisations souhaitées.

Pour créer un utilisateur en reprenant la définition d'un utilisateur modèle, il suffit de :

- Faire la création de l'utilisateur et de son hôte avec son mot de passe, sans droit.
- Recopier les droits du modèle dans ceux du nouvel utilisateur en copiant les valeurs des tables concernées dans la BD mysql : user, db, tables_priv et colums_priv.
- Les tables host et procs_priv ne sont pas concernées par les rôles.

Ces manipulations peuvent être intégrées dans une procédure stockées.

Le dictionnaire des données : la BD Information schema

http://dev.mysql.com/doc/refman/5.0/fr/information-schema.html

MySQL contient deux bases de données par défaut pour gérer les données du SGBD :

- **la BD MySQL** : première version du dictionnaire des données, principalement pour gérer les droits.
- **la BD information_shéma** : dictionnaire des données plus standard

Présentation

Avec la version 5.0 apparaît un dictionnaire des données : c'est la BD information_schéma.

Le dictionnaire des données contient des méta-données : des données sur les données.

Le dictionnaire des données contient toutes les informations sur les bases de données gérées par le serveur.

Cette BD est visible quand on fait un show « databases ».

Schéma du dictionnaire des données

Liste des tables

On peut voir les tables de la BD information_schema en faisant un show tables

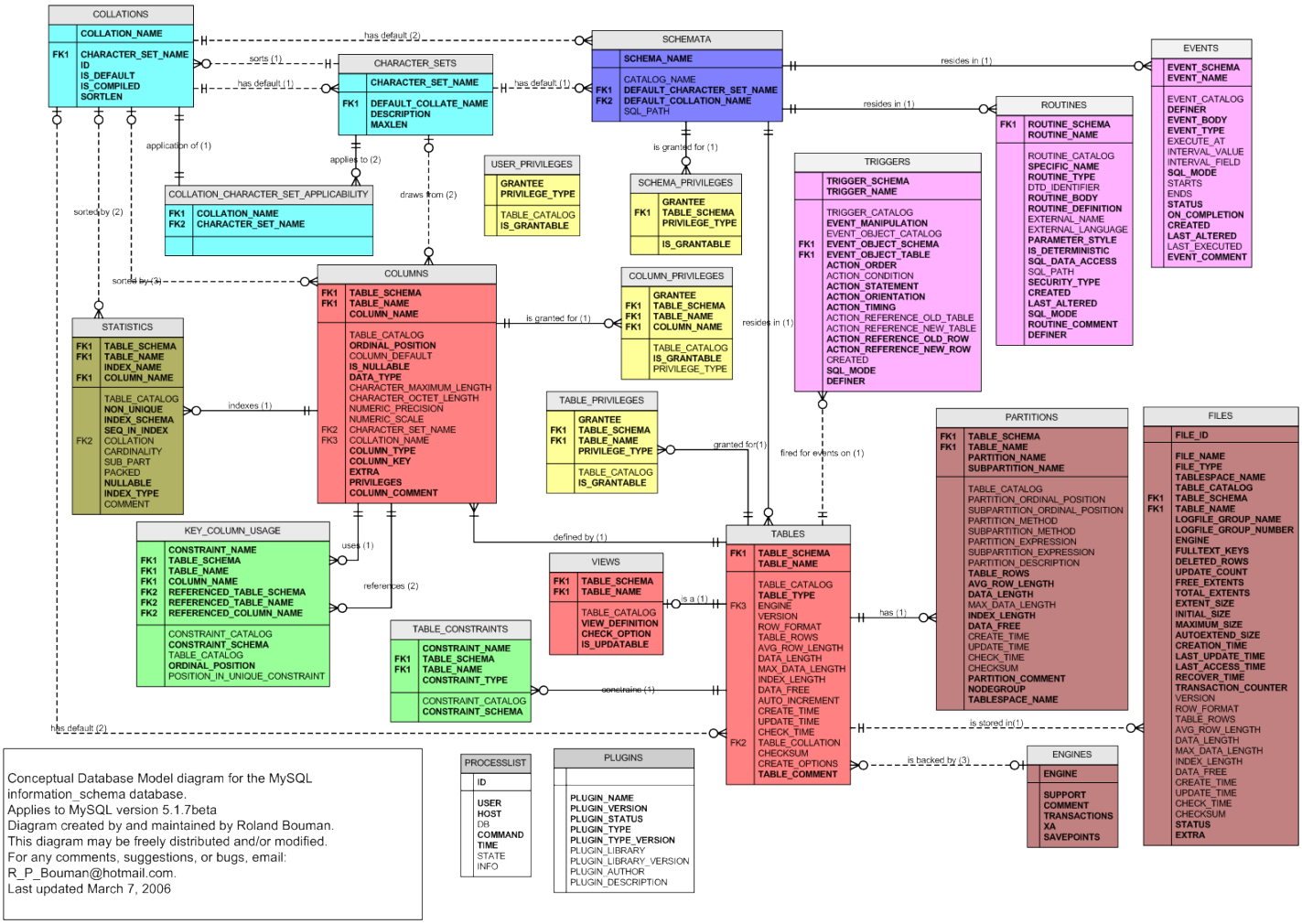
```
mysql> use information_schema
Database changed
mysql> show tables;
