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

https://dev.mysql.com/doc/refman/5.5/en/
https://dev.mysql.com/doc/refman/5.6/en/
https://dev.mysql.com/doc/refman/5.7/en/
https://dev.mysql.com/doc/refman/8.0/en/

Bertrand LIAUDET

SOMMAIRE

Les paragraphes en jaune sont les paragraphes de synthèse. Les autres apportent des précisions.

Dernière édition : février 2020

GESTION DES UTILISATEURS
ET DE LEURS PRIVILEGES (DROITS)

En bleu : le résumé : 29 slides

Documentation

Versions

Il y a, en 2019, 4 versions accessibles de MySQL : 5.5, 5.6, 5.7 et 8 .0

Documentation sur l'administration du serveur

https://dev.mysql.com/doc/refman/5.5/en/server-administration.html

https://dev.mysql.com/doc/refman/8.0/en/server-administration.html

Ici on trouve la documentation sur la BD système mysql

Documentation sur l'administration des BD

https://dev.mysql.com/doc/refman/5.5/en/sql-syntax-server-administration.html

https://dev.mysql.com/doc/refman/8.0/en/sql-syntax-server-administration.html

Ici on trouve la documentation sur la création des utilisateurs, l'optimisation des tables, la commande SHOW et d'autres éléments.

Principes de recherches

Googler

Googler un mot clé, mysql, la version et aller sur la page de la documentation officielle trouvée.

Par exemple : set password mysql 8

https://dev.mysql.com/doc/refman/5.6/en/server-administration.html

Recherche directe dans le manuel

Zone de recherche : Search / Search this Manual

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.

Règles importantes

- **Protéger l'hôte (la machine) 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.

Autres règles

Pour connaître l'hôte serveur :

```
shell> hostname
```

Pour vérifier la situation du port MySQL :

```
shell> telnet hôte_serveur 3306
```

Le telnet doit dire que la connexion est interdite : le port est bloqué.

Ou alors le telnet doit signaler, salement, que la connexion est utilisée par MySQL.

Ou alors, 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.

Sur MAC : installer brew. Faire un brew install telnet. Ensuite, utiliser telnet.

Sur PC : C :>appwiz.cpl + activer/désactiver fonctionnalités windows + cocher client telnet

- **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 SE (sur la machine du serveur). 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.

Remarque pour administrateur

La commande tcpdump sous linux permet d'analyser les flux de données qui circulent sur un port.

https://openmaniak.com/fr/tcpdump.php

```
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.

Règles importantes

- **Tous les utilisateurs (clients, applications) doivent avoir un mot de passe** (ça c'est pour un serveur en production : pour un serveur local sur votre machine en phase de développement, ce n'est pas utile).
- **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 ! (ça c'est pour un serveur en production : pour un serveur local sur votre machine en phase de développement, ce n'est pas utile).
- **Eviter de créer plusieurs comptes avec des droits d'administrateur.**

Autres règles

- **Ne stocker aucun mot de passe en clair.** Utilisez toujours une fonction de cryptage ou de hachage à sens unique : password().
- **En version 5.5 et 5.6, la table user de la database mysql ne doit être accessible que par root !** La table user contient les mots de passe (attribut password). 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 ! Dans les version 5.7 et 8.0, l'attribut password n'est plus stocké dans la table user.
- **Choisir des mots de passe assez compliqués et éviter de les perdre !** Trouver des systèmes mnémotechniques mais compliqué.
- **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
  - ✓ Utiliser des fonctions du langage serveur qui vérifient les saisies : htmlspecialchars() ou htmlentities() en PHP, par exemple

- **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. Dans les dernières version de mysql, le skip-networking est associée par défaut.

--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 ! Comme ça, on est sur que personne ne pourra se connecter à part les utilsateurs sur la machine hôte du serveur.

--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 une limitation des droits sur le fichier my.cnf :***

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

600 (2+4, 0, 0) permet au propriétaire de lire et d'écrire, et rend le fichier inaccessible aux autres.

Ces deux techniques sont correctement sécurisées à condition, dans la deuxième configuration, de ne pas laisser un accès à son poste de travail quand on est identifié !

Conclusion

La première source d'intrusion est l'erreur humaine : évitez de ne pas mettre de password, de laisser traîner votre password, de donner votre password, de laisser votre machine accessible !

Gestion des utilisateurs et de leurs mots de passe

Résumé

Documentation

Googler un mot clé, mysql, la version et aller sur la page de la documentation officielle trouvée.
Par exemple : set password mysql 8

Gestion des utilisateurs : le DCL : data control language

- Create user
- Drop user
- Set password
- Grant
- Revoke

Connexion d'un utilisateur au serveur

- ***Si le serveur est local :***

