MySQL Replikation für Einsteiger

Oli Sennhauser
FromDual GmbH
Uster / Schweiz

Schlüsselworte
MySQL, Replikation, Scale-Out, Master, Slave

Einleitung
Die Replikation ist eine der wichtigsten Funktionalitäten von MySQL. Sei es aus Performance- oder Hochverfügbarkeitsgründen, um die MySQL Replikation kommt man früher oder später nicht darum herum.

Was ist Replikation?
Unter Replikation versteht man das Weiterreichen von Daten von einer Datenbank, welche als Master bezeichnet wird, auf eine oder mehrere andere Datenbanken, die sogenannten Slaves. Diese Daten stammen von Applikationen, welche ihre Daten in den Master schreiben.

Unter Daten verstehen wir entweder DML-Statements (`UPDATE`, `INSERT`, `DELETE`, etc.) oder binäre Events welche die Daten(-änderungen) beinhalten.

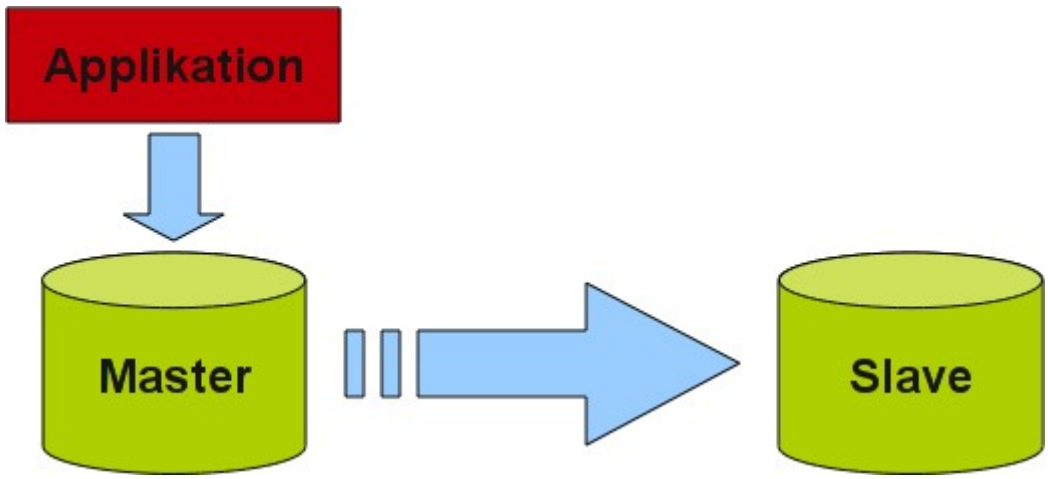

Abb. 1: Master/Slave Replikation

Die MySQL Replikation
Durch das einschalten des Binary Logs wird der Master dazu veranlasst, seine Datenänderungen in diese Log Dateien zu schreiben.

Der Slave verbindet sich mittels des IO-Treads auf den Master und fordert die entsprechenden Informationen an, welche er gerne als nächstes applizieren möchte. Sobald der Slave diese Daten erhalten hat, legt er sie in sein Relay Log ab, welches als Zwischenpuffer und Kopie der Binary Logs dient.

Ein zweiter Thread, der sogenannten SQL-Thread liest nun dieses Relay Log aus und appliziert die darin enthaltenen Informationen auf den Slave.

Dieser ganze Replikationsprozess erfolgt asynchron.

