



MySQL Replikation

DOAG Konferenz 2012, Nürnberg

Oli Sennhauser

Senior MySQL Consultant, FromDual GmbH

oli.sennhauser@fromdual.com



Über FromDual GmbH

- **FromDual bietet neutral und unabhängig:**
 - **Beratung für MySQL und Galera Cluster**
 - **Support für MySQL und Galera Cluster**
 - **MySQL Remote-DBA Dienstleistungen**
 - **MySQL Schulungen**
- **Oracle Silver Partner (OPN)**



www.fromdual.com

Kunden

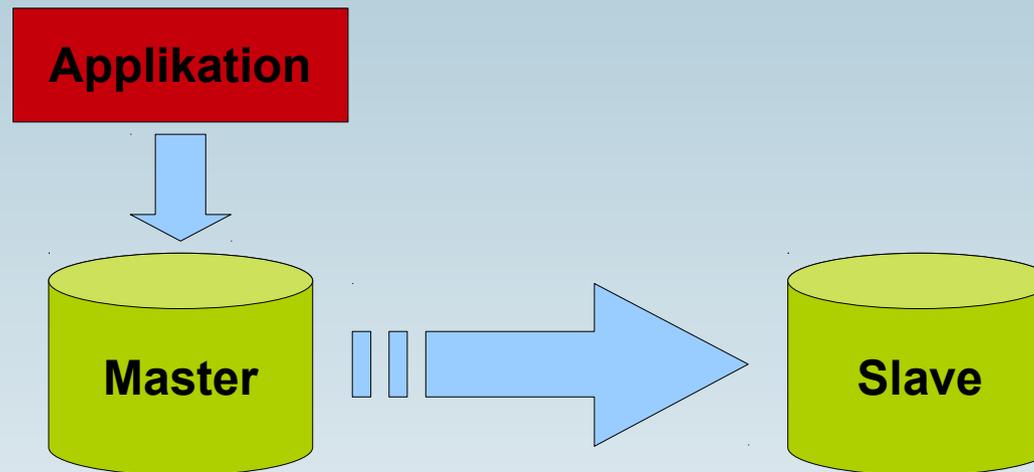


MySQL Replikation

- › Was ist Replikation?
- › Wie funktioniert MySQL Replikation?
- › Wie wird Replikation aufgesetzt?
- › Wann brauche ich MySQL Replikation?
- › Neuerungen in 5.1, 5.5 und 5.6

Was ist Replikation?

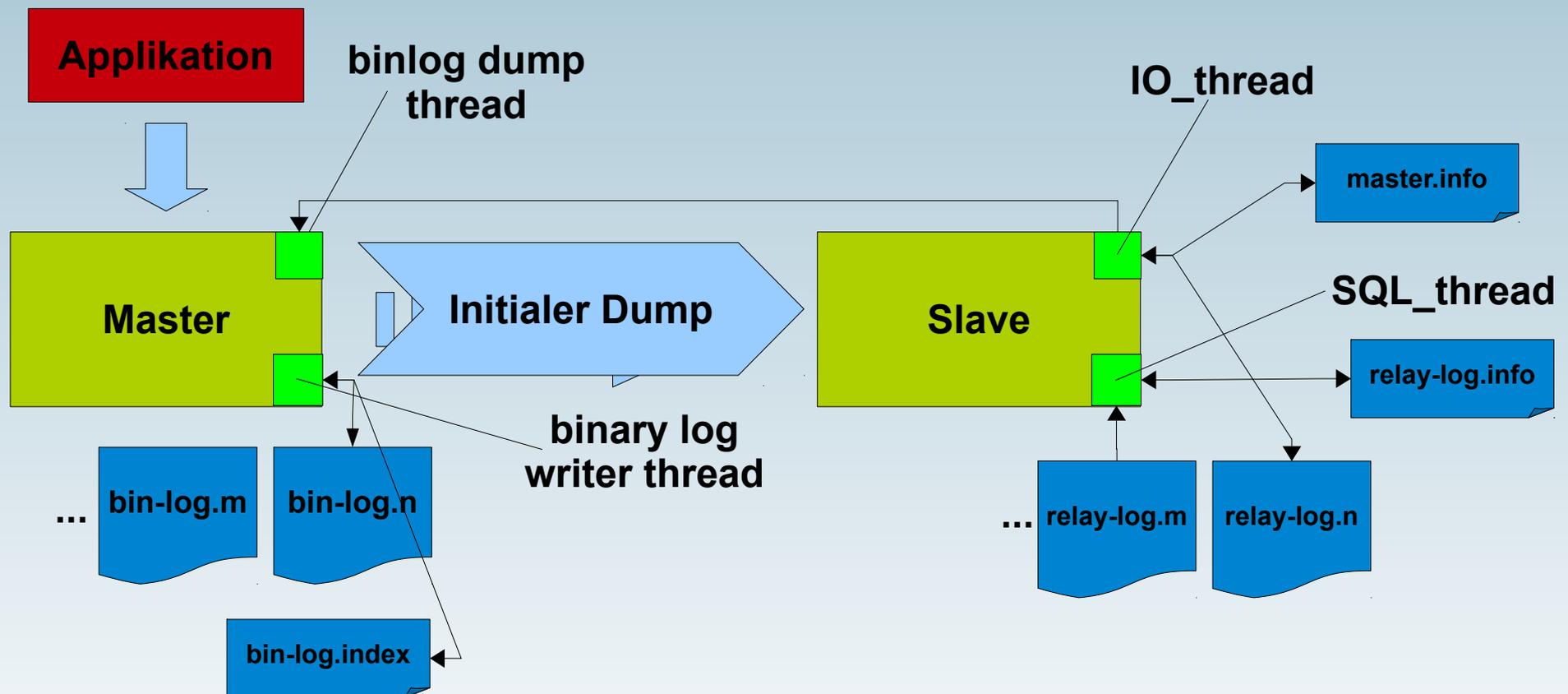
- Daten von einer Datenbank auf eine oder mehrere andere Datenbanken weiterreichen:



