MySQL, administration, niveau 1

Dr Ing Pierre-Emmanuel Gros
pierre-emmanuel.gros@neuresys.fr

Objectifs Pédagogiques

- Décomposer l'architecture de la base de données MySQL
- Installer le SGBD MySQL
- Déterminer les principales fonctionnalités de l'administration d'une BDD MySQL
- Créer et gérer une base de données
- Gérer l'accès aux utilisateurs
- Gérer la sécurité de la base de données

Sommaire

- Introduction et installation
- Objets d'une base MySQL
- Connexions, droits d'accès, sécurité
- Moteurs de stockage et plug-ins
- Maintenance d'un serveur MySQL
- Optimisation

Introduction et installation

- Introduction et installation
- Versions et types de licences.
- Outils MySQL, mysqlshow, myphpadmin.
- Paramétrage du serveur (my.cnf, sql_mode, mode strict...).
- Démarrage/arrêt.
- Installations (binaire, à partir des sources...).
- MysqlAdmin.
- Installation de MySQL Workbench.
- Changements de versions, migration

Introduction et installation
Historique.

- Vague notion d'historique
- Présentation des versions
- Le moteur Maria DB est présenté

Historique

- MySQL est un système de gestion de base de données relationnelle (SGBDR) à source ouverte.
- Comme toutes les autres bases de données relationnelles, MySQL utilise des tables, des contraintes, des déclencheurs, des rôles, des procédures stockées et des vues comme composants avec lesquels vous travaillez.
- MySQL est écrit en C et C ++ et comporte des fichiers binaires pour les systèmes suivants: Microsoft Windows, OS X, Linux, AIX, BSDi, FreeBSD, HP-UX, IRIX, NetBSD, Novell Netware, etc.
- MySQL a été créé par MySQL AB en 1994 par une société suédoise créée par David Axmark, Allan Larsson et Michael «Monty» Widenius. La première version de MySQL est sortie en 1995. En 2008, Sun Microsystems a acheté MySQL AB. En 2010, Sun Microsystems a été acquis par Oracle.
- MySQL est actuellement maintenu par Oracle Corporation.

Versions Et Types de Licences

- MySQL est passé en licence GPL à partir de la version 3.23.19 (juin 2000)
- Version 4.0 : première version en octobre 2001, stable depuis mars 2003
- Version 4.1 : première version en avril 2003, stable depuis octobre 2004
- Version 5.0 : première version en décembre 2003, stable en octobre 2005
- Version 5.1 : première version en novembre 2005, Release Candidate en septembre 2007
- Version 5.2 : distribuée en avant-première (ajout du nouveau moteur de stockage Falcon) en février 2007, cette ligne a ensuite été renommée 6.0
- Version 5.5 : Version stable depuis octobre 2010
- Version 5.6 : Version stable depuis février 2013
- Version 5.7 : Version stable depuis octobre 2015
- Version 6.0 : première version alpha en avril 2007, abandonnée depuis le rachat de MySQL par oracle en décembre 2010
- Version 8.0 : Version stable depuis avril 2018

MariaDB

- En 2009, à la suite du rachat de MySQL par Sun Microsystems et des annonces du rachat de Sun Microsystems par Oracle Corporation, Michael Widenius, fondateur de MySQL, quitte cette société10 pour lancer le projet MariaDB, dans une démarche visant à remplacer MySQL tout en assurant l'interopérabilité.
- Il s'agit d'un fork communautaire de MySQL
- L'idée est d'avoir des binaires compatibles, des outils identiques ainsi qu'une compatilité SQL et données binaires

MariaDB vs MySQL

- La totalité des différences sont https://mariadb.com/kb/en/library/mariadb-vs-mysql-compatibility/
- Principalement, MariaDB a plus de fonctionnalités mais pas le support Oracle.
- MySQL Cluster est remplacé par Gallera Cluster

Introduction et installation
Installation

- Répertoire d'une distribution MySQL
- Installation depuis le binaire
- Installation depuis les sources

Type d'installation

- Oracle fournit un ensemble de distribution binaire pour la plus part des plateformes (Windows, Unix), mais aussi une version a compiler.
- Lors de cette installation, une liste de répertoire va être mise en place :
  - Bin : mysqld server et client
  - Docs : une documentation de MySQL
  - Man : une documentation Unix
  - Include/Lib : Fichier de developpement
  - Share: fichier support de MySQL
  - Support-file : fichier utilitaire MySQL

Installation de MySQL

- Le binaire mysql-debug est la version de MySQL compiler avec les symboles de deboggage MySQL.
- Il faut créer un utilisateur dédié sous Unix pour executer le binaire MySQL et inialise le demon MySQL
  - shell> groupadd mysql
  - shell> useradd -r -g mysql -s /bin/false mysql
  - Shell> chown -R mysql:mysql .
  - shell> bin/mysqld --initialize --user=mysql
  - shell> bin/mysql_ssl_rsa_setup
  - shell> bin/mysqld_safe --user=mysql &
- Le serveur doit demarrer sur le port 3306 et/ou un socket Unix

Installation détaillé sous Unix

- Pour installer MySQL en binaire pour un serveur Linux sous Débian,
- Il faut en premier lieu potentiellement télécharger l'archive pour apt
  - Wget mysql-apt-config…_all.deb
  - Sudo dpkg –i mysql-apt-config….._all.deb

- Un mode graphique se présente afin de savoir si on veut installer le serveur, les utilitaires …
- Cette sélection va enrichir les dépôts avec des liens vers MySQL.

Installation détaillé sous Unix

- Ensuite il faut mettre a jour les paquets depuis le dépôt de MySQL
  - Apt-get update
- Puis initier l'install du serveur MySQL
  - Apt-get install mysql-server-X
- Le mot de passe de l'admin est demandé pour la suite de l'install.
- En fin d'installation, le service est normalement démarré et est opérationnel:
  - Mysql –u root -p

Installation depuis les sources

- Lors de l'installation par les sources de MySQL, certaines options doivent être positionnées lors de la compilation.
- En particulier les option de pondération pour la recherche plaintext (modification des sources).

- L'installation par les sources à du sens :
  - Pour maitriser l'installation du produit
  - Pour éviter d'utiliser le mot de passe root pour l'installation
- Afin de faire une installation par les sources, il faut télécharger les sources de MySQL, et faire un apt-get update, apt-get upgrade

Installation depuis les sources

- Pour compiler MySQL, il faut les paquets suivant:
  - Gcc, g++
  - Git
  - Make,cmake,cmake-gui
  - Libncurses5-dev ou ncurses-devel
  - Bison/flex
- Soit les sources sont téléchargé tel quel, soit via git:
  - Git clone https://githubcom/mysql/mysql-server.git
  - Cd mysql-server
  - Git branch pour verifier le branche
  - Git checkout <*branch*>

Installation par les sources

- Pour une installation de MySQL depuis le code source, il faut créer l'utilisateur mysql
  - Groupadd mysql
  - Useadd –r –g mysql mysql

- Sous Linux, il faut pour compiler demander le support de boost et du support de l'atomicité de gcc
  - Cmake . –DDOWNLOAD_BOOST=1 –DWITH_BOOST=/tmp –DENABLE_DOWNLOADS=1 -DHAVE_GCC_ATOMIC_BUILTINS=1
  - Make, make install DESTDIR=« repertoire d'installation »

Installation par les sources

- Il faut maintenant assigner les bon droits au repertoire d'installation
  - Chown –R mysql:mysql /usr/local/mysql
- Puis on demande à MySqld (depuis 5.7) l'initialisation du repertoire de données:
  - Mysqld –initialize –datadir=/repertoire de données
- Dans le datadir, dans le fichier erreur on retrouve le password généré pour root
  - 2018-11-30T17:05:16.812942Z 5 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: eA6wt,yg9teT

Initialisation depuis les sources

- Créer un fichier my.cnf minimal et demarré mysqld avec l'option defaults-file
- Enfin essayer de se connecter en mode root en utilisant le mots de passe récupérer précédemment.

```
[mysqld]
# set basedir to your installation path
basedir=D:/MySQL/mysql-8.0.13-winx64
# set datadir to the location of your data directory
datadir=D:/MySQL/mysql-8.0.13-winx64/data

Démarrer le serveur
d:\MySQL\mysql-8.0.13-winx64\bin>mysqld --defaults-
file=d:\MySQL\mysql-8.0.13-wi
nx64\my.cnf
```

Initilisation par les sources

- Enfin, il faut changer le mot de passe de l'utilisateur root

```
alter user 'root'@'localhost' identified by 'eA6wt,yg9teT'
replace `
```

Licence MySQL

- Présentation des différents types de licences
- Avoir un ordre d'idée du prix des différentes licences

Historique des licences

En 2008, Sun Microsystems a acquis MySQL en tant que tremplin pour entrer sur le marché des bases de données.

En 2010, Oracle Corporation a acquis Sun Microsystems afin d'enrichir son portefeuille en matériel, Java et d'intégrer la Marché MySQL.

Aujourd'hui, les logiciels MySQL sont tous deux disponibles via l'open source (ou licence publique générale).

Dans le cadre du modèle de licence commerciale, Oracle permet aux éditeurs de logiciels indépendants (ISV) et aux revendeurs à valeur ajoutée (VAR) de distribuer les Logiciels MySQL intégrés à leurs propres applications / solutions logicielles et aux utilisateurs finaux pour obtenir une licence d'abonnement pour utiliser MySQL pour leurs opérations commerciales internes.

Licence GPL

Le déploiement et l'utilisation des logiciels MySQL sous le modèle de licence open source ou GPL signifie que l'utilisateur est libre d'utiliser les programmes, de les modifier et de les redistribuer avec ou sans changement.

Le code source du logiciel est ouvert et peut être ajusté et / ou modifié jusqu'à les goûts de l'utilisateur final.

Toute modification apportée au code source doit cependant être faite publiquement disponible pour tous les autres utilisateurs finaux des programmes logiciels open source MySQL.

Licence Commercial

- Néanmoins, les organisations d'utilisateurs finaux qui ne souhaitent pas partager le code source de leurs solutions reposant sur MySQL ou les utilisateurs finaux qui ne veulent pas compter sur le support de la communauté veulent généralement obtenir une licence d'abonnement commercial pour les logiciels MySQL.

Licence et Clients

- Différents types de licence du logiciel commercial MySQL les programmes sont disponibles, y compris:
  - Licences d'abonnement
  - Licences ASFU (Application Specific Full Use)
  - Licence de logiciel embarqué (ESL)
- Sont concerné
  - les Fabricants d'équipement d'origine (OEM),
  - éditeurs de logiciels indépendants (ISV),
  - revendeurs à valeur ajoutée (VAR) et d'autres distributeurs qui combinent et distribuent des logiciels sous licence commerciale avec MySQL logiciels
- Ces personnes ne souhaitent généralement pas distribuer le code source de leurs produits et souhaitent fournir leur propre support technique

Licences d'abonnement

- Sous ce modèle de licence, les organisations d'utilisateurs finaux peuvent obtenir une version commerciale du logiciel MySQL. Ces licences peuvent uniquement être obtenues auprès d'Oracle directement ou auprès d'un revendeur Oracle agréé.
- La licence est basée sur un modèle d'abonnement dans lequel les logiciels MySQL ne peuvent être utilisés que pour les opérations commerciales internes de l'organisation en conjonction avec toute application.
- Si et quand le la durée de l'abonnement expire, les utilisateurs finaux doivent soit renouveler leur abonnement, soit désinstaller le logiciel MySQL.

Licences ASFU (Application Specific Full Use)

- Sous ce modèle de licence, une organisation d'utilisateur final peut obtenir une version commerciale de MySQL complète.
- Ces licences ne peuvent être obtenues que par un partenaire Oracle ayant développé ses propres application commerciale sur une plate-forme de base de données MySQL et est entré dans en contrat afin de distribuer le logiciel MySQL en en conjonction avec l'application spécifique.
- La licence est généralement basée sur un modèle perpétuel dans lequel les logiciels MySQL ne peuvent être utilisé pour les opérations commerciales internes de l'organisation en conjonction avec l'application spécifique.
- L'utilisation du logiciel MySQL pour toute autre application n'est pas autorisée.

Licence de logiciel embarqué (ESL)

- Sous ce modèle de licence, les organisations d'utilisateurs finaux ne savent souvent même pas que le logiciel MySQL est utilisé dans le produit car fournit dans une solution indépedante.
- L'utilisateur final n'a pas (le droit de) le logiciel MySQL directement.
- Tout accès au logiciel MySQL doit obligatoirement être effectué via le appareil / solution elle-même.
- Les licences de logiciel embarqué sont généralement vendues à perpétuité.

Différentes Editions

MySQL est présent dans les éditions suivantes:

- MySQL Classic Edition
- MySQL Enterprise Edition
- MySQL Standard Edition
- MySQL Cluster
- MySQL Embedded *
  - Moteur de base de données complète, sans administration, à destination des fournisseurs de logiciels, le matériel et le logiciel SaaS
- MySQL Community Edition (GPL) *

MySQL Classic et Standard

- MySQL Classic Edition inclut les fonctionnalités suivantes:
  - Serveur de base de données MySQL
  - Connecteurs MySQL
  - Réplication MySQL
  - Moteur de stockage: MyISAM
- MySQL Standard Edition comprend les fonctionnalités suivantes:
  - Serveur de base de données MySQL
  - Connecteurs MySQL
  - Réplication MySQL
  - MySQL Workbench
  - Moteur de stockage: MyISAM
  - Moteur de stockage: InnoDB

MySQL Enterprise

MySQL Enterprise Edition inclut les fonctionnalités suivantes:

- Serveur de base de données MySQL
- Moteurs de stockage MySQL (InnoDB, MyISAM, etc.)
- Connecteurs MySQL (JDBC, ODBC, .Net, etc.)
- Réplication MySQL – Fabric MySQL
- Partitionnement MySQL
- Utilitaires MySQL
- MySQL Workbench
- Sauvegarde MySQL Enterprise
- MySQL Enterprise Monitor
- MySQL Enterprise HA
- Evolutivité de MySQL Enterprise
- Sécurité MySQL Enterprise
- Audit d'entreprise MySQL

MySQL Cluster

- MySQL Cluster CGE (Carrier Grade Edition) comprend des outils de gestion, de surveillance de la sécurité et de audit de la base de données MySQL Cluster, associé à l'accès au support Oracle Premier. Il comprend le Fonctionnalités suivantes:
  - Cluster MySQL
  - Gestionnaire de cluster MySQL
  - Sauvegarde MySQL Enterprise
  - MySQL Enterprise Monitor - MySQL Workbench Standard Edition

MySQL Community Edition

MySQL Community Edition est la version librement téléchargeable, disponible sous l'open source General. Public License (GPL) et est pris en charge par une large communauté de développeurs open source.

- Le MySQL Community Edition inclut une architecture de moteur de stockage enfichable et plusieurs moteurs de stockage, tels que:
  - InnoDB, MyISAM
  - **NDB (cluster MySQL) (il est possible d'installer MySQL Cluster GPL)**
  - Mémoire
  - Federated
  - archive , CSV
- Il comprend également les fonctionnalités suivantes:
  - Serveur de base de données MySQL
  - Connecteurs MySQL (JDBC, ODBC, .Net, etc.)
  - Réplication MySQL
  - Fabric MySQL
  - Partitionnement MySQL
  - Utilitaires MySQL
  - MySQL Workbench

Edition de MySQL

	Classic	Standard	Enterprise	Cluster CGE
MySQL Database Server	X	X	X	X
MySQL Connectors	X	X	X	X
MySQL Replication	X	X	X	X
MySQL Router			X	X
MySQL Partitioning			X	X
MySQL Utilities			X	X
MySQL Workbench			X	X
Storage Engine: MyISAM	X	X	X	X
Storage Engine: InnoDB		X	X	X
Storage Engine: NDB				X
MySQL Entreprise Tools			X	X
MySQL Cluster Tools				X
Certification Oracle		X	X	X

Tarif (Ordre d'idée)

- Les prix mouvants peuvent être obtenue via Oracle

MySQL Perpetual License	Prix	Support	Métrique
MySQL Classic Edition (1-4 socket server)	$1,800	$396.00	Serveur
MySQL Classic Edition (5+ socket server)	$3,600	$792.00	Serveur
MySQL Standard Edition (1-4 socket server)	$6,000	$1,320.00	Serveur
MySQL Standard Edition (5+ socket server)	$12,000	$2,640.00	Serveur
MySQL Enterprise Edition (1-4 socket server)	$15,000	$3,300.00	Serveur
MySQL Enterprise Edition (5+ socket server)	$30,000	$6,600.00	Serveur
MySQL Cluster Carrier Grade Edition (1-4 socket server)	$30,000	$6,600.00	Serveur
MySQL Cluster Carrier Grade Edition (5+ socket server)	$60,000	$13,200.00	Serveur

Feature SQL MySQL version GPL

- SQL et NoSQL pour développer des applications relationnelles et NoSQL
- MySQL Document Store, y compris le protocole X, l'API XDev et MySQL Shell
- Dictionnaire de données transactionnel avec instructions Atomic DDL pour une fiabilité améliorée
- Architecture de moteur de stockage plugable (InnoDB, NDB, MyISAM, etc.)
- Réplication MySQL pour améliorer les performances et l'évolutivité des applications
- Réplication de groupe MySQL pour la réplication de données tout en offrant une tolérance aux pannes, un basculement automatique et de l'élasticité
- MySQL InnoDB Cluster fournira une solution intégrée, native et à haute disponibilité pour MySQL
- MySQL Router pour un routage transparent entre votre application et tous les serveurs MySQL dorsaux
- Le partitionnement MySQL pour améliorer les performances et la gestion des applications de bases de données volumineuses
- Procédures stockées pour améliorer la productivité des développeurs
- Déclencheurs pour appliquer des règles métier complexes au niveau de la base de données
- Vues pour s'assurer que les informations sensibles ne sont pas compromises
- Schéma de performances pour la surveillance de la consommation de ressources au niveau utilisateur / application Schéma d'information permettant un accès facile aux métadonnées
- Connecteurs MySQL (ODBC, JDBC, .NET, etc.) pour la construction d'applications dans plusieurs langues
- MySQL Workbench pour la modélisation visuelle, le développement SQL et l'administration

Savoir la licence

- Pour savoir si vous déployez la version open source du logiciel MySQL ou si vous utilisez la version commerciale du logiciel MySQL exécutez la commande suivante:

SELECT @@ license

- Le résultat de cette requête fournira une sortie qui indiquera si l'instance MySQL est l'édition (appelée GPL - Licence publique générale) ou si l'instance MySQL est une instance commerciale édition.

Introduction et installation
Outils mysql, mysqlshow, phpMyAdmin

- Présentation de l'outil CLI de base Mysql
- Présentation de l'outil mysqlshow
- Intallation de phpMyAdmin

mysql

- mysql est un simple shell SQL avec des capacités d'édition de lignes d'entrée. Il prend en charge l'utilisation interactive et non interactive.
- Lorsqu'ils sont utilisés de manière interactive, les résultats de la requête sont présentés dans un format de tableau ASCII. Lorsqu'il est utilisé de manière non interactive (par exemple, en tant que filtre), le résultat est présenté dans un format séparé par des tabulations.
- Le format de sortie peut être modifié à l'aide des options de commande.

mysql

- Invoquez-le à l'invite de votre interpréteur de commande comme suit:
  - shell> mysql *db_name*
  - shell> mysql --user=*user_name* --password *db_name*
- En tapant Ctrl + C, mysql tente de tuer l'instruction en cours. Si cela ne peut pas être fait, ou si Control + C est tapé à nouveau avant que l'instruction soit tuée, mysql se ferme.
- Vous pouvez exécuter des instructions SQL dans un fichier de script (fichier de commandes) comme ceci:
- shell> mysql *db_name* < *script.sql* > *output.tab*

mysql

- Sous Unix, le client mysql enregistre les instructions exécutées de manière interactive dans un fichier d'historique. Par défaut, ce fichier s'appelle .mysql_history dans votre répertoire personnel. Pour spécifier un autre fichier, définissez la valeur de la variable d'environnement MYSQL_HISTFILE.
- Le fichier .mysql_history doit être protégé avec un mode d'accès restrictif, car des informations sensibles peuvent y être écrites, tel que le texte d'instructions SQL contenant des mots de passe.

mysql

- Le client mysql est généralement utilisé de manière interactive, comme ceci:
  - shell> mysql *db_name*
- Cependant, il est également possible de placer vos instructions SQL dans un fichier, puis de dire à mysql de lire ses entrées à partir de ce fichier.
  - shell> mysql *db_name* < *text_file*
- Si vous utilisez déjà mysql, vous pouvez exécuter un fichier de script SQL en utilisant la commande source ou \. commander:
  - mysql> source *file_name*

MySQLShow

- Le client mysqlshow peut être utilisé pour voir rapidement quelles bases de données existent, leurs tables ou les colonnes ou index d'une table.
- mysqlshow fournit une interface de ligne de commande à plusieurs instructions SQL SHOW (commande de base des métadata MySQL)

```
shell> mysqlshow [options] [db_name [tbl_name [col_name]]]
```

MySQLShow

Si aucune base de données n'est fournie, une liste de noms de bases de données est affichée.

- Si aucune table n'est donnée, toutes les tables correspondantes de la base de données sont affichées.
- Si aucune colonne n'est indiquée, toutes les colonnes et tous les types de colonnes correspondants du tableau sont affichés.
- La sortie affiche uniquement les noms des bases de données, tables ou colonnes pour lesquelles vous disposez de certains privilèges.

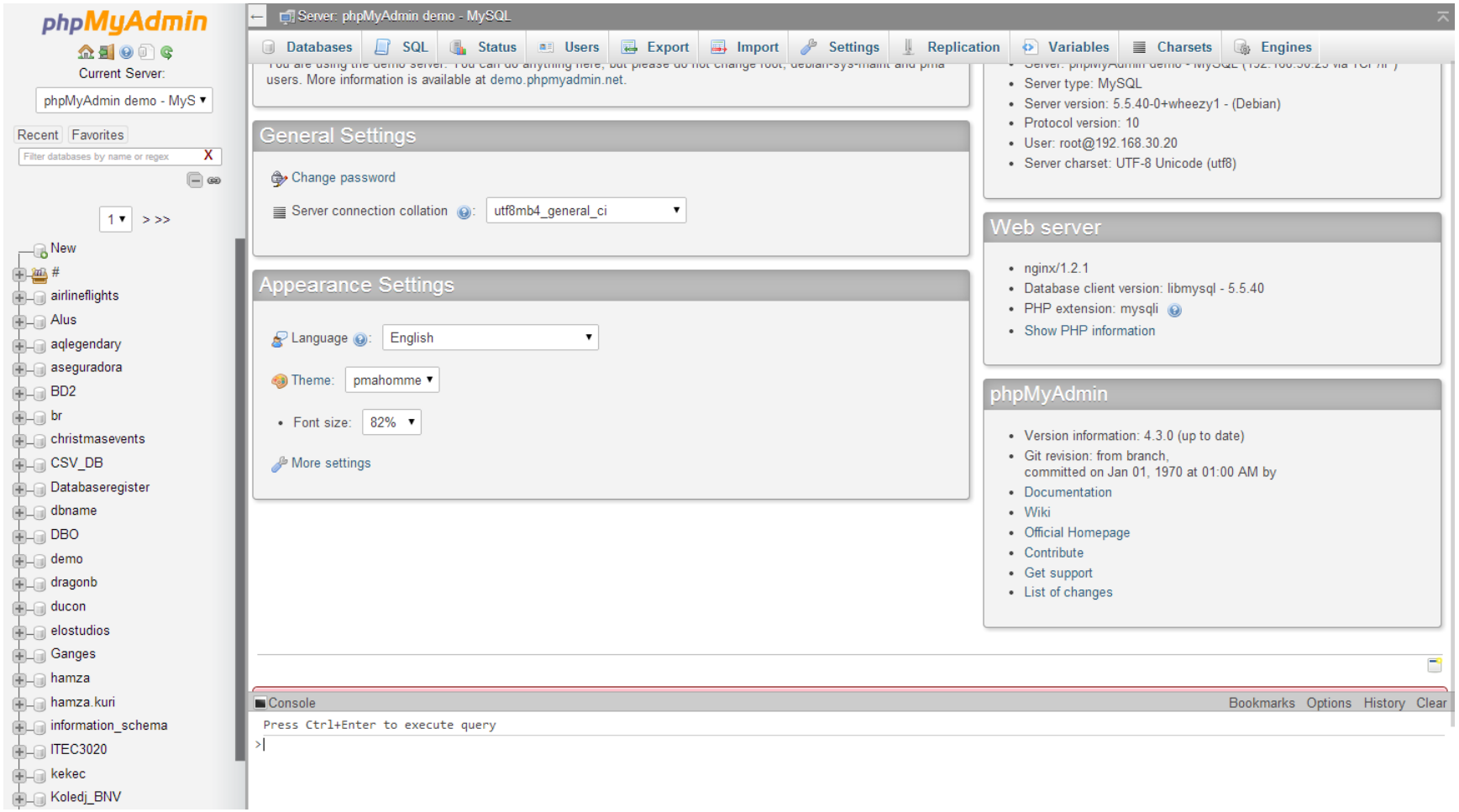
Si le dernier argument contient des caractères génériques de shell ou SQL (*,?,% Ou _), seuls les noms correspondants par le caractère générique sont affichés.

PhpMyAdmin

- Il s'agit de l'une des plus célèbres interfaces pour gérer une base de données MySQL sur un serveur PHP. De nombreux hébergeurs, gratuits comme payants, le proposent ce qui évite à l'utilisateur d'avoir à l'installer.

- Cette interface pratique permet d'exécuter, très facilement et sans grandes connaissances en bases de données, des requêtes comme les créations de table de données, insertions, mises à jour, suppressions et modifications de structure de la base de données, ainsi que l'attribution et la révocation de droits et l'import/export. Ce système permet de sauvegarder commodément une base de données sous forme de fichier .sql et d'y transférer ses données, même sans connaître SQL.

PhpMyAdmin


phpMyAdmin
Current Server:
phpMyAdmin demo - MyS
Recent
Favorites
Filter databases by name or regex
New
#
airlineflights
Alus
aqlegendary
aseguradora
BD2
br
christmasevents
CSV_DB
Databaseregister
dbname
DBO
demo
dragonb
ducon
elostudios
Ganges
hamza
hamza.kuri
information_schema
ITEC3020
kekec
Koledj_BNV
Server: phpMyAdmin demo - MySQL
Databases
SQL
Status
Users
Export
Import
Settings
Replication
Variables
Charsets
Engines
users. More information is available at demo.phpmyadmin.net.
General Settings
Change password
Server connection collation:
utf8mb4_general_ci
Appearance Settings
Language:
English
Theme:
pmahomme
Font size:
82%
More settings
Server type: MySQL
Server version: 5.5.40-0+wheezy1 - (Debian)
Protocol version: 10
User: root@192.168.30.20
Server charset: UTF-8 Unicode (utf8)
Web server
nginx/1.2.1
Database client version: libmysql - 5.5.40
PHP extension: mysqli
Show PHP information
phpMyAdmin
Version information: 4.3.0 (up to date)
Git revision: from branch,
committed on Jan 01, 1970 at 01:00 AM by
Documentation
Wiki
Official Homepage
Contribute
Get support
List of changes
Console
Bookmarks
Options
History
Clear
Press Ctrl+Enter to execute query

Installation PhpMyAdmin

1. Installation d'une Stack LAMP - (Linux, Apache, MariaDB / MySQL, PHP)
   phpMyAdmin nécessite une pile LAMP (Linux, Apache, MariaDB / MySQL, PHP). Si vous ne l'avez pas déjà installé sur votre serveur, vous pouvez le faire en exécutant les commandes suivantes

```
apt-get install apache2 php mysql-server
```

2. Installation de phpMyAdmin
   Une fois Apache, PHP et MySQL installés, vous pouvez installer phpMyAdmin. Le paquet est inclus dans le référentiel officiel Ubuntu 16.04 et peut être facilement installé avec la commande ci-dessous

```
apt-get install phpmyadmin
```

Installation PhpMyAdmin

3. Définir le mot de passe pour l'utilisateur phpMyAdmin

Il s'agit du mot de passe l'utilisateur phpMyAdmin dans MySQL qui va autoriser l'outil a se connecter, browser, modifier les tables (et donc ce n'est pas l'utilisateur root de mysql)

4. Testez et vérifiez l'installation de phpMyAdmin sur Ubuntu 16.04

You can access it at http://yourIPaddress/phpmyadmin and log in with your MySQL username and password.

Introduction et installation
Paramétrage du serveur (my.cnf, sql_mode, mode strict...).

- Début de configuration d'un serveur MySQL.
- Découverte du fichier my.cnf
- Modification des paramètres sql_mode et mode_strict

My.ini

Le fichier my.cnf ou my.ini est le fichier de configuration du serveur MySQL.

Les programmes fournis par MySQL (mysqld, mysql, mysqldump) viennent y chercher leurs directives.

- Sous Linux, le fichier my.cnf est recherché dans les repertoires /etc/, /etc/mysql/ $MYSQL_HOME
- Sous windows, le fichier seras cherché dans WINDIR/my.ini, c:\my.ini,INSTALLDIR\my.ini

- Si plusieurs fichiers my.cnf existent, les options qu'ils contiennent seront lues dans cet ordre, et toutes prise en compte.
- En cas de redefinition, c'est la dernière occurrence qui l'apporte.

My.ini

Si le fichier ne se trouve pas dans l'un de ces répertoires ou s'il ne se nomme pas my.cnf/my.ini, il faut l'indiquer aux programmes.

Dans ce cas, l'option defaults-file qui permet d'indiquer au programme l'emplacement du fichier de configuration.

- Mysqld –defaults-file=/user/mysql/conf/mysql_server.cnf

Démarrage de MySQLd

```
c:\Program Files\MySQL\MySQL Server 8.0\bin>"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld.exe" --defaults-file="C:\ProgramData\MySQL\MySQL Server 8.0\my.ini" --console
2018-12-13T21:18:19.443263Z 0 [Warning] [MY-010915] [Server] 'NO_ZERO_DATE', 'NO_ZERO_IN_DATE' and 'ERROR_FOR_DIVISION_BY_ZERO' sql modes should be used with strict mode. They will be merged with strict mode in a future release.
2018-12-13T21:18:19.444993Z 0 [System] [MY-010116] [Server] C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld.exe (mysqld 8.0.13) starting as process 9232
2018-12-13T21:18:21.072648Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2018-12-13T21:18:21.109830Z 0 [System] [MY-010931] [Server] C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld.exe: ready for connections. Version: '8.0.13'  socket: ''  port: 3306  MySQL Community Server - GPL.
2018-12-13T21:18:21.219032Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060
```

My.ini

- Le fichier de configuration est organisé en sections
- Une section est composée d'un nom, placé entre crochets.
  - Le nom de la section du serveur est [mysqld]
  - Le nom de la section du client est [mysql]
- La syntaxe des optons est de la forme nom_option=valeur. Les options composées de plusieurs mots sont séparées par le caractère '–'
- Les commentaires sont des lignes préfixér par le caractère # ou ;

Extrait de my.ini

```
[mysqld]

# The next three options are mutually exclusive to SERVER_PORT below.
# skip-networking
enable-named-pipe
# shared-memory

# shared-memory-base-name=MYSQL

# The Pipe the MySQL Server will use
socket=MYSQL

# The TCP/IP Port the MySQL Server will listen on
port=3306

# Path to installation directory. All paths are usually resolved relative to this.
# basedir="C:/Program Files/MySQL/MySQL Server 8.0/"

# Path to the database root
datadir=C:/ProgramData/MySQL/MySQL Server 8.0/Data

# The default character set that will be used when a new schema or table is
# created and no character set is defined
# character-set-server=

# The default authentication plugin to be used when connecting to the server
default_authentication_plugin=mysql_native_password

# The default storage engine that will be used when create new tables when
default-storage-engine=INNODB

# Set the SQL mode to strict
sql-mode="STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION"
```

Paramétrage au lancement de mysqld

Il est possible de configurer le serveur en lui passant les options en paramètre lors du lancement du programme mysqld.

Ces dernières sont les même que celles mises dans le fichier de configuration mais précédées de deux tiret

- Mysqld –datadir=/home/pierre/mysql/data

Paramétrage dynamique du serveur

Si l'utilisateur possède le droit super, il peut changer certaines option dynamiquement.

En cour de session, il est possible de lancer un set SESSION paramètre=valeur.

- SET SESSION sort_buffer_size=1048576.
- SHOW GLOBAL VARIABLES LIKE 'sort_buffer_size'
  - résultat:2097144
- SHOW SESSION VARIABLES LIKE 'sort_buffer_size'
  - résultat: 1048576

Paramétrage dynamique du serveur

En faisant un SET GLOBAL, le changement a lieu pour tous les clients qui se connectent au serveur après l'exécution de la commande.

Cependant cette valeur n'est pas prise en compte pour les sessions en cours.

Néanmoins, ces changements dynamiques ne sont pas persistants car non placé dans un fichier de configuration.

Visualisation de la configuration

Pour visualiser la configuration actuelle du serveur, vous pouvez utiliser

- la commande SHOW GLOBAL VARIABLES,
- soit les tables GLOBAL_VARIABLES du schéma INFORMATION_SCHEMA,
- ou encore la commande SELECT @@global.nom_variable.

Par exemples SELECT * from INFORMATION_SCHEMA.GLOBAL_VARIABLES WHERE VARIABLE_NAME='datadir'

Une erreur est d'exécuter la commande SHOW VARIABLES (variables de session) au lieu de SHOW GLOBAL VARIABLES

SQL_MODE

Le comportement par défaut du serveur MySQL est assez permissif, notamment en ce qui concerne la cohérence des données.

Par exemple, si une chaîne de caractère est trop longue pour être insérée dans une colonne, le serveur effectueras l'insertion en tronquant la chaîne et en affichant un warning.

L'option sql_mode permet de régler la façon dont MySQL va régler ce genre de problème en modifiant le comportement du serveur:

- En renforçant la cohérence du serveur
- En rendant le code SQL portable
- En demandant à MySQL de se comporter comme des SGBD proche.

SQL_MODE

```
mysql> SHOW SESSION VARIABLES LIKE 'sql_mode';
+---------------+-----------------------------------------------+
| Variable_name | Value                                         |
+---------------+-----------------------------------------------+
| sql_mode      | STRICT_TRANS_TABLES,NO_ENGINE_SUBSTITUTION |
+---------------+-----------------------------------------------+
1 row in set (0.01 sec)
```

SQL_MODE

- Vous pouvez combiner plusieurs modes SQL en les plaçant à la suite dans la partie mysqld du fichier my.ini

```
[mysqld]
sql_mode=STRICT_TRANS_TABLES,STRICT_ALL_TABLES,NO_ZERO_IN_DATE
```

SQL_MODE

- ANSI
  - Ce mode modifie la syntaxe et le comportement pour se conformer davantage au SQL standard. Equivalent aux valeurs REAL_AS_FLOAT, PIPES_AS_CONCAT, ANSI_QUOTES, IGNORE_SPACE, et ONLY_FULL_GROUP_BY.
- STRICT_TRANS_TABLES
  - Si une valeur ne peut pas être insérée telle que donnée dans une table transactionnelle, l'instruction est annulé.
- TRADITIONNEL
  - Faites en sorte que MySQL se comporte comme un système de base de données SQL «traditionnel». Une description simple de ce mode est «donnez une erreur au lieu d'un avertissement» lors de l'insertion d'une valeur incorrecte dans une colonne

SQL_MODE

- ANSI_QUOTES
  - Traite "comme un caractère de citation d'identifiant (comme le` caractère de citation) et non comme un caractère de citation de chaîne. Vous pouvez toujours utiliser `pour citer des identifiants avec ce mode activé.
- NO_ENGINE_SUBSTITUTION
  - Contrôlez la substitution automatique du moteur de stockage par défaut lorsqu'une instruction telle que CREATE TABLE ou ALTER TABLE spécifie un moteur de stockage désactivé ou non compilé.
- ONLY_FULL_GROUP_BY
  - Rejetez les requêtes pour lesquelles la liste de sélection, la condition HAVING ou la liste ORDER BY font référence à des colonnes non agrégées qui ne sont ni nommées dans la clause GROUP BY, ni fonctionnellement dépendantes des colonnes GROUP BY (déterminées uniquement par).
- PIPES_AS_CONCAT
  - Traiter || en tant qu'opérateur de concaténation de chaînes (identique à CONCAT ()) plutôt qu'en tant que synonyme de OR.
- IGNORE_SPACE
  - Autorise les espaces entre un nom de fonction et le caractère (.
- REAL_AS_FLOAT
  - Traiter REAL comme synonyme de FLOAT. Par défaut, MySQL considère REAL comme un synonyme de DOUBLE.

Installation
Changements de versions, migration

- Problématique d'une migration
- Process d'une migration.

Changements de versions, migration

- La mise à niveau est une procédure courante, car vous corrigez des bogues dans la même série de versions de MySQL ou avec des fonctionnalités importantes entre les versions principales de MySQL. Vous effectuez d'abord cette procédure sur certains systèmes de test pour vous assurer que tout fonctionne correctement, puis sur les systèmes de production.
- Le downgrading est moins commun. Il est généralement exécuté en raison d'un problème de compatibilité ou de performances sur un système de production qui n'a pas été découvert lors de la vérification initiale de la mise à niveau sur des systèmes de test.

Upgrading MySQL

La mise à jour est une opération risqué (pour MySQL ou autre):

- Faire un backup des données
- Valider le chemin des mise a jour:
  - Pour MySQL seul le schemin des GA (général available) est supporté.
  - Il est recommander de faire les mise a jour en plusieurs fois (5.5 vers 5.6, puis vers 5.7 et enfin vers 5.8)
- Vérifier dans le fichier de configuration la validité des variables systèmes

Upgrading MySQL

- Il est conseillé de consulter les changements apportés par la version de la migration.
- En effet, outre certaines améliorations, il est possible que vous tombiez sur des régressions.
- Enfin une mise à jour de version n'est pas une opération anodine. De ce fait, il est conseillé de faire un backup des données avant.

Upgrading de MySQL

- Il existe plusieurs manières de mettre à jour une instance.
- La plus simple est :
  - d'effectuer une sauvegarde logique (mysqldump),
  - supprimer les fichier de données,
  - Mettre à jour les binaires
  - Réinjecter les données
- Le hic est la taille des données qui peut empêcher de faire cette démarche.
- Dans ce cas, il est possible de simplement mettre a jour les binaires et de redémarrer le serveur
  - Cette démarche peut être problématique

Upgrade par modification de binaires

- Il s'agit d'un remplacement des binaires.
- En premier lieu vérifier que innodb_fast_shutdown est a 1 ou 0 (arrêt rapide ou pas).
- Effectuer un mysqladmin -u root -p shutdown
- Changer les binaires et redémarrer le system:
  - mysqld_safe --user=mysql --datadir=*/path/to/existing-datadir*
- Lorsque vous démarrez le serveur MySQL, il détecte automatiquement la présence de tables de dictionnaire de données. Sinon, le serveur les crée dans le répertoire de données, les remplit avec des métadonnées, puis poursuit sa séquence de démarrage normale. Au cours de ce processus, le serveur met à niveau les métadonnées de tous les objets de base de données, y compris les bases de données, les tablespaces, les tables système et utilisateur, les vues et les programmes stockés.