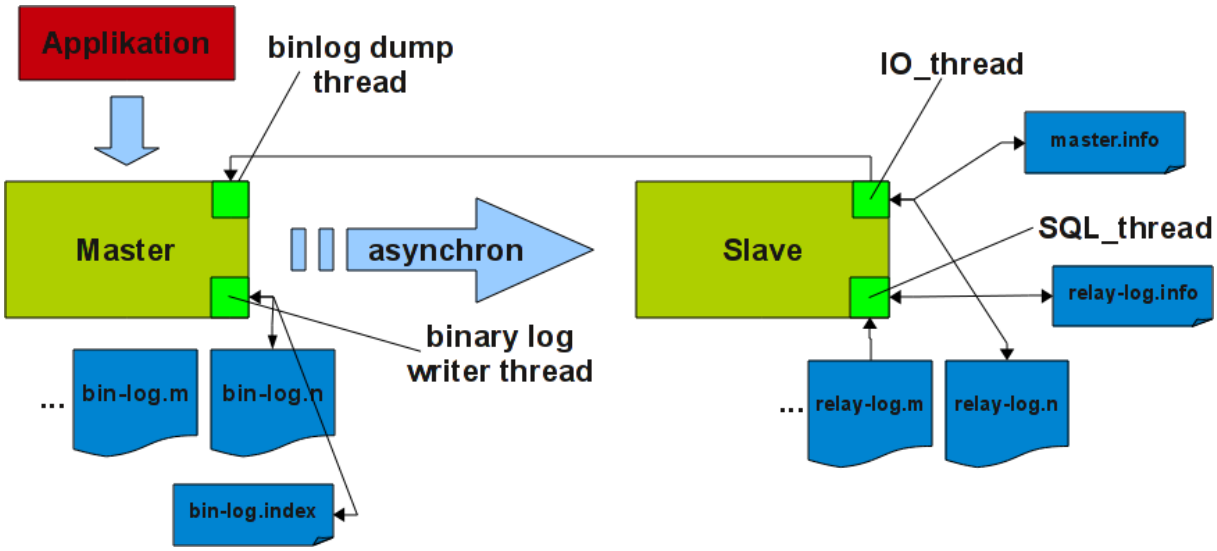

Abb. 2: MySQL Replikation

Vorbereiten des Masters
Um eine MySQL Datenbank zum Master zu machen müssen zwei Bedingungen erfüllt sein. Als erstes muss in der MySQL Konfigurationsdatei das Binary Log mittels der Variable `log-bin` eingeschaltet werden. Zusätzlich muss in einem Replikationsverbund jeder Komponente eine eindeutige ID zugewiesen werden, was mit der Variable `server-id` erfolgt.

Um diese Einstellungen zu aktivieren muss die Datenbank anschliessend neu gestartet werden. Eine entsprechende Downtime ist also für diesen Schritt mit einzuplanen.

Anschliessend muss auf dem Master der entsprechende Nutzer angelegt werden, mit welchem die Slaves auf den Master zugreifen sollen:

```
CREATE USER 'replication'@'192.168.1.%'
IDENTIFIED BY 'secret';

GRANT REPLICATION SLAVE ON *.*
TO 'replication'@'192.168.1.%';
```

Um einen Slave aufsetzen ist es zudem notwendig einen initialen konsistenten Dump des Masters zu erstellen, welcher die Information der dazugehörenden Binary Log Position, beinhaltet. Diese geschieht mit dem Befehl:

```
mysqldump --all-databases --single-transaction \
          --master-data > full_dump.sql
```

Aufsetzen des Slaves
Um einen neuen Slave aufzusetzen empfiehlt es sich, zuerst auf einem separaten Rechner eine neue MySQL Instanz zu installieren und zu starten. Diese Instanz muss eine eignen `server-id` aufweisen, da sonst die MySQL Replikation mit einer Fehlermeldung abbrechen wird.

Anschliessend wird dem zukünftigen Slave mitgeteilt, wo er seinen Master findet und welcher Nutzer für die Replikation verwendet werden soll:

```
CHANGE MASTER TO master_host='192.168.1.42'
, master_port=3306, master_user='replication'
, master_password='secret';
```

Dann kann der auf dem Master gezogene initiale Dump auf dem Slave eingespielt werden. Nach einem finalen Überprüfen aller Einstellungen kann der Slave mit dem Befehl `START SLAVE` gestartet werden.

Der Slave verbindet sich nun automatisch mit dem Master und holt sich die ihm noch fehlenden Daten ab.

Probleme beim Aufsetzen und Betrieb
Sofern man der Anleitung in der MySQL Dokumentation folgt funktioniert die MySQL Replikation einwandfrei. Die häufigsten Ursachen warum die MySQL Replikation fehlschlägt liegt darin, dass diese Dokumentation nicht korrekt befolgt wurde.