```
shell> mysql –u nomUtilisateur -p
```

- ***Si le serveur est distant :***

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

A noter dans ce cas, qu'il faut en général mettre à jour le fichier de configuration (my.ini ou my.cnf) : il faut mettre en commentaire le bind-address en mettant un # devant.

La situation est à analyser en fonction du contexte précis du serveur et du client.

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

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

Connaître la version de MySQL : select version()

```
mysql> select version();
```

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**

Connaître l'identité effective : select user()

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

Le résultat peut être : moi@localhost ou moi@maMachine. C'est le nom de l'utilisateur et son hôte (sa machine)

Connaître le profil de connexion : select current_user()

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

Le current_user est un profil d'utilisateur et pas l'utilisateur avec son nom et sa machine.

On peut par exemple avoir un profil de connexion correspondant à :

- un nom d'utilisateur pouvant se connecter de n'importe quelle machine : **moi@'%'**
- un nom d'utilisateur pouvant se connecter de la machine du serveur : **moi@localhost**
- un nom d'utilisateur pouvant se connecter de sa machine uniquement : **moi@maMachine**
- n'importe quel utilisateur se connectant à partir d'une machine particulière : '**%**'**@cetteMachine**
- un nom d'utilisateur pouvant se connecter de sa machine uniquement : **moi@maMachine**

Consultation de la liste des utilisateurs

- ***Version à partir de 5.7 du serveur :***

Il n'y a plus d'attribut password dans table mysql.user.

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

On trouve l'utilisateur root qui peut se connecteur à partir de la machine hôte du serveur.

Les 3 autres utilisateurs sont des utilisateurs techniques qu'il ne faut pas modifier.

Les intitulés sys et session reprennent la terminologie ORACLE.

- ***Version jusqu'à 5.6 du serveur :***

On trouve un attribut password dans table mysql.user

```
mysql> select host, user, password from mysql.user;
+-----------+------+----------+
| host      | user | password |
+-----------+------+----------+
| localhost | root |          |
| 127.0.0.1 | root |          |
| ::1       | root |          |
| localhost |      |          |
+-----------+------+----------+
```

On trouve 4 utilisateurs par défaut qui peuvent se connecter sur la machine du serveur.

La machine serveur correspond à localhost, 127.0.0.1 ou ::1

Création d'un utilisateur

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

IDENTIFIED BY en majuscules.

Quand on crée un utilisateur, il ne peut pas créer de base de données et il n'accède qu'à la BD information_schéma.

En version <= 5.6, il peut accéder à la BD « test » et créer des tables à l'intérieur.

Il faudra donner des droits à l'utilisateur.

Pour créer un utilisateur pouvant se connecter à partir de n'importe quelle machine, on écrit :

```
CREATE USER nomUtilisateur@'%'
```

Renommer un utilisateur

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

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

Suppression d'un utilisateur

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

Modification et suppression du password

- ***Versions 5.7 et 8***

```
ALTER USER 'root'@'localhost' IDENTIFIED BY ‘newPassword’';
```

Pour supprimer le PASSWORD, on lui donne une valeur à ‘’ ;

- ***Versions 5.5 et 5.6***

```
SET PASSWORD [FOR nomUser ] = PASSWORD(‘motDePasse’)
```

La commande SET PASSWORD s’applique par défaut à l’utilisateur courant. Pas besoin de le préciser.

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

```
SET PASSWORD [FOR nomUser ] = ‘’ ;
```

Pour supprimer le PASSWORD, on lui donne une valeur à ‘’ ;

Obliger un utilisateur à changer de PASSWORD (depuis 5.6) :

```
ALTER USER ,toto’@'localhost' PASSWORD EXPIRE;
```

Consulter les droits d'un utilisateur :

- ***Pour n'importe quel utilisateur :***

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

- ***Pour soi-même :***

```
mysql> SHOW GRANTS;
```

- ***Droit d'USAGE***

Quand on crée un utilisateur, il n'a qu'un seul droit : le droit d'usage.

Ca lui permet de se connecter mais il ne peut rien faire d'autre !