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| CHARACTER_SETS                        |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMN_PRIVILEGES                     |
| ENGINES                               |
| EVENTS                                |
| FILES                                 |
| GLOBAL_STATUS                         |
| GLOBAL_VARIABLES                      |
| KEY_COLUMN_USAGE                      |
| PARAMETERS                            |
| PARTITIONS                            |
| PLUGINS                               |
| PROCESSLIST                           |
| PROFILING                             |
| REFERENTIAL_CONSTRAINTS               |
| ROUTINES                              |
| SCHEMATA                              |
| SCHEMA_PRIVILEGES                     |
```

```
| SESSION_STATUS                        |
| SESSION_VARIABLES                     |
| STATISTICS                            |
| TABLES                                |
| TABLESPACES                           |
| TABLE_CONSTRAINTS                     |
| TABLE_PRIVILEGES                      |
| TRIGGERS                              |
| USER_PRIVILEGES                       |
| VIEWS                                 |
| INNODB_CMP_RESET                      |
| INNODB_TRX                            |
| INNODB_CMPMEM_RESET                   |
| INNODB_LOCK_WAITS                     |
| INNODB_CMPMEM                         |
| INNODB_CMP                            |
| INNODB_LOCKS                          |
+---------------------------------------+
37 rows in set (0.00 sec)
```

