Postgres-XC クラスタにおける高可用機能

鈴木幸市
Postgres-XC Development Group

PgDay 2012 Japan
November 30th, 2012
Tokyo, Japan

NTT DATA

Outline of the Talk

- Postgres-XC overview
  - What it is
  - Architecture and scalability
- SPOF analysis
- Failure characteristics
  - Comparison with a commercial one
- Failure handling and HA
- Current status and future schedule

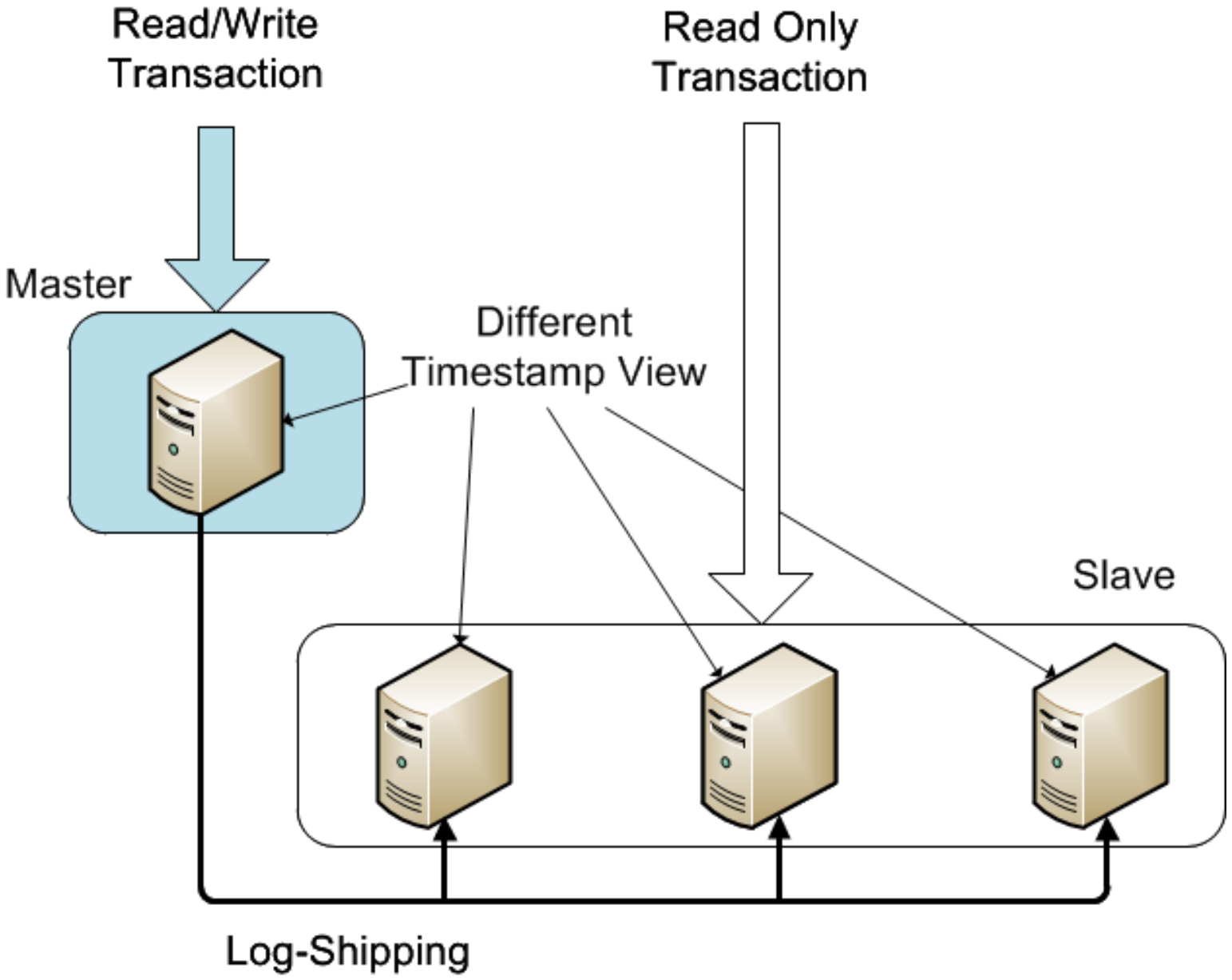
Postgres-XC Overview

NTT DATA

