

# MySQL

# Administration

- [Introduction](#)
  - [Plan du cours](#)
- [Installer MySQL](#)
  - [Installer MySQL sous Unix / Linux](#)
- [Optimization](#)
  - [New Page](#)
- [Sécurité](#)
  - [Authentification Simple et administration des droits](#)
  - [Connection et SSL](#)
  - [Installation de SSL avec MariaDB sous Debian](#)
- [Administration MySQL](#)
  - [Exportation des données en CSV](#)
  - [Tuning MySQL](#)
  - [SQL Mode](#)
  - [Migration](#)
  - [Transaction](#)
  - [Administration d'un base de donnée](#)
  - [Administration des tables](#)

# Introduction

# Plan du cours

## Document

[www.pilou.online/files/MySQLAdministration1.pdf](http://www.pilou.online/files/MySQLAdministration1.pdf)

## 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

## PROGRAMME DE FORMATION

### 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.

### Travaux pratiques

- Installation sous Linux/Windows. Installation de MySQL Workbench.

## 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.

## Travaux pratiques

- Création et utilisation de vues. Modification de la définition d'une table.

## 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.

## Travaux pratiques

- Sécurisation complète de la post-installation.
- Création de comptes MySQL.
- Gestion des privilèges et mots de passe.

## 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.

# Travaux pratiques

- Changement du paramétrage InnoDB. Mise en oeuvre de transactions.

## 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.

# Travaux pratiques

- Activation des journaux. Utilisation des outils de chargement/déchargement. Sauvegarde et restauration avec "mysqldump".
- Utilisation du journal binaire. Sauvegarde et récupération incrémentale.

# Installer MySQL

# Installer MySQL sous Unix / Linux

## Résolution des dépendances

MySQL dépend de la bibliothèque libaio. L'initialisation du répertoire de données et les étapes de démarrage du serveur suivantes échoueront si cette bibliothèque n'est pas installée localement.

```
pilou@ubuntu: ~$ sudo apt-get install libaio1 [sudo] password for pilou: Reading package lists...
Done Building dependency tree Reading state information...
Done libaio1 is already the newest version (0.3.111-1).
libaio1 set to manually installed. 0 upgraded, 0 newly installed, 0 to remove and 68 not upgraded.
```

## Téléchargement et décompression de l'archive

Pour installer une distribution binaire de fichier tar compressée, décompressez-la à l'emplacement d'installation de votre choix.

```
pilou@ubuntu: ~$ mkdir mysql80

pilou@ubuntu: ~$ cd mysql80

pilou@ubuntu: ~/mysql80$ wget https://dev.mysql.com/get/Downloads/MySQL-8.0/mysql-8.0.13-linux-glibc2.12-x86_64.tar --2018-12-30 18:10:42--
https://dev.mysql.com/get/Downloads/MySQL-8.0/mysql-8.0.13-linux-glibc2.12-x86_64.tar
```

```
Resolving dev.mysql.com (dev.mysql.com)...  
  
137.254.60.11 Connecting to dev.mysql.com (dev.mysql.com)|137.254.60.11|:443... connected.  
HTTP request sent, awaiting response... 302 Found Location:  
https://cdn.mysql.com//Downloads/MySQL-8.0/mysql-8.0.13-linux-glibc2.12-x86_64.tar [following]  
--2018-12-30 18:10:48-- https://cdn.mysql.com//Downloads/MySQL-8.0/mysql-8.0.13-linux-  
glibc2.12-x86_64.tar Resolving cdn.mysql.com (cdn.mysql.com)... 23.210.41.222 Connecting to  
cdn.mysql.com (cdn.mysql.com)|23.210.41.222|:443... connected. HTTP request sent, awaiting  
response... 200 OK Length: 472883200 (451M) [application/x-tar] Saving to: 'mysql-8.0.13-  
linux-glibc2.12-x86_64.tar'  
  
mysql-8.0.13-linux-glib 100%[=====>] 450,98M 6,86MB/s in 72s  
  
2018-12-30 18:12:01 (6,22 MB/s) - 'mysql-8.0.13-linux-glibc2.12-x86_64.tar' saved  
[472883200/472883200]  
  
pilou@ubuntu: ~/mysql80$ ls mysql-8.0.13-linux-glibc2.12-x86_64.tar
```

L'archive contient le serveur MySQL, l'outil MySQL Router ainsi qu'un banc de test pour MySQL.

```
pilou@ubuntu: ~/mysql80$ tar xvf mysql-8.0.13-linux-glibc2.12-x86_64.tar mysql-8.0.13-linux-  
glibc2.12-x86_64.tar.xz mysql-router-8.0.13-linux-glibc2.12-x86_64.tar.xz mysql-test-8.0.13-  
linux-glibc2.12-x86_64.tar.xz  
pilou@ubuntu: ~/mysql80$ tar xvf mysql-8.0.13-linux-glibc2.12-x86_64.tar.xz  
pilou@ubuntu: ~/mysql80$ ls mysql-8.0.13-linux-glibc2.12-x86_64 bin include LICENSE man  
README.router support-files docs lib LICENSE.router README share
```

L'archive une fois décompresser contient:

- bin binaire mysqld, et outils client
- docs Documentation
- man Documentation
- include fichier de développement
- lib fichier de développement
- share Fichiers partagé par les bases MySQL
- support-files fichier support

NB: A ce niveau il n'y a pas de repertoire data qui a été crée.

Sous ubuntu, faire

```
cd /usr/lib/x86_64-linux-gnu
sudo ln -s libtinfo.so.6 libtinfo.so.5
```

Le répertoire bin contient:

```
pilou@pilou-pc: ~/Formation/mysql-8.0.20-linux-glibc2.12-x86_64$ ls
bin docs include lib LICENSE man README share support-files
```

# Initialisation de mysql

L'initialisation de MySQL permet de créer le repertoire data et de créer un utilisateur root

```
pilou@pilou-pc: ~/Formation/mysql-8.0.20-linux-glibc2.12-x86_64$ ./bin/mysqld --defaults-
file=simplemy.ini --console --initialize
pilou@pilou-pc: ~/Formation/mysql-8.0.20-linux-glibc2.12-x86_64$ ./bin/mysqld --defaults-
file=simplemy.ini --console
```

Le fichier ini étant:

```
[mysqld]
basedir=/home/pilou/Formation/mysql-8.0.20-linux-glibc2.12-x86_64
datadir=/home/pilou/Formation/simpleinit/data
log-error=/home/pilou/Formation/simpleinit/mysqld.log
```

Dans le fichier de log on retrouve le mot de passe de l'administrateur:

```
2020-05-11T21:03:21.864864Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2020-05-11T21:03:23.439599Z 6 [Note] [MY-010454] [Server] A temporary password is generated
for root@localhost: e;ActH??b0Es
2020-05-11T21:03:33.232344Z 0 [System] [MY-010116] [Server] /home/pilou/Formation/mysql-
8.0.20-linux-glibc2.12-x86_64/bin/mysqld (mysqld 8.0.20) starting as process 13368
```

## Test de MySQL

```
pilou@ubuntu: ~$ mysql -S /tmp/mysql.sock -u root -p
Enter password: Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection
id is 10 Server version: 8.0.13

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may
be trademarks of their respective owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

# Post Installation

La première chose à faire est de changer le password du root en utilisant l'outil mysql

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'piloupilou'; Query OK, 0 rows affected
(0.04 sec)

mysql> quit
```

Pour arrêter le serveur, l'outil cli mysqladmin permet de demander l'arrêt du serveur

```
/bin/mysqladmin -h localhost -u root -p shutdown
```

Côté serveur cela donne

```
2018-12-31T17:01:56.370223Z 15 [System] [MY-013172] [Server] Received SHUTDOWN from user
root.
```

# Création d'un fichier ini de base

Sur le serveur, dans le repertoire de mysql, on va créer un fichier my.cnf minimaliste afin de setter quelques variable

```
[mysqld]
port = 3306
socket = /tmp/mysql.sock skip-external-locking
key_buffer_size = 16K
max_allowed_packet = 1M
table_open_cache = 4
sort_buffer_size = 64K
read_buffer_size = 256K
read_rnd_buffer_size = 256K
net_buffer_length = 2K
thread_stack = 128K
```

# Optimization

# New Page

## MySQL Tuner

```
pierre@sd-124453: ~$ mysqltuner
>> MySQLTuner 1.7.2 - 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
Please enter your MySQL administrative login: root
Please enter your MySQL administrative password: [!!] Attempted to use login credentials, but
they were invalid.
pierre@sd-124453: ~$ mysqltuner
>> MySQLTuner 1.7.2 - 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
Please enter your MySQL administrative login: root
Please enter your MySQL administrative password: [OK] Currently running supported MySQL
version 5.7.28-0ubuntu0.18.04.4
[OK] Operating on 64-bit architecture

----- Log file Recommendations
-----
[--] Log file: /var/log/mysql/error.log(29K)
[OK] Log file /var/log/mysql/error.log exists
[OK] Log file /var/log/mysql/error.log is readable.
[OK] Log file /var/log/mysql/error.log is not empty
[OK] Log file /var/log/mysql/error.log is smaller than 32 Mb
[!!] /var/log/mysql/error.log contains 10 warning(s).
[OK] /var/log/mysql/error.log doesn't contain any error.
[--] 4 start(s) detected in /var/log/mysql/error.log
```

```
[--] 1) 2019-11-23T08:47:02.036400Z 0 [Note] /usr/sbin/mysqld: ready for connections.
[--] 2) 2019-11-23T08:37:40.958958Z 0 [Note] /usr/sbin/mysqld: ready for connections.
[--] 3) 2019-11-23T08:37:37.655761Z 0 [Note] mysqld: ready for connections.
[--] 4) 2019-11-23T08:37:33.079748Z 0 [Note] mysqld: ready for connections.
[--] 4 shutdown(s) detected in /var/log/mysql/error.log
[--] 1) 2019-11-23T08:45:37.822837Z 0 [Note] /usr/sbin/mysqld: Shutdown complete
[--] 2) 2019-11-23T08:37:39.291637Z 0 [Note] mysqld: Shutdown complete
[--] 3) 2019-11-23T08:37:36.367427Z 0 [Note] mysqld: Shutdown complete
[--] 4) 2019-11-23T08:33:50.907834Z 0 [Note] /usr/sbin/mysqld: Shutdown complete
```

----- Storage Engine Statistics

```
[--] Status: +ARCHIVE +BLACKHOLE +CSV -FEDERATED +InnoDB +MEMORY +MRG_MYISAM +MyISAM
+PERFORMANCE_SCHEMA
```

```
[--] Data in InnoDB tables: 11M (Tables: 92)
```

```
[OK] Total fragmented tables: 0
```

----- Security Recommendations

```
[OK] There are no anonymous accounts for any database users
```

```
[OK] All database users have passwords assigned
```

```
[--] Bug #80860 MySQL 5.7: Avoid testing password when validate_password is activated
```

----- CVE Security Recommendations

```
[OK] NO SECURITY CVE FOUND FOR YOUR VERSION
```

----- Performance Metrics

```
[--] Up for: 1h 19m 42s (1K q [0.365 qps], 109 conn, TX: 3M, RX: 788K)
```

```
[--] Reads / Writes: 89% / 11%
```

```
[--] Binary logging is disabled
```

```
[--] Physical Memory      : 31.3G
```

```
[--] Max MySQL memory     : 352.4M
```

```
[--] Other process memory: 1.1G
```

```
[--] Total buffers: 192.0M global + 1.1M per thread (151 max threads)
```

```
[--] P_S Max memory usage: 72B
```

```
[--] Galera GCache Max memory usage: 0B
```

```
[OK] Maximum reached memory usage: 194.1M (0.61% of installed RAM)
```

```
[OK] Maximum possible memory usage: 352.4M (1.10% of installed RAM)
```

[OK] Overall possible memory usage with other process is compatible with memory available  
[OK] Slow queries: 0% (0/1K)  
[OK] Highest usage of available connections: 1% (2/151)  
[!!] Aborted connections: 3.67% (4/109)  
[!!] name resolution is active : a reverse name resolution is made for each new connection and can reduce performance  
[!!] Query cache may be disabled by default due to mutex contention.  
[!!] Query cache efficiency: 0.0% (0 cached / 1K selects)  
[OK] Query cache prunes per day: 0  
[OK] Sorts requiring temporary tables: 0% (0 temp sorts / 129 sorts)  
[OK] No joins without indexes  
[!!] Temporary tables created on disk: 59% (39 on disk / 66 total)  
[OK] Thread cache hit rate: 98% (2 created / 109 connections)  
[OK] Table cache hit rate: 98% (353 open / 360 opened)  
[OK] Open file limit used: 0% (14/5K)  
[OK] Table locks acquired immediately: 100% (102 immediate / 102 locks)

----- Performance schema

-----  
[--] Memory used by P\_S: 72B  
[--] Sys schema is installed.