Upgrading de MySQL

- La mise en œuvre de la réplication peut permettre de faire une mise à jour sans interruption de service.
- Mettre a jour les esclaves de réplication en interdisant le Traffic vers les esclaves lors de la mise à jour.
- Faire la promotion d'un esclave, et faire le rattrapage des données.

Vérification après insertion

- Le client mysqlcheck va vérifier l'upgrade:
  - mysqlcheck -u root -p --all-databases --check-upgrade
- Il vérifie:
  - Qu'aucune table n'utilise des fonctions ou des types de données obsolètes.
  - Qu'il ne doit y avoir aucun fichier .frm orphelin.
  - Valide les triggers

MySQLCheck

```
C:\Program Files\MySQL\MySQL Server 8.0\bin>mysqlcheck -u root -p --all-databases --check-upgrade
Enter password: *****
annuaireautoentrepreneur.type                      OK
mysql.columns_priv                                 OK
mysql.component                                    OK
mysql.db                                           OK
mysql.default_roles                                OK
mysql.engine_cost                                  OK
mysql.func                                         OK
mysql.general_log                                  OK
mysql.global_grants                                OK
mysql.gtid_executed                                OK
mysql.help_category                                OK
mysql.help_keyword                                 OK
mysql.help_relation                                OK
mysql.help_topic                                   OK
mysql.innodb_index_stats                           OK
mysql.innodb_table_stats                           OK
mysql.password_history                             OK
mysql.plugin                                       OK
mysql.procs_priv                                   OK
mysql.proxies_priv                                 OK
mysql.role_edges                                   OK
mysql.server_cost                                  OK
```

Gérer les partitions avec MySQL 8

- Des avertissements sont émis pour les tables qui utilisent un partitionnement non natif, car ce dernier est supprimé de MySQL 8.0.
- Supprimer les partitions non InnoDB:
  - SELECT TABLE_SCHEMA, TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE NOT IN ('innodb', 'ndbcluster') AND CREATE_OPTIONS LIKE '%partitioned%';
- Soit passer les tables sur le storage InnoDB, soit enlever les partitions:
  - ALTER TABLE *table_name* ENGINE = INNODB;
  - ALTER TABLE *table_name* REMOVE PARTITIONING;

Après la migration, vérifier la taille des éléments

- Valider que la taille des noms des clef étrangère ne dépasse pas 64 caractères.
- Les éléments SET ou ENUM ne doivent pas dépassé 255 caractères
- Vérifier la compatibilité du SQL utilisé pour les applicatifs

MySQL Workbench

- Présentation des écrans de MySQL Workbench

MySQL Workbench

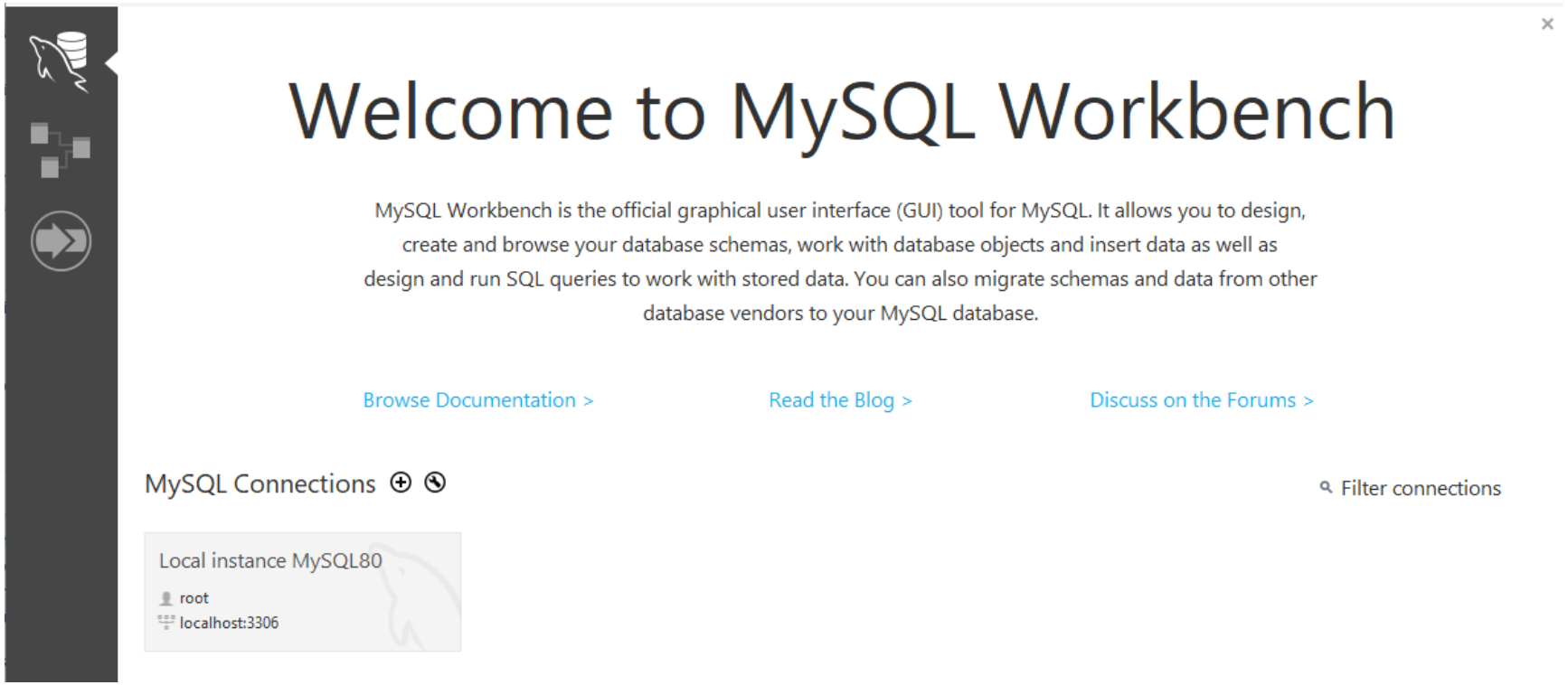
- MySQL Workbench est un outil visuel de conception de base de données qui intègre le développement SQL, l'administration, la conception, la création et la maintenance de base de données dans un seul environnement de développement intégré pour le système de base de données MySQL.
- Actuellement en version 8.0.
- MySQL Workbench propose deux éditions différentes: une édition open source et une édition propriétaire. La "Community Edition" est un produit complet. La version propriétaire "Standard Edition" étend l'édition Community Edition avec une série de modules et de plugins.

Fonctionalités

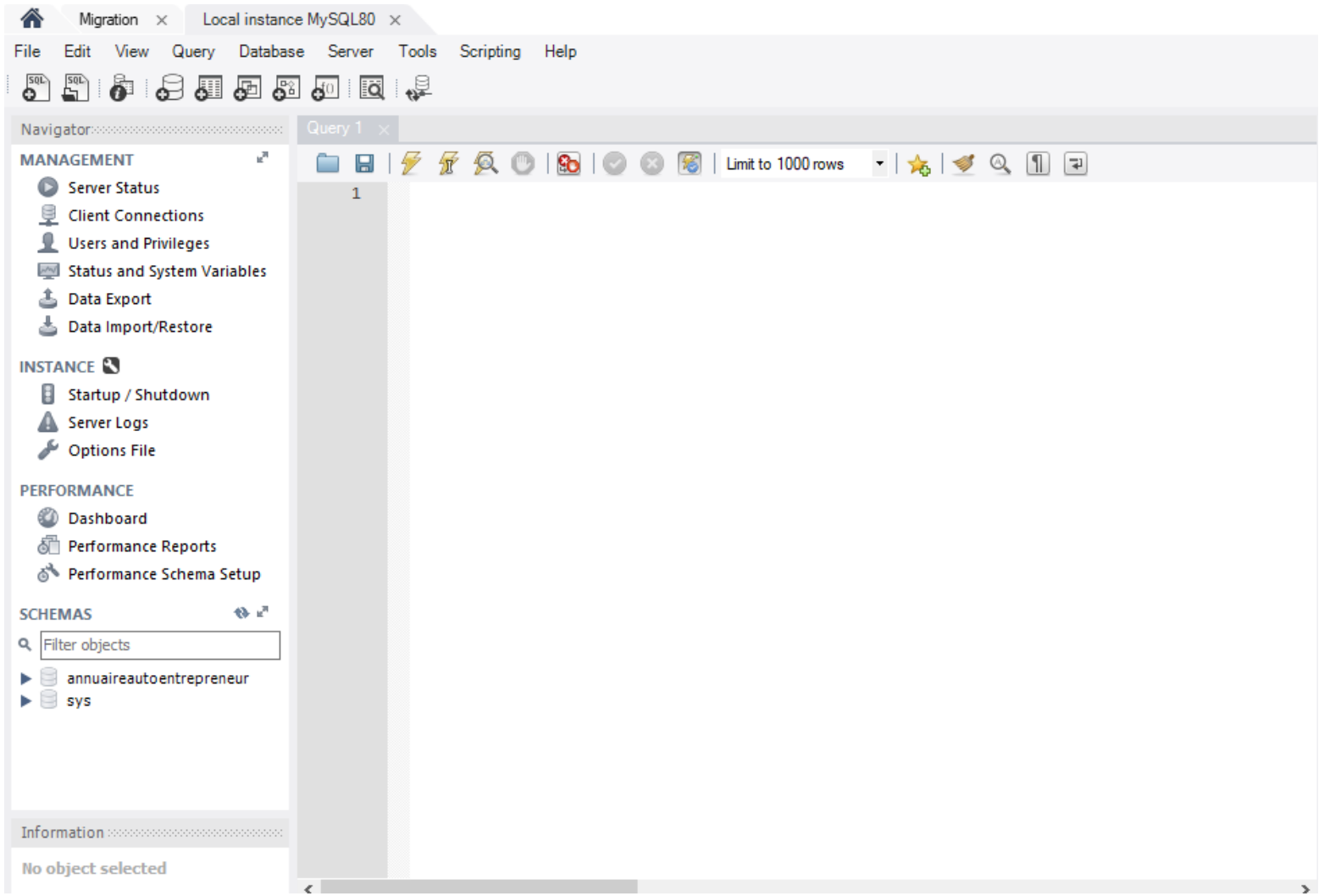
- Général
  - Connexion à la base de données et gestion des instances
  - Actions dirigées par un assistant
  - Entièrement scriptable avec Python et Lua
  - Support pour les plugins personnalisés
  - Prise en charge des fonctionnalités de MySQL Enterprise (journal d'audit, pare-feu et sauvegarde d'entreprise)
- Éditeur SQL
  - Navigation, inspection et recherche dans les objets de schéma
  - Surligneur de syntaxe SQL et analyseur d'instructions
  - Complétion de code SQL et aide contextuelle
  - Tunneling de connexion SSH
  - La modélisation des données
- Diagramme ER
  - Reverse engineering à partir de scripts SQL et d'une base de données dynamique
  - Transférer l'ingénierie vers des scripts SQL et une base de données dynamique
  - Synchronisation de schéma

- Administration des bases de données
  - Démarrer et arrêter des instances de base de données
  - Configuration de l'instance
  - Gestion de compte de base de données
  - Navigation de variables d'instance
  - Navigation dans le fichier journal
  - Exportation / importation de données
- Suivi de la performance
  - Métriques du schéma de performances
  - Tableau de bord d'instance MySQL
  - Statistiques de requête
- Migration de base de données
  - Toute base de données compatible ODBC
  - Prise en charge native: Microsoft SQL Server, PostgreSQL, SQL Anywhere, SQLite et Sybase ASE

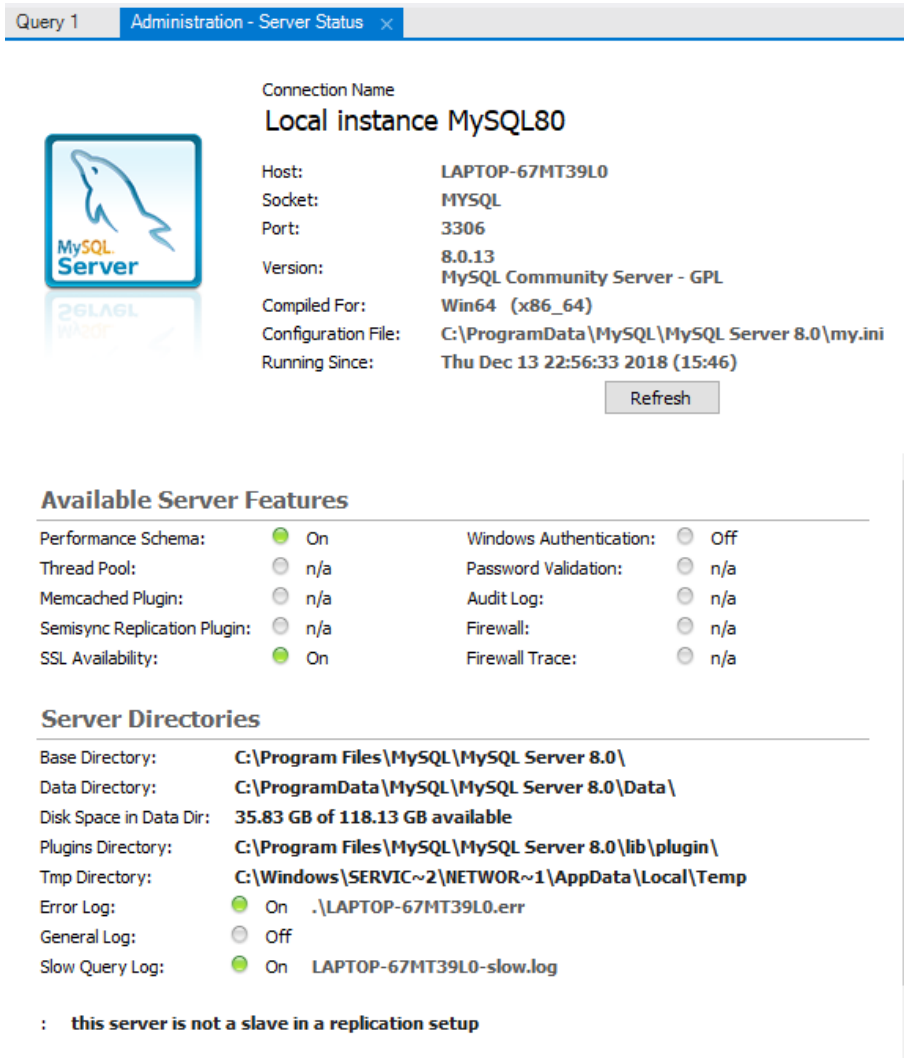
Liste des serveurs



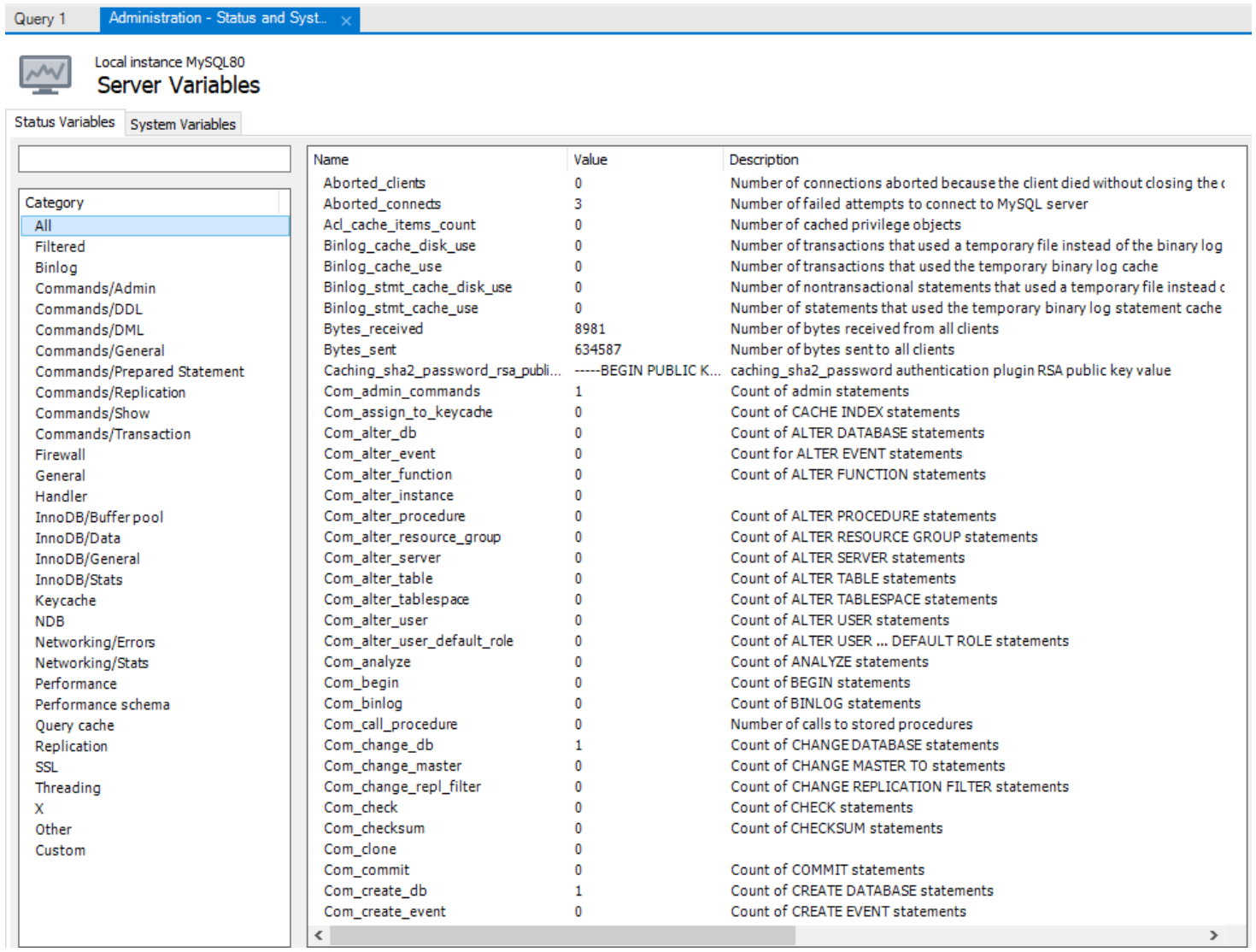
Présentation des onglets de MySQL Workbench



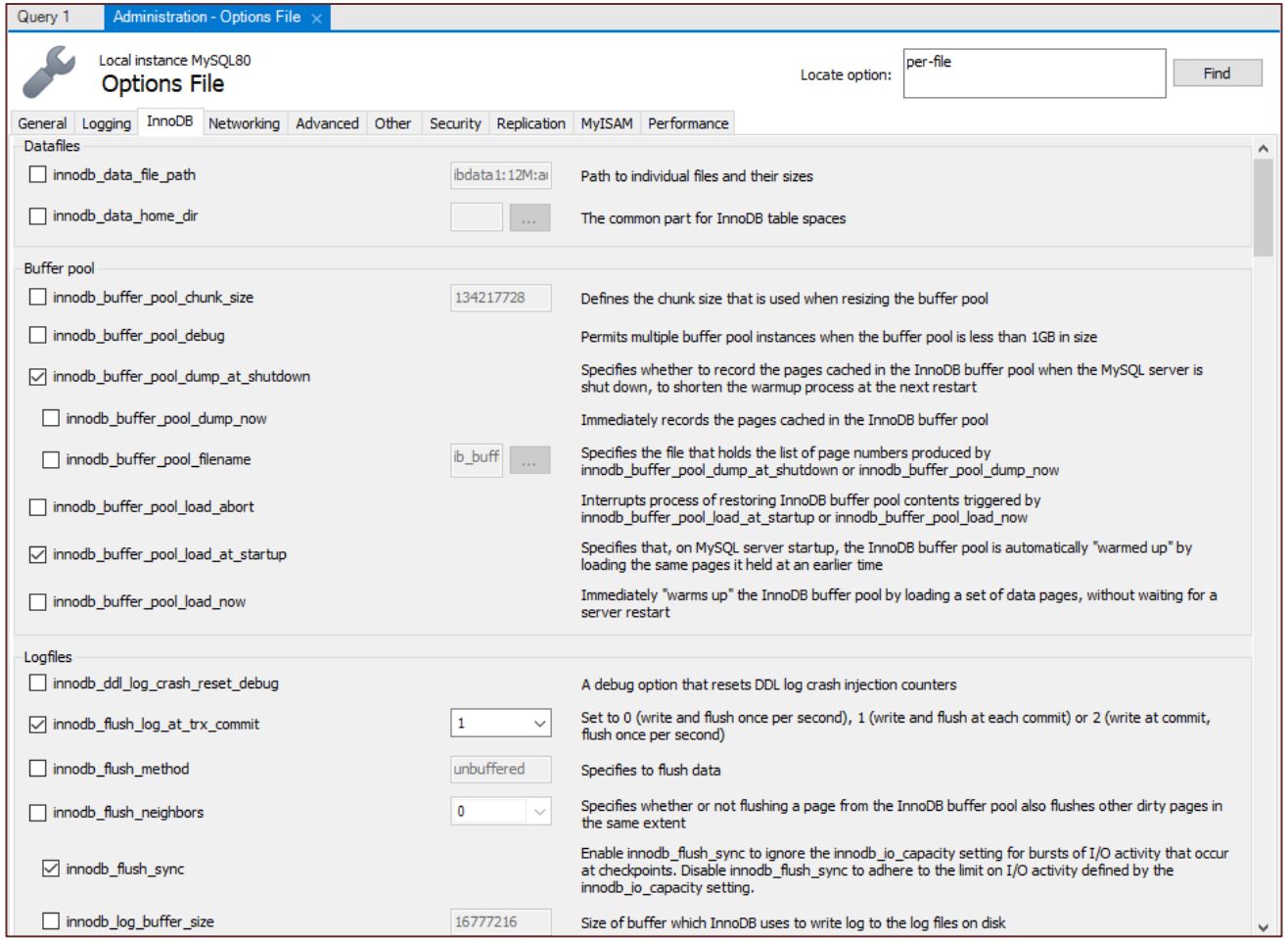
Statut du server MySQL



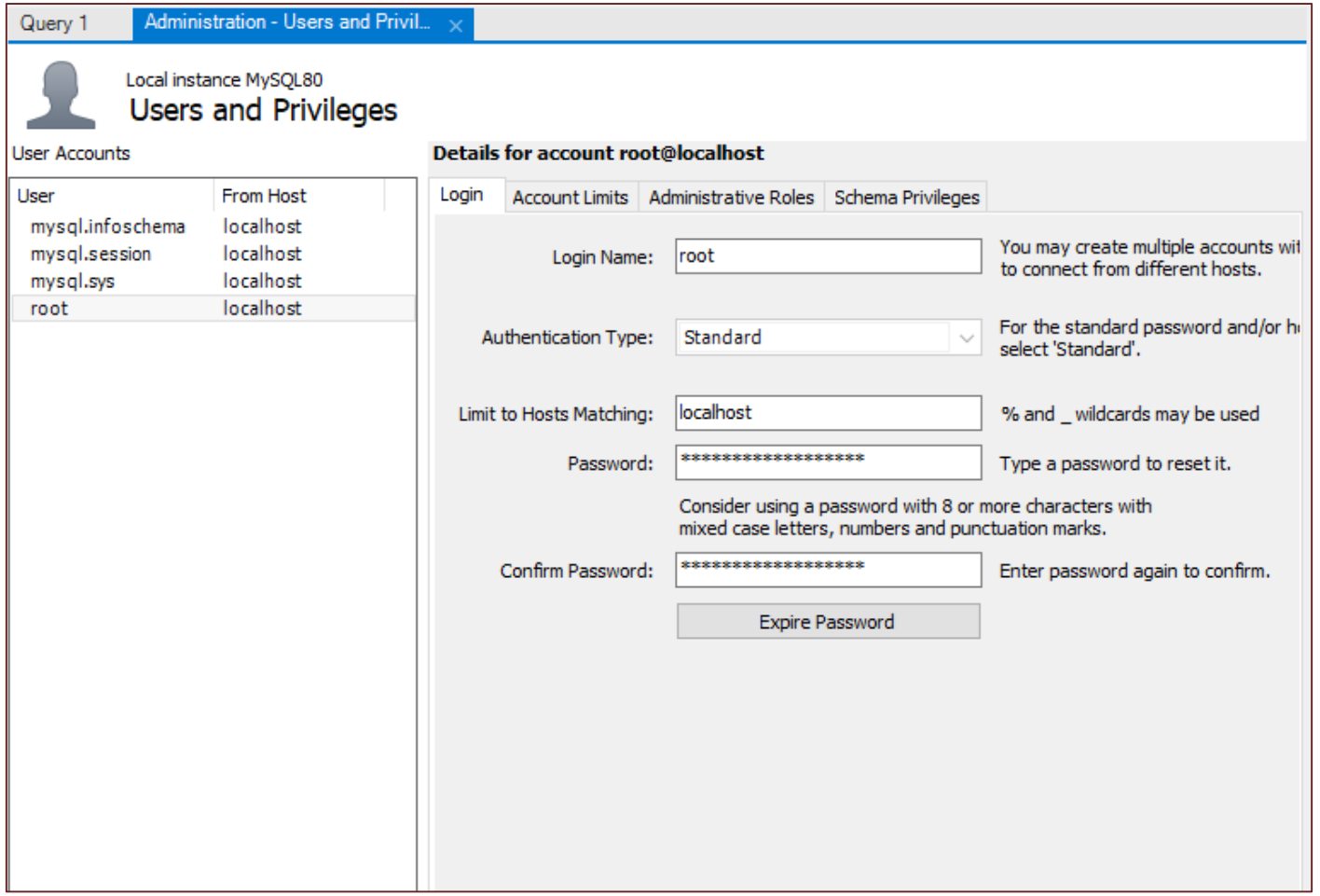
Statut des variables MySQL



My.ini en version graphique



Gestion des utilisateurs



Objets d'une base MySQL

- Types de tables (MyISAM, MEMORY, MERGE...).
- Modification de la définition d'une table.
- Index.
- Gestion des vues.
- Base information_schema.

Objets d'une base MySQL
Types de tables (MyISAM, MEMORY, MERGE...).

- Quels sont les type de tables?
- Découvertes des différents storage engine

Type de Table

MySQL fournit différents moteurs de stockage pour ses tables, comme suit:

- MyISAM
- InnoDB
- MERGE
- MEMORY (HEAP)
- ARCHIVE
- CSV
- FEDERATED

Chaque moteur de stockage a ses propres avantages et inconvénients. Il est essentiel de comprendre les fonctionnalités de chaque moteur de stockage et de choisir celle qui convient le mieux à vos tables afin d'optimiser les performances de la base de données. Dans les sections suivantes, nous aborderons chaque moteur de stockage et ses fonctionnalités afin que vous puissiez choisir celui à utiliser.

MyISAM

- MyISAM étend l'ancien moteur de stockage ISAM. Les tables MyISAM sont optimisées pour la compression et la vitesse. Les tables MyISAM sont également portables entre les plates-formes et les systèmes d'exploitation.
- La taille de la table MyISAM peut atteindre 256 To. De plus, les tables MyISAM peuvent être compressées en tables en lecture seule pour économiser des espaces.
- Au démarrage, MySQL vérifie la corruption des tables MyISAM et les répare même en cas d'erreur.
- Les tables MyISAM ne sont pas sécurisées pour les transactions. Avant MySQL version 5.5, MyISAM était le moteur de stockage par défaut lors de la création d'un table sans moteur de stockage explicite
- À partir de la version 5.5, MySQL utilise InnoDB comme moteur de stockage par défaut.

Création d'une table MyISAM

```
CREATE TABLE fyi_isam ( id INTEGER PRIMARY KEY, title
VARCHAR(80), count INTEGER ) ENGINE = MYISAM;
```

InnoDB

Les tables InnoDB prennent entièrement en charge les transactions et les transactions conformes à ACID. Ils sont également optimaux pour la performance.

La table InnoDB prend en charge les opérations de clés étrangères, de validation, d'annulation et de transfert.

La taille d'une table InnoDB peut atteindre 64 To. Comme MyISAM, les tables InnoDB sont portables entre différentes plates-formes et systèmes d'exploitation. MySQL vérifie et répare les tables InnoDB, si nécessaire, au démarrage.

Création d'une table InnoDB

```
CREATE TABLE fyi_isam ( id INTEGER PRIMARY KEY, title
VARCHAR(80), count INTEGER );
CREATE TABLE fyi_isam ( id INTEGER PRIMARY KEY, title
VARCHAR(80), count INTEGER ) ENGINE = INNODB;
```

MERGE

- Une table MERGE est une table virtuelle qui combine plusieurs tables MyISAM ayant une structure similaire à une table. Le moteur de stockage MERGE est également appelé moteur MRG_MyISAM.
- La table MERGE n'a pas ses propres index; il utilise plutôt les index des tables de composants.
- À l'aide de la table MERGE, vous pouvez accélérer les performances lorsque vous joignez plusieurs tables. MySQL vous permet uniquement d'effectuer des opérations SELECT, DELETE, UPDATE et INSERT sur les tables MERGE.
- Si vous utilisez l'instruction DROP TABLE sur une table MERGE, seule la spécification MERGE est supprimée. Les tables sous-jacentes ne seront pas affectées.

MERGE

```
mysql> CREATE TABLE t1 ( -> a INT NOT NULL AUTO_INCREMENT
PRIMARY KEY, -> message CHAR(20)) ENGINE=MyISAM;
mysql> CREATE TABLE t2 ( -> a INT NOT NULL AUTO_INCREMENT
PRIMARY KEY, -> message CHAR(20)) ENGINE=MyISAM;
mysql> INSERT INTO t1 (message) VALUES
('Testing'),('table'),('t1'); mysql> INSERT INTO t2
(message) VALUES ('Testing'),('table'),('t2');
mysql> CREATE TABLE total ( -> a INT NOT NULL
AUTO_INCREMENT, -> message CHAR(20), INDEX(a)) ->
ENGINE=MERGE UNION=(t1,t2) INSERT_METHOD=LAST;
```

MERGE

Les définitions et les index de la table sous-jacente doivent se conformer étroitement à la définition de la table MERGE.

La conformité est vérifiée lorsqu'une table faisant partie d'une table MERGE est ouverte, et non lorsque la table MERGE est créée.

Si une des tables échoue aux contrôles de conformité, l'opération ayant déclenché l'ouverture de la table échoue.

Cela signifie que les modifications apportées aux définitions des tables dans un MERGE peuvent provoquer un échec lors de l'accès à la table MERGE.

Les contrôles de conformité appliqués à chaque table sont les suivants:

- La table sous-jacente et la table MERGE doivent avoir le même nombre de colonnes.
- L'ordre des colonnes dans la table sous-jacente et dans la table MERGE doit correspondre.
- De plus, la spécification de chaque colonne correspondante de la table MERGE parent et des tables sous-jacentes est comparée et doit satisfaire à ces vérifications:
- Le type de colonne dans la table sous-jacente et la table MERGE doivent être égaux.
- La longueur de la colonne dans la table sous-jacente et la table MERGE doivent être égales.
- La colonne de la table sous-jacente et la table MERGE peuvent être NULL.
- La table sous-jacente doit avoir au moins autant d'index que la table MERGE.
- La table sous-jacente peut avoir plus d'indices que la table MERGE, mais ne peut en avoir moins.

MEMORY

Les tables de la mémoire sont stockées dans la mémoire et utilisent des index de hachage pour qu'ils soient plus rapides que les tables MyISAM.

La durée de vie des données des tables de la mémoire dépend de la disponibilité du serveur de base de données. Le moteur de stockage en mémoire est anciennement appelé HEAP.

MEMORY

Les performances de MEMORY sont limitées par les conflits résultant de **l'exécution d'un seul thread** et de la surcharge de verrouillage de table lors du traitement des mises à jour. Cela limite l'évolutivité lorsque la charge augmente, en particulier pour les mix d'instructions qui incluent des écritures.

Malgré le traitement en mémoire des tables MEMORY, elles ne sont pas nécessairement plus rapides que les tables InnoDB sur un serveur occupé, pour des requêtes à usage général ou sous une charge de travail en lecture / écriture.

En particulier, le verrouillage des tables impliqué dans les mises à jour peut ralentir l'utilisation simultanée des tables MEMORY à partir de plusieurs sessions. Selon les types de requêtes exécutées sur une table MEMORY, vous pouvez créer des index en tant que structure de données de hachage par défaut (pour rechercher des valeurs uniques basées sur une clé unique) ou en tant que structure de données B-tree à usage général (pour tous les types).

MEMORY

```
CREATE TABLE lookup (id INT, INDEX USING HASH (id)) ENGINE
= MEMORY;
CREATE TABLE lookup (id INT, INDEX USING BTREE (id)) ENGINE
= MEMORY;
```

MEMORY et Replication

Les tables MEMORY d'un serveur deviennent vides quand il est arrêté et redémarré. Si le serveur est un maître de réplication, ses esclaves ne sont pas conscients que ces tables sont devenues vides. Le contenu est donc obsolète si vous sélectionnez des données dans les tables des esclaves.

Archive

Le moteur de stockage d'archives vous permet de stocker un grand nombre d'enregistrements dans un format compressé à des fins d'archivage afin d'économiser de l'espace disque. Le moteur de stockage d'archives compresse un enregistrement lorsqu'il est inséré et le décompresse à l'aide de la bibliothèque zlib au fur et à mesure de sa lecture. Les tables d'archives n'autorisent que les instructions INSERT et SELECT. Les tables ARCHIVE ne supportent pas les index, une analyse complète de la table est donc nécessaire pour lire les lignes.

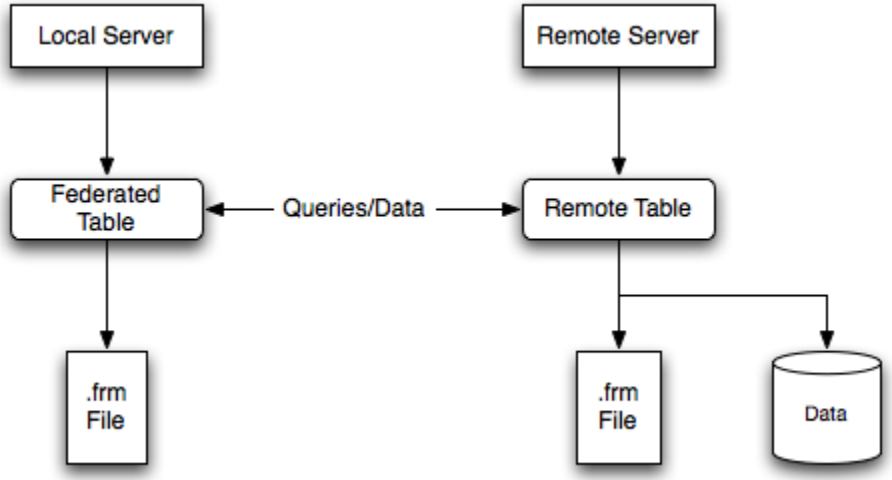
Une utilité des tables archives est de stocké des informations semblables en grand nombre (table de log par exemple)

CSV

Le moteur de stockage CSV stocke les données au format de fichier CSV (valeurs séparées par des virgules). Un tableau CSV constitue un moyen pratique de migrer des données vers des applications non SQL, telles que des tableurs. La table CSV ne prend pas en charge le type de données NULL. De plus, l'opération de lecture nécessite une analyse complète de la table.

FEDERATED

Le moteur de stockage FEDERATED vous permet de gérer les données d'un serveur MySQL distant sans utiliser la technologie de cluster ou de réplication. La table fédérée locale ne stocke aucune donnée. Lorsque vous interrogez des données à partir d'une table fédérée locale, les données sont extraites automatiquement des tables fédérées distantes.



FEDERATED

Pour créer une table FEDERATED, procédez comme suit:

- Créez la table sur le serveur distant. Vous pouvez également noter la définition de la table d'une table existante, en utilisant éventuellement l'instruction SHOW CREATE TABLE.
- Créez la table sur le serveur local avec une définition de table identique, mais en ajoutant les informations de connexion qui lient la table locale à la table distante.

FEDERATED TABLE via Connection

```
CREATE TABLE federated_table ( id INT(20) NOT NULL
AUTO_INCREMENT, name VARCHAR(32) NOT NULL DEFAULT '', other
INT(20) NOT NULL DEFAULT '0', PRIMARY KEY (id), INDEX name
(name), INDEX other_key (other) ) ENGINE=FEDERATED
DEFAULT CHARSET=utf8mb4
CONNECTION='mysql://fed_user@remote_host:9306/federated/tes
t_table';
```

FEDERATED via Server

```
CREATE SERVER fedlink FOREIGN DATA WRAPPER
mysql OPTIONS (USER 'fed_user', HOST
'remote_host', PORT 9306, DATABASE
'federated');


CREATE TABLE test_table ( id INT(20) NOT
NULL AUTO_INCREMENT, name VARCHAR(32) NOT
NULL DEFAULT '', other INT(20) NOT NULL
DEFAULT '0', PRIMARY KEY (id), INDEX name
(name), INDEX other_key (other) )
ENGINE=FEDERATED DEFAULT CHARSET=utf8mb4
CONNECTION='fedlink/test_table';
```

FEDERATED

Les éléments suivants indiquent les fonctionnalités que le moteur de stockage FEDERATED prend en charge et ne prend pas en charge:

- Le serveur distant doit être un serveur MySQL.
- La table distante pointée sur une table FEDERATED doit exister pour que vous puissiez accéder à la table par le biais de la table FEDERATED.
- Il est possible qu'une table FEDERATED en pointe une autre, mais veillez à ne pas créer de boucle.
- Une table FEDERATED ne supporte pas les index au sens habituel. comme l'accès aux données de la table est géré à distance, c'est la table distante qui utilise les index. **Cela signifie que, pour une requête ne pouvant utiliser aucun index et nécessitant une analyse complète de la table, le serveur extrait toutes les lignes de la table distante et les filtre localement.** Cela se produit indépendamment de tout WHERE ou LIMIT utilisé avec cette instruction SELECT; ces clauses sont appliquées localement aux lignes renvoyées. Les requêtes qui n'utilisent pas les index peuvent donc entraîner de mauvaises performances et une surcharge du réseau. **De plus, étant donné que les lignes renvoyées doivent être stockées en mémoire, une telle requête peut également entraîner le swap, voire l'interruption du serveur local.**

Résumé

Fonctionnalités	Innodb	MyISAM	Memory	CSV	Archive
B-tree					
Backup Recovery					
Clustering de données					
Clustering d'index					
Compression					
Cache de données					
Encryption					
Foreign Key					
Full Text					
Hash Index					
Type de Lock	Par ligne	Par table	Par table	Par table	Par ligne
MVCC (Transactionel)					
Replication					
Limit	64TB	256 TB	Mémoire	Disque	Disque
Transaction					

Objets d'une base MySQL
Modification de la définition d'une table.

