MySQL - concept général

Système de gestion de base de données

MySQL est un système de gestion de base de données, un serveur de bases de données relationnelles. Une base de données est un ensemble organisé de données. SQL signifie « Structured Query Language »

Emplacement des fichiers

/usr/bin Programmes clients
/usr/sbin serveur mysqld
/var/lib/mysql Fichiers de log et bases de données
/usr/share/doc/packages Documentation
/usr/share/mysql Fichiers de messages d'erreurs et jeux de caractères
sql-bench Suites de tests

Comptes par défaut

Le script **mysql_install_db** démarre le serveur mysqld et initialise les tables de droits, avec les paramètres suivants :

Deux comptes MySQL **root** sont créés en tant qu'administrateurs ayant tous les droits. Le mot de passe de l'utilisateur initial **root** est vide ou définit pendant l'installation.

Deux comptes **utilisateur anonyme** sont créés, qui peuvent faire ce qu'ils veulent avec toutes les tables dans la base de données '**test**' ou commençant par '**test_**'. Cela signifie qu'un utilisateur peut se connecter sans mot de passe et être traité comme un utilisateur anonyme.

Les connexions doivent être faîtes en spécifiant le nom d'hôte. Ces comptes ont tous les droits dans les bases **test** ou dont le nom commence par **test_**.

Pour modifier les mots de passe utiliser **SET PASSWORD**
shell> mysql -u root
mysql> SET PASSWORD FOR ''@'localhost' = PASSWORD('nouveau_mot');
mysql> SET PASSWORD FOR ''@'host_name' = PASSWORD('nouveau_mot');
Pour connaître le **host_name** taper
mysql> SELECT Host, User FROM mysql.user;
On peut utiliser la commande UPDATE pour changer les mots de passe des comptes anonymes.
shell> mysql -u root
mysql> UPDATE mysql.user SET Password = PASSWORD('nouveau_mot') WHERE User = '';
flush privileges demande au serveur de relire les tables de droits
mysql> FLUSH PRIVILEGES;
si on préfère supprimer les comptes anonymes
shell> mysql -u root
mysql> DELETE FROM mysql.user WHERE User = '';
mysql> FLUSH PRIVILEGES;

Introduction

Connexion au serveur MySql
mysql -h hote -u utilisateur -p
pour se déconnecter
QUIT ou EXIT
pour obtenir le numéro de version du serveur MySQL
SELECT VERSION(), CURRENT_DATE ;
entrer plusieurs requêtes sur une seule ligne
SELECT VERSION() ; SELECT NOW() ;
entrer une commande sur plusieurs, ligne : la fin d'une requête se termine par ' ;'
mysql> SELECT
-> USER()
-> ,
-> CURRENT_DATE ;
Annuler une requête en cour de frappe, sur une seule ligne taper
\c
afficher les bases de données existantes
SHOW DATABASES ;
accéder à une base
use test
créer une base de donnée
create database test ;
se connecter à mysql directement dans une base
mysql -u utilisateur -ppassword test
créer une table
CREATE TABLE animal (nom VARCHAR(20), maitre VARCHAR(20),
-> espece VARCHAR(20), sexe CHAR(1), naissance DATE, mort DATE) ;
pour afficher la structure de la table
describe animal ;
créer un fichier pour remplir une table, chaque champ doit être séparé par une tabulation
pour importer un fichier dans une table :
LOAD DATA LOCAL INFILE "animal.txt" INTO TABLE animal ;
ajouter des enregistrement un par un (utiliser NULL pour indiquer une valeur manquante)
mysql> INSERT INTO animal
-> VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL) ;
récupérer des informations
SELECT quoi_selectionner
FROM quel_table
WHERE conditions_a_satisfaire
pour tout sélectionner
SELECT * from animal ;
corriger une erreur dans une table, il existe au moins 2 façons de le faire. la première consiste a vider la table, corriger le fichier et le recharger
mysql> SET AUTOCOMMIT=1 ; # Utilisé pour une recréation rapide de la table
mysql> DELETE FROM animal ;
mysql> LOAD DATA LOCAL INFILE "animal.txt" INTO TABLE animal ;
ou corriger l'enregistrement avec update
mysql> UPDATE animal SET naissance = "1989-08-31" WHERE nom = "Bowser" ;
sélectionner des lignes particulières :
mysql> SELECT * FROM animal WHERE nom = "Bowser" ;
avec les booléennes
mysql> SELECT * FROM animal WHERE espece = "chien" AND sexe = "f" ;
plus complexe
mysql> SELECT * FROM animal WHERE (espece = "chat" AND sexe = "m")
-> OR (espece = "chien" AND sexe = "f") ;
sélectionner des colonnes
mysql> SELECT nom, naissance FROM animal ;
pour minimiser le résultat à des champs uniques utiliser DISTINCT
mysql> SELECT DISTINCT maitre FROM animal ;

trie des enregistrements
mysql> SELECT nom, naissance FROM animal ORDER BY naissance ;
pour trier en respectant la casse
mysql> SELECT nom, naissance FROM animal ORDER BY BINARY naissance ;
pour trier dans l'ordre décroissant
mysql> SELECT nom, naissance FROM animal ORDER BY naissance DESC ;
on peut trier sur plusieurs colonnes
mysql> SELECT nom, espece, naissance FROM animal ORDER BY espece, naissance DESC ;
ici l'ordre décroissant n'est appliqué qu'à naissance.

Calcul des Dates

Pour calculer l'âge par rapport à une date de naissance, il faut calculer la différence entre l'année en cour et l'année de naissance, puis soustraire la date courante si la date du jour se produit plus tôt dans l'année civile que la date de naissance
mysql> SELECT nom, naissance, CURRENT_DATE,
-> (YEAR(CURRENT_DATE)-YEAR(naissance))
-> - (RIGHT(CURRENT_DATE,5)<RIGHT(naissance,5))
-> AS age
-> FROM animal ;
YEAR extrait l'année de la date et RIGHT(naissance,5) extrait les 5 caractères les plus à droite de la date qui représente MM-DD (année civile).
Pour calculer l'âge d'un animal à sa mort, s'il est déjà mort
mysql> SELECT nom, naissance, mort,
-> (YEAR(mort)-YEAR(naissance)) - (RIGHT(mort,5)<RIGHT(naissance,5))
-> AS age
-> FROM animal WHERE mort IS NOT NULL ORDER BY age ;
si l'animal est mort, le champ mort est différent de NULL.
pour afficher dans combien de mois est leur anniversaire
mysql> SELECT nom, naissance, MONTH(naissance) FROM animal ;
ceux dont l'anniversaire est dans 5 mois :
mysql> SELECT nom, naissance FROM animal WHERE MONTH(naissance) = 5 ;
autres valeurs utilisables : YEAR(), MONTH(), et DAYOFMONTH().
on peut utiliser des fonctions de calcul d'intervalle
mysql> SELECT nom, naissance FROM animal
-> WHERE MONTH(naissance) = MONTH(DATE_ADD(NOW(), INTERVAL 1 MONTH)) ;
autre manière pour un résultat similaire
mysql> SELECT nom, naissance FROM animal
-> WHERE MONTH(naissance) = MOD(MONTH(NOW()), 12) + 1 ;
valeur NULL : on ne peut pas utiliser de comparateur : '=', '<', '>' ou '<>'. il faut utiliser IS NULL ou IS NOT NULL.

