MySQL-Server im Teamwork - Replikation und Cluster

Administrator-Runde Berlin, 2016-Jan-7

Jörg Brühe

Senior Support Engineer, FromDual GmbH

joerg.bruehe@fromdual.com

FromDual GmbH

www.fromdual.com

Support

ORACLE Silver Partner

Beratung

remote-DBA

Schulung

Zur Person

- **Entwicklung verteiltes SQL-DBMS:**
  Unix-Portierung,
  Anschluss Archivierungs-Tools (ADSM, NetWorker)
- **MySQL Build Team:**
  Release-Builds inkl. Tests, Paketierung, Skripte, ...
- **DBA:**
  MySQL für eine Web-Plattform
  (Master-Master-Replikation)
- **Support-Ingenieur (FromDual):**
  Support + Remote-DBA für MySQL / MariaDB / Percona
  mit oder ohne Galera Cluster

Inhalt

MySQL Server: Architektur

Binlog

Replikation

Galera Cluster

Vergleich

Beispiele / Wann was (nicht)

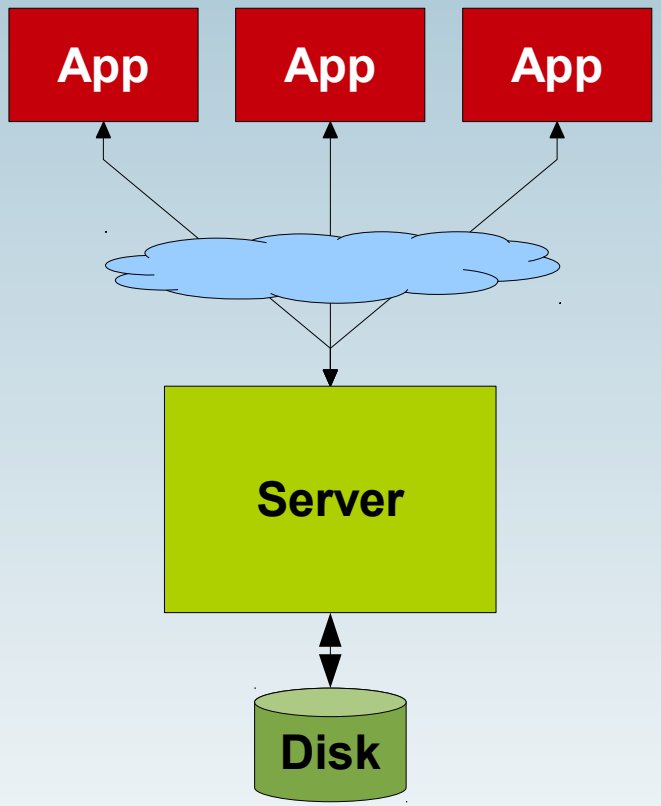
Allgemeines

- Konzepte, nicht Details:
  „der Wald, nicht die Bäume“
- MySQL 5.5 / 5.6 (aktuelle GA-Versionen)
- Übertragbar von MySQL (Oracle) auf Percona und MariaDB

- Nicht anwendbar auf „embedded“ MySQL
- Nicht betrachtet: NDB = „MySQL Cluster“

- **MySQL Server: Architektur**
  - Binlog
  - Replikation
  - Galera Cluster
  - Vergleich
  - Beispiele / Wann was (nicht)