----- ThreadPool Metrics

-----  
[--] ThreadPool stat is disabled.

----- MyISAM Metrics

-----  
[!!] Key buffer used: 18.3% (3M used / 16M cache)  
[OK] Key buffer size / total MyISAM indexes: 16.0M/43.0K  
[!!] Read Key buffer hit rate: 50.0% (6 cached / 3 reads)

----- InnoDB Metrics

-----  
[--] InnoDB is enabled.  
[--] InnoDB Thread Concurrency: 0  
[OK] InnoDB File per table is activated  
[OK] InnoDB buffer pool / data size: 128.0M/11.3M  
[!!] Ratio InnoDB log file size / InnoDB Buffer pool size (75 %): 48.0M \* 2/128.0M should be equal 25%

```
[OK] InnoDB buffer pool instances: 1
[--] Number of InnoDB Buffer Pool Chunk : 1 for 1 Buffer Pool Instance(s)
[OK] Innodb_buffer_pool_size aligned with Innodb_buffer_pool_chunk_size &
Innodb_buffer_pool_instances
[OK] InnoDB Read buffer efficiency: 99.54% (137963 hits/ 138598 total)
[OK] InnoDB Write log efficiency: 91.67% (2772 hits/ 3024 total)
[OK] InnoDB log waits: 0.00% (0 waits / 252 writes)
```

----- AriaDB Metrics

-----  
[--] AriaDB is disabled.

----- TokuDB Metrics

-----  
[--] TokuDB is disabled.

----- XtraDB Metrics

-----  
[--] XtraDB is disabled.

----- RocksDB Metrics

-----  
[--] RocksDB is disabled.

----- Spider Metrics

-----  
[--] Spider is disabled.

----- Connect Metrics

-----  
[--] Connect is disabled.

----- Galera Metrics

-----  
[--] Galera is disabled.

----- Replication Metrics

-----  
[--] Galera Synchronous replication: NO  
[--] No replication slave(s) for this server.

```
[--] This is a standalone server.

----- Recommendations
-----

General recommendations:

  Control warning line(s) into /var/log/mysql/error.log file
  MySQL started within last 24 hours - recommendations may be inaccurate
  Reduce or eliminate unclosed connections and network issues
  Configure your accounts with ip or subnets only, then update your configuration with skip-
name-resolve=1

  When making adjustments, make tmp_table_size/max_heap_table_size equal
  Reduce your SELECT DISTINCT queries which have no LIMIT clause

Variables to adjust:

  query_cache_size (=0)
  query_cache_type (=0)
  query_cache_limit (> 1M, or use smaller result sets)
  tmp_table_size (> 16M)
  max_heap_table_size (> 16M)
  innodb_log_file_size * innodb_log_files_in_group should be equal to 1/4 of buffer pool
size (=64M) if possible.

pierre@sd-124453: ~$
```

# key\_buffer

Changer le `key_buffer` alloue plus de mémoire à MySQL, ce qui peut considérablement accélérer vos bases de données, à condition que vous ayez de la mémoire libre. La taille du fichier `key_buffer` ne doit généralement pas occuper plus de 25% de la mémoire système lors de l'utilisation du moteur de table MyISAM, et jusqu'à 70% pour InnoDB. Si la valeur est trop élevée, les ressources sont gaspillées.

Selon la documentation de MySQL, pour les serveurs disposant de 256 Mo (ou plus) de RAM et de nombreuses tables, il est recommandé de définir 64 Mo. Les serveurs avec 128 Mo de RAM et moins de tables peuvent être configurés à 16 Mo, la valeur par défaut. Les sites Web avec encore moins de ressources et de tableaux peuvent avoir cette valeur plus basse.

# max\_allowed\_packet

Ce paramètre vous permet de définir la taille maximale d'un paquet à envoyer. Un paquet est un seul état SQL, une seule ligne envoyée à un client ou un journal envoyé d'un maître à un esclave. Si vous savez que votre serveur MySQL va traiter de gros paquets, il est préférable d'augmenter la taille de votre plus gros paquet. Si cette valeur est trop petite, vous recevrez une erreur dans votre journal des erreurs.

# thread\_stack

Cette valeur contient la taille de la pile pour chaque thread. MySQL considère que la valeur par défaut de la variable thread\_stack est suffisante pour une utilisation normale. Toutefois, si une erreur liée à la thread\_stack est enregistrée, vous pouvez l'augmenter.

# thread\_cache\_size

Si thread\_cache\_size est «désactivé» (défini sur 0), toute nouvelle connexion en cours d'établissement nécessite la création d'un nouveau thread. Lorsque les connexions se désengagent, le fil est détruit. Sinon, cette valeur définit le nombre de threads inutilisés à stocker dans un cache jusqu'à ce qu'ils aient besoin d'être utilisés pour une connexion. Généralement, ce paramètre a peu d'incidence sur les performances, sauf si vous recevez des centaines de connexions par minute. À ce moment, vous devez augmenter cette valeur pour permettre à la majorité des connexions d'être établies sur des threads mis en cache.

# max\_connections

Ce paramètre définit le nombre maximal de connexions simultanées. Il est préférable de prendre en compte le nombre maximal de connexions que vous avez eu par le passé avant de définir ce nombre. Vous disposerez ainsi d'un tampon entre ce nombre supérieur et la valeur max\_connections. Notez que cela n'indique pas le nombre maximum d'utilisateurs sur votre site Web en même temps; il indique plutôt le nombre maximal d'utilisateurs qui font des demandes simultanément.

# Securité

# Authentification Simple et administration des droits

MySQL implémente un système sophistiqué de contrôle d'accès et de privilèges qui vous permet 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.

Le contrôle d'accès MySQL comporte deux étapes lorsqu'un client se connecte au serveur:

- Vérification de la connexion: un client qui se connecte au serveur de base de données MySQL doit avoir un nom d'utilisateur et un mot de passe valides. De plus, l'hôte à partir duquel le client se connecte doit correspondre à l'hôte dans la table de droits MySQL.
- Vérification de la demande: une fois la connexion établie avec succès, MySQL vérifie, pour chaque instruction émise par le client, si le client dispose des privilèges suffisants pour exécuter cette instruction. MySQL peut vérifier un privilège au niveau de la base de données, de la table et du champ.

## Création de comptes d'utilisateurs à l'aide de l'instruction MySQL CREATE USER

MySQL fournit l'instruction CREATE USER qui vous permet de créer un nouveau compte utilisateur. La syntaxe de l'instruction CREATE USER est la suivante:

```
CREATE USER "compte_utilisateur" IDENTIFIED BY "mot de passe";
```

Le compte utilisateur au format 'nom\_utilisateur' @ 'nom\_hôte' est suivi de la clause CREATE USER.

Le mot de passe est spécifié dans la clause IDENTIFIED BY. Le mot de passe doit être en texte clair. MySQL chiffrera le mot de passe avant de sauvegarder le compte utilisateur dans la table user.

```
CREATE USER myuser@localhost IDENTIFIED BY 'myuser';  
Query OK, 0 rows affected (0,06 sec)
```

Puis la connexion peut s'établir ainsi:

```
./bin/mysql -u myuser -h localhost -p
```

L'utilisateur peut se connecter mais n'a pour l'instant aucun droit

```
show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
+-----+  
1 row in set (0,00 sec)
```

Les droits de notre utilisateurs sont ainsi:

```
SHOW GRANTS FOR 'myuser'@'localhost';  
+-----+  
| Grants for myuser@localhost |  
+-----+  
| GRANT USAGE ON *.* TO `myuser`@`localhost` |  
+-----+  
1 row in set (0,00 sec)
```

# creation d'un utilisateur avec un plugin d'authentification

Il est possible d'utiliser un plugin d'authentification spécifique ici un plugin sha256

```
CREATE USER 'sha256user'@'localhost'  
-> IDENTIFIED WITH sha256_password BY 'password';
```

# Augmentation de la sécurité via le plugin de validation de password

Le plugin `validate_password` sert à tester les mots de passe et à améliorer la sécurité. Le plugin expose un ensemble de variables système qui vous permettent de définir une politique de mot de passe.

```
mysql> INSTALL PLUGIN validate_password SONAME 'validate_password.so';  
Query OK, 0 rows affected, 1 warning (0,07 sec)  
  
mysql> SELECT PLUGIN_NAME, PLUGIN_STATUS  
-> FROM INFORMATION_SCHEMA.PLUGINS  
-> WHERE PLUGIN_NAME LIKE 'validate%';  
+-----+-----+  
| PLUGIN_NAME | PLUGIN_STATUS |  
+-----+-----+  
| validate_password | ACTIVE |  
+-----+-----+  
1 row in set (0,00 sec)
```

On teste maintenant la création d'un mot de passe pauvre

```
create user 'bob'@'%' IDENTIFIED BY 'test';  
ERROR 1819 (HY000): Your password does not satisfy the current policy requirements
```

Il est possible de charger une stratégie de validation:

- La stratégie `LOW` teste uniquement la longueur du mot de passe. Les mots de passe doivent comporter au moins 8 caractères. Pour changer cette longueur, modifiez `validate_password_length`.

- La stratégie MEDIUM ajoute les conditions selon lesquelles les mots de passe doivent contenir au moins un caractère numérique, un caractère minuscule, un caractère majuscule et un caractère spécial (non alphanumérique). Pour modifier ces valeurs, modifiez `validate_password_number_count`, `validate_password_mixed_case_count` et `validate_password_special_char_count`.
- La stratégie STRONG ajoute la condition selon laquelle les sous-chaînes de mot de passe de longueur égale ou supérieure à 4 ne doivent pas correspondre aux mots du fichier de dictionnaire, s'il en a été spécifié. Pour spécifier le fichier de dictionnaire, modifiez `validate_password_dictionary_file`.

Ici on selectionne la stratégie medium par défaut

```
SET GLOBAL validate_password_policy = 1;
Query OK, 0 rows affected (0,00 sec)

mysql> create user 'bob'@'%' identified by 'aA!12345678';
Query OK, 0 rows affected (0,05 sec)
```

# Assignment d'un droit a un utilisateur

La commande GRANT permet d'associer un droit à un utilisateur. Nous donnons ici le droit show databases à l'utilisateur myuser

```
mysql> grant show databases on *.* TO 'myuser'@'localhost';
Query OK, 0 rows affected (0,13 sec)

mysql> \q
Bye
```

Nous pouvons maintenant le tester

```
pilou@lubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ ./bin/mysql -u myuser -h
localhost -p
mysql> show databases;
+-----+
| Database |
```

```
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| testUtf8 |
| testdb |
+-----+
6 rows in set (0,00 sec)
```

# Mise en place de Rôle

En règle générale, vous avez plusieurs utilisateurs avec le même ensemble de privilèges. Le moyen d'octroyer et de révoquer des privilèges à plusieurs utilisateurs de modifier les privilèges de chaque utilisateur individuellement demande beaucoup de temps.

Pour faciliter les choses, MySQL a fourni un nouvel objet appelé rôle, 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.

