Galera in MariaDB 10.4

State of the Art and Plans

Seppo Jaakola
Codership

- Seppo Jaakola
- One of the Founders of Codership

- Codership – Galera Replication developers
- Partner of MariaDB for developing and supporting MariaDB Galera Cluster
- Galera releases since 2009

Agenda

- Galera in 10.4 Status
- Galera Cluster Upgrading
- Streaming Replication
- XA Transaction Support
- Spider Cluster

Galera in 10.4 and Beyond

Galera 4.0

- Group Commit Support <refactor for MariaDB>
- Non Blocking DDL <testing>
- Huge transactions by streaming replication <testing>
- Inconsistency Voting Protocol <testing>

MariaDB 10.4

- Gcache Encryption <implementation>
- MariaDB GTID Compatibility <requirement>

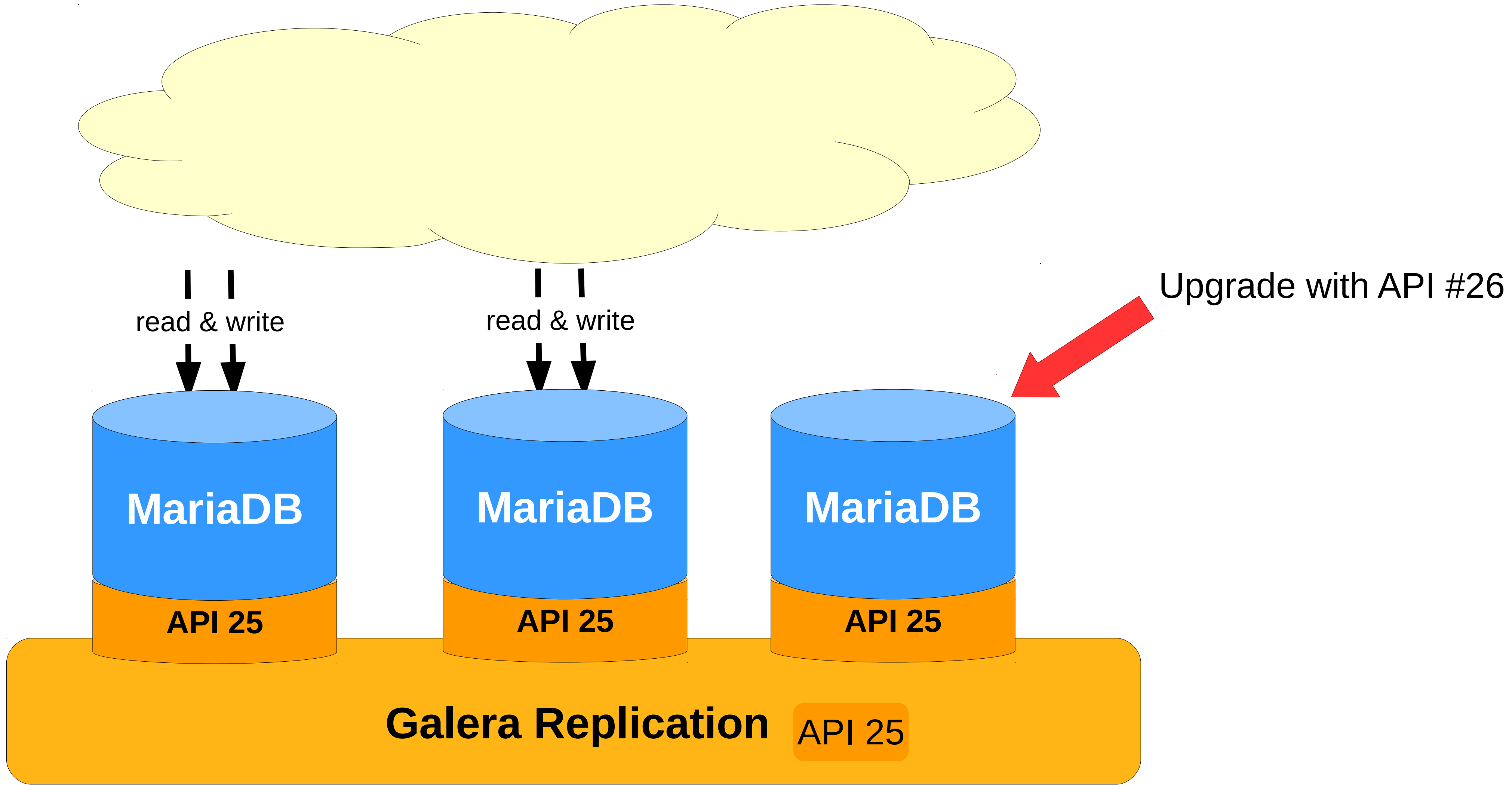
Galera 4.1

- XA transaction Support <implementation>
- Spider Cluster <design>

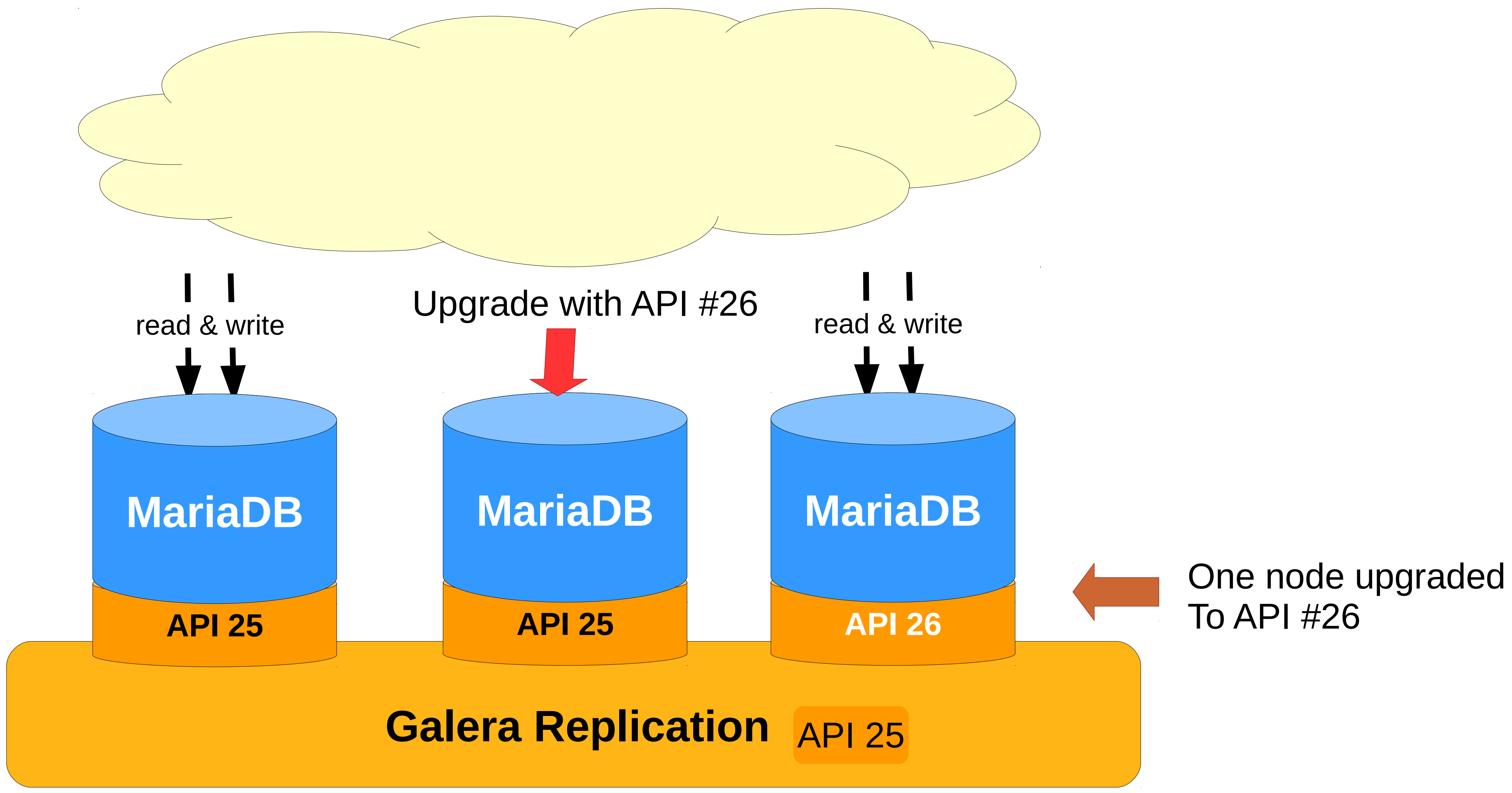
Galera Upgrade

wsrep API Change

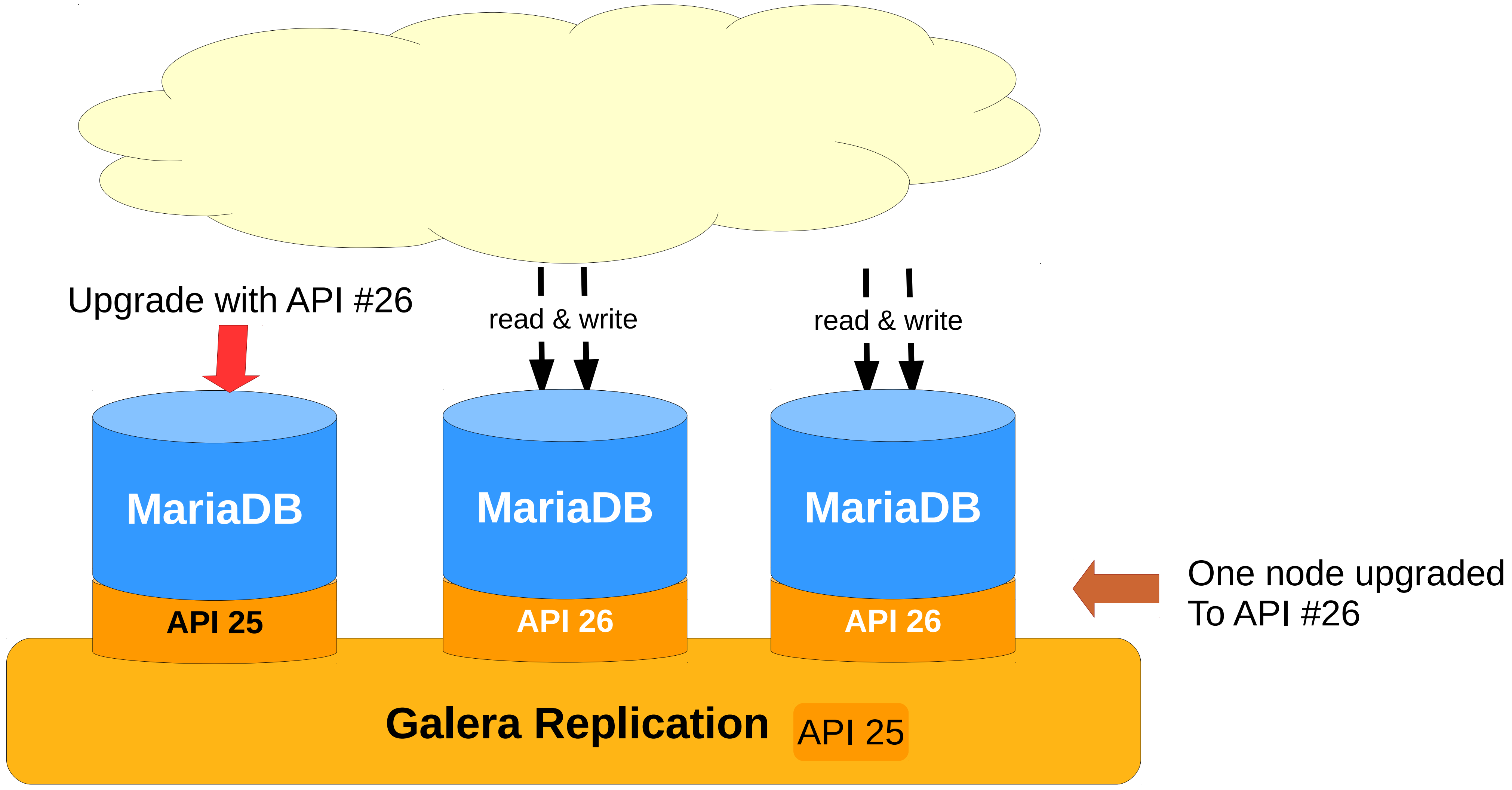
Galera Rolling Upgrades



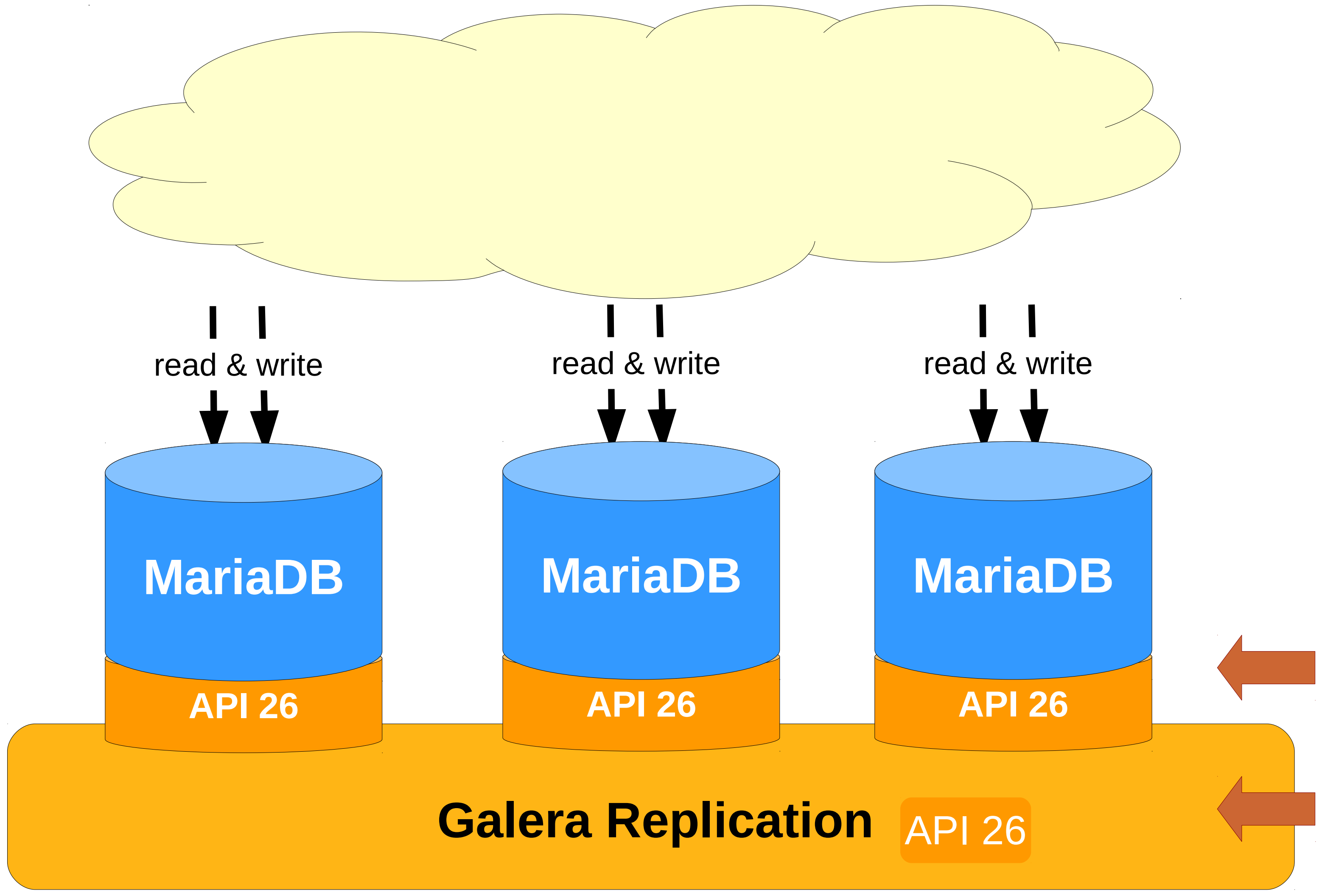