Client-Server-DBMS



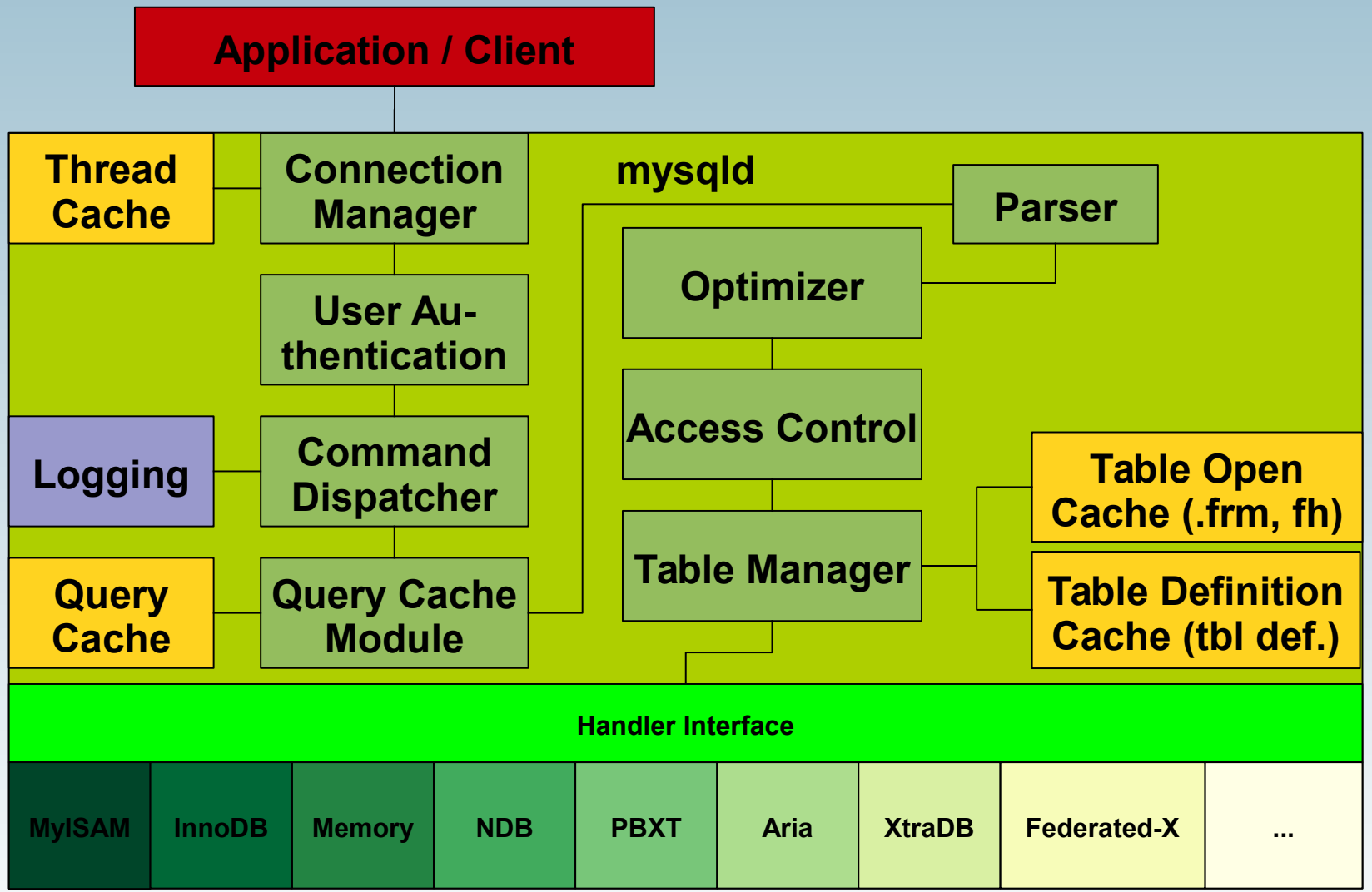
Client (Applikation)
lokal oder remote

Socket, LAN oder Internet

Server ist eigener Prozess
Multi-threaded:
1 Thread je Session

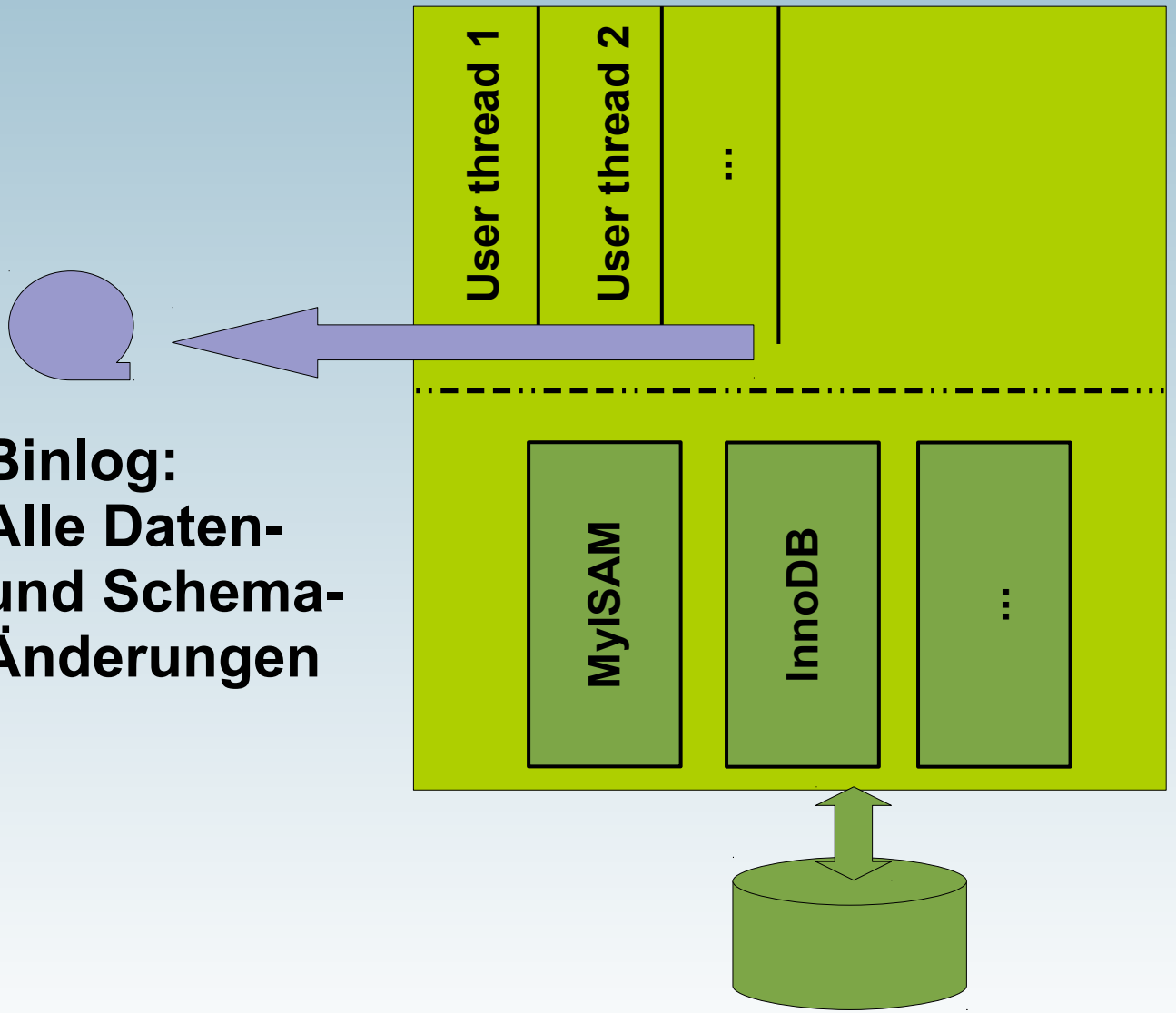
Platte / SSD, lokal oder SAN

Server intern



- MySQL Server: Architektur
- ➡ **Binlog**
- Replikation
- Galera Cluster
- Vergleich
- Beispiele / Wann was (nicht)

Ebenen + Binlog



Binlog:
Alle Daten- und Schema-Änderungen

SQL-Ebene:

- Parser
- Optimizer
- Privilegien
- Query Cache
- …

Handler Interface

Datei-Ebene:

- Tabellen-Handler
- InnoDB:
  - Satz-Zugriffe
  - Satz-Sperren
  - Recovery
  - ...