What is Postgres-XC

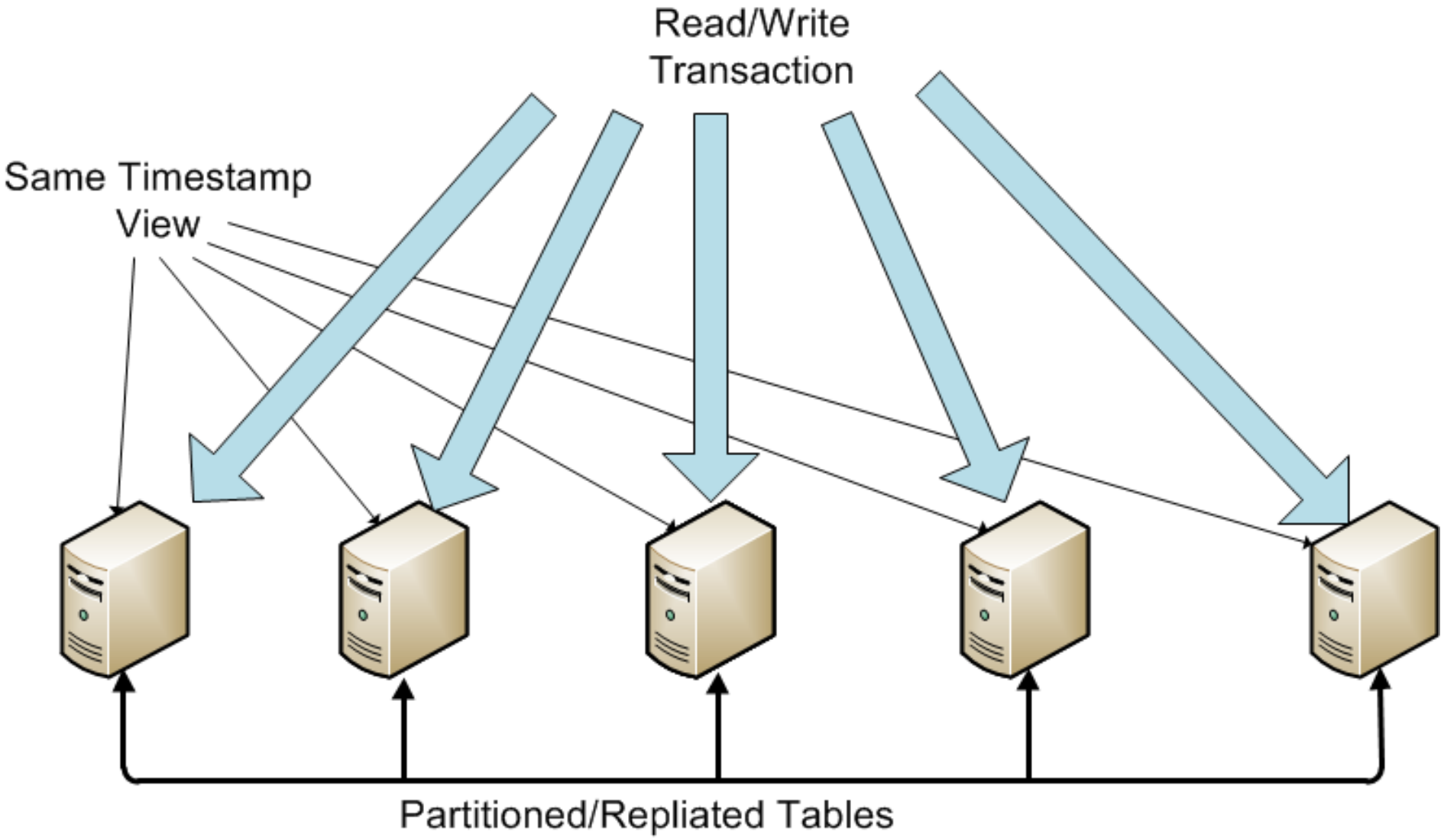
- Symmetric PostgerSQL cluster
  - No master/slave replication
  - No read-only clusters
  - Every node can issue both read/write
  - Every node provides single consistent database view
  - Transparent transaction management
- Not just a replication
  - Each table can be replicated/distributed by sharding
  - Parallel transaction/query execution
    - So both read/write scalability