Ein häufig gemachter Fehler besteht zum Beispiel darin, dass kein initiales konsistentes Backup der Daten für das Aufsetzen des Slaves verwendet wurde. Die geschieht zum Beispiel dann, wenn der Parameter `--single-transaction` nicht verwendet wird oder wenn jedes Schema einzeln auf dem Slave eingespielt wird. Dann kommt es zum Beispiel zu Primary Key Verletzungen.

Probleme, welche sich beim Betrieb ergeben sind oft dadurch begründet, dass die Master/Slave Replikation nicht überwacht wird. Zu überwachen ist, dass der Slave nicht allzu oft und allzu weit hinter dem Master hinterher hinkt. Ist dies der Fall, gilt zu überprüfen, was hierfür die Ursache ist (zum Beispiel zu schwache Hardware, nicht optimale Abfragen, etc.).

Auf dem Master sollte man von Zeit zu Zeit die Binary Logs weg löschen, wenn diese nicht mehr gebraucht werden um ein voll laufen der Platten zu vermeiden. Dies geschieht mit dem Befehl: `PURGE BINARY LOGS`.

Bei Fehlmanipulation oder mit der Statement basierten Replikation (SBR) kann es auch zu Dateninkonsistenzen zwischen Master und Slave kommen. Diese sollten behoben oder der gegebenenfalls der Slave neu aufgesetzt werden.

Wann wird die MySQL Replikation gebraucht?
Die MySQL Replikation wird für verschiedene Szenarien eingesetzt. Oft wird sie verwendet um Hochverfügbarkeit zu erreichen. Fällt der Master aus, kann auf dem Slave weiter gearbeitet werden. Bei einem sehr hohen Leseaufkommen können mehrere Slaves verwendet werden, um diese Last zu bewältigen (Lese-Scale-Out). Ein ähnliches Szenario ist dann gegeben, wenn wir den Slave für das Backup verwenden wollen oder zu Reporting- und Datenanalyse Zwecke einsetzen.

