Un aperçu de la nouvelle sauvegarde sous MySQL 6.0

par Robin Schumacher Cédric Duprez (traducteur)

Date de publication : 01/09/2008

Dernière mise à jour :

Cet article est la traduction de *A Quick Look at MySQL 6.0's New Backup* (disponible **ici**) et a pour but de vous présenter les fonctionnalités du nouvel outil de sauvegarde et restauration de MySQL 6.0.

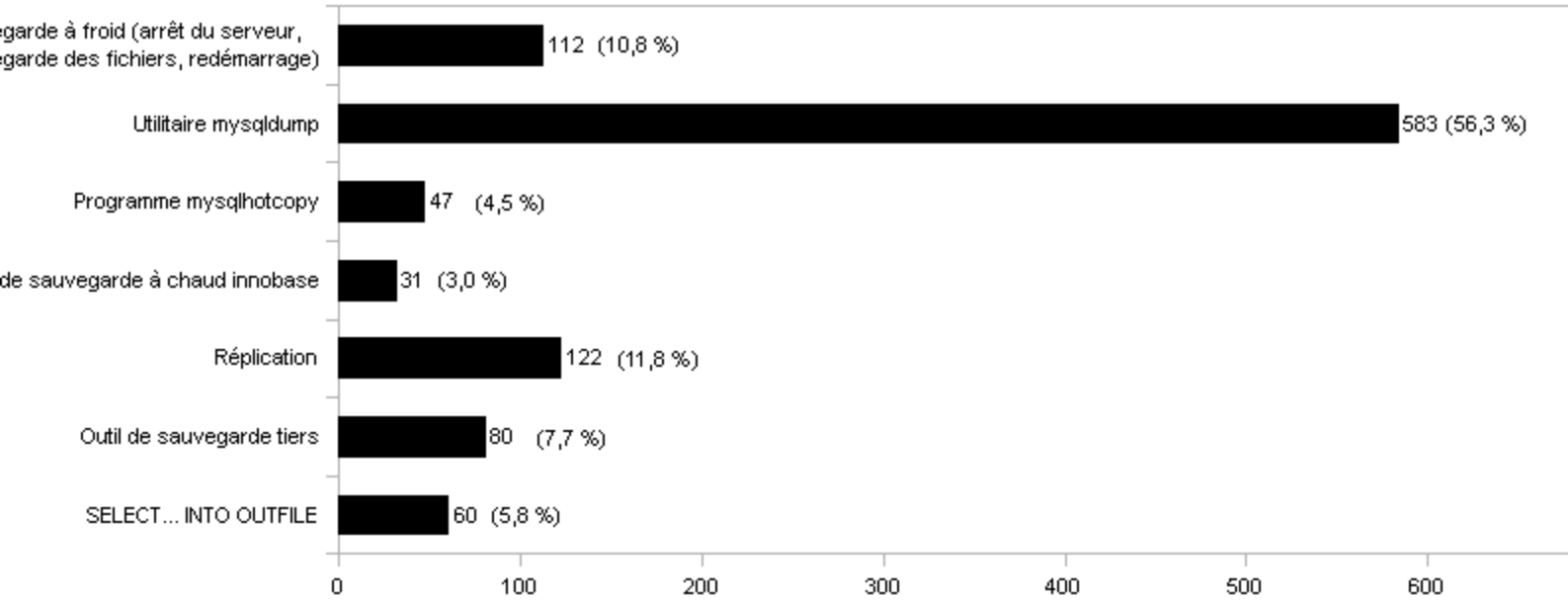
Introduction

Je me souviens encore de ce jour. Je travaillais pour un important intégrateur de systèmes chez un très grand compte, et j'étais responsable de quelques bases de données DB2. J'avais également la malheureuse obligation de prendre soin de quelques vieilles bases de données IMS (des bases de données hiérarchiques IBM utilisées il y a un bon moment), auxquelles je ne connaissais pratiquement rien. Un matin, on me demande de faire une petite modification dans une très grosse et importante base IMS (juste supprimer un enregistrement) ; que pouvait-il arriver de mal ? Le type qui connaissait vraiment bien IMS étant parti, j'ai donc écrit un programme pour effectuer la tâche (oui, il fallait bel et bien écrire un programme pour cela) et je l'ai lancé au bout d'un moment. Je me suis reconnecté le lendemain et j'ai essayé d'accéder à certains enregistrements de la base de données. Rien. Tout avait disparu, si ce n'est... ta-daaaa ! le seul enregistrement que je voulais supprimer. Dès l'instant où j'ai réalisé mon erreur, j'ai pleinement pris conscience de deux choses :

1 j'aurai de sérieux problèmes si je n'avais pas de sauvegarde ;
2 et j'avais désespérément besoin de changer de sous-vêtements.

Heureusement pour moi, j'avais une sauvegarde que j'ai restaurée (même si ça a pris du temps), et j'ai bien retenu la leçon ce jour-là : toujours mettre en oeuvre de bonnes techniques de récupération de désastres. Les administrateurs de bases de données (DBA) se concentrent le plus souvent sur des points comme l'optimisation des performances qui, certes, sont importantes ; mais en fin de compte, un DBA est payé pour faire très bien une chose : protéger les données. Ne respectez pas cette règle, et vous obtiendrez, tous frais payés, un aller simple pour l'ANPE. Cela étant dit, en tant qu'administrateur de bases de données, je me suis toujours assuré de connaître en profondeur les outils de sauvegarde et restauration de mes bases, et de pratiquer régulièrement différents scénarios de restauration sur mes serveurs de test, à partir des sauvegardes effectuées sur mes bases tous les jours.