```
mysql> show grants
+-----------------------------------------+
| Grants for nouvelUtilisateur@localhost  |
+-----------------------------------------+
| GRANT USAGE ON *.* TO ` nouvelUtilisateur `@`localhost` |
+-----------------------------------------+
```

En cas d'oubli du password de root ! Démarrer le serveur sans les privilèges : skip-grant-tables

Pour redémarrer le serveur sans privilèges, on utilise l'option –skip-grant-table au démarrage.

Toutefois, ça peut s'avérer plus compliqué.

Il faut regarder la documentation : **googler skip-grant-tables mysql 8** ou mysql 5 pour arriver sur la documentation officielle mysql. **Googler set password myql 8** de même ou mysql 5.

Regardez aussi bien les messages d'erreur si le serveur ne démarre pas.

- ***Oubli du password de root : exemple version 8***

Démarrage du serveur :

```
C:\mysql\mysql-8.0.13-winx64\bin\mysqld  --initialize-insecure
C:\mysql\mysql-8.0.13-winx64\bin\mysqld --console -- skip-grant-tables
                --explicit_defaults_for_timestamp  --shared-memory
```

Modification du password

```
mysql> FLUSH PRIVILEGES;
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass';
```

- ***Oubli du password de root : exemple version 5.5***

Démarrage du serveur :

```
C:\mysql\mysql-5.6.42-win32\bin\mysqld --console --skip-grant-tables
```

Modification du password

```
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('');
```

Précisions - 1 : Présentation détaillée

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;
```

Précisions - 2 : 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.

Précisions - 3 : 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 !!!

Précisions - 4 : Création d'un utilisateur – 3 : mysql .user (version mysql <= 5.6)

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).

Précisions - 5 : 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 commande ALTER USER (depuis mysql 5.6)

- ***Obliger un utilisateur à changer de PASSWORD (depuis 5.6) :***

```
ALTER USER ,toto'@'localhost' PASSWORD EXPIRE;
```

- ***Donner un mot de passe : (depuis 5.7)***

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'newPassword'';
```

Par la modification directe de la table des USER (version 5.5 et 5.6)

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).

Précisions - 6 : 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).

Précisions - 7 : Modification des mots de passe, versions 5.5 et 5.6

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).

Précisions - 8 : Suppression des mots de passe, versions 5.5 et 5.6

Par la commande SET PASSWORD FOR

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

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).

Précisions - 9 : Gestion MySQL des mots de passe, versions 5.5 et 5.6

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

- ***Privilèges ou droits***

  La notion de privilège est identique à celle de droit.

- ***Selon l'hôte***

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

- ***Consulter les privilèges d'un utilisateur :***

  Pour n'importe quel utilisateur :

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

  Pour soi-même :

  ```
  mysql> SHOW GRANTS;
  ```

- ***Liste des privilèges existants***

  ```
  mysql> 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 le droit à l'utilisateur 1 de consulter la table emp :***

```
mysql> GRANT SELECT ON emp TO user1;
```

- ***Consultation et insertion pour user1 et user 2***

```
mysql> GRANT SELECT, INSERT ON emp TO user1, user2 ;
```

- ***Limiter les droits à certains attributs***

- ***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@'%';
```

ou

```
mysql> REVOKE insert ON emp TO user1;
```

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

```
mysql> REVOKE all ON *.* FROM admin2@localhost;
```

ou

```
mysql> REVOKE all privileges ON *.* FROM admin2@localhost identified by
    'mdpAdmin';
```

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

```
mysql> REVOKE grant option ON *.* FROM admin2@localhost identified by
    'mdpAdmin';
```

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;
```

Précisions – 1 : PT 1

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.

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.

Précisions – 2 : 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@'%';
```

Précisions – 3 : 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;
```

Précisions - 4 : 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.

Précisions - 5 : 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.

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 : user, BD, table, attribut

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 (= 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**.

Il s'agit essentiellement de DELETE, CREATE, ALTER, DROP

Niveau 4 : colonne

Les droits de niveau colonne s'appliquent à certaines colonnes et certains tuples d'une table.

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

Il s'agit de INSERT, UPDATE et SELECT.

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.

Les principaux privilèges sont les suivants :

Nom	Classe	Niveau	Droit
USAGE	Serveur	1 : User	Pour créer un utilisateur sans droits.
ALL [PRIVILEGES]	Serveur	1 : User	Tous les privilèges, sauf WITH GRANT OPTION
WITH GRANT OPTION	Serveur	1 : User	Autorise l'utilisation de GRANT

Nom	Classe	Niveau	Droit
CREATE	Serveur	2 : BD, Table	Création (toutes les créations possible)
DROP	Serveur	2 : BD, Table	Destruction de bases ou de tables
ALTER	Serveur	3 : Table	Autorise l'utilisation de ALTER.

Nom	Classe	Niveau	Droit
SHOW DATABASES	Client	1 : User	Autorise l'utilisation de SHOW DATABASES.
SELECT	Client	Tuples	Autorise l'utilisation de SELECT.
INSERT	Client	Tuples	Autorise l'utilisation de INSERT.
UPDATE	Client	Tuples	Autorise l'utilisation de UPDATE.
DELETE	Client	Tuples	Autorise l'utilisation de DELETE.