- Daten = UPDATE, INSERT, DELETE, ... (= DML Statements) oder (binäre) „Events“

MySQL Replikation

- Wie funktioniert die MySQL Replikation?



Vorbereiten des Masters

- **Auf dem Master im `my.cnf`:**
 - **Binary log einschalten:**
`log-bin = bin-log`
 - **Server ID setzen:**
`server_id = 42`
 - **Muss im Replikations-Setup Unique sein!**
- **Server neu starten (downtime!)**

User anlegen und Dump

- Auf dem Master:
- Replikations-User anlegen:

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

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

- Initialen **konsistenten!** Dump ziehen:

```
mysqldump --all-databases --single-  
transaction (--lock-all-tables)  
--master-data > full_dump.sql
```

Aufsetzen des Slaves

- Neue Datenbank erstellen (2. Maschine)

```
./scripts/mysql_install_db \  
--datadir=/var/lib/mysql
```

- Andere Server ID setzen:

- `server_id = 43`

- Slave auf seinen Master ansetzen:

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

- Initialen konsistenten Dump einspielen:

```
mysql -u root < full_dump.sql
```

Kontrolle und starten

- Kontrolle:

SHOW SLAVE STATUS\G

```
Slave_IO_State:  
  Master_Host: 192.168.1.42  
  Master_User: replication  
  Master_Port: 3306  
  Master_Log_File: bin-log.000001  
  Read_Master_Log_Pos: 51819  
  Relay_Log_File: relay-bin.000001  
  Relay_Log_Pos: 4  
  Slave_IO_Running: No  
  Slave_SQL_Running: No
```

- Starten des Slaves:

SLAVE START;

Probleme beim Aufsetzen

- **MySQL Replikation funktioniert grundsätzlich gut!**
- **Gründe warum Probleme auftreten:**
 - **Nicht sauberes befolgen der Anleitungen!**
 - **Nicht konsistentes Backup**
 - **OHNE `--single-transaction / --lock-all-tables` :-)**
 - **Jedes Schema einzeln (`--all-databases`) :-)**
- **MySQL Dokumentation: How to Set Up Replication**
<http://dev.mysql.com/doc/refman/5.5/en/replication-howto.html>

Probleme im Betrieb

- **Slave wird nicht überwacht**
- **Binary logs werden nicht aufgeräumt**

```
expire_logs_days = n  
PURGE BINARY LOGS TO 'bin-log.000013';
```
- **„Fummeln“ auf dem Slave**
- **Master/Slave Lag (hinterher hinken)**
- **Master/Slave Drift (auseinander laufen)**
→ Wenn Master und Slave auseinander laufen MUSS der Slave neu aufgesetzt werden
- **Filtern auf dem Master ist meist keine gute Idee!**