Nous allons créer une base de données exemple:

```
CREATE DATABASE crm;
Query OK, 1 row affected (0,05 sec)

mysql> use crm;
Database changed
mysql> CREATE TABLE customer(
```

```
-> id INT PRIMARY KEY AUTO_INCREMENT,  
-> first_name varchar(255) NOT NULL,  
-> last_name VARCHAR(255) NOT NULL,  
-> phone VARCHAR(15) NOT NULL,  
-> email VARCHAR(255)  
-> );  
Query OK, 0 rows affected (0,11 sec)  
  
mysql> INSERT INTO customer(first_name, last_name, phone, email)  
-> VALUES(' John', ' Doe', '( 408)-987-7654', ' john.doe@mysql.org' ),  
-> (' Lily', ' Bush', '( 408)-987-7985', ' lily.bush@mysql.org' );  
Query OK, 2 rows affected (0,06 sec)  
Records: 2 Duplicates: 0 Warnings: 0
```

Puis nous créons 3 rôle permettant respectivement d'être un super utilisateur de la base, de pouvoir la lire et de pouvoir l'écrire

```
mysql> CREATE ROLE crm_dev, crm_read, crm_write;  
Query OK, 0 rows affected (0,05 sec)  
  
mysql> GRANT ALL ON crm.* TO crm_dev;  
Query OK, 0 rows affected (0,07 sec)  
  
mysql> GRANT SELECT ON crm.* TO crm_read;  
Query OK, 0 rows affected (0,11 sec)  
  
mysql> GRANT INSERT, UPDATE, DELETE ON crm.* TO crm_write;  
Query OK, 0 rows affected (0,05 sec)
```

Puis nous créons des utilisateurs

```
mysql> -- developer user  
mysql> CREATE USER crm_dev1@localhost IDENTIFIED BY 'Secure$1782';  
Query OK, 0 rows affected (0,03 sec)  
  
mysql> -- read access user  
mysql> CREATE USER crm_read1@localhost IDENTIFIED BY 'Secure$5432';
```

```
Query OK, 0 rows affected (0,01 sec)
```

```
mysql> -- read/write users
```

```
mysql> CREATE USER crm_write1@localhost IDENTIFIED BY 'Secure$9075';
```

```
Query OK, 0 rows affected (0,01 sec)
```

```
mysql> CREATE USER crm_write2@localhost IDENTIFIED BY 'Secure$3452';
```

```
Query OK, 0 rows affected (0,14 sec)
```

Enfin nous associons nos utilisateurs a nos rôle

```
mysql> GRANT crm_dev TO crm_dev1@localhost;
```

```
Query OK, 0 rows affected (0,08 sec)
```

```
mysql>
```

```
mysql> GRANT crm_read TO crm_read1@localhost;
```

```
Query OK, 0 rows affected (0,01 sec)
```

```
mysql>
```

```
mysql> GRANT crm_read, crm_write TO crm_write1@localhost, crm_write2@localhost;
```

```
Query OK, 0 rows affected (0,11 sec)
```

Il est possible de voir les droits d'un utilisateur, qui sont en fait les droits de l'utilisateur associé aux rôles:

```
SHOW GRANTS FOR crm_dev1@localhost;
```

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

```
| Grants for crm_dev1@localhost |
```

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

```
| GRANT USAGE ON *.* TO `crm_dev1`@`localhost` |
```

```
| GRANT `crm_dev`@`%` TO `crm_dev1`@`localhost` |
```

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

```
2 rows in set (0,00 sec)
```

Pour voir les droits d'un utilisateur en fonction d'un rôle, il faut le demander explicitement via la clause USING

```
mysql> SHOW GRANTS FOR crm_writel@localhost USING crm_write;
+-----+
| Grants for crm_writel@localhost |
+-----+
| GRANT USAGE ON *.* TO `crm_writel`@`localhost` |
| GRANT INSERT, UPDATE, DELETE ON `crm`.* TO `crm_writel`@`localhost` |
| GRANT `crm_read`@`%`,`crm_write`@`%` TO `crm_writel`@`localhost` |
+-----+
3 rows in set (0,00 sec)
```

Les droits sont associé a un rôle. Lors de la connection à MySQL, l'utilisateur doit spécifier le rôle qu'il souhaite utiliser

```
pilou@lubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ ./bin/mysql -u crm_read1 -h
localhost -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 8.0.13 MySQL Community Server - GPL

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select current_role();
+-----+
| current_role() |
+-----+
| NONE |
+-----+
1 row in set (0,00 sec)

mysql> SET ROLE crm_read;
Query OK, 0 rows affected (0,00 sec)

mysql> select current_role() ;
```

```
+-----+
| current_role() |
+-----+
| `crm_read`@`%` |
+-----+
1 row in set (0,00 sec)

mysql> use crm;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

# connexion via PAM

Les plugins d'authentification sont multiples en version entreprise et sont souvent en version GPL avec mariadb.

Il est possible d'authentifier les utilisateurs via PAM en utilisant le systeme unix pour les authentifier en lieu et place de MySQL.

Sous mariadb installer le plugin auth\_pam

```
INSTALL SONAME 'auth_pam';
```

Puis créer un utilisateur qui pourras se connecter via pam

```
CREATE USER 'test_pam'@'localhost' IDENTIFIED VIA pam USING 'mariadb';
```

Créer l'utilisateur sous Linux (ici ubuntu)

```
addUser test_pam
```

Dans le repertoire /etc/pam.d on va editer la configuration pam de MySQL en demandant a ce que la vérification des utilisateurs se fasse via les comptes Unix

```
#%PAM-1.0
```

```
@include common-auth
```

```
@include common-account
```

```
@include common-session-noninteractive
```

# Connection et SSL

## maximum de connection et maximum de connection utilisateur

- `max_user_connections` : Le nombre maximum de connexions simultanées autorisées sur un compte utilisateur MySQL donné. Une valeur de 0 (valeur par défaut) signifie «aucune limite». Cette variable a une valeur globale qui peut être définie au démarrage ou à l'exécution du serveur. Il a également une valeur de session en lecture seule qui indique la limite effective de connexion simultanée qui s'applique au compte associé à la session en cours.
- `max_connections` Le nombre maximum autorisé de connexions client simultanées

Par défaut dans l'installation :

15

1

```
mysql> select @@max_user_connections ;
```

2

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

3

```
| @@max_user_connections |
```

4

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

5

```
| 0 |
```

6

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

7

```
1 row in set (0,00 sec)
```

8

```
□
```

9

```
mysql> select @@max_connections ;
```

10

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

11

```
| @@max_connections |
```

12

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

13

```
| 151 |
```

14

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

15

1 row in set (0,00 sec)

Un bon conseil est de fixer `max_user_connections` à 50 à 75% de vos paramètres `max_connections`. Vous définissez cette valeur dans la section `mysqld` de votre `my.cnf`:

2

1

```
max_connections = 400
```

2

```
max_user_connections=200
```

# maximum de connection pour un utilisateur

Le settings précédent concerne une mise en place assez globale du nombre de connection.

Il est possible de signifier des limits plus fine en terme de temps et de ressources

Il existe différents types de limites pouvant être utilisés:

- `MAX_QUERIES_PER_HOUR` Limite le compte à X requêtes par heure.
- `MAX_UPDATES_PER_HOUR` Limite le compte à X relevés UPDATE par heure.
- `MAX_CONNECTIONS_PER_HOUR` Limite le compte à un total de X connexions par heure.
- `MAX_USER_CONNECTIONS` Limite le compte à un total de X connexions simultanées pour le compte.

Par exemple, on limite le nombre de connection de myuser à 5

2

1

```
mysql> ALTER USER 'myuser'@'localhost' WITH MAX_USER_CONNECTIONS 5;
```

2

```
□
```

# LOCK et Unlock Account

Account lock et Account unlock permet de verrouiller ou pas un utilisateur

9

1

```
ALTER USER 'myuser'@'localhost' ACCOUNT LOCK;
```

2

```
Query OK, 0 rows affected (0,09 sec)
```

3

```
□
```

4

```
mysql> \q
```

5

```
Bye
```

6

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ ./bin/mysql -u myuser -h localh
```

7

```
Enter password:
```

8

```
ERROR 3118 (HY000): Access denied for user 'myuser'@'localhost'. Account is locked.
```

9

```
□
```

# Mise en place de SSL

Pour l'instant la connection entre le client et le serveur est faite en claire.

23

1

status

2

-----

3

./bin/mysql Ver 8.0.13 for linux-glibc2.12 on x86\_64 (MySQL Community Server - GPL)

4

□

5

Connection id: 10

6

Current database:

7

Current user: root@localhost

8

SSL: Not in use

9

Current pager: stdout

10

Using outfile: ''

11

Using delimiter: ;

12

Server version: 8.0.13 MySQL Community Server - GPL

13

Protocol version: 10

14

Connection: Localhost via UNIX socket

15

Server characterset: utf8mb4

16

Db characterset: utf8mb4

17

Client characterset: utf8mb4

18

Conn. characterset: utf8mb4

19

UNIX socket: /tmp/mysql.sock

20

Uptime: 29 min 39 sec

21

□

22

Threads: 2 Questions: 22 Slow queries: 0 Opens: 136 Flush tables: 2 Open tables: 106 Queries

23

-----

# Création de l'autorité de certification

Exécutez les commandes suivantes pour créer les clés de l'autorité de certification (CA):

7

1

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ mkdir ssl_keys
```

2

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ openssl genrsa 2048 > ./ssl_key
```

3

```
Generating RSA private key, 2048 bit long modulus (2 primes)
```

4

```
.....+++++
```

5

```
.....+++++
```

6

```
e is 65537 (0x010001)
```

7

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ openssl req -sha1 -new -x509 -r
```

# Création de la clef serveur et du certificat serveur

Exécutez les commandes suivantes pour créer la clé SSL et le certificat du serveur:

3

1

```
openssl req -sha1 -newkey rsa:2048 -days 3650 -nodes -keyout ./ssl_keys/server-key.pem > ./s
```

2

```
openssl x509 -sha1 -req -in ./ssl_keys/server-req.pem -days 3650 -CA ./ssl_keys/ca-cert.pem
```

3

```
openssl rsa -in ./ssl_keys/server-key.pem -out ./ssl_keys/server-key.pem
```

## Création de la clef serveur et du certificat client

Exécutez les commandes suivantes pour créer la clé SSL et le certificat du client:

3

1

```
openssl req -sha1 -newkey rsa:2048 -days 3650 -nodes -keyout ./ssl_keys/client-key.pem > ./s
```

2

```
openssl x509 -sha1 -req -in ./ssl_keys/client-req.pem -days 3650 -CA ./ssl_keys/ca-cert.pem
```

3

```
openssl rsa -in ./ssl_keys/client-key.pem -out ./ssl_keys/client-key.pem
```

## Sortie de OpenSSL

Pour avoir de bon certifiact, il est important de selectionner des CN différents pour les CA, server et client

77

1

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ ./ssl.sh
```

2

```
Generating RSA private key, 2048 bit long modulus (2 primes)
```

3

```
.....+++++
```

4

```
.....+++++
```

5

```
e is 65537 (0x010001)
```

6

```
You are about to be asked to enter information that will be incorporated
```

7

```
into your certificate request.
```

8

```
What you are about to enter is what is called a Distinguished Name or a DN.
```

9

```
There are quite a few fields but you can leave some blank
```

10

```
For some fields there will be a default value,
```

11

```
If you enter '.', the field will be left blank.
```

12

```
-----
```

13

```
Country Name (2 letter code) [AU]:
```

14

State or Province Name (full name) [Some-State]:

15

Locality Name (eg, city) []:

16

Organization Name (eg, company) [Internet Widgits Pty Ltd]:

17

Organizational Unit Name (eg, section) []:

18

Common Name (e.g. server FQDN or YOUR name) []: CA

19

Email Address []:

20

Ignoring -days; not generating a certificate

21

Generating a RSA private key

22

.....+++++

23

.....+++++

24

writing new private key to '/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86\_64/ssl\_keys

25

-----

26

You are about to be asked to enter information that will be incorporated

27

into your certificate request.

28

What you are about to enter is what is called a Distinguished Name or a DN.

29

There are quite a few fields but you can leave some blank

30

For some fields there will be a default value,

31

If you enter '.', the field will be left blank.

