

Journalisation

Le serveur MySQL utilise quatre types de journaux

- le journal binaire (*binary log* ou encore *binlog*)
- le journal des requêtes lentes (*slow query log*),
- le journal général (*general query log*),
- le journal des erreurs (*error log*), le seul des quatre à être activé par défaut.



Le relay-log, un autre type de journal, est créé par le serveur lors de la réplication. Il n'est est abordé dans cette section.

Le journal binaire ou *binlog* :

- ❑ Élément central de la réplication MySQL
- ❑ stocke, sous un format binaire, toutes les requêtes qui modifient les objets de la base de données (INSERT, UPDATE, DELETE, DROP, CREATE, ALTER...).
- ❑ utile à la restauration des données.

Activer le journal binaire :

- utilisez l'option log-bin

Définir le fichier Index :

- il faut paramétrer l'option log-bin-index



La journalisation binaire peut être désactivée à chaud, mais seulement pour la session d'un client, avec l'option `SQL_LOG_BIN`.

Une configuration de base pourrait être :

```
[mysqld]  
log-bin = /var/lib/mysql/mysql-bin
```

- Si `log-bin-index` n'a pas été défini,

par défaut le fichier sera créé dans le même répertoire que les journaux binaires (`/var/lib/mysql`)
- un chemin relatif est possible pour l'option `log-bin`.
 - Dans ce cas, le chemin sera relatif par rapport au répertoire de données. Pour éviter toute confusion, utilisez toujours un chemin absolu.

Il n'est cependant pas possible de l'activer à chaud.

- une fois que vous l'avez activé,
- vous avez la possibilité de le désactiver mais seulement pour la session courante,
 - sans redémarrer le serveur grâce à la variable `sql_log_bin`.

Cela peut être utile, par exemple, pour exécuter une requête de maintenance sur un serveur et ne pas vouloir la journaliser.

```
mysql> SET SESSION sql_log_bin = 'OFF';  
mysql> OPTIMIZE TABLE client;  
mysql> SET SESSION sql_log_bin = 'ON';
```

Journalisation (Journal binaire)

Le format utilisé pour le journal :

- un format binaire : les entrées du journal ne sont pas directement lisibles par un humain.
- Vous aurez donc recours à `mysqlbinlog` à chaque fois que vous aurez besoin de lire le contenu des journaux binaires, en précisant le ou les fichiers que vous souhaitez décoder :
 - **Exemple** : `$ mysqlbinlog /var/lib/mysql/mysql-bin.000001`
- Les informations du journal binaire peuvent aussi être visualisées dans le client texte mysql avec la commande `SHOW BINLOG EVENTS`.

```
[mysql> SHOW BINLOG EVENTS
```

```
[    -> ;
```

Log_name	Pos	Event_type	Server_id	End_log_pos	Info
binlog.000001	4	Format_desc	1	124	Server ver: 8.0.19, Binlog ver: 4
binlog.000001	124	Previous_gtid	1	155	
binlog.000001	155	Anonymous_Gtid	1	232	SET @@SESSION.GTID_NEXT= 'ANONYMOUS'
binlog.000001	232	Query	1	355	CREATE DATABASE menagerie /* xid=12 */

```
4 rows in set (0,07 sec)
```

Journalisation (Journal binaire)

Trois formats possibles pour le journal binaire sont possibles :

STATEMENT : le format par défaut jusqu'en MySQL 5.6,

- ☐ toujours extrêmement courant.
- ☐ Le texte des requêtes est directement transcrit dans le journal binaire,
- ☐ journal facile à interpréter par un humain.

ROW : le format par défaut à partir de MySQL 5.7.

- ☐ plus rapide et plus sûr (notamment pour la réplication,
- ☐ format à privilégier.
- ☐ Seuls les changements de chaque ligne sont conservés dans le journal binaire,
 - ☐ la requête originale est perdue.
- ☐ Le principal inconvénient de ce format est de rendre les requêtes difficilement lisibles par un humain.

MIXED : Ce format vise à concilier les avantages des deux formats précédents :

- ☐ les requêtes sont stockées au format STATEMENT par défaut
 - ☐ sauf lorsqu'elles risquent de provoquer des incohérences de données sur les esclaves de réplication, auquel cas le format ROW est automatiquement utilisé.
- ☐ ce format reste dangereux
 - ☐ car étant moins répandu,
 - ☐ il souffre de plusieurs bugs pouvant conduire à des pertes de données sur les esclaves de réplication.