Binlog

- Alle ausgeführten Daten-Änderungen
- Alle ausgeführten Schema-Änderungen
- Zeitstempel
- Zwingend für Point-in-Time-Recovery „PITR“
- Unabhängig von Tabellen-Handler
- Formate „statement“, „row“ und „mixed“
- Segmente mit konfigurierbarer Größe
- Fortlaufend nummeriert

- MySQL Server: Architektur
- Binlog
- ➡ **Replikation**
- Galera Cluster
- Vergleich
- Beispiele / Wann was (nicht)

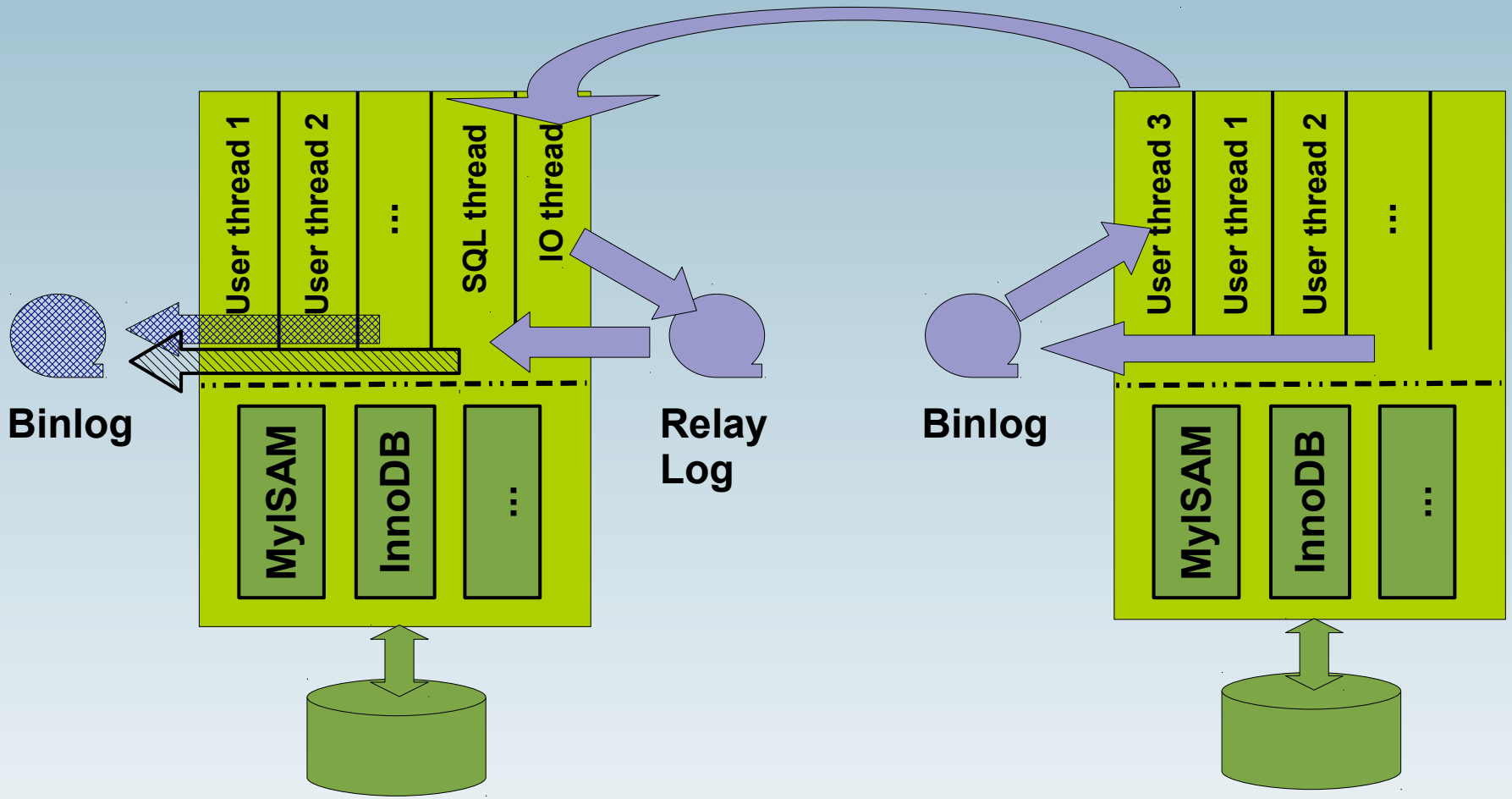
Replikation bei MySQL

- Anwendungen kommunizieren mit „Master“
- „Master“ protokolliert alle Änderungen
- „Slave“ hat identischen Anfangszustand