Master/Slave with Log Shipping



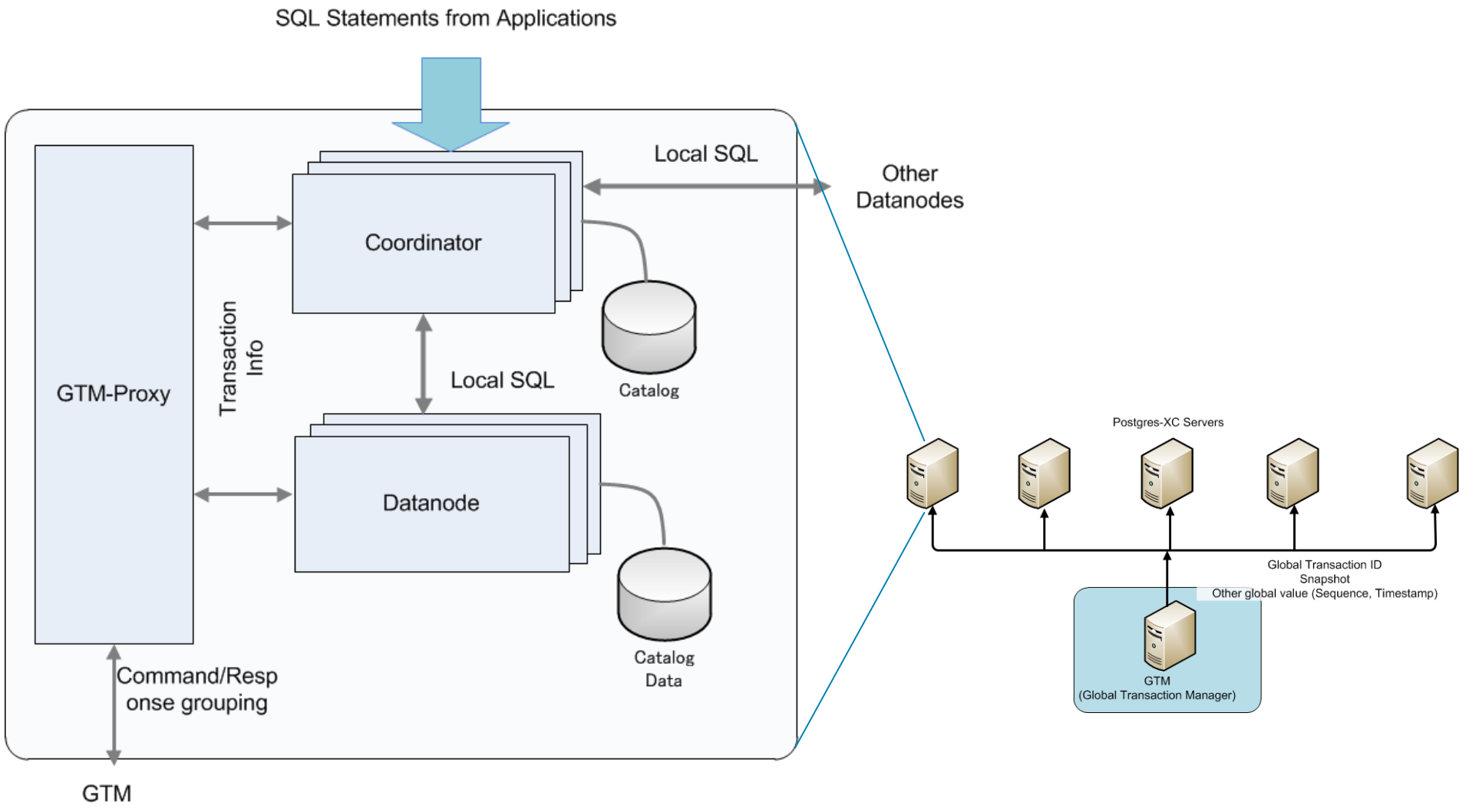
October 24th, 2012

Postgres-XC Symmetric Cluster



October 24th, 2012

Architecture and Configuration



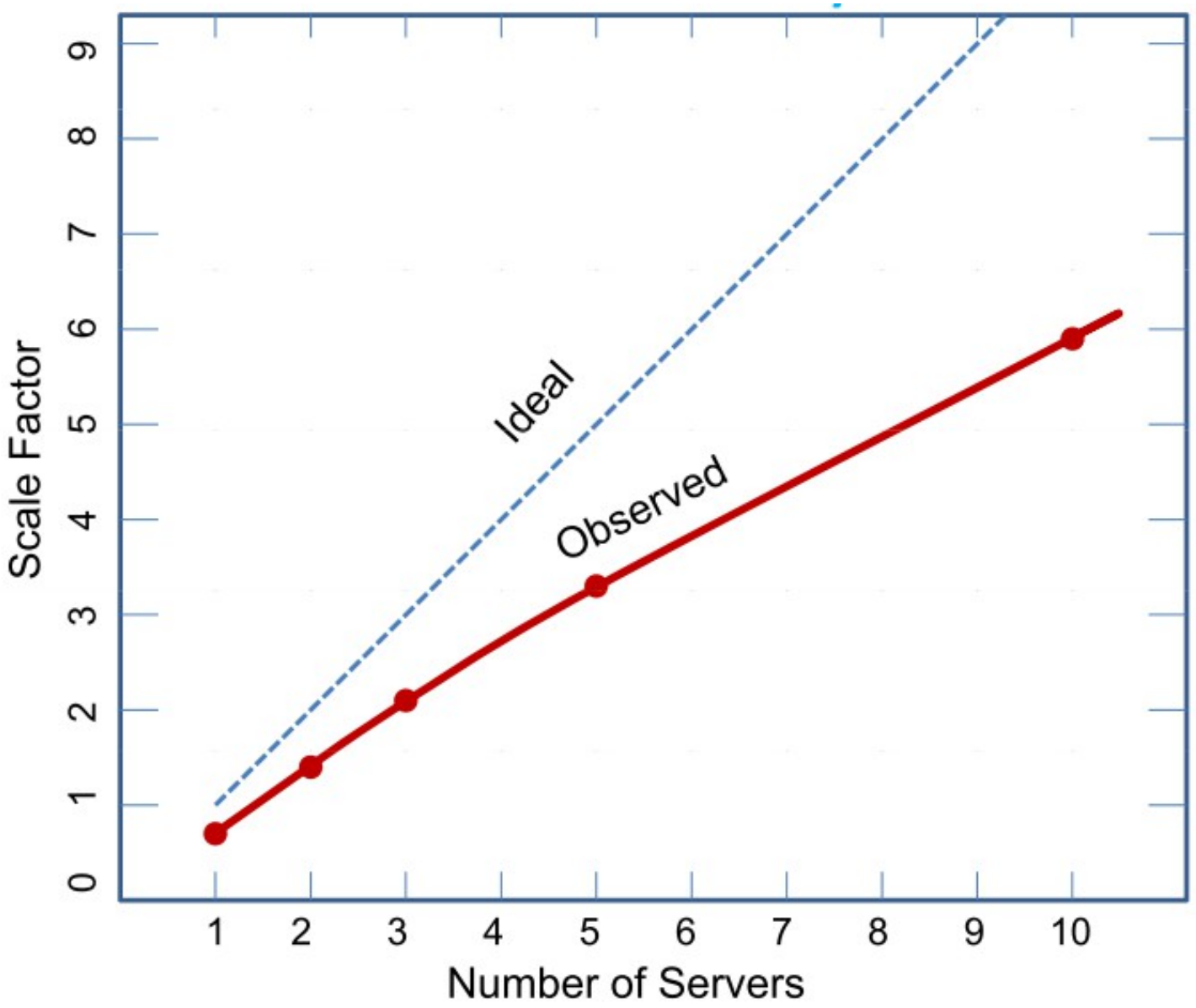
October 24th, 2012

Outline of Components

- GTM (Global Transaction Manager)
  - Distributed MVCC
    - Provide global transaction ID (GXID) to all the transactions
    - Provide global snapshot to all the transactions
  - Sequence
- GTM_Proxy
  - Group communications to GTM and reduce amount of GTM network workload
- Coordinator
  - Handles incoming SQL statements
    - Parse, plan, conduct execution in datanodes and the coordinator.
    - Integrate local results from each datanode involved.
- Datanode
  - Actual data storage
    - Almost vanilla PostgreSQL