32

-----

33

Country Name (2 letter code) [AU]:

34

State or Province Name (full name) [Some-State]:

35

Locality Name (eg, city) []:

36

Organization Name (eg, company) [Internet Widgits Pty Ltd]:

37

Organizational Unit Name (eg, section) []:

38

Common Name (e.g. server FQDN or YOUR name) []: server

39

Email Address []:

40

□

41

Please enter the following 'extra' attributes

42

to be sent with your certificate request

43

A challenge password []:

44

An optional company name []:

45

Signature ok

46

subject=C = AU, ST = Some-State, O = Internet Widgits Pty Ltd, CN = server

47

Getting CA Private Key

48

writing RSA key

49

Ignoring -days; not generating a certificate

50

Generating a RSA private key

51

.....+++++

52

.....+++++

53

writing new private key to '/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86\_64/ssl\_keys

54

-----

55

You are about to be asked to enter information that will be incorporated

56

into your certificate request.

57

What you are about to enter is what is called a Distinguished Name or a DN.

58

There are quite a few fields but you can leave some blank

59

For some fields there will be a default value,

60

If you enter '.', the field will be left blank.

61

-----

62

Country Name (2 letter code) [AU]:

63

State or Province Name (full name) [Some-State]:

64

Locality Name (eg, city) []:

65

Organization Name (eg, company) [Internet Widgits Pty Ltd]:

66

Organizational Unit Name (eg, section) []:

67

Common Name (e.g. server FQDN or YOUR name) []:client

68

Email Address []:

69

□

70

Please enter the following 'extra' attributes

71

to be sent with your certificate request

72

A challenge password []:

73

An optional company name []:

74

Signature ok

75

subject=C = AU, ST = Some-State, O = Internet Widgits Pty Ltd, CN = client

76

Getting CA Private Key

77

writing RSA key



# Modification de MySQL

Il faut indiquer à MySQL où se trouvent les différentes clés et certificats:

24

1

```
[mysqld]
```

2

```
port = 3306
```

3

```
socket = /tmp/mysql.sock
```

4

```
skip-external-locking
```

5

```
key_buffer_size = 16K
```

6

```
max_allowed_packet = 1M
```

7

```
table_open_cache = 4
```

8

```
sort_buffer_size = 64K
```

9

```
read_buffer_size = 256K
```

10

```
read_rnd_buffer_size = 256K
```

11

```
net_buffer_length = 2K
```

12

```
thread_stack = 128K
```

13

```
table_open_cache=500
```

14

```
secure_file_priv=/tmp
```

15

```
max_connections = 400
```

16

```
max_user_connections=200
```

17

```
ssl-ca=/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/ssl_keys/ca-cert.pem
```

18

```
ssl-cert=/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/ssl_keys/server-cert.pem
```

19

```
ssl-key=/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/ssl_keys/server-key.pem
```

20

```
ssl-cipher=DHE-RSA-AES256-SHA
```

21

```
□
```

22

```
[client]
```

23

```
ssl-cert=/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/ssl_keys/client-cert.pem
```

24

```
ssl-key=/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/ssl_keys/client-key.pem
```

Après redemarrage, le serveur signale que le certificat est auto signé

2

1

```
2019-01-05T08:44:05.815408Z 0 [Warning] [MY-010068] [Server] CA certificate
```

2

```
/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/ssl_keys/ca-cert.pem is self signed.
```

et les parametre SSL sont bien chargé





# Test

Nous allons nous connecter en SSL sur le serveur en demandant explicitement a utiliser la connection TCP (ce qui force l'utilisation de SSL)

34

1

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ ./bin/mysql --defaults-file=hc
```

2

```
Enter password:
```

3

```
Welcome to the MySQL monitor. Commands end with ; or \g.
```

4

```
Your MySQL connection id is 8
```

5

```
Server version: 8.0.13 MySQL Community Server - GPL
```

6

```
□
```

7

```
Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.
```

8

```
□
```

9

```
Oracle is a registered trademark of Oracle Corporation and/or its
```

10

```
affiliates. Other names may be trademarks of their respective
```

11

```
owners.
```

12

```
□
```

13

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

14

□

15

```
mysql> status
```

16

```
-----
```

17

```
./bin/mysql Ver 8.0.13 for linux-glibc2.12 on x86_64 (MySQL Community Server - GPL)
```

18

□

19

```
Connection id: 8
```

20

```
Current database:
```

21

```
Current user: root@localhost
```

22

```
SSL: Cipher in use is DHE-RSA-AES256-SHA
```

23

```
Current pager: stdout
```

24

```
Using outfile: ''
```

25

```
Using delimiter: ;
```

26

```
Server version: 8.0.13 MySQL Community Server - GPL
```

27

```
Protocol version: 10
```

28

```
Connection: localhost via TCP/IP
```

29

Server charsetset: utf8mb4

30

Db charsetset: utf8mb4

31

Client charsetset: utf8mb4

32

Conn. charsetset: utf8mb4

33

TCP port: 3306

34

Uptime: 20 sec

Installation de SSL pour MariaDB

Créez un répertoire nommé ssl dans le répertoire /etc/mysql/

3

1

```
$ cd /etc/mysql
```

2

```
$ sudo mkdir ssl
```

3

```
$ cd ssl
```

4

1

La valeur du nom commun utilisée pour les certificats/clés du serveur et du client doit être

2

Nom commun de l'AC : administrateur MariaDB

3

Nom commun du serveur : serveur MariaDB

4

Nom commun du client : client MariaDB

Tapez la commande suivante pour créer une nouvelle autorité de certification:

# Installation de SSL avec MariaDB sous Debian

Modifiez le fichier `/etc/mysql/mariadb.conf.d/50-server.cnf` (ou `/etc/mysql/mariadb.cnf`) comme suit :  
La valeur du nom commun utilisée pour les certificats/clés du serveur et du client doit être différente de la valeur du nom commun utilisée pour le certificat CA. Pour éviter tout problème, je les règle comme suit. Sinon, vous obtiendrez une erreur d'échec de la vérification de la certification. Par conséquent, définissez-le comme suit :

Nom commun de l'AC : administrateur MariaDB

Nom commun du serveur : serveur MariaDB

Nom commun du client : client MariaDB

Créez un répertoire nommé `ssl` dans le répertoire `/etc/mysql/`

```
$ cd /etc/mysql
$ sudo mkdir ssl
$ cd ssl
```

Tapez la commande suivante pour créer une nouvelle clé CA :

```
sudo openssl genrsa 4096 > ca-key.pem
```

Tapez la commande suivante pour générer le certificat à l'aide de cette clé :

```
$ sudo openssl req -new -x509 -nodes -days 365000 -key ca-key.pem -out ca-cert.pem
```

Maintenant, vous devez avoir deux fichiers comme suit :

- `/etc/mysql/ssl/ca-cert.pem` – Fichier de certificat pour l'autorité de certification (CA).
- `/etc/mysql/ssl/ca-key.pem` – Fichier clé pour l'autorité de certification (CA).

## Créer le certificat SSL du serveur

Pour créer la clé du serveur, exécutez :

```
sudo openssl req -newkey rsa:2048 -days 365000 -nodes -keyout server-key.pem -out server-req.pem
```

Ensuite, créer la clé RSA du serveur, saisissez :

```
openssl rsa -in server-key.pem -out server-key.pem
```

Enfin, signez le certificat du serveur, exécutez :

```
openssl x509 -req -in server-req.pem -days 365000 -CA ca-cert.pem -CAkey ca-key.pem -  
set_serial 01 -out server-cert.pem
```

Maintenant, vous devez avoir des fichiers supplémentaires :

- /etc/mysql/ssl/server-cert.pem – Fichier de certificat du serveur MariaDB.
- /etc/mysql/ssl/server-key.pem – Fichier de clé du serveur MariaDB.

Vous devez utiliser les deux fichiers ci-dessus sur le serveur MariaDB lui-même et sur tout autre nœud que vous allez utiliser pour le trafic de cluster/réplication. Ces deux fichiers sécuriseront la communication côté serveur.

## Configurer le serveur MariaDB pour utiliser SSL

Modifiez le fichier /etc/mysql/mariadb.conf.d/50-server.cnf comme suit :

```
### MySQL Server ###  
## Securing the Database with ssl option and certificates ##  
## There is no control over the protocol level used. ##  
## mariadb will use TLSv1.0 or better. ##  
#ssl  
ssl-ca=/etc/mysql/ssl/ca-cert.pem  
ssl-cert=/etc/mysql/ssl/server-cert.pem  
ssl-key=/etc/mysql/ssl/server-key.pem  
## Set up TLS version here. For example TLS version 1.2 and 1.3 ##  
tls_version = TLSv1.2, TLSv1.3
```

Enregistrez et fermez le fichier. Sécurisez les clés à l'aide de la commande chmod/commande chown :

```
sudo chown -Rv mysql:root /etc/mysql/ssl/
```

# Administration MySQL

# Exportation des données en CSV

Pour exporter les données en CSV, il est possible de préciser à la sortie d'une requete SQL, le OUTPUT FILE

Mais il faut préciser dans le fichier my.cnf un répertoire sécurisé:

```
mysql> select * from myidtables INTO OUTFILE 'myids.csv' ;
ERROR 1290 (HY000): The MySQL server is running with the --secure-file-priv option so it
cannot execute this statement
```

Dans le fichier my.cnf, rajouter le repertoire de sauvegarde des données:

```
secure_file_priv=/tmp
```

Une fois signifier, les données peuvent se faire sauvegardé dans le repertoire donné.

```
mysql> SHOW VARIABLES LIKE "secure_file_priv";
+-----+-----+
| Variable_name | Value |
+-----+-----+
| secure_file_priv | /tmp/ |
+-----+-----+
1 row in set (0,00 sec)

mysql> select * from myidtables INTO OUTFILE '/tmp/myids.csv' ;
Query OK, 1 row affected (0,00 sec)
```

Il est possible de signifier le caractère de terminaison inter valeur ainsi que le caractère entourant les valeurs

```
select * from myidtables INTO OUTFILE '/tmp/myids2.csv' FIELDS ENCLOSED BY '"' TERMINATED BY
';' ;
```

## Importation des données

Il est possible de reimporter des données CSV dans une table mysql.  
En premier lieu, nous créons une nouvelle table

```
mysql> create table import( i int);  
Query OK, 0 rows affected (0,06 sec)
```

Puis nous utilisons la requete LOAD DATA INFILE ""nom de fichier"" into table ""table name""

```
mysql> LOAD DATA INFILE '/tmp/myids2.csv' into table import FIELDS ENCLOSED BY '"' TERMINATED  
BY ';' ;  
Query OK, 1 row affected (0,12 sec)  
Records: 1 Deleted: 0 Skipped: 0 Warnings: 0
```

## Exportation de données avec des en-têtes de colonne

Il serait pratique que le fichier CSV contienne la première ligne comme en-tête de colonne afin que le fichier soit plus compréhensible.

Pour ajouter les en-têtes de colonne, vous devez utiliser l'instruction UNION comme suit:

```
mysql> select 'key' union select * from myidtables INTO OUTFILE '/tmp/myids3.csv' FIELDS  
ENCLOSED BY '"' TERMINATED BY ';' ;
```

Donne comme fichier

```
<source lang=' bash' >  
"key"  
"1"  
</source>
```

# Tuning MySQL

## innodb\_file\_per\_table

Cette option permet d'avoir 1 fichier par table InnoDB au lieu d'un seul fichier