Master/Slave Drift

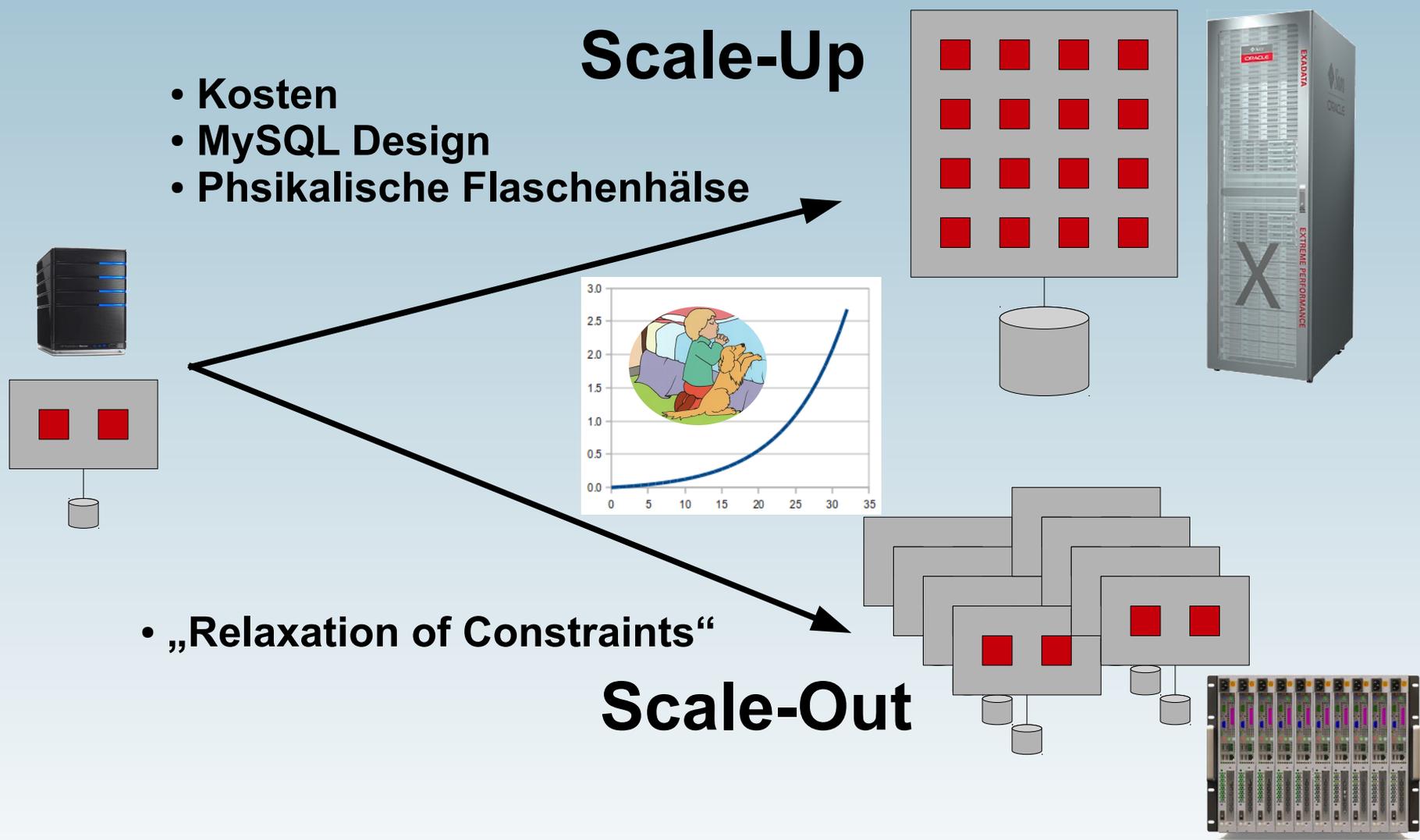
- **Master und Slave können auseinander laufen weil:**
 - Auf dem Slave geschrieben wird („rumfummeln“!)
 - `sql_bin_log = 0`
 - `sql_slave_skip_counter = 1`
 - `slave_skip_errors=...`
 - Nicht-deterministische Abfragen/Funktionen → viel besser mit Row Based Replikation (RBR)
 - Filtern auf Master oder Slave
 - Temporäre Tabellen mit SBR
- **Unterschiede können gefunden werden mit Tools aus dem Maatkit:**
 - `mk-table-checksum`
- **und gefixed werden mit:**
 - `mk-table-sync`
- **Ansonsten → Slave neu aufsetzen!!!**

Wann brauche ich die MySQL Replikation?