- Slave holt alle Änderungen vom Master und wendet sie bei sich an

- Replikation läuft asynchron
- Slave stoppt Replikation bei Abweichung

Slave holt Binlog vom Master



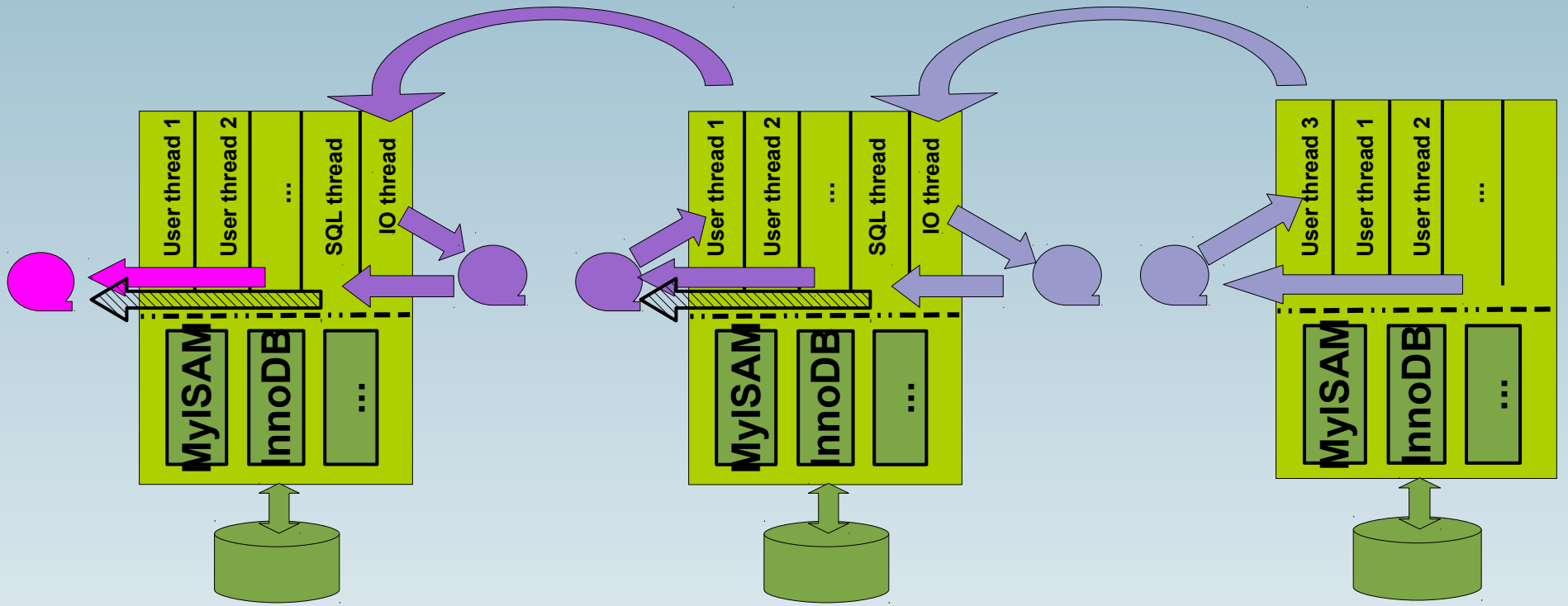
Slave:
“log-bin = FILE”, sonst kein Binlog
“log_slave_updates = 1” für Weiterleitung

Master:
“log-bin = FILE”, sonst kein Binlog
(keine Master-Funktion)

Typische Anwendungen

- „High Availability“
- Geo-Redundanz
- Höhere Lese-Last unterstützen
  (= „read scale-out“)
- Read-Only-Instanz(en)
  für z.B. Backup oder Reports
- Verzögerte Replikation ist möglich
- Filterung (nach DB oder Tabelle) ist möglich

Replikations-Kaskade



- **Empfehlung: „read-only = 1“ auf Slave „log_slave_updates = 1“**
- **mehrere Slaves an einem Master möglich**

Einträge im Binlog

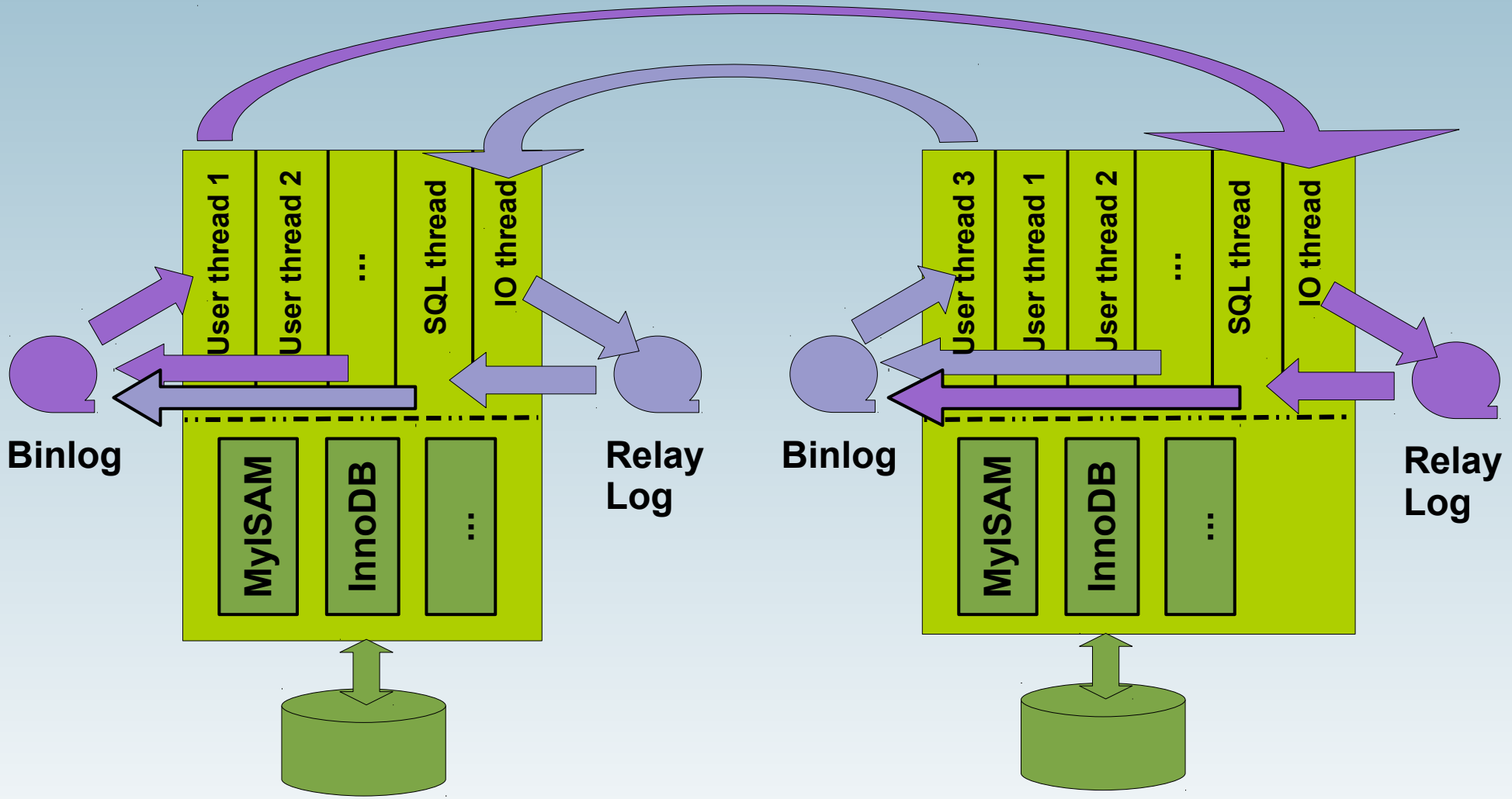
Ursprünglich:

- Identifikation durch Filename und Position
- Replikation: „`change master to ...`“ mit Host, Port, User, Password, File, Position
- Siehe auch „`mysqldump --master-data`“

Ab MySQL 5.6:

- GTID = „Global Transaction ID“
- Replikation: „`change master to ...`“ mit Host, Port, User, Password und „`auto_position = 1`“

Master-Master-Replikation



- **Überlappende Änderungen sind fatal!**

Anmerkungen zur Replikation

- Master-Master ist umstritten, Vorsicht!
- Replikation erhöht den Lese-Durchsatz, aber nicht/kaum den Schreib-Durchsatz
- Replikation hat File-IO und Netzlast
- Format „row“ ist effizienter, aber weniger lesbar
- Mit MySQL 5.7 ist Multi-Master-Replikation möglich
- Große Installation: booking.com
- Lese-Tipp (Giuseppe Maxia, August 2015): `datacharmer.blogspot.de`

MySQL Server: Architektur

Binlog

Replikation

➡ **Galera Cluster**

Vergleich

Beispiele / Wann was (nicht)

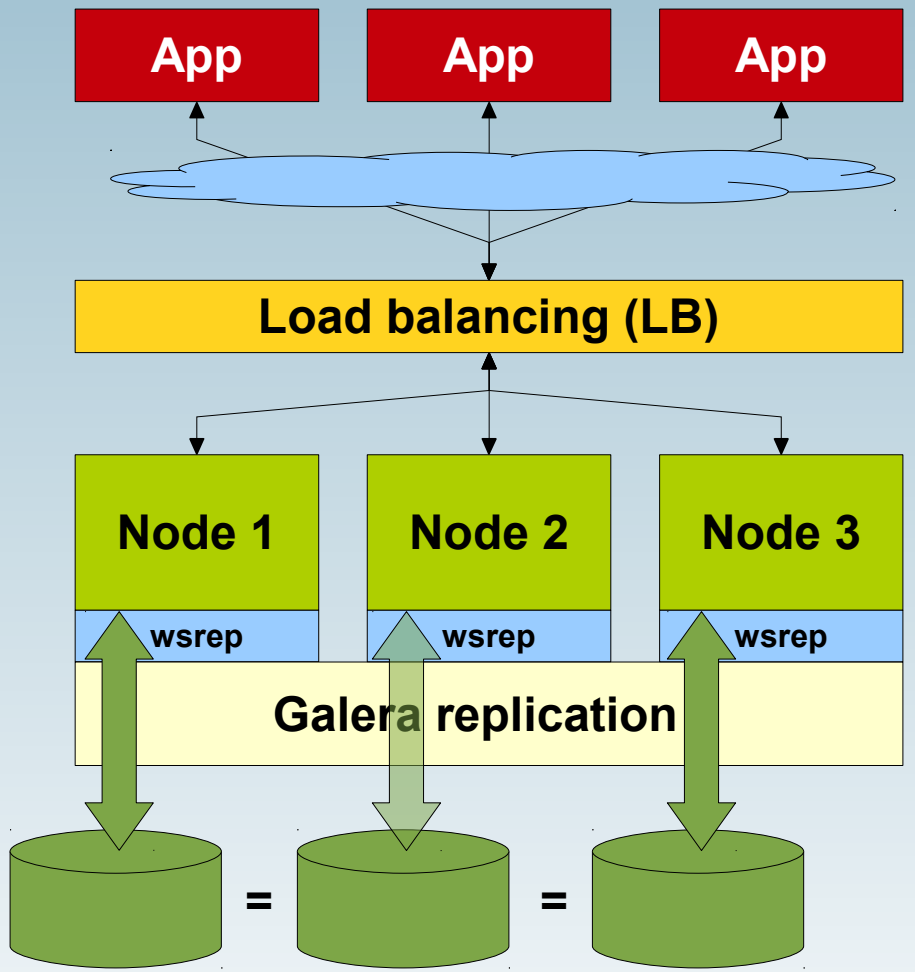
Schwächen der Replikation

- Asynchron
- Asymmetrisch
- Nur ein Schreib-Knoten
- Paralleles Schreiben verursacht Abbruch
- HA braucht Failover nach Knoten-Ausfall
- Jeder Knoten ist SPOF für seine Slaves, Ausfall erzwingt Struktur-Änderung
  (Erleichterung in 5.7 durch Multi-Source-Replikation)
- Dynamische Änderungen sind schwierig