Création de rôles sous MySQL

Définition d'un rôle

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.

Solution MySQL : création d'utilisateur servant de modèle

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.

Application du modèle à un nouvel utilisateur

1) Créer un utilisateur avec son hôte et son mot de passe, sans droit.
2) Faire un SHOW GRANTS sur l'utilisateur servant de modèle
3) Appliquer les GRANT affichés à l'utilisateur qu'on a créé.

Cette séquence pourrait être écrite dans une procédure stockée.

Les bases de données installées par défaut

Bases de données préinstallées

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

Présentation

MySQL contient **4 bases de données** par défaut pour gérer les données du SGBD (on dit aussi « schema à la place de BD ».

Les **BD mysql** et la **BD information schéma** représentent **le dictionnaire des données,** c'est à dire des informations sur les bases de données et les utilisateurs.

La BD mysql est une première version du dictionnaire des données. On y trouve la table user.
https://dev.mysql.com/doc/refman/8.0/en/system-schema.html

La BD information_schema est le dictionnaire des données « moderne ».
https://dev.mysql.com/doc/refman/8.0/en/innodb-information-schema-transactions.html

Les BD performance_schema et sys fournissent des **informations de performance.**
https://dev.mysql.com/doc/refman/8.0/en/performance-schema.html
https://dev.mysql.com/doc/refman/8.0/en/sys-schema.html

Le dictionnaire des données 1ère version : La BD mysql

La BD mysql (version 5.5)

Présentation

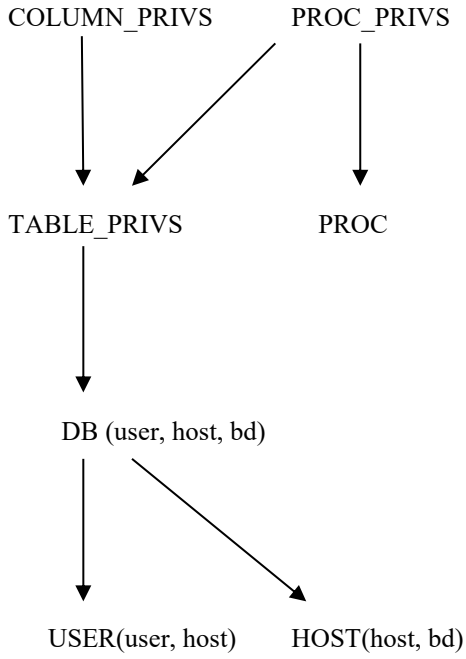
C'est une base de donnée qui sert surtout à gérer :

- **les utilisateurs et les droits.**
- **la définition des procédures et des fonctions.**

On y trouve une table « user » avec les attributs « host » et « user »

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.

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
| func                      | procédures et fonctions

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

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

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.

Actualisation de la BD mysql : flush privileges

Recharger les tables de droits

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 ordres du DML ne sont pas répercutées en mémoire vive.

Pour actualiser la mémoire vive :

```
mysql> flush privileges
```

On peut aussi faire :

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

ou

```
shell> mysqladmin reload
```

Précisions – 1 : 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** :

Précisions – 2 : 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',

Précisions – 3 : 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;
```

Précisions – 4 : 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.

Précisions – 5 : 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
```

Le dictionnaire des données : la BD Information schema

Présentation

la BD information_shéma

C'est un dictionnaire des données plus standard apparu avec la version 5.0.

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 ».

http://dev.mysql.com/doc/refman/5.5/en/information-schema.html

http://dev.mysql.com/doc/refman/8.0/en/information-schema.html

https://dev.mysql.com/doc/refman/8.0/en/data-dictionary.html : que dans la version 8.0