Recherche de modèles

'_' représente n'importe quel caractère, '%' un nombre arbitraire de caractères. on n'utilise pas = ni <> mais LIKE et NOT LIKE.
mysql> SELECT * FROM animal WHERE nom LIKE "b%" ;
mysql> SELECT * FROM animal WHERE nom LIKE "%w%" ;
mysql> SELECT * FROM animal WHERE nom LIKE "_____" ;
les expressions régulières étendues. avec ce type de modèle, utiliser REGEXP et NOT REGEXP ou RLIKE et NOT RLIKE
'.' représente n'importe quel caractère, [...] n'importe quel caractère inclus entre les crochets, fonctionne aussi comme [a-z].
'*' nombre arbitraire de caractère, '^' pour spécifier en début de chaîne ou '$' en fin de chaîne. REGEXP n'est pas sensible à la casse, utiliser binary pour la respecter.
mysql> SELECT * FROM animal WHERE nom REGEXP "^b" ;
mysql> SELECT * FROM animal WHERE nom REGEXP "^[bB]" ;
mysql> SELECT * FROM animal WHERE nom REGEXP BINARY "^b" ;
mysql> SELECT * FROM animal WHERE nom REGEXP "fy$" ;
REGEXP cherche le motif recherché donc pas besoin de * au contraire de LIKE :

mysql> SELECT * FROM animal WHERE nom REGEXP "w";
pour recherche une chaîne exacte : .n signifie répéter n fois.
mysql> SELECT * FROM animal WHERE nom REGEXP "^.....$";
mysql> SELECT * FROM animal WHERE nom REGEXP "^.5$";

Compter les lignes

La fonction COUNT() compte le nombre de résultats non NULL
mysql> SELECT COUNT(*) FROM animal;
mysql> SELECT maitre, COUNT(*) FROM animal GROUP BY maitre;
la clause GROUP BY est nécessaire pour grouper tous les enregistrements par maître, sinon on a une erreur
mysql> SELECT espece, sexe, COUNT(*) FROM animal GROUP BY espece, sexe;
mysql> SELECT espece, sexe, COUNT(*) FROM animal
-> WHERE espece = "chien" OR espece = "chat"
-> GROUP BY espece, sexe;
mysql> SELECT espece, sexe, COUNT(*) FROM animal
-> WHERE sexe IS NOT NULL
-> GROUP BY espece, sexe;

Utiliser plus d'une table

mysql> SELECT animal.nom,
-> (TO_DAYS(date) - TO_DAYS(naissance))/365 AS age, remarque
-> FROM animal, evenement
-> WHERE animal.nom = evenement.nom AND type = "mise bas";
on peut joindre une table sur elle-même
mysql> SELECT p1.nom, p1.sexe, p2.nom, p2.sexe, p1.espece
-> FROM animal AS p1, animal AS p2
-> WHERE p1.espece = p2.espece AND p1.sexe = "f" AND p2.sexe = "m";
Obtenir des informations à propos des bases de données et des tables
Show databases
montrer dans quelle base on est
select database()
montrer les tables
Show Tables
afficher les champs de la table
Describe animal

Utilisation de mysql en mode batch

Pour utiliser mysql en mode batch, placer les commandes dans un fichier et utiliser la commande
mysql < fichier-batch
capturer l'affichage
mysql < fichier.batch > mysql.out
la sortie en mode batch est différente du mode interactif, pour obtenir le même affichage utiliser l'option mysql -t . Pour écrire les commandes
mysql -vvv.
pour exécuter un fichier depuis la commande
source nom_fichier

Exemples de requêtes usuelles

valeur maximale d'une colonne
SELECT MAX(article) AS article FROM shop
la ligne contenant le maximum d'une certaine colonne
SELECT article, dealer, price FROM shop WHERE price=(SELECT MAX(price) FROM shop);
la même chose en triant dans l'ordre décroissant, puis afficher la première ligne
SELECT article, dealer, price FROM shop ORDER BY price DESC LIMIT 1;
maximum d'une colonne par groupe
SELECT article, MAX(price) AS price FROM shop GROUP BY article
la ligne contenant la plus grande valeur d'un certain champ par rapport à un groupe
SELECT article, dealer, price FROM shop s1 WHERE price=(SELECT MAX(s2.price) FROM shop s2 WHERE s1.article = s2.article);
la même en plusieurs étapes, en créant une table temporaire
CREATE TEMPORARY TABLE tmp (
article INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,
price DOUBLE(16,2) DEFAULT '0.00' NOT NULL);
LOCK TABLES shop read;
INSERT INTO tmp SELECT article, MAX(price) FROM shop GROUP BY article;
SELECT shop.article, dealer, shop.price FROM shop, tmp
WHERE shop.article=tmp.article AND shop.price=tmp.price;
UNLOCK TABLES;
DROP TABLE tmp;
ou encore (en une étape)
SELECT article,
SUBSTRING(MAX(CONCAT(LPAD(price,6,'0'),dealer)), 7) AS dealer,
0.00+LEFT(MAX(CONCAT(LPAD(price,6,'0'),dealer)), 6) AS price
FROM shop
GROUP BY article;
Utiliser les variables utilisateur. Trouver l'article avec le plus haut et le plus bas prix
mysql> SELECT @min_price :=MIN(price),@max_price :=MAX(price) FROM shop;
mysql> SELECT * FROM shop WHERE price=@min_price OR price=@max_price;
utiliser les clés étrangères
CREATE TABLE person (
id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
name CHAR(60) NOT NULL,
PRIMARY KEY (id)
);
CREATE TABLE shirt (
id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
style ENUM('t-shirt', 'polo', 'dress') NOT NULL,
color ENUM('red', 'blue', 'orange', 'white', 'black') NOT NULL,
owner SMALLINT UNSIGNED NOT NULL REFERENCES person(id),
PRIMARY KEY (id)
);
INSERT INTO person VALUES (NULL, 'Antonio Paz');
INSERT INTO shirt VALUES
(NULL, 'polo', 'blue', LAST_INSERT_ID()),
(NULL, 'dress', 'white', LAST_INSERT_ID()),
(NULL, 't-shirt', 'blue', LAST_INSERT_ID());
INSERT INTO person VALUES (NULL, 'Lilliana Angelovska');
INSERT INTO shirt VALUES
(NULL, 'dress', 'orange', LAST_INSERT_ID()),
(NULL, 'polo', 'red', LAST_INSERT_ID()),
(NULL, 'dress', 'blue', LAST_INSERT_ID()),
(NULL, 't-shirt', 'white', LAST_INSERT_ID());
SELECT * FROM person;

```
+---+--------------+
|-id-|------name-----|
+---+--------------+
|-1-|-Antonio-Paz------|
```

```
|-2-|-Lilliana-Angelovska-|
+---+--------------+
```

SELECT * FROM shirt;

```
+---+------+-----+-----+
|-id-|-style-|-color-|-owner-|
+---+------+-----+-----+
|-1-|-polo---|-blue--|---1-|
|-2-|-dress--|-white-|---1-|
|-3-|-t-shirt-|-blue--|---1-|
|-4-|-dress--|-orange-|---2-|
|-5-|-polo---|-red---|---2-|
|-6-|-dress--|-blue--|---2-|
|-7-|-t-shirt-|-white-|---2-|
+---+------+-----+-----+
```

SELECT s.* FROM person p, shirt s WHERE p.name LIKE 'Lilliana%' AND s.owner = p.id AND s.color <> 'white';

```
+---+-----+-----+-----+
|-id-|-style-|-color-|-owner-|
+---+-----+-----+-----+
|-4-|-dress-|-orange-|-2---|
|-5-|-polo-|-red--|-2---|
|-6-|-dress-|-blue--|-2---|
+---+-----+-----+-----+
```

Recherche sur deux clefs

SELECT champ1_index, champ2_index FROM test_table WHERE champ1_index = '1'
OR champ2_index = '1'

avec UNION

SELECT field1_index, field2_index FROM test_table WHERE field1_index = '1'
UNION
SELECT field1_index, field2_index FROM test_table WHERE field2_index = '1';

calcul du nombre de visites par jour

CREATE TABLE t1 (year YEAR(4), month INT(2) UNSIGNED ZEROFILL,
day INT(2) UNSIGNED ZEROFILL);
INSERT INTO t1 VALUES(2000,1,1),(2000,1,20),(2000,1,30),(2000,2,2),
(2000,2,23),(2000,2,23);
SELECT year,month,BIT_COUNT(BIT_OR(1«day)) AS days FROM t1
GROUP BY year,month;

Utiliser AUTO_INCREMENT. L'attribut AUTO_INCREMENT peut être utilisé pour générer un identifiant unique pour les nouvelles lignes

CREATE TABLE animals (
id MEDIUMINT NOT NULL AUTO_INCREMENT,
name CHAR(30) NOT NULL,
PRIMARY KEY (id)
);
INSERT INTO animals (name) VALUES ("dog"),("cat"),("penguin"),
("lax"),("whale"),("ostrich");
SELECT * FROM animals;

On peut récupérer la valeur utilisée de la clé AUTO_INCREMENT avec la fonction LAST_INSERT_ID() ou la fonction API mysql_insert_id().