```
innodb_file_per_table=ON
```

## innodb\_stats\_on\_metadata

Lorsque l'option est définie sur ON, les statistiques d'index InnoDB sont mises à jour lors de l'exécution de `SHOW TABLE STATUS`, de `SHOW INDEX` ou de la requête d'`INFORMATION_SCHEMA.TABLES` ou d'`INFORMATION_SCHEMA.STATISTICS`. Ces statistiques incluent la cardinalité et le nombre d'entrées. Elles sont utilisées par l'optimiseur pour trouver un plan d'exécution optimal.

Ainsi, même si les instructions `SELECT` ne peuvent pas modifier les statistiques réelles, MySQL met à jour les statistiques des tables InnoDB. C'est contre-intuitif.

Est-ce utile? Pas vraiment, car InnoDB calculera toujours des statistiques lorsque vous ouvrez une table pour la première fois et lorsque des parties importantes de la table ont été modifiées (et lorsque vous exécutez `ANALYZE TABLE`).

Maintenant, pourquoi avons-nous eu une charge de lecture aussi élevée lorsque `innodb_stats_on_metadata` a été défini sur ON? Pour InnoDB, les statistiques sont estimées à partir de plongées à index aléatoire, ce qui se traduit par des lectures aléatoires.

## innodb\_buffer\_pool\_instances

Diviser le pool mémoire en plusieurs "régions" peut permettre d'améliorer l'accès aux données de façon concurrente, de diminuer les conflits issus des threads lors d'opérations de lecture ou d'écriture sur des pages verrouillées.

Chaque pool utilise alors sa propre liste d'accès aux données, sa liste de "flush", son propre mécanisme de LRU et surtout son propre buffer "Mutex".

Tous ces éléments rendent chaque région indépendante et permettent de diminuer les phénomènes de contention.

Ce paramètre n'est efficace que si la définition mémoire du `innodb_buffer_pool_size` est supérieure à 1 Go.

## `innodb_change_buffer_max_size`

La variable `innodb_change_buffer_max_size` permet de configurer la taille maximale de la mémoire tampon de changement sous forme de pourcentage de la taille totale du pool de mémoire tampon. Par défaut, `innodb_change_buffer_max_size` est défini sur 25. Le paramètre maximal est 50.

Pensez à augmenter `innodb_change_buffer_max_size` sur un serveur MySQL avec une activité importante d'insertion, de mise à jour et de suppression, où la fusion de tampons de modification ne suit pas le rythme des nouvelles entrées de tampon de modifications, ce qui lui permet d'atteindre sa taille maximale.

Envisagez de réduire `innodb_change_buffer_max_size` sur un serveur MySQL avec des données statiques utilisées pour la génération de rapports, ou si la mémoire tampon de modification utilise trop d'espace mémoire partagé avec le pool de mémoire tampon, ce qui entraîne la disparition plus rapide des pages du pool de mémoire tampon.

## `query_cache_type`

Sert de cache au requete coté serveur. A desactiver, les performances sont mauvaise.

Sinon mettre le `query_cache_type` à 2

```
query_cache_type = 1  
query_cache_size = 256M
```

`query_cache_type` est le type de cache que l'on va adopter:

0 = pas de cache

1 = met en cache toutes les requetes sauf celles qui ont le flag "SELECT S\_NO\_CACHE"

2 = met en cache seulement les requetes qui comportent le flag "SELECT SQL\_CACHE"

# innodb\_flush\_method

La bonne valeur pour éviter un double buffering est O\_DIRECT

# SQL Mode

Le serveur MySQL peut fonctionner dans différents modes SQL et peut appliquer ces modes différemment pour différents clients, en fonction de la valeur de la variable système `sql_mode`. Les administrateurs de base de données peuvent définir le mode SQL global pour répondre aux exigences opérationnelles du serveur de site. Chaque application peut définir son mode de session SQL en fonction de ses propres exigences.

- `ALLOW_INVALID_DATES`

N'effectuez pas de vérification complète des dates. Vérifiez uniquement que le mois est compris entre 1 et 12 et le jour entre 1 et 31.

- `ANSI_QUOTES`

Traitez "comme un caractère de citation d'identificateur (comme le ` caractère de citation) et non comme un caractère de citation de chaîne.

- `ERROR_FOR_DIVISION_BY_ZERO`

Le mode `ERROR_FOR_DIVISION_BY_ZERO` affecte le traitement de la division par zéro, ce qui inclut `MOD (N, 0)`. Pour les opérations de changement de données (`INSERT`, `UPDATE`), son effet dépend également de l'activation du mode SQL strict.

- ◦ Si ce mode n'est pas activé, la division par zéro insère `NULL` et ne produit aucun avertissement.
- ◦ Si ce mode est activé, la division par zéro insère `NULL` et génère un avertissement.

- `HIGH_NOT_PRECEDENCE`

Définiie la précedence de l'opérateur `NOT` .

- `NO_BACKSLASH_ESCAPES`

Désactive l'utilisation du caractère barre oblique inverse (`\`) en tant que caractère d'échappement dans les chaînes. Lorsque ce mode est activé, la barre oblique inverse devient un caractère ordinaire comme un autre.



```
-----+ 1 row in set (0,04 sec)
```

# Test SQL Mode

la variable `sql_mode` permet de savoir le mode sql par default du serveur

```
mysql> SHOW SESSION VARIABLES LIKE 'sql_mode';
+-----+
+-----+
-----+
| Variable_name | Value |
+-----+
+-----+
-----+
| sql_mode |
ONLY_FULL_GROUP_BY, STRICT_TRANS_TABLES, NO_ZERO_IN_DATE, NO_ZERO_DATE, ERROR_FOR_DIVISION_BY_ZERO
, NO_ENGINE_SUBSTITUTION |
+-----+
+-----+
-----+
1 row in set (0,00 sec)
```

Il est possible de tester le mode et d'override pour la session la prise en compte du SQL

```
mysql> create table test(i INT); Query OK, 0 rows affected (0,12 sec)

mysql> insert into test values (2/0); ERROR 1365 (22012): Division by 0

mysql> set sql_mode=; Query OK, 0 rows affected (0,00 sec)

mysql> insert into test values (2/0); Query OK, 1 row affected (0,06 sec)
```

# Migration

## Migration

La façon la plus simple est de faire un dump des données en SQL via mysqldump et de reinjecter les données dans une base vierge. Malheureusement ce n'est pas forcément possible du au fait de la taille des données. Nous allons essayer de faire une migration directe

## InnoDB Fast Shutdown

Le mode d'arrêt InnoDB. Si la valeur est 0, InnoDB effectue un arrêt lent, une purge complète et une fusion de la mémoire tampon de modification avant la fermeture. Si la valeur est 1 (valeur par défaut), InnoDB ignore ces opérations à l'arrêt, processus appelé arrêt rapide. Si la valeur est 2, InnoDB vide ses journaux et s'arrête, comme si MySQL s'était écrasé; aucune transaction validée n'est perdue, mais l'opération de récupération sur incident prend plus de temps au prochain démarrage.

L'arrêt lent peut prendre des minutes, voire des heures dans les cas extrêmes, où d'importantes quantités de données sont encore mises en mémoire tampon. Utilisez la technique d'arrêt lent avant de mettre à niveau ou de rétrograder les versions majeures de MySQL, afin que tous les fichiers de données soient entièrement préparés au cas où le processus de mise à niveau met à jour le format de fichier. Il est possible de voir la variable via

```
SHOW SESSION VARIABLES LIKE 'innodb_fast%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| innodb_fast_shutdown | 1 |
+-----+-----+
1 row in set (0,02 sec)
```

et positionner la valeur à 0

```
mysql> set global innodb_fast_shutdown=0; Query OK, 0 rows affected (0,00 sec)
```

# Migration

1. Passer le maximum de table InnoDB qui ont des capacités auto reparatrice
2. Bloquer les connections entrantes via iptables
3. executer mysqlcheck

```
pilou@ubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64$ ./bin/mysqlcheck --defaults-  
file=/home/pilou/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/my.cnf -u root -h localhost -p --  
all-databases --check-upgrade  
Enter password:  
crm.customer 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  
mysql.servers OK  
mysql.slave_master_info OK  
mysql.slave_relay_log_info OK  
mysql.slave_worker_info OK  
mysql.slow_log OK
```

mysql.tables\_priv OK  
mysql.time\_zone OK  
mysql.time\_zone\_leap\_second OK  
mysql.time\_zone\_name OK  
mysql.time\_zone\_transition OK  
mysql.time\_zone\_transition\_type OK  
mysql.user OK  
sqlmode.test OK  
sys.sys\_config OK  
testdb.import OK  
testdb.myidtables OK  
testdb.t2 OK  
testdb.t4 OK

# Transaction

Une transaction, c'est un ensemble de requêtes qui sont exécutées en un seul bloc. Ainsi, si une des requêtes du bloc échoue, on peut décider d'annuler tout le bloc de requêtes (ou de quand même valider les requêtes qui ont réussi). On peut donc considérer que chaque requête constitue une transaction qui est automatiquement commitée. Par défaut, MySQL est donc en mode "autocommit".

Pour quitter ce mode, il suffit de lancer la requête suivante :

```
set autocommit=0;
Query OK, 0 rows affected (0,00 sec)
```

Inserons une donnée dans une table

```
mysql> use transaction;
Database changed
mysql> create table testtransaction(i integer) engine='INNODB';
Query OK, 0 rows affected (0,09 sec)
```

| Connection 1  | Connection 2  |
|---|---|
| <pre>mysql&gt; begin; Query OK, 0 rows affected (0,00 sec) mysql&gt; insert into testtransaction values (2); Query OK, 1 row affected (0,00 sec) mysql&gt; select * from testtransaction;</pre> | <pre>mysql&gt; select * from testtransaction;  Empty set (0,01 sec)</pre> |

Lors du commit:

| Connection 1 | Connection 2 |
|--------------|--------------|
|--------------|--------------|

```
mysql> commit ;  
Query OK, 0 rows affected (0,10 sec)
```

```
select * from testtransaction;  
+-----+  
| i     |  
+-----+  
|    2 |  
+-----+  
1 row in set (0,00 sec)
```

L'operation de rollback permet d'annuler une transaction:

```
mysql> begin;  
Query OK, 0 rows affected (0,00 sec)  
  
mysql> insert into testtransaction values (5);  
Query OK, 1 row affected (0,00 sec)  
  
mysql> select * from testtransaction;  
+-----+  
| i     |  
+-----+  
|    2 |  
|    5 |  
+-----+  
2 rows in set (0,00 sec)  
  
mysql> rollback;  
Query OK, 0 rows affected (0,01 sec)  
  
mysql> select * from testtransaction;  
+-----+  
| i     |  
+-----+  
|    2 |  
+-----+  
1 row in set (0,00 sec)
```

Il faut faire attention que les comit/rollback en concerne pas les opération ddl (create table, alter ...)

| Connection 1  | Connection 2  |
|---|---|
| <pre>mysql&gt; begin   -&gt; ; Query OK, 0 rows affected (0,00 sec)  mysql&gt; alter table testtransaction add column testcolumn integer; Query OK, 0 rows affected (0,12 sec) Records: 0 Duplicates: 0 Warnings: 0</pre> | <pre>mysql&gt; select * from testtransaction; +-----+-----+   i       testcolumn   +-----+-----+        2            NULL   +-----+-----+ 1 row in set (0,00 sec)</pre> |

Lors du rollback:

| Connection 1   | Connection 2  |
|--|---|
| <pre>mysql&gt; rollback ; Query OK, 0 rows affected (0,00 sec)</pre> | <pre>mysql&gt; select * from testtransaction; +-----+-----+   i       testcolumn   +-----+-----+        2            NULL   +-----+-----+ 1 row in set (0,00 sec)</pre> |

## Transaction Level

Il est possible de savoir le niveau des transaction via:

```
-- mysql> show variables like 'tx_isolation';
-- sous mysql 8 tx_isolation a été renommé en transaction_isolation
mysql> show variables like 'transaction_isolation';
+-----+-----+
| Variable_name      | Value          |
+-----+-----+
| transaction_isolation | REPEATABLE-READ |
+-----+-----+
1 row in set (0,00 sec)
```

## Repeatable Read

C'est le niveau d'isolation par défaut pour InnoDB. Des lectures cohérentes dans la même transaction lisent l'instantané établi par la première lecture. Cela signifie que si vous émettez plusieurs instructions SELECT simples (non verrouillables) dans la même transaction, ces instructions SELECT sont également cohérentes.

| Connection 1  | Connection 2   |
|---|--|
| <pre>SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; Query OK, 0 rows affected (0,00 sec)  mysql&gt; create table transactionlevel (i integer); Query OK, 0 rows affected (0,12 sec)  mysql&gt; begin -&gt; ; Query OK, 0 rows affected (0,00 sec)  mysql&gt; insert into transactionlevel values (2); Query OK, 1 row affected (0,00 sec)</pre> | <pre>mysql&gt; SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;  Query OK, 0 rows affected (0,00 sec)  mysql&gt; begin; Query OK, 0 rows affected (0,00 sec)  mysql&gt; select * from transactionlevel; Empty set (0,00 sec)</pre> |

Après le commit de la première connexion, on obtient:

| Connection 1   | Connection 2  |
|--|---|
| <pre>commit; Query OK, 0 rows affected (0,06 sec)  mysql&gt; select * from transactionlevel; +-----+   i       +-----+       2   +-----+ 1 row in set (0,00 sec)</pre> | <pre>mysql&gt; select * from transactionlevel; Empty set (0,00 sec)</pre> |

Il faut attendre le commit de la transaction pour voir les changements;