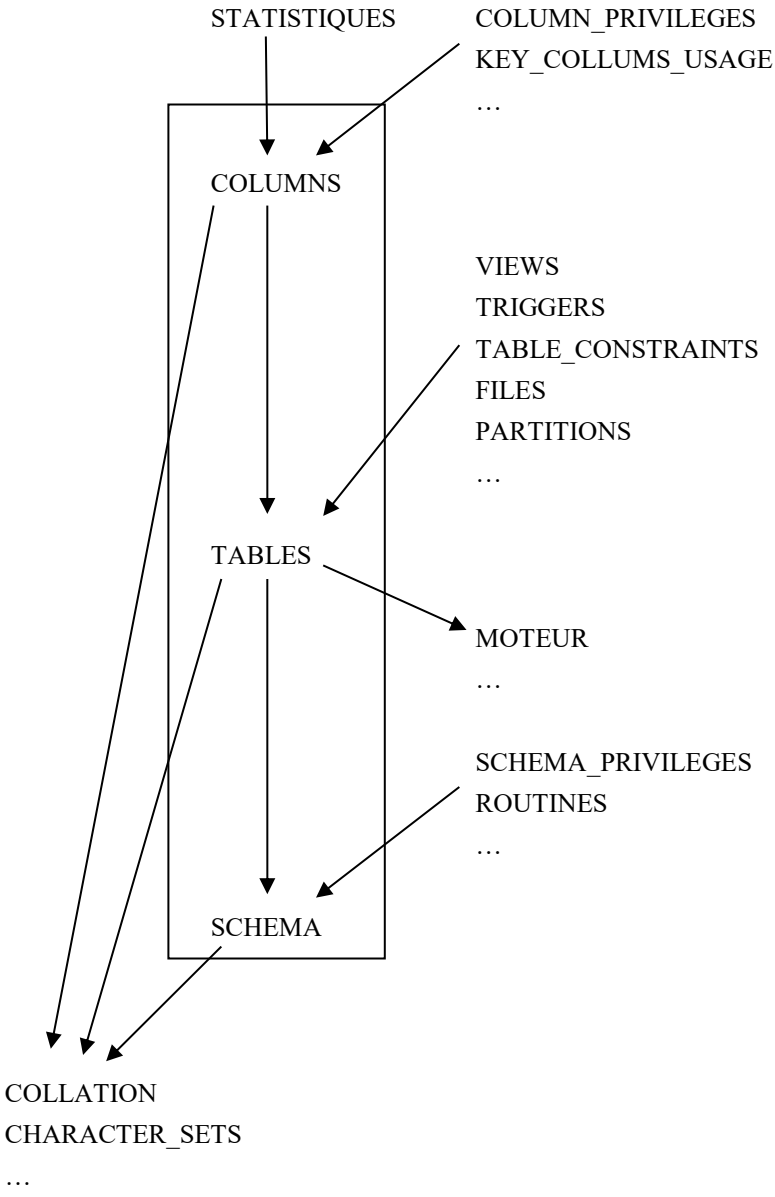
https://dev.mysql.com/doc/refman/8.0/en/innodb-information-schema-transactions.html

Schéma du dictionnaire des données – MySQL 5.1

https://dev.mysql.com/doc/refman/8.0/en/data-dictionary-information-schema.html

1 - Graphe des tables

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



2 – Utilisation de quelques tables du dictionnaire

```
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 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';
```

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.

2 - Ré-écriture de DESC avec le dictionnaire des attributs

Le code suivant permet de réécrire la commande DESC avec une procédure stockée.

On peut se placer dans mysql.