Pour les tables MyISAM et BDB on peut spécifier AUTO_INCREMENT sur une colonne secondaire d'une clef multi-colonnes. Dans ce cas la valeur généré est calculée de la façon suivante :

MAX(auto_increment_column)+1) WHERE prefix=given-prefix .

Utile lorsqu'on veut placer des données dans des groupes ordonnés.

CREATE TABLE animals (

grp ENUM('fish','mammal','bird') NOT NULL,
id MEDIUMINT NOT NULL AUTO_INCREMENT,
name CHAR(30) NOT NULL,
PRIMARY KEY (grp,id)
);
INSERT INTO animals (grp,name) VALUES("mammal","dog"),("mammal","cat"),
("bird","penguin"),("fish","lax"),("mammal","whale"),
("bird","ostrich");
SELECT * FROM animals ORDER BY grp,id;
Qui retourne :

```
+-----+---+------+
|-grp--|-id-|-name---|
+-----+---+------+
|-fish--|-1-|-lax---|
|-mammal-|-1-|-dog---|
|-mammal-|-2-|-cat---|
|-mammal-|-3-|-whale--|
|-bird--|-1-|-penguin-|
|-bird--|-2-|-ostrich-|
+-----+---+------+
```

Utilisation de MySQL avec Apache

Pour changer le format d'archivage d'Apache dans MySQL pour le rendre plus lisible, mettre ceci dans la configuration d'Apache :
**LogFormat **
**"\"%h\",%%Y%m%d%H%M%St,%>s,\"%b\",\"%Content-Typeo\", **
\"%U\",\"%Refereri\",\"%User-Agenti\""
Avec MySQL, vous pouvez exécuter une requête de cette manière :
LOAD DATA INFILE '/local/access_log' INTO TABLE table_name
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' ESCAPED BY '\\'

Administration du serveur

mysqld est le serveur MySQL

mysqld-max version du serveur qui inclut des fonctionnalités supplémentaires

mysqld-safe est un script de démarrage du serveur. Tente de démarrer mysqld-max s'il existe sinon mysqld.

mysql.server est un script de démarrage du serveur, utilisé sur les systèmes qui ont un dossier contenant des services système. Il invoque mysqld-safe pour démarrer le serveur.

mysqld_multi est un script de démarrage qui peut lancer ou arrêter différentes instances du serveur, installées sur le système.

mysql_install_db crée les tables de droits MySQL, avec les droits par défaut.

mysql_fix_privilege_tables script utilisé après une mise à jour de MySQL pour mettre à jour les tables de droits, et les adapter aux nouvelles versions de MySQL.

myisamchk utilitaire pour décrire, vérifier, optimiser et réparer les tables MyISAM

make_binary_distribution crée une version compilée de MySQL.

mysqlbug script de rapport de bug de MySQL.

Pour connaître les moteurs de stockage que votre serveur supporte, utiliser la commande **SHOW ENGINES ; safe_mysqld** est la méthode recommandée pour démarrer un démon mysqld. Il ajoute des fonctionnalités de sécurité telles que le redémarrage automatique lorsqu'une erreur survient et l'enregistrement d'informations d'exécution dans un fichier de log. Par défaut il essaie de lancer mysqld-max s'il existe. Pour remplacer le comportement par défaut et spécifier explicitement le serveur à utiliser, spécifier **–mysqld** ou **–mysqld-version**.

Le mode SQL du serveur

Le serveur MySQL peut fonctionner avec différents modes SQL, et peut s'appliquer au niveau de la connexion du client. Cela permet aux applications d'adapter le comportement du serveur en fonction de leur attentes. Le mode définit quelle syntaxe SQL le serveur doit supporter, et quels types de vérification il doit faire. Pour donner un mode par défaut au démarrage : **–sql-mode="modes"**. On peut modifier le mode à chaud avec **SET [SESSION|GLOBAL] sql_mode='modes'**

Modes

ANSI change la syntaxe et le comportement pour être plus compatible avec le standard SQL

STRICT_TRANS_TABLES Si une valeur n'a pu être insérée dans une table transactionnelle sans modification, la commande est annulée. Pour une tables non-transactionnelle, la commande est annulée si cela survient dans une ligne unique ou dans la première ligne d'une insertion multiple.

TRADITIONAL le serveur se comporte comme un système SQL traditionnel

ALLOW_INVALID_DATES N'autorise pas la vérification totale des dates. Vérifie simplement que le mois est dans l'intervalle de 1 à 12, et que le jour est dans l'intervalle de 1 à 31. C'est très pratique pour les applications Web où la date est obtenue de 3 champs différents, et que vous voulez stocker exactement la date saisie sans validation. Ce mode s'applique aux colonnes de type DATE et DATETIME. Il ne s'applique pas aux colonnes TIMESTAMP, qui demandent toujours une date valide.

ANSI_QUOTES Traite '"' comme un délimiteur d'identifiant (comme le caractère MySQL "') et non comme un délimiteur de chaînes. Vous pouvez toujours utiliser "' pour délimiter les identifiants en mode ANSI. Avec ANSI_QUOTES activée, vous ne pouvez pas utiliser les guillemets doubles pour délimiter une chaîne de caractères, car ce sera uniquement interprété comme un identifiant.

ERROR_FOR_DIVISION_BY_ZERO Produit une erreur en mode strict et sinon une alerte, lorsque MySQL doit tenter une division par 0 ou un MOD(X,0)) durant une commande INSERT/ UPDATE. Si ce mode n'est pas activé, MySQL retourne simplement NULL pour les divisions par zéro. Si utilisé avec l'attribut IGNORE, MySQL génère une alerte pour les divisions par zéro, mais le résultat de l'opération sera NULL

IGNORE_SPACE Permet les espaces entre le nom de la fonction et le caractère '('. Cela force les noms de fonctions a être traités comme des mots réservés. En conséquence, si vous voulez accéder aux bases, tables et colonnes dont le nom est un mot réservé, vous devez le mettre entre délimiteurs.

NO_AUTO_VALUE_ON_ZERO affecte la gestion des colonnes de type AUTO_INCREMENT. Normalement, vous générez le prochain numéro de séquence dans la colonne en insérant soit NULL soit 0 dedans. NO_AUTO_VALUE_ON_ZERO supprime ce comportement pour 0 pour que seule la valeur NULL génère le prochain numéro de séquence. Ce mode est utile si vous avez stocké la valeur 0 dans la colonne AUTO_INCREMENT de la table. Ce n'est pas recommandé. Par exemple, si vous voulez exporter une table avec mysqldump et que vous la rechargez, normalement MySQL va générer de nouveaux identifiants pour les lignes avec la valeur 0, ce qui entraînera une différence avec la table originale. En activant NO_AUTO_VALUE_ON_ZERO avant de recharger le fichier exporter, vous évitez des problèmes. mysqldump va automatiquement inclure les commandes nécessaires dans l'export, pour activer NO_AUTO_VALUE_ON_ZERO.

NO_DIR_IN_CREATE Lors de la création d'une table, ignore les directives INDEX DIRECTORY et DATA DIRECTORY. Cette option est pratique sur un esclave de réplication.