```
mysql> select * from transactionlevel;
Empty set (0,00 sec)

mysql> commit;
Query OK, 0 rows affected (0,00 sec)

mysql> select * from transactionlevel;
+-----+
| i     |
+-----+
|    2 |
+-----+
1 row in set (0,00 sec)
```

## Table Lock vs Row Lock

Avantages du verrouillage au niveau de la ligne:

- Moins de conflits de verrous lors de l'accès à différentes lignes dans de nombreux threads.
- Moins de changements pour les annulations.
- Permet de verrouiller une seule ligne pendant une longue période.

Inconvénients du verrouillage au niveau de la ligne:

- Prend plus de mémoire que les verrous au niveau de la page ou de la table.
- Est plus lent que les verrous au niveau de la page ou de la table lorsqu'il est utilisé sur une grande partie de la table car vous devez acquérir beaucoup plus de verrous.

Avec les verrous de niveau supérieur, vous pouvez également prendre en charge plus facilement des verrous de différents types pour optimiser l'application, car la charge de verrouillage est inférieure à celle des verrous de niveau ligne.

Les verrous de table sont supérieurs aux verrous de page ou de ligne dans les cas suivants:

- La plupart des instructions pour la table sont des lectures.

# Table Lock

## Read Lock

Un verrou READ présente les caractéristiques suivantes:

- Un verrou READ pour une table peut être acquis par plusieurs sessions simultanément. De plus, d'autres sessions peuvent lire des données de la table sans acquérir le verrou.
- La session qui détient le verrou READ peut uniquement lire les données de la table, mais ne peut pas écrire. De plus, les autres sessions ne peuvent pas écrire de données dans la table tant que le verrou READ n'est pas libéré. Les opérations d'écriture d'une autre session seront mises dans les états en attente jusqu'à ce que le verrou READ soit libéré.
- Si la session est terminée, normalement ou de manière anormale, MySQL libérera tous les verrous implicitement. Cette fonctionnalité est également pertinente pour le verrou WRITE.

| Connection 1  | Connection 2   |
|---|--|
| <pre>mysql&gt; create table t (i integer); Query OK, 0 rows affected (0.08 sec)  mysql&gt; lock table t read; Query OK, 0 rows affected (0.00 sec)  mysql&gt; insert into t values(3); ERROR 1099 (HY000): Table 't' was locked with a READ lock and can't be updated</pre> | <pre>mysql&gt; insert into t values(3); -- la connexion est bloqué</pre> |

## Write Lock

Un verrou WRITE présente les caractéristiques suivantes:

- La seule session qui détient le verrou d'une table peut lire et écrire des données à partir de la table.
- D'autres sessions ne peuvent pas lire et écrire des données dans la table tant que le verrou WRITE n'est pas libéré.

| Connection 1  | Connection 2  |
|---|---|
| <pre>mysql&gt; lock table testtransaction write; Query OK, 0 rows affected (0,00 sec)  mysql&gt; insert into testtransaction values(1,2); Query OK, 1 row affected (0,00 sec)</pre> | <pre>select * from testtransaction; -- la connection 2 est bloqué</pre> |

## Statut des lock

Il est possible d'avoir le statut des connection via SHOW PROCESS LIST

```
***** 4. row *****
  Id: 10
  User: root
  Host: localhost:33736
  db: transaction
Command: Query
  Time: 28
  State: Waiting for table metadata lock
  Info: select * from testtransaction
4 rows in set (0,00 sec)
```

## Kill

Il est parfois nécessaire de tuer une transaction afin de libérer les lock.

Dans ce cas, on doit récupérer la liste des process et executer une requete KILL

```
SHOW PROCESSLIST \G;
***** 1. row *****
```

```
    Id: 4
    User: event_scheduler
    Host: localhost
    db: NULL
Command: Daemon
    Time: 16478
    State: Waiting on empty queue
    Info: NULL
***** 2. row *****
    Id: 8
    User: root
    Host: localhost: 33716
    db: myisamtest
Command: Sleep
    Time: 5193
    State:
    Info: NULL
***** 3. row *****
    Id: 9
    User: root
    Host: localhost: 33734
    db: transaction
Command: Sleep
    Time: 5
    State:
    Info: NULL
***** 4. row *****
    Id: 11
    User: root
    Host: localhost: 33738
    db: transaction
Command: Query
    Time: 0
    State: starting
    Info: SHOW PROCESSLIST
4 rows in set (0,00 sec)
```

ERROR:

No query specified

```
mysql> kill 9
-> ;
Query OK, 0 rows affected (0,00 sec)
```

KILL autorise un modificateur CONNECTION ou QUERY:

- KILL CONNECTION est identique à KILL sans modificateur: il met fin à la connexion associée à processlist\_id donné, après avoir terminé toute instruction en cours d'exécution par la connexion.
- KILL QUERY met fin à l'instruction en cours d'exécution de la connexion, mais la laisse elle-même intacte.

Si vous avez le privilège PROCESSUS, vous pouvez voir tous les threads. Si vous disposez du privilège CONNECTION\_ADMIN ou SUPER, vous pouvez supprimer tous les threads et toutes les instructions. Sinon, vous ne pouvez voir et tuer que vos propres discussions et déclarations.

## Row Lock

Si vous interrogez des données puis insérez ou mettez à jour des données associées dans la même transaction, l'instruction SELECT standard ne donne pas une protection suffisante. D'autres transactions peuvent mettre à jour ou supprimer les mêmes lignes que celles que vous venez de consulter. InnoDB prend en charge deux types de lectures de verrouillage offrant une sécurité supplémentaire:

## SELECT... FOR SHARE

Définit un verrou de mode partagé sur toutes les lignes lues. D'autres sessions peuvent lire les lignes, mais ne peuvent pas les modifier jusqu'à ce que votre transaction soit validée. Si l'une de ces lignes a été modifiée par une autre transaction qui n'a pas encore été validée, votre requête attend jusqu'à la fin de cette transaction, puis utilise les dernières valeurs.

|              |              |
|--------------|--------------|
| Connection 1 | Connection 2 |
|--------------|--------------|

```
mysql> begin ;
Query OK, 0 rows affected (0,00 sec)
```

```
mysql> select * from testtransaction where
i=2 FOR SHARE;
```

```
+-----+-----+
| i     | testcolumn |
+-----+-----+
| 2    | NULL      |
+-----+-----+
1 row in set (0,00 sec)
```

```
mysql> insert into testtransaction
values(1,2);
Query OK, 1 row affected (0,00 sec)
```

```
mysql> begin;
Query OK, 0 rows affected (0,02 sec)
```

```
mysql> update testtransaction set
testcolumn='pilou';
-- la transaction est bloqué
ERROR 1205 (HY000): Lock wait timeout
exceeded;
try restarting transaction
```

seule les lignes impacté par le select sont locké

Connection 1

```
mysql> select * from testtransaction where
i=2 FOR SHARE;
```

```
+-----+-----+
| i     | testcolumn |
+-----+-----+
| 2    | NULL      |
+-----+-----+
1 row in set (0,00 sec)
```

Connection 2

```
mysql> begin;
Query OK, 0 rows affected (0,02 sec)
```

```
mysql> update testtransaction set
testcolumn=45 where i=1;
Query OK, 2 rows affected (0,00 sec)
Rows matched: 2 Changed: 2 Warnings: 0
```

## SELECT ... FOR UPDATE

Une commande SELECT ... FOR UPDATE va lire les dernières données disponibles pour chaque ligne, et pose un verrou dessus en même tant qu'il lit. De cette façon, il pose le même verrou que la commande UPDATE .

Connection 1

Connection 2

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

mysql> select * from SolrCoresPreallocated
order by id limit 1 for update;
+----+-----+----+-----+
| id | used_status | sid | cid |
+----+-----+----+-----+
| 1 |          0 |  0 | 400 |
+----+-----+----+-----+
1 row in set (0.00 sec)
```

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

mysql> select * from SolrCoresPreallocated
order by id limit 1 for update;
ERROR 1205 (HY000): Lock wait timeout
exceeded; try restarting transaction
mysql> rollback;
Query OK, 0 rows affected (0.00 sec)
```

## NOWAIT

Si une ligne est verrouillée par une transaction, une transaction `SELECT ... FOR UPDATE` ou `SELECT ... FOR SHARE` demandant la même ligne verrouillée doit attendre que la transaction bloquante libère le verrou de ligne. Ce comportement empêche les transactions de mettre à jour ou de supprimer les lignes qui sont interrogées pour des mises à jour par d'autres transactions. Cependant, il n'est pas nécessaire d'attendre qu'un verrou de ligne soit libéré si vous souhaitez que la requête soit renvoyée immédiatement lorsqu'une ligne demandée est verrouillée, ou si le fait d'exclure des lignes verrouillées du jeu de résultats est acceptable.

Pour éviter d'attendre que d'autres transactions libèrent des verrous de ligne, les options `NOWAIT` et `SKIP LOCKED` peuvent être utilisées avec les instructions de lecture `SELECT ... FOR UPDATE` ou `SELECT ... FOR SHARE`.

## NOWAIT

Une lecture de verrouillage qui utilise `NOWAIT` n'attend jamais d'acquiescer un verrou de ligne. La requête s'exécute immédiatement, échouant avec une erreur si une ligne demandée est verrouillée.

## SKIP LOCKED

Une lecture de verrouillage qui utilise `SKIP LOCKED` n'attend jamais d'obtenir un verrou de ligne. La requête s'exécute immédiatement, en supprimant les lignes verrouillées du jeu de résultats.

# Administration d'un base de donnée

## Création de Base de données

CREATE DATABASE où comment créer une nouvelle base de données dans MySQL Server. Pour créer une nouvelle base de données dans MySQL, utilisez l'instruction CREATE DATABASE avec la syntaxe suivante:

```
<source lang='bash'>
```

```
CREATE DATABASE [ IF NOT EXISTS] nom_base[ CHARACTER SET charset][ COLLATE collation]
```

Tout d'abord, vous spécifiez le nom de la base de données après la clause CREATE DATABASE. Le nom de la base de données doit être unique dans l'instance du serveur MySQL. Si vous essayez de créer une base de données avec un nom qui existe déjà, MySQL génère une erreur.

```
mysql> create database testdb;
Query OK, 1 row affected (0,08 sec)

mysql> create database testdb;
ERROR 1007 (HY000): Can't create database 'testdb'; database exists
mysql>
```

Deuxièmement, pour éviter une erreur en cas de création accidentelle d'une base de données existante, vous pouvez spécifier l'option IF NOT EXISTS. Dans ce cas, MySQL ne génère pas d'erreur, mais termine l'instruction CREATE DATABASE.

Troisièmement, vous pouvez spécifier le jeu de caractères et le classement de la nouvelle base de données au moment de la création. Si vous omettez les clauses CHARACTER SET et COLLATE, MySQL utilise le jeu de caractères et le classement par défaut de la nouvelle base de données.

```
mysql> create database testUtf8 CHARACTER SET utf8 COLLATE utf8_unicode_ci;
Query OK, 1 row affected, 2 warnings (0,05 sec)
```

Le jeu de caractères et la collation de la base de données affectent les aspects suivants du fonctionnement du serveur:

- Pour les instructions CREATE TABLE, le jeu de caractères de la base de données et le classement sont utilisés comme valeurs par défaut pour les définitions de table si le jeu de caractères de la table et le classement ne sont pas spécifiés.
- Pour les instructions LOAD DATA qui n'incluent pas de clause CHARACTER SET, le serveur utilise le jeu de caractères indiqué par la variable système character\_set\_database pour interpréter les informations contenues dans le fichier.
- Pour les routines stockées (procédures et fonctions), le jeu de caractères de la base de données et le classement en vigueur au moment de la création de la routine sont utilisés comme jeu de caractères et le classement des paramètres de données de caractères pour lesquels la déclaration ne comprend aucun CHARACTER SET ni aucun attribut COLLATE. Pour remplacer cela, indiquez explicitement CHARACTER SET et COLLATE.