Galera Rolling Upgrades



Galera Rolling Upgrades



Galera Rolling Upgrades



All nodes upgraded
To API #26

API #26 features now
Enabled in replication

Streaming Replication

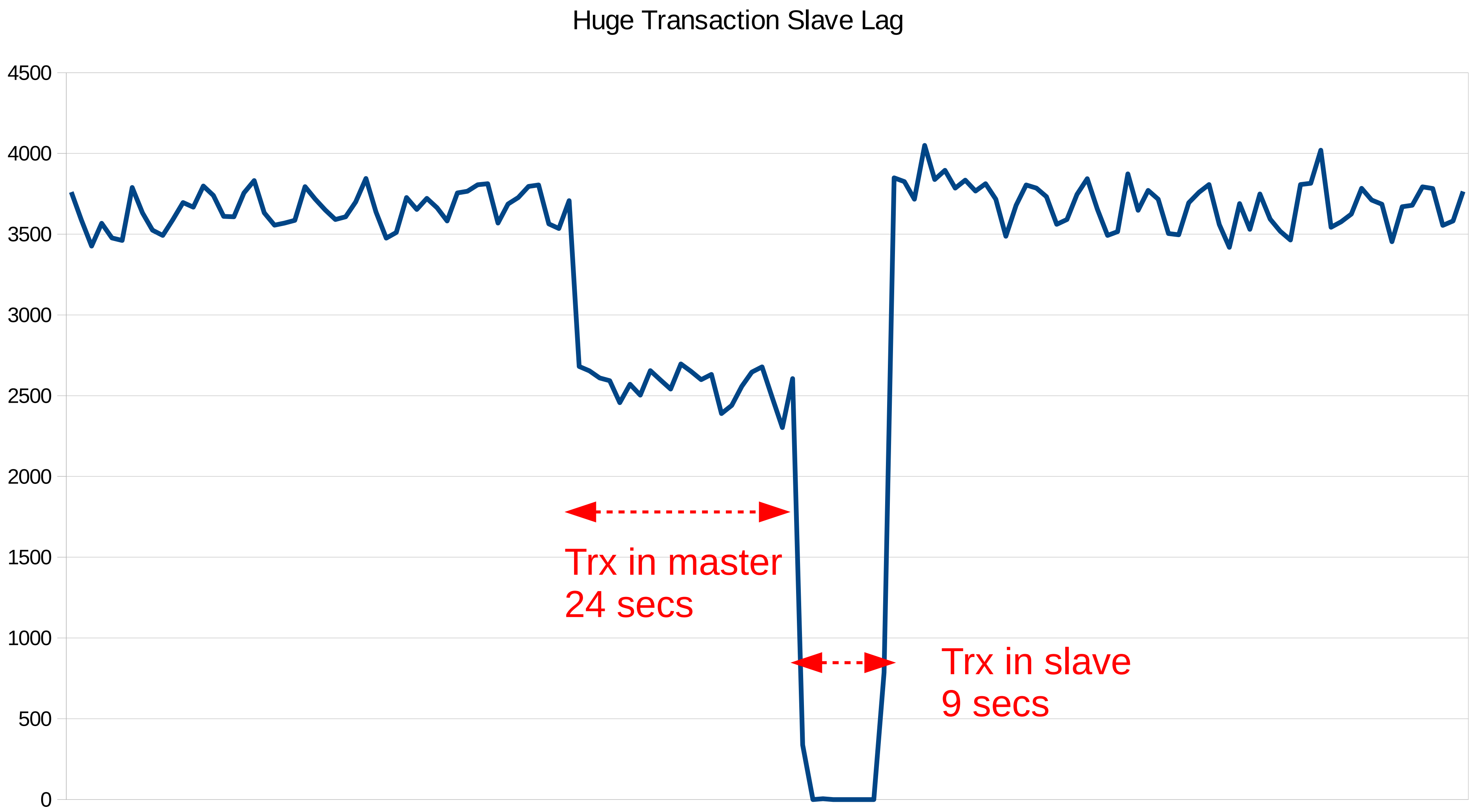
Huge Transaction Support

Huge Transaction Demo Setup

1. Two nodes
2. Steady load of pure autocommit updates to measure trx throughput
3. A huge table with ~1.5M rows
4. Run update on huge table to modify all rows

- → monitor trx/sec rate in the cluster when the huge transaction kicks in

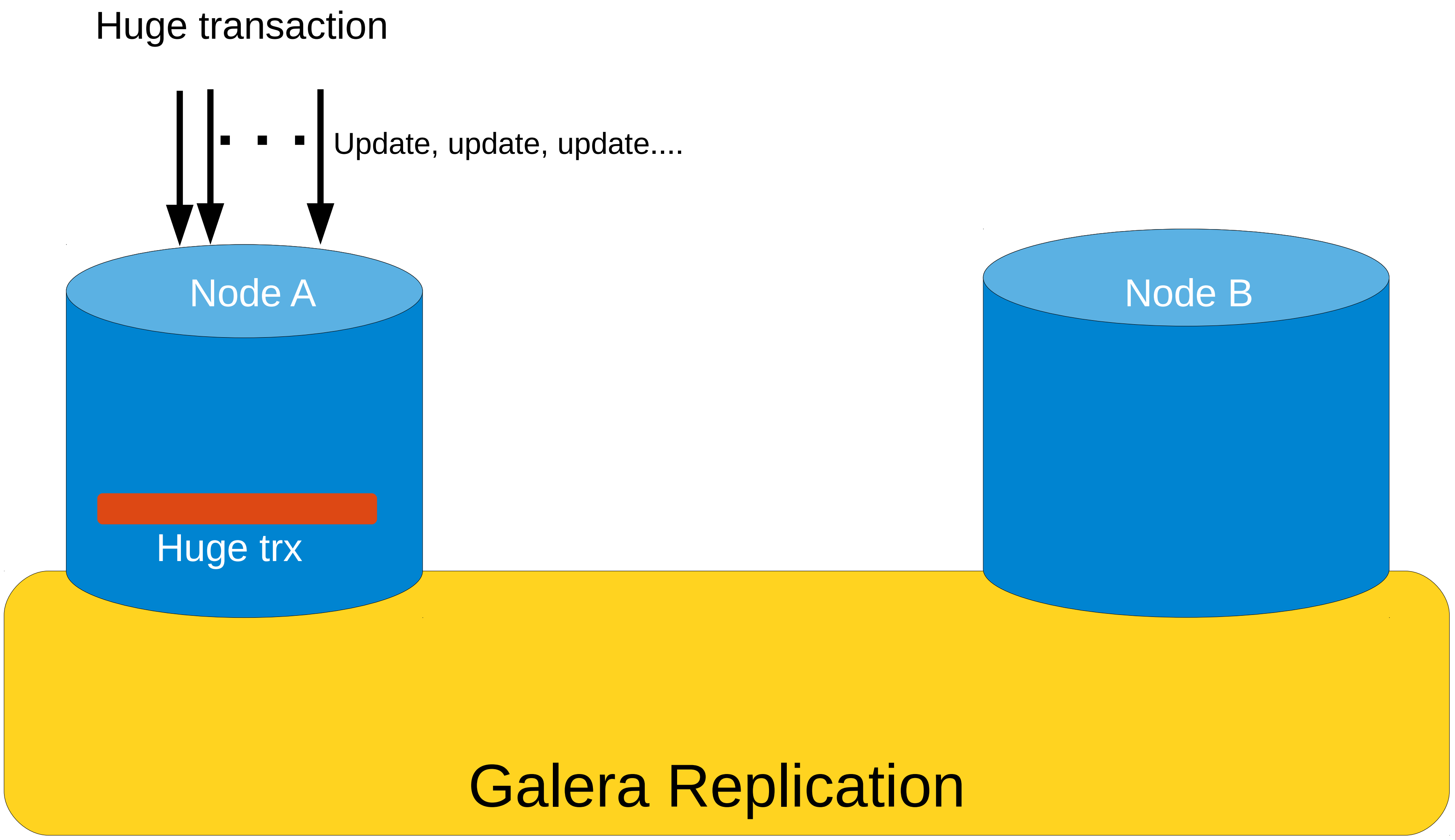
Impact of Huge Transaction



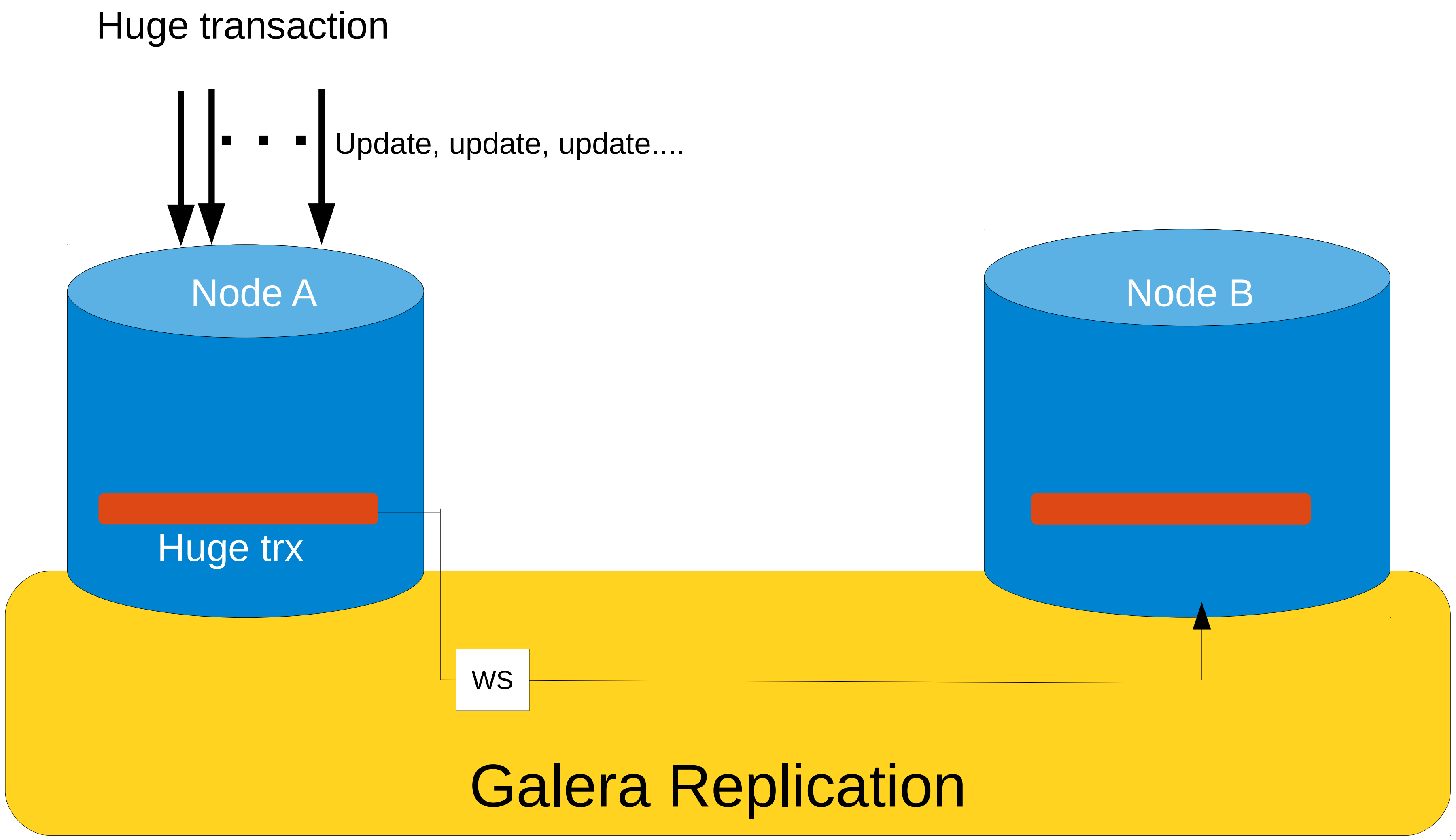
Streaming Replication

- Transaction is replicated, gradually in small fragments, during transaction processing
  - i.e. before actual commit, we replicate a number of small scale fragments
- Size threshold for fragment replication is configurable
- Replicated fragments are applied in slave transactions in all cluster nodes
  - ➔ Fragments hold locks in all nodes and cannot be conflicted later