Pour en venir à MySQL, il existe différentes méthodes à votre disposition pour sauvegarder vos bases de données, la plus populaire étant l'utilitaire *mysqldump*, comme l'a montré un de nos sondage effectué il y a quelques temps sur le site web de MySQL :



Résultat du sondage : comment sauvegardez-vous vos bases de données MySQL en production ?

Un total de 1035 votes ont été enregistrés dans ce sondage jusqu'à maintenant.

Bien que les méthodes ci-dessus remplissent correctement leur rôle de sauvegarde / restauration, il y a toujours une marge d'amélioration. Et dans MySQL 6.0 (actuellement en version alpha), un nouvel utilitaire de sauvegarde et restauration fait son apparition ; nombreux sont ceux qui y trouveront des bénéfices. Jetons un oeil à ce nouvel

outil de sauvegarde / restauration et faisons lui faire un parcours de test à travers différents scénarios usuels, pour voir comment il s'en sort.

I - Aperçu de l'utilitaire de sauvegarde et restauration de MySQL 6.0

L'utilitaire de sauvegarde / restauration de MySQL 6.0 est un tout nouvel outil du serveur, que vous devriez trouver facile à aborder et à utiliser au quotidien. Une des fonctionnalités de ce nouvel utilitaire que j'affectionne particulièrement est le fait qu'il permet d'exécuter les commandes de sauvegarde et restauration directement depuis une ligne de commande du client *mysql*, ce qui signifie qu'il n'est pas nécessaire de sortir du shell vers une ligne de commande du système d'exploitation et/ou d'écrire des scripts système pour lancer une sauvegarde. De ce point de vue, le nouvel outil de sauvegarde reflète ce qu'un DBA trouverait dans le répertoire sauvegarde/restauration avec Microsoft SQL Server (selon mon expérience, c'est, par rapport à tous les autres éditeurs de bases de données, ce que j'ai trouvé de plus simple, de plus facile à utiliser et de plus *idiot-proof*).