```
mysql> SHOW CHARACTER SET ;
+-----+-----+-----+-----+
| Charset | Description | Default collation | Maxlen |
+-----+-----+-----+-----+
| armSCII8 | ARMSCII-8 Armenian | armSCII8_general_ci | 1 |
| ascii | US ASCII | ascii_general_ci | 1 |
| big5 | Big5 Traditional Chinese | big5_chinese_ci | 2 |
| binary | Binary pseudo charset | binary | 1 |
| cp1250 | Windows Central European | cp1250_general_ci | 1 |
| cp1251 | Windows Cyrillic | cp1251_general_ci | 1 |
| cp1256 | Windows Arabic | cp1256_general_ci | 1 |
| cp1257 | Windows Baltic | cp1257_general_ci | 1 |
| cp850 | DOS West European | cp850_general_ci | 1 |
| cp852 | DOS Central European | cp852_general_ci | 1 |
| cp866 | DOS Russian | cp866_general_ci | 1 |
| cp932 | SJIS for Windows Japanese | cp932_japanese_ci | 2 |
| dec8 | DEC West European | dec8_swedish_ci | 1 |
| eucjpms | UJIS for Windows Japanese | eucjpms_japanese_ci | 3 |
| euckr | EUC-KR Korean | euckr_korean_ci | 2 |
| gb18030 | China National Standard GB18030 | gb18030_chinese_ci | 4 |
| gb2312 | GB2312 Simplified Chinese | gb2312_chinese_ci | 2 |
| gbk | GBK Simplified Chinese | gbk_chinese_ci | 2 |
| geostd8 | GEOSTD8 Georgian | geostd8_general_ci | 1 |
| greek | ISO 8859-7 Greek | greek_general_ci | 1 |
| hebrew | ISO 8859-8 Hebrew | hebrew_general_ci | 1 |
```

```

| hp8 | HP West European | hp8_english_ci | 1 |
| keybcs2 | DOS Kamenicky Czech-Slovak | keybcs2_general_ci | 1 |
| koi8r | KOI8-R Relcom Russian | koi8r_general_ci | 1 |
| koi8u | KOI8-U Ukrainian | koi8u_general_ci | 1 |
| latin1 | cp1252 West European | latin1_swedish_ci | 1 |
| latin2 | ISO 8859-2 Central European | latin2_general_ci | 1 |
| latin5 | ISO 8859-9 Turkish | latin5_turkish_ci | 1 |
| latin7 | ISO 8859-13 Baltic | latin7_general_ci | 1 |
| macce | Mac Central European | macce_general_ci | 1 |
| macroman | Mac West European | macroman_general_ci | 1 |
| sjis | Shift-JIS Japanese | sjis_japanese_ci | 2 |
| swe7 | 7bit Swedish | swe7_swedish_ci | 1 |
| tis620 | TIS620 Thai | tis620_thai_ci | 1 |
| ucs2 | UCS-2 Unicode | ucs2_general_ci | 2 |
| ujis | EUC-JP Japanese | ujis_japanese_ci | 3 |
| utf16 | UTF-16 Unicode | utf16_general_ci | 4 |
| utf16le | UTF-16LE Unicode | utf16le_general_ci | 4 |
| utf32 | UTF-32 Unicode | utf32_general_ci | 4 |
| utf8 | UTF-8 Unicode | utf8_general_ci | 3 |
| utf8mb4 | UTF-8 Unicode | utf8mb4_0900_ai_ci | 4 |
+-----+-----+-----+-----+
41 rows in set (0,00 sec)

```

Il est possible de savoir pour une base de donnée le jeux de données et la collation utilisé:

```

mysql> SELECT @@character_set_database, @@collation_database;
+-----+-----+
| @@character_set_database | @@collation_database |
+-----+-----+
| utf8mb4 | utf8mb4_0900_ai_ci |
+-----+-----+
1 row in set (0,00 sec)

```

```

show databases;
+-----+
| Database |
+-----+
| information_schema |

```

```
| mysql |
| performance_schema |
| sys |
| testUtf8 |
| testdb |
+-----+
6 rows in set (0,00 sec)
```

Enfin, pour accéder à la base de données nouvellement créée, vous utilisez la commande USE database comme suit:

```
mysql> USE testdb;
```

# Utilisation d'une base de données

Si USE n'a jamais été utilisé, la requete suivante renverras NULL

```
select database();
+-----+
| database() |
+-----+
| NULL |
+-----+
1 row in set (0,00 sec)
```

Après l'utilisation de USE

```
mysql> use testdb;
Database changed
mysql> select database();
+-----+
| database() |
+-----+
```

```
| testdb |
+-----+
1 row in set (0,00 sec)
```

# Storage Engine

```
show engines;
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+
+-----+
| Engine | Support | Comment | Transactions | XA | Savepoints |
+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+
+-----+
| FEDERATED | NO | Federated MySQL storage engine | NULL | NULL | NULL |
| InnoDB | DEFAULT | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |
| PERFORMANCE_SCHEMA | YES | Performance Schema | NO | NO | NO |
| MyISAM | YES | MyISAM storage engine | NO | NO | NO |
| MRG_MYISAM | YES | Collection of identical MyISAM tables | NO | NO | NO |
| BLACKHOLE | YES | /dev/null storage engine (anything you write to it disappears) | NO | NO | NO |
| MEMORY | YES | Hash based, stored in memory, useful for temporary tables | NO | NO | NO |
| CSV | YES | CSV storage engine | NO | NO | NO |
| ARCHIVE | YES | Archive storage engine | NO | NO | NO |
+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+
+-----+
9 rows in set (0,00 sec)
```

Lors de la création de table, il est possible d'utiliser un des storage engine (ici CSV)

```
CREATE TABLE t2 (i INT NOT NULL) ENGINE = CSV;
Query OK, 0 rows affected (0,08 sec)
```

```
mysql> insert into t2 values(2);  
Query OK, 1 row affected (0,03 sec)
```

Dans le datadir de mysql, on va trouver un repertoire correspondant à la base de données ainsi que le fichier de metadata et de données de la table en CSV:

```
pilou@lubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/data/testdb$ ls  
t2_336.sdi t2.CSM t2.CSV  
pilou@lubuntu: ~/mysql80/mysql-8.0.13-linux-glibc2.12-x86_64/data/testdb$ more t2.CSV  
2
```

En plus de stocker les métadonnées relatives aux objets de base de données dans le dictionnaire de données, MySQL 8 les stocke sous forme sérialisée. Ces données sont appelées informations de dictionnaire sérialisées (SDI). InnoDB stocke les données SDI dans ses fichiers de tablespace. D'autres moteurs de stockage stockent les données SDI dans des fichiers .sdi créés dans le répertoire du schéma. Les données SDI sont générées dans un format JSON compact.

Il est possible de positionner une table vers une autre storage engine:

```
alter table t2 engine INNODB;  
Query OK, 1 row affected (0,17 sec)  
Records: 1 Duplicates: 0 Warnings: 0
```

En général, les storage engines se doivent de vérifier la transformation des tables:

```
mysql> create table t3 (i INT) ;  
Query OK, 0 rows affected (0,10 sec)  
  
mysql> alter table t3 engine CSV;  
ERROR 1178 (42000): The storage engine for the table doesn't support nullable columns
```

## Renommer une table

Il est possible de renommer une table via la commande RENAME : RENAME ancien\_nom TO nouveau\_nom

```
rename table t3 to t4;  
Query OK, 0 rows affected (0,14 sec)
```

Mais le renommage doit être fait en faisant attention au dépendance

```
create table idtable ( id INT auto_increment, PRIMARY KEY (id));  
Query OK, 0 rows affected (0,11 sec)  
  
mysql> insert into idtable values(0);  
Query OK, 1 row affected (0,09 sec)  
  
mysql> create view testview as select * from idtable;  
Query OK, 0 rows affected (0,09 sec)  
  
mysql> rename table idtable to myidtables;  
Query OK, 0 rows affected (0,11 sec)  
  
mysql> select * from testview;  
ERROR 1356 (HY000): View 'testdb.testview' references invalid table(s) or column(s) or  
function(s) or definer/invoker of view lack rights to use them
```

# Administration des tables

## Optimiser les tables MySQL et défragmenter pour récupérer de l'espace

Si votre application effectue beaucoup de suppressions et de mises à jour sur la base de données MySQL, il est très probable que vos fichiers de données MySQL soient fragmentés.

Cela entraînera beaucoup d'espace inutilisé et pourrait également affecter les performances.

Il est donc fortement recommandé de défragmenter vos tables MySQL de manière continue.

## Mise en place de la table de test

Il est intéressant de se doter d'une table de test

```
create table randomint(value integer);
INSERT INTO randomint ( value ) VALUES ( rand() * 3333 );

-- repeter l'operation suivante plusieurs fois

INSERT INTO randomint ( value ) SELECT value * rand() FROM randomint;
```

## Requete utiles

La requête suivante permet de lister toutes les bases de données et de calculer la taille de chacune d'elles en Mo.

```
SELECT table_schema AS NomBaseDeDonnees, ROUND(SUM( data_length + index_length ) / 1024 / 1024, 2) AS BaseDonneesMo
FROM information_schema.TABLES GROUP BY TABLE_SCHEMA;
```

Il est également possible de visualiser en détail la taille d'une base de données spécifiquement. La requête ci-dessous présente la taille d'une base en Mo, en Ko et aussi en octets. Il faut juste adapter la requête en remplaçant "nom\_base\_de\_donnees" par la base de votre choix.

```
SELECT CONCAT( SUM(ROUND( ( (DATA_LENGTH + INDEX_LENGTH - DATA_FREE) / 1024 / 1024),2)), 'Mo' ) AS TailleMo,
CONCAT( SUM(ROUND( ( (DATA_LENGTH + INDEX_LENGTH - DATA_FREE) / 1024 ),2)), 'Ko' ) AS TailleKo,
CONCAT( SUM(ROUND( ( (DATA_LENGTH + INDEX_LENGTH - DATA_FREE) ),2)), 'o' ) AS Tailleo
FROM information_schema.TABLES
WHERE TABLE_SCHEMA = 'nom_base_de_donnees';
```

Pour connaître la taille de chaque table inclus dans une base il est possible d'exécuter la requête ci-dessous :

```
SELECT TABLE_NAME,
CONCAT(ROUND(((DATA_LENGTH + INDEX_LENGTH - DATA_FREE) / 1024 / 1024), 2), 'Mo') AS TailleMo
FROM information_schema.TABLES
WHERE TABLE_SCHEMA = 'nom_base_de_donnees'
```

Pour connaître uniquement la taille d'une table SQL il est possible d'exécuter la requête suivante :

```
SELECT
CONCAT(ROUND(((DATA_LENGTH + INDEX_LENGTH - DATA_FREE) / 1024 / 1024), 2), 'Mo') AS TailleMo
FROM information_schema.TABLES
WHERE TABLE_SCHEMA = 'nom_base_de_donnees'
AND TABLE_NAME = 'nom_table';
```

## data\_length et data\_free

l'instruction suivante permet d'afficher la data\_length et data\_free

```
mysql> select table_name,data_length,data_free from information_schema.tables where table_name
like 'randomint';
+-----+-----+-----+
| TABLE_NAME | DATA_LENGTH | DATA_FREE |
+-----+-----+-----+
| randomint | 1589248 | 4194304 |
+-----+-----+-----+

1 row in set (0,00 sec)
```

ou alors:

```
mysql> show table status \G
***** 1. row *****
Name: randomint
Engine: InnoDB
Version: 10
Row_format: Dynamic
Rows: 32960
Avg_row_length: 48
Data_length: 1589248
Max_data_length: 0
Index_length: 0
Data_free: 2097152
Auto_increment: NULL
Create_time: 2019-01-14 11:34:49
Update_time: NULL
Check_time: NULL
Collation: utf8mb4_0900_ai_ci
Checksum: NULL
Create_options:
Comment:

1 row in set (0,00 sec)
```

## DATA\_LENGTH

Pour MyISAM, DATA\_LENGTH est la longueur du fichier de données, en octets.

Pour InnoDB, DATA\_LENGTH est la quantité approximative de mémoire allouée pour l'index clusterisé, en octets. Plus précisément, il s'agit de la taille de l'index clusterisé, exprimée en pages, multipliée par la taille de la page InnoDB.

## MAX\_DATA\_LENGTH

Pour MyISAM, MAX\_DATA\_LENGTH est la longueur maximale du fichier de données. Il s'agit du nombre total d'octets de données pouvant être stockés dans la table, en fonction de la taille du pointeur de données utilisé.

Non utilisé pour InnoDB.

## INDEX\_LENGTH

Pour MyISAM, INDEX\_LENGTH est la longueur du fichier d'index, en octets.

Pour InnoDB, INDEX\_LENGTH est la quantité approximative de mémoire allouée pour les index non clusterisés, en octets. Plus précisément, il s'agit de la somme des tailles d'index non clusterisés, en pages, multipliée par la taille de page InnoDB.

## DATA\_FREE

Nombre d'octets alloués mais non utilisés.

Les tables InnoDB indiquent l'espace libre de l'espace de table auquel la table appartient. Pour une table située dans l'espace de table partagé, il s'agit de l'espace libre de l'espace de table partagé. Si vous utilisez plusieurs espaces de table et que la table possède son propre espace de table, l'espace disponible ne concerne que cette table. Espace libre signifie le nombre d'octets dans des étendues complètement libres moins une marge de sécurité. Même si l'espace libre affiche 0, il est possible d'insérer des lignes tant qu'il n'est pas nécessaire d'attribuer de nouvelles extensions.

DATA\_FREE indique l'espace alloué sur le disque pour un tableau ou un fragment de données de disque, mais n'est pas utilisé par ce dernier. (L'utilisation des ressources de données en mémoire est signalée par la colonne DATA\_LENGTH.)

La taille du fichier est

```
-rw-r----- 1 pilou pilou 7340032 janv. 14 11:34 randomint.ibd
```

**Le Data\_free est difficile à prédire.**

- Si la table a été créée avec `innodb_file_per_table = 0`, il s'agit de l'espace disponible dans `ibdata1`.
- Si la table n'est pas partitionnée et qu'elle est minuscule, elle est généralement nulle.
- Non partitionné et d'une taille d'au moins mégaoctets, `Data_free` fait habituellement exactement 4 Mo, 5 Mo, 6 Mo ou 7 Mo. En effet, InnoDB obtient une "extension" de 8 Mo en anticipant le besoin de davantage d'espace.

Pour les tables partitionnées (avant les "partitions natives" de 5.7.xx), chaque `PARTITION` ressemble à une table et se comporte comme une table. Ainsi, le `Data_free` pour la table est vraiment la somme des 4-7 Mo pour chaque partition.

La table `INFORMATION_SCHEMA.TABLES` contient environ 20 colonnes, mais pour déterminer la quantité d'espace disque utilisée par les tables, nous nous intéresserons plus particulièrement à deux colonnes: `DATA_LENGTH` et `INDEX_LENGTH`.

- `DATA_LENGTH` est la longueur (ou la taille) de toutes les données de la table (en octets).
- `INDEX_LENGTH` est la longueur (ou la taille) du fichier d'index de la table (également en octets).

Armés de ces informations, nous pouvons exécuter une requête qui listera toutes les tables d'une base de données spécifique ainsi que l'espace disque (taille) de chacune. Nous pouvons même devenir un peu plus sophistiqués et convertir les valeurs de taille normale d'octets en quelque chose de plus utile et compréhensible pour la plupart des gens, comme les mégaoctets.

```
mysql> SELECT TABLE_NAME AS 'Table', ROUND((DATA_LENGTH + INDEX_LENGTH) / 1024 / 1024) AS
'Size (MB)'
FROM information_schema.TABLES WHERE TABLE_name LIKE "randomint" ORDER BY (DATA_LENGTH +
INDEX_LENGTH) ;
```

```
+-----+-----+
| Table | Size (MB) |
+-----+-----+
| randomint | 2 |
+-----+-----+
```

```
1 row in set (0,00 sec)
```

## ANALYZE TABLE





Checksum: NULL

Create\_options:

Comment:

1 row in set (0,00 sec)

Ce qui ne change pas grand chose dans MySQL, mais sur le disque

```
-rw-r----- 1 pilou pilou 114688 janv. 14 12:00 randomint.ibd
```

Pour rafraichir les statistiques, il faut faire un analyze table

```
mysql> analyze table randomint;
```

```
+-----+-----+-----+-----+
| Table                | Op      | Msg_type | Msg_text |
+-----+-----+-----+-----+
| myisamtest.randomint | analyze | status   | OK      |
+-----+-----+-----+-----+
```

1 row in set (0,09 sec)

□

```
mysql> show table status \G
```

```
***** 1. row *****
```

Name: randomint

Engine: InnoDB

Version: 10

Row\_format: Dynamic

Rows: 0

Avg\_row\_length: 0

Data\_length: 16384

Max\_data\_length: 0

Index\_length: 0

Data\_free: 0

Auto\_increment: NULL

Create\_time: 2019-01-14 12:00:11

```
Update_time: NULL
Check_time: NULL
Collation: utf8mb4_0900_ai_ci
Checksum: NULL
Create_options:
Comment:
```

```
1 row in set (0,00 sec)
```

# Format des tables

Afin de voir les différents format, nous allons créer une table de champs TEXT

```
mysql> create table compacttable (c1 LONGTEXT) row_format=compact;
Query OK, 0 rows affected (0,06 sec)

mysql> insert into compacttable values (REPEAT("0123456789", 10000));
Query OK, 1 row affected (0,10 sec)

mysql> select length(REPEAT("0123456789", 10000));
+-----+
| length(REPEAT("0123456789", 10000)) |
+-----+
|                               100000 |
+-----+
```

1 row in set (0,00 sec)

# Statut des tables

Il est possible d'avoir l'état d'une table ainsi

```
mysql> SHOW TABLE STATUS IN test1\G
```

```
***** 1. row *****
      Name: t1
      Engine: InnoDB
      Version: 10
      Row_format: Dynamic
      Rows: 0
      Avg_row_length: 0
      Data_length: 16384
      Max_data_length: 0
      Index_length: 16384
      Data_free: 0
      Auto_increment: 1
      Create_time: 2016-09-14 16:29:38
      Update_time: NULL
      Check_time: NULL
      Collation: latin1_swedish_ci
      Checksum: NULL
      Create_options:
      Comment:
```

## Erreur possible

Le format de ligne par défaut pour la table InnoDB est défini par la variable `innodb_default_row_format`. La valeur par défaut est différente dans MySQL 5.6 et 5.7. Si vous êtes maintenant conscient de cette différence et que vous utilisez la valeur par défaut de MySQL 5.6, vous pouvez rencontrer des problèmes tels que:

| Row Format                              | REDUNDANT                        | COMPACT                          | DYNAMIC                  | COMPRESSED               |
|---|----------------------------------|----------------------------------|--------------------------|--------------------------|
| Compact Storage Characteristics         | No                               | Yes                              | Yes                      | Yes                      |
| Enhanced Variable-Length Column Storage | No                               | No                               | Yes                      | Yes                      |
| Large Index Key Prefix Support          | No                               | No                               | Yes                      | Yes                      |
| Compression Support                     | No                               | No                               | No                       | Yes                      |
| Supported Tablespace Types              | system, file-per table, general* | system, file-per-table, general* | file-per-table, general* | file-per-table, general* |

| Row Format           | REDUNDANT             | COMPACT               | DYNAMIC   | COMPRESSED |
|----------------------|-----------------------|-----------------------|-----------|------------|
| Required File Format | Antelope or Barracuda | Antelope or Barracuda | Barracuda | Barracuda  |

## Compact

Compact est le format par défaut et est généralement approprié pour le format de fichier Antelope. Introduit dans MySQL 5.0.

Dans ce format (comme dans Redundant) les colonnes BLOB et TEXT sont partiellement stockées dans les pages de données. Au moins 767 octets sont stockés dans la ligne; tous les débords sont stockés dans des pages dédiées. Comme les tailles de lignes de Compact et Redundant sont d'environ 8000 octets, cela limite le nombre de colonnes BLOB ou TEXT qui peuvent être utilisées dans une table. Chaque page de BLOB page contient 16Ko, sans compter les données.

Les autres colonnes peuvent être stockées dans des pages différentes si elles dépassent la taille de ligne maximum par page.

## Redundant

Redundant est l'ancien, format non compressé supporté par les anciennes versions de MySQL. C'était le seul format disponible avant la version 5.0 et a été le mode par défaut dans MySQL 5.0.3. Il est recommandé de paramétrer `innodb_strict_mode` lorsque vous l'utilisez.

## Dynamic

Les tables en format Dynamic contiennent des enregistrements de taille variable, ce qui permet de mieux exploiter l'espace disque que Compact ou Redundant, notamment pour les tables contenant des BLOBs, mais moins que le format Compressed.

Il ne peut être utilisé qu'avec le format de fichier XtraDB Barracuda, et nécessite que les tables et index soient stockés dans leur propre tablespace, ce qui implique que les variables systèmes soient paramétrées `innodb_file_per_table=1` et `innodb_file_format=barracuda`. Il est recommandé de paramétrer `innodb_strict_mode` lors de l'utilisation de ce format.

La taille maximum des lignes est désormais de 65535 octets.

Avec le format Dynamic (et Compressed), les colonnes BLOB et TEXT sont stockées différemment de Compact. Si les données ne peuvent être contenues dans une ligne de page, alors seulement un pointeur sera stocké et contiendra l'adresse de la page dédiée. Chaque page externe contient une partie des données et l'adresse de la page suivante, si nécessaire. Les pointeurs ont une taille de 20Ko. Cela permet de stocker un grand nombre de colonnes BLOB ou de TEXT dans une table.

# Compressed

Le format Compressed permet de réduire la taille des données. Il ne peut être utilisé qu'avec le format de fichier XtraDB Barracuda, et nécessite que les tables et index soient stockés dans leur propre tablespace, ce qui implique que les variables systèmes soient paramétrées `innodb_file_per_table=1` et `innodb_file_format=barracuda`. Il est recommandé de paramétrer `innodb_strict_mode` lors de l'utilisation de ce format.

Le fait d'utiliser le format Compressed réduit aussi le `KEY_BLOCK_SIZE` par défaut. Si `KEY_BLOCK_SIZE` est omis de la clause `CREATE TABLE` ou `ALTER TABLE`, il sera par défaut à 8Ko - traditionnellement c'est 16Ko (cf. `innodb_page_size`). Il est aussi possible de régler le `KEY_BLOCK_SIZE` à 1Ko, 2Ko, 4Ko ou 16Ko. Le fait de le régler à 16Ko, la taille standard, provoquera généralement une compression minimale à moins qu'il n'y ait beaucoup de colonnes de type BLOB, TEXT ou VARCHAR.

Notez que préciser un `KEY_BLOCK_SIZE` spécifique dans une définition de table provoquera automatiquement une compression - Il n'est donc pas nécessaire de préciser l'option `ROW_FORMAT=COMPRESSED`.

Afin d'éviter trop de compression/décompression de pages, XtraDB/InnoDB essaye de garder les pages compressées et décompressées dans le buffer pool, quand il y a assez d'espace. Cela donne un cache plus gros. Lorsque la place vient à manquer, un algorithme de LRU vient décider de la suppression de pages compressées ou décompressées du buffer: pour soulager les CPUs, les pages compressées sont supprimées en priorité; pour soulager les I/O, les pages non compressées sont supprimées en priorité. Bien entendu, quand cela est nécessaire, la règle s'applique aux pages compressées et non compressées.