- **Es gibt verschiedene Szenarien:**
 - **Scale-Out Lösungen**
 - **Backup Slave**
 - **Datenanalyse, Reporting**
 - **Hochverfügbarkeit (HA)**
 - **(Zeitversetzte Datenstände vorhalten)**

MySQL Scale-Out vs Scale-Up



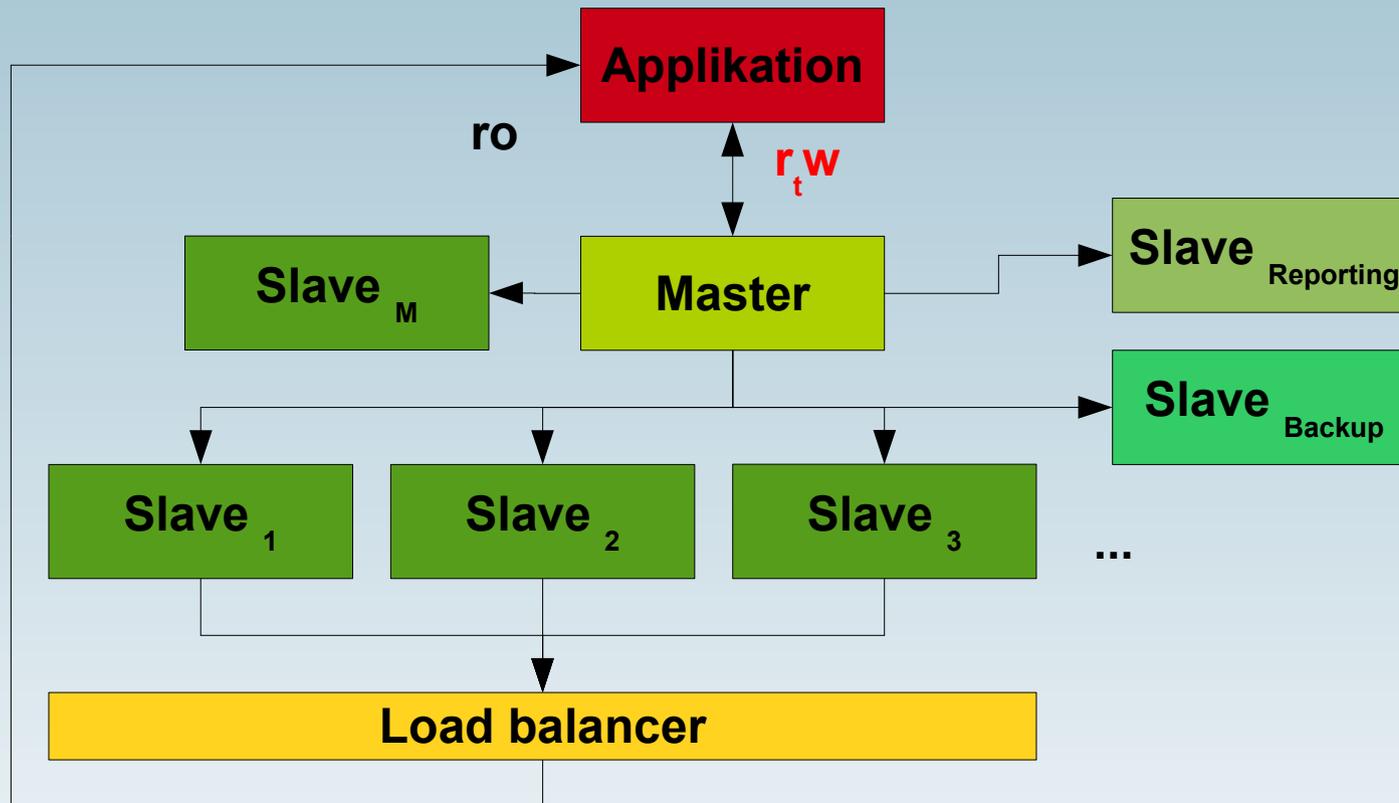
- Kosten
- MySQL Design
- Phsikalische Flaschenhalse