Bessere Alternative

- Synchrone Übertragung
- Symmetrischer Cluster
- Schreibzugriffe überall möglich
- Verteilte Konflikt-Analyse und -Behebung
- HA durch Kontinuität nach Knoten-Ausfall
- Dynamischer Eintritt / Austritt möglich

Galera Cluster



“shared nothing” Architektur

Inklusive Ausfall-Erkennung
und Redirection für HA

“Working Set Replication”

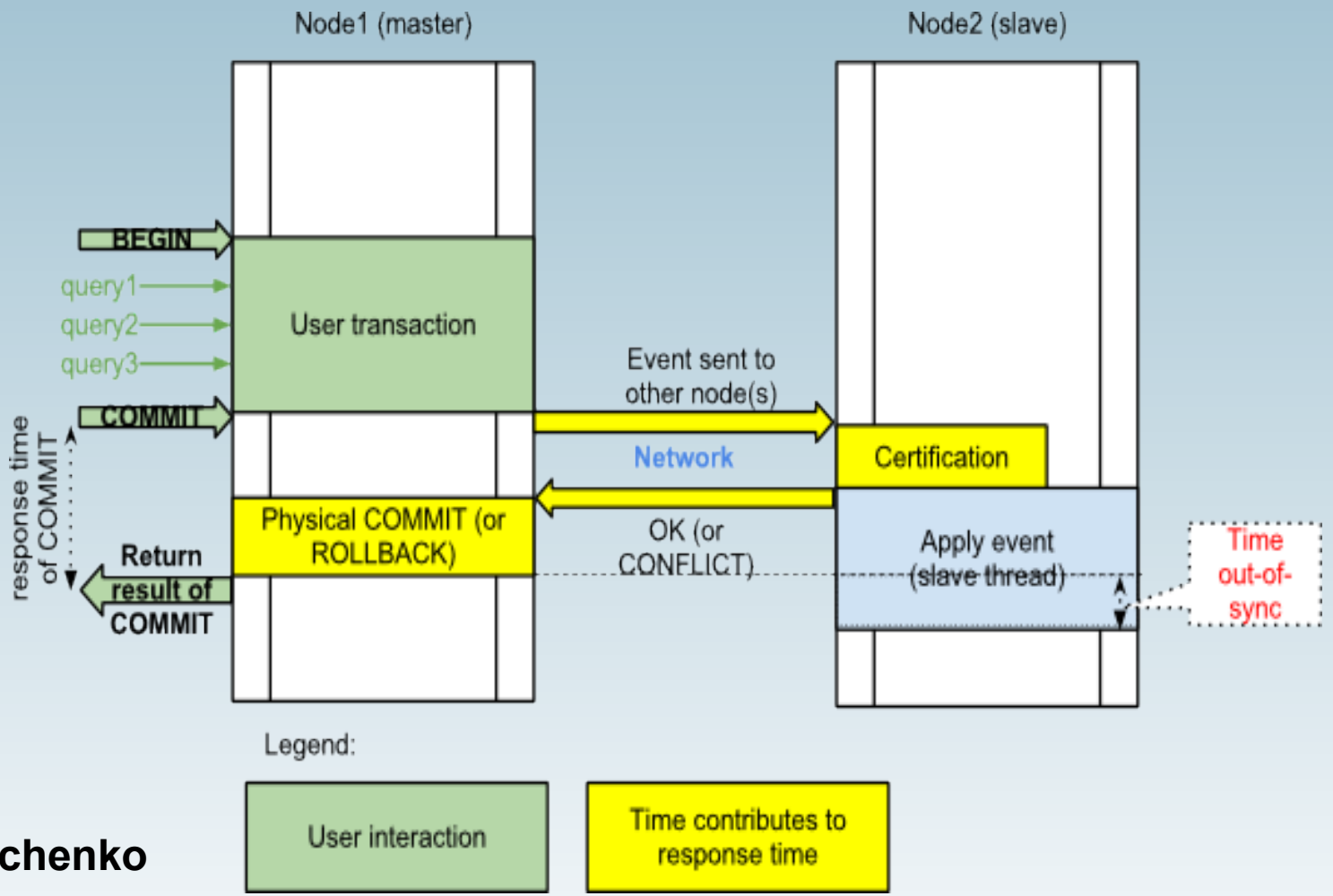
Vorzugsweise eigenes Netz

lokale Platten,
jeweils Daten komplett

Eigenschaften von Galera (1)

- + Basiert auf InnoDB (wg. Transaktionen und Rollback)
- + Überträgt auch Benutzer-Definitionen usw.
- + Quasi-synchrone Übertragung beim Commit, Prüfung auf Konflikt-Freiheit, effizient
- + Symmetrisch, HA ohne Server-Failover, Quorum
- + Kein Transaktions-Verlust
- + Scale-Out für Lesen, auch mehr Schreiben
- + Dynamischer Eintritt / Austritt möglich, automatische Synchronisation