La syntaxe dans la version alpha actuelle se présente ainsi :

syntaxe des commandes de sauvegarde et de restauration

```
BACKUP {DATABASE | SCHEMA} { * | db_name [, db_name] ... } TO 'nom_fichier_image'
[WITH COMPRESSION [COMPRESSION_ALGORITHM [=] nom_algorithme]];

RESTORE FROM 'nom_fichier_image';
```

Il faut toutefois garder à l'esprit que, comme cette écriture, MySQL 6.0 est en version alpha et que des changements supplémentaires de syntaxe seront effectués au fur et à mesure des versions bêta. Pour voir la syntaxe la plus récente de l'outil de sauvegarde et restauration de MySQL 6.0, reportez-vous au manuel 6.0 ici : **http://dev.mysql.com/doc/refman/6.0/en/backup-database-restore.html**.

Outre une syntaxe simple à utiliser et à retenir, une des caractéristiques les plus attendues du nouvel outil de sauvegarde 6.0, c'est la possibilité de faire des sauvegardes en ligne depuis le moteur MyISAM, ce qui, jusqu'à maintenant, devait être réalisé hors ligne. Quand je dis que la sauvegarde 6.0 pour MyISAM se fait en ligne, cela signifie qu'elle n'est pas bloquante pour l'activité DML ; ainsi, toute activité d'INSERT, UPDATE et DELETE peut tourner sans encombres pendant que la sauvegarde se poursuit. Il faut toutefois noter que toute requête DDL (ALTER TABLE, etc.) sera suspendue et bloquée durant la sauvegarde.

En plus d'être en ligne pour l'activité DML, la sauvegarde MyISAM en 6.0 est également une sauvegarde « native », et non une sauvegarde de nature « logique ». Une sauvegarde native diffère d'une sauvegarde logique en ce que le fichier de sauvegarde fait à proprement parler partie du système de fichiers de l'OS sous-jacent et n'est pas un fichier SQL lisible comme l'est une sauvegarde logique. Par exemple, les fichiers de sauvegarde produits par l'utilitaire *mysqldump* sont des sauvegardes logiques. L'avantage d'une sauvegarde native, c'est qu'elle tourne plus vite et prend moins d'espace de stockage qu'une sauvegarde logique.

Au delà de sa capacité à faire des sauvegardes MyISAM en ligne (sans bloquer le DML), l'utilitaire de sauvegarde 6.0 peut faire de même avec les bases de données transactionnelles comme InnoDB, Falcon et tout autre moteur orienté transactionnel supportant les instantanés cohérents (*consistent snapshot*). Pour les utilisateurs de MySQL Cluster, notez que la nouvelle sauvegarde 6.0 ne prendra pas en compte Cluster, principalement parce que Cluster dispose déjà de sa propre sauvegarde en ligne.

Pour le moment, la différence entre MyISAM et les moteurs transactionnels réside dans le fait que les sauvegardes des moteurs transactionnels sont logiques par nature, alors que MyISAM dispose d'une sauvegarde native. Dans les versions à venir de la sauvegarde 6.0, des drivers natifs seront développés pour les moteurs transactionnels clefs de MySQL.

Pour montrer en quoi le nouvel utilitaire de sauvegarde 6.0 diffère de *mysqldump*, le tableau suivant résume les principales différences :

Sauvegarde MySQL 6.0 vs mysqldump

Caractéristiques (avantages/inconvénients)	mysqldump	Sauvegarde 6.0
Avantages		
Basé sur SQL (exécution simple, depuis un client *mysql*)		X
Commande de restauration (exécution simplifiée)		X
Pas de blocage DML pour les moteurs transactionnels	X	X
Sauvegarde au format logique (commandes SQL en lecture/écriture)	X	X
Sauvegarde sélective des bases	X	X
Supporte tous les objets clefs des bases utilisateurs (tous les objets clefs sont protégés)	X	X
Sauvegarde des schémas uniquement (pas de données, uniquement les instructions DDL)	X	
Sauvegarde sélective d'objets	X	
Pilote natif (pour MyISAM et quelques autres en version 1)		X
Pas de blocage DML pour MyISAM		X
Compression (gain de place de stockage)		X
Cryptage (caractéristique prévue en version 1)		X
Utilisable pour les bases de données > 20 Go		X
Inconvénients		
Bloque les instructions DML en MyISAM	X	
Fichiers de sauvegardes plus volumineux que souhaité (plutôt que des sauvegardes natives pour tous les moteurs)	X	X
Sauvegarde logique pour les moteurs transactionnels	X	X
Pas de sauvegarde complète d'une instance MySQL		X
Exécution plus lente qu'une sauvegarde native pour le moteur MyISAM		X