NO_FIELD_OPTIONS N'affiche pas les options spécifiques à MySQL dans le résultat de SHOW CREATE TABLE. Ce mode est utilisé par mysqldump dans un souci de portabilité

NO_KEY_OPTIONS N'affiche pas les options spécifiques à MySQL dans le résultat de SHOW CREATE TABLE. Ce mode est utilisé par mysqldump dans un souci de portabilité.

NO_TABLE_OPTIONS N'affiche pas les options de tables spécifiques à MySQL (comme ENGINE) dans le résultat de SHOW CREATE TABLE. Ce mode est utilisé par mysqldump dans un souci de portabilité.

NO_ZERO_DATE Ne permet pas l'utilisation de '0000-00-00' comme date valide. Vous pouvez toujours insérer des dates nulles avec l'option IGNORE. NO_ZERO_IN_DATE N'accepte pas les dates où le mois ou le jour vaut 0. Si utilisé avec L'option IGNORE, la date '0000-00-00' sera insérée pour chaque date invalide.

NO_UNSIGNED_SUBTRACTION Dans les opérations de soustraction, ne marque pas le résultat UNSIGNED si un des opérandes est non signé. Notez que cela fait que UNSIGNED BIGINT n'est plus totalement utilisable dans tous les contextes.

ONLY_FULL_GROUP_BY N'autorise pas les requêtes dont la clause GROUP BY fait référence à une colonne qui n'est pas sélectionnée.

PIPES_AS_CONCAT Traite || comme un opérateur de concaténation (identique à CONCAT()) au lieu d'être un synonyme de OR.

REAL_AS_FLOAT Traite le type REAL comme un synonyme FLOAT plutôt que comme un synonyme de DOUBLE.

STRICT_ALL_TABLES Active le mode strict pour tous les moteurs de stockage. Les valeurs invalides sont rejetées. Plus de détails suivent.

STRICT_TRANS_TABLES Active le mode strict pour tous les moteurs de stockage transactionnels. Les valeurs invalides sont rejetées. Plus de détails suivent.

Variables système dynamique

modifier une variable au lancement du serveur
mysqld –key_buffer_size=16M
modifier une variable à chaud
mysql> SET sort_buffer_size = 10 * 1024 * 1024 ;
Spécifier explicitement s'il s'agit d'une variable global ou de session
mysql> SET GLOBAL sort_buffer_size = 10 * 1024 * 1024 ;
mysql> SET SESSION sort_buffer_size = 10 * 1024 * 1024 ;
Voir les variables du serveur
SHOW variables
Affiche des informations sur le statut du serveur
SHOW STATUS
Remet à 0 de nombreuses variables de status
FLUSH STATUS
Arrêter le serveur : si un utilisateur a les droit SHUTDOWN il peut arrêter le serveur avec
mysqladmin shutdown

Sécurité général du serveur

MySQL dispose d'un système de sécurité basé sur des liste de contrôle d'accès (ACL) pour toutes les connexions, requêtes et opérations que l'utilisateur peut faire. Il y'a aussi le support des connexions SSL entre le client et le serveur MySQL.

Suivre les règles suivante :

- Ne donnez jamais à personne (sauf au compte MySQL root) accès à la table user de la base mysql !. Le mot de passe chiffré est le vrai mot de passe de MySQL.
- Apprenez le système de droits MySQL. Les commandes GRANT et REVOKE sont utilisés pour contrôler les accès à MySQL. Ne donnez pas plus de droits nécessaire. Ne donnez jamais de droits à tous les serveurs hôtes.
- Ne stockez jamais de mot de passe en clair dans votre base. À la place, utilisez MD5(), SHA1() ou une autre fonction de signature injective
- utilisez des mots de passe fort.
- Placez votre serveur derrière un parefeu

Liste de vérification

- Essayez la commande **mysql -u root** sans mot de passe
- Utilisez la commande **SHOW GRANTS** et vérifiez qui a accès à quoi. Puis utilisez la commande **REVOKE** pour retirer les droits inutiles.

Protéger MySQL contre les attaques

Toutes les informations autre que les mots de passes sont transférées en clair. Pour sécuriser le traffic, il est recommandé d'utiliser le protocole compressé et d'utiliser ssh pour établir des connexions chiffrées.

- N'exécutez jamais le démon MySQL avec l'utilisateur root. Pour démarrer mysqld sous un autre nom d'utilisateur Unix, ajoutez la ligne user qui spécifie le nom de l'utilisateur, dans le fichier d'options de [mysqld] de /etc/my.cnf.
- N'autorisez pas l'utilisation de liens symboliques pour les tables. Cette fonctionnalité peut être désactivée avec l'option **–skip-symbolic-links**. C'est particulièrement important si vous utilisez mysqld comme root, car tout utilisateur a alors le droit d'écrire des données sur le disque, n'importe où sur le système ! !
- Vérifiez que l'utilisateur Unix qui exécute mysqld est le seul utilisateur avec les droits de lecture et écriture dans le dossier de base de données.
- Ne donnez pas le droit de PROCESS à tous les utilisateurs. Le droit SUPER peut être utilisé pour fermer des connexions clients, changer les variables systèmes et contrôler la réplication.
- Ne donnez pas le droit de FILE à tous les utilisateurs. Tout utilisateur qui possède ce droit peut écrire un fichier n'importe où sur le serveur, avec les droits hérités du démon mysqld.
- Si vous voulez restreindre le nombre de connexions d'un utilisateur, vous pouvez le faire en utilisant la variable **max_user_connections** de mysqld. La commande GRANT dispose aussi d'options de contrôle des ressources, pour limiter l'utilisation du serveur par un compte utilisateur.

Options de démarrage qui concerne la sécurité

–local-infile [=(0|1)] si vous utilisez –local-infile=0 alors vous ne pourrez pas utiliser LOAD DATA LOCAL INFILE

–safe-user=create activé, tout utilisateur ne peut pas créer d'autres utilisateurs avec les droits de GRANT, s'il ne dispose pas des droits d'insertion dans la table mysql.user Si vous voulez donner un accès à un utilisateur pour qu'il puisse créer des utilisateurs avec les droits dont il dispose, vous pouvez lui donner les droits suivant : **GRANT INSERT(user) ON mysql.user TO 'user'@'hostname' ;** cela va assurer que l'utilisateur ne peut pas modifier une colonne directement, mais qu'il peut exécuter la commande GRANT sur d'autres utilisateurs.

–secure-auth interdit l'identification pour les comptes qui ont d'anciens mot de passe (<4.1.1)

–skip-grant-tables force le serveur à ne pas utiliser les tables de droits. Donne tous les droits a tous les utilisateurs.

–skip-name-resolve les nom d'hôtes ne sont pas résolus. toutes les valeurs de la colonne host dans les tables de droits doivent être des adresses ip, ou bien localhost

–skip-networking Ne pas accepter les connexions TCP/IP venant du réseau, mais uniquement via socket Unix.

Problèmes de sécurité avec LOAD DATA LOCAL

La commande **LOAD DATA** peut lire des données sur le serveur hôte ou charger un fichier sur le client avec LOCAL.

Pour le client en ligne de commande mysql, **LOAD DATA LOCAL** peut être activé en spécifiant l'option **–local-infile[=1]**, ou désactivé avec **–local-infile=0**.

Vous pouvez désactiver toutes les commandes **LOAD DATA LOCAL** du serveur MySQL en démarrant mysqld avec **–local-infile=0**. Similairement, pour mysqlimport, les options –local et -L activent le chargement distant de fichiers. Dans ce cas, il faut que le serveur accepte aussi cette configuration pour que l'opération fonctionne.