MEA (ou equivalent)

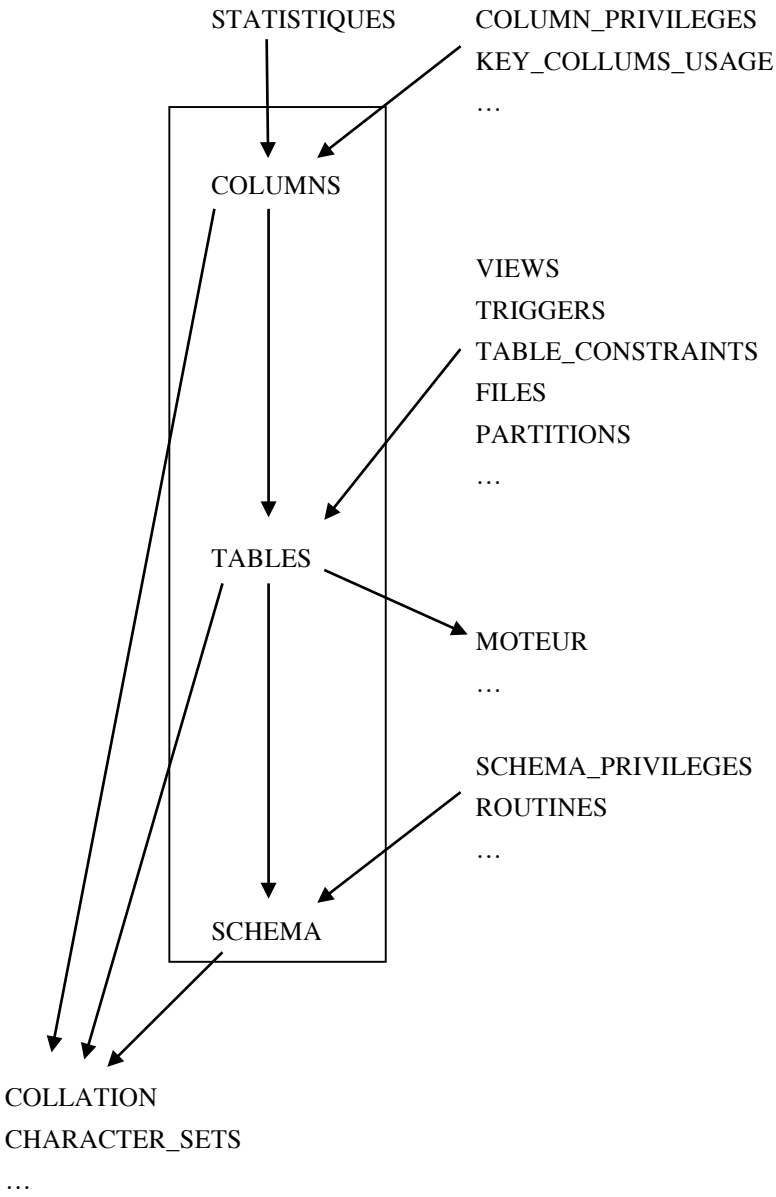
Le schéma ci-dessous, qu'on trouve sur internet, présente un MEA de la BD information_schema
Version MySQL : 5.1.
Formalisme utilisé : I I : 1-1 // O I : 0-1 // O < : 0-N // I < : 1-N

Remaque : le schéma n'étant pas hiérarchisé, il est peu lisible.



Première approche du graphe des tables

La colonne vertébrale du graphe est : colonne ⟶ table ⟶ schéma



Thématique des tables

TABLES de la BD INFORMATION_SCHEMA	
Nom de la table	**Usage**
Schemata, tables, columns, views, triggers, routines	Ces 6 tables décrivent la notion dont elles portent le nom. La table de toutes les BD (schemata), de toutes les tables, de tous les attributs (columns), de toutes les vues, de tous les triggers, de toutes les procédures stockées. Schemata (**schema_name**, ...) Tables (**#schema_name, table_name** ...) Columns (**#(schema_name, table_name), columns_name** ...) Views (**#schema_name, table_name** ...) : table_name c'est view_name Triggers (**#trigger_schema, trigger_name** ...)
User_privilege, Schema_privileges, table_privileges, column_privilege	Ces 4 tables décrivent les privilèges par utilisateur, BD, table et attribut.
Key_column_usage	Table des clés primaires et étrangères listées par attribut. **(#(table_schema, table_name, column_name)**, **constraint_name**,...)
Table_constraints	Table contraintes d'intégrité listées par table. (**#(table_schema, table_name), constraint_name**, ...)
Statistic	Table des statistiques pour l'optimisation
Character_sets, Collations, Collation_character_set_applicability	Ces 3 tables décrivent des éléments graphiques

Description des attributs des tables de description des BD

```
mysql> use information_schema
```

Liste des BD du serveur

```
mysql> select schema_name from information_schema.schemata ;
```

est équivalent à :

```
mysql> show databases ;
```

Liste des tables du serveur

```
mysql> select table_names from tables ;
```

Les tables du serveur, ce sont toutes les tables des différentes BD et aussi les tables spécifiques du dictionnaire.

Liste des tables d'une BD particulière

```
mysql> use maBD
mysql> show tables ;
```

est équivalent à :

```
mysql>  select table_name from information_schema.tables
        where table_schema ='maBD';
```

Description des attributs des tables de privilèges

La table user_privilèges

```
mysql> desc user_privileges;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| GRANTEE        | varchar(81)  | NO   |     |         |       |
| TABLE_CATALOG  | varchar(512) | YES  |     | NULL    |       |
| PRIVILEGE_TYPE | varchar(64)  | NO   |     |         |       |
| IS_GRANTABLE   | varchar(3)   | NO   |     |         |       |
+----------------+--------------+------+-----+---------+-------+
4 rows in set (0.02 sec)
```

- Grantee : nom de l'utilisateur avec son hôte.
- Table_catalog : inutile pour le moment.
- Privilege_type : nom du privilège. Par défaut la table user_privileges contient tous les privilèges de root: soit 25 lignes
- Is_grantable : précise si le privilège est transférable ou pas.