Share the binary

Scalability

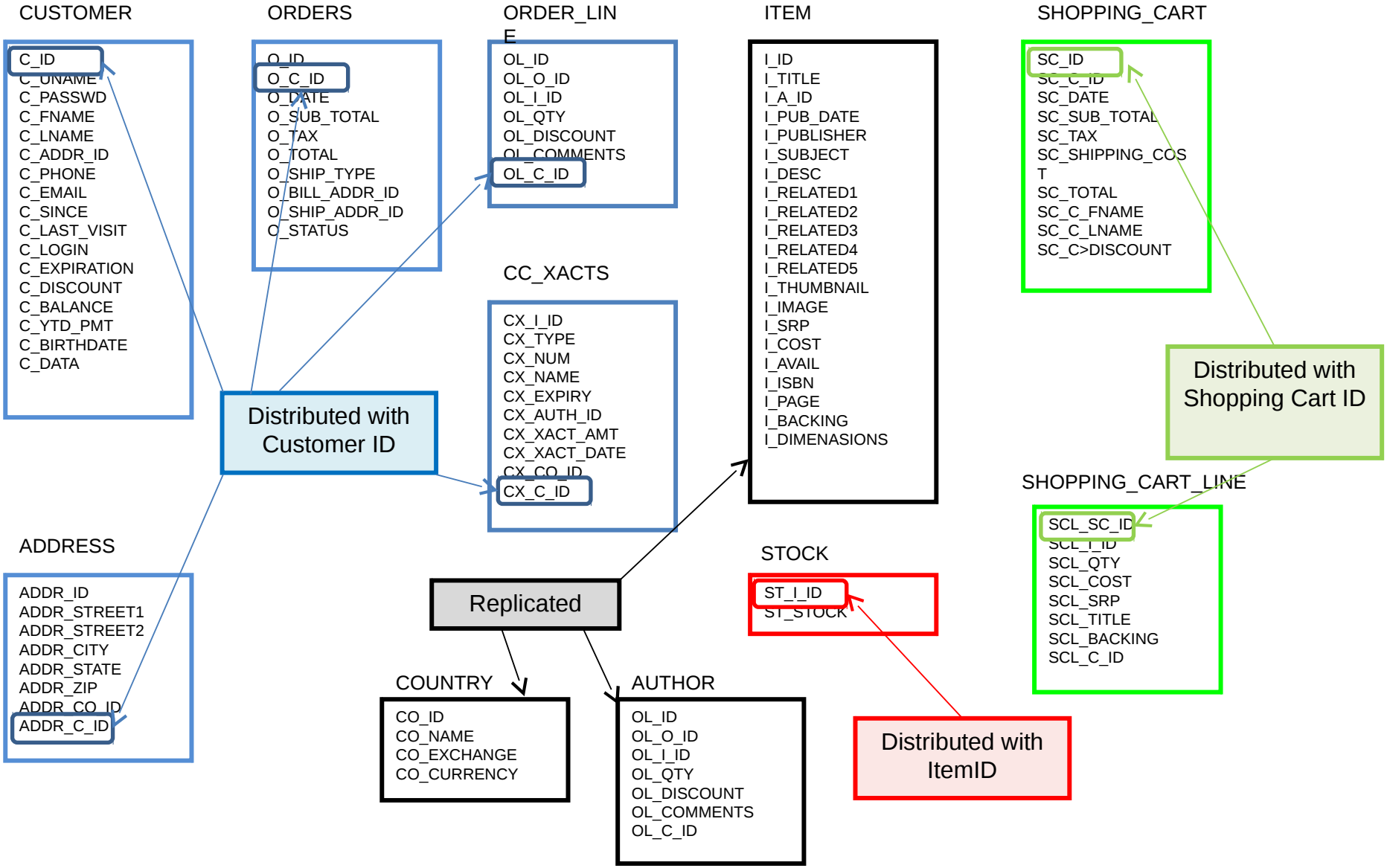


DBT-1 (Rev)

October 24th, 2012

Combining sharding and replication

DBT-1 example



October 24th, 2012

How XC Scales both Read/Write

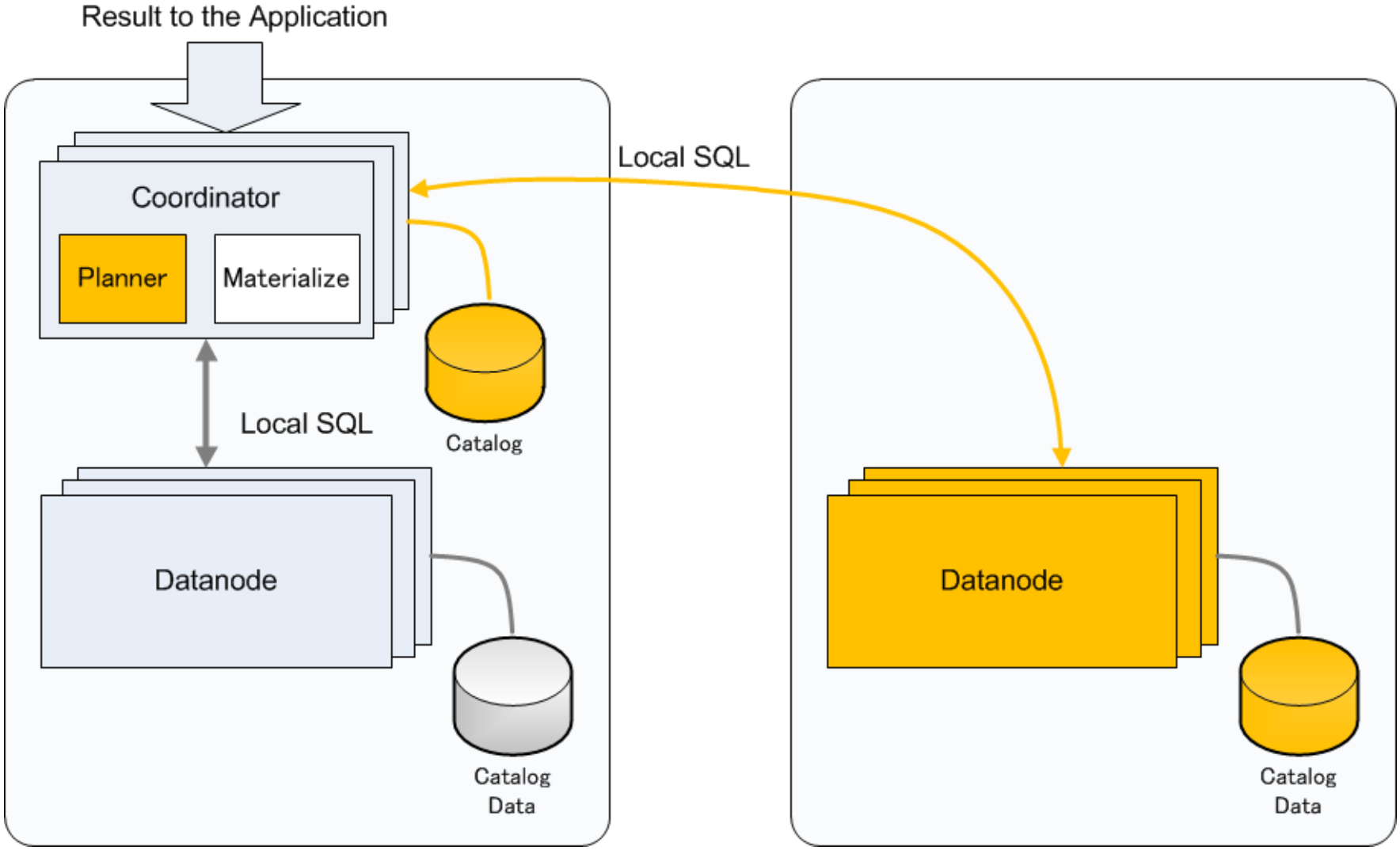
- Transaction Tables → Sharding
  - Only one write
  - Parallel writes in datanodes
- Master Tables → Replication
  - Relatively static: Not significant many-writes overhead
  - Local join with transaction tables → Most join operation can be done locally in datanodes