- Commande DROP et TRUNCATE
- Commande RENAME
- Commande ALTER TABLE

Modification d'une table

Il est possible de supprimer une table grâce à la clause *DROP*, il existe aussi des commandes moins extrêmes permettant

- L'ajout de colonnes
- La modification de colonnes
- La suppression de colonnes

Suppression d'éléments

La clause *DROP* permet d'éliminer des vues, des index et même des tables. Cette clause est toutefois à utiliser avec parcimonie dans la mesure où elle est irréversible.

La suppression d'une vue se fait avec la syntaxe suivante :

- DROP VIEW Nom_de_la_vue

La suppression d'un index se fait avec la syntaxe suivante :

- DROP INDEX Nom_de_l_index

La suppression d'une table se fait avec la syntaxe suivante :

- DROP TABLE Nom_de_la_table

Suppression des données

La clause *DROP* lorsqu'elle est utilisée sur une table élimine les données ainsi que la structure de la table. Il est possible de supprimer uniquement les données en conservant la structure de la table grâce à la clause *TRUNCATE*.

La suppression des données d'une table se fait avec la syntaxe suivante :

- TRUNCATE TABLE Nom_de_la_table

Truncate vs Delete

Delete <table> where <condition>

1. Supprime certaines ou toutes les lignes d'une table.
2. Une clause WHERE peut être utilisée pour supprimer certaines lignes. Si aucune condition WHERE n'est spécifiée, toutes les lignes seront supprimées.
3. Tous les déclencheurs DELETE de la table sont déclenchés.
4. Supprime les lignes ligne par ligne, une à la fois, et enregistre une entrée dans les journaux des transactions. Il est donc plus lent que TRUNCATE.
5. Chaque ligne supprimée est verrouillée et nécessite donc plus de verrous et de ressources de base de données.

Truncate vs Delete

Truncate <table>

- Supprime toutes les lignes d'une table.
- Ne nécessite pas de clause WHERE, vous ne pouvez donc pas filtrer les lignes lors de la troncature.
- Aucun trigger n'est déclenché lors de cette opération car elle ne fonctionne pas sur des lignes individuelles.

RENAME

Il peut parfois être intéressant de renommer une table, c'est la clause *RENAME* qui permet cette opération. La syntaxe de cette clause est :

- RENAME TABLE Ancien_Nom TO Nouveau_Nom

Alter Table

- ALTER TABLE vous permet de changer la structure d'une table existante.
- Pour utiliser ALTER TABLE, vous devez avoir les droits ALTER, INSERT, et CREATE sur la table.
- ALTER TABLE effectue une copie temporaire de la table originale. Les modifications sont faites sur cette copie, puis l'original est effacée, et enfin la copie est renommée pour remplacer l'originale. Cette méthode permet de rediriger toutes les commandes automatiquement vers la nouvelle table sans pertes. Durant l'exécution de ALTER TABLE, la table originale est lisible par d'autres clients. Les modifications et insertions sont reportées jusqu'à ce que la nouvelle table soit prête.

Suppression de colonnes

La clause *ALTER* permet la modification des colonnes d'une table. Associée avec la clause *DROP COLUMN*, elle permet de supprimer des colonnes. La syntaxe est la suivante :

ALTER TABLE Nom_de_la_table **DROP COLUMN** Nom_de_la_colonne

Il faut noter que la suppression de colonnes n'est possible que dans le cas où:

- La colonne ne fait pas partie d'une vue
- La colonne ne fait pas partie d'un index
- La colonne n'est pas l'objet d'une contrainte d'intégrité

Ajout/ Modification de colonnes

Associée avec la clause *ADD*, la clause *ALTER* permet l'ajout de colonnes à une table. La syntaxe est la suivante :

- ALTER TABLE Nom_de_la_table
  ADD Nom_de_la_colonne Type_de_donnees

Vous pouvez renommer une colonne avec la syntaxe CHANGE ancien_nom_de_colonne create_definition.

ALTER TABLE t1 CHANGE Nom_de_la_colonne Nouveau_nom INTEGER;

Modification d’engine pour une table

Pour convertir une table d'un moteur de stockage à un autre, utilisez une instruction ALTER TABLE indiquant le nouveau moteur:

- ALTER TABLE t ENGINE = InnoDB;

Si vous essayez d'utiliser un moteur de stockage qui n'est pas compilé dans ou qui est compilé mais qui est désactivé, MySQL crée une table à l'aide du moteur de stockage par défaut.

Objets d'une base MySQL
Index

- Qu'est ce qu'un index?
- Différent type d'index?
- Relation entre les index et les contraintes

Index

Les index sont utilisés pour accélérer les requêtes (notamment les requêtes impliquant plusieurs tables, ou les requêtes de recherche), et sont indispensables à la création de clés, étrangères et primaires, qui permettent de garantir l'intégrité des données de la base.

Index

Lorsque vous créez un index sur une table, MySQL stocke cet index sous forme d'une structure particulière, contenant les valeurs des colonnes impliquées dans l'index. Cette structure stocke les valeurs **triées** et permet d'accéder à chacune de manière efficace et rapide.

Index Unique/Clef Primaire

Avoir un index UNIQUE sur une colonne (ou plusieurs) permet de s'assurer que jamais vous n'insérerez deux fois la même valeur (ou combinaison de valeurs) dans la table.

Lorsque vous mettez un index UNIQUE sur une table, vous ne mettez pas seulement un index, vous ajoutez surtout une **contrainte**.

Les clefs « unique » identifient une ligne d'une table.

- Elles identifient les lignes existantes
- Et les lignes à venir

Les clefs primaires sont des clefs uniques n'autorisant pas la valeur NULL

Il n'existe qu'une clef primaire par table

Les clefs uniques simples peuvent être multiples et contenir des valeurs NULL.

PRIMARY/UNIQUE KEY

Sur une table existante

- ALTER TABLE <TABLE identifier> ADD [CONSTRAINT <CONSTRAINT identifier>] PRIMARY KEY (<COLUMN expression> {, <COLUMN expression>}...)
- ALTER TABLE <TABLE identifier> ADD [CONSTRAINT <CONSTRAINT identifier>] UNIQUE (<COLUMN expression> {, <COLUMN expression>}...)

A la création d'une table

- CREATE TABLE TABLE_NAME (id_col INT PRIMARY KEY, col2 CHARACTER VARYING(20), ... key_col SMALLINT UNIQUE, ...)

PRIMARY KEY/AUTO INCREMENT

Il s'agit de la capacité de signaler à MySQL qu'une clef primaire est :

- Un entier
- Chaque ligne obtient automatiquement une valeur donnée par MySQL

Syntaxe

- CREATE TABLE <table name> (nom colonne type entier PRIMARY KEY AUTO_INCREMENT)

Auto Increment

```
CREATE TABLE `livres` (
 livre_id int(11) NOT NULL AUTO_INCREMENT,
 titre varchar(50) DEFAULT NULL,
 auteurs varchar(50) DEFAULT NULL,
 description text,
 PRIMARY KEY (livre_id)
)
```

Auto Increment

```
insert into livres(titre,auteurs,description) values('un
titre','moi','mon livre')

Select * from livres
+----------+----------+---------+--------------+
| livre_id | titre    | auteurs | description |
+----------+----------+---------+--------------+
|        1 | un titre | moi     | mon livre    |
+----------+----------+---------+--------------+
```

AUTO_INCREMENT

- Si vous spécifiez une colonne AUTO_INCREMENT dans une table, la table InnoDB va ajouter dans le dictionnaire de données un compteur spécial appelé le compteur auto-incrément, qui est utilisé pour assigner les nouvelles valeurs de la colonne. Le compteur est stocké uniquement en mémoire, et non pas sur le disque.

- La requête exécutée par MySQL est :
- **SELECT MAX(ai_col) FROM T**

- La valeur lue par la commande est incrémentée d'une unité, et assignée à la colonne auto-incrément et au compteur de table. Si la table est vide, la valeur de 1 est assignée.
- Après l'initialisation du compteur d'auto-incrémentation, si un utilisateur insère une ligne qui définit explicitement la valeur de la colonne, et que cette valeur est plus grande que la valeur courante du compteur, le compteur prend alors cette valeur. Si l'utilisateur ne spécifie pas de valeur, MySQL incrémente le compteur d'une unité, et assigne une nouvelle valeur à cette colonne.

- **Le comportement du mécanisme auto-increment n'est pas défini si vous assignez une valeur négative à la colonne, ou si cette dernière dépasse la capacité de la colonne.**

FOREIGN KEY

Il s'agit d'une contrainte référentielle entre deux tables.

Il s'agit du champ d'une table qui a une référence sur une clef unique d'une autre table

Il s'agit de l'idée d'associer deux tables, par exemple une table de clients et une table de commandes.

- Chaque commande est faite par un client
- Chaque client ne fait pas forcement une commande

Il y a donc une référence des commandes vers les clients.

FOREIGN KEY

Syntaxe:

- CREATE TABLE <nom de table> (id INTEGER PRIMARY KEY, col2 VARCHAR(20), col3 INTEGER, ... **FOREIGN KEY(col3) REFERENCES <nom de table> (key_col)**)
- **ALTER TABLE <nom de table> ADD FOREIGN KEY (<nom de colonne> REFERENCES <nom de table>**

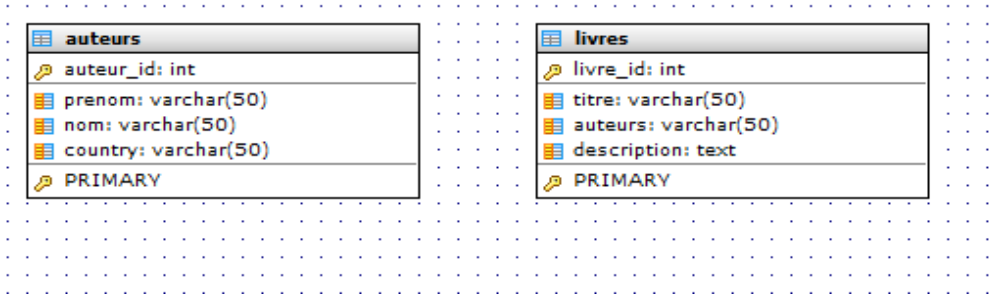
Exemple:

```
CREATE TABLE auteurs (
  auteur_id int(11) NOT NULL PRIMARY KEY AUTO_INCREMENT,
  prenom varchar(50) DEFAULT NULL,
  nom varchar(50) DEFAULT NULL,
  country varchar(50) DEFAULT NULL
)
```

FOREIGN KEY

Volonté de mettre en relation nos livres avec nos auteurs:

- **ALTER TABLE** livres **CHANGE** auteurs auteurs INT
- **ALTER TABLE** livres **ADD CONSTRAINT** `relation_auteurs` **FOREIGN KEY** (`auteurs`) REFERENCES `auteurs`(`auteur_id`);



FOREIGN KEY

- Maintenant la table livres est « protégée » si les auteurs n'existent pas.

```
mysql> insert into livres(titre,auteurs,description) values ('un
titre',1,'livre');

ERROR 1452 (23000): Cannot add or update a child row: a foreign
key constraint fails (`bibliotheque`.`livres`, CONSTRAINT
`livres_ibfk_1` FOREIGN KEY (`auteurs`) REFERENCES `auteurs`
(`auteur_id`))
```

Exemple

Nous insérons l'auteur , et MySQL accepte maintenant notre insertion.

```
mysql> insert into auteurs(prenom,nom,country) values('moi','toi','france');
Query OK, 1 row affected (0.00 sec)

mysql> select * from auteurs;
+-----------+--------+------+---------+
| auteur_id | prenom | nom  | country |
+-----------+--------+------+---------+
|         1 | moi    | toi  | france  |
+-----------+--------+------+---------+
1 row in set (0.00 sec)

mysql> insert into livres (titre,auteurs,description) values ('mon titre',1,'livre');
Query OK, 1 row affected (0.00 sec)
```

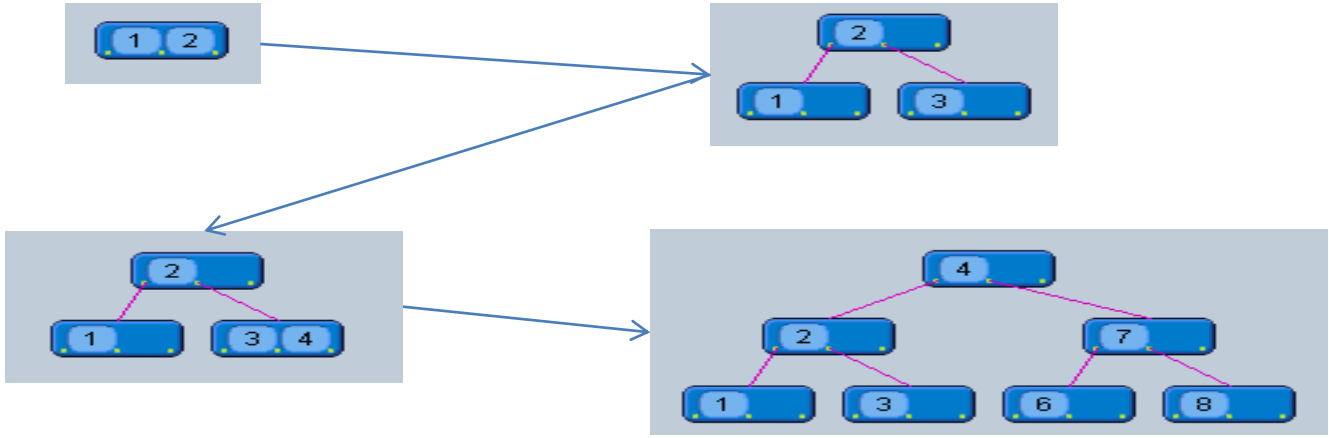
L’objet Btree (Index)

Arbre de recherche équilibré

Capable d’avoir plus d’une clef par nœud (limite la taille de l’arbre)

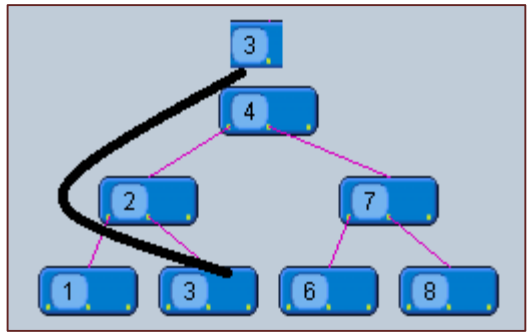
Modélise les index d’une base de données

L'objet Btree : Insertion



- Ce qu'il faut retenir:
  - Un Btree est un arbre de recherche binaire
  - Plusieurs clefs peuvent être contenues dans une feuille

Btree : Recherche



- L’arbre de recherche (ici un Btree) est très fort pour chercher un élément

BTREE/HASH

Tout d'abord, selon le moteur de stockage utilisé, vous pouvez tout simplement pas avoir le choix (par exemple InnoDB est exclusivement en utilisant BTREE pour son indice).

En outre, BTREE est le type d'index par défaut pour la plupart des moteurs de stockage.

Maintenant ... Il ya des cas, lors de l'utilisation d'autres types d'index peut entraîner une amélioration des performances. Il existe (cas relativement rare) quand un index HASH peut aider. Notez que quand un index HASH est créé, un indice BTREE est également produit. C'est en partie dû au fait que les index hash ne peuvent résoudre les prédicats d'égalité. (une condition telle que WHERE Prix> 12,0 pouvait pas être manipulée par un index hash).

En bref: Continuer à utiliser BTREE, que ce soit implicitement (si BTREE est la valeur par défaut pour le stockage d'occasion), ou explicitement. Renseignez-vous sur les autres types d'index afin que vous sachiez à leur sujet serait le besoin s'en fait sentir.

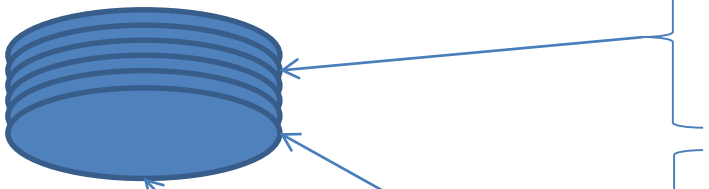
Index HASH sont plus génériques (non limités à un type particulier d'application ou de données), et on peut généralement suivre sa compréhension intuitive de hachages pour obtenir un indice quant au moment où ceux-ci peuvent surperformer.

En Pratique

PK	Cru	Qualité	DEVRE	MILLESIME
1	Beaujolais	Excellent	11	1987
2	Chenas	Mediocre	12	1990
3	Julienas	Bonne	12	1990
5	Beaujolais	Mediocre	10	1985
6	Julienas	Bonne	12	1987
7	Chenas	Mediocre	11	1989
14	Chenas	Bonne	10	1994
15	Julienas	Mediocre	11	1995
18	Beaujolais	Bonne	10	1990
25	Julienas	Bonne	12	1988

La table des vins est organisée en fonction des PK

En Pratique



Le disque dur

PK	Cru	Qualité	DEVRE	MILLESIME
1	Beaujolais	Excellent	11	1987
2	Chenas	Mediocre	12	1990
3	Julienas	Bonne	12	1990
5	Beaujolais	Mediocre	10	1985
6	Julienas	Bonne	12	1987
7	Chenas	Mediocre	11	1989
14	Chenas	Bonne	10	1994
15	Julienas	Mediocre	11	1995
18	Beaujolais	Bonne	10	1990
25	Julienas	Bonne	12	1988

Les PK sont indexées via un Btree+ (organisées par clef croissante) et les index "secondaires" sont organisés en Btree

En Pratique

Index BTREE sur les cru	
Beaujolais	1,5,18
Chenas	2,7,14
Julienas	3,6,15,25

Index BTREE sur degree	
10	5,14,18
11	1,7,15
12	2,3,6,25

PK	Cru	Qualité	DEVRE	MILLESIME
1	Beaujolais	Excellent	11	1987
2	Chenas	Mediocre	12	1990
3	Julienas	Bonne	12	1990
5	Beaujolais	Mediocre	10	1985
6	Julienas	Bonne	12	1987
7	Chenas	Mediocre	11	1989
14	Chenas	Bonne	10	1994
15	Julienas	Mediocre	11	1995
18	Beaujolais	Bonne	10	1990
25	Julienas	Bonne	12	1988

Il y 2 index secondaires, les “cru” et les “degree” organisés en Btree

En Pratique

Index BTREE sur les cru	
Beaujolais	1,5,18
Chenas	**14,2,7**
Julienas	3,6,15,25

Index BTREE sur degree	
10	5,14,18
11	1,7,15
12	**2,3,6,25**

PK	Cru	Qualité	DEVRE	MILLESIME
1	Beaujolais	Excellent	11	1987
2	Chenas	Mediocre	12	1990
3	Julienas	Bonne	12	1990
5	Beaujolais	Mediocre	10	1985
6	Julienas	Bonne	12	1987
7	Chenas	Mediocre	11	1989
14	Chenas	Bonne	10	1994
15	Julienas	Mediocre	11	1995
18	Beaujolais	Bonne	10	1990
25	Julienas	Bonne	12	1988

Select * from table where cru='chenas' AND millesime>1986 AND degree=12

En pratique

Select * from table where cru=‘chenas’ AND millesime>1986 AND degree=12

Utilisation des index cru et chenas

En interne:

C =LIRE index CRU entée CHENAS

D =LIRE index degre entée 12

L=union (C,D)

Puis Select * from L where millesime>1986

Btree & Btree+

Index BTREE sur les cru	
Beaujolais	1,5,18
Chenas	**14,2,7**
Julienas	3,6,15,25

Index BTREE sur degree	
10	5,14,18
11	1,7,15
12	**2,3,6,25**

Fusion	
Chenas	**14,2,7**
12	**2,3,6,25**

Index BTREE+ sur les cru	
Beaujolais	1,5,18
Chenas	**2,7,14**
Julienas	3,6,15,25

Index BTREE+ sur degree	
10	5,14,18
11	1,7,15
12	**2,3,6,25**

Fusion (btree+)	
Chenas	**2**,7,14
12	**2**,3,6,25

En pratique

	BTREE	BTREE+	Hash Index
Recherche			
Recherche Multiple			
Efficacité vis-à-vis du disque			
Efficacité vis-à-vis de l'opérateur '='			
Efficacité vis-à-vis de l'opérateur < ou >			

Objets d'une base MySQL
Gestion des vues

- Comment créer une vue?
- Type d'algorithme pour une vue.

Les vues

Les vues sont des tables virtuelles issues de l'assemblage d'autres tables en fonction de critères. Techniquement les vues sont créées à l'aide d'une requête *SELECT*.

Elles ne stockent pas les données qu'elles contiennent mais conservent juste la requête permettant de les créer.

Les vues

- La requête *SELECT* qui génère la vue référence une ou plusieurs tables. La vue peut donc être, par exemple:
  - une jointure entre différentes tables
  - l'agrégation ou l'extraction de certaines colonnes d'une table
  - créée à partir d'une autre vue.

Les vues

- Contrôler l'intégrité en restreignant l'accès aux données pour améliorer la confidentialité.
  - Partitionnement vertical et/ou horizontal pour cacher des champs aux utilisateurs, ce qui permet de personnaliser l'affichage des informations suivant le type d'utilisateur.
- Masquer la complexité du schéma.
  - Indépendance logique des données, utile pour donner aux utilisateurs l'accès à un ensemble de relations représentées sous la forme d'une table. Les données de la vue sont alors des champs de différentes tables regroupées, ou des résultats d'opérations sur ces champs.
- Modifier automatiquement des données sélectionnées (*sum*(), *avg*(), *max*(),...).
  - Manipuler des valeurs calculées à partir d'autres valeurs du schéma.
- Conserver la structure d'une table si elle doit être modifiée.
  - Le schéma peut ainsi être modifié sans qu'il ne soit nécessaire de changer les requêtes du côté applicatif.

Les vues

Colonne calculée

- Une table ne devrait pas avoir de colonne calculée. Les vues servent à cela.

Compatibilité

- Lors d'une mise à jour du schéma, il est possible via des vues de « masquer » ces modifications.

Création d'une vue

- CREATE [ALGORITHM = {MERGE | TEMPTABLE | UNDEFINED}]VIEW [Nom de la vue AS[SELECT requete]

- Requête: Il s'agit de la requête permettant de "remplir" la vue

- 3 algorithmes pour créer une vue MERGE, TEMPTABLE,UNDEFINED

Création d'une vue

- La requête associé ne soit pas contenir de sous select
- Les tables temporaires ne peuvent participer à une vue
- Une vue ne peut être associé à des triggers

Exemple

- Il s'agit de la création d'une vue des produits plus chers que la moyenne:

```
CREATE VIEW vwProducts
AS
SELECT productCode,       productName,       buyPrice FROM products
WHERE
buyPrice >  (
     SELECT AVG  (buyPrice)     FROM  products)ORDER BY buyPrice DESC
             )
```

Comment ca marche ?

- En interne, les systèmes se font souvent par réécriture de requêtes
- Définition :

```
CREATE VIEW vwProducts  AS SELECT productCode,       productName,
buyPrice FROM products WHERE buyPrice >  ( SELECT AVG  (buyPrice)
FROM  products)ORDER BY buyPrice DESC )
```

- Requête:

```
Select productCode from vwProducts
```

- En interne:

```
Select TMP.productCode from
(SELECT productCode, productName,       buyPrice FROM products AS TMP
WHERE
buyPrice >  (
     SELECT AVG  (buyPrice)     FROM  products)ORDER BY buyPrice DESC
            )
)
```

MERGE/TEMPTABLE/UNDEFINED

- Il s'agit des stratégies de MySQL pour utiliser une requête au regard de la requête de la vue:
- MERGE: Uniquement applicable si une ligne de la vue correspondent à une ligne dans la table sous jacente. Ces types de vues sont « updatable » (Il s'agit d'une vue par réécriture).
- TEMPTABLE: MySQL va construire une table temporaire avec les données des tables sous jacentes. Ces types de vue ne sont pas updatable. (Dans ce cas, une table est créée avec CREATE TABLE XXX AS …)
- UNDEFINES: MySQL choisie la stratégie.

Mise à jour des vues

En général pas possible sauf :

- La requête qui génère la vue doit permettre à MySQL de retrouver la trace de l'enregistrement à modifier dans la ou les tables sous-jacentes ainsi que celle de toutes les valeurs de chaque colonne. La requête *SELECT* créant la vue ne doit donc pas contenir de clause *DISTINCT*, *GROUP BY*, *HAVING*... et autres fonctions d'agrégation.
- En pratique , une vue peut rarement être mise à jour

Mise à jour des vues

Les contraintes sont donc :

La vue ne doit contenir qu'une table

La requête ne doit pas contenir de GROUP BY, de HAVING, d'agrégats, de colonnes calculées..

La vue ne doit pas contenir de vues qui ne soient pas updatable.

Modification d'une vue

- Destruction d'une vue:
  - DROP VIEW [IF EXISTS] [nom de la vue]

- Modification d'une vue:
  - ALTER [ALGORITHM = {MERGE | TEMPTABLE | UNDEFINED}] VIEW [nom de la vue] AS [SELECT statement]

Vue matérialisée

Il s’agit d’une table se comportant comme une vue

Lorsque les données changent dans les tables sous jacentes, alors la table « vue » doit changer

N ’existe pas en MySQL

Travaux sur les vues matérialisées

- PostgreSQL Materialized Views, Jonathan Gardner, http://tech.jonathangardner.net/wiki/PostgreSQL/Materialized_Views
- Materialized Views that Really Work by Dan Chak, http://www.pgcon.org/2008/schedule/events/69.en.html
- Flexview, http://code.google.com/p/flexviews/

Algorithme de base

1. créer une « vraie » vue sur les données sous jacentes
2. créer une table effectuant la liaison entre la « vraie » vue et la vue matérialisée
3. créer la vue matérialisée
4. gérer le « rafraichissement » de la vue matérialisée.

Algorithme de base : outil

- Il est utile de pouvoir créer des procédures stockées permettant de gérer cet algorithme.

- Ces procédures en général doivent pouvoir exécuter du SQL.

Execute, les vues et MySQL

- Algorithme de la photo:

```
CREATE TABLE matviews (
  mv_name VARCHAR NOT NULL PRIMARY KEY – nom de la vue materialisé
  , v_name VARCHAR NOT NULL – nom de la vue
  , last_refresh TIMESTAMP WITH TIME ZONE –date du dernier “refresh”
);
```

Création de la vue materialisé via une procedure stocké:
CREATE PROCEDURE createView(in nom_de_vue VARCHAR(50))
Creation d’une table “mat_”nom_de_vue via la commande
Insertion du nom de la vue dans la table matviews
Gestion des log

Objets d'une base MySQL
Base information_schema

- Qu'est ce que la base Information Schema?
- Comment exploiter cette base?

information_schema

- Chaque moteur de base de données stocke les informations des bases de données dans une base de données.
- La base de donnée spécifique à MySQL s'appel information_schema et contient donc les informations sur les bases de données, tables, plugin…
- Cette base de données est initialisé lors de l'installation de MySQL.

Information_schema

- Avant la version 5.0.2 de MySQL, la commande SHOW permettait d'obtenir les métadonnées d'une base. Cette commande a été maintenue (et même enrichie) dans les versions ultérieures, pour des raisons de compatibilités et pour ne pas frustrer les utilisateurs habitués à simplicité de la syntaxe de SHOW.

- L'utilisation du catalogue se fait par des requêtes SQL « classiques » (SELECT ... FROM INFORMATION_SCHEMA... WHERE...), sans faire appel à une syntaxe particulière, et de manière très flexible (toute la syntaxe SQL ainsi que les fonctions de MySQL sont à votre disposition pour préciser vos critères de recherche dans le catalogue).
  Il est possible, en une seule requête, grâce aux jointures, d'obtenir des informations sur différents objets (base de données, tables, index, colonnes...), là où, avec la commande SHOW, il aurait fallu faire plusieurs interrogations successives.

information_schema

- INFORMATION_SCHEMA fournit un accès aux métadonnées de la base de données, aux informations sur le serveur MySQL, telles que le nom d'une base de données ou d'une table, le type de données d'une colonne ou des privilèges d'accès. D'autres termes parfois utilisés pour ces informations sont dictionnaire de données et catalogue système.

- La base de données INFORMATION_SCHEMA contient plusieurs tables en lecture seule. Il s'agit en fait de vues et non de tables de base. Par conséquent, aucun fichier ne leur est associé et vous ne pouvez pas y définir de déclencheurs. En outre, il n'y a pas de répertoire de base de données portant ce nom.
- Bien que vous puissiez sélectionner INFORMATION_SCHEMA en tant que base de données par défaut avec une instruction USE, vous pouvez uniquement lire le contenu des tables et non effectuer d'opérations INSERT, UPDATE ou DELETE sur celles-ci.

DATABASE

- Dans le catalogue, les bases de données sont nommées « schema ». Les informations concernant ces bases sont donc décrites dans la vue système **SCHEMATA**.
- Pour obtenir toutes les informations sur une base précise, utilisez la requête suivante
  - SELECT * FROM INFORMATION_SCHEMA.SCHEMATA WHERE SCHEMA_NAME = 'nom_base';
- Trois informations sont utiles dans cette vue système :
  - SCHEMA_NAME : le nom de la base de données ;
  - DEFAULT_CHARACTER_SET_NAME : le jeu de caractères par défaut ;
  - DEFAULT_COLLATION_NAME : la collation par défaut ;
- Toutes ces informations sont des paramètres fournis lors de la création de la base de données dans la commande CREATE DATABASE.
  Le jeu de caractères et la collation par défaut peuvent être modifiés par la commande ALTER DATABASE.

TABLES

- Les métadonnées des tables sont présentes dans la vue système **TABLES**.
- Pour obtenir toutes les tables d'une base de données, utilisez la requête suivante :
  - SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'nom_base';

TABLES

A chaque ligne renvoyée par la requête correspond une table, une vue ou une table temporaire de la base spécifiée dans la clause WHERE.

Les informations couramment utilisées dans cette vue système sont :

- TABLE_SCHEMA : le nom de la base de donnée ;
- TABLE_NAME : le nom de la table ;
- TABLE_TYPE : cette donnée peut prendre 3 valeurs :
  - TABLE : l'objet est une table ;
  - VIEW : l'objet est une vue. A noter que pour une vue, toutes les autres informations suivantes sont à NULL ;
  - TEMPORARY : l'objet est une table temporaire ;
- ENGINE : le moteur utilisé pour la table (InnoDB, MyISAM...) ;
- TABLE_ROWS : le nombre de lignes contenues dans la table ;
- DATA_LENGTH : taille des données dans la table, en **octets** ;
- MAX_DATA_LENGTH : taille maximale autorisée pour les données, en octets ;
- INDEX_LENGTH : taille des index, en octets ;
- AUTO_INCREMENT : indique si la table possède une colonne auto-incrémentée. Si c'est le cas, la donnée vaut 1, sinon elle vaut NULL ;
- CREATE_TIME : date et heure de création de la table ;
- TABLE_COLLATION : la collation par défaut sur la table. Il n'y a pas d'information sur le jeu de caractères par défaut, mais il peut se déduire de la collation.

information_schema

Rechercher les tables d'une base db5 via information_schema

```
mysql> SELECT table_name, table_type, engine
       FROM information_schema.tables
       WHERE table_schema = 'db5'
       ORDER BY table_name;
+------------+------------+--------+
| table_name | table_type | engine |
+------------+------------+--------+
| fk         | BASE TABLE | InnoDB |
| fk2        | BASE TABLE | InnoDB |
| goto       | BASE TABLE | MyISAM |
| into       | BASE TABLE | MyISAM |
| k          | BASE TABLE | MyISAM |
| kurs       | BASE TABLE | MyISAM |
| loop       | BASE TABLE | MyISAM |
| pk         | BASE TABLE | InnoDB |
| t          | BASE TABLE | MyISAM |
| t2         | BASE TABLE | MyISAM |
| t3         | BASE TABLE | MyISAM |
| t7         | BASE TABLE | MyISAM |
| tables     | BASE TABLE | MyISAM |
| v          | VIEW       | NULL   |
| v2         | VIEW       | NULL   |
| v3         | VIEW       | NULL   |
| v56        | VIEW       | NULL   |
+------------+------------+--------+
17 rows in set (0.01 sec)
```

COLUMNS

Les métadonnées sur les colonnes (ou champs) sont contenues dans la vue système **COLUMNS**.

Pour obtenir toutes les colonnes d'une table dans une base de données, utilisez la requête suivante :

- SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_SCHEMA = 'nom_base' AND TABLE_NAME = 'nom_table' ORDER BY ORDINAL_POSITION;

COLUMNS

Les informations les plus utiles dans cette vue système sont les suivantes :

- TABLE_SCHEMA : le nom de la base de donnée ;
- TABLE_NAME : le nom de la table à laquelle appartient la colonne ;
- COLUMN_NAME : le nom de la colonne ;
- ORDINAL_POSITION : le rang de la colonne dans la table (ordre de présentation) ;
- COLUMN_DEFAULT : la valeur par défaut de la colonne ;
- IS_NULLABLE : indique si la colonne accepte la valeur NULL. Cette information (sous forme de chaîne de caractères) peut prendre 2 valeurs :
  - 'NO' : les valeurs NULL ne sont pas acceptées ;
  - 'YES' : les valeurs NULL sont possibles ;
- DATA_TYPE : le type de la colonne (varchar, char, int, date...) ;
- CHARACTER_MAXIMUM_LENGTH : le nombre maximum de caractères pour les champs de type chaîne de caractères (char, varchar, text...) ;
- CHARACTER_OCTET_LENGTH : cette valeur est généralement égale à la précédente, sauf pour les jeux de caractères multi-octets, comme l'UTF8 (dans ce cas précis, CHARACTER_OCTET_LENGTH = 3 * CHARACTER_MAXIMUM_LENGTH puisque l'UTF8 est codé sur 3 octets) ;
- NUMERIC_PRECISION : contient la précision (déclarée ou implicite) d'un champ de type numérique. Par exemple, la précision d'un champ de type DECIMAL (4, 2) est 4 ;
- NUMERIC_SCALE : contient l'échelle (déclarée ou implicite) d'un champ de type numérique. Par exemple, l'échelle d'un champ de type DECIMAL (4, 2) est 2 ;
- CHARACTER_SET_NAME : le jeu de caractère d'une colonne de type chaîne de caractères ;
- COLUMN_TYPE : le type de la colonne. Par rapport à la donnée DATA_TYPE, cette valeur est complétée du nombre de caractères maximum pour les chaînes de caractères, de la précision pour les entiers, de la précision et de l'échelle pour les nombres décimaux. Cette colonne précise également, pour les entiers, s'ils ont un signe ou non (UNSIGNED) ;
- COLUMN_KEY : donnée, quand elle est renseignée, indiquant que la colonne est indexée ; elle peut prendre 3 valeurs :
  - 'PRI' : la colonne fait partie de la clef primaire de la table ;
  - 'UNI' : la colonne fait partie d'une clef unique ;
  - 'MUL' : la colonne fait partie d'un index non unique (par exemple, parce qu'elle constitue une clef étrangère) ;
- EXTRA : dans cette colonne, on trouve notamment l'information AUTO_INCREMENT quand la colonne est auto-incrémentée ;
- PRIVILEGES : les privilèges dont dispose l'utilisateur qui a exécuté l'interrogation du dictionnaire sur la colonne ;
- COLUMN_COMMENT : un commentaire éventuellement renseigné lors de la création d'une table.

Primary Key

```
--clef primaire d'une table
SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE
TABLE_SCHEMA = 'nom_base' AND TABLE_NAME = 'nom_table' AND
COLUMN_KEY = 'PRI' ORDER BY ORDINAL_POSITION;
```

INDEX

- La vue système **STATISTICS** qui contient les métadonnées sur les index, même si nous venons de voir qu'une partie de cette information se retrouve également dans la vue système décrivant les colonnes.
- SELECT * FROM INFORMATION_SCHEMA.STATISTICS WHERE TABLE_SCHEMA = 'nom_base' AND TABLE_NAME = 'nom_table';

INDEX

Les informations dans cette vue système sont :

- TABLE_SCHEMA : le nom de la base de donnée ;
- TABLE_NAME : le nom de la table à laquelle appartient l'index ;
- NON_UNIQUE : indicateur numérique de non unicité des valeurs de l'index :
  - 0 : les valeurs de l'index doivent être uniques. C'est le cas des index liés aux clefs primaires ou aux contraintes d'unicité ;
  - 1 : les valeurs de la colonne ne doivent pas nécessairement être uniques. C'est le cas des index liés aux clefs étrangères ;
- INDEX_NAME : le nom de l'index. Pour les index de clefs primaires, leur nom est PRIMARY ; pour les clefs étrangères, ils sont, par défaut, préfixés par 'FK_' ;
- SEQ_IN_INDEX : indique l'ordre de la colonne dans l'index. Dans le cas des clefs concaténées sur plusieurs champs, cette donnée indique l'ordre de la colonne dans l'index de la clef primaire. Dans ce dernier cas, l'unicité des valeurs porte donc sur l'ensemble de la clef (chaque colonne n'a pas nécessairement de contrainte d'unicité, mais la combinaison des différents champs est nécessairement unique) ;
- COLUMN_NAME : le nom de la colonne indexée ;
- CARDINALITY : le nombre de lignes indexées ;
- INDEX_TYPE : le type d'index, par exemple BTREE.

Contraintes

Pour obtenir toutes les informations sur les contraintes d'une table, il faut utiliser la combinaison de deux nouvelles vues systèmes de métadonnées :

- **TABLE_CONSTRAINTS**
- **KEY_COLUMN_USAGE**.

La première des deux vues fournit principalement le type de la contrainte, alors que la seconde donne les tables et colonnes qui utilisent cette contrainte.