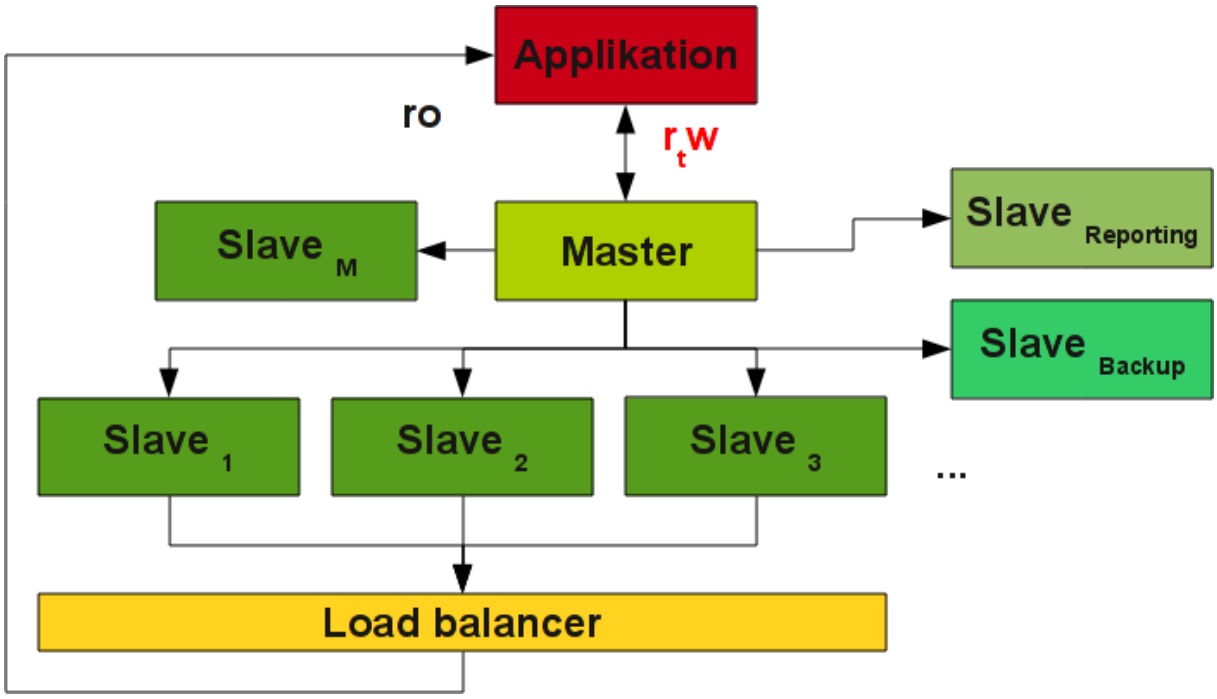

Abb. 3: MySQL Scale-Out

Neuerungen in der MySQL Replikation
Mit MySQL 5.1 ist neu die Row basierte Replikation (RBR) hinzugekommen. Da die "alte" Statement basierte Replikation (SBR) Nachteile betreffen Datensicherheit und Datenkonsistenz mit sich bringt, ist es empfehlenswert, das neue Format zu verwenden, wenn höchste Anforderungen an die Datenkonsistenz bestehen. Das Umschalten erfolgt einfach durch das ändern des Parameters `binlog_format` auf "`row`".

RBR gilt heute als die sicherste Art der Replikation.

Die Replikation bis und mit MySQL 5.1 erfolgt asynchron. Diese bedeutet, dass wir keine Garantie dafür haben, dass Daten, welche auf dem Master geschrieben wurden, auch auf dem Slave angekommen sind.

Um den Datenverlust im Falle eines Master-Crashes zu verhindern ist in MySQL 5.5 die semi-synchrone Replikation hinzugekommen. Dieses Funktionalität kann in MySQL 5.5 zugeschaltet werden und sie stellt sicher, dass mindestens 1 Slave die Daten vom Master auch wirklich erhalten hat. Somit haben wir Sicherheit, dass im Fall eines Crashes keine Daten verloren gehen und wir mit dem vollständigen Datenset auf dem Slave weiterarbeiten können.

Diesen Vorteil handeln wir uns auf Kosten einer schlechteren Schreibperformance ein.

Ein Nachteil der in MySQL 5.1 hinzugekommen Row basierten Replikation (RBR) ist, das unter Umständen die Datenmenge, welche vom Master auf den Slave geschickt wird, je nach Art der Abfragen, signifikant anwächst.

Um diesem Verhalten entgegen zu wirken hat MySQL in Version 5.6 die neue Funktionalität namens „Row Image Control" eingeführt. Mit dem Parameter `binlog_row_image` kann gesteuert werden ob eine Zeile komplett oder ob nur die geänderten Spalten übermittelt werden sollen. Dadurch lässt

sich die Datenmenge in einigen Szenarien signifikant reduzieren und auch die Slave-Performance verbessern.

Ebenfalls mit MySQL 5.6 neu hinzugekommen sind die Crash-sicheren Binary Logs. Es sollte jetzt also nicht mehr möglich sein, dass beim Crash eines Master irgendwelche Inkonsistenzen zwischen Transaktionslog und Binary Log auftreten. Das sollte auch die Replikation zum Slave noch robuster machen.

Um auch auf dem Slave bei der Replikation mehr Robustheit zu erlangen können die beiden Dateien `master.info` und `relay-log.info` neu als Tabellen in der Datenbank abgelegt werden. Zur Zeit werden diese noch als (nicht Crash-sichere) MyISAM Tabellen angelegt. Man kann diese aber von Hand selber noch nach InnoDB umwandeln.

Weiter hinzugekommen ist das Remote Binary Log Shipping. Mit dem Programm `mysqlbinlog` können jetzt neu die Binary Logs vom Master auf eine andere Maschine kopiert werden, damit im Falle eines Disk-Ausfalles die Binary Logs noch vorhanden sind.

Delayed Replikation ist ebenfalls neu hinzugekommen. Früher musste man hierfür eine Skript eines Fremdanbieters bemühen. Neu kann man einen Slave mit dem Befehl `CHANGE MASTER TO MASTER_DELAY` hinterher hinken lassen.

Kontaktadresse:
Oli Sennhauser
FromDual GmbH
Rebenweg 6
CH – 8610 Uster

Telefon:	+41 44 – 940 24 82
Fax:	+41 43 – 55 68204
E-Mail	oli.sennhauser@fromdual.com
Internet:	www.fromdual.com