Après ce bref aperçu, passons maintenant à un rapide test de la nouvelle sauvegarde MySQL 6.0. Tous les tests qui suivent ont été réalisés sur une petite machine Linux Fedora (1 Go de RAM, un seul CPU, etc.) avec MySQL 6.0.6 ; gardez donc à l'esprit que les performances et autres auxquels vous êtes habitués dépendront de votre environnement et d'autres facteurs.

II - Sauvegarde et restauration MyISAM avec MySQL 6.0 Backup

Essayons d'abord la nouvelle sauvegarde sur une base de données de plus de 1 Go, uniquement en MyISAM, et composée des objets suivants :

Table Name	Engine	Rows	Data length	Index length
broker	MyISAM	23	812 B	3 kB
client	MyISAM	5600	686.9 kB	113 kB
client_transaction	MyISAM	18675	1.2 MB	774 kB
client_transaction_hist	MyISAM	112050	7.7 MB	1 kB
client_transaction_hist2	MyISAM	1120500	76.9 MB	1 kB
client_transaction_hist3	MyISAM	14678550	1 GB	1 kB
investment	MyISAM	415	26.3 kB	13 kB
investment_type	MyISAM	10	248 B	2 kB
office_location	MyISAM	7	568 B	2 kB

La sauvegarde est réalisée comme suit :

```
mysql> backup database gimm to '/mybackups/gimm.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 1         |
+-----------+
1 row in set (58.64 sec)
```

Afin de tester la nature en ligne et non bloquantes en DML de la sauvegarde, j'ai lancé quelques opérations d'UPDATE pendant que la sauvegarde tournait :

```
mysql> update broker set broker_last_name = 'MORTONS' WHERE broker_last_name = 'MORTON';
Query OK, 1 row affected (0.19 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> update broker set broker_last_name = 'POTT' WHERE broker_last_name = 'POTTS';
Query OK, 1 row affected (0.19 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Aucun problème n'est rencontré, les opérations DML ont été exécutées sans hésitation pendant la sauvegarde. Comparez ce résultat à une commande *mysqldump* réalisée sur la même base de données, qui a pris environ 4 minutes...

```
mysqldump -uroot -pthisisntmypassword --port=3309 --extended-insert --quick gimm > gimm.dmp
```

Et comment cette opération de mise à jour est suspendue jusqu'à la fin de la sauvegarde...

```
mysql> update broker set broker_last_name = 'POTTS' WHERE broker_last_name = 'POTT';
```

... et vous verrez que la nouvelle sauvegarde MySQL a tourné environ 75% plus vite pour cette base MyISAM et a permis aux instructions DML de se poursuivre sans interruption, ce qui est une nouveauté bienvenue comparée aux capacités actuelles de *mysqldump*. Je dois quand même dire que lors de mes tests initiaux, la première sauvegarde MyISAM que j'ai réalisée a pris plus de temps (environ 2 minutes de plus), alors que les sauvegardes suivantes ont pris le même temps que celui affiché ci-dessus. L'équipe chargée de la sauvegarde MySQL étudie les raisons de ce constat, et je suis certain qu'ils vont résoudre le problème à l'avenir.

Vous pouvez suivre les exécutions de sauvegarde et restauration comme celle ci-dessus par l'intermédiaire d'une nouvelle table de métadonnées actuellement stockée dans la base mysql. Elle vous donne un bon nombre de détails à propos de votre sauvegarde (par exemple la taille de la sauvegarde, le nombre d'objets sauvegardés, etc.) et les exécutions de restauration pour un serveur particulier :