Contraintes

```
-- clef etrangere
SELECT * FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE
TABLE_SCHEMA = 'nom_base' AND TABLE_NAME = 'nom_table' AND
CONSTRAINT_TYPE = 'FOREIGN KEY';
-- composition de le clef et reference
SELECT k.CONSTRAINT_SCHEMA, k.CONSTRAINT_NAME, k.TABLE_NAME,
k.COLUMN_NAME , k.REFERENCED_TABLE_SCHEMA,
k.REFERENCED_TABLE_NAME, k.REFERENCED_TABLE_NAME FROM
INFORMATION_SCHEMA.KEY_COLUMN_USAGE AS k INNER JOIN
INFORMATION_SCHEMA.TABLE_CONSTRAINTS AS c ON k.CONSTRAINT_SCHEMA
= c.CONSTRAINT_SCHEMA AND k.CONSTRAINT_NAME = c.CONSTRAINT_NAME
WHERE c.CONSTRAINT_TYPE = 'FOREIGN KEY';
```

Contraintes Tables Contraints

Parmi les informations utiles de la vue système **TABLE_CONSTRAINTS**, on trouve :

- CONSTRAINT_SCHEMA : la base de donnée de la contrainte ;
- CONSTRAINT_NAME : le nom de la contrainte (qui est aussi le nom de l'index associé à la contrainte) ;
- TABLE_SCHEMA : le nom de la base de données de la table sur laquelle porte la contrainte (information identique à la base de la contrainte ci-dessus) ;
- TABLE_NAME : le nom de la table à laquelle appartient la contrainte ;
- CONSTRAINT_TYPE : le type de contrainte, qui peut prendre les valeurs suivantes :
  - 'PRIMARY KEY' : clef primaire ;
  - 'UNIQUE' : unicité des valeurs ;
  - 'FOREIGN KEY' : clefs étrangères.

Cependant, cette seule vue système ne permet pas de connaître la table et la colonne référencée par la clef étrangère.

Contraintes Tables Key Column Usage

Les informations supplémentaires qui nous intéressent sont alors les suivantes :

- REFERENCED_TABLE_SCHEMA : base contenant la table référencée par la clef étrangère ;
- REFERENCED_TABLE_NAME : nom de la table référencée par la clef étrangère ;
- REFERENCED_COLUMN_NAME : nom de la colonne référencée.

Procédure Stockées et Fonction

Les métadonnées renvoyées par la vue ROUTINES sont les suivantes :

- SPECIFIC_NAME : le nom de la routine ;
- ROUTINE_SCHEMA : la base de données dans laquelle est définie la procédure ou la fonction ;
- ROUTINE_NAME : le nom de la routine ;
- ROUTINE_TYPE : le type de la routine, qui peut prendre les deux valeurs suivantes :
  - 'FUNCTION' : fonction (procédure stockée renvoyant une valeur) ;
  - 'PROCEDURE' : procédure stockée ;
- DTD_IDENTIFIER : le type de donnée renvoyé (pour les fonctions) ;
- ROUTINE_DEFINITION : le code du corps de la fonction ou de la procédure stockée (toutes les instructions entre les mots clefs BEGIN et END) ;
- CREATED : date et heure de création ;
- LAST_ALTERED : date et heure de dernière modification ;
- DEFINER : l'utilisateur ayant créé la routine.

TRIGGERS

Parmi les données renvoyées par la vue trigers sont :

- TRIGGER_SCHEMA : la base de données dans laquelle figure le déclencheur ;
- TRIGGER_NAME : le nom du déclencheur ;
- EVENT_MANIPULATION : l'évènement sur lequel se déclenche le trigger, qui peut être soir INSERT, soit UPDATE (en MySQL 5.0, il n'existe pas encore de trigger sur l'instruction DELETE);
- EVENT_OBJECT_SCHEMA : la base contenant la table sur laquelle porte le déclencheur ;
- EVENT_OBJECT_TABLE : la table sur laquelle porte le déclencheur ;
- ACTION_ORDER : l'ordre de déclenchement du trigger dans la liste des triggers portant sur le même objet. En version 5.0, cette valeur est limitée à 0, puisqu'il ne peut y avoir qu'un seul déclencheur par table ;
- ACTION_STATEMENT : le code exécuté à l'appel du déclencheur ;
- ACTION_ORIENTATION : en version 5.0, seul les triggers au niveau ligne étant disponibles, cette colonne ne contient que la valeur 'ROW'. Cela devrait évoluer avec les futures versions de MySQL, si les triggers "instruction" (qui se déclenchent une seule fois et non pas pour chaque ligne) sont implémentés ;
- ACTION_TIMING : le moment de déclenchement du trigger, qui peut être soit avant (BEFORE) soit après (AFTER) l'évènement déclenchant ;
- DEFINER : l'utilisateur ayant créé le trigger.

Utilisateurs

La vue système **USER_PRIVILEGES** fournit des informations très générales sur les utilisateurs déclarés.
Ces informations sont issues de la table MYSQL.USER:

- GRANTEE : le nom d'utilisateur et l'hôte de connexion, séparés par un @. Dans MySQL, les droits sont attribués à un nom d'utilisateur, se connectant depuis un hôte (une adresse IP ou '%' pour autoriser une connexion depuis n'importe quel hôte). C'est cette combinaison d'informations qui permet de définir les droits ;
- PRIVILEGE_TYPE : le type de privilèges accordés à l'utilisateur, parmi lesquels on trouve notamment :
  - 'SELECT' : droit de sélection de données ;
  - 'INSERT' : droit d'insertion de données ;
  - 'UPDATE' : droit de mise à jour de données ;
  - 'DELETE' : droit de suppression de données ;
  - 'CREATE' : droit de création d'objets (bases ou tables) ;
  - 'DROP' : droit de suppression d'objets (bases ou tables) ;
  - 'FILE' : lire ou écrire des fichiers sur le serveur (commandes LOAD DATA... INFILE ou SELECT ... INTO OUTFILE) ;
  - 'INDEX' : droit de créer ou supprimer des index ;
  - 'CREATE TEMPORARY TABLES' : droit de créer des tables temporaires ;
  - 'EXECUTE' : droit d'exécuter des procédures stockées ou des fonctions ;
  - 'CREATE VIEW' : droit de créer des vues ;
  - 'CREATE ROUTINE' : droit de créer des procédures stockées, des fonctions ou des triggers ;
  - 'ALTER ROUTINE' : droit de modifier des procédures stockées, des fonctions ou des triggers ;
  - 'CREATE USER' : droit de créer des utilisateurs ;
  - 'USAGE' : équivalent à « aucun privilège », si ce n'est le droit de se connecter au serveur ;
- IS_GRANTABLE : indique si l'utilisateur peut lui-même octroyer les privilèges dont il dispose. Dans la commande GRANT de MySQL, l'option "WITH GRANT OPTION" donne à l'utilisateur la possibilité d'octroyer le privilège qu'il possède à d'autres utilisateurs. C'est cette option qui est décrite ici. Cette information (sous forme de chaîne de caractères) peut prendre 2 valeurs :
  - 'NO' : l'utilisateur ne peut pas octroyer ses privilèges ;
  - 'YES' : l'utilisateur peut octroyer ses privilèges.

Schema Privilèges

La vue système **SCHEMA_PRIVILEGES** donne des informations sur les droits des utilisateurs au niveau de chaque base de données présente sur le serveur.

On y trouve les champs suivants :

- GRANTEE : Le nom de l'utilisateur et son hôte de connexion ;
- TABLE_SCHEMA : le nom de la base de données ;
- PRIVILEGE_TYPE : le type de privilège (voir liste précédente) ;
- IS_GRANTABLE : indique si l'utilisateur peut octroyer ses privilèges.

Table Privilèges

La vue système **TABLE_PRIVILEGES** donne des informations sur les droits des utilisateurs au niveau de chaque table d'une base de données.
On y trouve les champs suivants :

- GRANTEE : Le nom de l'utilisateur et son hôte de connexion ;
- TABLE_SCHEMA : le nom de la base de données ;
- TABLE_NAME : le nom de la table ;
- PRIVILEGE_TYPE : le type de privilège de table (SELECT, INSERT, UPDATE, ALTER, DROP, INDEX et CREATE VIEW) ;
- IS_GRANTABLE : indique si l'utilisateur peut octroyer ses privilèges.

Column Privilèges

La vue système **COLUMN_PRIVILEGES** donne des informations sur les droits des utilisateurs au niveau de chaque colonne.

On y trouve les champs suivants :

- GRANTEE : Le nom de l'utilisateur et son hôte de connexion ;
- TABLE_SCHEMA : le nom de la base de données ;
- TABLE_NAME : le nom de la table ;
- COLUMN_NAME : le nom de la colonne ;
- PRIVILEGE_TYPE : le type de privilège de colonne (SELECT, INSERT, et UPDATE) ;
- IS_GRANTABLE : indique si l'utilisateur peut octroyer ses privilèges.

Connexions, droits d'accès, sécurité

- Authentification des utilisateurs.
- Structure des tables de la base MySQL.
- Gestion des utilisateurs et de leurs privilèges.
- Droits sur les vues et les traitements stockés.
- Utilisation de SSL.

Connexions, droits d'accès, sécurité
Authentification des utilisateurs.

- Structuration de MySQL en plugin
- Plugin d'Authentification
- Plugin de securisation des clefs
- Plugin de Keyring

Authentification des utilisateurs.

MySQL comprend plusieurs composants et plugins qui implémentent des fonctionnalités de sécurité:

- Plugins pour authentifier les tentatives des clients pour se connecter au serveur MySQL
- Plugin de validation de mot de passe pour la mise en œuvre de stratégies de force de mot de passe et l'évaluation de la force de mots de passe potentiels
- Les plugins Keyring qui fournissent un stockage sécurisé pour les informations sensibles.

Un utilisateur MySQL

- MySQL stocke les comptes dans la table utilisateur de la base de données système mysql. Un compte est défini en termes de nom d'utilisateur et de l'hôte client ou des hôtes à partir desquels l'utilisateur peut se connecter au serveur. Pour plus d'informations sur la représentation du compte dans la table des utilisateurs
- Le compte **peut** aussi avoir un mot de passe. MySQL supporte les plugins d'authentification, il est donc possible qu'un compte s'authentifie en utilisant une méthode d'authentification externe

Contraintes et caractéristiques des utilisateurs

- Les noms d'utilisateur, tels qu'ils sont utilisés par MySQL à des fins d'authentification, n'ont rien à voir avec les noms d'utilisateur (noms de connexion) utilisés par Windows ou Unix.
- Les noms d'utilisateurs MySQL peuvent comporter jusqu'à 32 caractères.
- Pour authentifier les connexions client des comptes qui utilisent l'authentification native MySQL (implémentée par le plug-in d'authentification mysql_native_password), le serveur utilise les mots de passe stockés dans la table user. Ces mots de passe sont distincts des mots de passe pour la connexion à votre système d'exploitation.
- Les mots de passe stockés dans la table des utilisateurs sont cryptés à l'aide d'algorithmes spécifiques au plugin.

Se Connecter

Pour vous connecter à un serveur MySQL avec un client de ligne de commande, spécifiez les options de nom d'utilisateur et de mot de passe nécessaires pour le compte que vous souhaitez utiliser:

```
shell> mysql --user=username --password=password db_name
```

Créer des utilisateurs

- Pour créer des comptes MySQL, utilisez les instructions account-management destinées à la création de comptes et à l'établissement de leurs privilèges, telles que CREATE USER et GRANT. Ces instructions amènent le serveur à apporter les modifications appropriées aux tables d'attribution sous-jacentes.
- Pour apporter des modifications, vous devez vous connecter au serveur MySQL en tant qu'utilisateur root MySQL, qui dispose du privilège CREATE USER.

Créer des utilisateurs

- Commencez par utiliser le programme mysql pour vous connecter au serveur en tant qu'utilisateur root MySQL:
- Après vous être connecté au serveur en tant que root, vous pouvez ajouter de nouveaux comptes. L'exemple suivant utilise les instructions CREATE USER et GRANT pour configurer deux comptes:

```
mysql> CREATE USER 'finley'@'localhost' IDENTIFIED
BY 'password';
mysql> GRANT ALL PRIVILEGES ON *.* TO
'finley'@'localhost'
-> WITH GRANT OPTION;
mysql> CREATE USER 'finley'@'%' IDENTIFIED BY
'password';
mysql> GRANT ALL PRIVILEGES ON *.* TO 'finley'@'%'
-> WITH GRANT OPTION;
```

Créer des utilisateurs

Deux comptes ont un nom d'utilisateur finley.

Les deux sont des comptes super-utilisateurs disposant de tous les privilèges.

- Le compte 'finley' @ 'localhost' ne peut être utilisé que lors de la connexion à partir de l'hôte local.
- Le compte 'finley' @ '%' utilise le caractère générique '%' pour la partie hôte. Il peut donc être utilisé pour se connecter à partir de n'importe quel hôte.

Supprimer un utilisateur

- Pour supprimer un compte, utilisez l'instruction DROP USER,

```
mysql> DROP USER 'finley'@'localhost';
```

Utilisateurs systèmes

Lors de l'initialisation du répertoire de données, MySQL crée des comptes utilisateur qui doivent être considérés comme réservés:

- 'root' @ 'localhost: utilisé à des fins administratives. Ce compte a tous les privilèges et peut effectuer n'importe quelle opération. À proprement parler, ce nom de compte n'est pas réservé, en ce sens que certaines installations renomment le compte racine en quelque chose d'autre pour éviter d'exposer un compte hautement privilégié avec un nom connu.
- 'mysql.sys'@'localhost': Utilisé comme DEFINER pour les objets de schéma sys. L'utilisation du compte mysql.sys évite les problèmes si un administrateur de base de données renomme ou supprime le compte root. Ce compte est verrouillé afin qu'il ne puisse pas être utilisé pour les connexions client.
- 'mysql.session'@'localhost': Utilisé en interne par des plugins pour accéder au serveur. Ce compte est verrouillé afin qu'il ne puisse pas être utilisé pour les connexions client.
- 'mysql.infoschema'@'localhost': utilisé comme DEFINER pour les vues INFORMATION_SCHEMA. L'utilisation du compte mysql.infoschema évite les problèmes si un administrateur de base de données renomme ou supprime le compte racine. Ce compte est verrouillé afin qu'il ne puisse pas être utilisé pour les connexions client.

Limitation des ressources

- Un moyen de restreindre l'utilisation des ressources du serveur MySQL par le client consiste à définir la variable système globale max_user_connections sur une valeur différente de zéro.
- Cela limite le nombre de connexions simultanées pouvant être établies par un compte donné, mais n'impose aucune limite à ce qu'un client peut faire une fois connecté.
- De plus, la définition de max_user_connections ne permet pas la gestion de comptes individuels.

Limitation des ressources

Pour résoudre ces problèmes, MySQL autorise des limites pour l'utilisation de ces ressources de serveur pour des comptes individuels:

- Le nombre de requêtes qu'un compte peut émettre par heure
- Le nombre de mises à jour qu'un compte peut émettre par heure
- Le nombre de fois qu'un compte peut se connecter au serveur par heure
- Le nombre de connexions simultanées au serveur par un compte

Limitation des ressources

- Pour établir des limites de ressources pour un compte au moment de la création du compte, utilisez l'instruction CREATE USER. Pour modifier les limites d'un compte existant, utilisez ALTER USER. Fournissez une clause WITH qui nomme chaque ressource à limiter. La valeur par défaut pour chaque limite est zéro (aucune limite). Par exemple, pour créer un nouveau compte pouvant accéder à la base de données client, mais uniquement de manière limitée, émettez les instructions suivantes:

```
mysql> CREATE USER 'francis'@'localhost' IDENTIFIED BY 'frank'
-> WITH MAX_QUERIES_PER_HOUR 20
-> MAX_UPDATES_PER_HOUR 10
-> MAX_CONNECTIONS_PER_HOUR 5
-> MAX_USER_CONNECTIONS 2;
```

Connexions, droits d'accès, sécurité Authentification des utilisateurs.

- Comment les utilisateurs sont authentifier sur MySQL?
- Qu'est ce qu'un plugin d'authentification?

Plugin Authentification

- Lorsqu'un client se connecte au serveur MySQL, ce dernier utilise le nom d'utilisateur fourni par le client et l'hôte du client pour sélectionner la ligne de compte appropriée dans la table système mysql.user. Le serveur authentifie ensuite le client, en déterminant dans la ligne de compte quel plug-in d'authentification s'applique au client
- Si le serveur ne trouve pas le plug-in, une erreur se produit et la tentative de connexion est rejetée.
- Sinon, le serveur appelle ce plug-in pour authentifier l'utilisateur et le plug-in lui renvoie un statut indiquant si l'utilisateur a fourni le mot de passe correct et s'il est autorisé à se connecter.

Deux Fonctionnalités

- Les plugins d'authentification permettent aux administrateurs de bases de données de choisir et de modifier facilement la méthode d'authentification utilisée pour les comptes MySQL individuels.
- Les plugins d'authentification permettent aux clients de se connecter au serveur MySQL avec les informations d'identification appropriées pour les méthodes d'authentification stockant les informations d'identification ailleurs que dans la table système mysql.user. Par exemple, des plugins peuvent être créés pour utiliser des méthodes d'authentification externes telles que PAM, ID de connexion Windows, LDAP ou Kerberos.

Utilisation des plugins

- Un plug-in d'authentification est spécifié par utilisateur avec une extension de la syntaxe GRANT ou CREATE USER
  - GRANT <privileges> ON <level> TO <user> IDENTIFIED VIA <plugin> [USING <string>]
  - CREATE USER <user> IDENTIFIED VIA <plugin> [USING <string>]
- La clause USING clause fournit les informations d'authentification pour un plugin, son format est complètement défini par le plugin

Utilisation des plugins

- En utilisant le plugin natif, par exemple on crée l'utilisateur qui seras authentifié via le plugin natif
- CREATE USER mysqltest_up1 IDENTIFIED VIA mysql_native_password USING '*E8D46CE25265E545D225A8A6F1BAF642FEBEE5CB';

Plugin d'Authentification

- Le plugin mysql_native_password implémente l'authentification en fonction de cette méthode de hachage de mot de passe natif.
- Plug-ins SHA-256 effectuant l'authentification à l'aide du hachage de mot de passe SHA-256. C'est un cryptage plus fort que celui disponible avec l'authentification native.
- Un plug-in côté client qui envoie le mot de passe au serveur sans hachage ni chiffrement.
- Un plugin qui effectue une authentification externe en utilisant PAM (Pluggable Authentication Modules), permettant au serveur MySQL d'utiliser PAM pour authentifier les utilisateurs MySQL.

Plugin d'Authentification

- Un plug-in qui effectue une authentification externe sur Windows, permettant à MySQL d'utiliser des services Windows natifs pour authentifier les connexions client. Les utilisateurs qui se sont connectés à Windows peuvent se connecter à partir du client MySQL au serveur en fonction des informations de leur environnement, sans spécifier de mot de passe supplémentaire.
- Plugins effectuant l'authentification à l'aide de LDAP (Lightweight Directory Access Protocol) pour authentifier un utilisateur MySQL

MySQL Native Plugin

- MySQL inclut le plugin mysql_native_password qui implémente l'authentification native. c'est-à-dire une authentification basée sur la méthode de hachage du mot de passe utilisée avant l'introduction de l'authentification enfichable.
- Le plugin mysql_native_password existe sous les formes serveur et client: Le plug-in côté serveur est intégré au serveur, ne doit pas être chargé explicitement et ne peut pas être désactivé en le déchargeant.
- Le plug-in côté client est intégré à la bibliothèque client libmysqlclient et est disponible pour tout programme lié à libmysqlclient.

SHA 256 Native Plugin

- Le plugin sha256_password existe dans le serveur et leclient:
  - Le plug-in côté serveur est intégré au serveur, ne doit pas être chargé explicitement et ne peut pas être désactivé en le déchargeant.
  - Le plug-in côté client est intégré à la bibliothèque client libmysqlclient et est disponible pour tout programme lié à libmysqlclient.
- Pour configurer un compte qui utilise le plug-in sha256_password pour le hachage du mot de passe SHA-256, utilisez l'instruction suivante, où mot de passe correspond au mot de passe du compte souhaité:
  - CREATE USER 'sha256user'@'localhost' IDENTIFIED WITH sha256_password BY '*password*';

SHA256 Native Plugin

- Pour démarrer le serveur avec le plug-in d'authentification par défaut défini sur sha256_password, placez ces lignes dans le fichier d'options du serveur:
  - [mysqld]
  - default_authentication_plugin=sha256_password
- Le plugin sha256_password est alors utilisé par défaut pour les nouveaux comptes. En conséquence, il est possible de créer le compte et de définir son mot de passe sans nommer explicitement le plug-in:
  - CREATE USER 'sha256user'@'localhost' IDENTIFIED BY '*password*';

Windows Native Plugin

- MySQL Server peut utiliser les services Windows natifs pour authentifier les connexions client. Les utilisateurs qui se sont connectés à Windows peuvent se connecter à partir du client MySQL au serveur en fonction des informations de leur environnement, sans spécifier de mot de passe supplémentaire.
- Le plugin doit être donnée à chaque démarrage du serveur.
  - [mysqld]
  - plugin-load-add=authentication_windows.dll

Windows Native Plugin

- Les utilisateurs Windows, Rafal et Tasha, peuvent être autorisés à se connecter à MySQL, ainsi que tous les utilisateurs du groupe Administrators ou Power Users. Pour ce faire, créez un compte MySQL nommé sql_admin qui utilise le plug-in Windows pour l'authentification:
  - CREATE USER sql_admin IDENTIFIED WITH authentication_windows AS 'Rafal, Tasha, Administrators, "Power Users"';
- Après avoir créé le compte sql_admin, un utilisateur connecté à Windows peut tenter de se connecter au serveur à l'aide de ce compte:
  - C:\> mysql --user=sql_admin

Windows Native Plugin

- Aucun mot de passe n'est requis ici. Le plugin authentication_windows utilise l'API de sécurité Windows pour vérifier quel utilisateur Windows se connecte.
- Si cet utilisateur s'appelle Rafal ou Tasha, ou fait partie du groupe Administrator ou Power User, le serveur accorde l'accès et le client est authentifié en tant que sql_admin et dispose des privilèges octroyés au compte sql_admin. Sinon, le serveur refuse l'accès.

Connexions, droits d'accès, sécurité
Gestion des utilisateurs et de leurs privilèges

- Privilège des utilisateurs
- Commande GRANT et REVOKE
- Mise en place des rôles

Gestion des droits et privilèges

- MySQL implémente un système sophistiqué de contrôle d'accès et de privilèges vous permettant de créer des règles d'accès complètes pour la gestion des opérations client et d'empêcher efficacement les clients non autorisés d'accéder au système de base de données.

Type de privilège sous MySQL

La fonction principale du système de privilèges MySQL est d'authentifier un utilisateur qui se connecte à partir d'un hôte donné et de lui associer des privilèges sur une base de données telle que SELECT, INSERT, UPDATE et DELETE.

Il y a des choses que vous ne pouvez pas faire avec le système de privilège MySQL:

- Vous ne pouvez pas spécifier explicitement qu'un utilisateur donné doit se voir refuser l'accès. Autrement dit, vous ne pouvez pas explicitement faire correspondre un utilisateur et ensuite refuser la connexion.
- Vous ne pouvez pas spécifier qu'un utilisateur dispose des privilèges pour créer ou supprimer des tables dans une base de données, mais pas pour créer ou supprimer la base de données elle-même. Un mot de passe s'applique globalement à un compte.
- Vous ne pouvez pas associer un mot de passe à un objet spécifique tel qu'une base de données, une table ou une routine.

Schema Mysql

- La base de données mysql contient cinq tables de privilèges. Vous manipulez ces tables indirectement via des instructions telles que GRANT et REVOKE.
  - user: contient les colonnes compte d'utilisateur et privilèges globaux. MySQL utilise la table user pour accepter ou refuser une connexion depuis un hôte. Un privilège accordé dans la table user est effectif pour toutes les bases de données sur le serveur MySQL.
  - db: contient les privilèges de niveau base de données. MySQL utilise la table db pour déterminer la base de données à laquelle un utilisateur peut accéder et à partir de quel hôte. Un privilège accordé au niveau de la base de données dans la table de base de données s'applique à la base de données et tous les objets appartiennent à cette base de données, par exemple des tables, des déclencheurs, des vues, des procédures stockées, etc.

Schema Mysql

- table_priv et columns_priv: contient les privilèges au niveau table et au niveau colonne. Un privilège accordé dans la table table_priv s'applique à la table et à ses colonnes, tandis qu'un privilège accordé à la table columns_priv s'applique uniquement à une colonne spécifique d'une table.
- procs_priv: contient les privilèges de fonctions et de procédures stockées

Type de privilège sous MySQL

Les privilèges accordés à un compte MySQL déterminent les opérations que le compte peut effectuer. Les privilèges MySQL diffèrent selon le contexte d'application et le niveau de fonctionnement:

- administratifs permettent aux utilisateurs de gérer le fonctionnement du serveur MySQL. Ces privilèges sont globaux car ils ne sont pas spécifiques à une base de données particulière.
- base de données s'appliquent à une base de données et à tous les objets qu'elle contient. Ces privilèges peuvent être accordés pour des bases de données spécifiques ou globalement, de sorte qu'ils s'appliquent à toutes les bases de données.
- pour des objets de base de données tels que des tables, des index, des vues et des routines stockées peuvent être accordés pour des objets spécifiques dans une base de données

Privileges

- ALL, ALL PRIVILEGES:Ces spécificateurs de privilège sont des raccourcis pour «tous les privilèges disponibles à un niveau de privilège donné»
- ALTER Permet d'utiliser l'instruction ALTER TABLE pour modifier la structure des tables. ALTER TABLE requiert également les privilèges CREATE et INSERT. Renommer une table nécessite ALTER et DROP sur l'ancienne table, CREATE et INSERT sur la nouvelle table.
- ALTER ROUTINE Permet l'utilisation d'instructions qui modifient ou suppriment les routines stockées
- CREATE Permet l'utilisation d'instructions qui créent de nouvelles bases de données et tables.

Privilèges

- CREATE ROLE. Active l'utilisation de l'instruction CREATE ROLE. (Le privilège CREATE USER permet également d'utiliser l'instruction CREATE ROLE.)
- CREATE ROUTINE Permet l'utilisation d'instructions qui créent des routines stockées (procédures stockées et fonctions). table, telle que DROP TABLE, INSERT, UPDATE ou SELECT.
- CREATE USER Active l'utilisation des instructions ALTER USER, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, RENAME USER et REVOKE ALL PRIVILEGES.
- CREATE VIEW Active l'utilisation de l'instruction CREATE VIEW. DELETE Permet aux lignes d'être supprimées des tables d'une base de données.
- DROP Permet l'utilisation d'instructions qui suppriment (suppriment) les bases de données, les tables et les vues existantes. Le privilège DROP est requis pour utiliser l'instruction ALTER TABLE ...

Privilèges

- DROP ROLE. (Le privilège CREATE USER permet également d'utiliser l'instruction DROP ROLE.) Vous permet d'accorder ou de révoquer à d'autres utilisateurs les privilèges que vous possédez vous-même.
- INDEX Permet l'utilisation d'instructions qui créent ou suppriment (suppriment) des index. INDEX s'applique aux tables existantes. Si vous disposez du privilège CREATE pour une table, vous pouvez inclure des définitions d'index dans l'instruction CREATE TABLE.
- INSERT Permet aux lignes d'être insérées dans les tables d'une base de données. INSERT est également requis pour les instructions table-maintenance ANALYZE TABLE, OPTIMIZE TABLE et REPAIR TABLE.
- LOCK TABLES Permet l'utilisation d'instructions explicites LOCK TABLES pour verrouiller les tables pour lesquelles vous disposez du privilège SELECT. Cela inclut l'utilisation de verrous en écriture, ce qui empêche d'autres sessions de lire la table verrouillée.

Privilèges

- SELECT Permet aux lignes d'être sélectionnées à partir des tables d'une base de données.
- SHUTDOWN Active l'utilisation des instructions SHUTDOWN et RESTART,
- TRIGGER Active les opérations de déclenchement. Vous devez disposer de ce privilège pour qu'une table crée, supprime, exécute ou affiche des déclencheurs pour cette table.
- UPDATE Permet aux lignes d'être mises à jour dans les tables d'une base de données.

Privilèges

	Serveur	Base de Données	Table	Colonne	Procedure
ALL					
ALTER	■	■	■		
ALTER ROUTINE	■	■			■
CREATE	■	■	■		
CREATE ROUTINE	■	■			
CREATE USER	■				
CREATE VIEW	■	■	■		
DELETE	■	■	■		
DROP	■	■	■		
EXECUTE	■	■	■		
GRANT OPTION	■	■	■		■
INDEX	■	■	■		
INSERT	■	■	■	■	
SELECT	■	■	■	■	
SHUTDOWN	■				
TRIGGER	■	■	■		
UPDATE	■	■	■	■	

GRANT

Après avoir créé un nouveau compte d'utilisateur, l'utilisateur ne dispose d'aucun privilège. Pour accorder des privilèges à un compte d'utilisateur, utilisez l'instruction GRANT. Voici une illustration de la syntaxe de l'instruction GRANT:

GRANT privilege,[privilege],.. ON privilege_level

TO user [IDENTIFIED BY password]

[REQUIRE tsl_option]

[WITH [GRANT_OPTION | resource_option]];

GRANT

- Tout d'abord, spécifiez un ou plusieurs privilèges après le mot clé GRANT. Si vous accordez plusieurs privilèges à l'utilisateur, chaque privilège est séparé par une virgule.
- Ensuite, spécifiez le niveau de privilège qui détermine le niveau auquel les privilèges s'appliquent. MySQL supporte les niveaux global (*. *), Base de données (database. *), Table (database.table) et colonne. Si vous utilisez le niveau de privilège de la colonne, vous devez spécifier une colonne ou une liste de colonnes séparées par des virgules après chaque privilège.
- Ensuite, placez l'utilisateur que vous souhaitez accorder des privilèges. Si l'utilisateur existe déjà, l'instruction GRANT modifie ses privilèges.

GRANT

Ensuite, vous indiquez si l'utilisateur doit se connecter au serveur de base de données via une connexion sécurisée telle que SSL, X059, etc.

Enfin, la clause facultative WITH GRANT OPTION vous permet d'accorder aux autres utilisateurs ou de supprimer d'autres utilisateurs les privilèges que vous possédez.

Pour accorder tous les privilèges au compte d'utilisateur super @ localhost, utilisez l'instruction suivante

- GRANT ALL ON *.* TO 'super'@'localhost' WITH GRANT OPTION

La clause ON *. * Désigne toutes les bases de données et tous les objets des bases de données. WITH GRANT OPTION permet à super @ localhost d'accorder des privilèges à d'autres utilisateurs.

GRANT

Pour créer un utilisateur disposant de tous les privilèges dans la base de données exemple classicmodels, utilisez les instructions suivantes:

- CREATE USER auditor @ localhost IDENTIFIÉ PAR 'baleine'; GRANT ALL on classicmodels. * TO auditor @ localhost;

Vous pouvez accorder plusieurs privilèges dans une seule instruction GRANT. Par exemple, vous pouvez créer un utilisateur capable d'exécuter les instructions SELECT, INSERT et UPDATE sur la base de données classicmodels à l'aide des instructions suivantes:

- CREATE USER rfc IDENTIFIED BY 'shark'; GRANT SELECT, UPDATE, DELETE ON classicmodels. * TO rfc;

REVOKE

Afin de révoquer les privilèges d'un compte utilisateur, vous utilisez l'instruction MySQL REVOKE. MySQL vous permet de révoquer un ou plusieurs privilèges ou tous les privilèges d'un utilisateur.

Ce qui suit illustre la syntaxe de révocation de privilèges spécifiques d'un utilisateur:

REVOKE privilege_type [(column_list)] [, priv_type [(column_list)]] ... ON [type d'objet] privilege_level DE l'utilisateur [, utilisateur] ...

- Tout d'abord, spécifiez une liste de privilèges que vous souhaitez révoquer d'un utilisateur juste après le mot clé REVOKE. Vous devez séparer les privilèges par des virgules.
- Deuxièmement, spécifiez le niveau de privilège auquel les privilèges sont révoqués dans la clause ON.
- Troisièmement, spécifiez le compte d'utilisateur pour lequel vous souhaitez révoquer les privilèges dans la clause FROM.

Notez que pour révoquer les privilèges d'un compte d'utilisateur, vous devez disposer du privilège GRANT OPTION et des privilèges que vous révoquez.

Roles MySql

Pour faciliter les choses, MySQL a fourni un objet appelé role, qui est une collection nommée de privilèges.

Si vous souhaitez accorder le même ensemble de privilèges à plusieurs utilisateurs, procédez comme suit:

- Tout d'abord, créez un nouveau rôle.
- Deuxièmement, accordez des privilèges au rôle.
- Troisièmement, accordez le rôle aux utilisateurs. Si vous souhaitez modifier les privilèges des utilisateurs, vous devez modifier uniquement les privilèges du rôle attribué.

Les modifications prendront effet pour tous les utilisateurs auxquels le rôle a été attribué.

Create Role

Supposons que vous développiez une application utilisant unebase de données CRM.

Pour interagir avec la base de données CRM, vous devez créer des comptes pour les développeurs ayant besoin d'un accès complet à la base de données.

En outre, vous devez créer des comptes pour les utilisateurs n'ayant besoin que d'un accès en lecture et les autres utilisateurs ayant besoin d'un accès en lecture / écriture.

Pour éviter d'accorder des privilèges à chaque compte d'utilisateur individuellement, vous créez un ensemble de rôles et accordez les rôles appropriés à chaque compte d'utilisateur.
Pour créer de nouveaux rôles, utilisez l'instruction CREATE ROLE:

- CREATE ROLE crm_dev, crm_read, crm_write;

Grant Role

Pour accorder des privilèges à un rôle, vous utilisez l'instruction GRANT.

L'instruction suivante accorde tous les privilèges au rôle crm_dev:

- GRANT ALL ON crm. * TO crm_dev;

L'instruction suivante accorde le privilège SELECT au rôle crm_read: 1

- GRANT SELECT ON crm. * TO crm_read;

L'instruction suivante accorde les privilèges INSERT, UPDATE et DELETE au rôle crm_write:

- GRANT INSERT, UPDATE, DELETE ON crm. * TO crm_write;

Associer des utilisateurs aux rôles

Supposons que vous ayez besoin d'un compte utilisateur en tant que développeur, d'un compte utilisateur pouvant disposer d'un accès en lecture seule et de deux comptes utilisateur pouvant disposer d'un accès en lecture / écriture. Pour créer de nouveaux utilisateurs, utilisez les instructions CREATE USER comme suit:

- CREATE USER crm_dev1 @ localhost IDENTIFIED BY 'Secure $ 1782';

utilisateur d'accès en lecture

- CREATE USER crm_read1 @ localhost IDENTIFIED BY 'Secure $ 5432';

utilisateurs en lecture / écriture

- CREATE USER crm_write1 @ localhost IDENTIFIED BY 'Secure $ 9075';
- CREATE USER crm_write2 @ localhost IDENTIFIED BY 'Secure $ 3452';

Pour attribuer des rôles aux utilisateurs, vous utilisez l'instruction GRANT:

- GRANT crm_dev TO crm_dev1 @ localhost;
- GRANT crm_read TO crm_read1 @ localhost;
- GRANT crm_read, crm_write TO crm_write1 @ localhost, crm_write2 @ localhost;

Associer des rôles par défaut

Maintenant, si vous vous connectez à MySQL à l'aide du compte utilisateur crm_read1 et essayez d'accéder à la base de données CRM:

- mysql -u crm_read1 -p
- Entrer le mot de passe: **********
- mysql> USE crm;

L'instruction a émis le message d'erreur suivant:

- ERREUR 1044 (42000): Accès refusé pour l'utilisateur 'crm_read1' @ 'localhost' à la base de données 'crm'

En effet, lorsque vous attribuez des rôles à un compte d'utilisateur, les rôles ne deviennent pas automatiquement actifs lorsque le compte d'utilisateur se connecte au serveur de base de données.

Rôle par défaut

Si vous appelez la fonction CURRENT_ROLE ():

- SELECT current_role ();
- | NONE|

NONE est renvoyé, c'est-à-dire aucun rôle actif.

Pour spécifier les rôles qui doivent être actifs chaque fois qu'un compte utilisateur se connecte au serveur de base de données, vous utilisez l'instruction SET DEFAULT ROLE.

L'instruction suivante définit la valeur par défaut pour le compte crm_read1 @ localhost avec tous ses rôles attribués.

- SET DEFAULT ROLE ALL TO crm_read1 @ localhost;

Rôle Actif

Un compte utilisateur peut modifier les privilèges effectifs de l'utilisateur actuel dans la session en cours en spécifiant les rôles attribués actifs.

- SET ROLE NONE L'instruction suivante définit le rôle actif sur NONE, ce qui signifie qu'aucun rôle actif.
- SET ROLE ALL Pour définir les rôles actifs sur tous les rôles attribués, vous utilisez
- SET DEFAULT ROLE Pour définir les rôles actifs sur les rôles par défaut définis par l'instruction
- Pour définir des rôles nommés actifs, vous utilisez: 1 SET ROLE grant_role_1, grant_role_2, ...

Connexions, droits d'accès, sécurité
Type de droits

- Quel sont les types de droits?
- Quand les droits prennent effet?

Droits d'administration

Les droits d'administration permettent d'administrer le serveur MySQL. Il doivent en principe être reservés aux comptes administrateur.

Ils sont définis au niveau global, cad à tous les objet du serveur.

Droit Administration

Droit	Description
CREATE TEMPORARY TABLES	Crée de tables temporaires
CREATE USER	Créer, modifier, supprimer les comptes utilisateurs (CREATE, DROP, RENAME)
FILE	Lire ou écire dans les fichiers stockés sur la machine hôte du serveur MySQL(OUTFILE, LOAD DATA)
GRANT OPTION	Permet de transmettre ses droits à d'autres comptes
LOCK TABLES	Verrouiller les tables
PROCESS	Afficher les threads
RELOAD	Réinitialiser les journaux, les tables, les statistiques (FLUSH,RESET)
REPLICATION CLIENT	Superviser et gérer la réplication (SHOW MASTER STATUS, SHOW SLAVE STATUS)

Droit Administration

Droit	Description
REPLICATION SLAVE	Utilisé par le compte de réplication pour récupéré les évènements du journal binaire du maître
SHUTDOWN	Arrreter le serveur
SUPER	Executer diverses commandes d'administration (CHANGE MASTER TO, KILL, SET GLOBAL..)

Droits au niveau des schémas

Il est parfois nécéssaire de définir des droits sur la globalité d'un schéma. Les droits portent alors sur l'ensemble des objects appartenant à ce schéma.