Règles de sécurité et droits d'accès au serveur

Table name	utilisateur	base	hôte
Scope fields	Host	Host	Host
	User	Db	Db
	Password	User	
Privilege fields	Select_priv	Select_priv	Select_priv
	Insert_priv	Insert_priv	Insert_priv
	Update_priv	Update_priv	Update_priv
	Delete_priv	Delete_priv	Delete_priv
	Index_priv	Index_priv	Index_priv
	Alter_priv	Alter_priv	Alter_priv
	Create_priv	Create_priv	Create_priv
	Drop_priv	Drop_priv	Drop_priv
	Grant_priv	Grant_priv	Grant_priv
	References_priv	References_priv	References_priv
	Reload_priv		
	Shutdown_priv		
	Process_priv		
	File_priv		
	Show_db_priv		
	Super_priv		
	Create_tmp_table_priv	Create_tmp_table_priv	Create_tmp_table_priv
	Lock_tables_priv	Lock_tables_priv	Lock_tables_priv
	Execute_priv		
	Repl_slave_priv		
	Repl_client_priv		
	ssl_type		
	ssl_cypher		
	x509_issuer		
	x509_cubject		
	max_questions		
	max_updates		
	max_connections		

Lors de la seconde étape du contrôle d'accès (vérification de la requête), le serveur peut, suivant la requête, consulter aussi les tables tables_priv et columns_priv. Les champs de ces tables sont :

Nom de la table	tables_priv	columns_priv
Champ	Host	Host
	Db	Db
	User	User
	Table_name	Table_name
		Column_name
Droit	Table_priv	Column_priv
	Column_priv	
Autre champ	Timestamp	Timestamp
	Grantor	

Chaque table de droit contient des champs d'identification et des champs de droits.

- Les champs d'identification déterminent quels utilisateurs correspondent à cette ligne dans la table.

- Les champs de droits indiquent si le droit est donné. Les champs d'identification sont des chaînes :

Nom de la colonne	Type
Host	CHAR(60)
User	CHAR(16)
Password	CHAR(16)
Db	CHAR(64)
Table_name	CHAR(60)
Column_name	CHAR(60)

Dans les tables **user**, **db** et **host** tous les champs de droits sont déclarés avec le type **ENUM('N','Y')**

Dans les tables **tables_priv** et **column_priv** les champs de droits sont déclarés comme des champs de type **SET**.

Nom de la table	Nom du champs	Valeurs possibles
tables_priv	Table_priv	'Select', 'Insert', 'Update', 'Delete', 'Create', 'Drop', 'Grant', 'References', 'Index', 'Alter'
tables_priv	Column_priv	'Select', 'Insert', 'Update', 'References'
columns_priv	Column_priv	'Select', 'Insert', 'Update', 'References'

Le serveur utilise les tables de droits comme ceci :

- La table user détermine si le serveur accepte ou rejette la connexion. Pour les connexions acceptées, tous les privilèges donnés dans la table user indiquent des privilèges globaux. Ces droits s'appliquent à toutes les bases du serveur.
- Les champs d'identification de la table db déterminent quels utilisateurs peuvent accéder à quelles bases, depuis quel hôte. Les champs de droits indiquent les opérations permises. Les droits s'appliquent alors à toutes les bases du serveur.
- La table host est utilisée comme extension de la table db lorsque vous voulez qu'une ligne de la table db s'applique à plusieurs hôtes dans votre réseau, laissez la colonne host vide dans la table db.
- Les tables tables_priv et columns_priv sont similaires à la table db, mais elles s'appliquent au niveau des tables et des colonnes, plutôt qu'au niveau des tables.

Notez que les droits d'administration ne sont spécifiés que dans la table user, tout comme FILE.

Pour voir les droit d'un utilisateur :

SHOW GRANTS FOR 'bob'@'pc84.example.com' ;

Un outil de diagnostique pratique est le script mysqlaccess.

Droits fournis par MySQL

Droit	Colonne	Contexte
ALTER	Alter_priv	tables
DELETE	Delete_priv	tables
INDEX	Index_priv	tables
INSERT	Insert_priv	tables
SELECT	Select_priv	tables
UPDATE	Update_priv	tables
CREATE	Create_priv	bases de données, tables ou index
DROP	Drop_priv	bases de données ou tables
GRANT	Grant_priv	bases de données ou tables
REFERENCES	References_priv	bases de données ou tables
CREATE TEMPORARY TABLES	Create_tmp_table_priv	administration du serveur
EXECUTE	Execute_priv	administration du serveur
FILE	File_priv	accès aux fichiers du serveur
LOCK TABLES	Lock_tables_priv	administration du serveur
PROCESS	Process_priv	administration du serveur
RELOAD	Reload_priv	administration du serveur
REPLICATION CLIENT	Repl_client_priv	administration du serveur
REPLICATION SLAVE	Repl_slave_priv	administration du serveur
SHOW DATABASES	Show_db_priv	administration du serveur
SHUTDOWN	Shutdown_priv	administration du serveur
SUPER	Super_priv	administration du serveur

Les droits de **SELECT**, **INSERT**, **UPDATE** et **DELETE** vous permettent de faire des opérations sur les lignes qui existent, dans une table existante d'une base.

La commande **SELECT** requiert le droit de **SELECT** uniquement si des lignes sont lues dans une table. Vous pouvez exécuter une commande **SELECT** même sans aucun droit d'accès à une base de données dans le serveur. Par exemple, vous pourriez utiliser le client mysql comme une simple calculatrice :

mysql> SELECT 1+1 ;

mysql> SELECT PI()*2 ;

Le droit de **INDEX** vous donne le droit de créer et détruire des index de table.

Le droit de **ALTER** vous donne le droit de modifier une table avec la commande **ALTER TABLE**.Les droits de **CREATE** et **DROP** vous permettent de créer de nouvelles tables et bases de données, et de les supprimer.

Le droit de **GRANT** vous permet de donner les droits que vous possédez à d'autres utilisateurs.

Le droit de **FILE** vous donne la possibilité de lire et écrire des fichiers sur le serveur avec les commandes **LOAD DATA INFILE** et

SELECT ... INTO OUTFILE.

Les autres droits sont utilisés pour les opérations administratives qui sont exécutées par l'utilitaire **mysqladmin**. La table ci-dessous montre quelle commande est associée à mysqladmin avec un de ces droits :

Droit	Commande autorisée
RELOAD	reload, refresh, flush-privileges, flush-hosts, flush-logs et flush-tables
SHUTDOWN	shutdown
PROCESS	processlist
SUPER	kill

La commande **reload** indique au serveur de relire les tables de droits.

La commande **refresh** vide les tables de la mémoire, écrit les données et ferme le fichier de log. **flush-privileges** est un synonyme de **reload**. Les autres commandes **flush-*** effectuent des fonctions similaires à la commande **refresh** mais sont plus limitées dans leur application, et sont préférables dans certains contextes.

Par exemple, si vous souhaitez simplement vider les tampons dans le fichier de log, utilisez **flush-logs**, qui est un meilleur choix que **refresh**.

La commande **shutdown** éteint le serveur.

La commande **processlist** affiche les informations sur les threads qui s'exécutent sur le serveur. La commande **kill** termine un des threads du serveur. Vous pouvez toujours afficher et terminer vos propres threads, mais vous aurez besoin des droits de **PROCESS** pour afficher les threads, et le droit de SUPER pour terminer ceux qui ont été démarrés par d'autres utilisateurs.

C'est une bonne idée en général, de ne donner les droits de Grant qu'aux utilisateurs qui en ont besoin, et vous devriez être particulièrement vigilant pour donner certains droits :

- Le droit de **GRANT** permet aux utilisateurs de donner leurs droits à d'autres utilisateurs. Deux utilisateurs avec des droits différents et celui de **GRANT** pourront combiner leurs droits respectifs pour gagner un autre niveau d'utilisation du serveur.
- Le droit de **ALTER** peut être utilisé pour tromper le système en renommant les tables.
- Le droit de **FILE** peut servir à lire des fichiers accessibles à tous sur le serveur, et les placer dans une base de données. Le contenu pourra alors être lu et manipulé avec **SELECT**. Cela inclus le contenu de toutes les bases actuellement hébergées sur le serveur !
- Le droit de **SHUTDOWN** peut conduire au dénis de service, en arrêtant le serveur.
- Le droit de **PROCESS** permet de voir en texte clair les commandes qui s'exécutent actuellement, et notamment les changements de mot de passe.
- Le droit de **SUPER** peut être utilisé pour terminer les connexions ou modifier le mode opératoire du serveur.
- Les droits sur la base de données mysql peuvent être utilisés pour changer des mots de passe ou des droits dans la table des droits (Les mots de passe sont stockés chiffrés, ce qui évite que les intrus ne les lisent). S'ils accèdent à un mot de passe dans la table **mysql.user**, ils pourront l'utiliser pour se connecter au serveur avec cet utilisateur (avec des droits suffisants, le même utilisateur pourra alors remplacer un mot de passe par un autre).