Quand on crée un utilisateur avec un GRANT, on crée autant de lignes qu'on lui donne de privilèges. 25 lignes pour ALL PRIVILEGES.

Quand on crée un utilisateur avec un CREATE USER, on crée une seule ligne avec le droit : USAGE.

```
mysql> create user toto;
Query OK, 0 rows affected (0.00 sec)
mysql> select * from user_privileges\G
*************************** 26. row ***************************
GRANTEE: 'toto'@'%'
TABLE_CATALOG: NULL
PRIVILEGE_TYPE: USAGE
IS_GRANTABLE: NO
26 rows in set (0.00 sec)
```

Equivalence entre mysql et information_schema

```
Select * from information_schema.user_privileges
where grantee like "%toto%";
```

Est equivalent à

```
Select * from mysql.user where user='toto';
```

Les autres tables de privilèges

Les autres tables de privileges fonctionnent selon le même principes que les tables de privilèges de la BD mysql : cf. § Les 6 tables de droits de la BD mysql.

```
mysql> desc schema_privileges;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| GRANTEE        | varchar(81) | NO   |     |         |       |
| TABLE_CATALOG  | varchar(512)| YES  |     | NULL    |       |
| TABLE_SCHEMA   | varchar(64) | NO   |     |         |       |
| PRIVILEGE_TYPE | varchar(64) | NO   |     |         |       |
| IS_GRANTABLE   | varchar(3)  | NO   |     |         |       |
+----------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
mysql> desc table_privileges;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| GRANTEE        | varchar(81) | NO   |     |         |       |
| TABLE_CATALOG  | varchar(512)| YES  |     | NULL    |       |
| TABLE_SCHEMA   | varchar(64) | NO   |     |         |       |
| TABLE_NAME     | varchar(64) | NO   |     |         |       |
| PRIVILEGE_TYPE | varchar(64) | NO   |     |         |       |
| IS_GRANTABLE   | varchar(3)  | NO   |     |         |       |
+----------------+-------------+------+-----+---------+-------+
6 rows in set (0.02 sec)
mysql> desc column_privileges;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| GRANTEE        | varchar(81) | NO   |     |         |       |
| TABLE_CATALOG  | varchar(512)| YES  |     | NULL    |       |
| TABLE_SCHEMA   | varchar(64) | NO   |     |         |       |
| TABLE_NAME     | varchar(64) | NO   |     |         |       |
| COLUMN_NAME    | varchar(64) | NO   |     |         |       |
| PRIVILEGE_TYPE | varchar(64) | NO   |     |         |       |
| IS_GRANTABLE   | varchar(3)  | NO   |     |         |       |
+----------------+-------------+------+-----+---------+-------+
7 rows in set (0.01 sec)
```

Ré-écriture de DESC avec le dictionnaire des attributs

```
use mysql;

drop procedure if exists desc2;
delimiter //
Create procedure desc2(v_nomTable varchar(64), v_nomSchema
varchar(64))
BEGIN
SELECT
    c.column_name as Nom,
    c.column_type as Type,
    c.is_nullable as 'NULL',
    CASE WHEN k.column_name=c.column_name THEN 'Oui'
        ELSE 'Non'
    END as 'Key',
    c.column_default as Extra
FROM information_schema.columns c
    LEFT OUTER JOIN information_schema.key_column_usage k
    ON (k.column_name=c.column_name
    AND k.table_name= c.table_name
    AND k.table_schema= c.table_schema
    AND k.constraint_name= 'PRIMARY')
WHERE c.table_name=v_nomTable
AND c.table_schema=v_nomSchema;
END;
//
delimiter ;

call mysql.desc2('oeuvres', 'biblio');
```