- „Relaxation of Constraints“

Scale-Up

Scale-Out

Der MySQL Scale-Out Ansatz



Neuerungen in MySQL 5.1, 5.5 und 5.6



www.fromdual.com

- **Row-Based Replikation (5.1)**
- **Semi-Synchrone Replikation (5.5)**
- **Transaktions Basierte Replikation (5.6)**
- **Row Image Control (5.6)**
- **Crash Safe Binary Logs (5.6)**
- **Parallel Slave Worker Threads (5.6)**
- **...**

Row-Based Replikation (5.1)

- **Alt (≤ 5.0) Statement-Based Replikation (SBR)**
 - Statements werden übermittelt
 - **Problem: nicht-deterministische Abfragen/Funktionen**
 - inkonsistente Daten zwischen Master und Slave
- **Neu (≥ 5.1) Row-Based Replikation (RBR)**
 - Events / Trx werden übermittelt
 - `binlog_format = {ROW | MIXED | STATEMENT}`
 - **RBR ist die sicherste Art der Replikation!**
 - **Wie sehe ich trotzdem was passiert?**
`mysqlbinlog --base64-output=DECODE-ROWS
--verbose bin-log.000069`
 - **Mehr binlog Traffic, unterschiedliche Performance → Testen!**

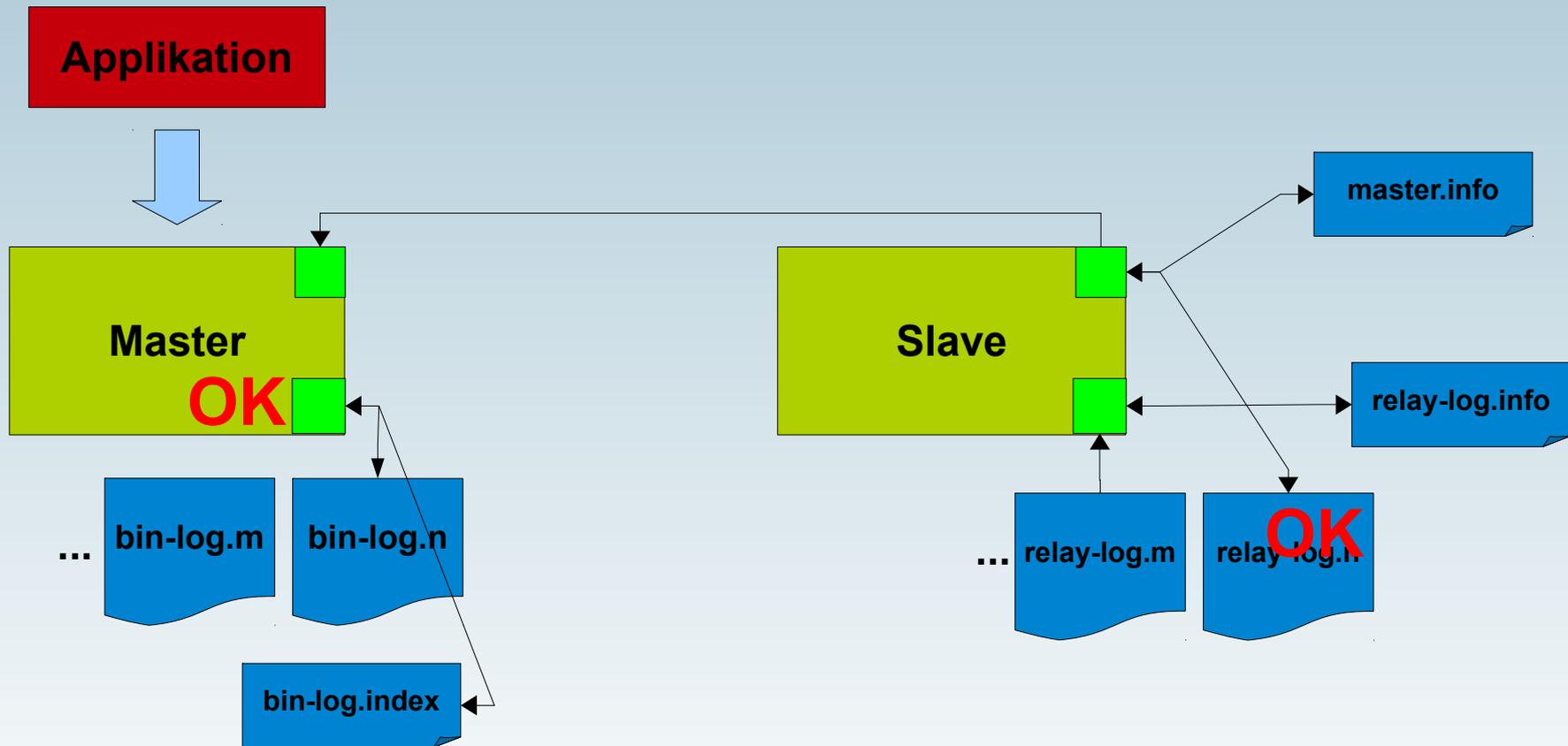
Semi-synchrone Replikation (5.5)