On crée la procédure : elle appartient à la BD. Elle sera donc utilisable soit quand on est dans la BD mysql (après un use mysql), soit en étant préfixée par la BD mysql (comme dans l'exemple ci-dessous).

```
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 ;

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

Précisions – 1 : Schéma du dictionnaire des données – MySQL 5.1

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

Liste des tables

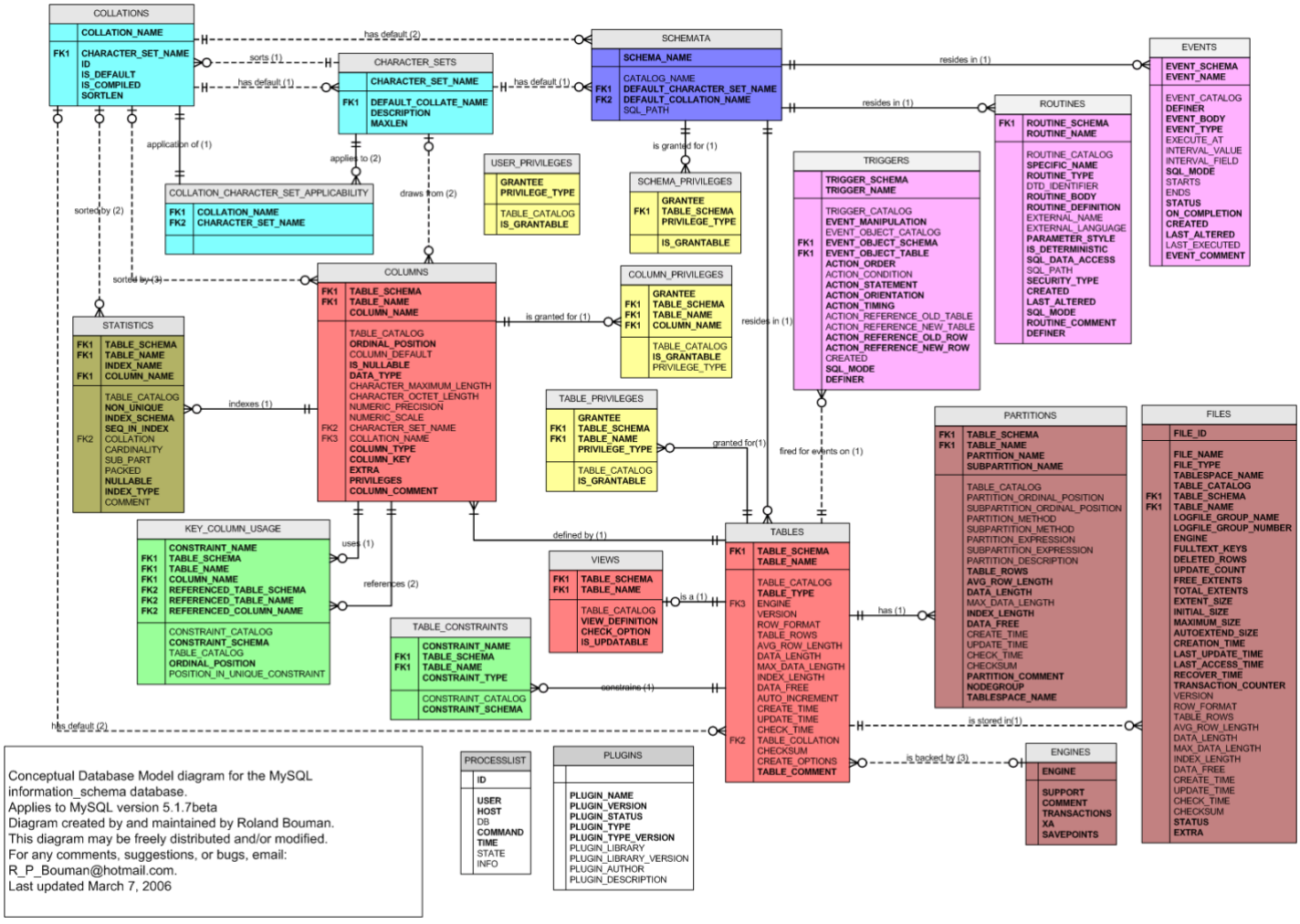
On peut voir les tables de la BD information_schema en faisant un show tables.
https://dev.mysql.com/doc/refman/8.0/en/data-dictionary-information-schema.html

```
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 équivalent)

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.



Précisions – 2 : 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)
```

Interface utilisateur : vues et procédure stockées

Principes d'une interface utilisateur

Plutôt que de laisser un accès direct aux tables de la BD, on peut proposer un accès à travers des vues ou des procédures stockées.

Ca sécurise les données en limitant les accès et en cachant les tables d'origine.

Donner des droits sur une vue : pour faciliter et sécuriser l'accès aux donner par les users

On peut créer des vues et ne donner des droits d'accès que sur ces vues : ça favorise la sécurisation et ça facilite l'accès, les vues étant orientées « usage ».

La vue permet aussi de restreindre l'accès aux tuples.

Pour que cet usage soit pertinent, il faut pouvoir appliquer les opérations du DML (INSERT, UPDATE, DELETE) à une vue.

Exemple

On veut limiter les droits d'accès de toto@localhost à la consultation du n°, du nom et du job des employés des départements 10 et 20 (table emp, BD empdept) :

```
Create view vueEmp as
Select ne, nom, job
From emp where nd in (10, 20);
```

```
Grant SELECT on empdept.vueEmp to toto@localhost;
```

Rappels sur les vues – 1 : Présentation

LES RAPPELS VIENNENT DU COURS : « SGBD_SQL_02_Select_multitables ».

Une commande **SELECT** peut être conservée dans un objet de la BD appelé "**VIEW**" (vue).

En tant que table, une vue n'a qu'une existence virtuelle :

- Les tuples n'ont pas d'existence physique ;
- La vue est recalculée à chaque utilisation ;

La vue est équivalente à une table dans sa structure externe : **elle a des attributs** comme une table.

La vue est équivalente à une table pour la consultation : **on peut faire des SELECT** sur une vue comme sur une table. Toutefois, le contenu de la vue n'a pas d'existence physique : il est recalculé à chaque utilisation de la vue qui se comporte donc comme un select.

Sous certaines conditions, on peut ajouter, modifier ou supprimer des tuples dans une vue et par conséquent dans la ou les tables associées à la vue.

Rappels sur les vues – 2 : Gestion des vues

1 : CREATE VIEW

La syntaxe de la création d'une vue est la suivante :