Droit Schémas

Droit	Description
ALTER	Modifier les schémas et les tables (ALTER SCHEMA/DATABASE/TABLE)
CREATE	Créer des schémas et des tables (CREATE SCHEMA/DATABASE/TABLE)
CREATE TEMPORARY TABLE	Créer des tables temporaires (CREATE TEMPORARY TABLES)
CREATE VIEW	Créer des vues (CREATE VIEW)
DELETE	Effeacer des enregistrements d'une table (DELETE)
DROP	Supprimer des schémas ou des tables (DROP SCHEMA/DATABASE/TABLE)
EVENT	Programmer les évènements de l'ordonnancer d'évènements (CREATE EVENT)
GRNT OPTION	Permet de transmettre ses droits à d'autres comptes

Droit Schémas

Droit	Description
INDEX	Créer et supprimer des index (CREATE/DROP/INDEX)
INSERT	Insérer des enregistrements dans une table (INSERT)
LOCK TABLES	Verrouiller les tables (LOCK TABLES)
SELECT	Afficher les enregistrements d'une tables (SELECT,DESCRIBE,SHOW,CREATE TABLE)
SHOW VIEW	Voir le code SQL d'une vue (SHOW CREATE VIEW)
UPDATE	Modifier des enregistrements (UPDATE)

Droits au niveau des tables

Il peut arriver que l'application ne fasse qu'insérer des données dans une table. Dans ce case, il est possible de spécifier des droits pour ne permettre de manipuler que la table et ses données.

Droits au niveau des tables

Droit	Description
ALTER	Modifier les tables (ALTER TABLE)
CREATE	Créer des tables (CREATE TALBE)
DELETE	Effacer des enregistrments d'une table (DELETE)
DROP	Supprimer des tables (DROP TABLE)
GRANT OPTION	Permet de transmettre ses droits à d'autres comptes
INDEX	Créer et supprimer des index (CREATE/DROP INDEX)
INSERT	Insérer des enregistrements dans une table
SELECT	Afficher les enregistrements d'une table
TRIGGER	Créer et supprimer des triggers (CREATE/DROP TRIGGER)
UPDATE	Modifier des enregistrements

Droits au niveau des colonnes

LA granularité la plus fine donne la possibilité d'ajouter, d'afficher et de modifier les données de certaines colonnes d'une table.

Les droits sur les colonnes empêchent les requêtes d'être mise dans le cache de requêtes.

Il n'est pas possible d'avoir le droit DELETE pour une colonne parce que la granularité d'un effacement est l'enregistrement.

Droits au niveau des colonnes

Droit	Description
INSERT	Insérer des enregistrements dans une table
SELECT	Affichier les enregistrement d'une table
UPDATE	Mdoifier des enregistrements

Droits sur les routines

LE système de droits s'applique également pour contrôler si un utilisateur peut créer, modifier ou supprimer des routines stockées

Droit	Description
CREATE ROUTINE	Créer des procédures et des fonctions stockées
ALTER ROUTINE	Modifier ou supprimer des procédures et des fonctions stockées
EXECUTE	Exécuter des procédures et des fonctions stockées
GRANT OPTION	Permet de transmettre ses droits à d'autre comptes.

Sécurisation des vues et des routines

L'accès aux objets de la base de données est sécurisé au niveau des comptes utilisateurs avec un système qui vous y autorise ou non.

Cependant, même si vous avez le droit pour accéder à l'objet, il n'est pas certain que vous ayez le droit d'accéder au données auxquelles il se réfère.

Pour exécuter une vue, il faut que l'utilisateur ait le droit de manipuler les objets de la vue.

Pour une vue, c'est la clause SQL SECURITY qui permet de paramétrer le comportement des objets sous jacent de la vue.

SQL SECURITY

- INVOKER: la vue est exécutée avec les droits de l'utilisateur
- DEFINER: la vue est exécutée avec les droits de son créateur
    - CREATE SQL SECURITY DEFINER VIEW <view name> as select…

- LA clause SQL SECURITY existe également pour les routines, son comportement est similaire.

FLUSH PRIVILEGES

Si vous modifiez directement les tables d'attribution à l'aide d'instructions telles que INSERT, UPDATE ou DELETE, vos modifications n'auront aucune incidence sur la vérification des privilèges jusqu'à ce que vous redémarriez le serveur ou que vous lui indiquiez de recharger les tables. Si vous modifiez directement les tables de droits sans oublier de les recharger, vos modifications sont sans effet tant que vous n'avez pas redémarré le serveur. Cela peut vous amener à vous demander pourquoi vos modifications ne semblent faire aucune différence!

Pour que le serveur recharge les tables d'attribution, effectuez une opération de vidage des privilèges. Cela peut être fait en émettant une instruction FLUSH PRIVILEGES ou en exécutant une commande mysqladmin flush-privileges ou mysqladmin reload.

Si vous modifiez les tables d'attribution indirectement à l'aide d'instructions de gestion de compte telles que GRANT, REVOKE, SET PASSWORD ou RENAME USER, le serveur remarque ces modifications et charge immédiatement les tables d'attribution en mémoire.

Connexions, droits d'accès, sécurité
Utilisation de SSL.

- Mise en place de SSL dans MySQL
- Associer un utilisateur a une connectivité SSL

Chiffrement des informations

- Les informations qui transitent entre un serveur et ses clients circulent en clair et peuvent être facilement interceptées.
- Pour contrer cela, il est possible de chiffrer la communication.
- Il existe plusieurs manières de créer un certificats. Deux utilitaires de création de certificat sont possibles, mysql_ssl_rsa_setup issu des outils MySQL et OpenSSL qui est une alternative du système.

mysql_ssl_rsa_setup

- L'utilitaire mysql_ssl_rsa_setup est un outil client de MySQL qui permet de générer automatiquement toutes les clefs nécéssaires à la mise en œuvre d'une connexion chiffrée.
- Shell> mysql_ssl_rsa_setup --datadir=/var/lib/mysql
- Le résultat est ainsi :

```
Generating a 2048 bit RSA private key
....+++
.......+++
writing new private key to 'ca-key.pem'
-----
Loading 'screen' into random state - done
Generating a 2048 bit RSA private key
.....+++
.......+++
writing new private key to 'server-key.pem'
-----
Loading 'screen' into random state - done
Generating a 2048 bit RSA private key
....+++
..+++
writing new private key to 'client-key.pem'
-----
```

mysql_ssl_rsa_setup

- Les clefs sont ensuite bien créées dans le repertoire de destination (ici sous windows)

```
Directory of D:\mysql-5.7.10-winx64\data

01/13/2016  04:19 PM    <DIR>          .
01/13/2016  04:19 PM    <DIR>         ..
01/13/2016  04:16 PM                56 auto.cnf
01/13/2016  04:19 PM             1,679 ca-key.pem
01/13/2016  04:19 PM             1,074 ca.pem
01/13/2016  04:19 PM             1,078 client-cert.pem
01/13/2016  04:19 PM             1,675 client-key.pem
...
01/13/2016  04:19 PM             1,675 private_key.pem
01/13/2016  04:19 PM               451 public_key.pem
01/13/2016  04:19 PM             1,078 server-cert.pem
01/13/2016  04:19 PM             1,679 server-key.pem
```

OpenSSL

- L'utilitaire musql_ssl_rsa_setup vu précédemment générer des certificats auto signés. Une possibilité consiste à utiliser OpenSSL, en générant une autorité de certification.

```
# Créer un certificat de CA
openssl genrsa 2048 > ca-key.pem
openssl req -new -x509 -nodes -days 3600 \
        -key ca-key.pem -out ca.pem

# Créer un certificat de serveur, supprimer le mot de passe  et le signer
# server-cert.pem = clé publique, server-key.pem = clé privée
openssl req -newkey rsa:2048 -days 3600 \
        -nodes -keyout server-key.pem -out server-req.pem
openssl rsa -in server-key.pem -out server-key.pem
openssl x509 -req -in server-req.pem -days 3600 \
        -CA ca.pem -CAkey ca-key.pem -set_serial 01 -out server-cert.pem
```

OpenSSL

- Puis vient la génération de la clef du client

```
#Créer un certificat client, supprimer le mot de passe  et le
signer
# client-cert.pem = public key, client-key.pem = private key


openssl req -newkey rsa:2048 -days 3600 \
        -nodes -keyout client-key.pem -out client-req.pem
openssl rsa -in client-key.pem -out client-key.pem
openssl x509 -req -in client-req.pem -days 3600 \
        -CA ca.pem -CAkey ca-key.pem -set_serial 01 -out
client-cert.pem
```

Configuration du mode SSL

Une fois les certificats générés, le serveur doit prendre en compte les clés générées et permettre aux utilisateurs de chiffrer leur connexion.

Les étapes de configuration du serveur sont ainsi:

- Création d'un utilisateur utilisateur SSL.
- Configuration du serveur
- Configuration du client

Création d'un utilisateur SSL

- Un compte utilisateur doit être utilisé pour se connecter en SSL.

CREATE USER "neoflow"@"%" IDENTIFIED BY "motdepasse" REQUIRE SSL;

- Si l'utilisateur existe déjà le contenu de la table mysql.user indique si oui ou non il peut se connecter en SSL

Select user,Host,ssl_type from mysql.user where User='neoflow'

- Et lui rajouter les droits

Alter user 'newflow'@'%' require SSL

- Enfin forcer la modification via

FLUSH PRIVILEGES

Configuration SSL du serveur

- Par défaut, le serveur MySQL autorise les connexions SSL(option –ssl).
- Ce dernier doit prendre en comtpe les certificats générés dans le fichier my.cnf et dans la section mysqld

```
[mysqld]
ssl-ca=ca.pem
ssl-cert=server-cert.pem
ssl-key=server-key.pem
```

Configuration du client

- Il faut en premier lieu copier du serveur vers le client le certificats, et les clefs clients (ca.pem, client-cert.pem, client-key.pem)
- Puis soit modifier le fichier my.cnf du client soit passer les paramètres au client mysql

```
mysql --ssl-ca=ca.pem \
      --ssl-cert=client-cert.pem \
      --ssl-key=client-key.pem
```

Mode SSL

Par défaut, les programmes clients MySQL tentent d'établir une connexion chiffrée si le serveur prend en charge les connexions chiffrées, avec un contrôle supplémentaire disponible via l'option --ssl-mode:

En l'absence d'une option --ssl-mode, les clients tentent de se connecter à l'aide d'un cryptage, retombant sur une connexion non cryptée si une connexion cryptée ne peut pas être établie. C'est également le comportement avec une option explicite --ssl-mode = PREFFERED.

- Avec --ssl-mode = REQUIRED, les clients nécessitent une connexion chiffrée et échouent s'il est impossible de l'établir.
- Avec --ssl-mode = DISABLED, les clients utilisent une connexion non chiffrée.
- Avec --ssl-mode = VERIFY_CA ou --ssl-mode = VERIFY_IDENTITY, les clients exigent une connexion chiffrée et effectuent également une vérification du certificat de l'autorité de certification du serveur et (avec VERIFY_IDENTITY) du nom d'hôte du serveur dans son certificat.

Moteurs de stockage et plug-ins

- Moteurs de stockage MyISAM, InnoDB.
- Architecture et paramétrage InnoDB.
- Fonctionnement transactionnel du moteur InnoDB.
- Verrouillage des tables.
- Plug-ins : configuration et démarrage.

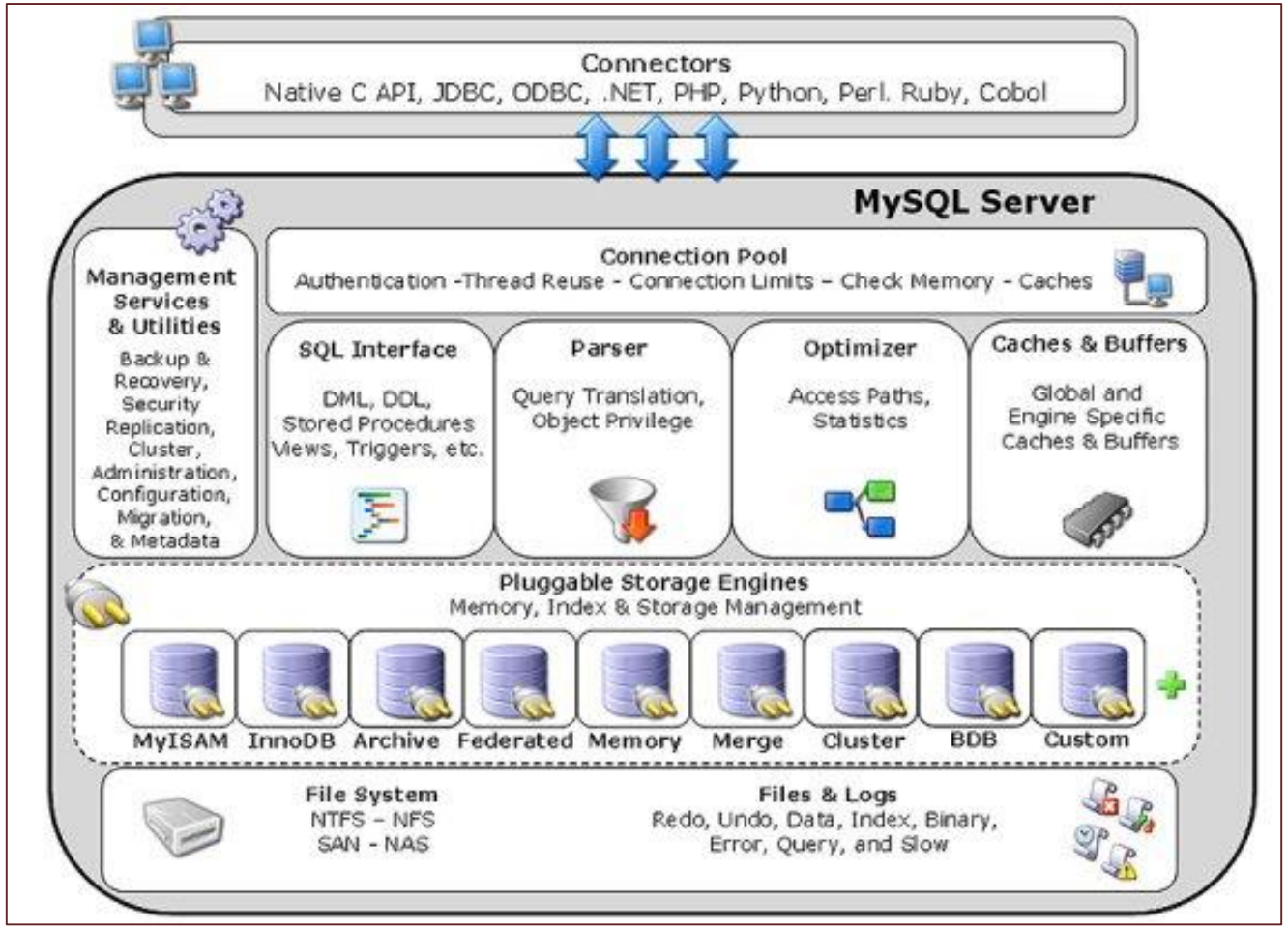
Moteurs de stockage et plug-ins
Moteurs de stockage MyISAM, InnoDB

- Qu'est ce que le plugable storage engine?
- Focus sur MyISAM et InnoDB

Moteurs de stockage MyISAM, InnoDB

- Une particularité de MySQL est de disposer de moteurs de stockages modulaires.
- Chaque moteur possède des caractéristiques particulières (performance, transactions, recherche full text).
- Un serveur MySQL se charge de créer un certain nombre de thread:
  - Un thread global est responsable de cla création et de gestion des connexions
  - Un thread surveilles les connexions et les termine.
  - Un thread pour la réplication, et un thread dédié à la sauvegarde des donnée sur le disque.

Plugable Storage Engine



Plugable Storage Engine

- Les moteurs de stockage sont des composants MySQL qui gèrent les opérations SQL pour différents types de tables. Les moteurs de stockage MySQL incluent à la fois ceux qui gèrent les tables sécurisées pour les transactions et ceux qui gèrent les tables non sécurisées pour les transactions. InnoDB est le moteur de stockage par défaut à partir de MySQL 5.5.5 (L'instruction CREATE TABLE dans MySQL 5.5 crée les tables InnoDB par défaut.)
- MySQL utilise une architecture de moteur de stockage enfichable qui permet aux moteurs de stockage d'être chargés et déchargés d'un serveur MySQL en cours d'exécution.
- Pour déterminer les moteurs de stockage pris en charge par votre serveur, utilisez l'instruction SHOW ENGINES. La valeur dans la colonne Support indique si un moteur peut être utilisé. Une valeur de YES, NO ou DEFAULT indique qu'un moteur est disponible, indisponible ou disponible et actuellement défini comme moteur de stockage par défaut.

SHOW ENGINES

```
mysql> SHOW ENGINES\G
*************************** 1. row ***************************
      Engine: PERFORMANCE_SCHEMA
     Support: YES
     Comment: Performance Schema
Transactions: NO
          XA: NO
  Savepoints: NO
*************************** 2. row ***************************
      Engine: InnoDB
     Support: DEFAULT
     Comment: Supports transactions, row-level locking, and foreign keys
Transactions: YES
          XA: YES
  Savepoints: YES
*************************** 3. row ***************************
      Engine: MRG_MYISAM
     Support: YES
     Comment: Collection of identical MyISAM tables
Transactions: NO
          XA: NO
  Savepoints: NO
```

Fichier FRM

- Quel que soit le moteur de stockage choisi, chaque table MySQL que vous créez est représentée sur le disque par un fichier .frm décrivant le format de la table. Le fichier porte le même nom que la table, avec une extension .frm. Le format .frm est identique sur toutes les plates-formes.
  - mysql> CREATE TABLE table1 (column1 CHAR(5)) ENGINE=MYISAM COMMENT '*';
  - shell> ls table1.*
- Le fichier frm représente la sutrcture de la table. Le fichier MYD (MyIsam Data) et MYI (MyISAM Index) sont propre au storage engine MyISAM

MyISAM

- MyISAM est basé sur l'ancien moteur de stockage ISAM (et n'est plus disponible).
- MyISAM est un moteur rapide avec une faible empreinte en mémoire (seul les index sont chagés en RAM)
- Son usage est envisageable sur des données non critiques dans les écarts de performances entre le moteur InnoDB et le moteur MyISAM.

MyISAM

- Chaque table MyISAM est stockée sur le disque dans deux fichiers. Les fichiers ont des noms qui commencent par le nom de la table et ont une extension pour indiquer le type de fichier. Le fichier de données a une extension .MYD (MYData). Le fichier d'index a une extension .MYI (MYIndex). La définition de la table est stockée dans le dictionnaire de données MySQL.
- Vous pouvez vérifier ou réparer les tables MyISAM avec le client mysqlcheck ou l'utilitaire myisamchk. Vous pouvez également compresser les tables MyISAM avec myisampack pour prendre beaucoup moins de place.
- CREATE TABLE t (i INT) ENGINE = MYISAM;

MyISAM fonctionnalités

- Prise en charge d'un véritable type VARCHAR; une colonne VARCHAR commence par une longueur stockée dans un ou deux octets.
- Les tables avec des colonnes VARCHAR peuvent avoir une longueur de ligne fixe ou dynamique.
- La somme des longueurs des colonnes VARCHAR et CHAR dans un tableau peut aller jusqu'à 64 Ko.
- Longueur arbitraire Contraintes UNIQUES.

MyISAM caractéristiques

- Toutes les valeurs de données sont d'abord stockées avec l'octet faible. Cela rend la machine de données et le système d'exploitation indépendants.
- Toutes les valeurs de clé numériques sont d'abord stockées avec l'octet de poids fort pour permettre une meilleure compression d'index
- Il y a une limite de (2 ^ 32) lignes dans une table MyISAM.
- Le nombre maximal d'index par table MyISAM est de 64.
- Le nombre maximal de colonnes par index est de 16. La longueur maximale de la clé est de 1000 octets.
- Lorsque les lignes sont insérées dans un ordre trié (comme lorsque vous utilisez une colonne AUTO_INCREMENT),
- Le traitement interne d'une colonne AUTO_INCREMENT par table est pris en charge.
- Les colonnes BLOB et TEXT peuvent être indexées.
- Les valeurs NULL sont autorisées dans les colonnes indexées. Cela prend 0 à 1 octet par clé.
- Chaque colonne de caractères peut avoir un jeu de caractères différent

My.ini example pour MyISAM

```
key_buffer_size                 = 8M
# Set to 25 - 33 % of RAM if you still use MyISAM
myisam_recover_options          = 'BACKUP,FORCE'
```

Support technique

Fonctionnalités	Support
Index B-tree	Oui
Sauvegarde / récupération à un moment donné	Oui
Prise en charge de la base de données de cluster	Non
Index clusterisé	Non
Données compressées	Oui
Caches de données	Non
Données cryptées	Oui
Support de clé étrangère	Non
Index de hachage	Non
Limites de stockage	256 To
Mise à jour des statistiques pour le dictionnaire de données	Oui

InnoDB

- InnoDB est le moteur par défaut sur MySQL. C'est un moteur transactionnel complet qui supporte la recherche full text.
- Ce moteur est légèrement moins rapide que MyISAM.
- Il mobilise plus de ressources puisque les tables InnoDB doivent être chargées en mémoire vive pour fonctionner de manière optimale.

InnoDB

Vous pouvez trouver les tables InnoDB utiles pour les raisons suivantes:

- Si votre serveur tombe en panne, les tables InnoDB peuvent etre rechargé sans problème.
- Le moteur de stockage InnoDB maintient son propre pool de mémoire tampon qui met en cache la table et indexe les données dans la mémoire principale lors de l'accès aux données.
- Si vous partionnez des données liées en différentes tables, vous pouvez configurer des clés étrangères qui appliquent l'intégrité référentielle.
- Optimisation des requestes en utilisant les clef primaires (vue comme des index)
- Vous pouvez compresser des tables et des index associés. Vous pouvez créer et supprimer des index avec beaucoup moins d'impact sur les performances et la disponibilité.

InnoDB

Fonctionnalités	Support
Index B-tree	Oui
Sauvegarde / récupération à un moment donné	Oui
Prise en charge de la base de données de cluster	Non
Index clusterisé	Oui
Données compressées	Oui
Caches de données	Oui
Données cryptées	Oui
Support de clé étrangère	Oui
Index de hachage	Non (mise en place du Adaptative Hash Index)
Limites de stockage	64 To
Mise à jour des statistiques pour le dictionnaire de données	Oui

Page Size

- Le Page Size est la façon qu'utilise une base de données pour gérer la mémoire.
- Il s'agit de la quantité minimal de mémoire pouvant être ecrit/lut en un seul cout.

- La taille de page par défaut est d'une base de donnée est de 4 Ko
- Si vous avez de grands ensembles de données ou que très peu d'écritures, cela peut améliorer les performances pour augmenter la taille de la page.
  - Pourquoi? Parce que plus de données peuvent être récupérées / adressées à la fois. Si la probabilité est grande que les données souhaitées se trouvent à proximité des données que vous venez de récupérer, ou juste après, vous pouvez simplement les récupérer en une seule opération. et tirez davantage parti de plusieurs technologies de mise en cache et d'extraction de données, en général depuis votre disque dur. Mais de l'autre côté, vous perdez de la place si vous avez des données qui ne remplissent pas la taille de la page ou qui sont juste un peu plus ou quelque chose.

Page Size et InnoDB

- Pour les versions jusqu'à MySQL 5.5 (inclus), la taille de chaque page InnoDB est fixée à 16 kilo-octets.
- Cette valeur représente un équilibre: assez volumineux pour contenir les données de la plupart des lignes, mais suffisamment petit pour minimiser les performances du transfert des données inutiles en mémoire.
- Depuis MySQL 5.6, la taille d'une page pour une instance InnoDB peut être de 4 Ko, 8 Ko ou 16 Ko, contrôlée par l'option de configuration innodb_page_size.
- Depuis MySQL 5.7.6, InnoDB prend également en charge les tailles de page de 32 Ko et 64 Ko.

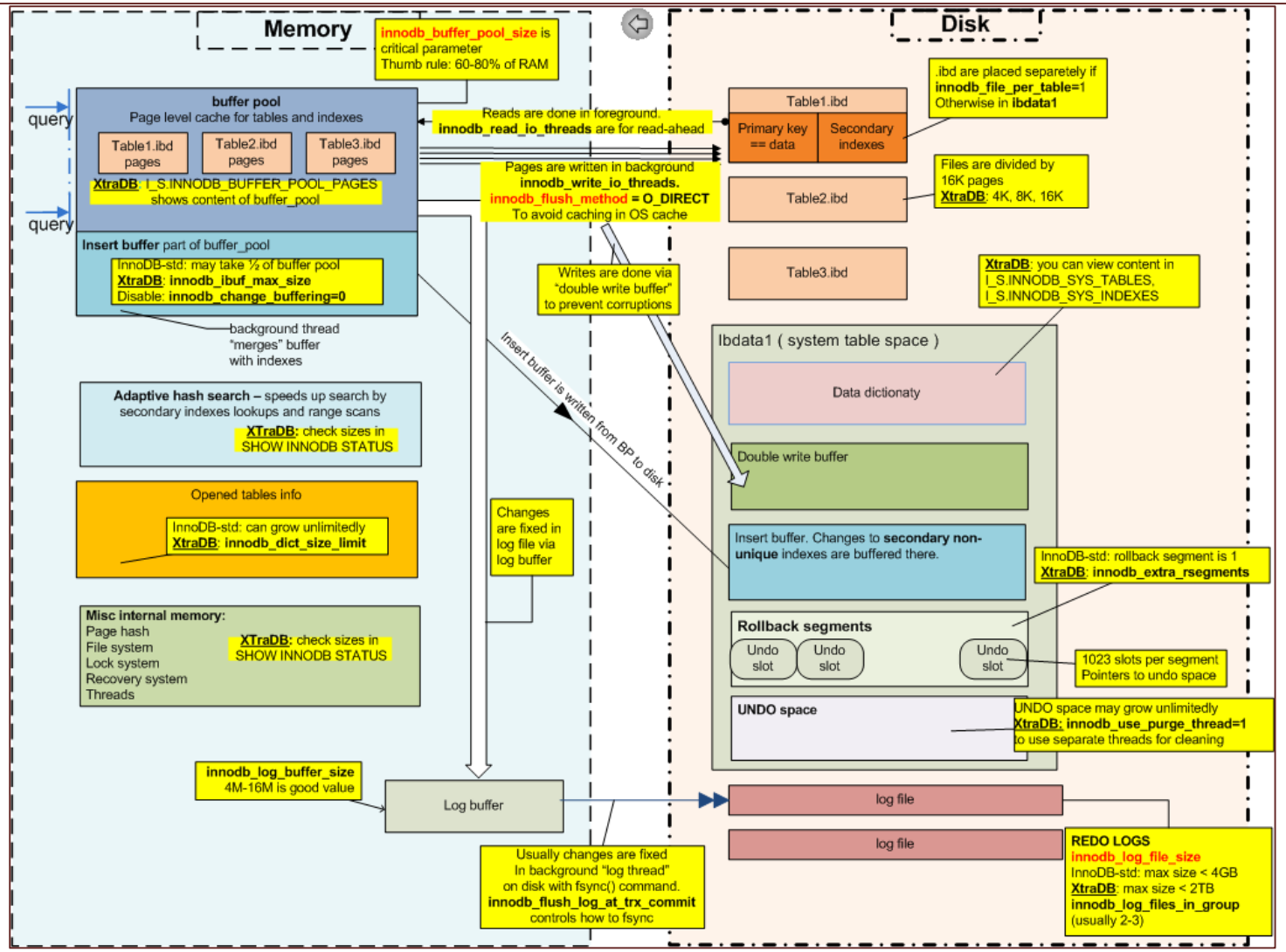
InnoDB

- Une table peut contenir un maximum de 1017 colonnes.
- Les colonnes générées virtuellement sont incluses dans cette limite.
- Une table peut contenir un maximum de 64 index secondaires.
- La longueur maximale du préfixe de la clé d'index est de 3 072 octets pour les tables InnoDB utilisant le format de ligne DYNAMIC ou COMPRESSED.
- La longueur maximale du préfixe de la clé d'index est de 767 octets pour les tables InnoDB utilisant le format de ligne REDUNDANT ou COMPACT.
  - Par exemple, vous pouvez atteindre cette limite avec un index de préfixe de colonne de plus de 191 caractères sur une colonne TEXT ou VARCHAR, en supposant un jeu de caractères utf8mb4 et un maximum de 4 octets pour chaque caractère. Toute tentative d'utilisation d'une longueur de préfixe de clé d'index supérieure à la limite renvoie une erreur.
- Les limites applicables aux préfixes de clé d'index s'appliquent également aux clés d'indexation de colonne complète. Si vous réduisez la taille de la page InnoDB à 8 ou 4 Ko en spécifiant l'option innodb_page_size lors de la création de l'instance MySQL, la longueur maximale de la clé d'index est réduite proportionnellement, en fonction de la limite de 3 072 octets pour une taille de page de 16 K
- Autrement dit, la longueur maximale de la clé d'index est de 1536 octets lorsque la taille de la page est de 8 Ko et de 768 octets lorsque la taille de la page est de 4 Ko. Un maximum de 16 colonnes est autorisé pour les index multicolonnes. Dépasser la limite renvoie une erreur.

Paramétrage InnoDB.

- InnoDB en Interne
- Paramétrage important de InnoDB

InnoDB


Memory
innodb_buffer_pool_size is critical parameter
Thumb rule: 60-80% of RAM
query
query
buffer pool
Page level cache for tables and indexes
Table1.ibd pages
Table2.ibd pages
Table3.ibd pages
XtraDB: I_S.INNODB_BUFFER_POOL_PAGES shows content of buffer_pool
Insert buffer part of buffer_pool
InnoDB-std: may take ½ of buffer pool
XtraDB: innodb_ibuf_max_size
Disable: innodb_change_buffering=0
background thread "merges" buffer with indexes
Adaptive hash search – speeds up search by secondary indexes lookups and range scans
XTraDB: check sizes in SHOW INNODB STATUS
Opened tables info
InnoDB-std: can grow unlimitedly
XtraDB: innodb_dict_size_limit
Misc internal memory:
Page hash
File system
Lock system
Recovery system
Threads
XTraDB: check sizes in SHOW INNODB STATUS
Reads are done in foreground.
innodb_read_io_threads are for read-ahead
Pages are written in background
innodb_write_io_threads.
innodb_flush_method = O_DIRECT
To avoid caching in OS cache
Writes are done via "double write buffer" to prevent corruptions
Insert buffer is written from BP to disk
Changes are fixed in log file via log buffer
innodb_log_buffer_size
4M-16M is good value
Log buffer
Usually changes are fixed In background "log thread" on disk with fsync() command.
innodb_flush_log_at_trx_commit
controls how to fsync
Disk
.ibd are placed separetely if innodb_file_per_table=1
Otherwise in ibdata1
Table1.ibd
Primary key == data
Secondary indexes
Table2.ibd
Files are divided by 16K pages
XtraDB: 4K, 8K, 16K
Table3.ibd
XtraDB: you can view content in I_S.INNODB_SYS_TABLES, I_S.INNODB_SYS_INDEXES
Ibdata1 (system table space)
Data dictionaty
Double write buffer
Insert buffer. Changes to secondary non-unique indexes are buffered there.
InnoDB-std: rollback segment is 1
XtraDB: innodb_extra_rsegments
Rollback segments
Undo slot
Undo slot
Undo slot
1023 slots per segment
Pointers to undo space
UNDO space
UNDO space may grow unlimitedly
XtraDB: innodb_use_purge_thread=1
to use separate threads for cleaning
log file
log file
REDO LOGS
innodb_log_file_size
InnoDB-std: max size < 4GB
XtraDB: max size < 2TB
innodb_log_files_in_group
(usually 2-3)

Principaux Paramètres InnoDB

- Les paramètres ne sont nécéssairement à modifier par rapport à la configuration par défaut!
- Innodb_buffer_pool_size représente la taille du cache mémoire. Il s'agit du cache princiapl d'InnoDB (buffer pool) où les données et les index sont fréquemment accédés et sotckées. Pour un serveur MySQL, il est courant de lui allouer la majeur partie de la mémoire du serveur.
- L'idée est que ce cache permet d'éviter les accès au disque.

Innodb_buffer_pool_size

- Si la base de données est petite, il est intéressant de modifier Innodb_buffer_pool_size afin qu'il contiennent l'ensemble des données et des index.
- Sinon, il faut essayer de faire tenir les tables les plus usité dans le pool_size.

Sizing Innodb_buffer_pool_size

- La pratique la plus couramment suivie consiste à définir cette valeur entre 70% et 80% de la mémoire vive du système. Bien que cela fonctionne bien dans la plupart des cas, cette méthode peut ne pas être optimale dans toutes les configurations

- Une méthode plus précise pour calculer la taille du pool de mémoire tampon InnoDB:
  - Commencez avec la RAM totale disponible.
  - Soustrayez une quantité appropriée pour tous les besoins du système d'exploitation.
  - Soustrayez un montant approprié pour tous les besoins MySQL (tels que divers tampons MySQL, tables temporaires, pools de connexions et tampons liés à la réplication).
  - Divisez le résultat par 105%, ce qui représente une approximation de la surcharge nécessaire à la gestion du pool de mémoire tampon lui-même.

Sizing Innodb_buffer_pool_size

- Pour les systèmes fonctionnant avec moins de 1 Go de RAM, il est préférable d'utiliser la valeur de configuration par défaut de MySQL de 128 Mo pour la taille du pool de mémoire tampon InnoDB.
- En considérant le cas de systèmes avec une taille de RAM comprise entre 1 Go et 32 Go, nous pouvons calculer les besoins du système d'exploitation à l'aide de ces heuristiques approximatives:
  - 256 Mo + 256 * log2 (taille de la RAM en Go)
- La rationalisation ici est que, pour les configurations à faible RAM, nous commençons avec une valeur de base de 256 Mo pour les besoins du système d'exploitation, puis nous augmentons cette allocation à une échelle logarithmique à mesure que la quantité de RAM augmente. De cette façon, nous pouvons trouver une formule déterministe pour allouer de la mémoire vive aux besoins de notre système d'exploitation.
- Pour les systèmes dont la taille de la RAM est supérieure à 32 Go, nous reviendrions au calcul des besoins du système d'exploitation en tant que 5% de la taille de la RAM de notre système et le même montant pour les autres besoins de MySQL.

Innodb_log_file_size

- Le journal redo_log permet à InnoDB d'offrir de bonne performances en écriture tout en garantissant l'intégrité des données en cas d'arrêt inopiné.
- InnoDB écrit de manière synchrone les modifications dans son journal transactionnel, et un thread en arrière plan vient modifier les données de manières asynchrone.
- Si la taille du journal transactionnel est limité, et si vous écrivez plus vite que le thread est capable de faire les modifications, alors InnoDB va bloquer les écritures pendant la récupération de mémoire se fait.

Sizing Innodb_log_file_size

- La valeur par défaut est de 48Mo convient aux applications écrivant peu, mais seras insuffisante en cas d'application transactionnel.
- L'hier est d'avoir suffisament d'espace pour contenir 1h e modificaiton en heure de pointe.
- Cela se fait en comparant 1 heure d'intervalle la valeur de la variable log_sequence_number qui apparait dans la section LOG de la sortie de SHOW ENGINE INNODB STATUS.
- Des valeurs telles que 256/512/1024 Mo sont courantes.

Innodb_file_per_table

- Dans les anciennes versions de MySQL, toutes les donénes et index de toutes les tables étaient stockées dans le tablespace partagé (ibdata1). Ce fichier pouvait dépasser le To si la base était volumineuse, ce qui peut être problématique en terme de gestion.
- Pour eviter ce déagrément, la solution a été d'utiliser la variable innodb_file_per_table=1, ce qui indique à InnoDB de stocker chaque table dans son propre fichier idb.
- En outre cela permet de récuperer l'espace disque des tables supprimer.
- Néanmoins, si la structure de la base est constituer de nombreuses petites tables, cette option devient génantes (trop de fichier a ouvrir pour l'OS).

Innodb_flush_method

- Quand des modifications sont écrites dans les fichiers de données, une partie est également conservées dans le cache du système de f fichier.
- Cette mise en cache peut être inutile car InnoDB dispose de son buffer pool.
- Si les disques sont rapides, cette double mise en cache est inutile et on peut l'éviter en utilisant innodb_flush_method=O_DIRECT.

InnoDB_flush_log_at_trx_commit

- InnoDB enregistre les informations de chaque transaction dans un cache afin d'éviter de solliciter le disque. Ces données résidant en mémoire, elle ne resistent pas à un crash du serveur.
- Au moment du COMMIT ces informations sont recopiées dans le journal d'InnoDB afin d'être ensuite appliquées aux donnes.
- Dès que les données sont copiées dans un journal elles sont considéré comme étant en sécurité. (Si un crash survient MySQL peut réecrir les données en utilisant son journal).

InnoDB_flush_log_at_trx_commit

- Il existe une difficulté au moment de l'écriture sur le fichier de log, le système d'exploitation peut décider de ne pas écrire les données et de les conserver dans les caches de l'OS.
- C'est pourquoi InnoDB demande au système d'exploitation de faire un flush après l'écriture pour s'assurer que les données sont bien sur le disque.
- InnoDB vous laisse le contrôle de cette faculté de flush(couteuse en terme de perfomance) visa la variable innodb_flush_log_at_trx_commit.

InnoDB_flush_log_at_trx_commit

Le paramètre par défaut de 1 est requis pour la conformité ACID complète. Les journaux sont écrits et vidés sur le disque à chaque validation de transaction.

Avec un réglage de 0, les journaux sont écrits et vidés sur le disque une fois par seconde. Les transactions pour lesquelles les journaux n'ont pas été vidés peuvent être perdues dans un crash.

Avec un réglage de 2, les journaux sont écrits après chaque validation de transaction et vidé sur le disque une fois par seconde. Les transactions pour lesquelles les journaux n'ont pas été vidés peuvent être perdues dans un crash.

Pour les réglages 0 et 2, le rinçage une fois par seconde n'est pas garanti à 100%. Le flush peut se produire plus fréquemment en raison de modifications de DDL et d'autres activités internes InnoDB entraînant le vidage des journaux indépendamment du paramètre innodb_flush_log_at_trx_commit, et parfois moins fréquemment en raison de problèmes de planification. Si les journaux sont vidés une fois par seconde, il est possible de perdre jusqu'à une seconde de transactions en cas d'accident.

Innodb_buffer_pool_instances

- Si le buffer pool est très gros il est possible que certaines opérations subissent des contentions. Il est alors conseillé d'utiliser plusieurs instanecs de buffer pool pour les diminuer.
- La valeur par défaut de 8 convient très souvent.

Innodb_buffer_pool_dump_at_shutdown/startup

- Ces deux options permettent de prendre une empreinte du buffer pool avant de s'arreter le recharger l'empreinte au démarrage.
- L'intérêt est de plus ou moins préserver le buffer pool dans un contexte de performance.

innodb_io_capacity

- Le paramètre innodb_io_capacity définit une limite supérieure du nombre d'opérations d'E / S effectuées par seconde par les tâches en arrière-plan InnoDB, telles que le vidage des pages du pool de mémoire tampon et la fusion des données du tampon de modification.

- La limite innodb_io_capacity est une limite totale pour toutes les instances de pool de mémoire tampon. Lorsque les pages modifiées sont vidées, la limite est divisée également entre les instances de pool de mémoire tampon.