```
mysql> select * from online_backup\G
```

```
*************************** 1. row ***************************
          backup_id: 1
         process_id: 0
         binlog_pos: 14860603
        binlog_file: ./my5-bin.000001
       backup_state: complete
          operation: backup
          error_num: 0
        num_objects: 9
        total_bytes: 1149371056
validity_point_time: 2008-08-19 14:14:04
         start_time: 2008-08-19 14:13:05
          stop_time: 2008-08-19 14:14:04
host_or_server_name: NULL
           username: root
        backup_file: /mybackups/gimm.backup
       user_comment:
            command: backup database gimm to '/mybackups/gimm.backup'
            drivers: MyISAM
```

Vous pouvez voir, dans la sortie ci-dessus, que la taille de la sauvegarde était légèrement supérieure à 1,1 Go, ce qui reflète approximativement la taille de la base MyISAM qui a été sauvegardée.

Une autre table de métadonnées vous montre les opérations en cours, selon cet agencement :

```
mysql> desc online_backup_progress;
```

```
+-------------+---------------------+------+-----+---------+-------+
| Field       | Type                | Null | Key | Default | Extra |
+-------------+---------------------+------+-----+---------+-------+
| backup_id   | bigint(20) unsigned | NO   |     | NULL    |       |
| object      | char(30)            | NO   |     | NULL    |       |
| start_time  | datetime            | YES  |     | NULL    |       |
| stop_time   | datetime            | YES  |     | NULL    |       |
| total_bytes | bigint(20)          | YES  |     | NULL    |       |
| progress    | bigint(20) unsigned | YES  |     | NULL    |       |
| error_num   | int(11)             | NO   |     | 0       |       |
| notes       | char(100)           | YES  |     | NULL    |       |
+-------------+---------------------+------+-----+---------+-------+
```

Avant la sortie de la version de production 6.0, on peut s'attendre à ce que ces tables deviennent comme comme les *general query logs* (logs de requêtes générales) et les *slow query logs* (logs de requêtes lentes) de MySQL, pour lesquelles on peut **définir des tables ou des fichiers** pour le stockage de leurs données.

Restaurer la base de données s'est avéré très simple, avec la nouvelle commande *restore* :

```
mysql> use mysql
Database changed
mysql> drop database gimm;
Query OK, 9 rows affected (1.34 sec)

mysql> restore from '/mybackups/gimm.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 2         |
```

```
+-----------+
1 row in set (56.48 sec)
```

```
mysql> select count(*) from gimm.client_transaction_hist3;
```

```
++----------+
| count(*) |
+----------+
| 14678550 |
+----------+
1 row in set (0.00 sec)
```

Restaurer la même base avec un fichier *mysqldump* (ce qui se fait habituellement par la commande suivante : *shell> mysql db_name < backup-file.sql*) a pris 8 minutes et 40 secondes, ce qui signifie que la nouvelle restauration apporte une réduction du temps de réponse de 89% par rapport à une restauration *mysqldump* classique. C'est assurément une fonctionnalité intéressante, sachant que la restauration de bases critiques dans un environnement de production doit habituellement être effectuée le plus rapidement possible.

III - Sauvegarde et restauration InnoDB avec MySQL 6.0 Backup

Essayons à présent la nouvelle sauvegarde sur une base de plus d'1 Go en InnoDB uniquement, identique à la base de données MyISAM que nous venons tout juste de tester :

Table Name	Engine	Rows	Data length	Index length
broker	MyISAM	23	812 B	3 kB
client	MyISAM	5600	686.9 kB	113 kB
client_transaction	MyISAM	18675	1.2 MB	774 kB
client_transaction_hist	MyISAM	112050	7.7 MB	1 kB
client_transaction_hist2	MyISAM	1120500	76.9 MB	1 kB
client_transaction_hist3	MyISAM	14678550	1 GB	1 kB
investment	MyISAM	415	26.3 kB	13 kB
investment_type	MyISAM	10	248 B	2 kB
office_location	MyISAM	7	568 B	2 kB