```
CREATE VIEW nom_vue AS
  SELECT …
```

ou

```
CREATE or REPLACE VIEW nom_vue AS
  SELECT …
```

2 : DROP VIEW

La syntaxe de la suppression d'une vue est la suivante :

```
DROP VIEW nom_vue ;
```

3 : SHOW TABLES : lister les vues existantes

```
SHOW TABLES ;
```

Ou bien

```
SELECT table_schema, table_name FROM information_schema.views;
```

4 : SHOW CREATE TABLE : consulter le code d'une vue

```
SHOW CREATE VIEW nom_vue;
```

5 : SELECT : utilisation d'une vue

Une vue s'utilise comme une table :

```
SELECT * fROM nom_vue;
```

Rappels sur les vues – 3 : Exemple

Création de la vue des départements contenant au moins un ANALYST (c'est un job) :

```
CREATE or REPLACE VIEW deptAvecAnalystes as
  Select distinct nd from emp
  where job = 'ANALYST'
  order by nd;
```

Utilisation de la vue

```
SELECT * fROM deptAvecAnalystes;
```

Rappels sur les vues – 4 : Les 2 usages des vues

1 : décomposer les requêtes en sous-requêtes

Une vue peut remplacer n'importe quelle table dans un select.

Les vues permettent ainsi de décomposer l'écriture des requêtes complexes.

Le select principal joue le rôle de programme principal. Les vues correspondant à des sous-requêtes qui peuvent prendre la place d'une table ou d'un attribut si la vue ne retourne qu'une

ligne.

Exemple

Tous les employés travaillant dans un département qui contient au moins un 'ANALYST' (c'est un métier).

On commence par créer la vue correspondant à la requête : « tous les départements contenant au moins un ANALYST (cf. exemple précédent) :

```
CREATE or REPLACE VIEW deptAvecAnalyste as
Select distinct nd from emp
where job = 'ANALYST'
order by nd;
```

Ensuite on traite la requête principale en exploitant la nouvelle vue :

```
Select ne, nom from emp
where nd in (select nd from deptAvecAnalyste);
```

ou :

```
Select distinct e.ne, e.nom from emp e, deptAvecAnalyste d
where e.nd =d.nd;
```

2 : Faciliter et sécuriser l'accès aux données par les utilisateurs

On peut créer des vues et ne donner des droits d'accès que sur ces vues : ça favorise la sécurisation et ça facilite l'accès, les vues étant orientées « usage ».

La vue permet aussi de restreindre l'accès aux tuples.

Pour que cet usage soit pertinent, il faut pouvoir appliquer les opérations du DML (INSERT, UPDATE, DELETE) à une vue.

Exemple

On veut limiter les droits d'accès de toto@localhost à la consultation du n°, du nom et du job des employés des départements 10 et 20 (table emp, BD empdept) :

```
Create view vueEmp as
Select ne, nom, job
From emp where nd in (10, 20);
```

```
Grant SELECT on empdept.vueEmp to toto@localhost;
```

Rappels sur les vues – 5 : Les vues modifiables

INSERT, UPDATE et REPLACE dans une vue

Vue mono-table

On peut faire des INSERT, des UPDATE et des REPLACE dans une vue mono-table : ils passeront dans la table correspondante, à condition que toutes les contraintes d'intégrité soient correctement prises en compte.

Vue multi-table

On peut faire des INSERT, un UPDATE ou un REPLACE dans une vue multi-table à condition que toutes les contraintes d'intégrité de la table correspondante soient correctement prises en compte.

Mais on ne peut créer, modifier ou remplacer qu'un seul tuple à la fois.

DELETE dans une vue

Vue mono-table

On peut faire des DELETE dans une vue mono-table : ils passeront dans la table correspondante, à condition que toutes les contraintes d'intégrité soient correctement prises en compte.

Vue multi-table

On ne peut pas faire de DELETE dans une vue multi-table.

Opération qui rendent une vue non modifiable

- Group by,
- Les Fonction de groupe : count(), sum(), etc.
- Distinct,
- Union (car il intègre un distinct)
- Les attributs calculés
- Jointure externe
- L'utilisation d'un « rownum » : ORACLE

Exemples

```
UPDATE emp3000
SET deptno=10
WHERE SAL > 3500;      // modification effectuée
```

```
UPDATE emp3000
SET sal=4000          // ça ne passera pas du fait du check option
WHERE deptno > 10;
```