- innodb_io_capacity doit être défini sur le nombre approximatif d'opérations d'E / S que le système peut effectuer par seconde.

innodb_io_capacity

- Le paramètre par défaut de 200 est généralement suffisant pour un SSD bas de gamme. Pour un disque SSD haut de gamme connecté au bus, envisagez un paramètre plus élevé, par exemple 1000. Pour les systèmes dotés de disques individuels à 5 400 tr / min ou 7 200 tr / min, vous pouvez réduire la valeur à 100, ce qui représente une proportion estimée des opérations d'E / S par seconde (IOPS) disponibles pour les disques durs de génération antérieure pouvant exécuter environ 100 IOPS.

Paramétrage exemple de InnoDB

```
innodb_strict_mode              = ON
innodb_file_format_check        = 1
innodb_file_format              = Barracuda                    # For dynamic and compressed InnoDB tables
innodb_buffer_pool_size         = 128M                         # Go up to 80% of your available RAM
innodb_buffer_pool_instances    = 8                            # Bigger if huge InnoDB Buffer Pool or high concurrency

innodb_file_per_table           = 1                            # Is the recommended way nowadays
 innodb_flush_method             = O_DIRECT                     # O_DIRECT is sometimes better for direct attached storage
 innodb_write_io_threads         = 8                            # If you have a strong I/O system or SSD
 innodb_read_io_threads          = 8                            # If you have a strong I/O system or SSD
 innodb_io_capacity              = 1000                         # If you have a strong I/O system or SSD

innodb_flush_log_at_trx_commit = 2                             # 1 for durability, 0 or 2 for performance
innodb_log_buffer_size          = 8M                           # Bigger if innodb_flush_log_at_trx_commit = 0
innodb_log_file_size            = 256M                         # Bigger means more write throughput but longer recovery
time
```

Paramétrage possible en sus

```
# Query Cache

query_cache_type              = 0                    # Set to 0 to avoid global QC Mutex
# query_cache_size              = 32M                  # Avoid too big (> 128M) QC because of QC clean-up lock!


# Session variables

sort_buffer_size              = 2M                   # Could be too big for many small sorts
tmp_table_size                = 32M                  # Make sure your temporary results do NOT contain BLOB/TEXT
attributes

read_buffer_size              = 128k                 # Resist to change this parameter if you do not know what
you are doing
read_rnd_buffer_size          = 256k                 # Resist to change this parameter if you do not know what
you are doing
join_buffer_size              = 128k                 # Resist to change this parameter if you do not know what
you are doing


# Other buffers and caches

table_definition_cache        = 1400                 # As big as many tables you have
table_open_cache              = 2000                 # connections x tables/connection (~2)
table_open_cache_instances    = 16                   # New default in 5.7
```

Paramétrages des connexions

- On peut vérifier le nombre de connexions maximum autorisées. Et le nombre de connexions maximum qui ont été ouvertes en même temps.

```
SELECT @@max_connections;
+-------------------+
| @@max_connections |
+-------------------+
|               100 |
+-------------------+

SHOW GLOBAL STATUS like "%used_connections";
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| Max_used_connections | 98    |
+----------------------+-------+
```

Paramétrages des connexions

- Dans le cas présent, il peut s'avérer judicieux d'augmenter la limite. Cela augmente les ressources consommées par le serveur, mais permettra de servir plus de clients à la fois.
- Lorsque le max_connections est atteint, les nouvelles connexions sont mises en file d'attente jusqu'à ce qu'une place se libère.
- Il ne s'agit donc pas d'augmenter cette limite à l'infini, mais si le serveur peut tolérer plus de connexions et que vous vous approchez ou atteignez le maximum autorisé, n'hésitez pas à l'augmenter

Nombre de connection dans my.ini

```
max_connections                 = 505
# Values < 1000 are typically good
max_user_connections            = 500
# Limit one specific user/application
thread_cache_size               = 505
# Up to max_connections makes sense
```

Moteurs de stockage et plug-ins
Fonctionnement transactionnel du moteur InnoDB..

Différent mode dans les transactions MySQL sur InnoDB

Transactions et MySQL

- **Qu'est ce qu'une Transaction ACID InnoDB**
  - Atomicity: La requête est exécuté dans une transaction soit complètement soit pas du tout !
  - Consistency: Les règles d'intégrité peuvent être rompue uniquement pendant la transaction
  - Isolation: Les données utilisées pendant une transaction ne peuvent pas être utilisé dans une autre transaction sans que la première soit terminé
  - Durability: Une transaction terminé n'est plus réversible.

InnoDB Transaction

- Les Transactions sont comprises entre les mots clefs *BEGIN* et *COMMIT*.
- Un Exemple:

```
mysql> CREATE TABLE t (f INT) TYPE=InnoDB;
mysql> BEGIN; Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO t(f) VALUES (1); Query OK, 1 row
affected (0.01 sec)
mysql> SELECT * FROM t;
    +------+
    |   f  |
    +------+
    |   1  |
    +------+ 1 row in set (0.02 sec)
mysql> ROLLBACK; Query OK, 0 rows affected (0.01 sec)
mysql> SELECT * FROM t; Empty set (0.00 sec)
```

COMMIT/ROLLBACK

- BEGIN commence une transaction
- COMMIT valide la transaction
- ROLLBACK annule la transaction

- Il peut y avoir plusieurs requêtes entre le BEGIN (début de transaction) et un COMMIT ou un ROLLBACK.

Lecture consistente

- Connection 1:
-

```
mysql> BEGIN;
Query OK,
INSERT INTO t (f) VALUES (1);
SELECT * FROM t;
+------+
|  f   |
+------+
|  1   |
```

- Connection 2:

```
SELECT * FROM t;
Empty set (0.02 sec)
```

Lecture consistente

- Connection 1:

```
mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
```

- Connection 2:

```
mysql> SELECT * FROM t;
+------+
|   f  |
+------+
|   1  |
+------+
1 row in set (0.00 sec)
```

Commits Explicite

- Il est possible de dire a MySQL que toutes les requetes seront dans des transactions:

```
SET AUTOCOMMIT=0;
```

- Connection 1:

```
mysql> INSERT INTO t (f) VALUES (2);
```

- Connection 2:

```
mysql> SELECT * FROM t;
+------+
| f |
+------+
| 1 |
+------+ 1 row in set (0.00 sec)
```

- Connection 1

```
mysql> COMMIT;
Query OK, 1 rows affected (0.00 sec)
```

Commits Explicite

- pour savoir si vous êtes (1) en mode autocommit ou non (0) : @@autocommit ;
- pour enlever l'autocommit (deux possibilités) :
- SET @@aut
- pour le

```
SELECT @@autocommit ;
```

```
SET @@autocommit = 0 ;
SET @@autocommit = off ;
```

```
SET @@autocommit = 1 ;
SET @@autocommit = ON ;
```

Niveau d'Isolation

- READ UNCOMMITTED: La transaction n'est pas isolé (lecture de données non "commité"). Autrement appelé "dirty read".
- READ COMMITTED: interdit la lecture sale
- REPEATABLE READ: **Il s'agit du mécanisme par défaut.** Les lectures sont cohérentes d'un début de transaction à la fin. On parle de lecture fantome.
- SERIALIZABLE: fait comme si les transactions étaient validées les unes après les autres et non en même temps

Niveau d'Isolation

- Pour connaître votre niveau d'isolation :

```
SELECT @@tx_isolation ;
```

- Pour le modifier, vous pouvez soit utiliser les syntaxes standard de SET :

```
SET tx_isolation = 'READ-COMMITTED' ; -- avec tiret
et apostrophes
SET tx_isolation = 1 ;
SET @@tx_isolation = 1 ;
SET GLOBAL tx_isolation = 'READ-COMMITTED' ;
SET GLOBAL tx_isolation = 1 ;
```

ISOLATION LEVEL :

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED ;
```

READ UNCOMMITTED

- READ UNCOMMITTED ``dirty read'' : Les commandes SELECT non-verrouillantes sont effectuées sans rechercher de versions plus récente de la ligne. Elles sont donc non-cohérentes, sous ce niveau d'isolation. Sinon, ce niveau fonctionne comme le niveau READ COMMITTED.

READ UNCOMMITTED

Connexion 1

```
set tx_isolation='read-uncommitted';

create table isolation_test (id int primary
key auto_increament, name varchar(8))
engine-InnoDB;

insert into isolation_test (name) values
('a'),('b');

select id,name from isolation_test;
+----+------+
| id | name |
+----+------+
|  1 | a    |
|  2 | b    |
+----+------+
```

Connexion 2

```
set autocommit=0;
insert into isolation_test (name) values
('a'),('b');
```

```
select id,name from isolation_test;
+----+------+
| id | name |
+----+------+
|  1 | a    |
|  2 | b    |
|  3 | a    |
|  4 | b    |
+----+------+
4 rows in set (0.00 sec)
```

- Les lignes 3,4 ne sont pas « committer » dans la connexion 2 mais déjà visible dans la connexion 1

READ COMMITTED

- Proche du niveau d'isolation d'Oracle. Toutes les commandes SELECT ... FOR UPDATE et SELECT ... LOCK IN SHARE MODE ne verrouille que les lignes d'index, et non pas les trous entre elles, ce qui permet l'insertion de nouvelles lignes, adjacentes aux lignes verrouillées.
- UPDATE et DELETE qui utilisent un seul index avec une condition de recherche unique ne verrouille que la ligne d'index trouvée, par le trou qui la précède. Mais, pour un intervalle de lignes traitées par UPDATE et DELETE, InnoDB doit méler des verrous de prochaine clé ou des verrous de trous et bloquer les insertions des autres utilisateurs dans les espaces couvert par l'intervalle. Ceci est nécessaire car les ``lignes fantômes" doivent être bloquées pour la réplication MySQL et la restauration.

READ COMMITTED

Connexion 1

```
mysql> set tx_isolation='read-committed';
Query OK, 0 rows affected (0.00 sec)

mysql> begin;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from isolation_test;
Empty set (0.00 sec)
```

Connexion 2

```
mysql> set tx_isolation='read-committed';
Query OK, 0 rows affected (0.00 sec)

mysql> begin;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from isolation_test;
Empty set (0.00 sec)
```

```
mysql> select * from isolation_test;
+----+------+
| id | name |
+----+------+
| 13 | c    |
| 14 | d    |
+----+------+
```

- Les lignes 13,14 sont « committer » dans la connexion 2 sont déjà visible dans la transaction de la connexion 1

REPEATABLE READ

Connexion 1

```
mysql> set tx_isolation='repeatable-read';
Query OK, 0 rows affected (0.00 sec)

mysql> begin;
mysql> select * from isolation_test ;
+----+------+
| id | name |
+----+------+
|  1 | a    |
|  2 | b    |
+----+------+
```

Connexion 2

```
begin;
delete from isolation_test where id=1;
mysql> commit;
mysql> select * from isolation_test;
+----+------+
| id | name |
+----+------+
|  2 | b    |
|  3 | a    |
+----+------+
```

```
mysql> select * from isolation_test ;
+----+------+
| id | name |
+----+------+
|  1 | a    |
|  2 | b    |
+----+------+
```

- La ligne 1 est un fantome.

Serializable

Connexion 1

```
mysql> set tx_isolation='serializable';
Query OK, 0 rows affected (0.00 sec)

mysql> begin;
```

Connexion 2

```
mysql> begin;
Query OK, 0 rows affected (0.00 sec)

mysql> delete from isolation_test;
Query OK, 2 rows affected (0.00 sec)
```

```
mysql> select * from isolation_test ;
```

- La connexion 1 est bloqué

Moteurs de stockage et plug-ins
Verrouillage des tables.

Pour implémenter une application de base de données à grande échelle, très utilisée ou très fiable, pour transférer du code substantiel depuis un système de base de données différent ou pour optimiser les performances de MySQL, il est important de comprendre le verrouillage InnoDB et le modèle de transaction InnoDB.

Lock InnoDB

Il y a plusieurs type de lock sur une table InnoDB:

- (Read) Shared Locks
- (Write) Exclusive Locks
- Intention Locks
- Record Locks
- Gap Locks
- Next-Key Locks
- Insert Intention Locks
- AUTO-INC Locks

Shared/Exclusive Lock

Il existe essentiellement deux types de mécanismes de verrouillage:

- Verrou de lecture (verrou partagé). Ce sont des verrous pris sur des ressources pour fournir des lectures cohérentes en interdisant les opérations d'écriture sur cette ressource.
- Verrouillage en écriture (verrouillage exclusif). Ce sont des verrous pris sur une ressource lorsque celle-ci doit être modifiée. Il ne peut y avoir qu'un seul verrou en écriture sur une ressource à un moment donné.

Shared Locks

Verrou qui permet à d'autres transactions de lire l'objet verrouillé et d'acquérir d'autres verrous partagés, sans y écrire.

Si la transaction T1 conserve un verrou (S) partagé sur la ligne r, les demandes de verrouillage de la ligne r provenant d'une transaction T2 distincte sont traitées comme suit:

- Une demande de verrouillage en S par T2 peut être accordée immédiatement. En conséquence, T1 et T2 tiennent tous deux un verrou S sur r.
- Une demande de verrouillage en X par T2 ne peut pas être accordée immédiatement.

Exclusive Lock

Verrou qui empêche toute autre transaction de verrouiller la même ligne. En fonction du niveau d'isolation de transaction, ce type de verrou peut empêcher d'autres transactions d'écrire sur la même ligne ou d'empêcher d'autres transactions de lire la même ligne.

Moteurs de stockage et plug-ins
Configuration, démarrage.

- Configuration basique du démarrage de MySQL

Skip_networking

- Dans le cas d'architectures ou le client est sur la même machine que le serveur, il est intéressant d'interdire les connexions distances.
- Non seulement on élimine les latences du réseau mais en plus on limite la surface d'attaque du serveur.
- En mettant en place l'option skip-networking:
  - Sous unix, la connexion se feras via un file socket
  - Sous windows, la connexion passeras via soi un pipes sou de la shared memory

Bind-address

- L'option bind-address permet d'attacher le serveur a une adresse IP ou a une interface réseau.
- En autre terme, on peut forcer les client a passer par une adresse IP particulière
  - Pour empêcher l'accès au serveur des clients distance, on peut positionner bind-address à 127.0.01

Skip-name-resolve

- L'option skip-name-resolve desactive la recherche des noms d'hôtes par DNS. Lors de la connexion des clients au serveur MySQL.
- Cela oblige les application à n'utiliser que des comptes ayant comme nom d'hôte une adresse IP.
- LE gain de cette option est double:
  - Le premier avantage est de réduire les rsique d'attaques par deni de service DNS ce qui empecherait le bon fonctionnement de MySQL.
  - Le deuxième est d'éviter le surcout d'une résolution DNS par MySQL lors de la connexion des client.

Skip-show-database

- Tout compte utilisateur peut exécuter la commande SHOW DATABASES ce qui permet de voir la liste des base de données du serveur.
- L'option skip-show-database force la possession du droit SHOW DATABASES pour voir la liste des bases de données.

Exemple de my.ini

```
secure_file_priv                = '/tmp'
skip_name_resolve               = 0
max_allowed_packet              = 16M
```

Moteurs de stockage et plug-ins
Plugin

- Quelques plugin MySQL.
- Présentation du plugin rewriter

Architecture MySQL

- L’architecture de MySQL est une architecture modulaire à base de plugin
- L'API de plug-in permet la création de plug-ins qui implémentent plusieurs fonctionnalités:
  - FullText
  - Storage
  - Authentification
  - Password
  - Demons ..

INSTALL PLUGIN

- INSTALL PLUGIN plugin_name SONAME 'shared_library_name'
- Cette déclaration installe un plugin dans le serveur. Il nécessite le privilège INSERT pour la table système mysql.plugin.

- plugin_name est le nom du plugin tel que défini dans la structure de descripteur de plugin contenue dans la bibliothèque

Plugin Storage Engine

L'architecture de moteur de stockage enfichable utilisée par MySQL Server permet aux moteurs de stockage d'être écrits en tant que plug-ins, puis chargés et déchargés sur un serveur en cours d'exécution. Pour une description de cette architecture

Disponible en nombre sur https://mariadb.com/kb/en/library/storage-engines/

Full-Text Plugins

- MySQL dispose d'un analyseur intégré qu'il utilise par défaut pour les opérations de texte Full Text (analyse du texte à indexer ou analyse d'une chaîne de requête pour déterminer les termes à utiliser pour une recherche). L'analyseur de texte intégral intégré est pris en charge avec les tables InnoDB et MyISAM.
- L'API du plugin vous permet d'utiliser un analyseur de texte intégral autre que l'analyseur de texte intégral intégré par défaut.
  - CREATE TABLE t
  - (
  - doc CHAR(255),
  - FULLTEXT INDEX (doc) WITH PARSER parser_name
  -) ENGINE=InnoDB;

Daemon Plugins

- Un plugin daemon est un type simple de plugin utilisé pour le code devant être exécuté par le serveur mais ne communiquant pas avec lui. Les distributions MySQL incluent un exemple de plug-in daemon qui écrit des messages de pulsation périodiques dans un fichier.

Plugin d'Authentification

- Des plug-ins d'authentification existent à la fois du côté serveur et du côté client.
- Les plug-ins côté serveur implémentent des méthodes d'authentification à utiliser par les clients lorsqu'ils se connectent au serveur. Un plug-in côté client communique avec un plug-in côté serveur pour fournir les informations d'authentification requises.
- Un plug-in côté client peut interagir avec l'utilisateur en effectuant des tâches telles que la sollicitation d'un mot de passe ou d'autres informations d'authentification à envoyer au serveur.

Plugin de Password

- Le serveur MySQL fournit une interface pour écrire des plugins testant les mots de passe. Un tel plugin implémente deux fonctionnalités:

- Rejet des mots de passe trop faibles dans les instructions attribuant des mots de passe (telles que les instructions CREATE USER et ALTER USER).

- Évaluation de la force des mots de passe potentiels pour la fonction SQL VALIDATE_PASSWORD_STRENGTH ().

Plugin Rewriter

- MySQL supporte les plugins de réécriture de requêtes qui peuvent examiner et éventuellement modifier les instructions SQL reçues par le serveur avant que le serveur ne les exécute.
- Un plugin côté serveur nommé Rewriter examine les instructions et peut les réécrire, en fonction de son cache en mémoire de règles de réécriture.

- Ces déclarations sont sujettes à la réécriture:
  - Depuis MySQL 8.0.12: SELECT, INSERT, REPLACE, UPDATE et DELETE.
  - Avant MySQL 8.0.12: SELECT uniquement.

Installation de Plugin

- Pour installer ou désinstaller le plugin Rewriter query rewrite, choisissez le script approprié situé dans le répertoire de partage de votre installation MySQL:

- install_rewriter.sql: Choisissez ce script pour installer le plug-in Rewriter et ses composants associés.

- uninstall_rewriter.sql: choisissez ce script pour désinstaller le plug-in Rewriter et ses composants associés.

Usage

- Nous allons commencer par l'exemple le plus simple possible: nous réécrirons la requête SELECT constante en SELECT constante +1
  - INSERT INTO query_rewrite.rewrite_rules(pattern, replacement) VALUES ('SELECT ?', 'SELECT ? + 1');
- On verifie l'existence et la règle et ses capacités
  - INSERT INTO query_rewrite.rewrite_rules(pattern, replacement) VALUES ('SELECT ?', 'SELECT ? + 1');
  - CALL query_rewrite.flush_rewrite_rules();

Usage

```
mysql> select 1;
+-------+
| 1 + 1 |
+-------+
|     2 |
+-------+

1 row in set, 1 warning (0.00 sec)
```

Maintenance d'un serveur MySQL

- Etat de session (variables, commande "show status"), arrêt forcé d'une session.
- Chargement : LOAD DATA, myimport, SELECT INTO OUTFILE, mysqldump.
- Journaux (général, erreurs, requêtes lentes...).
- Stratégies de sauvegarde.
- InnoDB Hot Backup, mysqlbinlog.
- Sauvegarde et récupération incrémentale.
- Planification.

Etat de session

Les variables de statut renseignent sur l'état et les performances du serveur et peuvent être affichées par la commande

Mysql> SHOW GLOBAL STATUS

Du fait du grand nombre de variable de statut, nous n'en donnons que les principales ici:

Etat de session

- Aborted: Aborted_clients et Aborted_connects indiquent respectivement le nombre de connexions mal terminées et le nombre de connexions infructueuses
- Log binaires: Les variables commençant par binlog renseignent sur les caches dédiés aux logs binaires et aux éventuels débordements
- Trafic réseau : Bytes_received et Bytes+sent renseignent respectivement sur la bande passante reçue et envoyée depuis le serveur
- Compteur de commandes: Les variables de statut débutant par com_ sont des compteurs et il y en a beaucoup..
- Elle recensent le nombre d'executions de toutes les commandes utilisables sous MySQL. Il est ainsi possible de compter le nombre de SELECT,UPDATE… afin de savoir si les données sont accéder majoritairement en lecture ou en écriture

Etat de session

- Compression: Compression est une variable de session uniquement qui indique si la compression est activé pour le client
- Connexion: Les variable de statu débutant par Connection donnent des indications sur les erreur au cours du processus de connexion. En particulier, Connection_errors _max_connections indique un dépassement du nombre de connexions permis et défini dans le paramètre max_connections (ce qui ne devrait pas se produire)
- Tables temporaires : Le préfixe created_tmp concerne la création des tables temporaires sur le disque et en mémoire
- Operation internes: Les variable débutant par Handler comptabilisent les opérations internes de MySQL (insertion, effacement, commit…)

Etat de session

- InnoDB: les variables de statut relatives au fonctionnement d'InnoDB.
- Statitiques sur les indes: Les statistiques sur les index sont stockées dans les variables de statut commentant par Key
- Last_query: Last_query_cost et Last_query_partial_plans sont réinitialisées après chaque explain. Elles représentent respectivement le cout arbitraire du meilleur plan estimé par l'optimiseur et le nombre d'itération pour le trouver.
- Open: Le préfixe Open désigne les tables et les fichiers ouverts. Une valeur trop importante de Opened_tables peut témoigner de la nécessité d'augmenter la valeur de la variable système table_open_cache

Etat de session

Perforance_schema : Les variables préfixées par performance_schema concernent la base performance_schema destinée a donner des indicateurs de performances sur l'activité du serveur.

Qcache: Les variables de statut débutant par qcache donnent des indications sur la mémoire tampon dédiée aux requêtes comme le nombre de sollicitations du cache ou la mémoire disponible

Queries et Questions: La variable de statut questions est le nombre de requêtes exécutées sur le serveur incluant celle depuis les procédures stockées. La variables question est identique mais sans les questions issues des Store Procedure

Réplication: Les variables préfixées par Rpl et Slave renseignent sur le fonctionnemnt de la réplication

Etat de session

- Select : Les variables Select caractérise les requetes select mal optimisées
- SSL: Les variables de statut débutant par SSL conernet les connexions sécurisées
- Table_locks : Table_locks_immediate et Table_locks_waited comptabilisent respectivement le nombde de demande de verrous obtenues immédiatement et en différé.
- Table_open_cache: Il s'agit du préfixé dédié aux variables de statut portant sur l'utilisation du cache
- Transactions: Les variables de statut concernant les transactions sont préfixé par Tc
- Uptime et Uptime_since_flush_status: Ces deux variables de statut désignent le temps de fonctionnement du serveur depuis son démarrage et le temps écoulé depuis le dernier flush status

Maintenance d'un serveur MySQL
Chargement : LOAD DATA, myimport, SELECT INTO OUTFILE, mysqldump

- Comment charger des données dans une base de données
- Différentes méthodes de chargement

Chargement de données

- Lors de l'insertion de nouvelles données dans MySQL, les éléments qui prennent du temps sont les suivants: (par ordre d'importance):
  - Synchroniser les données sur le disque (dans le cadre de la fin des transactions)
  - Ajout de nouvelles clés. Plus l'index est grand, plus la mise à jour des clés prend du temps.
  - Vérification contre les clés étrangères (si elles existent).
  - Ajout de lignes au moteur de stockage.
  - Envoi de données au serveur.

Désactivation des clés

- Vous pouvez désactiver temporairement la mise à jour d'index non uniques. Cela est surtout utile lorsqu'il n'y a aucune (ou très peu) de lignes dans la table dans laquelle vous insérez des données.

```
ALTER TABLE table_name DISABLE KEYS;
BEGIN;
... inserting data with INSERT or LOAD DATA ....
COMMIT;
ALTER TABLE table_name ENABLE KEYS;
```

Désactivation des clés

- Lors de l'insertion de grandes quantités de données, les contrôles d'intégrité prennent un temps considérable. Il est possible de désactiver les index UNIQUE et les contrôles de clés étrangères à l'aide des variables système unique_checks et foreign_key_checks

```
SET @@session.unique_checks = 0;
SET @@session.foreign_key_checks = 0;
```

Chargement de fichiers

Le moyen le plus rapide d'insérer des données dans MySQL consiste à utiliser la commande LOAD DATA INFILE.

Cela ne se fait pas aussi vite que de lire le fichier côté serveur, mais la différence n'est pas si grande.

LOAD DATA est très rapide car:

- il n'y a pas d'analyse de SQL.
- les données sont lues en gros blocs.
- si la table est vide au début de l'opération, tous les index non uniques sont désactivés pendant l'opération.
- Le moteur doit d'abord mettre en cache les lignes, puis les insérer dans de gros blocs .
- pour les tables vides, certains moteurs transactionnels ne consignent pas les données insérées dans le journal des transactions, car il est possible d'annuler l'opération en effectuant simplement un TRUNCATE sur la table.

SELECT ... INTO Syntaxe

La requête SELECT ... INTO de SELECT permet à un résultat de requête d'être stocké dans des variables ou écrit dans un fichier:

- SELECT ... INTO var_list sélectionne les valeurs de colonne et les stocke dans des variables.
- SELECT ... INTO OUTFILE écrit les lignes sélectionnées dans un fichier. Les terminateurs de colonne et de ligne peuvent être spécifiés pour produire un format de sortie spécifique.
- SELECT ... INTO DUMPFILE écrit une seule ligne dans un fichier sans aucune mise en forme.

SELECT ... INTO

- Le formulaire SELECT ... INTO OUTFILE 'nom_fichier' de SELECT écrit les lignes sélectionnées dans un fichier. Le fichier est créé sur l'hôte du serveur.
- Vous devez donc disposer du privilège FILE pour utiliser cette syntaxe. nom_fichier ne peut pas être un fichier existant, ce qui empêche notamment la destruction de fichiers tels que / etc / passwd et de tables de base de données. La variable système character_set_filesystem contrôle l'interprétation du nom de fichier.
- L'instruction SELECT ... INTO OUTFILE est principalement destinée à vous permettre de vider très rapidement une table dans un fichier texte sur la machine serveur.
- Si vous souhaitez créer le fichier résultant sur un autre hôte que l'hôte serveur, vous ne pouvez généralement pas utiliser SELECT ... INTO OUTFILE car il n'existe aucun moyen d'écrire un chemin d'accès au fichier par rapport au système de fichiers de l'hôte serveur.

SELECT ... INTO

FIELDS ESCAPED BY détermine comment écrire des caractères spéciaux. Si le caractère FIELDS ESCAPED BY n'est pas vide, il est utilisé si nécessaire pour éviter toute ambiguïté en tant que préfixe précédant les caractères suivants en sortie:

- Les champs échappés par caractère
- Les champs [facultatif] inclus par caractère
- Le premier caractère des valeurs FIELDS TERMINATED BY et LINES TERMINATED BY
- ASCII NUL (l'octet de valeur zéro; ce qui est réellement écrit après le caractère d'échappement est ASCII 0, et non un octet de valeur zéro)

REPLACE/IGNORE

- Les mots clés REPLACE et IGNORE contrôlent le traitement des lignes d'entrée qui dupliquent les lignes existantes avec des valeurs de clé uniques:
  - Si vous spécifiez REPLACE, les lignes d'entrée remplacent les lignes existantes. En d'autres termes, les lignes qui ont la même valeur pour une clé primaire ou un index unique qu'une ligne existante.
  - Si vous spécifiez IGNORE, les lignes qui dupliquent une ligne existante sur une valeur de clé unique sont ignorées.
- Si vous ne spécifiez aucune de ces options, le comportement dépend du type de mot-clé LOCAL spécifié.
  - Sans LOCAL, une erreur se produit lorsqu'une valeur de clé en double est trouvée et que le reste du fichier texte est ignoré.
  - Avec LOCAL, le comportement par défaut est le même que si IGNORE est spécifié. En effet, le serveur ne dispose d'aucun moyen pour arrêter la transmission du fichier en cours d'opération.

SELECT ... INTO

```
SELECT  * from table INTO OUTFILE 'D:\\Temp\somefile‘


SELECT a,b,a+b INTO OUTFILE '/tmp/result.txt' FIELDS
TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES
TERMINATED BY '\n' FROM test_table;
```

LOAD DATA INFILE

L'instruction LOAD DATA INFILE lit les lignes d'un fichier texte dans une table à une vitesse très élevée.

LOAD DATA INFILE est le complément de SELECT ... INTO OUTFILE.

Le serveur utilise le jeu de caractères indiqué par la variable système character_set_database pour interpréter les informations du fichier.

LOAD DATA INFILE

- Si LOCAL est spécifié, le fichier est lu par le programme client sur l'hôte client et envoyé au serveur. Le fichier peut être donné en tant que chemin complet pour spécifier son emplacement exact. S'il est indiqué sous forme de nom de chemin relatif, le nom est interprété par rapport au répertoire dans lequel le programme client a été démarré.
- Si LOCAL n'est pas spécifié, le fichier doit être situé sur l'hôte du serveur et est lu directement par le serveur. Le serveur utilise les règles suivantes pour localiser le fichier:
  - Si le nom de fichier est un chemin absolu, le serveur l'utilise comme indiqué.
  - Si le nom de fichier est un chemin d'accès relatif avec un ou plusieurs composants principaux, le serveur recherche le fichier par rapport au répertoire de données du serveur.

Load Data

```
LOAD DATA INFILE 'data.txt' INTO TABLE db2.my_table;

LOAD DATA INFILE '/tmp/test.txt' INTO TABLE test FIELDS
TERMINATED BY ',' LINES STARTING BY 'xxx';

LOAD DATA LOCAL INFILE 'C:/path/to/mytable.txt' IGNORE
INTO TABLE mytable
FIELDS TERMINATED BY '\t' LINES TERMINATED BY '\r\n'
```

MySQL Import

- L'utilitaire mysqlimport fournit en ligne de commande shell, l'équivalent de la commande SQL LOAD DATA INFILE.
- Il convient de s'assurer de la qualité des données à integrer avant de recourir à ce type d'import, car les doublons sur les clefs primaires et autres contraintes éventuelles sur les définitions de colonnes ne sont pas contrôlés.
- Si le fichier store.csv est issue de la commande mysqldbexport, alor le nom du fichier une fois dépossédé de son extension par mysqlimport est implicitement utilisé pour désigner la table dans laquelle les données doivent être importées

mysqlimport

```
shell> mysqlimport --local test imptest.txt
test.imptest: Records: 2  Deleted: 0  Skipped: 0  Warnings:
0
shell> mysql -e 'SELECT * FROM imptest' test
+------+----------------+
| id   | n              |
+------+----------------+
|  100 | Max Sydow      |
|  101 | Count Dracula  |
+------+----------------+
```

Mysqlimport

- Les options de séparateur, terminaison de champ et de ligne trouvent leurs équivalents dans l'instruction LOAD DATA INFILE.
- Afin de paralléliser le chargement de plusieurs fichier, il est possible de recourir à l'option –user-threads=N ainsi qu'à l'option –multiprocess qui affecte un processus au niveau de la base sous Windows et au niveau de la table sous Linux.

MySQLDump

- L'utilitaire mysqldump est l'outil consacré pour réaliser les sauvegardes.
- Sauvegarde d'une seul table:
  - Création dun fichier dump de la table neu.STORE
  - Mysqldump –u root neu STORE>store_dump.sql
- Sauvegarde de plusieurs tables
  - Création d'une sauvegarde des tables neu.STORE et neu.CLIENT
  - Mysqldump –u root neu–tables STORE CLIENT>dump.sql
- Sauvegarde d'une base de données
  - Sauvegarde de la base neu complete
  - Mysqldump –u root neu>neu.sql

MySQLDump

- Sauvegarde de plusieurs base
  - Mysqldump –u root –database neu neu2>neu.sql
- Sauvegarde de toutes les bases
  - Afin de sauvegarder toutes les bases d'une instance, on peut utiliser l'option –all-databases
  - Mysqldump –u root –all-databases>database.sql
- Inclusion des fonctions et procédures stockées
  - Les triggers sont inclus dans les sauvegardes, mais ce n'est pas le cas des fonctions et des procédures stockées. Il est nécéssaire pour les avoir, de recourir à l'option –routines
  - Mysqldump –u root –routines neu>neu.sql

Consistance de la sauvegarde

- Il convient de distinguer les tales s'appuyant sur un moteur transactionnel comme InnoDB et celles s'appuyant sur un moteur non transactionnel comme MyISAM.
- Dans le cas des tabels transactionnelles il est préférable de recourir à l'option –single-transaction.
- Dans ce case, la lecture de la base commence avec une instruction START TRANSACTION.
- Attention toutefois de ne pas modifier, renommer ou supprimer les structures de la table pendant la sauvegarde car ces opérations cassent la consistance de la base

Consistance de la sauvegarde

- Les tables non transactionnelles doivent être verrouller avant d'être sauvegarder.
- L'option –lock-tables verroucle les tables de la base dont la sauvegarde est en en cours, mais ne garantit par la consistance entre les différentes bases de données.
- Si la cohérence des données est vitale, il convient de recourir à l'option –lock-all-tables.

Accélération de la sauvegarde des données

- L'intégration de données au format texte par LOAD DATA INFILE est plus rapide que via des INSERT. Néanmoins, il n'y a plus de contrôle de cohérence des données lors de l'insertion.
- La valeur de la variable bulk_insert_buffer_size peut également jouer sur les performances.
- MySQL peut utiliser des tables temporaires pour insérer des données de façon massive. Il est dans ce cas intéressant de modifier les variables tmp_table_size et max_heap_table_size.

Maintenance d'un serveur MySQL
Journaux (général, erreurs, requêtes lentes...).

Log d'Erreur

- Le fichier de logs d'erreur contient des messages précieux qui passe inapercus si l'on ne pense pas à les inspecter régulièrement.
- Dans le fichier de configuration my.cnf, la variable log_error contient l'emplacement du fichier des log d'erreur.
- En général vous trouverez la valeur /var/log/mysql/error.log
- Dans ce fichier le message d'erreur affiche une idée des erreurs
- Il est possible de rediriger les erreurs vers soit syslog soit event viewer.

Log d'Erreur

```
log_error                          = <hostname>_error. #
log_timestamps                     = SYSTEM # MySQL 5.7,
equivalent to old behaviour
log_warnings                  = 2
# log_error_verbosity                = 3
# MySQL 5.7, equivalent to log_warnings = 2
# innodb_print_all_deadlocks     = 1
# wsrep_log_conflicts                = 1
# for Galera only!
```

Mise en place d'un writer de log

- La mise en place des log vers syslog se fait via le chargement de component_log_sink_syseventlog

```
INSTALL COMPONENT 'file://component_log_sink_syseventlog';
SET GLOBAL log_error_services = 'log_filter_internal;
log_sink_syseventlog';
```

Fermer un journal de log

- Si vous effacez le journal des erreurs à l'aide de FLUSH ERROR LOGS, de FLUSH LOGS ou de mysqladmin flush-logs, le serveur se ferme et rouvre tout fichier du journal des erreurs qu'il écrit. Pour renommer un fichier journal des erreurs, faites-le manuellement avant le vidage. Le vidage des journaux ouvre alors un nouveau fichier portant le nom de fichier d'origine.

```
mv host_name.err host_name.err-old
mysqladmin flush-logs
mv host_name.err-old backup-directory
```

Requêtes lentes

- Afin d'optimiser un serveur, il est indispensable d'activer l'historisation des requêtes lentes.
- Le paramétrage suivant, dans le fichier my.cnf, commande de stocker les requêtes de plus de 2 secondes dans une table mysql.slowçlog.
- La variable log-uqeries-not-using indexe, quand a elle historise dans la même table les requetes mettant en œuvre des jointures sans index.

Long_query_time

```
long_query_time=2
log_slow_queries=ON
log_output=TABLE
log-queries-not-using-indexes



SELECT * from mysql.slow_log WHERE start_time BETWEEN ....
AND .. AND query_time>2 ORDER bu query_time
```

Statut de log_slow_queries

```
mysql> SHOW VARIABLES LIKE 'log_slow_queries';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| log_slow_queries | OFF   |
+--------------------+-------+
```

Slow Query Log

- Si vous ne souhaitez pas ecrire dans une table, les requetes lentes seront écrite dans un fichier.
- Notez que si vous ne précisez aucun chemin, le journal est créé dans le répertoire de données (ce qui n'est pas idéal) et porte le nom *hostname*-slow.log. Le mieux est donc de préciser un chemin absolu, par exemple :

  [mysqld]
  log_slow_queries = /var/log/mysql/mysql-slow.log

- La seconde option importante à configurer est la durée à partir de laquelle une requête est considérée comme lente. Ce réglage se fait avec la variable long_query_time, qui peut être modifiée dynamiquement :

```
mysql> SHOW GLOBAL VARIABLES LIKE 'long_query_time';
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| long_query_time | 10    |
+-----------------+-------+
1 row in set (0.00 sec)
```

My.ini

```
# Slow Query Log


slow_query_log_file                 = <hostname>_slow.log
# Adjust AppArmor configuration:
/etc/apparmor.d/usr.sbin.mysqld
slow_query_log                      = 0
log_queries_not_using_indexes       = 0
long_query_time                     = 0.5
min_examined_row_limit              = 100
```

Maintenance d'un serveur MySQL
InnoDB Hot Backup, mysqlbinlog

- Présentation de mysqlbinlog
- Comment faire un backup avec InnoDB

mysqlbackup

La commande mysqlbackup vous permet de sauvegarder une instance MySQL en cours d'exécution, y compris les tables InnoDB, en perturbant le moins possible les opérations tout en produisant un instantané cohérent de la base de données.

Lorsque mysqlbackup copie des tables InnoDB, les lectures et les écritures dans les tables InnoDB peuvent continuer. Mysqlbackup peut également créer des fichiers de sauvegarde compressés et sauvegarder des sous-ensembles de tables et de bases de données. En conjonction avec le journal binaire MySQL, les utilisateurs peuvent effectuer une récupération à un point dans le temps. Mysqlbackup fait partie de l'abonnement MySQL Enterprise.

MySQLBinLog

- Les logs binaires enregistrés par MySQL ne sont pas lisibles directement. L'utilitaire mysqlbinlog sait les convertir afin de permettre l'exploitation et leur affichage.
- Il est possible d'identifier les transaction responsables d'un lag de replication, rechercher un erreur ou encore effectuer une restauration partielles des commande MySQL
- Pour lire un fichier de logs binaires, la syntaxe est la suivante
- Mysqlbinlog /var/log/mysql/mysql-bin.00001

Obtenir une liste des journaux binaires actuels

- Depuis mysql, exécutez la commande suivante show binary logs qui affichera tous les journaux binaires de votre système.