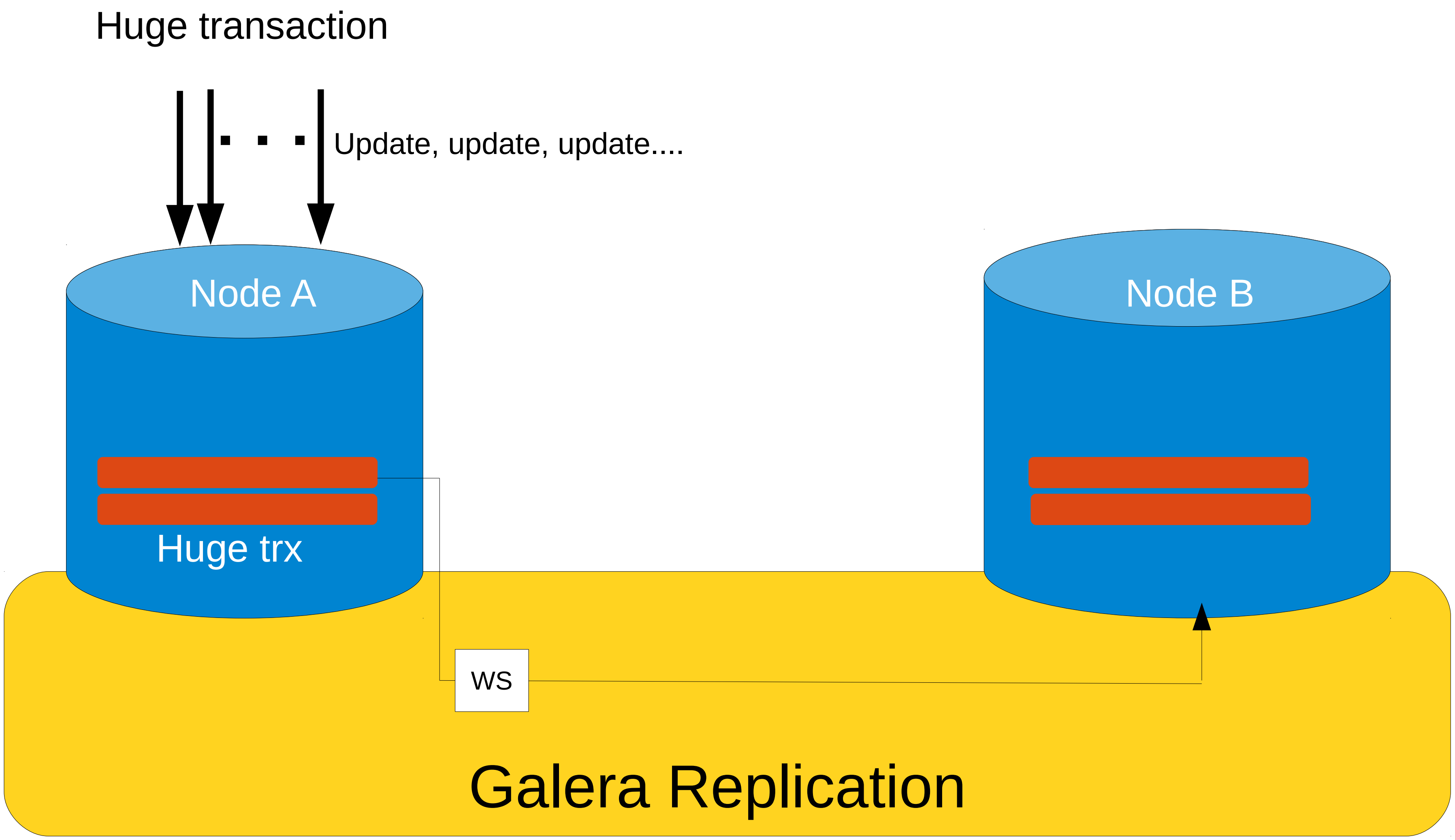
Streaming Replication



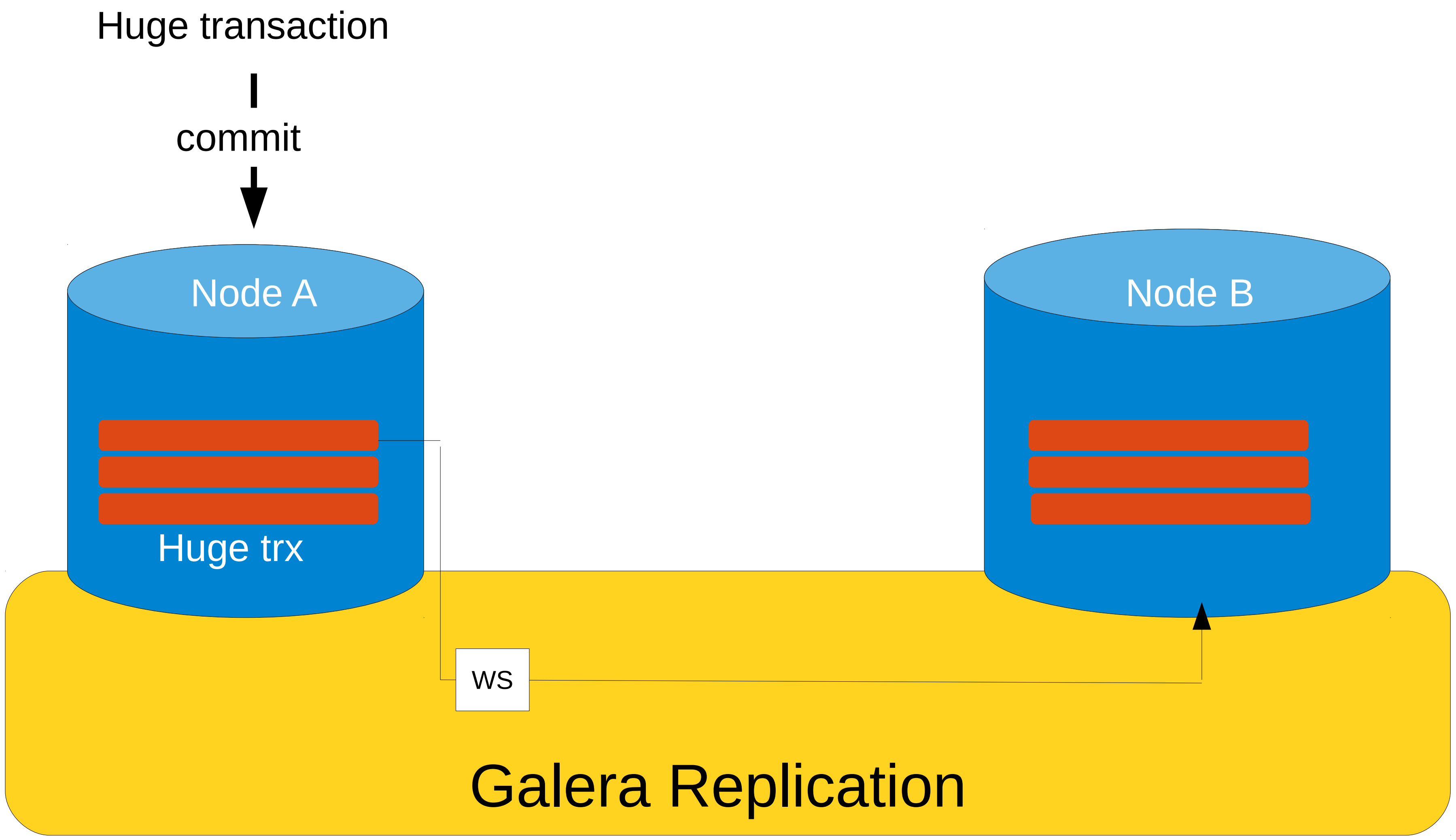
Streaming Replication



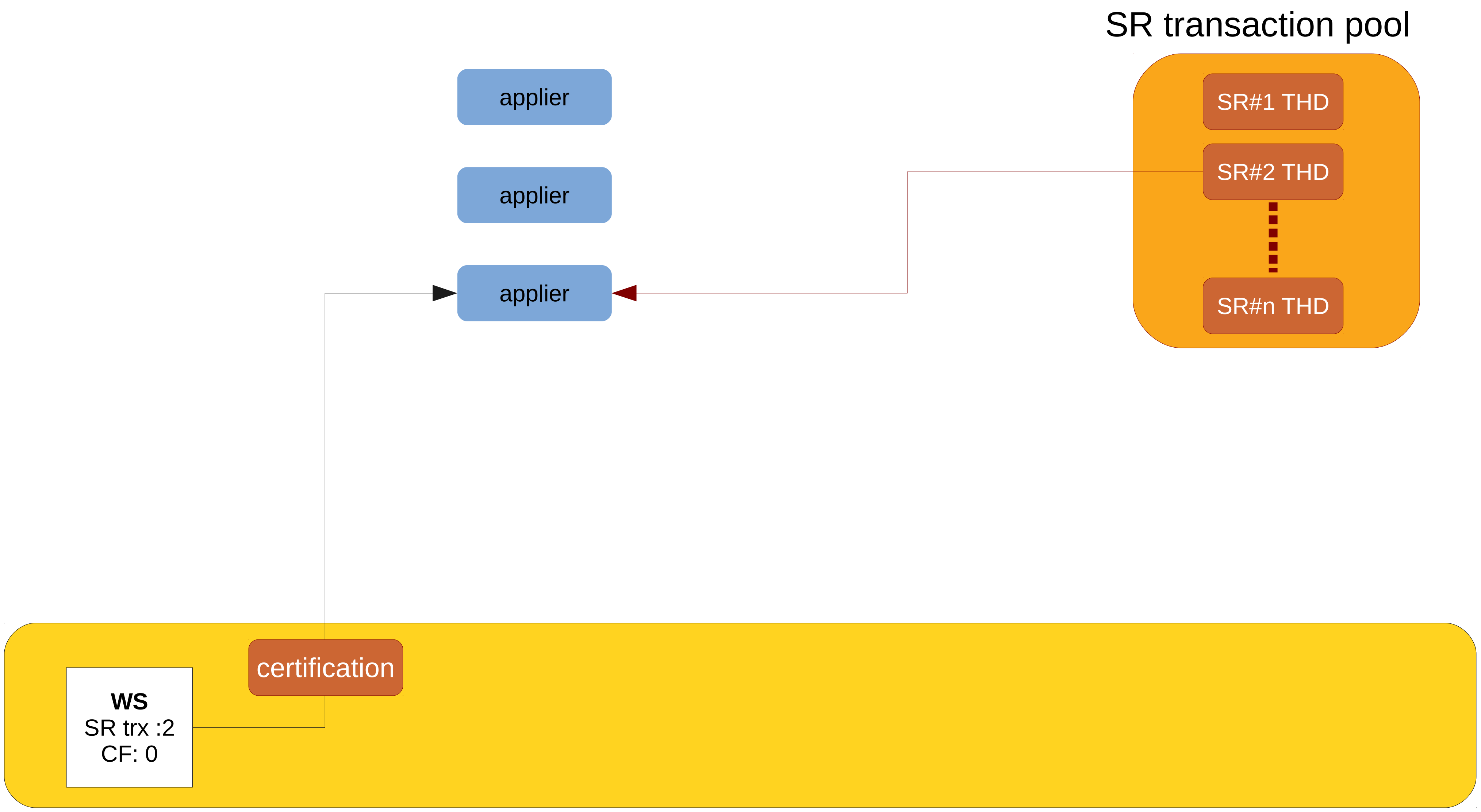
Streaming Replication



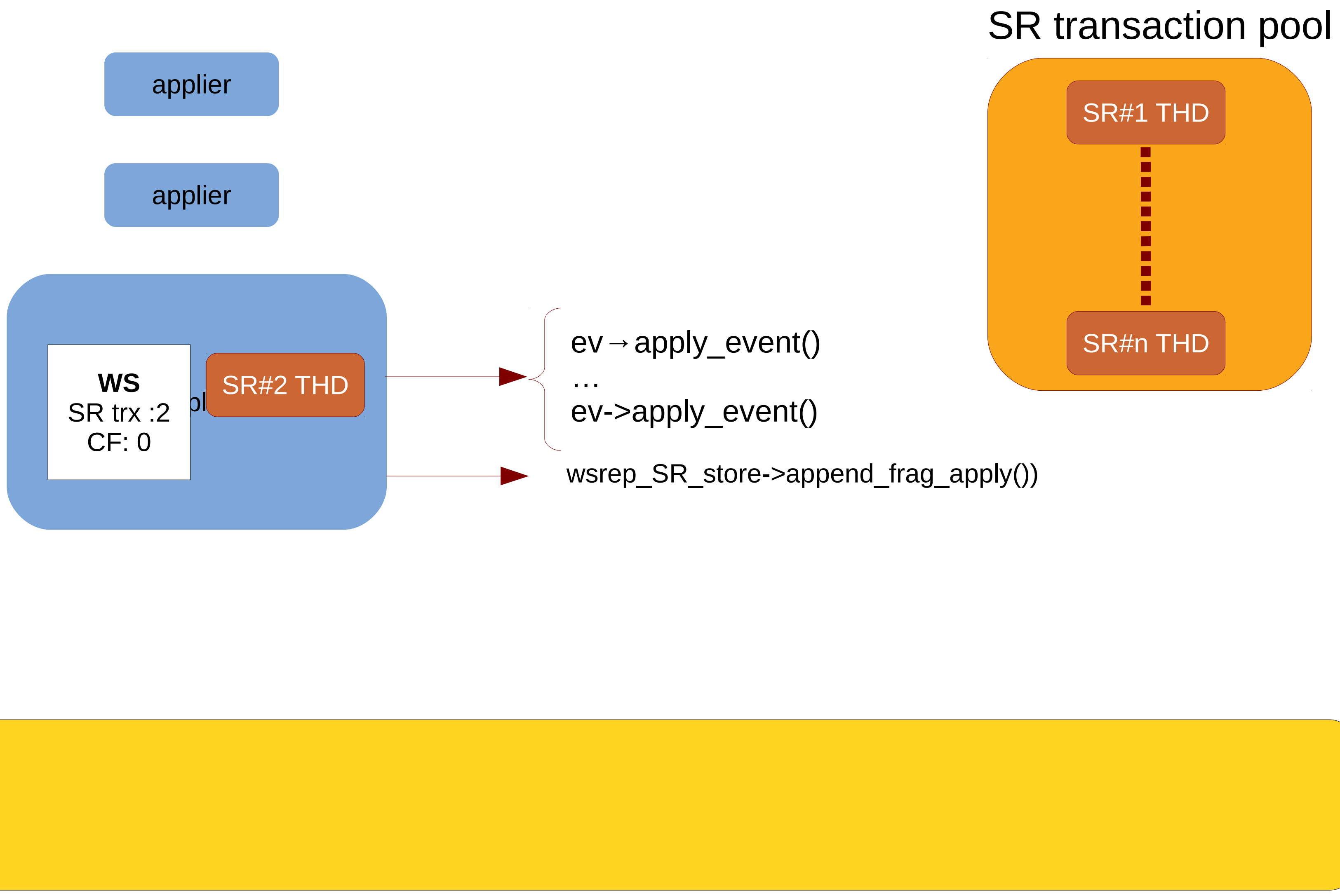
Streaming Replication



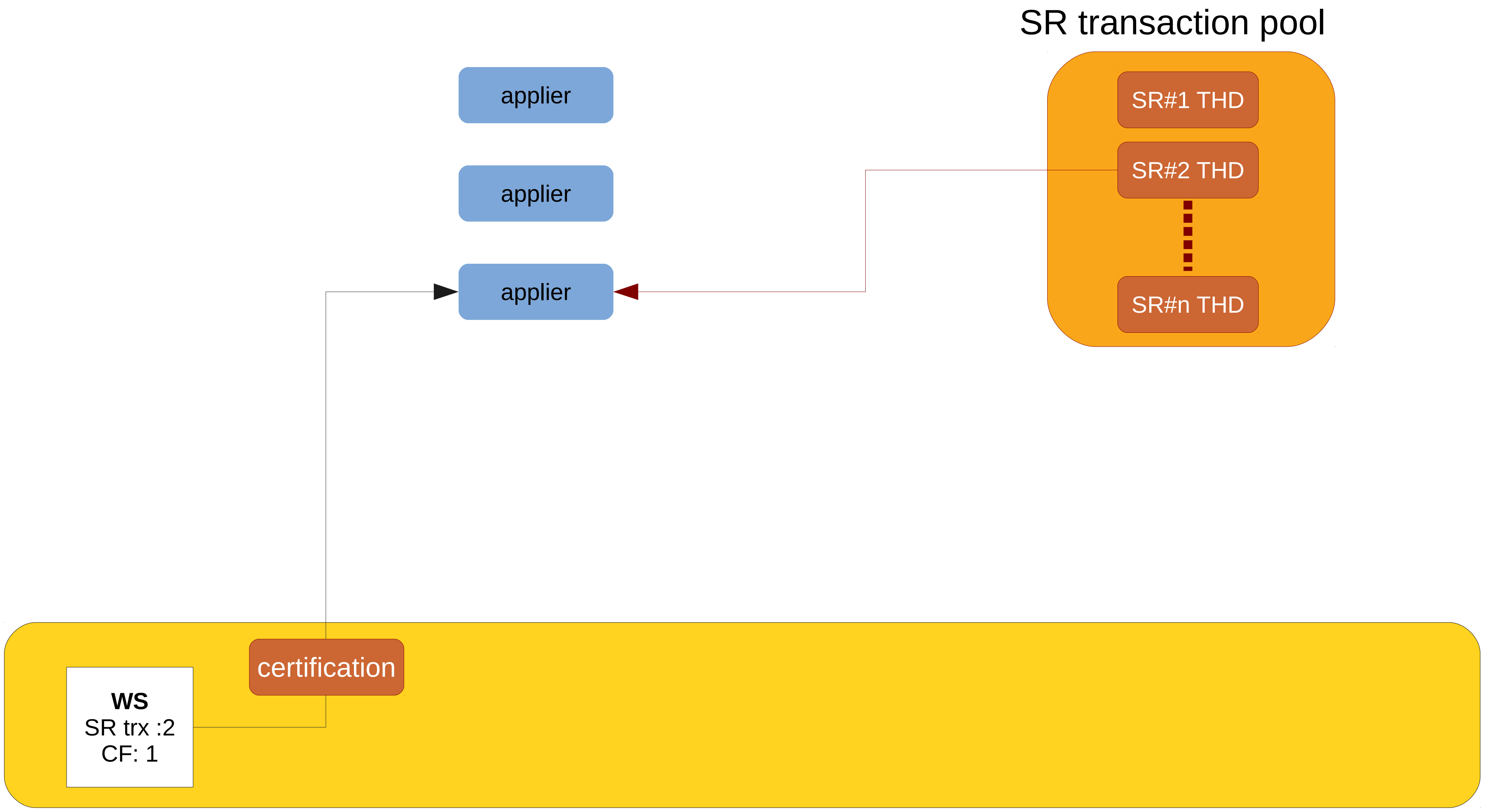
Fragment Transaction



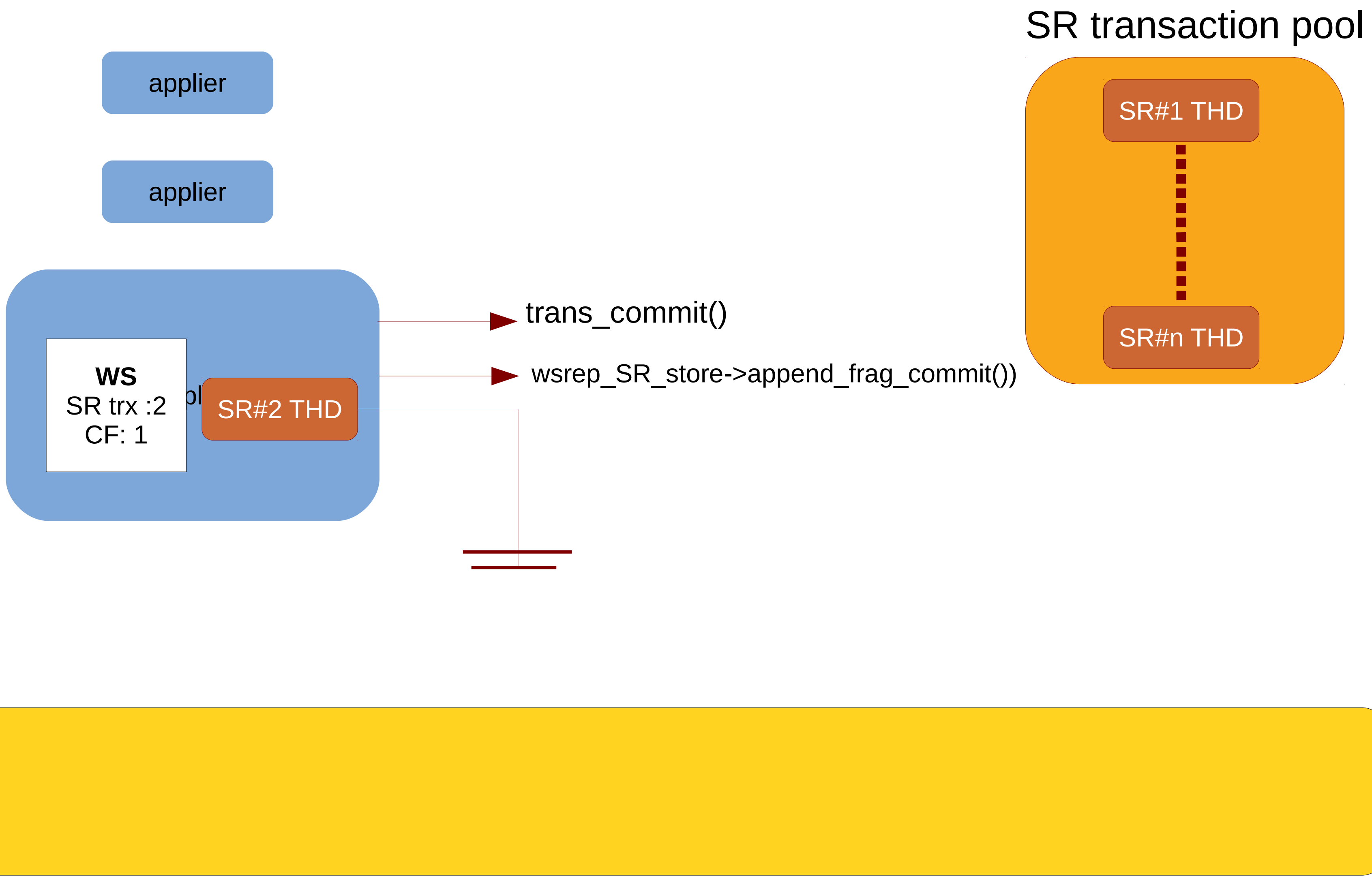
Fragment Transaction



Fragment Transaction



Fragment Transaction



Configuring Streaming Replication

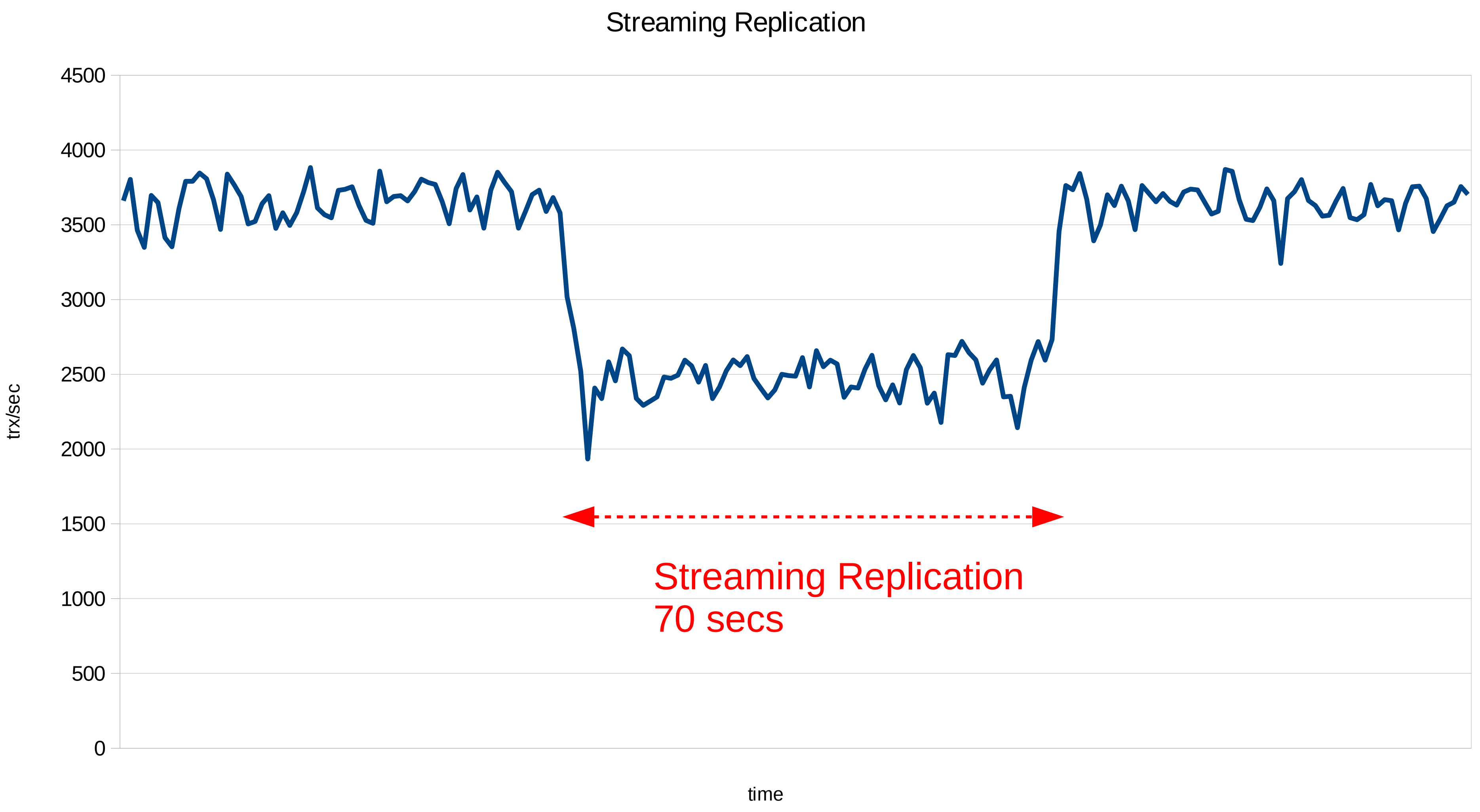