Journalisation (Journal binaire)

Trois formats possibles pour le journal binaire sont possibles :

```
mysql> SHOW VARIABLES LIKE 'binlog_format' \G
***** 1. row *****
Variable_name: binlog_format
Value: STATEMENT

mysql> INSERT INTO client (client_id, nom) VALUES (uuid(), 'Orianne');
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> SHOW WARNINGS \G
***** 1. row *****
Level: Note
Code: 1592
Message: Statement is not safe to log in statement format.

mysql> SET SESSION binlog_format='ROW';

mysql> SHOW VARIABLES LIKE 'binlog_format' \G
***** 1. row *****
Variable_name: binlog_format
Value: ROW

mysql> INSERT INTO client (client_id, nom) VALUES (uuid(), 'Orianne');
Query OK, 1 row affected (0.00 sec)
```


Le journal des requêtes lentes

Le journal des requêtes lentes (slow query log) a pour principal rôle :

- d'aider l'administrateur de base de données à identifier les requêtes dont la durée d'exécution pose problème.

Son activation se réalise à chaud ou à froid :

- avec l'option `slow_query_log`,
 - toutes les requêtes dont le temps d'exécution dépasse la valeur de l'option `long_query_time` (en secondes) seront journalisées.

Il est possible de stocker les informations sur les requêtes lentes

- dans une table (`mysql.slow_log`) avec le paramètre `log_output = TABLE`.
- ou dans fichier : avec le paramètre `log_output = FILE`.
- ou dans aucun de ces deux cas

- ce mode de journalisation pouvant provoquer des pertes de performances importantes,
- il vaut mieux ne jamais l'utiliser.

Exemple de configuration

```
[mysqld]
```

```
# Activer la journalisation des requêtes lentes
```

```
slow_query_log = ON
```

```
slow_query_log_file = /var/lib/mysql/mysql-slow.log
```

```
long-query-time = 1
```

```
log_queries_not_using_indexes
```

Exemple d'entrée dans le journal

```
# Time: 2016-05-04T13:05:14.216282Z
# User@Host: root[root] @ localhost [] Id:      3
# Query_time: 0.000661  Lock_time: 0.000403 Rows_sent: 2
  Rows_examined: 2
SET timestamp=1462367114;
select * from my_app.t;
```

Le journal des requêtes lentes

La structure du fichier est finalement assez simple

Time: 090819 17:39:03. : moment et l'heure de l'insertion de la requête dans le fichier

User@Host: daz[daz] @ localhost [] : l'utilisateur qui a exécuté la requête ;

Query_time: 0.182724 : le temps d'exécution de la requête ;

Lock_time: 0.000176 : la durée pendant laquelle la ressource (la table), est verrouillée

Rows_sent: 2 : le nombre d'enregistrements renvoyés par la requête

Rows_examined: 39034 : le nombre d'enregistrements traités ;

SET timestamp=1250696343; : le timestamp où la requête a été enregistrée dans le fichier

SELECT c.Name,Language FROM City as c JOIN CountryLanguage USING(CountryCode) WHERE IsOfficial='T' ORDER BY c.name,rand() DESC LIMIT 2; : la requête lente

Le journal des requêtes lentes

La structure du fichier est finalement assez simple mais devient vite volumineuse : **mysqldumpslow**

```
shell> mysqldumpslow mysql-slow.log
Reading mysql slow query log from mysql-slow.log
Count: 10 Time=15.30s (153s) Lock=0.00s (0s) Rows=0.0 (0),
daz[daz]@daz_server.fr
UPDATE 3_msg SET etat='S', notif='S', id_int='S' WHERE id_msg=N
Count: 2 Time=12.00s (24s) Lock=0.00s (0s) Rows=0.0 (0), daz[daz]@daz_server.fr
SELECT * FROM 1_msg WHERE etat not in ('S', 'S') LIMIT N
Count: 1 Time=5.00s (5s) Lock=0.00s (0s) Rows=0.0 (0), daz[daz]@daz_server.fr
TRUNCATE 3_msg
Count: 30653 Time=4.01s (122902s) Lock=0.00s (1s) Rows=0.0 (0),
daz[daz]@daz_server.fr
SELECT * FROM 5_msg WHERE unix_timestamp(last_notif) < (N -N) and
last_notif!='S' and id_canal='S' and (etat!='S' and etat!='S')
```

