Olivier DASINI
présente

Les nouveautés de MySQL 5.1

olivier@dasini.net

http://dasini.net/blog/

MySQL 5.1 – Fonctionnalités

- Le partitionnement
- Le programmateur d'évènements
- La journalisation par les données (row based)
- MySQL Cluster : support des données sur disque
- Les tables de journalisation
- Le moteur de stockage CSV
- Le support de XML/XPath
- mysqlslap

MySQL 5.1 – Le partitionnement

- Fragmenter une table en fonction des données qu'elle contient pour:
  - Améliorer les performances lors de la recherche (***pruning***)
  - Faciliter certaines tâches de maintenance
  - Stocker les données et les index sur différents disques

MySQL 5.1 – Le partitionnement

CREATE TABLE ... ENGINE = <engine>

PARTITION BY <type> (<expression>)

<engine> = MyIsam | InnoDB | Archive | Falcon | NDBCluster | Memory

<type> = RANGE | LIST | HASH | KEY

- 4 types de partitionnement:
  - **RANGE** (par intervalle)
  - **LIST** (par liste)
  - **HASH** ou **KEY** (par hachage)

MySQL 5.1 – Le partitionnement

```
-- Partitionnement par intervalle
CREATE TABLE City_part_range (
   ID int(11) NOT NULL AUTO_INCREMENT,
   Population int(11) NOT NULL DEFAULT 0,
   KEY ID (ID)
) ENGINE=MyISAM
   PARTITION BY RANGE (population) (
   PARTITION p1 VALUES LESS THAN (1000),
   PARTITION p2 VALUES LESS THAN (5000),
   PARTITION p3 VALUES LESS THAN (10000),
   PARTITION p5 VALUES LESS THAN MAXVALUE );
```

MySQL 5.1 – Le partitionnement

```
-- Partitionnement par key
CREATE TABLE City_part_key (
   ID int(11) NOT NULL AUTO_INCREMENT,
   Name char(35) NOT NULL DEFAULT '',
   CountryCode char(3) NOT NULL DEFAULT '',
   District char(20) NOT NULL DEFAULT '',
   Population int(11) NOT NULL DEFAULT 0,
   PRIMARY KEY (ID)
) ENGINE=InnoDB PARTITION BY KEY () PARTITIONS 6;
```

MySQL 5.1 – Le partitionnement

```
-- Partitionnement par liste
CREATE TABLE Country_part_list (
   Code char(3) NOT NULL DEFAULT '',
   Continent tinyint UNSIGNED DEFAULT 1,
   KEY (Code)
) ENGINE=MEMORY
   PARTITION BY LIST(Continent) (
   PARTITION pCateg1 VALUES IN (1),
   PARTITION pCateg2 VALUES IN (4),
   PARTITION pCateg3 VALUES IN (2,6),
   PARTITION pCateg4 VALUES IN (3,5,7));
```

MySQL 5.1 – Les évènements

- Planificateur de tâches (CRON-like) embarqué dans MySQL 5.1
- Exécute des requêtes, en fonction de la date et de l'heure
- De façon récurrente (EVERY) ou unique (AT)

MySQL 5.1 – Les évènements

CREATE EVENT nom_evenement ON SCHEDULE
<moment> DO <code_sql>

Peut être lancé une seule fois (AT) ou de manière répétitive (EVERY):
<moment> = AT | EVERY

L'évènement est constitué d'un ensemble de requêtes:
<code_sql> = requêtes sql

MySQL 5.1 – Les évènements

-- Créer une vue matérialisée rafraîchie toutes les 10 minutes

```
DELIMITER //
CREATE EVENT vue_materialisee
ON SCHEDULE EVERY 10 MINUTE
DO
BEGIN
  TRUNCATE TABLE _event.City_fra;
  INSERT INTO _event.City_fra
    SELECT * FROM world.City WHERE CountryCode='FRA'
    ORDER BY name;
END//
DELIMITER ;
```

MySQL 5.1 – Row based login

- Jusqu'à MySQL 5.0.x: statement based login seulement
  - Enregistrement de la requête dans le log binaire
- MySQL 5.1: row based login possible:
  - Enregistrement du résultat de la requête

MySQL 5.1 – Row based login

Trois options de journalisation :

- **Statement** : journalise la requête telle-quelle
- **Row** : journalise le résultat de la requête
- **Mixed** : MySQL choisit entre statement et row en fonction du contexte.

MySQL 5.1 – Row based login

-- Modification du mode de journalisation

```
mysql> SHOW VARIABLES LIKE 'binlog_format';
+---------------+-----------+
| Variable_name | Value     |
+---------------+-----------+
| binlog_format | STATEMENT |
+---------------+-----------+

mysql> SET SESSION binlog_format='ROW';

mysql> SHOW VARIABLES LIKE 'binlog_format';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| binlog_format | ROW   |
+---------------+-------+
```

MySQL 5.1 – MySQL Cluster

- Jusqu'à MySQL 5.0.x : Données et index qu'en mémoire.
- MySQL 5.1: Possible de mettre les données sur disque.