wsrep_trx_fragment_unit	Unit metrics for fragmenting, options are: • bytes WS size in bytes • events # of binlog events • rows # of rows modified • statements # of SQL statements issued
wsrep_trx_fragment_size	• Threshold size (in units), when fragment will be replicated • 0 = no streaming

Streaming Replication Demo Setup

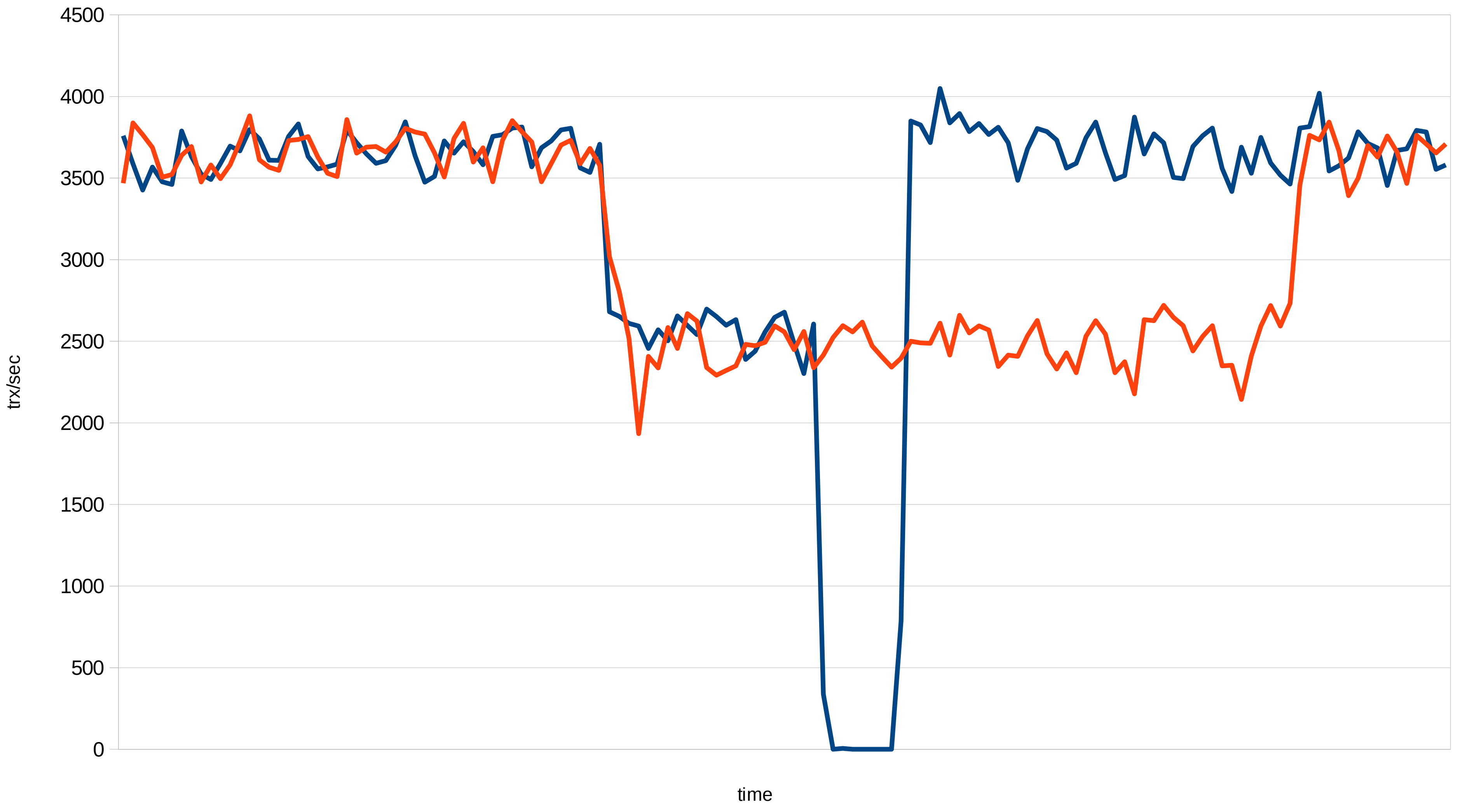
1. Same scenario as before
2. Configure node1 to fragment huge transaction in 10K batches
   - wsrep_trx_fragment_unit = bytes
   - wsrep_trx_fragment_size = 10000