```
mysql> backup database gim2 to '/mybackups/gim2.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 10        |
+-----------+
1 row in set (3 min 21.83 sec)
```

La mise à jour suivante, ainsi que d'autres, ont été effectuées pendant la sauvegarde, pour s'assurer que le DML n'était pas interrompu :

```
mysql> update gim2.office_location set office_city='WASHINGTON' where office_city = 'WASHINGTONS';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Comme vous pouvez le constater, la sauvegarde de la base InnoDB a pris un peu plus de temps que celle de la base MyISAM, du fait qu'il s'agit d'une sauvegarde logique, par rapport à une sauvegarde native. Dans la version alpha actuelle de MySQL 6.0.6, elle a tourné un peu plus lentement (10 secondes) qu'un *mysqldump* de la même

base. Toutefois, le nouveau MySQL 6.0 présente un avantage en termes de taille finale de sauvegarde ; le fichier de sauvegarde *mysqldump* était plus volumineux d'environ 500 MB que le fichier de sauvegarde 6.0.

La restauration de la base InnoDB a été effectuée ainsi :

```
mysql> restore from '/mybackups/gim2.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 15        |
+-----------+
1 row in set (4 min 44.11 sec)
```

Restaurer la même base InnoDB avec un fichier *mysqldump* a pris 11 minutes 30 secondes, ce qui signifie que, bien qu'il n'y ait pas eu une grosse différence de vitesse entre la sauvegarde *mysqldump* et 6.0 pour InnoDB, la nouvelle restauration 6.0 a apporté une réduction de 59% du temps de réponse par rapport à une restauration classique *mysqldump*, ce qui est, une nouvelle fois, plutôt sympa.

IV - Qu'en est-il des bases utilisant plusieurs moteurs ?

La nouvelle sauvegarde 6.0 fonctionne aussi correctement avec les bases de données utilisant plusieurs moteurs (sauf pour MySQL Cluster). Par exemple, dans la base MyISAM de test que nous venons d'utiliser, remplaçons une des tables MyISAM par une table InnoDB et regardons ce qui se produit :

```
mysql> use gimm;
Database changed
mysql> show tables;
```

```
+--------------------------+
| Tables_in_gimm           |
+--------------------------+
| broker                   |
| client                   |
| client_transaction       |
| client_transaction_hist  |
| client_transaction_hist2 |
| client_transaction_hist3 |
| investment               |
| investment_type          |
| office_location          |
+--------------------------+
9 rows in set (0.02 sec)
```

```
mysql> alter table client_transaction_hist2 engine=innodb;
Query OK, 1120500 rows affected (23.33 sec)
Records: 1120500  Duplicates: 0  Warnings: 0

mysql> backup database gimm to '/mybackups/gimmi.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 21        |
+-----------+
1 row in set (1 min 41.45 sec)
```

Nous pouvons constater une légère réduction du temps de réponse puisque la sauvegarde utilise le pilote natif MyISAM pour toutes les tables sauf une, et fait alors une sauvegarde logique de la table InnoDB contenant un million de lignes.

V - Essais de compression

La sauvegarde MySQL 6.0 possède une fonctionnalité de compression qui vous permet de réduire la taille des fichiers de sauvegarde en utilisant une compression standard zlib. Appeler la compression est très facile et se fait juste avec un simple mot clef.