Attention aux Effets de bord de l'option `log_queries_not_using_indexes`

- Votre journal peut devenir très verbeux.
- Parfois, il est moins coûteux à l'optimiseur de choisir d'analyser entièrement la table (*full table scan*) que d'utiliser un index.
- les requêtes journalisées avec l'option `log_queries_not_using_indexes`, ne seront pas forcément toutes des requêtes problématiques.
- D'autres paramètres :
 - `min_examined_row_limit` : nombre d'enregistrements minimal que doit examiner la requête pour qu'elle puisse être stockée dans le journal
 - `log-slow-admin-statements server` : enregistre uniquement les longues commandes d'administration, telles que `OPTIMIZE TABLE`, `ALTER TABLE...`

Autres outils d'analyse

D'autres outils permettent d'analyser les journaux des requêtes lentes : `mysqsla`, `mysql_slow_log_filter` et `mysql_slow_log_parser` :

<http://hackmysql.com/mysqsla>

http://www.mysqlperformanceblog.com/files/utils/mysql_slow_log_filter

http://www.mysqlperformanceblog.com/files/utils/mysql_slow_log_parser

Quel que soit l'outil utilisé, les instructions `SELECT` identifiées peuvent être optimisées avec la commande `EXPLAIN` afin de vérifier que les index sont correctement utilisés et pertinents.

Le journal des requêtes lentes : **Journaliser dans une table**

À partir de MySQL 5.1.6, toutes ces manipulations peuvent s'opérer à chaud.

```
mysql> SET GLOBAL slow_query_log = 'ON'; -- active le journal des requêtes lentes
mysql> SET GLOBAL log_output = 'TABLE';
-- sauvegarde les informations du journal des requêtes lentes dans la table mysql.slow_log
mysql> SET GLOBAL log_queries_not_using_indexes = 'OFF';
```

La structure de la table `mysql.slow_log` est très proche de celle du fichier journal des requêtes lentes.

Cependant la précision du temps est moins fine dans la table que dans le fichier
par exemple, 1,053634 seconde dans le fichier sera représenté par 00:00:01 dans la table.

Cette table n'est accessible qu'en lecture. Pour la vider, utilisez la seule commande possible ici :

`TRUNCATE TABLE.`

Le journal des erreurs (*error log*) : le seul des quatre journaux qui est activé par défaut.

- son extension en `.err`,
le préfixe étant par défaut le nom de la machine.
- Son emplacement s'obtient avec la commande `SHOW GLOBAL VARIABLES LIKE 'log_error'`.
 - Il peut être changé (le nom du fichier également),
 - en paramétrant l'option `log-error`.
- En cas de panne ou de comportement anormal, c'est l'outil privilégié du DBA en cours d'investigation.
 - Ce journal contient les informations relatives aux avertissements et erreurs du serveur MySQL.
 - Vous y trouverez aussi les messages liés aux moteurs de stockage ou aux fonctionnalités avancées comme la réplication ou le programmeur d'événements (*event scheduler*).

Le journal des erreurs

Exemple de configuration d'un journal des erreurs :

```
[mysqld]    log-error = /usr/local/mysql/logs/mysql-error.err
```

Visualisation du contenu d'un journal des erreurs :

```
$ tail mysql-error.err
121021 00:04:25 mysqld_safe Starting mysqld daemon with databases
from /home/daz/MySQL-5_6_7/data
121021 0:04:25 [Warning] TIMESTAMP with implicit DEFAULT value is
deprecated. Please use --explicit_defaults_for_timestamp server
option (see documentation for more details).
121021 0:04:25 [Note] Plugin 'FEDERATED' is disabled.
121021 0:04:25 InnoDB: The InnoDB memory heap is disabled
121021 0:04:25 InnoDB: Mutexes and rw_locks use GCC atomic builtins
121021 0:04:25 InnoDB: Compressed tables use zlib 1.2.3
121021 0:04:25 InnoDB: Using Linux native AIO
121021 0:04:25 InnoDB: CPU supports crc32 instructions
121021 0:04:25 InnoDB: Initializing buffer pool, size = 128.0M
121021 0:04:25 InnoDB: Completed initialization of buffer pool
121021 0:04:25 InnoDB: highest supported file format is Barracuda.
121021 0:04:25 InnoDB: 128 rollback segment(s) are active.
121021 0:04:25 InnoDB: Waiting for the background threads to start
121021 0:04:25 InnoDB: 1.2.7 started; log sequence number 10525020
121021 0:04:25 [Note] Server hostname (bind-address): '*'; port: 5605
121021 0:04:25 [Note] IPv6 is available.
121021 0:04:25 [Note] - '::' resolves to '::';
121021 0:04:25 [Note] Server socket created on IP: '::'.
121021 0:04:26 [Note] Event Scheduler: Loaded 0 events
121021 0:04:26 [Note] /home/daz/MySQL/bin/5.6.7/bin/mysqld: ready for
connections.
```