```
mysql> SHOW BINARY LOGS;
+------------------+-----------+
| Log_name         | File_size |
+------------------+-----------+
| mysqld-bin.000001 |     15740 |
| mysqld-bin.000002 |      3319 |
..
..
```

MySQLBinLog

```
mysqlbinlog mysqld-bin.000001

/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET @OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#170726 14:57:37 server id 1  end_log_pos 106   Start: binlog v 4, server v 5.1.73-log created 170726 14:57:37 at
startup
# Warning: this binlog is either in use or was not closed properly.
ROLLBACK/*!*/;
BINLOG '
IeZ4WQ8BAAAAZgAAAGoAAAABAAQANS4xLjczLWxvZwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAh5nhZEzgNAAgAEgAEBAQEEgAAUwAEGggAAAAICAgC
'/*!*/;
# at 106
#170726 14:59:31 server id 1  end_log_pos 182   Query   thread_id=2     exec_time=0     error_code=0
SET TIMESTAMP=1501095571/*!*/;
SET @@session.pseudo_thread_id=2/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=1, @@session.unique_checks=1,
@@session.autocommit=1/*!*/;
SET @@session.sql_mode=0/*!*/;
SET @@session.auto_increment_increment=1, @@session.auto_increment_offset=1/*!*/;
/*!\C latin1 *//*!*/;
SET @@session.character_set_client=8,@@session.collation_connection=8,@@session.collation_server=8/*!*/;
..
..
..
# at 14191
#170726 15:20:38 server id 1  end_log_pos 14311         Query   thread_id=4     exec_time=0     error_code=0
SET TIMESTAMP=1501096838/*!*/;
insert into salary(name,dept) values('Ritu', 'Accounting')
/*!*/;
DELIMITER ;
# End of log file
ROLLBACK /* added by mysqlbinlog */;
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
```

Obtenir des entrées pour une base de données spécifique

- La sortie par défaut de mysqlbinlog peut être écrasante, car elle affichera beaucoup de données de toutes les déclarations. En utilisant l'option -d, vous pouvez également spécifier un nom de base de données, qui s'affichera sur les événements qui se produisent sur la base de données donnée.

```
mysqlbinlog -d crm mysqld-bin.000001 > crm-events.txt
```

Désactiver le journal binaire pour la récupération

- Lorsque vous utilisez mysqlbinlog pour effectuer une restauration à partir d'une panne de base de données, vous ne souhaitez pas que votre processus de restauration crée des journaux binaires.
- Si tel est le cas, vous serez dans une boucle et vous continuerez à restaurer, car la restauration elle-même générera de nouveaux fichiers journaux binaires.

```
mysqlbinlog --disable-log-bin mysqld-bin.000001
```

Afficher uniquement les requêtes SQL dans la sortie

- Par défaut, comme vous le voyez dans la sortie des exemples précédents, outre les requêtes SQL, vous verrez également des informations supplémentaires dans la sortie de mysqlbinlog. Si vous souhaitez uniquement les requêtes SQL classiques et rien d'autre, utilisez l'option -s comme indiqué ci-dessous.

```
mysqlbinlog -short-form mysqld-bin.000001
SET TIMESTAMP=1501096106/*!*/;
insert into employee values(400,'Nisha','Marketing',9500)
/*!*/;
SET TIMESTAMP=1501096106/*!*/;
insert into employee values(500,'Randy','Technology',6000)
```

Afficher les entrées à partir d'une heure spécifique

- Cela s'avère très utile lorsque vous souhaitez extraire des données d'un fichier binaire à partir d'un laps de temps spécifique uniquement, afin de récupérer ou de reconstruire certaines activités de la base de données ayant eu lieu au cours de cette période.

```
mysqlbinlog --start-datetime="2017-08-16 15:00:00" mysqld-
bin.000001
```

Sauvegarde et récupération incrémentale

- Notion de sauvegarde incrémental.
- Comment récuperer les données sur une période

Sauvegarde et restauration

Avoir sauvegardé ses données et une bonne chose, mais la procédure de restauration doit être testée et validée.

La sauvegarde des données s'entend comme étant les fichiers de données et les fichier de configuariton.

La fréquence des sauvegardes, la durée de rétention des journaux binaires et la fréquence de synchronisation des fichier de sauvegarde doit être en adéquation avec la perte de données maximal admissible (RPO, recovery Point Objective).

RPO de deux heures

Si le RPO est de deux heures, une solition est :

- Activer les journaux binaires (log-bin)
- Forcer le changemetnd ejournal toutes les deux heures en rajoutant en crontab
  - Mysql –defaults-file=… -e « FLUSH LOGS »
- Synchroniser les journaux binaires sur un serveur distant toutes les deux heures
- Effectuer une sauvegarde quotidienne en fonction de la quantité de modification.

Si vous avez mise en place la réplication, vous pourrez effectuer les sauvegardes sur un esclave.

Sauvegarde et replication

La réplication n'est pas une méthode de sauvegarde, mais une méthode pour faciliter la sauvegarde.

En effet, si un utilisateur évoque un drop table dans un schema, la commande seras répliqué sur l'esclave et aucune sauvegarde ne seras évidemment faites.

RTO

Un autre paramètre important est le temps de restauration (RTO Recovery Time Objective), il sagit du temps imparti pour reconstruire les données.

Sauvegarde logique

La sauvegarde logique consiste à extraire de la base de données les instructions SQL permettant de regénérer la structure, les données, les procédures …

L'export des données nécéssite que le serveur MySQL soit démarré car c'est lui qui accède au fichiers de données.

L'outil utilisé est mysqldump ou faire des SELECT INTO.

- Les sauvegardes logiques ont les avantages suivants:
- Ce sont des fichiers textes visualisable simplement
- La sauvegarde est flexible due au option de mysqldump
- La restauration est simple et flexible.

Néanmoins, l'inconvénient est le temps important pour restauration (difficile pour des bases supérieurs à 20 Go)

Sauvegarde physique

La sauvegarde physique consite à sauvegarder directement les fichiers avec les outils systèmes.

La taille de la sauvegarde physique est souvent importante car nous allons sauvegarder en sus des données, les index.

Principalement que le serveur soit en route ou pas, cela consiste a archiver le data_directory.

La sauvegarde physique est simple (une copie de fichier) et portable directement vers une autre instance.

Néanmoins, leur taille et la potentialité que le fichier soit corrompus est un désavantage.

Sauvegarde incrémentale

Lors de la sauvegarde il est parfois complexe de faire une sauvegarde complete:

- Car cela prend du temps et peut dépasser la durée du RPO/RTO.
- La place sur le disque peut être limiter pour effectuer la sauvegarde complète.

Si vous souhaitez utiliser les sauvegardes incrémentales, il faudra commencer par une sauvegarde complète.

Il est possible de faire une sauvegarde complète en fin de semaine et de faire une sauvegarde incrémental chaque jour.

Sauvegarde incrémentale

La récupération à un moment donné repose sur les principes suivants:

- La source d'informations pour la récupération à un point dans le temps est l'ensemble des sauvegardes incrémentielles représentées par les fichiers journaux binaires générés à la suite de l'opération de sauvegarde complète. Par conséquent, le serveur doit être démarré avec l'option --log-bin pour activer la journalisation binaire
- Pour voir une liste de tous les fichiers journaux binaires, utilisez cette déclaration:

```
mysql> SHOW BINARY LOGS;
```

Récupération à un point dans le temps à l'aide du temps

Pour indiquer les heures de début et de fin de la récupération, spécifiez les options --start-datetime et --stop-datetime pour mysqlbinlog, au format DATETIME.

A titre d'exemple, supposons que le 20 avril 2005 à 10 h 00, une instruction SQL qui supprime une grande table soit exécutée. Pour restaurer la table et les données, vous pouvez restaurer la sauvegarde de la nuit précédente, puis exécuter la commande suivante:

```
shell> mysqlbinlog --stop-datetime = "2005-04-20 9:59:59" \
        /var/log/mysql/bin.123456 | mysql -u root -p
```

MySQL Optimisation

- Optimiser le serveur MySQL
- Les Caches
- Optimisation du schéma et des requetes
- Outils

Materiel et système d'exploitation

- Les architectures avec plusieurs processeurs ou plusieurs cœurs étant les plus courantes, on peut se demander s'il vaut mieux privilégier la vitesse ou le nombre de processeurs.
- Sur un serveeur esclave, vous aurez tout intérêt à privilégier la vitesse du processeur sur nombre de cœurs de manières a ce que la réplication reste la plus synchrone possible
- Sur un serveur maître, plu de cœurs impliquent une meilleur capacité à traiter des connexions simultanées.

Mémoire vive

La mémoire vive jour un rôle crucial dans les performance d'un serveur MySQL.

Avec InnoDB, la mémoire vive trouve son utilité pour les écritures car lorsqu'il est possible de conserver pendant un certain temps des modification de données dans le cache mémoires.

L'estimation de la mémoire se fait en fonction de la taille de la base de données (données+index), de l'évolution de volumétrie et de la quantité de données utile dans l'instant.

Mémoire

Regardez l'evolution de la variable Innodb_buffer_pool_reads lorsque le serveur est chargé.

Innodb_buffer_pool_reads est le nombre de lectures logiques que InnoDB n'a pas pu traiter à partir du pool de mémoire tampon et doit être lu directement à partir du disque.

Si les valeurs sont proches de 0 quasiment toutes les lectures de données sont faites en mémoire en alors que si les valeurs sont proches du nombre de lectures par seconde que peut fournir le disque, autgmeenter la mémoire et la taille du buffer pool va augmenter les performances.

Mémoire

- Supposons un disque ayant 2000 operation/seconde
- Effectuer un mysqladmin ext –ri1 | greo Innodb_buffer_pool_reads
- (Ignorer la première ligne donnant le nombre de lecture disque le démarage du serveur).

```
Innodb_buffer_pool_reads 141
Innodb_buffer_pool_reads 150
Innodb_buffer_pool_reads 157
```

Disque

La performance d'un disque se mesure en temps d'accès aux données et en débit.

Si l'applicatif effectue un grand nombre de petites requètes, alors le discriminant est le temps d'accès.

Si l'applicatif a besoin de beaucoup de données (application statistique) alors le discriminant est le débit de données.

Par ailleur la localité des données est augmenté si le disque est nettement supérieur à la taille de la base de données.

Disque

En règle général, il faut pour les disques SSD et les disques dure essayez de placer ceux-ci en RAID matériel avec Write Back

Les disques PCI sont mal utilisé par MySQL, et donc sont interessant pas avec un cout mal justifié. En effet, le moteur a été écrit principalement sur avec des disques « normaux ».

Optimisation du schéma

- L'idée est de voir l'utilisation des types de données
- Normalisé les schemas

Principes généraux

LA recherche du meilleur type de données possible est souvent une taches négligés car fastidieuse.

Néanmoins, cette phase est importante car lorsque la base est mise en production, le changement de type de données est complexe et risqué.

L'idée est que plus le type de données est simple et complact, plus il sera légéer et performant.

Enfin, il faut essayer d'éviter les types nullable sauf en cas de besoin impérieux. En effet , les types NULL sont un surcroit de travail pour la base.

Nombres

Les données numériques se classent en deux catégories les entiers et les nombres réels.

Pour les entiers le type le plus courant est INT (quatre octet), mais il existe également les tyeps TINYINT (un octet), SMALLINT(deux octets), MEDIUM INT (trois octets) et BIGINT (huit octets).

L'optimisation consiste à choisir le type le plus petit possible tout en ayant une plage de valeurs suffisante pour stocker toutes les valeurs possibles.

Lorsque seul les nombres positifs sont a gérer, alors il est possible de désigner le type comme étant UNSIGNED (ce qui élimine la possibilité d'avoir des nombres négatifs).

Nombres décimaux

Pour les nombres réels, vous avez le choix entre DECIMAL d'une part, FLOAT et DOUBLE d'autre part.

DECIMAL offre une fenêtre de valeurs plus petite mais stocker celles-ci de manière exacte.

Ainsi DECIMAL(5,2) permet de stocker des nombres ayant cinq chiffres significatifs dont deux après la virgules soit des cnombres complris entre -999,99 et 999,99

FLOAT et DOUBLE stockent au contraire toujours des valeurs approchées.

Chaîne de caractères

Il existe deux catégories de chaînes de caractères: les types CHAR/VARCHAR et les variantes de TEXT.

Les champs de chaînes de caractères possèdent tous lors de leurs définition un jeu de caractères et un interclassement.

Le jeu de caractère est le nombre de caractère possible (latin1 pour les applications en français, UTF8 pour les applications international) et occupe une place différente (de 1 à 4 octet par caractère).

L'interclassement est l'ordre des différents caractères les uns pas rapport au autres.

Chaîne de caractères

Le suffixe des interclassement permet de signaler si ce dernier est sensible/insensible à la casse ou encore.

Les champs CHAR ont une longeur fixe alors que les champs VARCHAR ont une longeur variable.

- CHAR(5) contiendras toujours 5 caractères et en fonction du jeux de caractère de 5 à 20 octets
- VARCHAR(5) contiendras au plus 5 caractère et en fonction du jeux de caractère et 5 à 20 octets.

Avec InnoDB, il est conseillé de toujours utilisé les champs VARCHAR. A contrario, le moteur Memory considèrent les VARCHAR comme des CHAR.

Chaîne de caractères

Le type TEXT sont des BLOB de caractère et ses variantes permettent de stocker des chaînes variablees plus longeus qu'avec des VARCHAR.

Néanmoins, les champs TEXT/BLOB:

- Ne peuvent pas avoir d'INDEX sur la totalité du champ, seul un préfixe est prise en compte.
- Ne sont pas stocké dans la table mais dans un espace dédié (partionnement vertical).
- Les tables temporaires utilisant des TEXT seront toutes crée sur le disque et non pas en mémoire.

ENUM/SET

ENUM et SET sont deux types de données très pratiques prenant en paramètre une liste de chaîne de caractères et stockant les résultats sou forme d'entier.

ENUM permet d'enregistrer une valeur parmi la liste de paramètre alors que SET permet d'enregistrer une ou plusieurs valeurs.

Un champ ENUM accepte jusqu'à 65535 valeurs possibles (1 ou deux octets) et SET accepte jusqu'à 64 paramètres et demande de 1 à 8 octets selon le nombre de paramètres.

Notion de base de données MySQL

- L'idée générale est d'avoir des « trucs » pour créer un schéma « résistant » au temps
- Quelques objets théoriques aident à modéliser ce schéma (en fait à éviter les erreurs)
- Le but est de séparer la modélisation des données de l'utilisation métier.
- On parle ainsi de normalisation de la base.

Objet théorique

- **1FN - première forme normale:**
  - tout attribut contient une valeur atomique
  - tous les attributs sont non répétitifs
  - tous les attributs sont constants dans le temps.

Le non-respect des deux premières conditions de la 1FN rend la recherche parmi les données plus lente parce qu'il faut analyser le contenu des attributs. La troisième condition quant à elle évite qu'on doive régulièrement mettre à jour les données.

1FN en pratique

Au lieu de :

Produit	Fournisseur
Téléviseur	VIDEO SA, HITEK LTD

Faire:

Produit	Fournisseur
Téléviseur	VIDEO SA
Téléviseur	HITEK LTD

1FN en pratique: une valeur atomique

- *L'attribut NOM est composé de 2 attributs atomiques.*
- Au lieu de :

CLIENT_ID	NOM
1	Gérard Dupont
2	Léon Durand

- Faire:

CLIENT_ID	NOM	PRENOM
1	Dupont	Gérard
2	Durand	Léon

1FN pratique: les attributs sont non répétitifs

- *L'attribut FOURNISSEURS est une liste.*
- *Au lieu de*

PRODUIT_ID	DESCRIPTION	FOURNISSEURS
1	Téléviseur	Sony, Sharp, LG

- *Faire*

Produit_id	Fournisseur_id
1	Sony
1	Sharp
1	LG

PRODUIT_ID	DESCRIPTION
1	Téléviseur

1FN en pratique: Constance des attributs

- *L'attribut AGE n'est pas constant dans le temps.*
- *Au lieu de :*

NOM	PRENOM	AGE
BOB	Eponge	35

- *Faire:*

NOM	PRENOM	DATE DE NAISANCE
BOB	Eponge	1977

Notion de base de données MySQL: Objet théorique

- **2FN - deuxième forme normale**
  - tous les attributs non-clés sont totalement dépendants fonctionnellement de la totalité de la clé primaire.

Le non-respect de la 2FN entraîne une redondance des données qui encombrent alors inutilement la mémoire et l'espace disque.

2FN en pratique

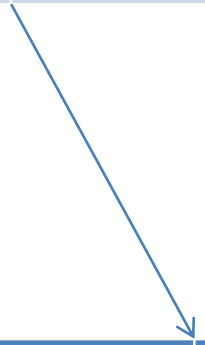
- Au lieu de :

Produit	Fournisseur	Adresse fournisseur
téléviseur	VIDEO SA	13 rue du cherche-midi
écran plat	VIDEO SA	13 rue du cherche-midi
téléviseur	HITEK LTD	25 Bond Street

2FN en pratique

- Faire:

Produit	Fournisseur
téléviseur	VIDEO SA
écran plat	VIDEO SA
téléviseur	HITEK LTD



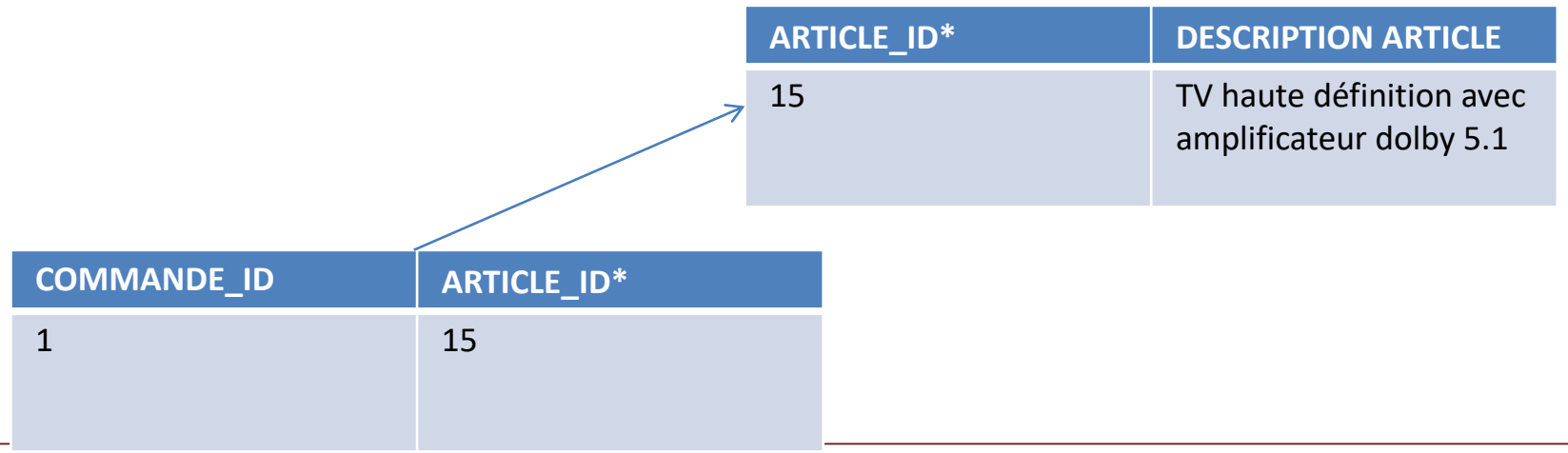
Fournisseur	Adresse
VIDEO SA	13 rue du cherche-midi
HITEK LTD	25 Bond Street

2FN en pratique

- *L'attribut DESCRIPTION_ARTICLE ne dépend que d'une partie de la clef primaire.*
- *Au lieu de :*

COMMANDE_ID	ARTICLE_ID*	DESCRIPTION ARTICLE
1	15	TV haute définition avec amplificateur dolby 5.1

- Faire

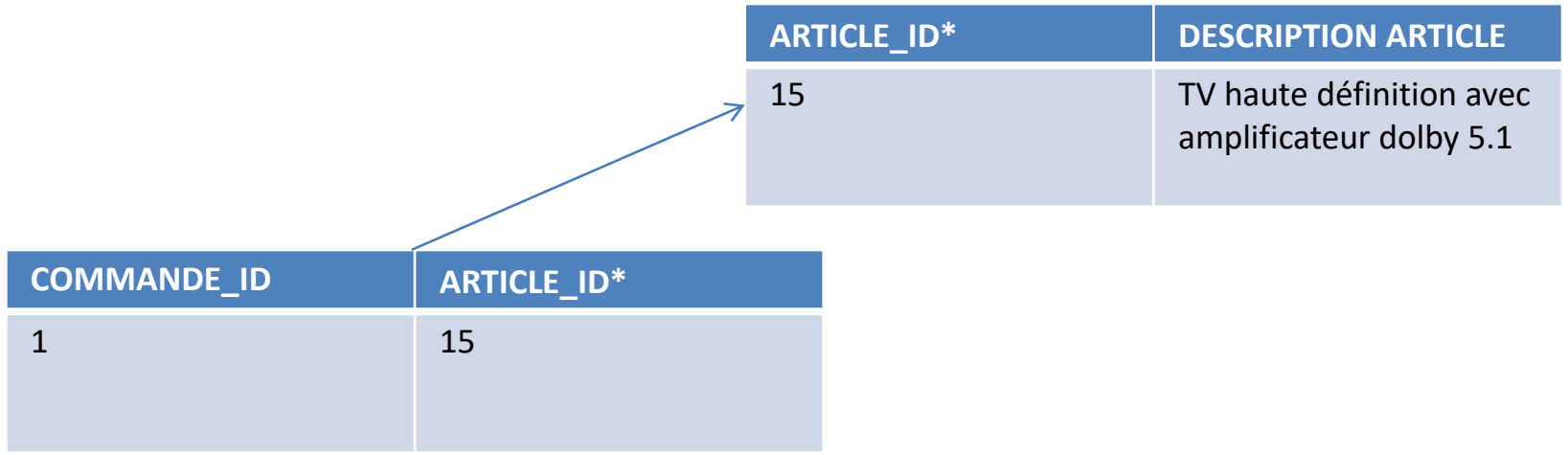


2FN en pratique

- *L'attribut DESCRIPTION_ARTICLE ne dépend que d'une partie de la clef primaire.*
- *Au lieu de :*

COMMANDE_ID	ARTICLE_ID*	DESCRIPTION ARTICLE
1	15	TV haute définition avec amplificateur dolby 5.1

- Faire



Objet théorique

- **3FN - troisième forme normale**
  - tout attribut n'appartenant pas à une clé ne dépend pas d'un attribut non clé

Le non-respect de la 3FN peut également entrainer une redondance des données.

3FN en pratique

- *L'attribut NOM_CLIENT dépend de CLIENT_ID.*
- *Au lieu de:*

COMMANDE_ID	CLIENT_ID	NOM_CLIENT
1	1	Durand

- *Faire :*

COMMANDE_ID	CLIENT_ID
1	1

CLIENT_ID	NOM_CLIENT
1	Durand

Objet théorique

- **FNBC - forme normale de Boyce – Codd**
  - Si une entité ou une relation en troisième forme normale a une clé composée, aucune des propriétés élémentaires de cette clé ne doit être en dépendance fonctionnelle d'une autre propriété.

Le non-respect de la 2FN, 3FN et la FNBC entraîne de la redondance. Une même information étant répétée un nombre considérable de fois.

FNBC en Pratique

- *Si Durand arrête d'enseigner les Mathématiques, on supprime la ligne et l'on perd la relation Matière-Salle.*

ENSEIGNANT_ID	MATIERE_ID	SALLE_ID
DURAND	MATHS	3A
DUPONT	ANGLAIS	6A

FNBC en Pratique

- *Si Durand arrête d'enseigner les Mathématiques, on supprime la ligne et l'on perd la relation Matière-Salle.*

ENSEIGNANT_ID	MATIERE_ID	SALLE_ID
DURAND	MATHS	3A
DUPONT	ANGLAIS	6A

Objet théorique

- **FNDC - forme normale domaine clef**
  - Une relation est en FNDC si et seulement si toutes les contraintes sont la conséquence logique des contraintes de domaines et des contraintes de clefs qui s'appliquent à la relation.

FNDC En pratique

- *On remarque que le type VL (véhicule léger) ou PL (poids lourd) est déterminé par la valeur du PTAC.*
- *Au lieu de :*

CONSTRUCTEUR	MODELE	TYPE	PTAC
Renault	Estafette	VL	2500
Iveco	Eurostar 440	PL	19000
Berliet	GDM 1934	PL	15000
VolksWagen 2 900	combi	VL	2 900

- *Faire :*

CONSTRUCTEUR	MODELE	PTAC
Renault	Estafette	2500
Iveco	Eurostar 440	19000
Berliet	GDM 1934	15000
VolksWagen 2 900	combi	2 900

Implémentation d'une liste

- Il est souvent utile de pouvoir modéliser une liste d'objets
  - Ex : une liste d'employés
- 4 types d'implémentation:
  - Sac d'objet: les éléments n'ont pas d'ordre précis
  - Liste chainée : les éléments se suivent (insertion simplifiée)
  - Liste doublement chainée : les éléments se suivent et se précèdent
  - Implémentation « tableau »

SAC d'objet

- Il s'agit d'une simple table avec une clef pour identifier la liste

ID LISTE	EMPLOYE ID
1	1
1	2
1	3
2	4

EMPLOYE_ID	NOM
1	X
2	XX
3	YY
4	ZZ

Liste chainée

- Il s'agit d'une simple table avec une clef pour identifier la liste

ID NOEUD	NEXT_ID	EMPLOYE ID
1	2	1
2	3	2
3	NULL	3
4	NULL	4

EMPLOYE_ID	NOM
1	X
2	XX
3	YY
4	ZZ

Liste doublement chainée

- Il s'agit d'une simple table avec une clef pour identifier la liste

ID NOEUD	NEXT_ID	PREVIOUS_ID	EMPLOYE ID
1	2	NULL	1
2	3	1	2
3	NULL	2	3
4	NULL	NULL	4

EMPLOYE_ID	NOM
1	X
2	XX
3	YY
4	ZZ

Liste Tableau

- Il s'agit d'une simple table avec une clef pour identifier la liste

ID LISTE	ORDER	EMPLOYE ID
1	1	1
1	2	2
1	3	3
2	1	4

EMPLOYE_ID	NOM
1	X
2	XX
3	YY
4	ZZ

Implémentation d'un arbre

- Il est souvent utile de pouvoir modéliser un arbre d'objets
  - Ex : la généalogie parents enfants
- 2 types d'implémentation « classique »:
  - Mode Parent Enfant
  - Mode Enfant Parent

Arbre Parent Enfant

- Il s'agit d'une simple table avec une clef pour identifier la liste

ID ARBRE	EMPLOYE_ID	CHILD_ID
1	1	1
1	2	NULL
1	3	NULL
2	4	NULL

PARENT_ID	CHILD_ID
1	2
1	3

EMPLOYE_ID	NOM
1	X
2	XX
3	YY
4	ZZ

Arbre Enfant Parent

- Il s'agit d'une simple table avec une clef pour identifier la liste

ID TREE	ID NOEUD	EMPLOYE_ID	ID PARENT
1	1	1	NULL
1	2	2	1
1	3	3	1
2	4	4	NULL

EMPLOYE_ID	NOM
1	X
2	XX
3	YY
4	ZZ

INDEX

- Qu'est ce qu'un index?
- Comment placer des indexs

Rôle d'un index

Lorsque les tables devinnent très volumineuses, le serveur met de plus en plus de temps à retrouver les donénes que les clients lui demandent et cette suractivité se traduit par des requêtes de plus en plus longues.

Un index est un structure de données liée à une table dont le rôle est comparable à celui d'un index dans un livre

Colonnes pouvant bénéficier d'un index

Toutes les colonnes d'une table peuvent en théorie bénéficier d'un dindex,.

En pratique les candidats eventuels sont les colonnes utilisées pour les jointures, les filtres et le tris.

Généralement MySQL ne peut dans son optimiseur utilisé qu'un index par table pour une requète.

La création d'index se fait ainsi:

- CREATE INDEX <nom de l'index> ON <table> (<colonne>)
- ou ALTER TABLE <table> ADD INDEX (<colonne>)

Colonnes pouvant bénéficier d'un index

Lors de la création d'index sur un BLOB/TEXT, il est possible de n'indexer que les N premier caractères du champ.

Cela s'appel un index sur un préfixe de colonne.

CREATE INDEX idx ON t(col(50)).

Enfin le but d'un index est d'être utilisé lors des requêtes ce que l'on peut vérifier via la requete EXPLAIN

EXPLAIN

On peut se faire une idée rapide des requêtes en cours d'exécution avec SHOW PROCESSLIST. Par ailleurs, dans le fichier de config /etc/mysql/my.cnf, on pourra activer le slow query log afin de voir quelles requêtes sont lentes à exécuter, et consomme donc des ressources.

Enfin, afin de comprendre comment le moteur SQL exécute la requête, on utilisera la commande EXPLAIN.

- sql > EXPLAIN SELECT * FROM categoriesG;

EXPLAIN

- Utiliser EXPLAIN est aussi simple que de l'ajouter au début d'une requête SELECT, mais cette dernière n'execute pas la requète. L'idée est de voir le plan de l'optimiseur.

```
EXPLAIN SELECT * FROM categoriesG;

*********************** 1. row ***********************
           id: 1
  select_type: SIMPLE
        table: categories
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 4
        Extra:
1 row in set (0.00 sec)
```

EXPLAIN

Nom de colonne	Description
Id	Désigne l'identifiant de la colonne utilisé pour le parcours séquentiel de la table. Dans le cas ou cet identifiant résulterait de l'assocaition de deux ou plusieurs colonnes, la valeur affiché est NULL.
select_type	De quel type de SELECT provient la table.
table	Nom d'alias de la table. Les tables temporaires matérialisées pour les sous-requêtes portent le nom <sous-requête>.
type	Comment les lignes sont trouvées dans la table (type de jointure).
possible_keys	clés dans la table qui pourraient être utilisées pour trouver des lignes dans la table
Key	Nom de la clé utilisée pour extraire les lignes. NULL si aucune clé n'a été utilisée.
key_len	Nombre d'octets de la clé utilisée
ref	La référence utilisée comme valeur de clé.
rows	Une estimation du nombre de lignes que nous allons trouver dans la table pour chaque recherche de clé.
Extra	Information additionnelles

select_type

Valeur	Description
DEPENDENT SUBQUERY	Premier SELECT d'une sosu requete dépendant d'une requete externe
DEPENDENT UNION	Il s'agit d'un select dans un union dépédant d'une requete externe
DERIVATED	SELECT dans la sous requete d'une clause FROM
MATERIALIZED	Sous reuqete que l'optimiseur a jugé utile de matrialiser ous la forme d'une table temporaire. Implique que la variable système comporte l'option materialized=ON
PRIMARY	Il s'agit du premier SELECT hors de toutes vue ou sous requete

select_type

SIMPLE	Pas d'union avec un autre SELECT ni de sous requete
SUBQUERY	Il s'agit du premier select d'un sous requete
UNCACHEABLE SUBQUERY	Sous requete dont le résultat doit être stocké en cache et ne peux être calculé que ligne à lgine
UNCACHEABLE UNION	Deuxième SELECT (ou plus) d'une sosu requete dont le résultat ne peut être stocké en cache
UNION	Il s'agit d'un select dans un union
UNION RESULT	Résultat d'un union

Colonne Type

Valeur	Description
ALL	Une analyse complète de la table est effectuée pour la table (toutes les lignes sont lues). C'est mauvais si la table est grande et que la table est jointe à une table précédente. Cela se produit lorsque l'optimiseur n'a trouvé aucun index utilisable pour accéder aux lignes.
const	Il n'y a qu'une seule ligne correspondante dans la table. La ligne est lue avant la phase d'optimisation et toutes les colonnes de la table sont traitées comme des constantes.
eq_ref	Un index unique est utilisé pour rechercher les lignes. C'est le meilleur plan possible pour trouver la rangée. fulltext Un index de texte intégral est utilisé pour accéder aux lignes.
index_merge	Un accès 'range' est effectué pour plusieurs index et les lignes trouvées sont fusionnées. La colonne clé indique quelles clés sont utilisées.
index_subquery	Identique à ref, mais utilisé pour les sous-requêtes transformées en recherches de clé.
index	Analyse complète de l'index utilisé.
range	La table est accessible avec une clé sur une ou plusieurs plages de valeurs.
ref_or_null	Identique à 'ref' mais en plus, une autre recherche de la valeur 'null' est effectuée si la première valeur n'a pas été trouvée. Cela se produit généralement avec des sous-requêtes.
ref	Toutes les lignes de la table mises en correspondance sont lues en s’appuyant sur l’index.

Colonne Extra

Distinct	Si une optimisation distincte (supprimer les doublons) a été utilisée. Ceci est marqué uniquement pour la dernière table du SELECT.
Full scan on NULL key	La table fait partie de la sous-requête et si la valeur utilisée pour la faire correspondre à la sous-requête sera NULL.
Using index	Seul l'index est utilisé pour extraire les informations nécessaires de la table. Il n'est pas nécessaire d'effectuer une recherche supplémentaire pour récupérer l'enregistrement réel
Using temporary	Une table temporaire est créée pour contenir le résultat. Cela se produit généralement si vous utilisez GROUP BY, DISTINCT ou ORDER BY.
Using where	Une expression WHERE (en plus de la recherche de clé possible) est utilisée pour vérifier si la ligne doit être acceptée. Si vous n'avez pas 'Utiliser où' avec un type de jointure de ALL, vous faites probablement quelque chose de mal!

INDEX

Lorsque vous disposez de plusieurs index pouvant être utilisés pour une requête, MySQL essaie d'identifier l'index le plus efficace pour la requête. Pour ce faire, il analyse des statistiques sur la distribution des données au sein de chaque index.

INDEX

```
mysql> EXPLAIN SELECT artist_id, name, country_id
    -> FROM artist
    -> WHERE type='Band'
    -> AND founded = 1980\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: artist
         type: ref
possible_keys: founded,founded_2,type
          key: founded
      key_len: 2
          ref: const
         rows: 1216
        Extra: Using where
```

SHOW INDEXES

- Dans cet exemple, MySQL doit choisir entre les index possibles énumérés dans possible_keys. L'optimiseur choisit un indice basé sur le coût estimé pour effectuer le moins de travail possible, et non ce qu'un humain considère comme le bon ordre. Nous pouvons utiliser la cardinalité d'index pour confirmer la raison probable de cette décision.