Ablauf



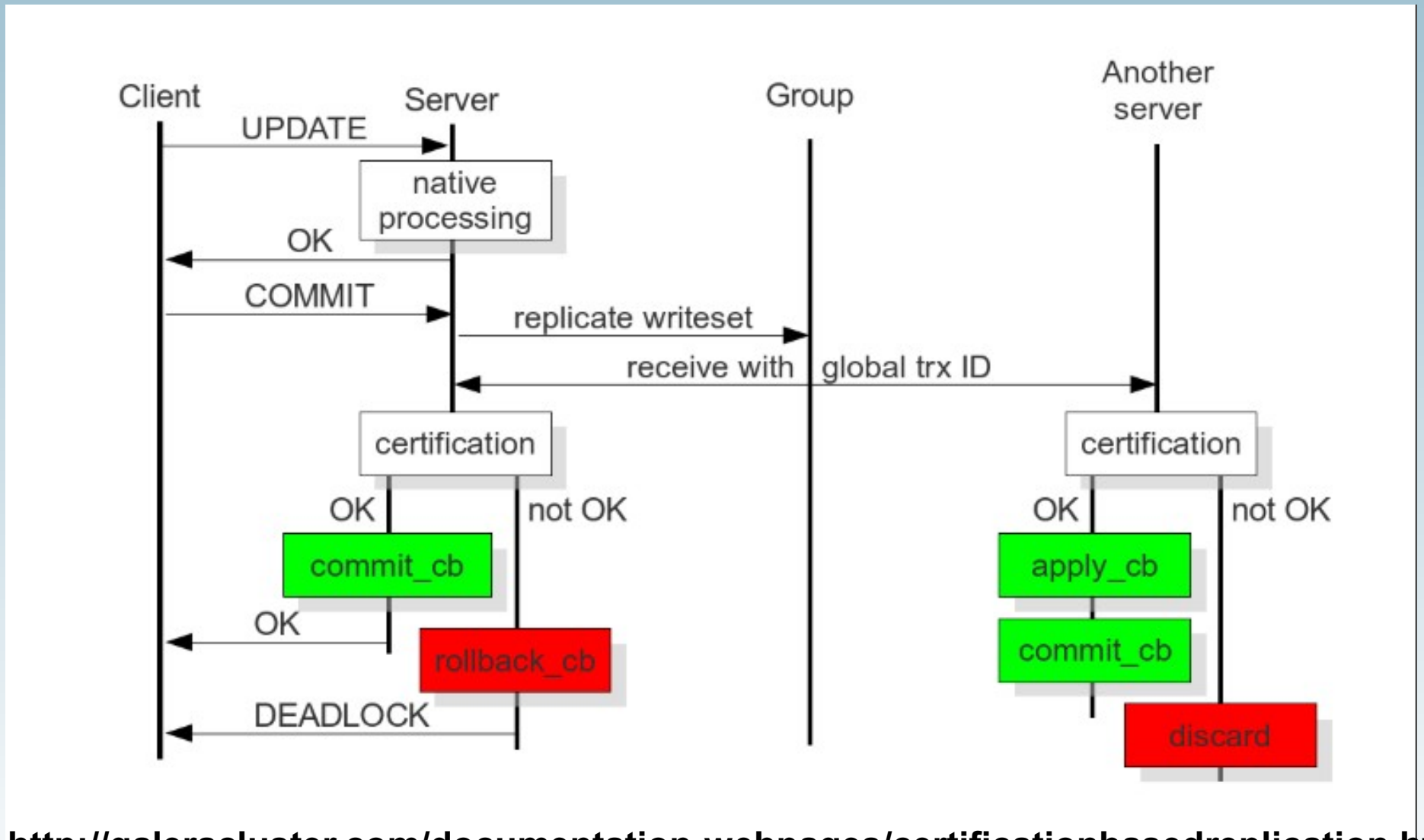
Graph by Vadim Tkachenko (Percona):

http://www.mysqlperformanceblog.com/2012/01/19/percona-xtradb-cluster-feature-2-multi-master-replication/

Eigenschaften von Galera (2)

- Patch der MySQL-Quellen
  (Codership bietet Binaries, auch MariaDB und Percona)
- Vorsicht bei Hot Spots (Zeilen)
- Späte Konflikt-Erkennung, kompl. Rollback
  (Prüfung erst bei Commit)
- Mindestgröße drei Knoten
- Synchronisations-Dauer bei großer DB
  (mysqldump -> xtrabackup oder rsync)

Zertifizierung bei Commit



http://galeracluster.com/documentation-webpages/certificationbasedreplication.html

MySQL Server: Architektur

Binlog

Replikation

Galera Cluster

➡ **Vergleich**

Beispiele / Wann was (nicht)

MySQL-Server im Teamwork

- Alternativen: Replikation oder Cluster

- Redundanz bei Maschine und Storage
- HA
- Scale-Out, besonders für Lese-Last
- Instanzen für Reports, Analyse, Backup
- Daten lokal verfügbar (Filialen, ...)

Vergleich (1)

Replikation		Galera
Standard		Zusatzprodukt
Alle Handler		InnoDB
Aufwärts-kompatibel		gleiche Versionen
Mind 2 Knoten		Mind 3 Knoten
HA durch Failover		HA ohne Änderung
Kommunikation:		
Hierarchisch, Kette		symmetrisch, parallel
asynchron		Quasi-synchron
Verzögerung möglich		sofort
Filtern möglich		alles

Vergleich (2)

Replikation		Galera
Lese-Scale-Out		Lese-Scale-Out
Schreiben konst.		Schreiben erhöht
1 Master:		
1* Write		1* Write
Konflikt lokal:		
Fehler bei Statement		Fehler bei Statement
n Master:		
n* Write		n* Write
Konflikt verteilt:		
Replikations-Abbruch		Rollback bei Commit

Vergleich (3)

Replikation		Galera
kurze Unterbrechung:		
Replikation fortsetzen		IST (inkrementeller Transfer)
lange Unterbrechung:		
Replikation fortsetzen		SST (kompletter Transfer)
Struktur-Änderung:		
Manuell / Zusatz-Tool		Automatisch / dynamisch
Aufsetzen:		
Schnappschuss,		Komplett-Transfer,
Master bleibt verfügbar		Donor tlw. Blockiert

CAP-Theorem

- C = Consistency (gleiche Daten überall)
- A = Availability (das System antwortet)
- P = Partition Tolerance (Netzwerk-Ausfall)