➡ monitor trx/sec rate in the cluster when streaming replication progresses

Streaming Replication

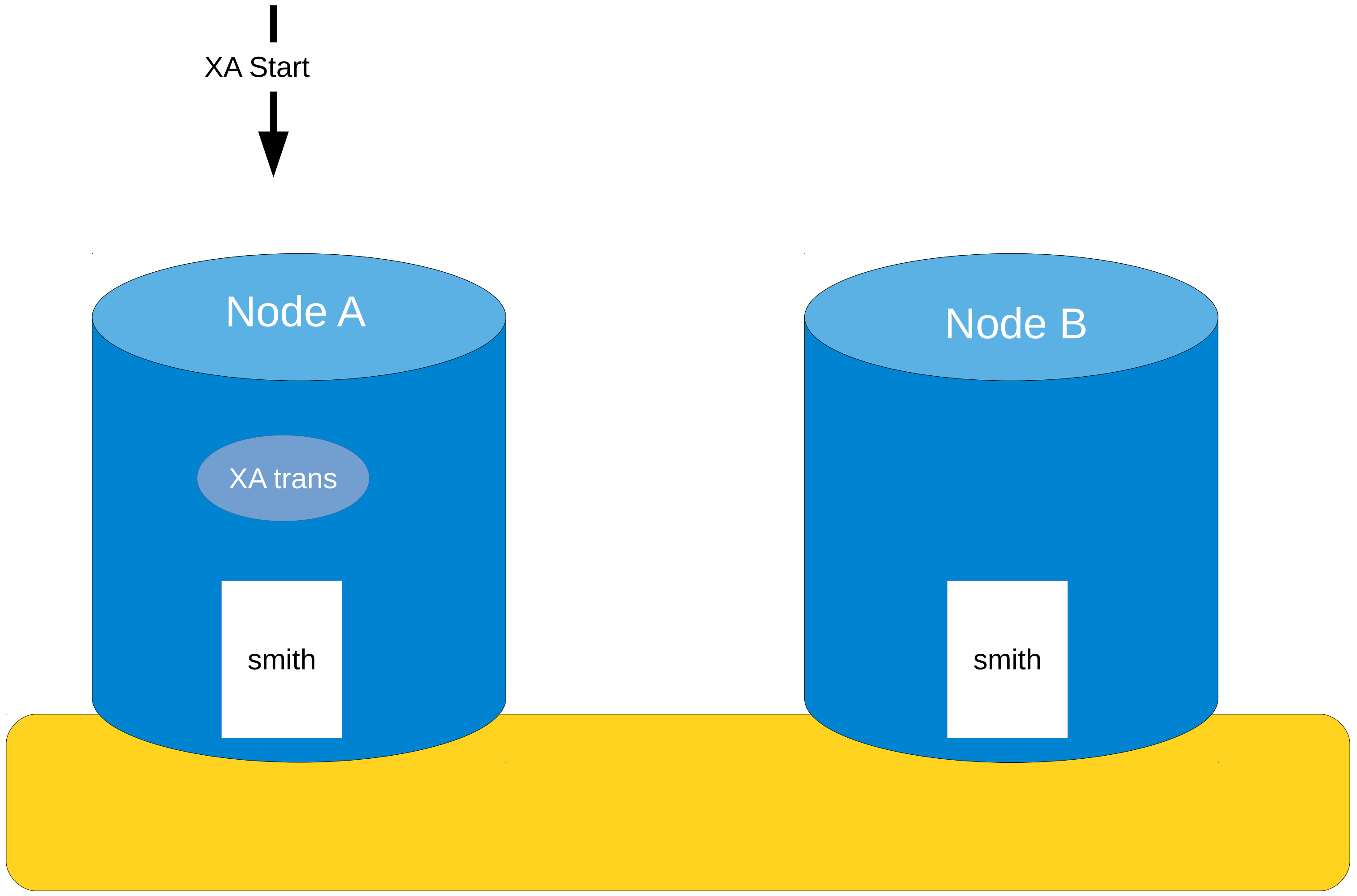


Streaming Replication

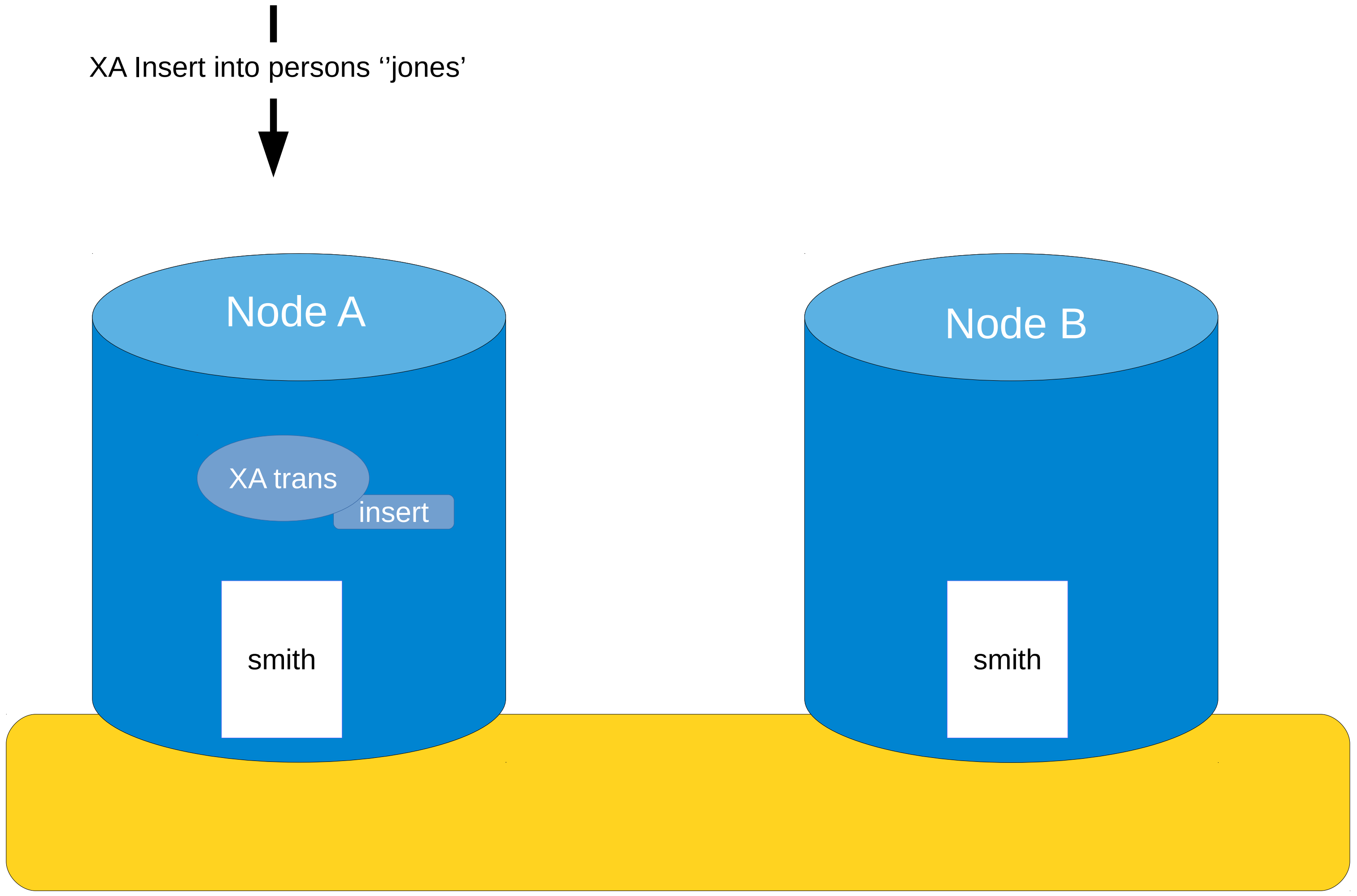


XA Transactions with Galera 3

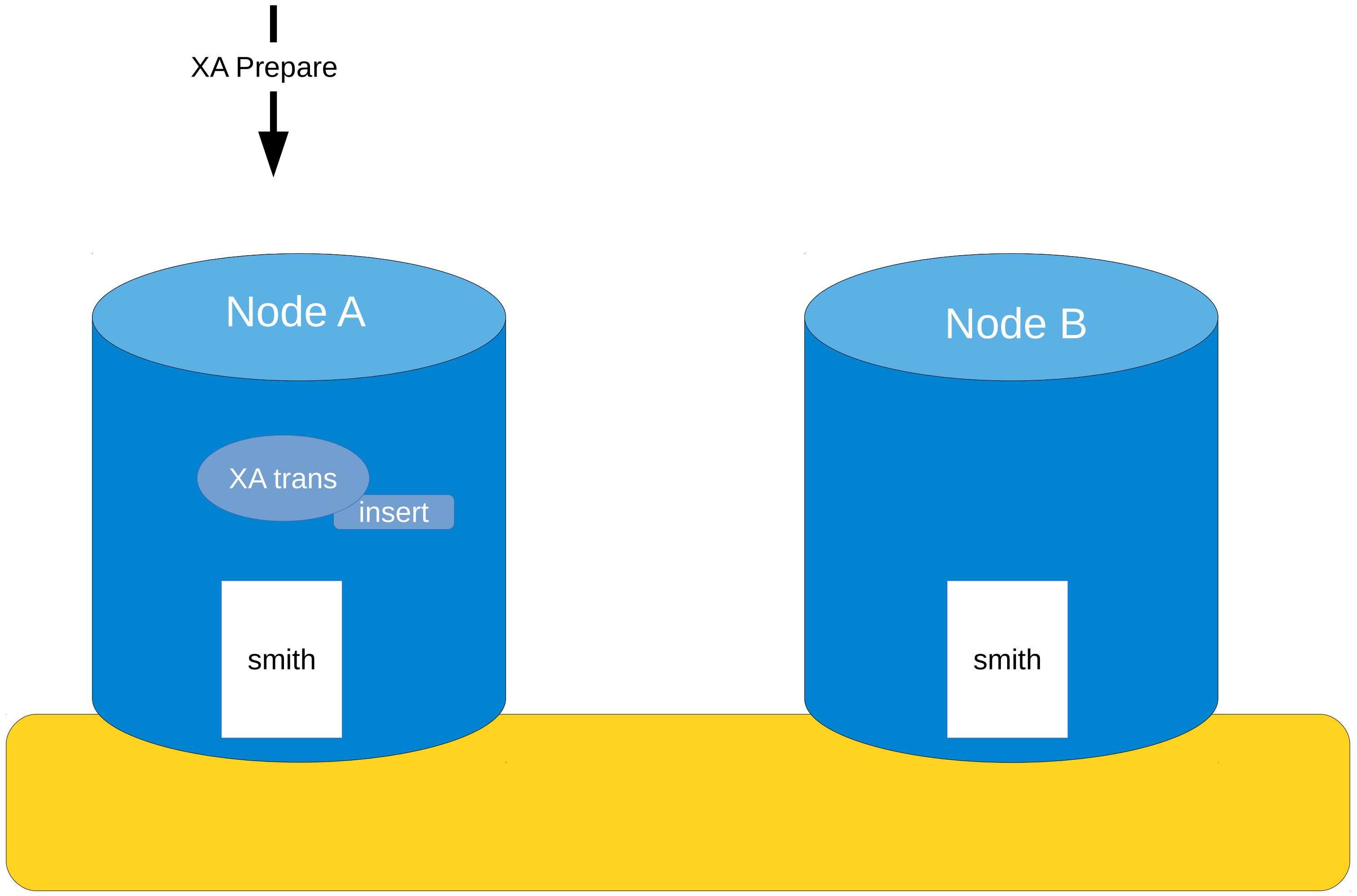
XA Transaction Support



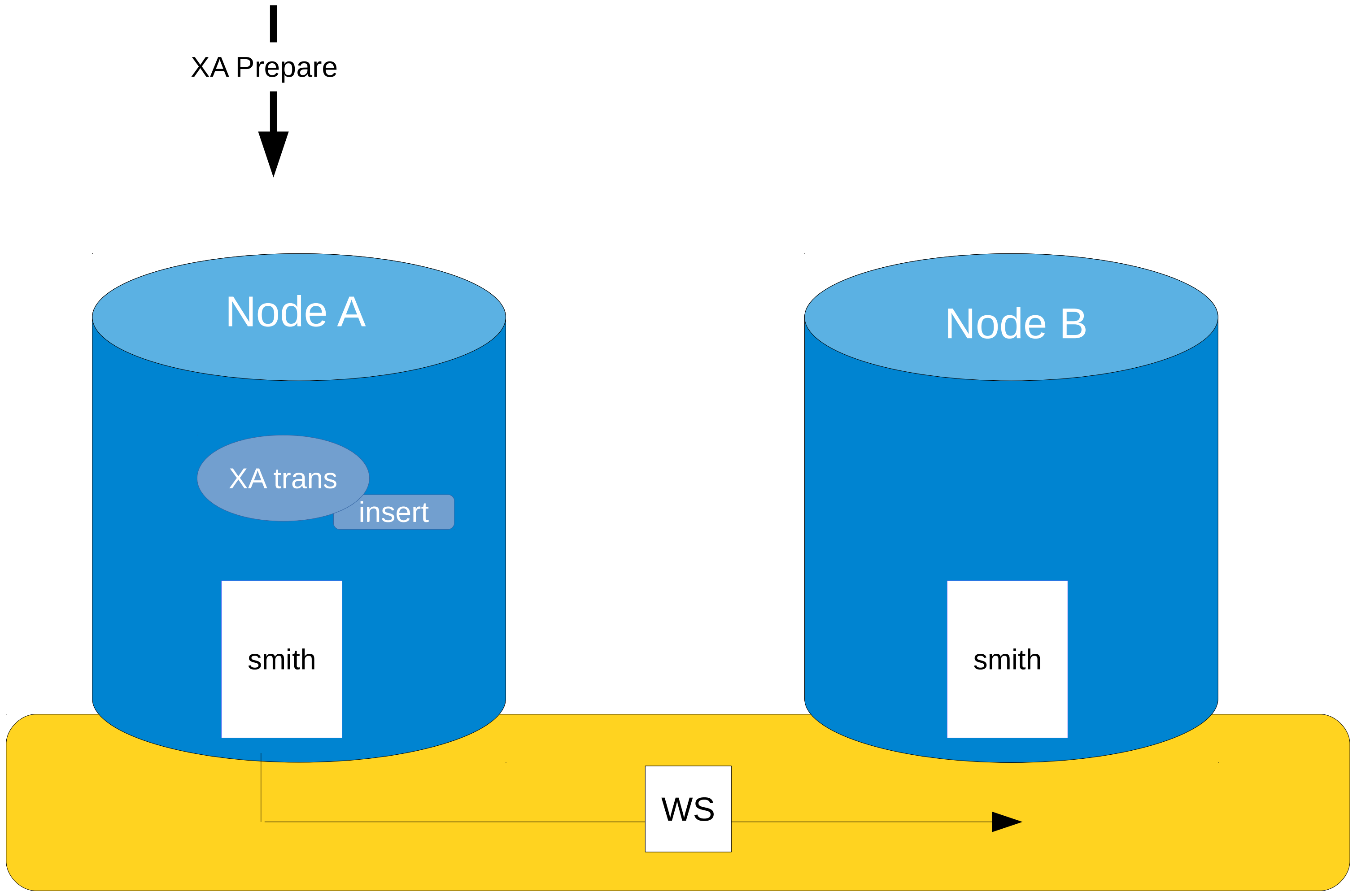
XA Transaction Support



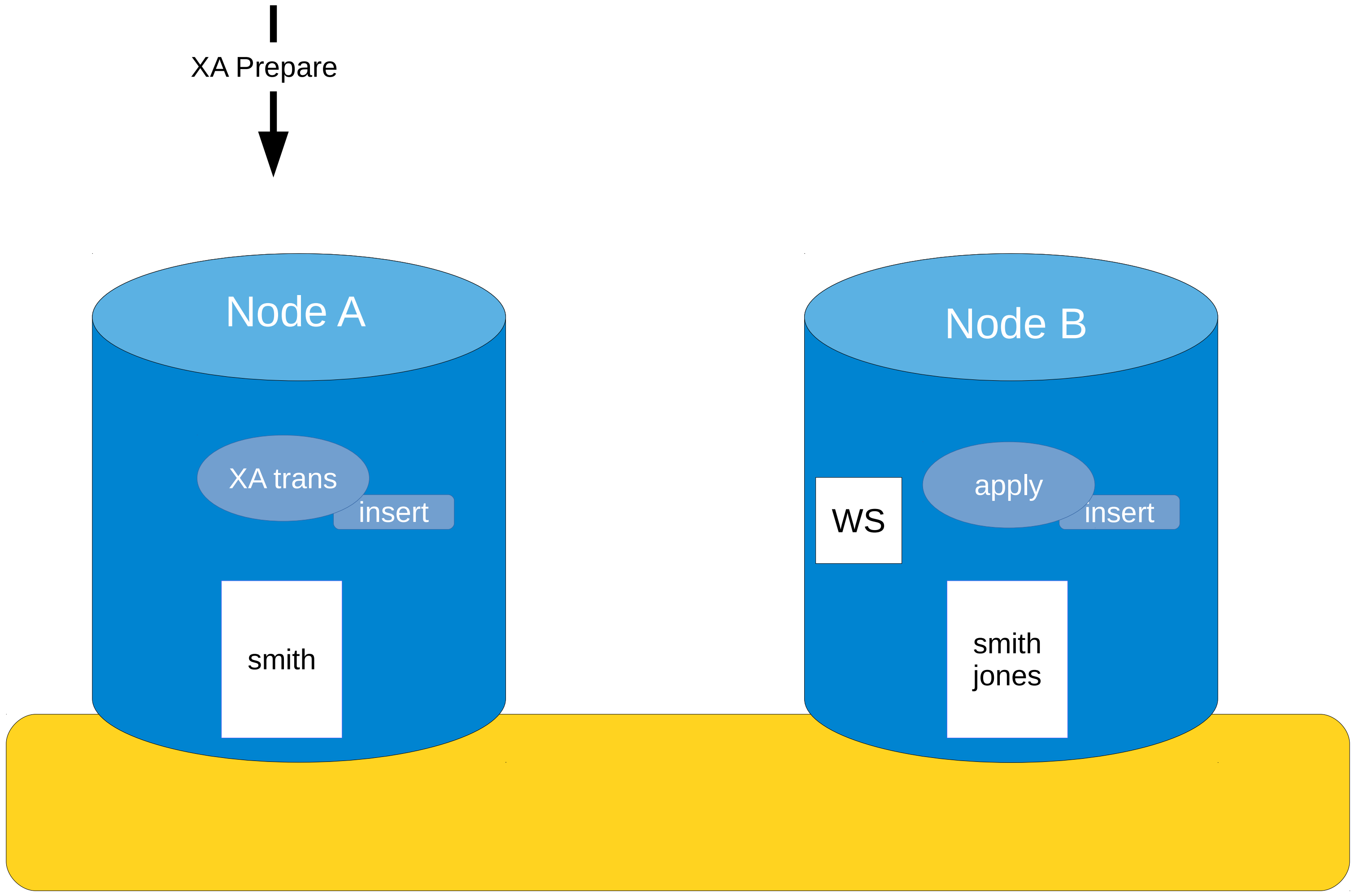
XA Transaction Support



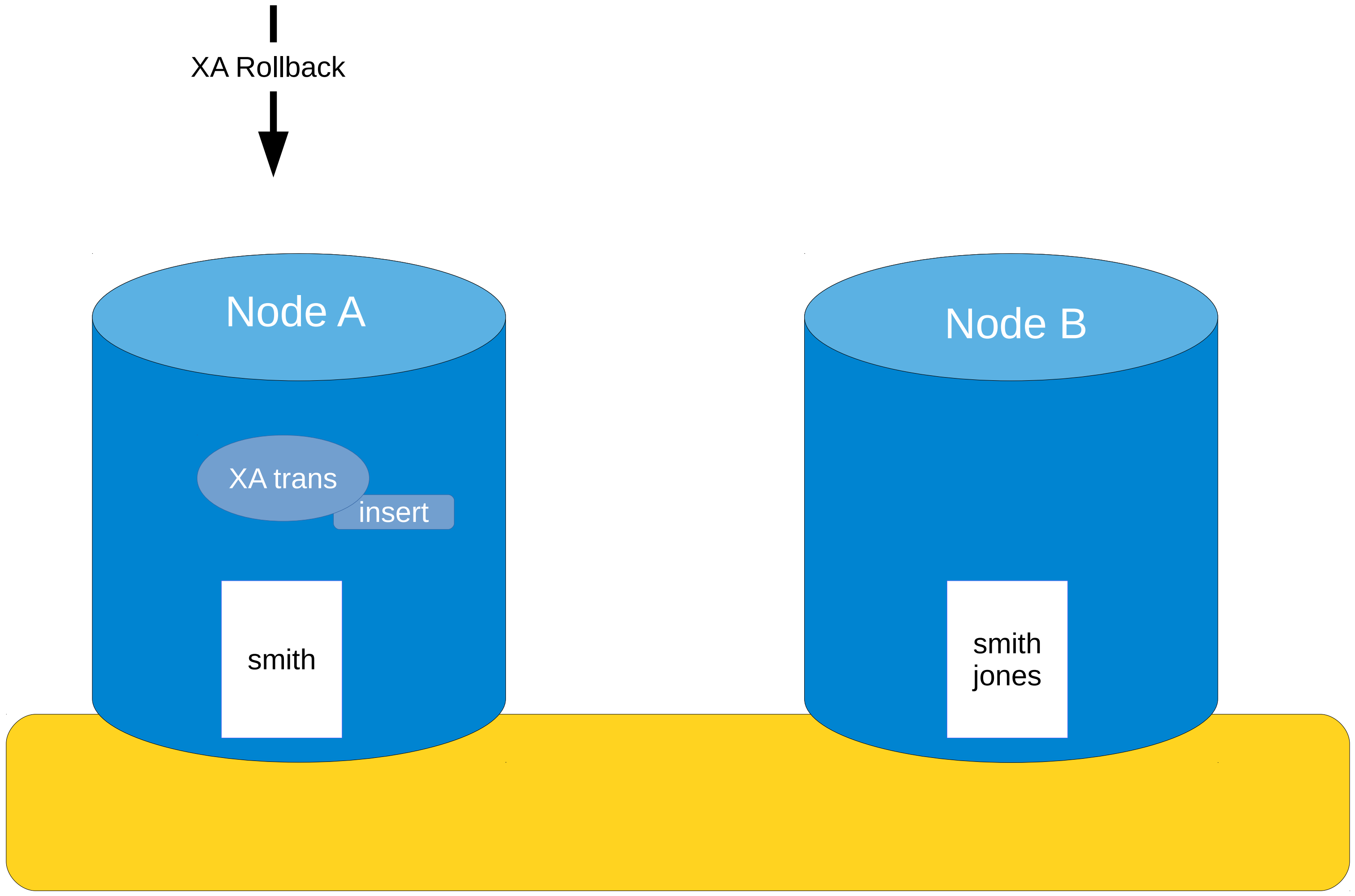
XA Transaction Support



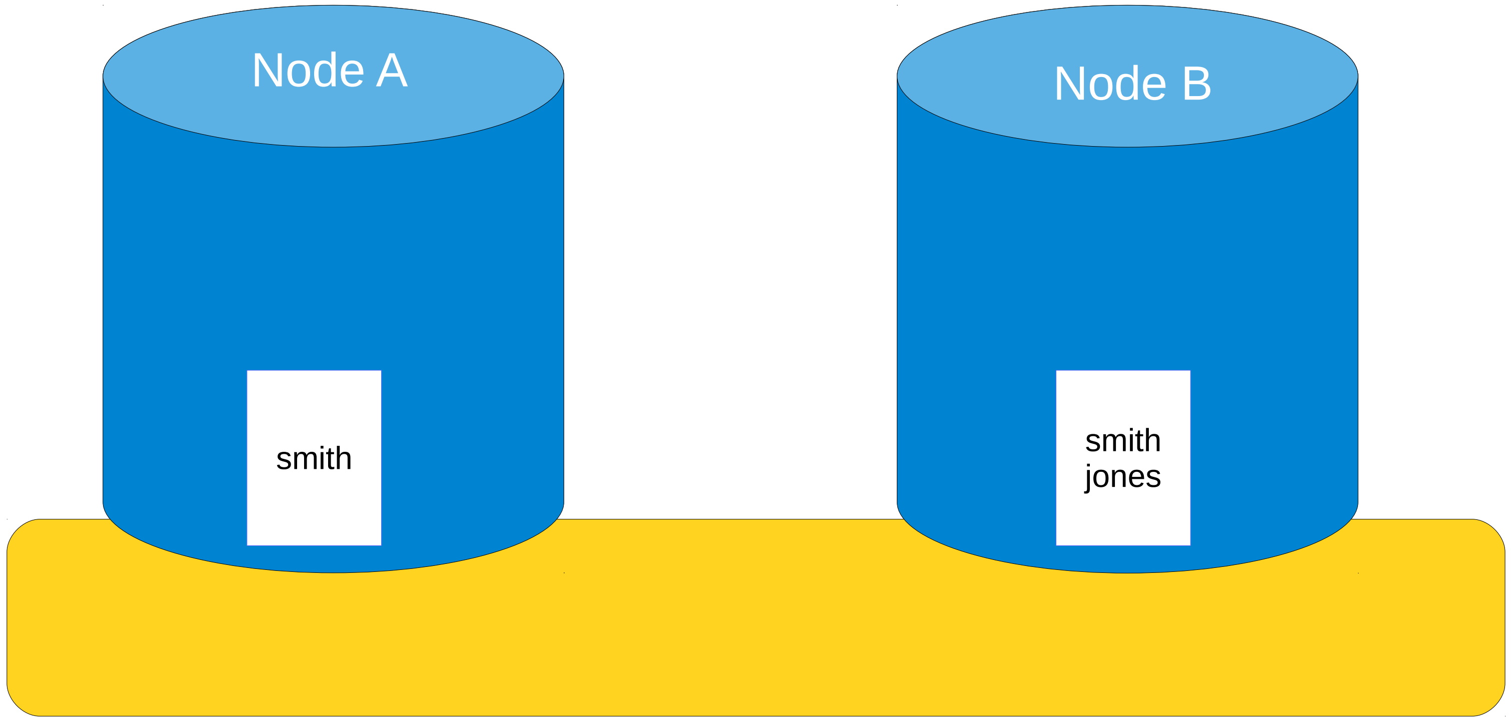
XA Transaction Support



XA Transaction Support

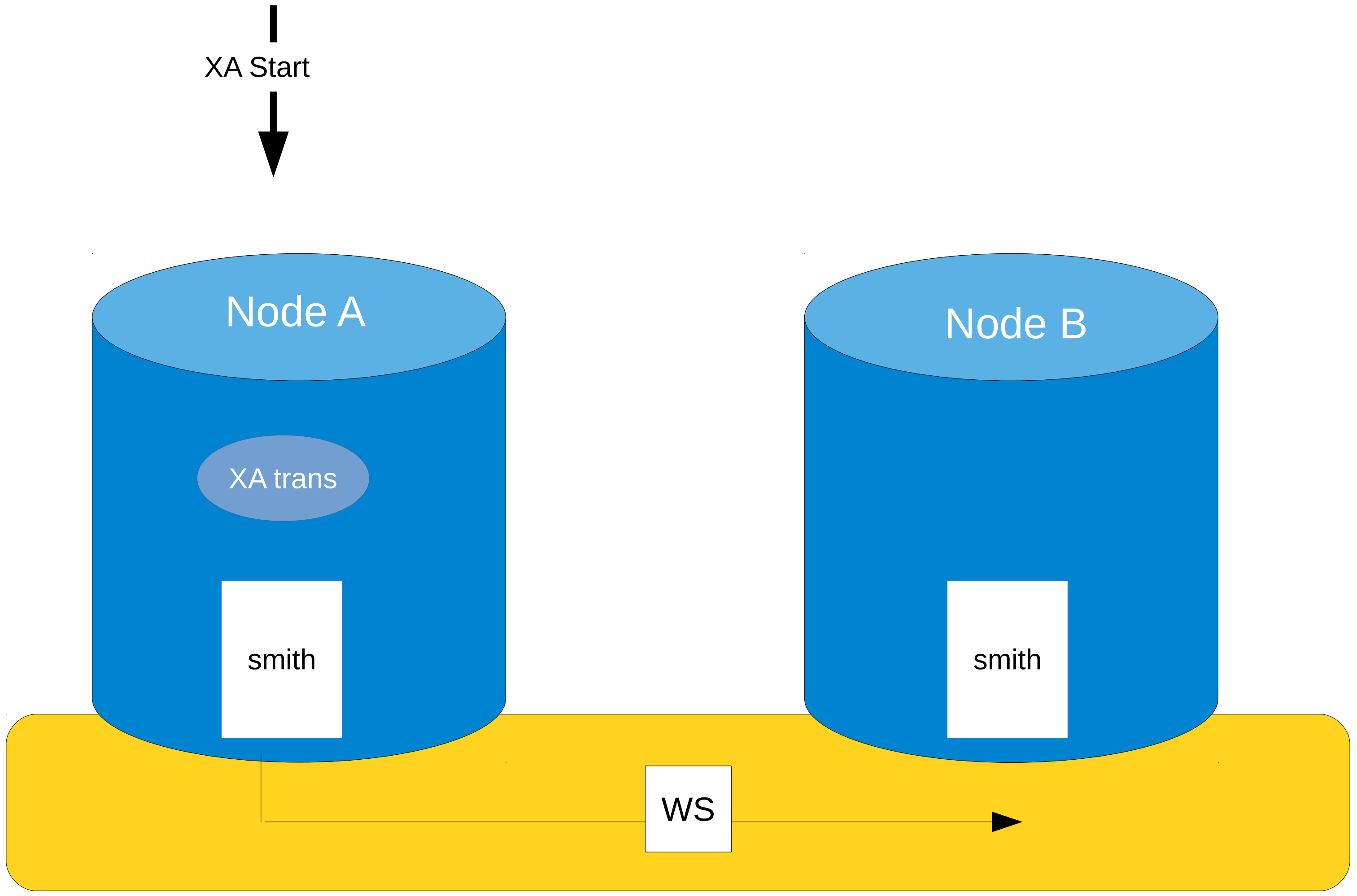


XA Transaction Support