Connexion au serveur MySQL

shell> mysql [-h nom_d_hote] [-u nom_d_utilisateur] [-pvotre_mot_de_passe]
shell> mysql [–host=nom_d_hote] [–user=nom_d_utilisateur] [–password=votre_mot_de_passe]
On peut spécifier les paramètres de connexion dans la section [client] du fichier de configuration my.cnf :
[client]

host=nom_d_hote
user=nom_d'utilisateur
password=votre_mot_de_passe
Contrôle d'accès, étape 1 : Vérification de la connexion

Votre identité est basée sur 3 informations :

- l'hôte depuis l'endroit où vous vous connectez
- Votre nom d'utilisateur MySQL
- Le mot de passe

La vérification est réalisée avec les 3 colonnes de la table user [host, user et password]. host peut être un nom d'hôte ou une ip. Pour les ip on peut spécifier le masque :(spécifier une plage)

GRANT ALL PRIVILEGES ON db.* TO david@'192.58.197.0/255.255.255.0' ;

Caractères utilisable pour host : % n'importe quel hôte, ou encore **x.y.%**

Pour connaître sous quel utilisateur on est connecté : **SELECT CURRENT_USER() ;**

Contrôle d'accès, étape 2 : Vérification de la requête

pour chaque requête, le serveur vérifie les droits. Ces droits peuvent provenir des tables user, db, tables_priv, column_priv. les droit sont vérifiés comme suit :

droits globaux
OR (droits de base AND droits d'hôte)
OR droits de table
OR droits de colonne

On devrait toujours tester les requêtes dans la table de droits, en utilisant l'utilitaire **mysqlaccess**.

Lors de la modification de privilèges, les droits **GRANT**, **REVOKE** et **PASSWORD** sont immédiatement pris en compte. Si on modifie les tables de droit manuellement avec **INSERT UPDATE** etc... on doit exécuter la commande **FLUSH PRIVILEGES** ou la commande **mysqladmin flush-privileges**, ou encore **mysqladmin reload**

syntaxe pour modifier le mot de passe :

SET PASSWORD FOR 'abe'@'host_name' = PASSWORD('eagle') ;

Gestion des comptes utilisateur

Les noms d'utilisateurs MySQL peuvent avoir jusqu'à 16 caractères. Il y'a 2 manières d'ajouter de nouveaux comptes :

- Utiliser la commande GRANT
- Manipuler la table des droits directement

exemples de syntaxe :

```
mysql> GRANT ALL PRIVILEGES ON generator.php TO 'monty'@'localhost'
-> IDENTIFIED BY 'un_mot_de_passe' WITH GRANT OPTION ;
mysql> GRANT ALL PRIVILEGES ON generator.php TO 'monty'@'%'
-> IDENTIFIED BY 'un_mot_de_passe' WITH GRANT OPTION ;
mysql> GRANT RELOAD,PROCESS ON generator.php TO 'admin'@'localhost' ;
```

```
mysql> GRANT USAGE ON generator.php TO 'dummy'@'localhost' ;
mysql> INSERT INTO user VALUES('localhost','monty',PASSWORD('un_mot_de_passe'),
-> 'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y') ;
mysql> INSERT INTO user VALUES('%','monty',PASSWORD('un_mot_de_passe'),
-> 'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y') ;
mysql> INSERT INTO user SET Host='localhost',User='admin',
-> Reload_priv='Y', Process_priv='Y' ;
mysql> INSERT INTO user (Host,User,Password)
-> VALUES('localhost','dummy','') ;
mysql> FLUSH PRIVILEGES ;
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
-> ON bankaccount.*
-> TO custom@localhost
-> IDENTIFIED BY 'stupid' ;
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
-> ON expenses.*
-> TO custom@whitehouse.gov
-> IDENTIFIED BY 'stupid' ;
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
-> ON customer.*
-> TO custom@'%'
-> IDENTIFIED BY 'stupid' ;
mysql> INSERT INTO user (Host,User,Password)
-> VALUES('localhost','custom',PASSWORD('stupid')) ;
mysql> INSERT INTO user (Host,User,Password)
-> VALUES('server.domain','custom',PASSWORD('stupid')) ;
mysql> INSERT INTO user (Host,User,Password)
-> VALUES('whitehouse.gov','custom',PASSWORD('stupid')) ;
mysql> INSERT INTO db
-> (Host,Db,User,Select_priv,Insert_priv,Update_priv,Delete_priv,
-> Create_priv,Drop_priv)
-> VALUES
-> ('localhost','bankaccount','custom','Y','Y','Y','Y','Y','Y') ;
mysql> INSERT INTO db
-> (Host,Db,User,Select_priv,Insert_priv,Update_priv,Delete_priv,
-> Create_priv,Drop_priv)
-> VALUES
-> ('whitehouse.gov','expenses','custom','Y','Y','Y','Y','Y','Y') ;
mysql> INSERT INTO db
-> (Host,Db,User,Select_priv,Insert_priv,Update_priv,Delete_priv,
-> Create_priv,Drop_priv)
-> VALUES('%','customer','custom','Y','Y','Y','Y','Y','Y') ;
mysql> FLUSH PRIVILEGES ;
```

Si vous voulez donner un accès spécifique à un utilisateur à partir de n'importe quelle machine d'un domaine donné, vous pouvez utiliser la commande GRANT, en utilisant '%' comme joker dans le nom de l'hôte

mysql> GRANT ...
-> ON generator.php
-> TO monutilisateur@"%.mondomaine.com"
-> IDENTIFIED BY 'monmotdepasse';

Pour faire la même chose en modifiant directement la table de droits, faites

mysql> INSERT INTO user VALUES ('%.mondomaine.com', 'monutilisateur',
-> PASSWORD('monmotdepasse'),...);
mysql> FLUSH PRIVILEGES;

Supprimer un compte utilisateur, utiliser la commande

DROP USER

Limiter les ressources utilisateur

- nombre de requêtes par heure
- nombre de modifications par heure
- Nombre de connexions réalisées par heures

Pour configurer et changer les limites d'un compte existant, utiliser la commande GRANT USAGE au niveau global avec ON *.*.
mysql> GRANT USAGE ON generator.php TO 'francis'@'localhost'
-> WITH MAX_QUERIES_PER_HOUR 100;; limite de requêtes à 100
mysql> GRANT USAGE ON generator.php TO 'francis'@'localhost'
-> WITH MAX_CONNECTIONS_PER_HOUR 0;; limite supprimée
Pour remettre à 0 les compteurs pour tous les comptes, faites
FLUSH USER_RESOURCES
puis
FLUSH PRIVILEGES

Configurer les mots de passe

les mots de passe peuvent être assignés avec mysqladmin.
mysqladmin -u user_name -h host_name password "newpasswd"
On peut utiliser SET PASSWORD
SET PASSWORD FRO 'user_name'@'%' = PASSWORD('newpasswd');
modifier son propre mot de passe
SET PASSWORD = PASSWORD('newpasswd');
On peut utiliser GRANT USAGE au niveau global ON generator.php
GRANT USAGE ON generator.php TO 'user_name'@'%' IDENTIFIED BY 'newpasswd';
on peut utiliser la table user directement. Pour établir un mot de passe à la création d'un compte
INSERT INTO user (Host,User,Password) VALUES('%','new_user','newpasswd');
FLUSH PRIVILEGES;
Pour changer le mot de passe d'un compte existant
UPDATE user SET Password = PASSWORD('newpasswd') WHERE Host = '%' AND User = 'user_name';
FLUSH PRIVILEGES;
En utilisant mysqladmin password ou GRANT .....IDENTIFIED BY la fonction PASSWORD() n'est pas nécessaire.
Il est possible de lister les mots de passe dans le groupe [client] de my.cnf. faire un chmod 400 /etc/mysql/my.cnf

Connexions sécurisées

MySQL n'utilise pas les connexions chiffrées par défaut, car cela ralentit considérablement le protocole de communication. Chiffrer les données est une tâche particulièrement coûteuse, qui peut ralentir considérablement les tâches principales de MySQL.