Par exemple, testons la compression avec notre base MyISAM d'1 Go :

```
mysql> backup database gimm to '/mybackups/gimm.backup' with compression;
```

```
+-----------+
| backup_id |
+-----------+
| 16        |
+-----------+
1 row in set (2 min 15.38 sec)
```

Et testons maintenant la compression avec la base InnoDB d'1 Go :

```
mysql> backup database gim2 to '/mybackups/gim2.backup' with compression;
```

```
+-----------+
| backup_id |
+-----------+
| 17        |
+-----------+
1 row in set (3 min 38.69 sec)
```

Vous noterez que dans la version alpha 6.0.6 actuelle, la sauvegarde MyISAM prend plus du double de temps en exécution par rapport à une sauvegarde non compressée, tandis que la sauvegarde logique InnoDB ne souffre pratiquement pas de l'utilisation de la compression. L'équipe de sauvegarde de MySQL étudie actuellement la question du temps de réponse sur les sauvegardes MyISAM, donc tenez-vous informé des mises à jour.

Dans les deux cas, toutefois, une réduction de plus de 80% de la taille du fichier de sauvegarde est effectuée, ce qui est un bon bénéfice. Par exemple, la sauvegarde compressée de la base MyISAM (de plus d'1 Go) et InnoDB fait une taille de 214 MB, contre 1,1 Go pour la sauvegarde non compressée MyISAM et 1,6 Go pour la sauvegarde InnoDB.

La restauration de la base MyISAM d'1 Go se fait par la commande normale de restauration :

```
mysql> use mysql;
Database changed
mysql> drop database gimm;
Query OK, 9 rows affected (1.39 sec)

mysql> restore from '/mybackups/gimm.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 18        |
+-----------+
1 row in set (44.39 sec)
```

```
mysql> select count(*) from gimm.client_transaction_hist3;
```

```
+----------+
| count(*) |
+----------+
| 14678550 |
+----------+
1 row in set (0.00 sec)
```

Comme vous pouvez le voir, la restauration de la base MyISAM a tourné plus vite que la restauration non compressée. Restaurer la base InnoDB d'1 Go à partir d'une sauvegarde compressée a pris sensiblement le même temps que la restauration non compressée :

```
mysql> drop database gim2;
Query OK, 9 rows affected (0.34 sec)

mysql> restore from '/mybackups/gim2.backup';
```

```
+-----------+
| backup_id |
+-----------+
| 20        |
+-----------+
1 row in set (4 min 30.38 sec)
```

```
mysql> use gim2;
Database changed
mysql> show tables;
```

```
+--------------------------+
| Tables_in_gim2           |
+--------------------------+
| broker                   |
| client                   |
| client_transaction       |
| client_transaction_hist  |
| client_transaction_hist2 |
| client_transaction_hist3 |
| investment               |
| investment_type          |
| office_location          |
+--------------------------+
9 rows in set (0.00 sec)
```

VI - Restrictions de MySQL Backup 6.0 version 1

Dans MySQL 6.0, la première version du nouvel outil de sauvegarde présente un certain nombre de restrictions que vous devez connaître, dont les plus importantes sont :

- la sauvegarde complète d'une instance MySQL n'est pas possible ; la première version du nouvel outil de sauvegarde est prévue uniquement pour les bases de données utilisateurs, et ne sauvegarde pas la base mysql, ou d'autres composants comme les logs binaires, les fichiers de configuration, etc. Pour vous assurer d'une sauvegarde solide, vous pouvez faire suivre vos sauvegardes à partir du nouvel outil de MySQL 6.0 par une commande *mysqldump* sur la base *mysql*, qui sera adaptée à la dynamique de votre instance (ce qui signifie, par exemple, que plus vous modifiez les identifiants de connexion, plus vous devez exécuter souvent des sauvegardes de votre base *mysql*) ;
- une restauration de l'état de la base à un instant donné est plus délicate pour des bases séparées, dans la mesure où les logs binaires de MySQL contiennent les opérations pour toutes les bases d'une instance MySQL. Un administrateur doit garder à l'esprit que le fait de rejouer les logs peut provoquer des cas dans lesquelles les bases ne pourront pas être restaurées par la nouvelle commande de restauration ;
- les pilotes natifs ne sont disponibles que pour les moteurs MyISAM, Federated, Merge, Blackhole et Example en version 1. Les pilotes natifs pour InnoDB et Falcon sont prévus pour la prochaine version ;
- les moteurs autres que MyISAM et les moteurs transactionnels doivent être sauvegardés hors ligne (ce qui signifie que les actions DML sont bloquées durant l'opération) ;
- MySQL Cluster n'est pas pris en charge dans la mesure où il possède son propre outil de sauvegarde ;
- les opérations DDL sont interrompues pendant tous les types de sauvegardes ;
- les commandes de sauvegarde/restauration ne peuvent pas être lancées depuis des évènements, ni depuis des procédures stockées ou des triggers dans la version 1 de l'utilitaire ;
- les restaurations ne peuvent pas être de nature sélective ; seule la totalité du fichier de sauvegarde peut être restaurée ;