```
DELETE FROM emp3000
WHERE sal < 3000;     // ça ne passera pas du fait du check option
```

TP

Pour ces exercices, mettez en résultat la commande et les résultats obtenus.

On travaille avec une base biblio.

1 : Connexion root@localhost

1. Ouvrez une console (cmd)
2. Ouvrez un client mysql « root ».
3. Afficher votre profil de connexion et votre identité effective.
4. Affichez les databases auxquelles vous avez accès.
5. Affichez les droits de l'utilisateur connecté.
6. Affichez la liste de tous les utilisateurs.
7. Supprimez tous les utilisateurs sauf root@localhost.
8. Consultez les utilisateurs dans l'interface graphique.

2 : Création root@'%'

9. Créer un utilisateur root pouvant se connecter de n'importe quelle machine (on utilisera '%')
10. Donner tous les droits à cet utilisateur.
11. Affichez les droits de cet utilisateur
12. Consultez les droits des utilisateurs dans l'interface graphique.

3 : Connexion root@maMachine

13. Dans une console, affichez le nom de l'hôte de votre machine ainsi que son adresse IP (Précisions 1 dans le cours).
14. Ouvrez un client mysql « root » en précisant le nom de l'hôte de votre machine. Vous pouvez aussi ouvrir un client « root » avec l'adresse IP
15. Afficher votre profil de connexion et votre identité effective : user() et current_user()
16. Affichez les databases auxquelles vous avez accès.

4 : Création de toto@'%'

17. Vous êtes connecté root@maMachine. Créez un utilisateur « toto » qui puisse se connecter de n'importe quelle machine, sans mot de passe, avec la commande CREATE USER.
18. Ouvrez un client mysql « toto »
19. Affichez les databases ; affichez les droits de « toto ». Que constatez-vous ?

5 : Donnez des droits à toto@'%'

20. Vous êtes connecté root@maMachine. Donnez tous les droits sur la base « biblio » à « toto ». Que constatez-vous ?
21. Vous êtes connecté root@localhost. Donnez tous les droits sur la base « biblio » à « toto ». Que constatez-vous ?
22. Ouvrez un client mysql « toto »

23. Affichez les droits de « toto ».
24. Affichez les databases de « toto ».
25. Affichez les tables de la database « biblio » de « toto ».

6 : Consultation du dictionnaire

26. Affichez le nombre de lignes des tables de la BD biblio à partir de la BD information_schema. On utilisera la table « tables ». L'attribut « table_rows » donne le nombre de ligne. L'attribut « table_schema » donne le nom de la BD. L'attribut « table_name » donne le nom de la table.
27. Affichez les attributs des tables de la BD biblio à partir de la BD information_schema. On utilisera la table « columns ».
28. Créer la procédure qui simule le « desc ». Rattachez la à la BD mysql. Utilisez là en étant dans la BD biblio.

7 : Mot de passe

29. Vous êtes connecté root@localhost. Donnez un mot de passe à l'utisateur « toto » (on peut utiliser le « set password for»)
30. Vérifier le mot de passe en vous connectant en tant que « toto ».

8 : Utilisateurs avec des droits spécifiques

31. Dans la BD biblio, créez une vue « dispo » avec les livres et leurs disponibilités. S'ils sont disponibles, on écrira « disponible », s'ils ne sont pas disponibles, on écrira « retour dans X jours », avec ou sans « s » à jour.
32. Créez deux utilisateurs : adhèrent et bibliothécaire pouvant se connecter de partout.
33. Donnez des droits de consultation de la vue « dispo » aux adhérents ; Vérifiez les droits des adhérents.
34. Donnez des droits de consultation de toutes les tables de la BD biblio aux bibliothécaires ainsi que des droits de consultation, d'insertion sur les tables des emprunts, et des adhérents ; Vérifiez les droits des adhérents.
35. Connectez-vous comme adhérent puis comme bibliothécaire et vérifiez les « select » possibles.

9 : Connexion à distance

36. Essayer de vous connecter sur la machine de votre voisin : en tant que « adherent » et en tant que « bibliothecaire ». (N'oubliez-pas de mettre en commentaire le bind-address du my.ini du serveur !) si nécessaire. La commande « hostname » vous donne le nom de votre machine.
37. Afficher votre profil de connexion et votre identité effective.
38. Affichez vos droits.
39. Affichez les databases auxquelles vous avez accès.
40. Affichez les tables auxquelles vous avez accès dans les databases.
41. Essayez de faire des consultations, du DML, du DDL.

10 : Vous avez perdu le mot de passe de root !

42. Vous avez perdu vos mots de passe ! Eteignez et redémarrer le serveur de façon à pouvoir vous reloguer. Quelles options devez-vous utiliser ?
43. Connectez-vous à partir du serveur non protégé. Affichez l'identité et le profil de connexion. Que constatez-vous ?
44. Changez le password de root.