Pour utiliser les connexions sécurisées utiliser le script configure avec les options **–with-vio** et **–with-openssl**

Pour vérifier que le serveur supporte les connexions sécurisées utiliser la commande : **SHOW VARIABLES LIKE 'have_openssl'**

DIR=‘pwd‘/openssl
PRIV=$DIR/private
mkdir $DIR $PRIV $DIR/newcerts
cp /usr/share/ssl/openssl.cnf $DIR
replace ./demoCA $DIR – $DIR/openssl.cnf
Créez les dossiers nécessaires : $database, $serial et $new_certs_dir optionnel
touch $DIR/index.txt
echo "01" > $DIR/serial
Génération du cerificat d'autorité (CA)
openssl req -new -x509 -keyout $PRIV/cakey.pem -out $DIR/cacert.pem -config $DIR/openssl.cnf

Création des clé et requêtes serveur
openssl req -new -keyout $DIR/server-key.pem -out $DIR/server-req.pem -days 3600 -config $DIR/openssl.cnf
Supprimez la passe-phrase de la clé (optionnel)
openssl rsa -in $DIR/server-key.pem -out $DIR/server-key.pem
Signez le certificat serveur
openssl ca -policy policy_anything -out $DIR/server-cert.pem -config $DIR/openssl.cnf -infiles $DIR/server-req.pem
Créez les clé et requêtes client
openssl req -new -keyout $DIR/client-key.pem -out $DIR/client-req.pem -days 3600 -config $DIR/openssl.cnf
Supprimez la passe-phrase de la clé (optionnel)
openssl rsa -in $DIR/client-key.pem -out $DIR/client-key.pem
Signez le certificat client
openssl ca -policy policy_anything -out $DIR/client-cert.pem -config $DIR/openssl.cnf -infiles $DIR/client-req.pem
Créez le fichier my.cnf que vous pourrez utiliser pour tester les différents certificats
cnf=""
cnf="$cnf [client]"
cnf="$cnf ssl-ca=$DIR/cacert.pem"
cnf="$cnf ssl-cert=$DIR/client-cert.pem"
cnf="$cnf ssl-key=$DIR/client-key.pem"
cnf="$cnf [mysqld]"
cnf="$cnf ssl-ca=$DIR/cacert.pem"
cnf="$cnf ssl-cert=$DIR/server-cert.pem"
cnf="$cnf ssl-key=$DIR/server-key.pem"
echo $cnf | replace " " ’
’ > $DIR/my.cnf
To test MySQL
mysqld –defaults-file=$DIR/my.cnf &
mysql –defaults-file=$DIR/my.cnf
pour tester les connexions SSL, lancez le serveur comme ceci :
mysqld –default-file=/path/to/my.cnf &
puis lancer le programme client en utilisant le même fichier d’options :
mysql –defaults-file=/path/to/my.cnf

Options de GRANT avec SSL

MySQL peut vérifier les certificats X509 en plus de la combinaison habituelle de nom d’utilisateur et mot de passe.
Différentes possibilités pour limiter les connexions :

l’option REQUIRE SSL requière que les connexions soient chiffrées avec SSL

mysql> GRANT ALL PRIVILEGES ON test.* TO root@localhost
-> IDENTIFIED BY "goodsecret" REQUIRE SSL;

REQUIRE X509 impose au client d’avoir un certificat valide, mais le certificat lui même importe peu, la seul restriction est qu’il doit être possible de vérifier la signature avec une autorité de certification.

mysql> GRANT ALL PRIVILEGES ON test.* TO root@localhost
-> IDENTIFIED BY "goodsecret" REQUIRE X509;

REQUIRE ISSUER “issuer” restreint les tentatives de connexions : le client doit se présenter avec un certificat X509 valide, émit par l’autorité de certification “issuer”. Utiliser un certificat x509 implique obligatoirement des chiffrements, donc l’option SSL est sous entendue.

mysql> GRANT ALL PRIVILEGES ON test.* TO root@localhost
-> IDENTIFIED BY "goodsecret"
-> REQUIRE ISSUER "C=FI, ST=Some-State, L=Helsinki,
"> O=MySQL Finland AB, CN=Tonu Samuel/Email=tonu@mysql.com";

REQUIRE SUBJECT “subject” impose au client d’avoir un certificat X509 valide avec le sujet “subject”

mysql> GRANT ALL PRIVILEGES ON test.* TO root@localhost
-> IDENTIFIED BY "goodsecret"
-> REQUIRE SUBJECT "C=EE, ST=Some-State, L=Tallinn,
"> O=MySQL demo client certificate,
"> CN=Tonu Samuel/Email=tonu@mysql.com";

REQUIRE CIPHER “cipher” est utilisé pour s’assurer que les chiffrements sont suffisamment robuste, et que la bonne longueur de la clé

est utilisée.
mysql> GRANT ALL PRIVILEGES ON test.* TO root@localhost
-> IDENTIFIED BY "goodsecret"
-> REQUIRE CIPHER "EDH-RSA-DES-CBC3-SHA";
Les options SUBJECT, ISSUER et CIPHER peuvent être combinées :
mysql> GRANT ALL PRIVILEGES ON test.* TO root@localhost
-> IDENTIFIED BY "goodsecret"
-> REQUIRE SUBJECT "C=EE, ST=Some-State, L=Tallinn,
"> O=MySQL demo client certificate,
"> CN=Tonu Samuel/Email=tonu@mysql.com"
-> AND ISSUER "C=FI, ST=Some-State, L=Helsinki,
"> O=MySQL Finland AB, CN=Tonu Samuel/Email=tonu@mysql.com"
-> AND CIPHER "EDH-RSA-DES-CBC3-SHA";

Options SSL en ligne de commande

–ssl indique que le serveur autorise les connexions SSL, on doit spécifier également les options –ssl-ca ,–ssl-cert et –ssl-key

–ssl-ca=file_name chemin vers le fichier avec une liste des autorités de certifications SSL connus

–ssl-capath=directory_name le chemin où se trouve les certificats SSL au format PEM

–ssl-cipher=file_name nom du fichier de certificat SSL à utiliser pour établir une connexion sécurisée.

–ssl-cipher=cipher_list liste de chiffrements autorisées, à utiliser avec SSL a le même format que la commande openssl ciphers.

–ssl-key=file_name nom du fichier de la clé SSL à utiliser pour établir une connexion sécurisée.

Prévention des désastres et restauration

Pour sauvegarder des bases de données :

Faire un **LOCK TABLES** sur les tables concernées, suivi d'un **FLUSH TABLES** pour celles-ci. On a besoin que d'un verrou en lecture, cela permet aux autres threads de continuer à effectuer des requêtes sur les tables dont on fait la copie des fichiers. **FLUSH TABLES** est requis pour s'assurer que toutes les pages d'index actifs soient écrits sur le disque avant de commencer la sauvegarde.

Pour faire une sauvegarde d'une table SQL, il suffit d'utiliser **SELECT INTO OUTFILE** ou **BACKUP TABLE**. On peut utiliser aussi **mysqldump** ou le script **mysqlhotcopy**.