MySQL 5.1 – MySQL Cluster

LOGFILE GROUP : permet gérer les undo log et le crash-recovery

-- Création d'un logfile group

```
CREATE LOGFILE GROUP lg_1
   ADD UNDOFILE 'undo_1.dat'
   INITIAL_SIZE 16M
   UNDO_BUFFER_SIZE 2M
   ENGINE NDB;
ALTER LOGFILE GROUP lg_1
   ADD UNDOFILE 'undo_2.dat'
   INITIAL_SIZE 12M
   ENGINE NDB;
```

MySQL 5.1 – MySQL Cluster

TABLESPACE : pour stocker les données.

-- Création d'un tablespace

```
CREATE TABLESPACE ts_1
   ADD DATAFILE 'data_1.dat'
   USE LOGFILE GROUP lg_1
   INITIAL_SIZE 32M
   ENGINE NDB;
ALTER TABLESPACE ts_1
   ADD DATAFILE 'data_2.dat'
   INITIAL_SIZE 48M
   ENGINE NDB;
```

MySQL 5.1 – MySQL Cluster

-- Table au format NDBCluster avec données sur disque

```
CREATE TABLE 'City' (
   'ID' int(11) NOT NULL AUTO_INCREMENT,
   'Name' char(35) NOT NULL DEFAULT '',
   'Population' int(11) NOT NULL DEFAULT '0',
   PRIMARY KEY ('ID')
) TABLESPACE ts_1
   STORAGE DISK
   ENGINE=NDB;
```

MySQL 5.1 – Table de journalisation

- **general log**: journaliser l'activité du serveur
- **slow query log**: journaliser seulement les requêtes lentes
- Journaliser dans un fichier ou dans un table

MySQL 5.1 – Table de journalisation

Activable à chaud

```
SET GLOBAL general_log = 'ON';
SET GLOBAL slow_query_log = 'ON';
```

Choix du format de journalisation

```
SET GLOBAL log_output = 'TABLE';
SET GLOBAL log_output = 'FILE';
SET GLOBAL log_output = 'TABLE,FILE';
SET GLOBAL log_output = 'NONE';
```

MySQL 5.1 – Table de journalisation

```
SELECT * FROM mysql.general_log WHERE argument like
'select%from%select%'\G

*************************** 1. row ***************************
  event_time: 2009-03-01 18:49:49
   user_host: root[root] @ localhost [127.0.0.1]
   thread_id: 12
   server_id: 3306
command_type: Query
    argument: select * from city where countrycode in (select code
from country where continent='europe') group by
countrycode

*************************** 2. row ***************************
  ...
```

MySQL 5.1 – Moteur de stockage CSV

- Stocker les données dans un fichier texte au format CSV
- Editable avec un tableur ou éditeur de textes
- Import / export par simple « copier/coller »

MySQL 5.1 – Moteur de stockage CSV

-- Création d'une table au format CSV

```
CREATE TABLE t_csv (
   id int,
   nom CHAR(50),
   prenom CHAR(50)
)
ENGINE=CSV;
```

MySQL 5.1 – XML /XPath

- Support basique de XML avec deux fonctions:
  - **ExtractValue** : extraire les valeurs des différentes balises
  - **UpdateXML** : modifier la sortie d'un document XML

MySQL 5.1 – XML /XPath

```
SELECT EXTRACTVALUE
(texte,'/profil/consultant[contains(@categorie,"MySQL")]/con
sultant) AS Expert_MySQL FROM consultant_xml;

+--------------------+
| Expert_MySQL |
+--------------------+
| Olivier DASINI  |
+--------------------+
```

MySQL 5.1 – mysqlslap

- Permet d'effectuer des tests de stress et de charge sur votre serveur MySQL
- Envoie des requêtes au serveur MySQL en créant plusieurs connexions simultanées
- Un rapport de diagnostic est crée sur la sortie standard.
- Possibilité de l'écrire dans un fichier au format CSV

MySQL 5.1 – mysqlslap

--Tests de performances avec des tables MyISAM et innoDB

```
mysqlslap --user=daz  --password
    --concurrency=1,100
    --iterations=10 –number-of-queries=1000
    --engine=myisam,innodb
    --auto-generate-sql --number-int-cols=2
    --number-char-cols=3
    --auto-generate-sql-load-type=mixed
```

MySQL 5.1 – mysqlslap

```
Benchmark
   Running for engine myisam
   Average number of seconds to run all queries: 1.576 seconds
   Minimum number of seconds to run all queries: 1.539 seconds
   Maximum number of seconds to run all queries: 1.631 seconds
   Number of clients running queries: 1
   Average number of queries per client: 1000
Benchmark
   Running for engine myisam
   Average number of seconds to run all queries: 1.792 seconds
   Minimum number of seconds to run all queries: 1.543 seconds
   Maximum number of seconds to run all queries: 2.107 seconds
   Number of clients running queries: 100
   Average number of queries per client: 10
Benchmark
   Running for engine innodb
   Average number of seconds to run all queries: 2.477 seconds
   ...
```

Olivier DASINI
vous a présenté

Les nouveautés de MySQL 5.1

http://dasini.net/blog/

olivier@dasini.net