Postgres-XC API

- Binary compatible with PostgreSQL
  - Limited support for ODBC
  - JDBC may have a couple of restrictions
- Compatible statements to PostgreSQL
  - Slight difference
    - CREATE TABLE, etc.
    - Constraints can be enforced only locally in each datanodes.
  - Extra
    - Coordinator/datanode membership management, etc.
      - CREATE/ALTER/DROP NODE, EXECUTE DIRECT...
    - Extension in aggregates
      - Combiner functions
    - Maintain consistency in point-in-time-recovery
      - CREATE BARRIER
- No load balancing so far
- You should notice
  - OID is local to each node

Join Example (1)

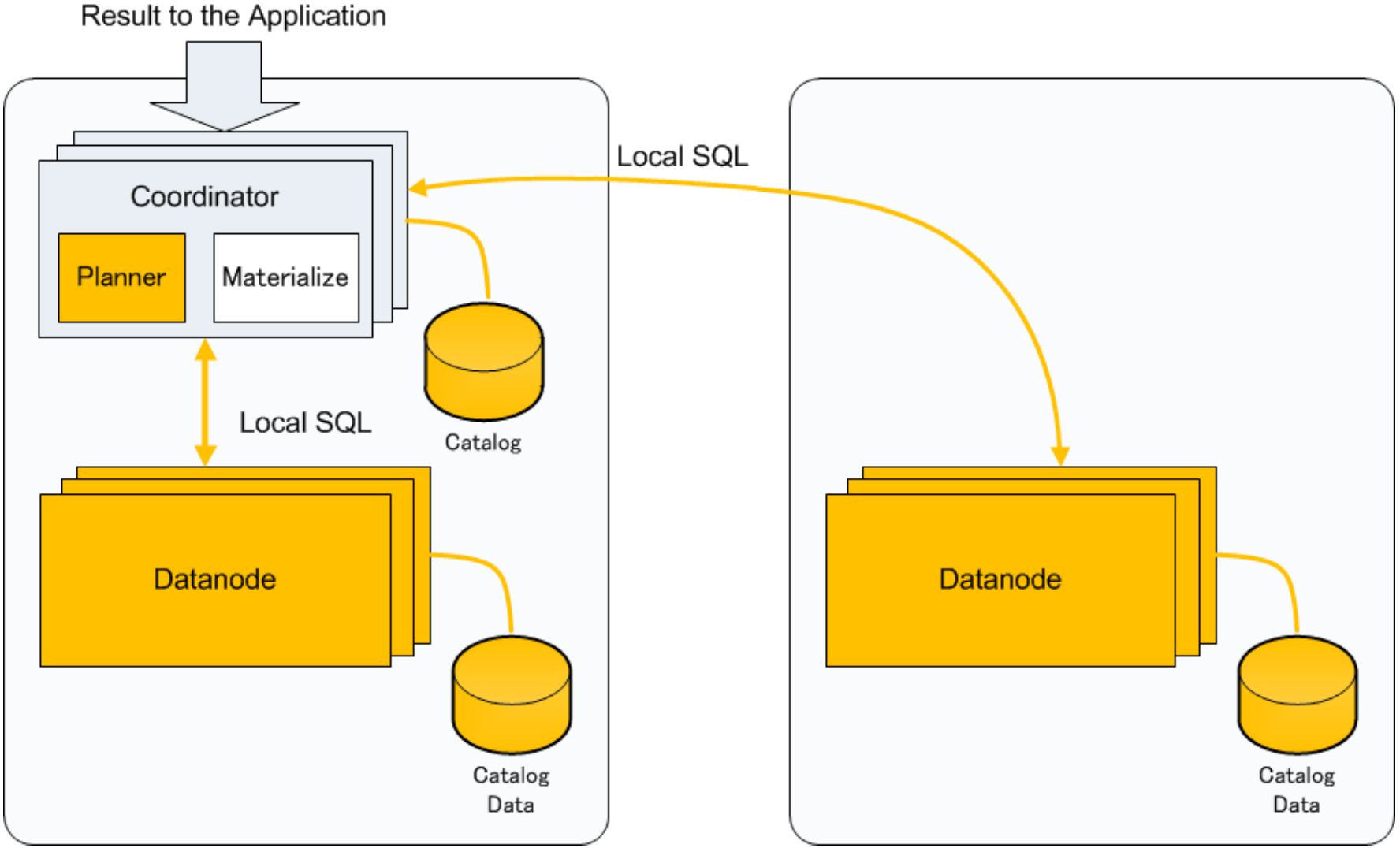
- Replicated Table and Partitioned Table
  - Can determine which datanode to go from WHERE clause