- d'autres médias, comme les bandes de sauvegarde, ne sont pas encore pris en charge ;
- la sauvegarde au niveau objet n'est pas implémentée ; la granularité est la base de données ;
- les sauvegardes incrémentales et différentielles ne sont pas possibles ; seule une sauvegarde totale de la base de données est possible.

VII - Les étapes de MySQL Backup

La première version de MySQL Backup 6.0 offre déjà de réels avantages, mais ne peut pas tout faire dès le premier jet. A l'heure actuelle, voici la *roadmap* de l'utilitaire de sauvegarde/restauration qui se dessine :

Version 1.1 de l'utilitaire de sauvegarde/restauration :

- pilote natif pour le moteur InnoDB ;
- pilote natif pour le moteur Falcon ;
- sauvegarde/restauration au niveau objet ;
- initialisation un esclave de réplication à partir d'un fichier de sauvegarde ;
- sauvegarde sur d'autres supports (par exemple sur bande, etc.) ;
- possibilité de sauvegarde/restauration d'une instance complète.

Version 1.2 de l'utilitaire de sauvegarde/restauration :

- sauvegarde incrémentale ;
- sauvegarde différentielle ;
- support XSBA ;
- mirroring de sauvegarde (écriture des sauvegardes à plusieurs endroits) ;
- checksum.

D'autres améliorations sont également prévues, mais celles-ci représentent les plus importantes dans la liste des souhaits des utilisateurs. S'il y a le moindre retour sur expérience ou suggestion d'amélioration que vous souhaiteriez inclure, n'hésitez pas à me contacter (robin.schumacher@sun.com).

Conclusion

Quelques années ont passées depuis que j'ai complètement massacré la base de données d'un client important, mais heureusement, j'ai finalement fait tout ce qu'il fallait parce que j'avais mis en place un plan de restauration élémentaire en place. En tant que responsable de MySQL, vous devez vous assurer que vous avez en place un plan solide pour protéger vos données et être capable de les restaurer en cas de catastrophe, qu'il s'agisse d'une implosion du serveur ou d'un utilisateur avec trop de pouvoir qui efface quelques lignes de données critiques de l'entreprise.

Le nouvel utilitaire de sauvegarde et restauration de MySQL 6.0 peut jouer un rôle fort dans votre plan de restauration en cas de désastre, donc il serait avisé de vous familiariser avec lui dès à présent. Vous pouvez télécharger MySQL 6.0 et avoir un aperçu de la nouvelle sauvegarde très facilement, en visitant **http://dev.mysql.com/downloads/mysql/6.0.html**. Et comme il est actuellement en version alpha, nous apprécierions que vous rapportiez le moindre bug sur notre système de bugs en ligne : **http://bugs.mysql.com/**. Enfin, n'oubliez pas de lire le manuel 6.0 de MySQL pour vous familiariser avec les détails de cet utilitaire, que vous trouverez à l'adresse : **http://dev.mysql.com/doc/refman/6.0/en/backup-database-restore.html**.

Nous apprécions particulièrement toute l'aide et le retour d'expérience que vous pourriez apporter sur le nouvel outil de sauvegarde et restauration, et, comme toujours, merci de votre soutien envers MySQL et Sun.