- Default **a**synchrone Replikation
 - Master wartet NICHT auf Slave!
 - Bei Crash: Trx ist nicht zwingend auf Slave
- Neu (5.5) optional **s**emi-synchrone Replikation
 - Plug-in (muss auf Master UND Slave aktiv sein!)
 - Master wartet auf Slave bis Timeout!
 - Nach Timeout (default 10 s) → Fallback auf asynchron
 - Bis Slave in Relay Log (sync) geschrieben hat
 - Bessere Datenintegrität (Master + mind. 1 Slave)
 - Schlechtere Performance (Commit + NW Roundtrip + Commit)
 - Master Commit, dann Crash, möglich dass Trx Slave nicht erreicht hat!

Semi-synchrone Replikation

- Wie funktioniert die semi-synchrone Replikation?



Semi-synchrone Replikation

- Plug-ins aktivieren:

```
INSTALL PLUGIN rpl_semi_sync_master SONAME 'semisync_master.so';
```

```
INSTALL PLUGIN rpl_semi_sync_slave SONAME 'semisync_slave.so';
```

- Prüfen ob erfolgreich:

```
SHOW PLUGINS;
```

Name	Status	Type	Library
rpl_semi_sync_master	ACTIVE	REPLICATION	semisync_master.so
rpl_semi_sync_slave	ACTIVE	REPLICATION	semisync_slave.so

- Semi-synchrone Replikation einschalten:

```
SET GLOBAL rpl_semi_sync_master_enabled = 1;
```

```
SET GLOBAL rpl_semi_sync_slave_enabled = 1;
```

- Slave (neu) starten:

```
STOP SLAVE IO_THREAD; START SLAVE IO_THREAD;
```

Replikation in MySQL 5.6 (1)

- **Transaktions Basierte Replikation (TBR?)**
 - Global Transaction Identifiers (GTID)
 - Keine M/S Inkonsistenzen mehr möglich!?!
 - Noch etwas wenig geschmeidig / komplizierter :-)
- **Row Image Control**
 - `binlog_row_image = {full | minimal | noblob}`
- **Parallele Worker Threads**
 - `slave_parallel_workers = <n>`
 - Parallelisieren pro Schema
 - Gut für Hosts mit Traffic auf vielen Schemata!

Replikation in MySQL 5.6 (2)

- **Crash safe Binary Logs**
 - Vollständige Events/Trx werden geloggt
 - Event-Länge + CRC32 Checksumme
- **Slave: `master.info` und `relay-log.info` zusätzlich in (MyISAM) Tabellen**
 - `slave_master_info` + `slave_relay_log_info`
 - **Umwandeln in InnoDB!**

Replikation in MySQL 5.6 (3)

- **Remote Binary Log Shipping:**

```
mysqlbinlog --read-from-remote-server  
--raw bin-log.000001 > bin-log.000001.dup
```

- **Delayed Replication**

```
CHANGE MASTER TO MASTER_DELAY=n;
```

- Früher `mk-slave-delay` (Maatkit)

- **Slave kann auf ein Netzwerk Interface gebunden werden:**

```
MASTER_BIND='eth1'
```

Synchrone Replikation mit Galera

- Codership (www.codership.com)
- Synchrone Replikation
 - “Certification Based Replication”, kein 2PC!
- Echte Multi-Master Replikation
- Skaliert sowohl lesend als auch schreibend

→ mehr dazu Erkan & Alex/Seppo

Q & A



www.fromdual.com



Fragen ?
Diskussion?

Wir haben noch Zeit für ein persönliches Gespräch...

- **FromDual bietet neutral und unabhängig:**
 - **MySQL Beratung**
 - **Remote-DBA für MySQL**
 - **Support für MySQL und Galera**
 - **MySQL Schulung**

www.fromdual.com/presentations