October 24th, 2012

Join Example (2)

- Replicated Table and Partitioned Table
  - Cannot determine which datanode to go



October 24th, 2012

SPOF Analysis

NTT DATA

Postgres-XC Single Point of Failure

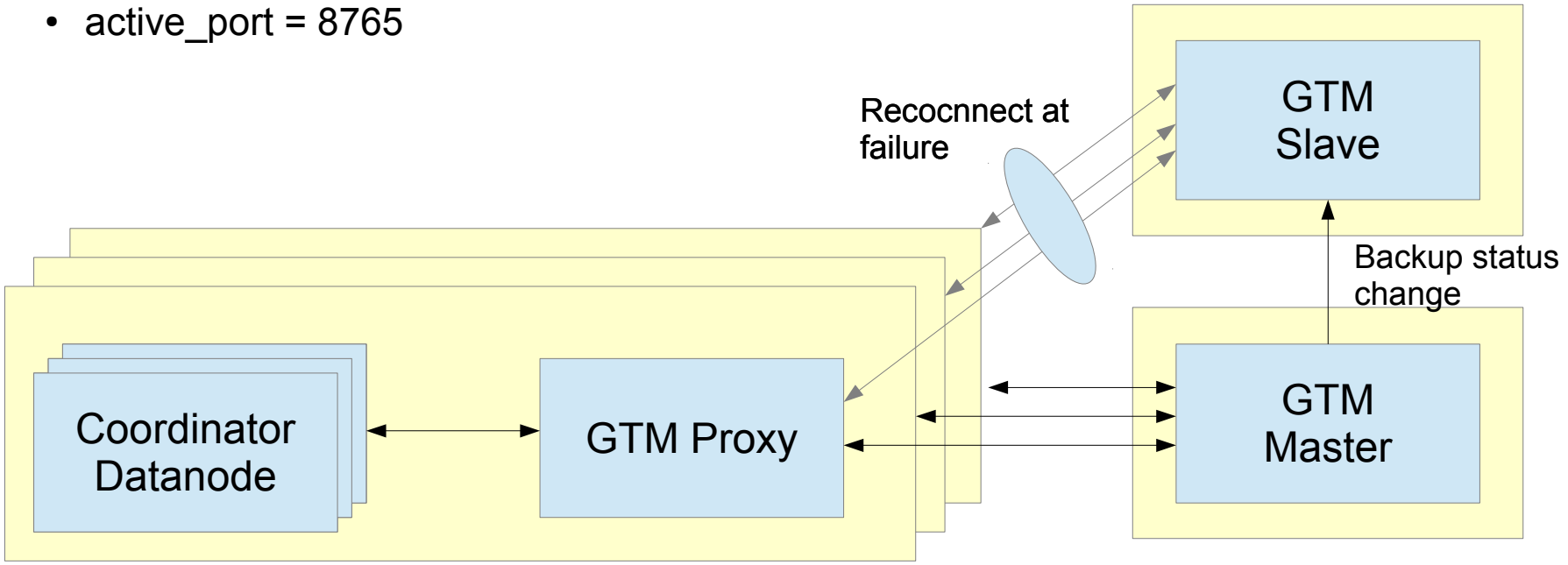
- GTM
  - Obviously SPOF
- GTM-Proxy
  - No persistent data hold
  - Just restart when fail
- Coordinator
  - Every coordinator is essentially a copy
  - When fails, other coordinators work
- Datanode
  - SPOF for sharded table

Backup for SPOF component

- GTM
  - Specific backup for GTM (GTM Standby)
    - Most information are kept on-memory
      - Open TXNs
        - Only the next GXID is needed to restart whole cluster, kept on disk.
    - Copies every internal status change to the backup
      - Similar to the log shipping in PostgreSQL
    - Can promote to the master
      - GTM-Proxy help this failover
- Datanode
  - Need backup
  - Can use PostgreSQL's means
    - Log Shipping
    - Shared disk
- Coordinator
  - Not critical but may want to have backups
  - Can use similar means as Datanodes.

GTM Slave (GTM Standby)

- Same binary to GTM
  - Backs up everything on the fly.
  - Can promote to the master (gtm_ctl promote)
  - Configure using gtm.conf
    - startup = ACT|STANDBY
    - active_host = 'active_gtm_host'
    - active_port = 8765



Datanodes

- Almost all the techniques for PostgreSQL backup/failover are available
  - Streaming replication
  - Shared disk re-mount
- Subject to coordinators
  - Coordinators should reconfigure failed datanode at failover
  - Coordinators should clean connections to failed datanode before reconfiguration
- GTM
  - Reconnect to (new) local GTM proxy

Coordinators

- Only catalog is stored
  - Very stable and static
  - All the coordinators are essentially the same copy
- Datanode HA technique can be applied
  - Streaming replication
  - Shared disk remount
- One more option at a failure
  - No failover
  - Remaining coordinators will take care of TXNs
  - Failed coordinator can be restored offline
    - Backup/restore
    - Copy catalogue from a remaining coordinator

Failure Characteristics

PostgreSQL

- Promote one of the slaves
- Application connect to promoted PostgreSQL
- Everything stops and then restarts

Postgres-XC

- Promote the slave of the failed component
- Reconfigure with new master
- Whole cluster continues to run, only affected TXNs fail

- Just one component failure may not lead to whole cluster fail.
- Can configure appropriate slaves for each component to continue XC operation.

