Replication Smackdown

Ronald Bradford

http://ronaldbradford.com

2016.10
Replication
   What, Why, How, MySQL Types
Production Systems
   Requirements, Needs, MTBF
A New Mindset
   Availability, Classification, Pipeline
Which is the best replication approach to use?

Answer: It depends
Which is the best replication approach to use?

Answer: It depends
What is replication?
What?

- Full
- SET

Tuesday, October 4, 16
• Copy of data
• Full or partial
Copy of data
- Full or partial
Translation of data
  - e.g. Oracle to MySQL
- Copy of data
  - Full or partial
- Translation of data
  - e.g. Oracle to MySQL
- Transformation of data
  - e.g. MySQL to DW
  - e.g. MySQL to Hadoop
ONE WAY FORM

CHAIN

FAN-OUT

FAN-IN

CIRCULAR
Why use replication?
Redundancy (copies)

Availability (distribution)

Failover

Scalability (read/write)

Performance (optimizations)

Backups (locking, load)

Consolidation
What are production requirements?
PROD REQS

- Acceptable latency
- Acceptable throughput
- Performance under load
- High availability
- Failover
- Disaster recovery
- Security

- Backups
- Load testing
- Monitoring
- Alerting
- Sizing

How does replication help achieve production requirements?
What does MySQL offer for replication?
[Traditional] MySQL Replication
- Asynchronous / Semi-synchronous
MySQL Cluster
- Galera (MySQL/Percona/MariaDB)
MySQL Group Replication (RC) / InnoDB Cluster (TBD)
Amazon RDS MAZ & Aurora
Others (e.g. Google Cloud SQL, Clustrix, DRBD)
What are barriers to usage?
Traditional

- async - since 3.23
  - lag
  - drift
  - consistency
  - throughput

- semi-sync - since 5.5
  - lag
  - drift
  - consistency
  - throughput

@RonaldBradford
#PerconaLive

EffectiveMySQL.com - Performance, Scalability, Site Reliability

Tuesday, October 4, 16
MySQL Cluster

Additional installation complexity
Data/SQL/Admin nodes
Different admin interface
Different backup strategy
LAN based
Same SQL syntax
Limited large join options
Syntax limitations
Feature limitation (e.g. MEMORY table), primary key
OS Limitations (Linux Only)
Hot data spots / Large transactions in multi-master write
Timeouts
Schema upgrades
LAN v WAN
Group Replication

- Syntax Limitations, Feature Limitation
- Hot Spots/Large transactions in multi-master write
- Supported on Linux, Windows, Solaris, FreeBSD, OSX

Requirements

- MySQL 5.7, GTID, binlog_format=ROW
- Other configuration settings
- RC only
- Only in labs
- Based on group replication limitations
- Helps solve the routing problem
- Simplified orchestration in JS
What are improvements to replication?
5.6
- crash save slaves, GTID, group commit, multi-threaded

5.7
- semi-sync improvements, multi-threaded V2, multi-source, XA support, group replication
MySQL
Percona
MariaDB
Custom
Others

Does not improve async/semi sync replication?
Improves [faster] failover


http://code.openark.org/blog/mysql/refactoring-replication-topology-with-pseudo-gtid
- parallel schema applier (5.6)
- parallel query applier (5.7)

Does improve async/semi sync replication?
Improves performance (i.e. lag)
Does not eliminate lag
How do we apply replication to our systems?
What are your business needs?
What are your business needs? What are [ideal] business needs?
What are your business needs?
What are [ideal] business needs?
What are [acceptable] business needs?
Mean Time Between Failure (MTBF)
Mean Time To Detect (MTTD)
Mean Time To Recover (MTTR)
Recovery Point Objective (RPO)
Recovery Time Objective (RTO)
The new mindset in architecture
1. Availability
2. Classification
3. Pipeline
Data Availability
DATA AVAILABILITY

- Ability to write data
- Ability to read data
- Ability to [read|write] cached data
- Ability to operate with no data
What do I mean?
What do I mean?

What is your definition of downtime?
Not database availability
  e.g. those maintenance windows

Not data availability
  e.g. Write/Read/Cache/None

It is all about service availability
  i.e. endpoints
Data Class
@RonaldBradford
#PerconaLive

EffectiveMySQL.com - Performance, Scalability, Site Reliability

Tuesday, October 4, 16
All data in a RDBMS schema (or even table) is not equal

- Users data (register, modify)
- Change password
- Last login date
- Add content
- Comment/Rate/Score
All data in a RDBMS schema (or even table) is not equal

- Users data (register, modify)
- Change password
- Last login date
- Add content
- Comment/Rate/Score

- Current Order
- Last Order
- Historical Orders
- Credit Card Details
Some data needs to be more highly available than other data
Some data access requires more responsiveness than others
Some data has acceptable data loss
Some data can be unavailable some of the time
Some data visibility can vary between users
All data should be secure, some more secure
Some data needs to be more highly available than other data
Some data access requires more responsiveness than others
Some data has acceptable data loss
Some data can be unavailable some of the time
Some data visibility can vary between users
All data should be secure, some more secure

Reclassification of data changes replication requirements
**Requirements**

- Acceptable latency
- Acceptable throughput
- Performance under load
- High availability
- Failover
- Disaster recovery
- Security

**Responsibilities**

- MTBF
- MTTD
- MTTR
- RPO
- RTO
Example (Financial)
Single greatest feature loss

- Referential Integrity
- A & C of ACID
Data Pipeline
A single request does not produce one synchronous response
Data is not stored in one RDBMS or product type
Data locality for responsiveness
Use product strengths for data manipulation
Rethinking how to choose MySQL Replication

@RonaldBradford
#PerconaLive
UTF-FTV
you-twit-face-flix-talk-vr
OLD WAY

- Found in many “traditional” frameworks/OSS products
- Users table in monolithic schema
- Synchronous web requests for information
- Polling for new/streaming information
- Data for application is available or not available
- Replication enables read scalability only
NEW WAY STORES

- Highly available synchronous store
- Columnar store
- Messaging (PUB/SUB)
- Graph
- Queue
- Search
Microservices
- Login/Logout
- Register/Maintain
- Log actions (login good/bad, click, mouse movement)
- Friends
- Friends interactions
Graph Store
Friends
Customized for relevancy/strength algorithm
Queue
Password changes, User profile changes
Lost password
User actions (success, failure) are published (PUB)

(SUB) Subscriber logs information

(SUB) Subscriber audits for unexpected behavior

(SUB) Subscriber notifies friends user is online/offline
Rate a comment
- Client only feedback
- Action held on client
- batched, overloaded, timed transmission
- Supports rate/unrate (client side only)
Conclusion
Do you choose a replication approach to match your [ideal] business needs and data store(s)

OR

Do you architect a data infrastructure to meet your [ideal] business needs and target specific replication (aka availability) approaches where applicable
What does this have to do with replication?

Answer: Everything