Effectuer une sauvegarde complète de la base de données :
mysqldump –tab=/path/to/backup –opt –all
ou
mysqlhotcopy base /path/to/backup
Arrêtez mysqld et le redémarrer avec l'option **–log-update[=nom-fichier]**. Les fichiers de log fournissent les informations dont vous avez besoin pour répliquer les modifications de la base de données qui sont subséquents au moment où vous avez utilisé mysqldump.

Si votre serveur MySQL est un esclave, quelque soit la sauvegarde que vous utilisez, lorsque vous sauvez vos données sur votre esclave, vous devez aussi sauver les fichiers **master.info** et **relay-log.info**, qui sont nécessaires pour relancer la réplication après la restauration des données de l'esclave. Si votre esclave doit traiter des commandes **LOAD DATA INFILE**, vous devez aussi sauver les fichiers nommés **SQL_LOAD-***, qui sont dans le dossier spécifié par **–slave-load-tmpdir**. Ce dossier vaut par défaut la valeur de la variable tmpdir, si elle n'est pas spécifiée. L'esclave aura besoin de ces fichiers pour relancer la réplication d'une opération **LOAD DATA INFILE** interrompue.

Si vous avez besoin de restaurer quelque chose, essayez d'abord de restaurer vos tables avec **REPAIR TABLE** ou **myisamchk -r** en premier. Cela devrait fonctionner dans 99.9% des cas. Si myisamchk ne réussi pas, essayez la procédure suivante (cela ne fonctionnera que si vous avez démarré MySQL avec **–log-update** :

Restaurez la sauvegarde originale de mysqldump.

Exécutez la commande suivante pour remettre en marche les mises à jour dans le log binaire :
shell> mysqlbinlog hostname-bin.[0-9]* | mysql

Politique de sauvegarde

Faire une sauvegarde de toutes les tables InnoDB dans toutes les bases :

shell> mysqldump –single-transaction –all-databases > backup_sunday_1_PM.sql

Le fichier .sql résultant, produit par mysqldump contient les commandes **SQL INSERT** qui peuvent être utilisées pour recharger les tables ultérieurement.

Pour effectuer des sauvegarde incrémentales, le serveur doit être lancé avec l'option **–log-bin** pour qu'il puisse stocker ces modifications au fur et à mesure des modifications des données. Cette option active le log binaire, ce qui fait que chaque commande qui modifie les données est enregistré dans un fichier appelé log binaire. A chaque fois que le serveur redémarre, MySQL créer un nouveau fichier de log binaire, en utilisant le numéro de séquence suivant.

Lorsque le serveur fonctionne on peut aussi lui dire de clore le fichier log et d'en ouvrir un nouveau avec la commande **SQL FLUSH LOGS** ou bien avec **mysqladmin flush-logs**. La commande **mysqldump** dispose aussi d'une option pour clore les fichiers de logs. Le fichier **.index** contient la liste de tous les fichiers de logs binaire du dossier de données. Ce fichier est utilisé durant les opérations de réplication.

utiliser mysqldump pour qu'il referme les logs binaires lors de la sauvegarde complète, et que le fichier de sauvegarde contienne les noms des nouveaux fichiers de logs
shell> mysqldump –single-transaction –flush-logs –master-data=2 –all-databases > blackup_sunday_1_PM.sql
pour supprimer les logs anciens, par exemple ceux qui sont antérieurs à la sauvegarde complète :
shell> mysqldump –single-transaction –flush-logs –master-data=2 –all-databases –delete-master-logs > backup_sunday_1_PM.sql
Noter que la commande est dangereuse si le serveur est un maitre réplicateur car les esclaves pourraient ne pas avoir traité le contenu des logs binaires. La description de la commande **PURGE MASTER LOGS** explique ce qui doit être vérifié avant d'effacer un fichier de log binaire.

Utiliser les sauvegardes pour la restauration :
shell> mysql < blachup_sunday_1_PM.sql
pour utiliser les sauvegardes incrémentales :
shell> mysqlbinlog mysqld-bin.000007 mysqld-bin.000008 | mysql
Il reste la dernière sauvegarde, qui est le dernier fichier de log binaire. de préférence spécifier un chemin vers un volume sécurisé pour l'option –log-bin.

Fichiers de log de MySQL

Le log d'erreurs Problèmes rencontrés lors du démarrage, de l'exécution ou de l'arrêt de mysqld.

Le log ISAM Garde une trace des changements liés au tables ISAM. Utilisé uniquement pour déboguer le code ISAM.

Le log de requêtes Connexions établies et requêtes exécutées.

Le log de mises à jour Désapprouvé : Enregistre toutes les commandes qui changent les données.

Le log binaire Enregistre toutes les commandes qui changent quelque chose. Utilisé pour la réplication.

Le log des requêtes lentes Enregistre toutes les requêtes qui ont pris plus de long_query_time à s'exécuter ou celles qui n'ont pas utilisé d'index.

Par défaut, tous les fichiers de log peuvent être trouvés dans le dossier de données de mysqld.On peut forcer mysqld à rouvrir les fichiers de log ou à passer à un nouveau log en exécutant FLUSH LOGS.

Le log d'erreurs : Contiennent les informations indiquant quand mysqld a été lancé ou arrêté, ainsi que les erreurs critiques. si mysqld s'arrête inopinément -> restarted mysqld. Si mysqld remarque une table à réparer/analyser, il l'écrit également.

Pour spécifier l'emplacement du log d'erreur : **–log-error[=file_name]**. Sans cette option, écrit sur la sortie standard.

Pour démarrer le log général de requêtes : **–log[=fichier]** (par défaut nommé 'hostname'.log)

Pour démarrer le log des requêtes lentes : **–log-slow-queries[=file_name]**. Toutes les requêtes plus longues que **long_query_time** secondes à s'exécuter. Le temps d'acquisition d'un verrou n'est par compté. Pour obtenir un sommaire des requêtes lentes, utiliser mysqldumpslow.

Entretient des fichier de log

mysql-log-rotate permet de gérer les logs. attention au logs pour la réplication.Pour forcer Mysql à utiliser de nouveaux fichiers de log : **mysqladmin flush-logs** ou **FLUSH-LOGS**.

flush-logs permet de fermer les logs puis les rouvre, donc il faut les déplacer avant.

Multiples serveurs sur une seule machine

au minimum, les options suivantes doivent être différentes sur chaque serveur :

–port=port_num port d'écoute TCP

–socket=path chemin du socket

–pid-file=path chemin du fichier PID

Les options suivantes, si utilisées, doivent être différentes pour chaque serveur :

–log=path

–log-bin=path

–log-update=path

–log-error=path

–log-isam=path

–bdb-logdir=path

Pour de meilleurs performances, on pour également spécifier les options suivantes, pour répartir la charge entre plusieurs disques physiques

–tmpdir=path

–bdb-tmpdir=path

Également, le dossier de données :

–datadir=path

En cas de plusieurs installations de MySQL à différents endroits, utiliser :

–base-dir=path

Pour spécifier de fichier d'option à utiliser :

–defauts-file

Pour lancer un serveur avec son fichier d'option
shell> mysqld –defaults-file=C :\my-opts1.cnf
pour l'éteindre

shell> mysqladmin –port=3307 shutdown
On peut compiler mysql avec différents port et socket pour chacun avec
./configure –with-tcp-port=port_number –with-unix-socket-path=file_name –prefix=/usr/local/mysql-4.0.17
Connaître les caractéristiques d'un serveur
mysqladmin –host=host_name –port=port_number variables
Pour lancer un serveur sans l'avoir compilé avec des valeurs par défaut
/path/to/mysqld_safe –socket=file_name –port=port_number
Pour utiliser un dossier de données différent, utiliser l'option
–datadir=path à mysqld_safe
Idem, en utilisant les variables d'environnement pour spécifier le nom du socket et le port
shell> MYSQL_UNIX_PORT=/tmp/mysqld-new.sock
shell> MYSQL_TCP_PORT=3307
shell> export MYSQL_UNIX_PORT MYSQL_TCP_PORT
shell> scripts/mysql_install_db
shell> bin/mysqld_safe &