„In einem verteilten System ist es unmöglich, gleichzeitig die drei Eigenschaften Konsistenz, Verfügbarkeit und Partitionstoleranz zu garantieren.“

https://de.wikipedia.org/wiki/CAP-Theorem

MySQL Server: Architektur

Binlog

Replikation

Galera Cluster

Vergleich

➡ **Beispiele / Wann was (nicht)**

Kommunikations-Ausfall (1)

Galera Cluster:

- Isolierter Knoten hat kein Quorum
  => nicht benutzbar
- Quorum ist gefährdet!
- Aktive Knoten schreiben „gcache“ als Files, Aufbewahrungsdauer?
- Umschaltung auf SST droht

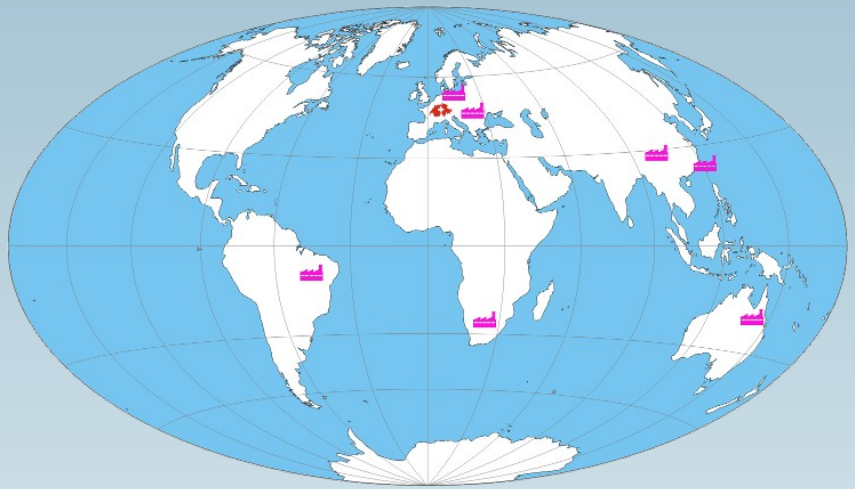
Kommunikations-Ausfall (2)

Replikation:

- **Master schreibt Log-Segmente als Files**
- **IO-Thread will von Binlog-Position / GTID lesen, probiert periodisch bis Erfolg**
- **„purge log“ vermeiden!**

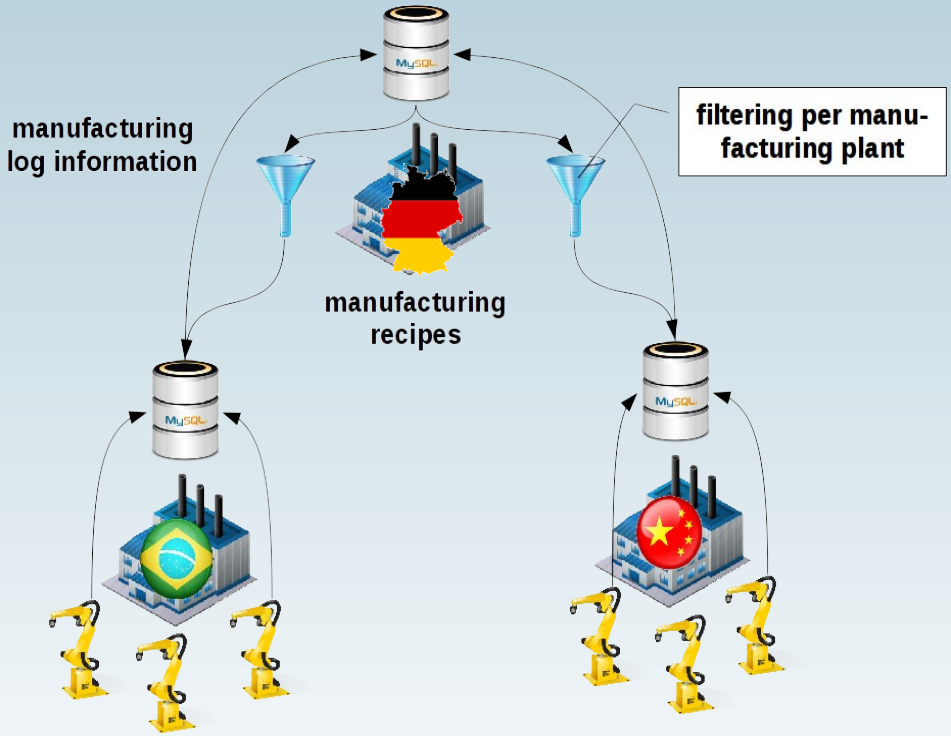
Replikation ist toleranter als Galera Cluster !

Beispiel: Globale Produktion



Anforderung:
Zentrale (D) und Werke (BR, CN, ...) mit selektiver Übertragung

Lösung: Replikation mit Filterung



Paralleles Schreiben + Konflikt

Galera:

- **Retry von autocommit-Statements möglich**
- **Transaktions-Konflikt führt zu Rollback**
  => Wiederholung durch Applikation

Replikation:

- **Slave bemerkt, kein Kontakt zur Applikation**
  => Replikation bricht ab

Replikation braucht Admin-Eingriff bei Konflikt !

Hot Spot

- Replikation: Häufig Abbrüche
- Galera: Hohe Rollback-Häufigkeit

=> Einen Schreib-Knoten auswählen !

Hochverfügbarkeit

Replikation:

- **Failover manuell (Reaktionszeit) oder automatisch (korrekt?)**
- **Slave Lag, Master-Auswahl**

Galera:

- **Symmetrisch, kein Rollenwechsel**
- **Virtuell-synchrone Replikation (kein Lag)**

=> Vorteil Galera

Q & A

Fragen ?

Diskussion?

Wir haben Zeit für ein persönliches Gespräch ...

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

www.fromdual.com/presentations