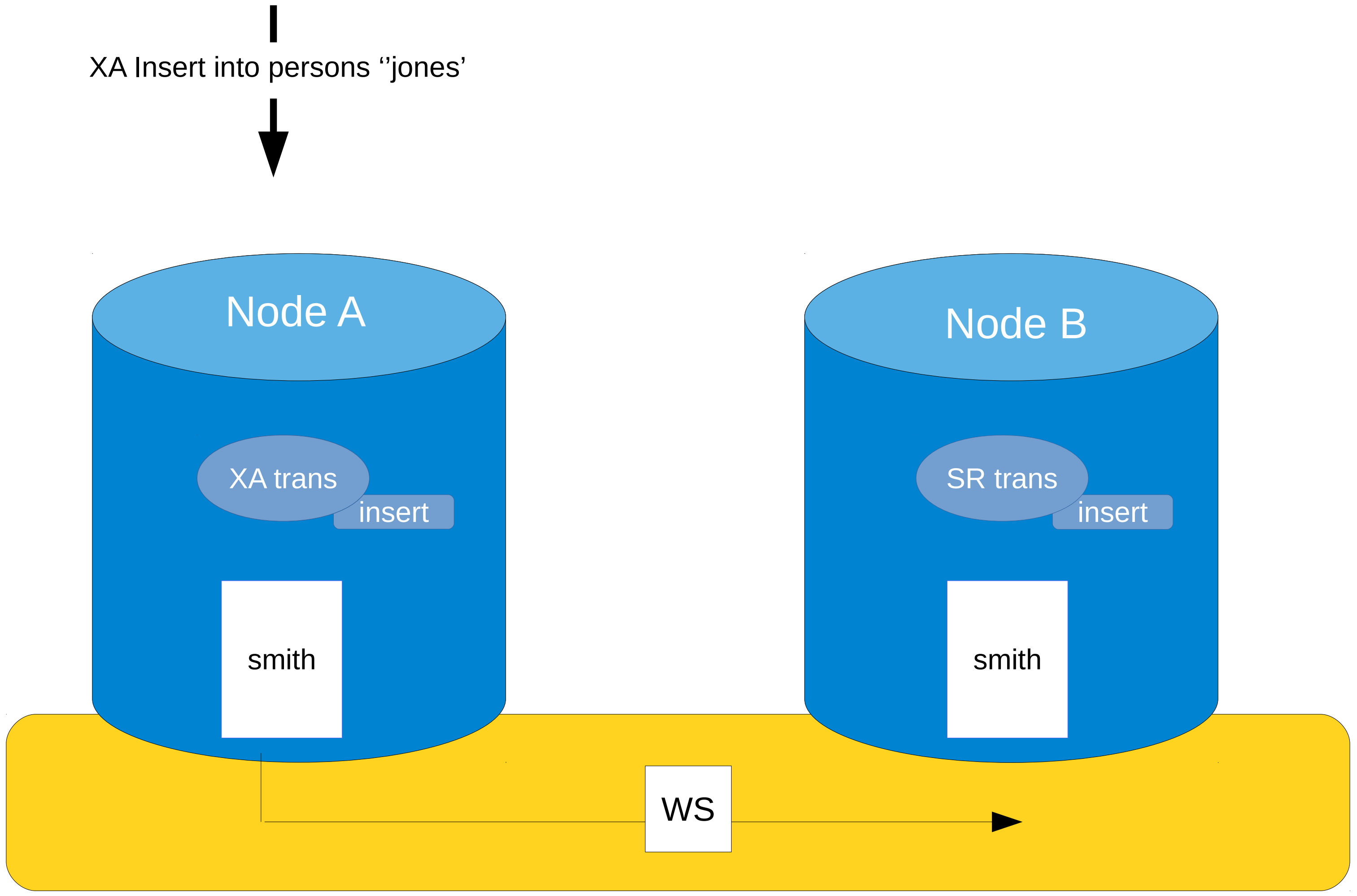


XA by Streaming Replication

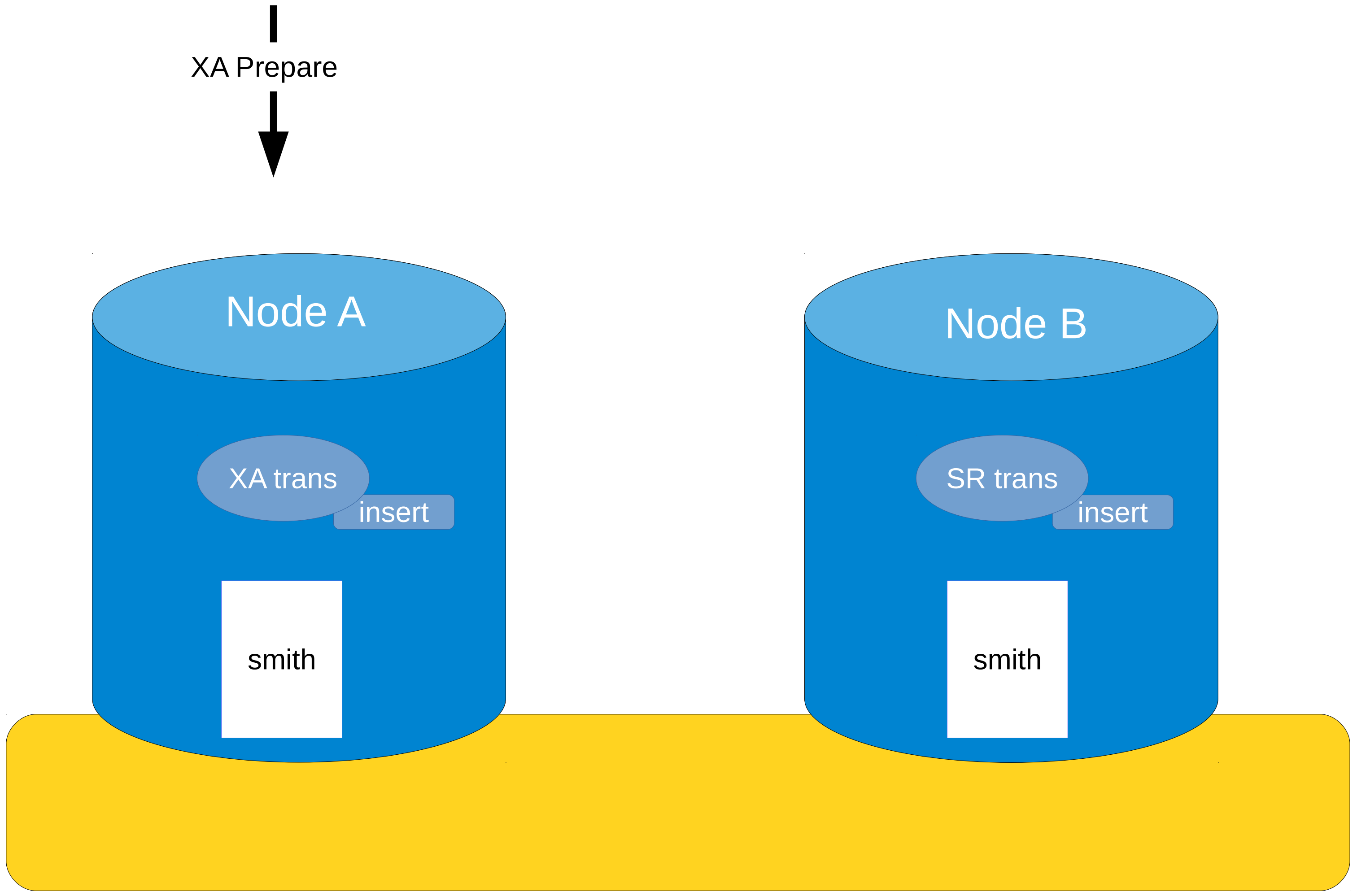
XA Transaction Support



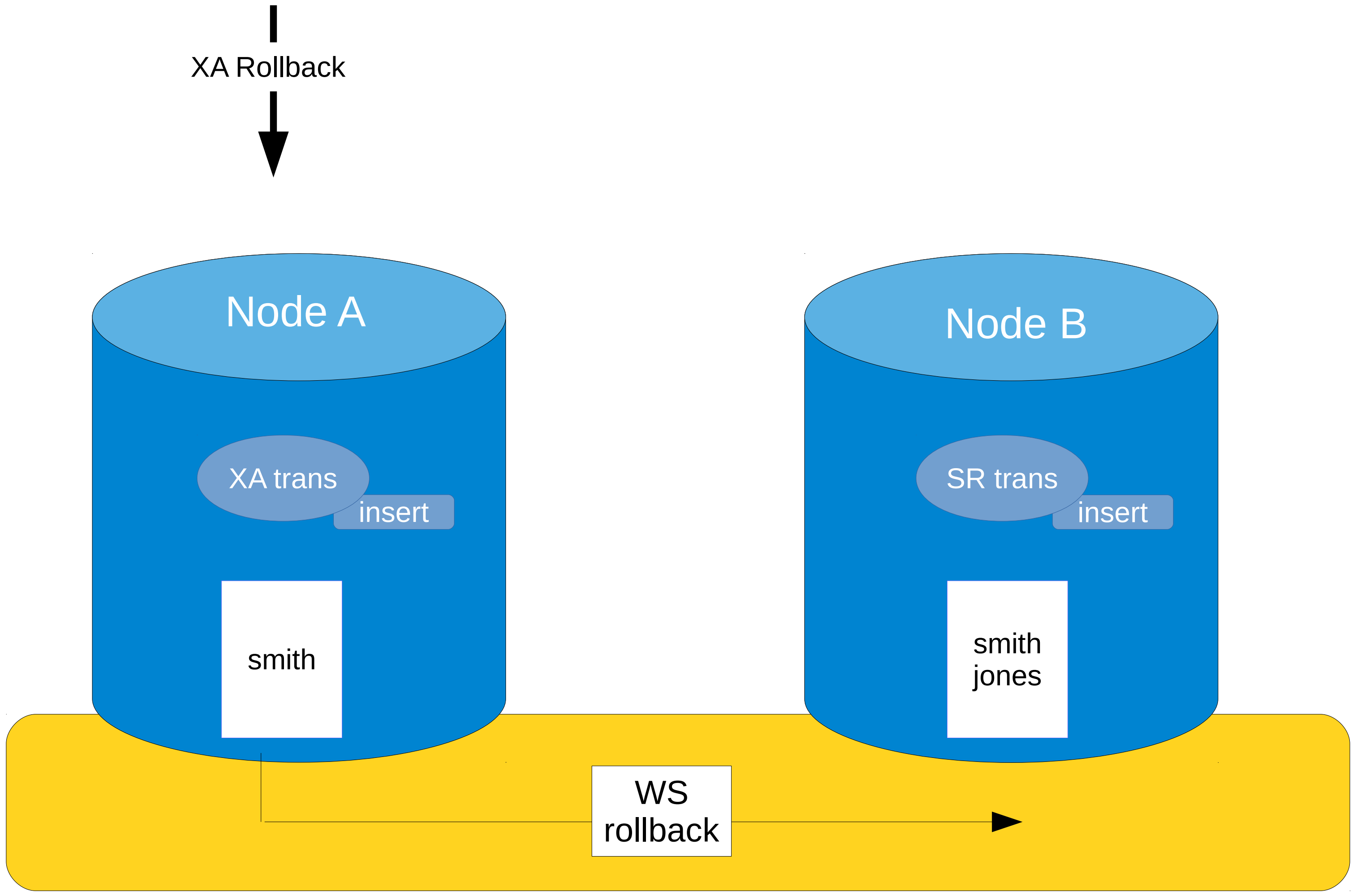
XA Transaction Support



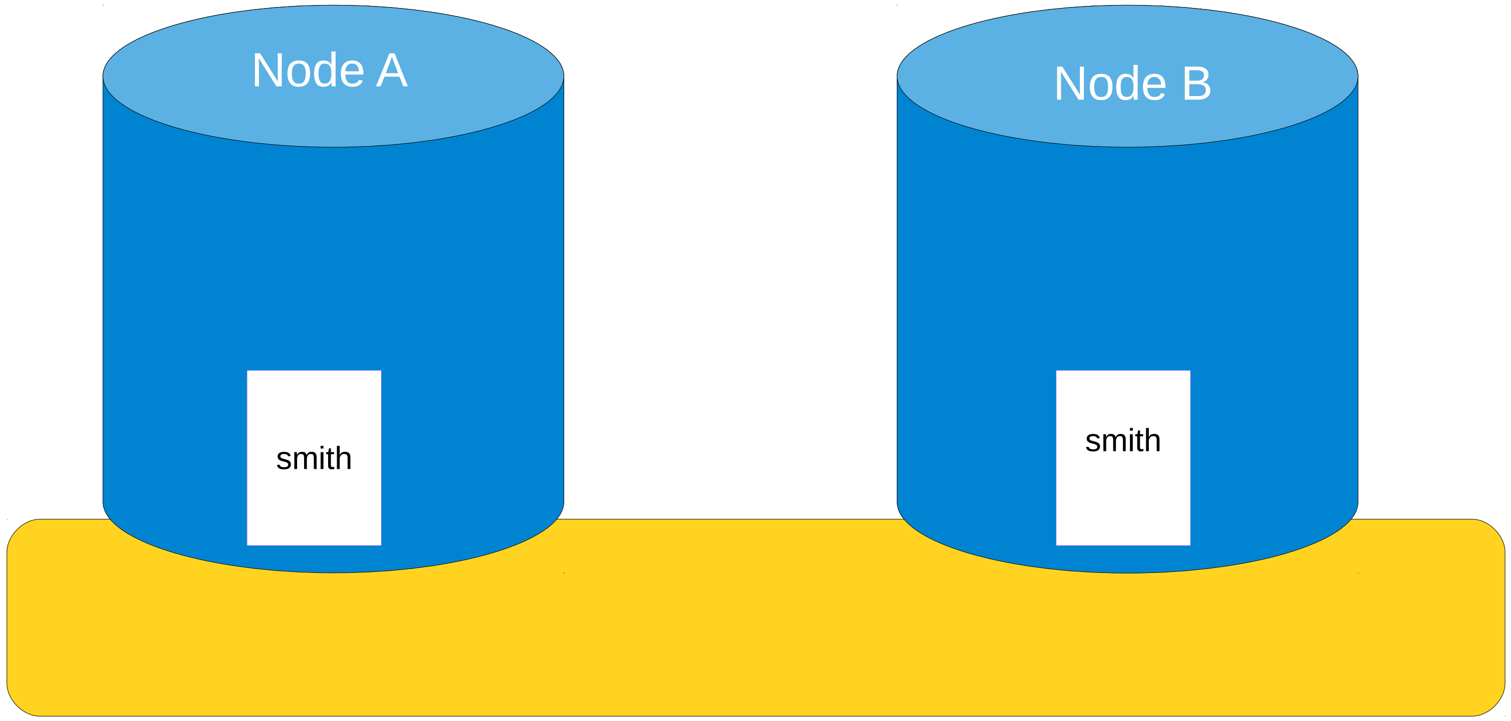
XA Transaction Support



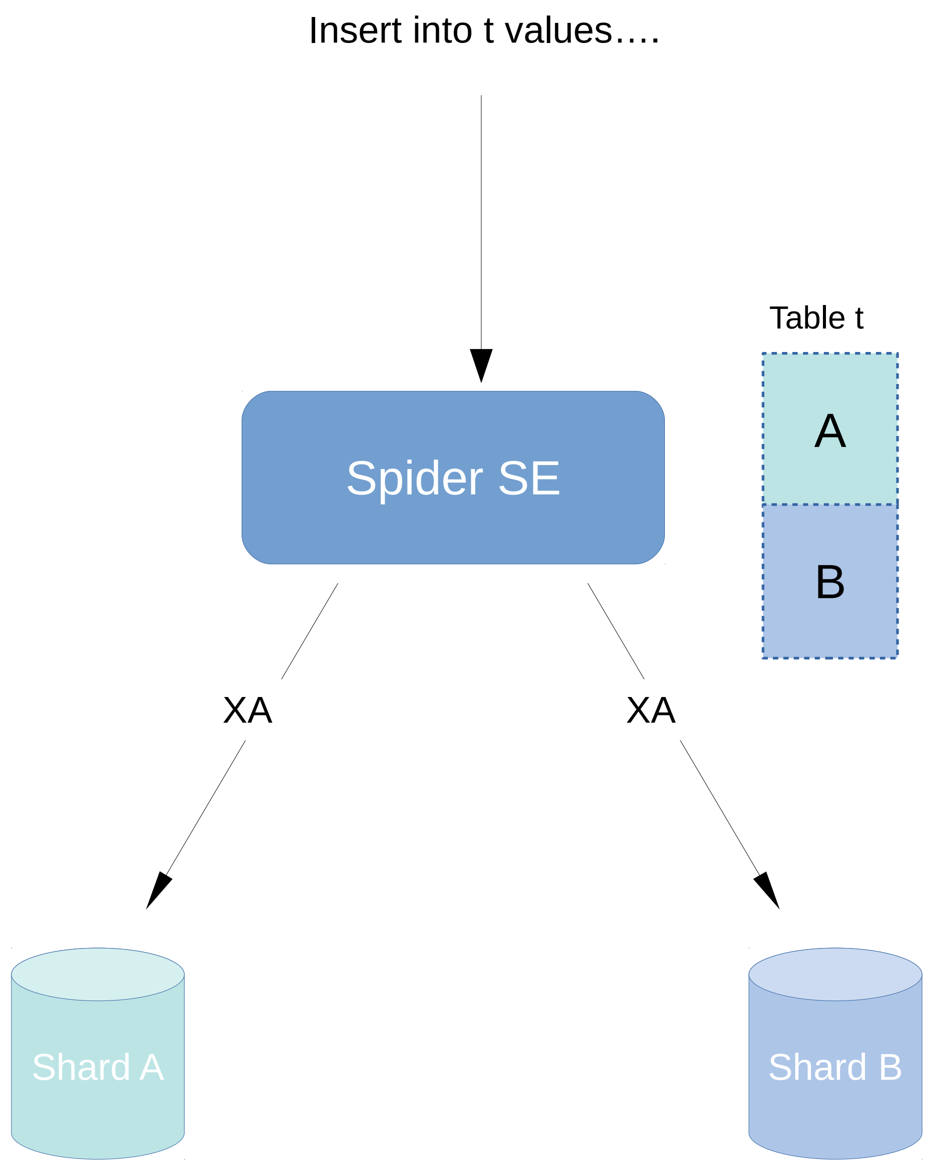
XA Transaction Support

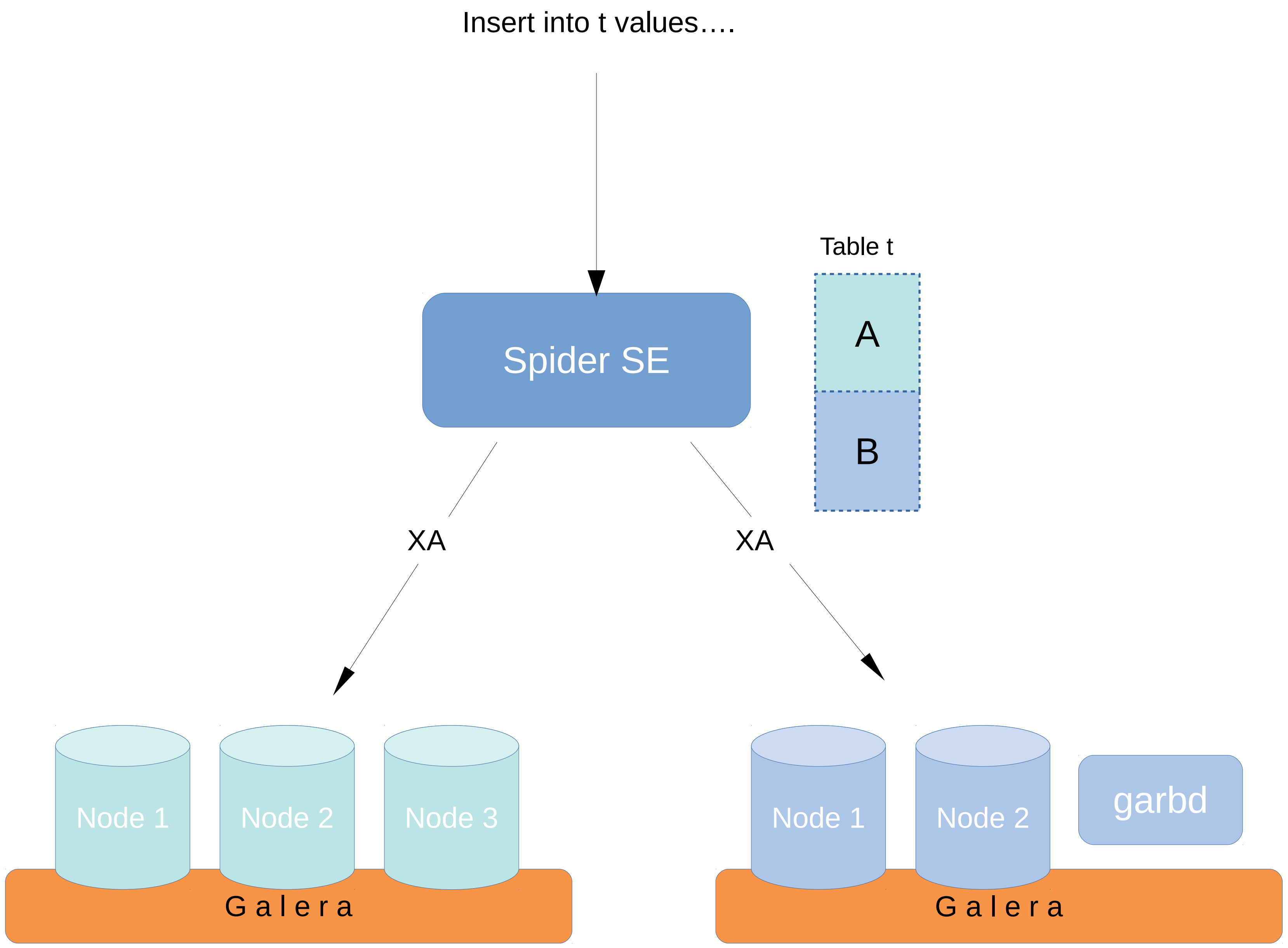


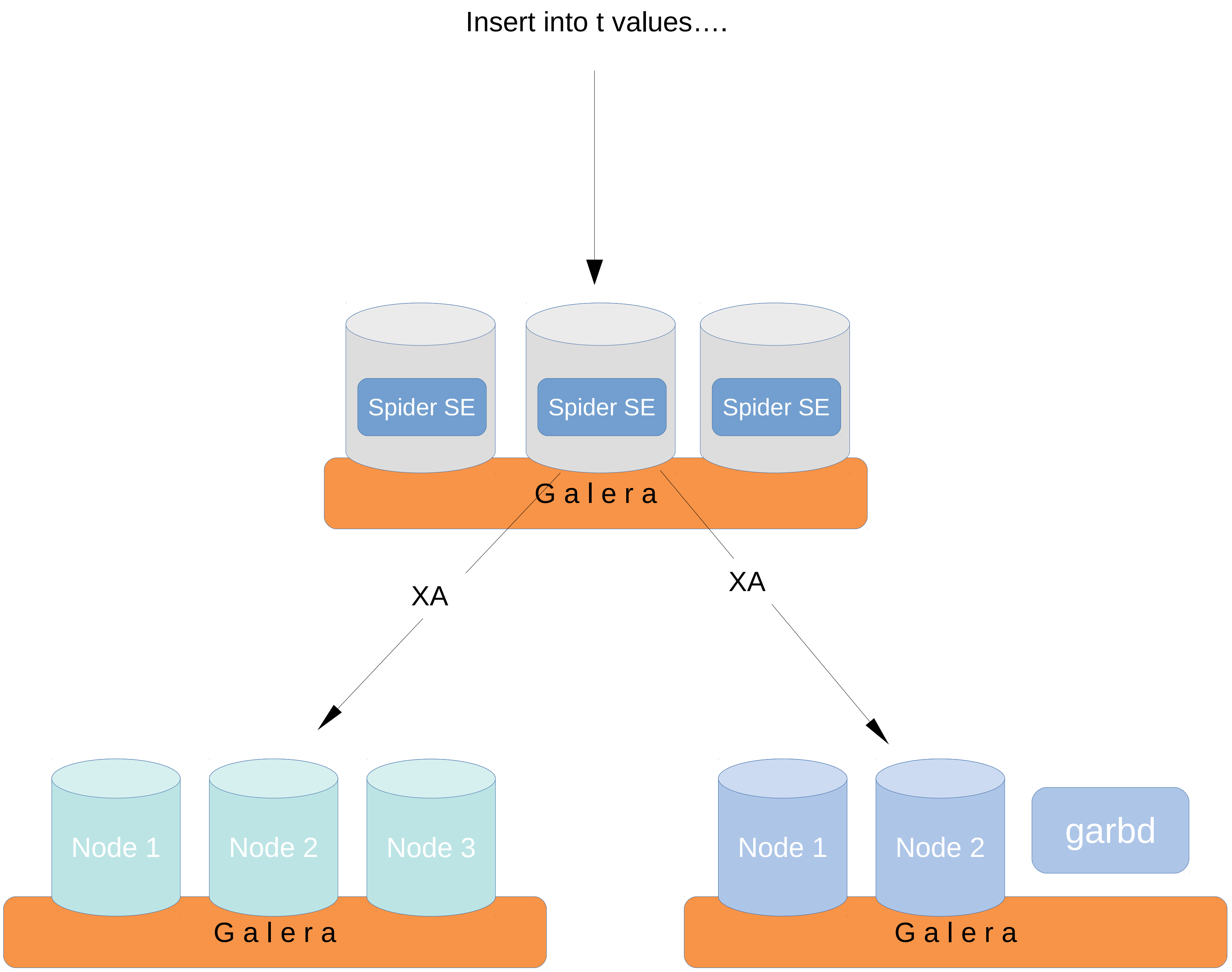
XA Transaction Support



Spider Cluster

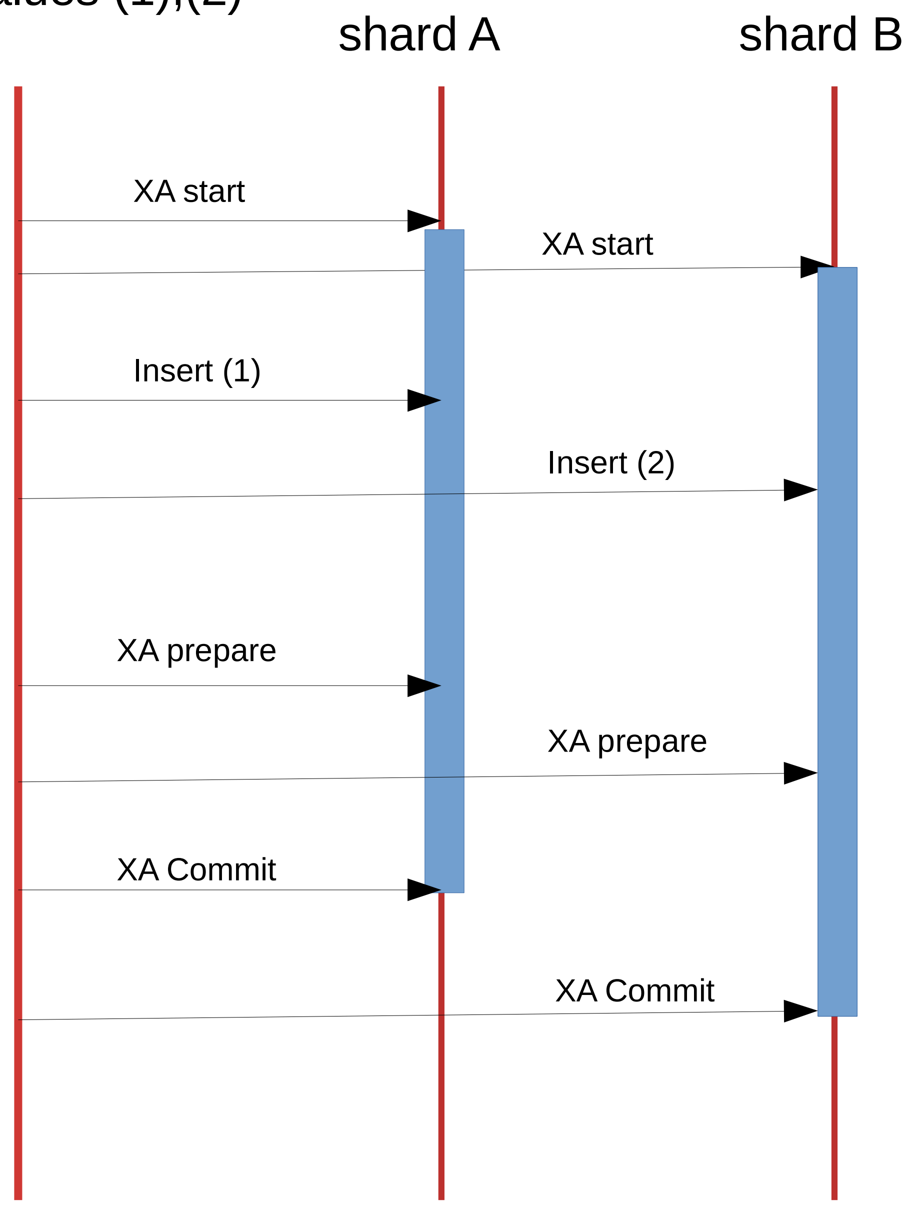

Insert into t values….
Spider SE
Table t
A
B
XA
XA
Shard A
Shard B


Insert into t values….
Table t
A
B
Spider SE
XA
XA
Node 1
Node 2
Node 3
Galera
Node 1
Node 2
garbd
Galera