```
mysql> SHOW INDEXES FROM artist\G
...
*************************** 3. row ***************************
       Table: artist
  Non_unique: 1
    Key_name: founded
Seq_in_index: 1
 Column_name: founded
   Collation: A
Cardinality: 846
...
*************************** 5. row ***************************
       Table: artist
  Non_unique: 1
    Key_name: type
Seq_in_index: 1
 Column_name: type
   Collation: A
Cardinality: 10
...
```

SHOW INDEXES

Ces informations montrent que la colonne fondée a une cardinalité plus élevée, c'est-à-dire un nombre plus élevé de valeurs uniques et, par conséquent, une probabilité plus élevée de trouver les enregistrements nécessaires avec moins de lectures à partir de l'index. Les informations statistiques ne sont qu'une estimation. L'analyse des données nous a appris qu'il n'y a que quatre valeurs uniques pour le type, mais les statistiques indiquent le contraire.

Une discussion sur la cardinalité n'est pas complète sans discuter de la sélectivité. Connaître le nombre de valeurs uniques dans un index n’est pas aussi utile que de comparer ce nombre au nombre total de lignes de l’index. La sélectivité est définie comme le nombre de valeurs distinctes par rapport au nombre d'enregistrements dans la table. La sélectivité idéale est une valeur de 1. Il s'agit d'une valeur unique non nulle pour chaque valeur.

SHOW INDEXES

Avoir un index avec une bonne sélectivité signifie que moins de lignes ont la même valeur. Une mauvaise sélectivité survient lorsqu'il y a peu de valeurs distinctes, par exemple le sexe ou un statut. Cette détermination peut non seulement être utilisée pour déterminer quand un index peut ne pas être efficace, mais également pour ordonner les colonnes dans un index multi-colonnes lorsque toutes les colonnes sont utilisées pour vos requêtes.

SHOW INDEXES

```
mysql> EXPLAIN SELECT artist_id, name, country_id
    -> FROM artist
    -> WHERE founded BETWEEN 1980 AND 1989 AND type='Band'\G
*************************** 1. row ***************************
...
possible_keys: founded,founded_2,type
          key: founded
      key_len: 2
          ref: NULL
         rows: 18690
        Extra: Using where

mysql> EXPLAIN SELECT artist_id, name, country_id
    -> FROM artist
    -> WHERE founded BETWEEN 1980 AND 1989 AND type='Combination'\G
*************************** 1. row ***************************
..
possible_keys: founded,founded_2,type
          key: type
      key_len: 1
          ref: const
         rows: 19414
        Extra: Using where
```

SHOW PROFILE

Les instructions SHOW PROFILE et SHOW PROFILES affichent des informations de profilage indiquant l'utilisation des ressources pour les instructions exécutées au cours de la session en cours.

Pour contrôler le profilage, utilisez la variable de session de profilage, dont la valeur par défaut est 0 (OFF). Activer le profilage en définissant le profilage sur 1 ou sur ON:

- SET profiling = 1;

SHOW PROFILES

```
mysql> SELECT @@profiling;
+-------------+
| @@profiling |
+-------------+
|           0 |
+-------------+
1 row in set (0.00 sec)

mysql> SET profiling = 1;
Query OK, 0 rows affected (0.00 sec)

mysql> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> CREATE TABLE T1 (id INT);
Query OK, 0 rows affected (0.01 sec)

mysql> SHOW PROFILES;
+----------+----------+--------------------------+
| Query_ID | Duration | Query                    |
+----------+----------+--------------------------+
|        0 | 0.000088 | SET PROFILING = 1        |
|        1 | 0.000136 | DROP TABLE IF EXISTS t1  |
|        2 | 0.011947 | CREATE TABLE t1 (id INT) |
+----------+----------+--------------------------+
3 rows in set (0.00 sec)
```

SHOW PROFILES

```
mysql> SHOW PROFILE FOR QUERY 1;
+----------------------+----------+
| Status               | Duration |
+----------------------+----------+
| query end            | 0.000107 |
| freeing items        | 0.000008 |
| logging slow query | 0.000015 |
| cleaning up          | 0.000006 |
+----------------------+----------+
4 rows in set (0.00 sec)
```

Optimisation applicative

- Mise en place d'un connexion pool
- Mise en place d'un cache applicatif

Eléments de base

Il faut un driver (JDBC, ODBC ou PHP)

Une connexion est une relation à la base de données qu'il faut ouvrir… et fermer.

Un « statement » modélise une requête

Un prepared statement modélise une requête dans le cache de requête.

Un resultSet est un résultat de requête.

Un cache de second niveau est un cache applicatif

Une connexion pool est une boite de connexion non fermée

Eléments techniques

Un cache de second niveau : ehCache

Une connexion pool : apache common dbcp

Faire du JDBC simple

```
public static Connection initConnection() throws SQLException, ClassNotFoundException
{
Class.forName("com.mysql.jdbc.Driver");
  return  DriverManager.getConnection("jdbc:mysql://localhost:3306/test","root","");
}
```

```
public static Long findByPrimaryKeyWithoutQueryPlan(Connection c,Long pk) throws
SQLException  {
ResultSet resultSet=null;
Statement statement=null;
try{Long result=0L;
 statement=c.createStatement();
resultSet=statement.executeQuery("select A from B where C=1");
while (resultSet.next())
{
result=resultSet.getLong(1);
}
return result;
}finally
{
if (resultSet!=null) resultSet.close();
if (statement!=null) statement.close();
}
}
```

Faire une connexion Pool

```
public static DataSource initConnectionPool()
{
BasicDataSource bs = new BasicDataSource();
bs.setDriverClassName("com.mysql.jdbc.Driver");
bs.setUsername("root");
bs.setUrl("jdbc:mysql://localhost:3306/test" );
bs.setLogAbandoned(true);
bs.setTestOnBorrow(true);
bs.setTestOnReturn(true);
bs.setTestWhileIdle(true);
bs.setMaxActive(30);
bs.setValidationQuery("select 1");
bs.setMaxIdle(5);
bs.setInitialSize(3);
bs.setPoolPreparedStatements(true);
return bs;
}
```

Faire une requête préparée

```
public static Long findByPrimaryKeyQueryPlan(Connection c,Long pk) throws SQLException  {
ResultSet resultSet=null;
PreparedStatement statement=null;
try{Long result=0L;
 statement=c.prepareStatement("select A from B where C=?");
statement.setLong(1, pk);
 resultSet=statement.executeQuery();
while (resultSet.next())
{
result=resultSet.getLong(1);
}
return result;
}finally
{
if (resultSet!=null) resultSet.close();
if (statement!=null) statement.close();
}
}
```

Faire un cache de second niveau

```
public  static Long findByPrimaryKeyThroughCache(Connection c,Long pk) throws
SQLException {
Cache cache = CacheManager.getInstance().getCache("cacheName");
Long result=null;
Element el=cache.get(pk);
if (el!=null)
{
result= (Long)el.getValue();
}else
{
result=findByPrimaryKeyQueryPlan(c,pk);
cache.put(new Element(pk,result));
}
return result;
}
```

Performance 1000 connexions/10000 requêtes

Sélectionner 1 ligne avec 1 clef primaire

Performance	Sans Pool, Sans requête préparée, sans cache	Avec Pool, Sans requête préparée, sans cache	Avec Pool, Avec requête préparée, sans cache	Avec Pool, Avec requête préparée, Avec cache
Connexion	6.721 ms /Connexion	0.611 ms /Connexion	0.601 ms /Connexion	0.621 ms /Connexion
Requête	0.5363 ms /requêtes	0.489 ms /requêtes	0.3483 ms /requêtes	0.011 ms /requêtes

Partitionement

- Mettre en place les partitions horizontales

Definition

- Le standard SQL ne fournit pas de fonctionnalités sur le stockage physique des données et pour cause, le langage SQL est conçu pour fonctionner indépendamment des structures de données et des supports de stockage pour les tables, les lignes et les colonnes.
- La plupart des systèmes de gestion de bases de données avancés ont évolué pour contrôler la localisation physique des données (filesystems, hardware…).

Avantages

- Le partitionnement inclut également les avantages ci-dessous :
  - La possibilité de stocker précisément les données d'une table sur un disque ou une partition filesystem.
  - Une gestion plus simple de la suppression des données obsolètes d'une table en supprimant simplement une ou plusieurs partitions sans avoir recours à la commande DELETE.
  - Des requêtes optimisées en ne balayant que les partitions rentrant dans les conditions des clauses WHERE.
  - Une maintenance améliorée et plus efficace (statistiques, réorganisations, check etc…) car cette maintenance peut ne s'appliquer que sur une partition d'une table.

Type de partition

Cette section évoque les types de partitionnement disponibles avec MySQL 4 types de partitionnement sont proposés :

- Partitionnement RANGE (partitionnement par intervalles) : applique les lignes dans les partitions en fonction des valeurs d'une colonne correspondant à un intervalle.
- Partitionnement LIST (partitionnement par liste) : similaire au partitionnement par intervalles, sauf que les partitions ne sont plus créées sur des intervalles de valeurs mais des valeurs discrètes (liste de valeurs).
- Partitionnement HASH (partitionnement par hâchage) : la partition correspond à une valeur retournée par une expression utilisateur appliquée sur les valeurs des colonnes des lignes insérées dans la table. Cette fonction peut être n'importe quelle expression valide MySQL qui retourne une valeur entière non négative.
- Partitionnement KEY (partitionnement par clé) : similaire au partitionnement par hachage, sauf qu'une ou plusieurs colonnes à évaluer sont fournies.

- Une utilisation courante du partitionnement dans les bases données consiste à ségréguer les données par date.

Partitionnement RANGE

- Une table qui est partitionnée par intervalles est partitionnée de telle manière que chaque partition contient les lignes correspondant à un intervalle donné.

- Les intervalles devraient être contigus mais ne doivent surtout pas se chevaucher. Ces intervalles sont données par l'opérateur VALUES LESS THAN.
- Si on décide de partitionner cette table en 4 partitions avec la clause PARTITION BY RANGE appliquée sur la colonne store_id

```
CREATE TABLE employees (
                id INT NOT NULL,
                fname VARCHAR(30),
                lname VARCHAR(30),
                hired DATE NOT NULL DEFAULT '1970-01-01',
                separated DATE NOT NULL DEFAULT '9999-12-31',
                job_code CHAR(1),
                store_id INT NOT NULL
)

PARTITION BY RANGE (store_id) (
                PARTITION p0 VALUES LESS THAN (6),
                PARTITION p1 VALUES LESS THAN (11),
                PARTITION p2 VALUES LESS THAN (16),
                PARTITION p3 VALUES LESS THAN (21)
);
```

Partitionnement RANGE

- Que se passe-t-il si un employé est inséré pour un identifiant store_id non défini dans les partitions (plus grand que 20 dans notre cas) ? Une erreur est alors générée, car dans le schéma de partitionnement, aucune règle n'est donnée au serveur pour la valeur 21. Cette erreur peut être évitée en utilisant la méthode « catchall » de la clause VALUES LESS THAN de la commande CREATE TABLE afin de gérer les valeurs plus hautes que les valeurs explicitement données pour les partitions.

```
REATE TABLE employees (
               id INT NOT NULL,
               fname VARCHAR(30),
               lname VARCHAR(30),
               hired DATE NOT NULL DEFAULT '1970-01-01',
               separated DATE NOT NULL DEFAULT '9999-12-31',
               job_code CHAR(1),
               store_id INT NOT NULL
)

PARTITION BY RANGE (store_id) (
      PARTITION p0 VALUES LESS THAN (6),
      PARTITION p1 VALUES LESS THAN (11),
      PARTITION p2 VALUES LESS THAN (16),
      PARTITION p3 VALUES LESS THAN MAXVALUE
);
```

Partitionnement LIST

- Le partitionnement par valeurs discrètes est très similaire au partitionnement par intervalles sur bien des points. Comme dans le partitionnement par intervalles, chaque partition est explicitement définie. Dans le partitionnement par valeurs discrètes, chaque partition est définie sur une liste figée de valeurs au lieu d'un intervalle.

```
CREATE TABLE employees (
            id INT NOT NULL,
            fname VARCHAR(30),
            lname VARCHAR(30),
            hired DATE NOT NULL DEFAULT '1970-01-01',
            separated DATE NOT NULL DEFAULT '9999-12-31',
            job_code CHAR(1),
            store_id INT NOT NULL
)
PARTITION BY LIST(ASCII( UCASE(job_code) )) (
            PARTITION management VALUES IN (68, 77, 79, 80),
            PARTITION sales VALUES IN (66, 76, 83),
            PARTITION technical VALUES IN (65, 69, 71, 73, 84),
            PARTITION clerical VALUES IN (75, 78, 89),
            PARTITION support VALUES IN (67, 70, 74, 82, 86),
            PARTITION unassigned VALUES IN (NULL, 0, 32)
);
```

Partitionnement HASH

- Le partitionnement HASH est utilisé primairement pour assurer une distribution des données sur un nombre déterminé de partitions. Avec le partitionnement par intervalles ou par valeurs discrètes, l'utilisateur doit spécifier explicitement dans quelle partition une valeur d'une colonne donnée doit être stockée; avec le partitionnement par hachage (HASH partitioning), MySQL se charge de déterminer dans quelle partition stocker la ligne, il suffit de simplement spécifier la colonne ou l'expression sur la colonne.

- Pour partitionner une table en mode HASH, il faut ajouter à la commande CREATE TABLE la clause PARTITION BY HASH

```
CREATE TABLE employees (
   id INT NOT NULL,
   fname VARCHAR(30),
   lname VARCHAR(30),
   hired DATE NOT NULL DEFAULT '1970-01-01',
   separated DATE NOT NULL DEFAULT '9999-12-31',
   job_code CHAR(1),
   store_id INT
)
PARTITION BY HASH(YEAR(hired))
PARTITIONS 4;
```

Partitionnement KEY

- Le partitionnement par clé est similaire au partitionnement par hachage, sauf que dans le partitionnement par hachage une expression utilisateur est utilisée (MOD, CEIL etc…), la fonction de hachage pour le partitionnement par clé est en revanche quant à elle fournie par le serveur MySQL.
- La syntaxe de partitionnement par clé est : CREATE TABLE ... PARTITION BY KEY.

- KEY accepte une liste d'une ou plusieurs colonnes. Lorsque les colonnes ne sont pas spécifiées dans la clé de partitionnement, la clé primaire ou l'index unique de la table est utilisée :

```
CREATE TABLE k1 (
    id INT NOT NULL,
    name VARCHAR(20),
    UNIQUE KEY (id)
)
PARTITION BY KEY()
PARTITIONS 2;
```

Partition

- Pour sélectionner des données uniquement dans une partition spécifique, il suffit de surcharger l'instruction SQL par le mot clé PARTITION en désignant le nom des partitions à accéder, après la clause FROM du nom de table.

```
SELECT * FROM <table_name> PARTITION (<partition_name>);
```

Optimizer

- MySQL fournit un contrôle de l'optimiseur via des variables système qui affectent la manière dont les plans de requête sont évalués, des optimisations commutables, des indicateurs d'optimisation et d'index, ainsi que le modèle de coût de l'optimiseur.

Contrôle de l'évaluation du plan de requête

L'optimiseur de requêtes a pour tâche de trouver un plan optimal pour l'exécution d'une requête SQL. Parce que la différence de performance entre les «bons» et les «mauvais» plans peut être de plusieurs ordres de grandeur (secondes ou heures, voire plusieurs jours), la plupart des optimiseurs de requêtes, y compris celui de MySQL, effectuent une recherche plus ou moins exhaustive plan parmi tous les plans possibles d'évaluation des requêtes. Pour les requêtes de jointure, le nombre de plans possibles étudiés par l'optimiseur MySQL augmente de manière exponentielle avec le nombre de tables référencées dans une requête.

Contrôle de l'évaluation du plan de requête

Le comportement de l'optimiseur en ce qui concerne le nombre de plans qu'il évalue peut être contrôlé à l'aide de deux variables système:

- La variable optimizer_prune_level indique à l'optimiseur d'ignorer certains plans en fonction d'estimations du nombre de lignes consultées pour chaque table. Cette option est activée (optimizer_prune_level = 1) par défaut. Toutefois, si vous pensez que l'optimiseur a raté un meilleur plan de requête, vous pouvez désactiver cette option (optimizer_prune_level = 0), avec le risque que la compilation de la requête prenne beaucoup plus de temps.
- La variable optimizer_search_depth indique jusqu'où doit aller l'optimiseur dans l'exploration de chaque plan incomplet pour déterminer s'il doit être développé davantage. Des valeurs plus petites de optimizer_search_depth peuvent entraîner des ordres de grandeur de temps de compilation des requêtes plus courts.

Optimizer Switch

- La variable système optimizer_switch permet de contrôler le comportement de l'optimiseur. Sa valeur est un ensemble d'indicateurs dont chacun a la valeur on ou off pour indiquer si le comportement de l'optimiseur correspondant est activé ou non. Cette variable a des valeurs globales et de session et peut être modifiée à l'exécution. La valeur globale par défaut peut être définie au démarrage du serveur.

Optimizer Switch

```
mysql> SELECT @@optimizer_switch\G
*************************** 1. row ***************************
@@optimizer_switch: index_merge=on,index_merge_union=on,
                    index_merge_sort_union=on,
                    index_merge_intersection=on,
                    engine_condition_pushdown=on,
                    index_condition_pushdown=on,
                    mrr=on,mrr_cost_based=on,
                    block_nested_loop=on,batched_key_access=off,
                    materialization=on,semijoin=on,loosescan=on,
                    firstmatch=on,duplicateweedout=on,
                    subquery_materialization_cost_based=on,
                    use_index_extensions=on,
                    condition_fanout_filter=on,derived_merge=on,
                    use_invisible_indexes=off,skip_scan=on
```

Batched Key Access

L’idée principale de BKA est d’accumuler plusieurs clés dans un tampon et d’accéder ensuite à la table jointe en modifiant éventuellement l’ordre des recherches pour optimiser la séquence des recherches de disque.

Les premières expériences sur BKA ont montré qu'il pouvait améliorer les performances des requêtes de jointure jusqu'à 2 à 5 fois en fonction de la taille du tampon de jointure utilisé. Il a également amélioré l'exécution des jointures sur un cluster NDB en réduisant considérablement le nombre d'allers-retours entre les nœuds de stockage MySQL Server et Cluster.

block_nested_loop

C’est une variante de la jointure de boucle imbriquée simple utilisée pour joindre deux relations R et S (les opérandes de jointure "externe" et "interne", respectivement).

Supposons | R | <| S |.

Dans une jointure de boucle imbriquée traditionnelle, S sera analysé une fois pour chaque tuple de R.

S'il y a beaucoup de tuples R qualifiants, et en particulier s'il n'y en a pas. index applicable pour la clé de jointure sur S, cette opération coûtera très cher.

L'algorithme de jointure de boucle imbriquée par bloc améliore la jointure simple de boucle imbriquée en scannant seulement S pour chaque groupe de R.

index_condition_pushdown

La condition d'indexation (ICP) est une optimisation pour le cas où MySQL récupère les lignes d'une table à l'aide d'un index. Sans ICP, le moteur de stockage parcourt l'index pour localiser les lignes de la table de base et les renvoie au serveur MySQL, qui évalue la condition WHERE pour les lignes. Si ICP est activé et si certaines parties de la condition WHERE peuvent être évaluées en utilisant uniquement les colonnes de l'index, le serveur MySQL envoie cette partie de la condition WHERE au moteur de stockage. Le moteur de stockage évalue ensuite la condition d'indexation poussée à l'aide de l'entrée d'index. Seule la ligne lue dans la table est satisfaite. ICP peut réduire le nombre de fois que le moteur de stockage doit accéder à la table de base et le nombre de fois que le serveur MySQL doit accéder au moteur de stockage.

Range Optimization

La lecture de lignes à l'aide d'une analyse d'intervalle sur un index secondaire peut entraîner de nombreux accès au disque aléatoires à la table de base lorsque la table est volumineuse et n'est pas stockée dans le cache du moteur de stockage. Grâce à l'optimisation MRR (Disk-Sweep Multi-Range Read), MySQL essaie de réduire le nombre d'accès au disque aléatoires pour les analyses de plages en analysant d'abord l'index uniquement et en collectant les clés pour les lignes correspondantes. Ensuite, les clés sont triées et enfin, les lignes sont extraites de la table de base en utilisant l'ordre de la clé primaire. La motivation de la fonction MRR à balayage de disque est de réduire le nombre d'accès aléatoires au disque et d'effectuer une analyse plus séquentielle des données de la table de base.

L'optimisation de la lecture multi-plages offre les avantages suivants:

- MRR permet d'accéder aux lignes de données de manière séquentielle plutôt que dans un ordre aléatoire, en fonction des nuplets d'index. Le serveur obtient un ensemble de tuples d'index qui répondent aux conditions de la requête, les trie selon l'ordre des ID de lignes de données et utilise les n-uplets triés pour extraire les lignes de données dans l'ordre. Cela rend l'accès aux données plus efficace et moins coûteux.

- MRR permet le traitement par lots des demandes d'accès aux clés pour les opérations nécessitant un accès aux lignes de données via des nuplets d'index, tels que les analyses d'index de plage et les équi-jointes utilisant un index pour l'attribut de jointure. MRR effectue une itération sur une séquence de plages d'index pour obtenir des nuplets d'index qualifiants. Au fur et à mesure de l'accumulation de ces résultats, ils sont utilisés pour accéder aux lignes de données correspondantes. Il n'est pas nécessaire d'acquérir tous les tuples d'index avant de commencer à lire les lignes de données.

Optimizer Hints

Une autre façon de contrôler l'optimiseur consiste à utiliser des indicateurs d'optimiseur, qui peuvent être spécifiés dans des instructions individuelles. Etant donné que les indicateurs d'optimisation s'appliquent à chaque instruction, ils permettent de contrôler plus précisément les plans d'exécution des instructions que ce qui peut être obtenu avec optimizer_switch. Par exemple, vous pouvez activer une optimisation pour une table dans une instruction et désactiver l'optimisation pour une autre table. Les indications contenues dans une instruction ont priorité sur les indicateurs optimizer_switch.

Optimizer Hints

```
SELECT /*+ NO_RANGE_OPTIMIZATION(t3 PRIMARY, f2_idx) */ f1
  FROM t3 WHERE f1 > 30 AND f1 < 33;
SELECT /*+ BKA(t1) NO_BKA(t2) */ * FROM t1 INNER JOIN t2 WHERE
...;
SELECT /*+ NO_ICP(t1, t2) */ * FROM t1 INNER JOIN t2 WHERE ...;
SELECT /*+ SEMIJOIN(FIRSTMATCH, LOOSESCAN) */ * FROM t1 ...;
EXPLAIN SELECT /*+ NO_ICP(t1) */ * FROM t1 WHERE ...;
SELECT /*+ MERGE(dt) */ * FROM (SELECT * FROM t1) AS dt;
INSERT /*+ SET_VAR(foreign_key_checks=OFF) */ INTO t2 VALUES(2);
```

JOIN Order Hints

Il est possible d'indiquer l'ordre de JOIN via les instruction JOIN ORDER:

JOIN_FIXED_ORDER: force l'optimiseur à joindre des tables en utilisant l'ordre dans lequel elles apparaissent dans la clause FROM. Cela revient à spécifier SELECT STRAIGHT_JOIN.

JOIN_ORDER: Demandez à l'optimiseur de joindre des tables en utilisant l'ordre spécifié. L'optimiseur peut placer des tables non nommées n'importe où dans l'ordre de jointure, y compris entre des tables spécifiées.

JOIN_PREFIX: Demandez à l'optimiseur de joindre des tables en utilisant l'ordre de table spécifié pour les premières tables du plan d'exécution de la jointure. L'optimiseur place toutes les autres tables après les tables nommées.

JOIN_SUFFIX: Demandez à l'optimiseur de joindre des tables en utilisant l'ordre de table spécifié pour les dernières tables du plan d'exécution de la jointure. L'optimiseur place toutes les autres tables avant les tables nommées.

JOIN Order Hints

- Les indicateurs contrôlent le comportement des tables de semi-jointure fusionnées avec le bloc de requête externe.
- Si les sous-requêtes subq1 et subq2 sont converties en semi-jointures, les tables t4 @ subq1 et t5 @ subq2 sont fusionnées dans le bloc de requête externe. Dans ce cas, l'indicateur spécifié dans le bloc de requête externe contrôle le comportement des tables t4 @ subq1, t5 @ subq2.

```
SELECT
/*+ JOIN_PREFIX(t2, t5@subq2, t4@subq1)
    JOIN_ORDER(t4@subq1, t3)
    JOIN_SUFFIX(t1) */
COUNT(*) FROM t1 JOIN t2 JOIN t3
           WHERE t1.f1 IN (SELECT /*+ QB_NAME(subq1) */ f1 FROM t4)
             AND t2.f1 IN (SELECT /*+ QB_NAME(subq2) */ f1 FROM t5);
```

USE INDEX

- L'indicateur USE INDEX (index_list) indique à MySQL de n'utiliser qu'un seul des index nommés pour rechercher des lignes dans la table. La syntaxe alternative IGNORE INDEX (liste_index) indique à MySQL de ne pas utiliser un ou plusieurs index particuliers. Ces astuces sont utiles si EXPLAIN indique que MySQL utilise le mauvais index dans la liste des index possibles.

```
SELECT * FROM table1 USE INDEX (col1_index,col2_index)
  WHERE col1=1 AND col2=2 AND col3=3;

SELECT * FROM table1 IGNORE INDEX (col3_index)
  WHERE col1=1 AND col2=2 AND col3=3;
```

Optimisation diverse

HIGH_PRIORITY

HIGH _PRIORITY donne à SELECT une priorité plus élevée qu'une instruction qui met à jour une table. Vous ne devriez l'utiliser que pour les requêtes très rapides et qui doivent être effectuées en une fois. Une requête SELECT HIGH_PRIORITY émise alors que la table est verrouillée en lecture, même s'il existe une instruction de mise à jour en attente de libération de la table. Cela concerne uniquement les moteurs de stockage qui utilisent uniquement le verrouillage au niveau de la table (tel que MyISAM, MEMORY et MERGE).

STRAIGHT_JOIN

STRAIGHT_JOIN oblige l'optimiseur à joindre les tables dans l'ordre dans lequel elles sont répertoriées dans la clause FROM. Vous pouvez l'utiliser pour accélérer une requête si l'optimiseur joint les tables dans un ordre non optimal. STRAIGHT_JOIN peut également être utilisé dans la liste table_references.

SQL_BIG_RESULT/ SQL_SMALL_RESULT

SQL_BIG_RESULT ou SQL_SMALL_RESULT peuvent être utilisés avec GROUP BY ou DISTINCT pour indiquer à l'optimiseur que le jeu de résultats comporte plusieurs lignes ou est petit, respectivement. Pour SQL_BIG_RESULT, MySQL utilise directement les tables temporaires basées sur disque si elles sont créées et préfère effectuer le tri en utilisant une table temporaire avec une clé sur les éléments GROUP BY. Pour SQL_SMALL_RESULT, MySQL utilise des tables temporaires en mémoire pour stocker la table résultante au lieu du tri.

Outils d'optimisation

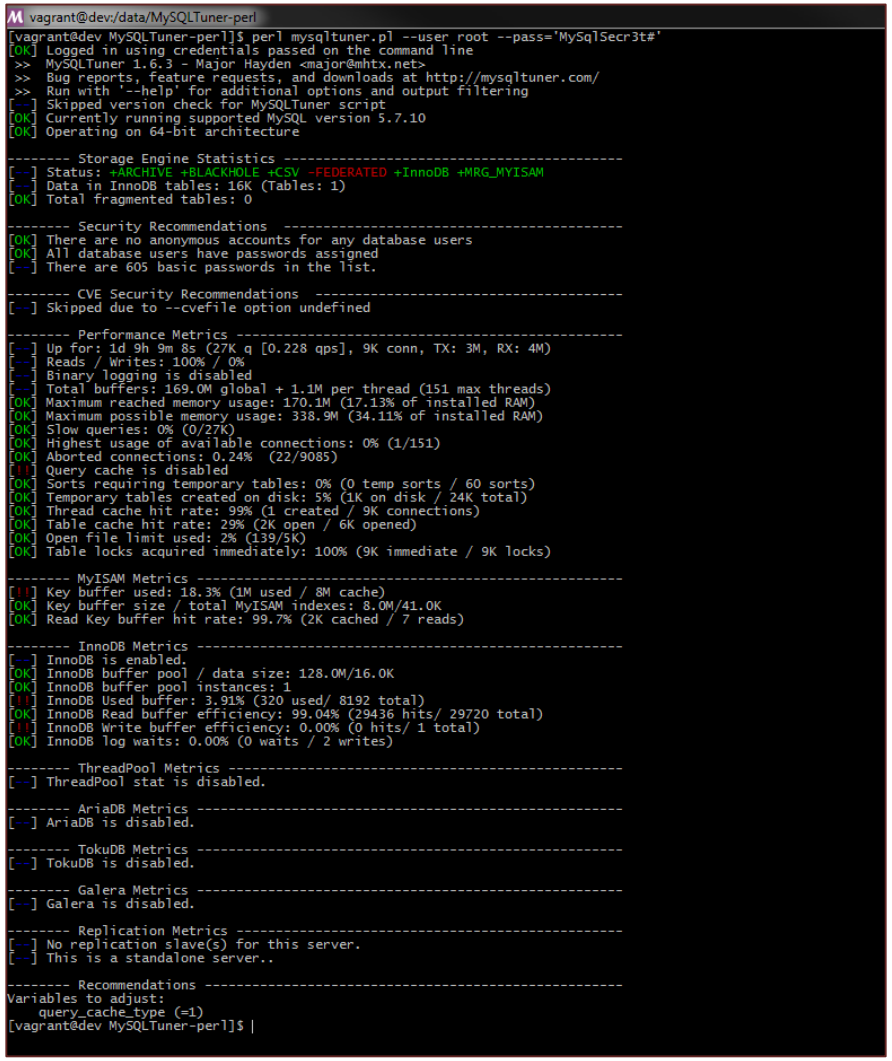
- Présentation de MySQL Tuner et validation d'une installation MySQL

MySQL-Tuner

MySQLTuner est un script écrit en Perl qui vous permet de passer rapidement en revue une installation MySQL et de procéder à des ajustements pour améliorer les performances et la stabilité. Les variables de configuration et les données d'état actuelles sont extraites et présentées dans un bref format, assorties de suggestions de performances de base.

MySQLTuner prend en charge environ 300 indicateurs pour MySQL / MariaDB / Percona Server dans cette dernière version.

MySQL-Tuner


vagrant@dev:/data/MySQLTuner-perl
[vagrant@dev MySQLTuner-perl]$ perl mysqltuner.pl --user root --pass='MySqlSecr3t#'
[OK] Logged in using credentials passed on the command line
>> MySQLTuner 1.6.3 - Major Hayden <major@mhtx.net>
>> Bug reports, feature requests, and downloads at http://mysqltuner.com/
>> Run with '--help' for additional options and output filtering
[--] Skipped version check for MySQLTuner script
[OK] Currently running supported MySQL version 5.7.10
[OK] Operating on 64-bit architecture
-------- Storage Engine Statistics ---
[--] Status: +ARCHIVE +BLACKHOLE +CSV -FEDERATED +InnoDB +MRG_MYISAM
[--] Data in InnoDB tables: 16K (Tables: 1)
[OK] Total fragmented tables: 0
-------- Security Recommendations ---
[OK] There are no anonymous accounts for any database users
[OK] All database users have passwords assigned
[--] There are 605 basic passwords in the list.
-------- CVE Security Recommendations --
[--] Skipped due to --cvefile option undefined
-------- Performance Metrics ---
[--] Up for: 1d 9h 9m 8s (27K q [0.228 qps], 9K conn, TX: 3M, RX: 4M)
[--] Reads / Writes: 100% / 0%
[--] Binary logging is disabled
[--] Total buffers: 169.0M global + 1.1M per thread (151 max threads)
[OK] Maximum reached memory usage: 170.1M (17.13% of installed RAM)
[OK] Maximum possible memory usage: 338.9M (34.11% of installed RAM)
[OK] Slow queries: 0% (0/27K)
[OK] Highest usage of available connections: 0% (1/151)
[OK] Aborted connections: 0.24% (22/9085)
[!!] Query cache is disabled
[OK] Sorts requiring temporary tables: 0% (0 temp sorts / 60 sorts)
[OK] Temporary tables created on disk: 5% (1K on disk / 24K total)
[OK] Thread cache hit rate: 99% (1 created / 9K connections)
[OK] Table cache hit rate: 29% (2K open / 6K opened)
[OK] Open file limit used: 2% (139/5K)
[OK] Table locks acquired immediately: 100% (9K immediate / 9K locks)
-------- MyISAM Metrics ---
[!!] Key buffer used: 18.3% (1M used / 8M cache)
[OK] Key buffer size / total MyISAM indexes: 8.0M/41.0K
[OK] Read Key buffer hit rate: 99.7% (2K cached / 7 reads)
-------- InnoDB Metrics ---
[--] InnoDB is enabled.
[OK] InnoDB buffer pool / data size: 128.0M/16.0K
[OK] InnoDB buffer pool instances: 1
[!!] InnoDB Used buffer: 3.91% (320 used/ 8192 total)
[OK] InnoDB Read buffer efficiency: 99.04% (29436 hits/ 29720 total)
[!!] InnoDB Write buffer efficiency: 0.00% (0 hits/ 1 total)
[OK] InnoDB log waits: 0.00% (0 waits / 2 writes)
-------- ThreadPool Metrics ---
[--] ThreadPool stat is disabled.
-------- AriaDB Metrics ---
[--] AriaDB is disabled.
-------- TokuDB Metrics ---
[--] TokuDB is disabled.
-------- Galera Metrics ---
[--] Galera is disabled.
-------- Replication Metrics --
[--] No replication slave(s) for this server.
[--] This is a standalone server..
-------- Recommendations --
Variables to adjust:
query_cache_type (=1)
[vagrant@dev MySQLTuner-perl]$

Percona Toolkit

Percona Toolkit est un ensemble d'outils de ligne de commande avancés utilisés par le personnel de Percona pour effectuer diverses tâches liées à MySQL, à MongoDB et au système, trop difficiles ou complexes à exécuter manuellement.

pt-mysql-summary : Indique comment se comporte le serveur MySQL

Résumé Optimisation

- En résumé l'optimisation est un suite de travaux autant matériels que logique. Nous essayons de les résumer ici.

Optimisation matériels

- Disposez de suffisamment de mémoire physique pour charger l'intégralité de votre fichier InnoDB - InnoDB est beaucoup plus rapide lorsque le fichier est accessible en mémoire plutôt que sur disque.
- Évitez à tout prix d'échanger des données - L'échange signifie lire à partir d'un disque, c'est lent.
- Utilisez une RAM sauvegardée sur batterie.
- Utilisez un RAID avancé - de préférence RAID 10 ou supérieur.
- Évitez RAID5 - la somme de contrôle nécessaire pour assurer l'intégrité est coûteuse.
- Séparez votre système d'exploitation de vos partitions de données, pas uniquement logiquement, mais les écritures et les lectures coûteuses sur le système d'exploitation auront un impact sur les performances de votre base de données.
- Placez votre espace temporaire mysql et vos journaux de réplication sur une partition distincte de vos données - les écritures en arrière-plan auront une incidence sur votre base de données lorsqu'elle va écrire / lire à partir du disque.
- Plus de disques signifie plus de vitesse.
- Les disques plus rapides sont meilleurs.
- Utilisez SAS (ex SCSI) plutôt SATA.
- Les petits disques sont plus rapides que les grands, en particulier dans les configurations RAID.
- Utilisez des contrôleurs RAID avec cache protégé par batterie.
- Évitez les RAID logiciels.

Optimisation matériels

- Envisagez d'utiliser des SSD (et non des unités de disque) pour votre partition de données. Ces cartes peuvent supporter des écritures supérieures à 2 Go / s pour presque toutes les quantités de données.
- Sous Linux, définissez la valeur swappiness sur 0 - aucune raison de mettre en cache des fichiers sur un serveur de base de données, il s'agit davantage d'un avantage pour le serveur Web.
- Montez le système de fichiers avec noatime et nodirtime si disponible - aucune raison de mettre à jour les heures de modification des fichiers de base de données pour l'accès.
- Utilisez le système de fichiers XFS - un système de fichiers plus rapide et plus petit que ext3 et offrant plus d'options pour la journalisation. Il a également été démontré qu'ext3 avait des problèmes de double mise en mémoire tampon avec MySQL.
- Réglez votre journal de système de fichiers XFS et vos variables de mémoire tampon - pour des performances optimales.
- Sur les systèmes Linux, utilisez le planificateur NOOP ou DEADLINE IO - les planificateurs CFQ et ANTICIPATORY se sont avérés lents par rapport aux planificateurs NOOP et DEADLINE.
- Utilisez un système d'exploitation 64 bits - plus de mémoire, adressable et utilisable par MySQL.
- Supprimez les paquets et les démons inutilisés des serveurs.
- Placez votre hôte qui utilise MySQL et votre hôte MySQL dans un fichier hosts - pas de recherches DNS.
- Ne tuer jamais un processus MySQL - vous allez corrompre votre base de données et exécuter les sauvegardes.
- Dédiez votre serveur à MySQL - les processus en arrière-plan et d'autres services peuvent voler à l'époque de la CPU.

Optimisation de Configuration

- Utilisez innodb_flush_method = O_DIRECT pour éviter un double tampon lors de l'écriture.
- Évitez les systèmes de fichiers O_DIRECT et EXT3 - vous allez sérialiser toutes vos écritures.
- Allouez suffisamment innodb_buffer_pool_size pour charger l'intégralité de votre fichier InnoDB en mémoire - moins de lectures à partir du disque.
- Ne faites pas d'innodb_log_file_size trop gros, avec des disques plus rapides et plus volumineux - le vidage plus fréquent est une bonne chose et réduit le temps de récupération en cas de collision.
- Ne mélangez pas les variables innodb_thread_concurrency et thread_concurrency - ces deux valeurs ne sont pas compatibles.
- Allouez un montant minimal pour max_connections - un trop grand nombre de connexions peut utiliser votre RAM et verrouiller votre serveur MySQL.
- Conservez thread_cache à un nombre relativement élevé, environ 16, pour éviter les lenteurs lors de l'ouverture de connexions.
- Utilisez skip-name-resol - pour supprimer les recherches DNS.
- Désactiver le cache de requete (pour MySQL < 8) et privilégier un cache applicatif
- Augmenter temp_table_size - pour empêcher les écritures sur le disque.
- Augmentez max_heap_table_size - pour empêcher les écritures sur le disque.
- Ne définissez pas votre sort_buffer_size trop haut car il est par connexion
- Surveillez les réponses key_read_requests et key_reads pour déterminer la taille de votre key_buffer - les demandes de lecture de clés doivent être supérieures à vos key_reads38.
- Définir innodb_flush_log_at_trx_commit = 0 améliorera les performances, mais si vous le réglez sur la valeur de 1, vous garantissez l'intégrité des données, vous vous assurerez également que la réplication n'est pas à la traîne.
- Ayez un environnement de test où vous pouvez tester vos configurations et les redémarrer souvent, sans affecter la production.

Optimisation de Schéma

- Gardez votre base de données la plus petite possible.
- Archiver les anciennes données - pour supprimer les retours de ligne excessifs ou les recherches sur les requêtes.
- Mettez des index sur vos données.
- Ne pas abuser des index.
- Compresser les types de données texte et blob - pour économiser de l'espace et réduire le nombre de lectures sur le disque.
- UTF 8 et UTF16 est plus lent que latin1.
- Utilisez les triggers avec parcimonie.
- Réduisez au minimum les données redondantes - ne dupliquez pas les données inutilement.
- Utilisez des tables de liaison plutôt que d'étendre les lignes.
- Faites attention à vos types de données, utilisez le plus petit possible pour vos données réelles.
- Séparez les données de BLOB/TEXT des autres données si d'autres données sont souvent utilisées pour les requêtes alors que les BLOB/TEXT ne le sont pas.
- Vérifiez et optimisez souvent les tables.
- Parfois, il est plus rapide de supprimer des index lors de l'ajout de colonnes, puis de rajouter des index.
- Utilisez différents moteurs de stockage pour des besoins différents.
- Utilisez le moteur de stockage ARCHIVE pour les tables de journalisation ou les tables d'audit - ceci est beaucoup plus efficace pour les écritures.
- Stockez les données de session dans un cache applicatif plutôt que dans MySQL
- Utilisez VARCHAR à la place de CHAR lors du stockage de chaînes de longueur variable - pour économiser de l'espace car CHAR est une longueur fixe et VARCHAR ne l'est pas (utf8 n'est pas affecté par cela).
- Modifiez les modifications de schéma progressivement: une petite modification peut avoir des effets dramatiques.
- Testez toutes les modifications de schéma dans un environnement de développement reflétant la production.

Optimisation SQL

- Utilisez le slow query log pour rechercher des requêtes lentes.
- Utilisez EXPLAIN pour déterminer si les requêtes fonctionnent correctement.
- Testez vos requêtes souvent pour voir si elles fonctionnent de manière optimale - les performances changeront avec le temps.
- Évitez les COUNT(*) sur des tables entières, il peut verrouiller la table entière.
- Utilisez GROUP BY au lieu de DISTINCT, le cas échéant.
- Utilisez des colonnes indexées dans les clauses WHERE, GROUP BY et ORDER BY.
- Conservez les index simples, ne réutilisez pas une colonne dans plusieurs index.
- Parfois, MySQL choisit le mauvais index, utilisez USE INDEX pour ce cas.
- Recherchez des problèmes liés à SQL_MODE = STRICT.
- Utilisez une LIMIT sur UNION au lieu de OR
- Utilisez INSERT ON DUPLICATE KEY ou INSERT IGNORE au lieu de UPDATE pour éviter le SELECT avant la mise à jour.
- Utilisez un champ indexé sur un ORDER BY au lieu de MAX.
- Évitez d'utiliser ORDER BY RAND ().
- LIMIT M, N peuvent réellement ralentir les requêtes, utilisez-les avec parcimonie.
- Utilisez UNION au lieu de sous-requêtes dans les clauses WHERE.
- Au redémarrage de MySQL, pensez a raffraichir les caches en executant quelques jeux de tests
- Utilisez DROP TABLE, puis CREATE TABLE au lieu de DELETE FROM pour supprimer toutes les données d'une table.
- Réduisez au minimum les données de votre requête. L'utilisation de * est excessive, la plupart du temps.
- Envisagez des connections pool au lieu de plusieurs connexions pour réduire le temps système.
- Lorsque la charge de votre serveur augmente, utilisez SHOW PROCESSLIST pour afficher les requêtes lentes / problématiques.
- Testez toutes les requêtes suspectes dans un environnement de développement dans lequel vous avez mis en miroir les données de production.