Arrêt et redémarrage du serveur

```
shell> cat mysql-error.err
```

```
090812 17:20:44 mysqld_safe Starting mysqld daemon with databases from /home/daz/sandboxes/msb_5_1_35/data
```

```
090812 17:20:44 InnoDB: Started; log sequence number 0 6973121
```

```
090812 17:20:44 [Note] /usr/local/mysql/5.1.35/bin/mysqld: ready for connections.
```

```
Version: '5.1.35-log' socket: '/tmp/mysql.sock' port: 3306 MySQL Community Server (GPL)
```

```
Killed
```

```
090812 17:55:07 mysqld_safe Number of processes running now: 0
```

```
090812 17:55:07 mysqld_safe mysqld restarted
```

```
090812 17:55:08 InnoDB: Started; log sequence number 0 6973121
```

```
090812 17:55:08 [Note] Recovering after a crash using /appli/log/mysql-bin
```

```
090812 17:55:08 [Note] Starting crash recovery...
```

```
090812 17:55:08 [Note] Crash recovery finished.
```

```
090812 17:55:08 [Note] /usr/local/mysql/5.1.35/bin/mysqld: ready for connections.
```

```
Version: '5.1.35-log' socket: '/tmp/mysql_sandbox5135.sock' port: 5135 MySQL Community Server (GPL)
```


Le journal général (general log) :

- permet d'enregistrer toute l'activité du serveur
- stocke les informations relatives à la connexion et à la déconnexion des clients ainsi que toutes les requêtes et toutes les commandes qui arrivent au processus mysqld, qu'elles soient valides ou non.

Le journal général (general log) enregistre les événements reçus par mysqld :

- les requêtes valides ou non envoyées par les clients
- les informations de connexion/déconnexion des clients.

Activation

- l'activer et le désactiver à chaud ou à froid avec l'option `general_log`.

Il est également possible de stocker les informations soit dans un fichier soit dans une table, comme pour le journal des requêtes lentes, la solution du fichier doit toujours être privilégiée.

Mode de stockage :

le journal général stocke les requêtes avant leur exécution.

- Par conséquent, le journal général n'a aucune des informations disponibles dans le journal des requêtes lentes. Son seul intérêt est de garder une trace des requêtes ou connexions en erreur.

Il est rare d'avoir besoin d'activer le journal général, d'autant plus qu'il peut remplir très rapidement le disque et que l'impact sur les performances est important.

Le journal général

- Commande d'activation : SET GLOBAL general_log = 'ON'
- Commande de désactivation : SET GLOBAL general_log = OFF'
- Le chemin et le nom du journal sont indiqués par la variable general_log_file
-

Exemple de journal général

```
2016-05-04T13:35:23.707140Z      4 Connect  root@localhost
on  using Socket  2016-05-04T13:35:23.707594Z      4
Query      select @@version_comment limit 1  2016-05-
04T13:35:47.578209Z      4 Query      select * from
my_app.t    2016-05-04T13:39:39.666551Z      4 Quit      2016-05-
04T13:39:50.895276Z      5 Connect  stephane@localhost  on
using Socket  2016-05-04T13:39:50.895379Z      5
Connect  Access denied for user      'stephane'@'localhost'
(using password: NO)
```

Exemple d'options à spécifier dans le script de rotation

- `daily` : la rotation doit être effectuée tous les jours
- `dateext` : au lieu d'avoir un nombre (1, 2, 3, ...), on demande d'utiliser une date (par défaut au format YYYYMMDD)
- `rotate n` : nombre de versions à conserver
- `nocompress` : on demande de ne pas compresser les versions
- `nocopytruncate` : ne pas réduire le fichier à zéro après la copie
- `ifempty` : fait la rotation même si le fichier est vide
- `nomissingok` : déclenche une erreur (non bloquante) si le fichier n'existe pas, ça peut être utile pour détecter une sauvegarde qui ne s'est pas effectuée
- `nocreate` : on ne recrée pas le fichier d'origine (/backups/mysql/all-databases.sql.gz dans mon exemple)
- `nooldir` : on ne déplace pas les versions dans un autre dossier
- `nomail` : ne pas envoyer les nouvelles versions par mail (ça risque de faire de gros messages sinon)
- `extension .sql.gz` : permet de conserver l'extension du fichier à la fin du nom (par défaut fichier.toto devient fichier.toto-YYYYMMJJ)