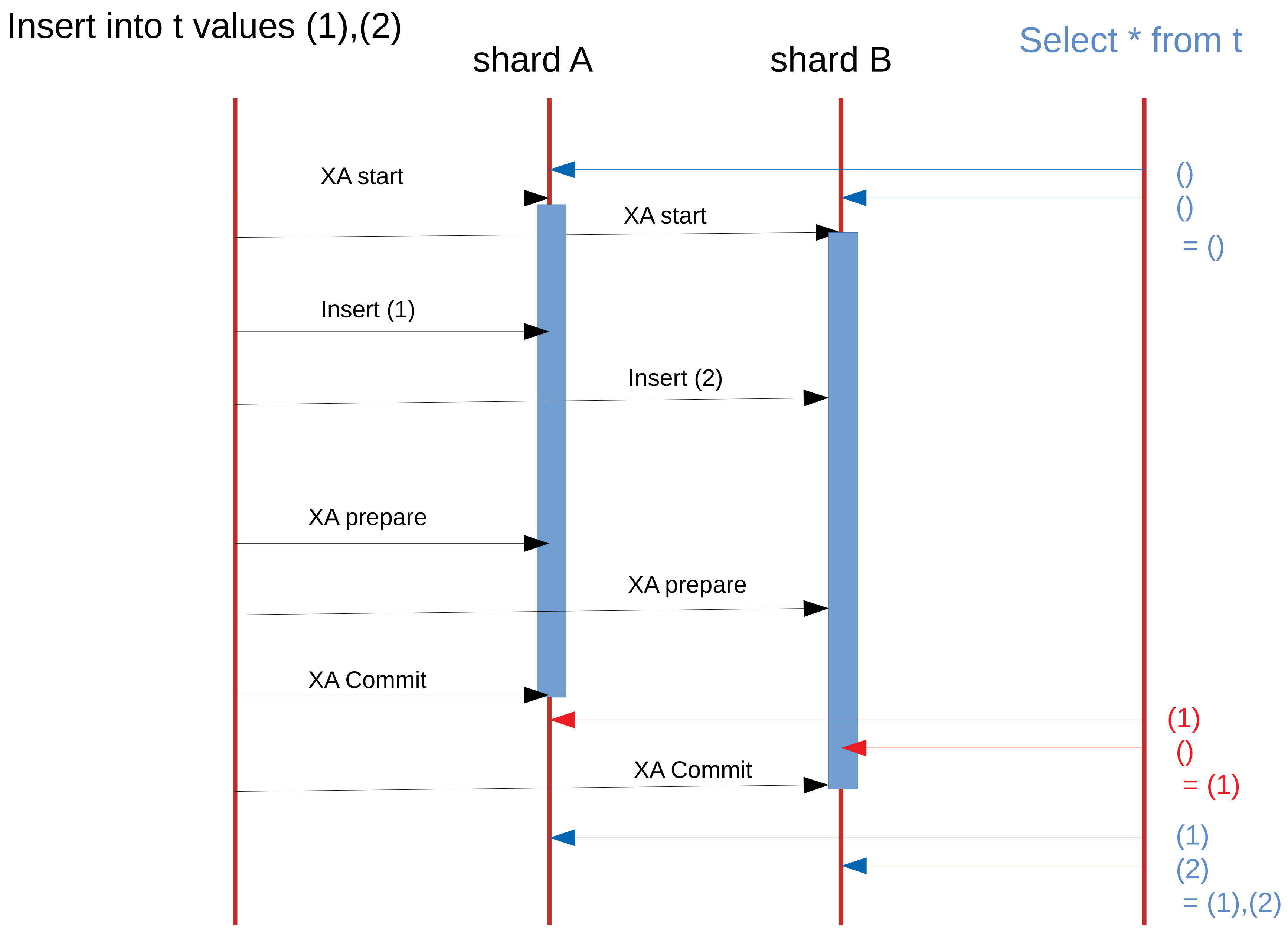

Insert into t values….
Spider SE
Spider SE
Spider SE
Galera
XA
XA
Node 1
Node 2
Node 3
Galera
Node 1
Node 2
garbd
Galera

Spider ACID

Insert into t values (1),(2)



Spider ACID


Insert into t values (1),(2)
shard A
shard B
Select * from t
XA start
XA start
Insert (1)
Insert (2)
XA prepare
XA prepare
XA Commit
XA Commit
()
()
= ()
(1)
()
= (1)
(1)
(2)
= (1),(2)

MariaDB Server MDEV-7974

backport fix for mysql bug#12161 (XA and binlog)

Details

Type:	✓ Task	Status:	IN PROGRESS
Priority:	Major		(View Workflow)
		Resolution:	Unresolved
Component/s:	None	Fix Version/s:	10.4
Labels:	upstream-fixed		
Epic Link:	Replication Enhancements		

Description

5.7 finally fixes the 10-yr-old bug#12161 — a.k.a. *prepared XA transactions are lost on disconnect*. They solved it by introducing a new XA_prepare_log_event. As we'll need to be able to read this event, we can as well merge the whole fix for this bug.

Issue Links

is duplicated by

MDEV-742 LP:803649 - Xa recovery failed on client disconne… OPEN

links to

Bug #12161 Xa recovery and client disconnection

Activity

All | **Comments** | History | Activity | Transitions

Elena Stepanova added a comment - 2017-01-24 12:04

Is it still possible to do it in 10.2? I'll set it to 10.2-ga to get on the radar, but feel free to unset if it can't be done.

People