XC vs. O*R*

Feature	Postgres-XC	O____ R__
Background Databsae	PostgreSQL	O_____
Architecture	Shared Nothing	Shared Everything
Number of Servers	Experience: 10 Maybe 20 or more	?? (Most deployments are two server configuration)
Hardware Requirement	None	Shared Disk
Read Scale	Yes	Yes
Write Scale	Yes	Depends (Application level partitioning)
Storage Failure	Limited impact Component failover Cluster keeps running	Whole cluster fails Cluster-wide failover
Server Failure	Affected components needs failover Others keep running	Remaining servers continues service

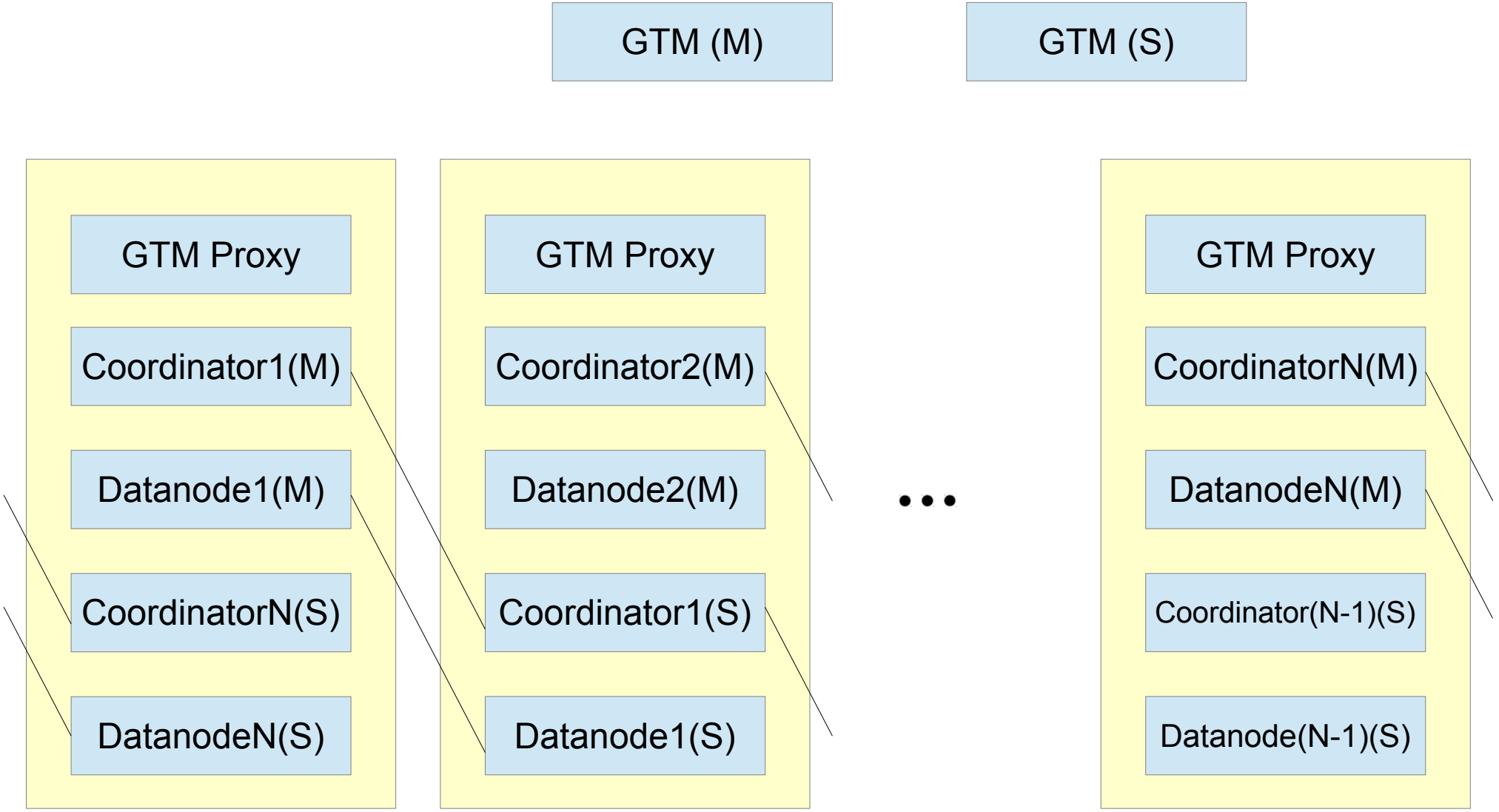
Failure Handling and HA

NTT DATA

General

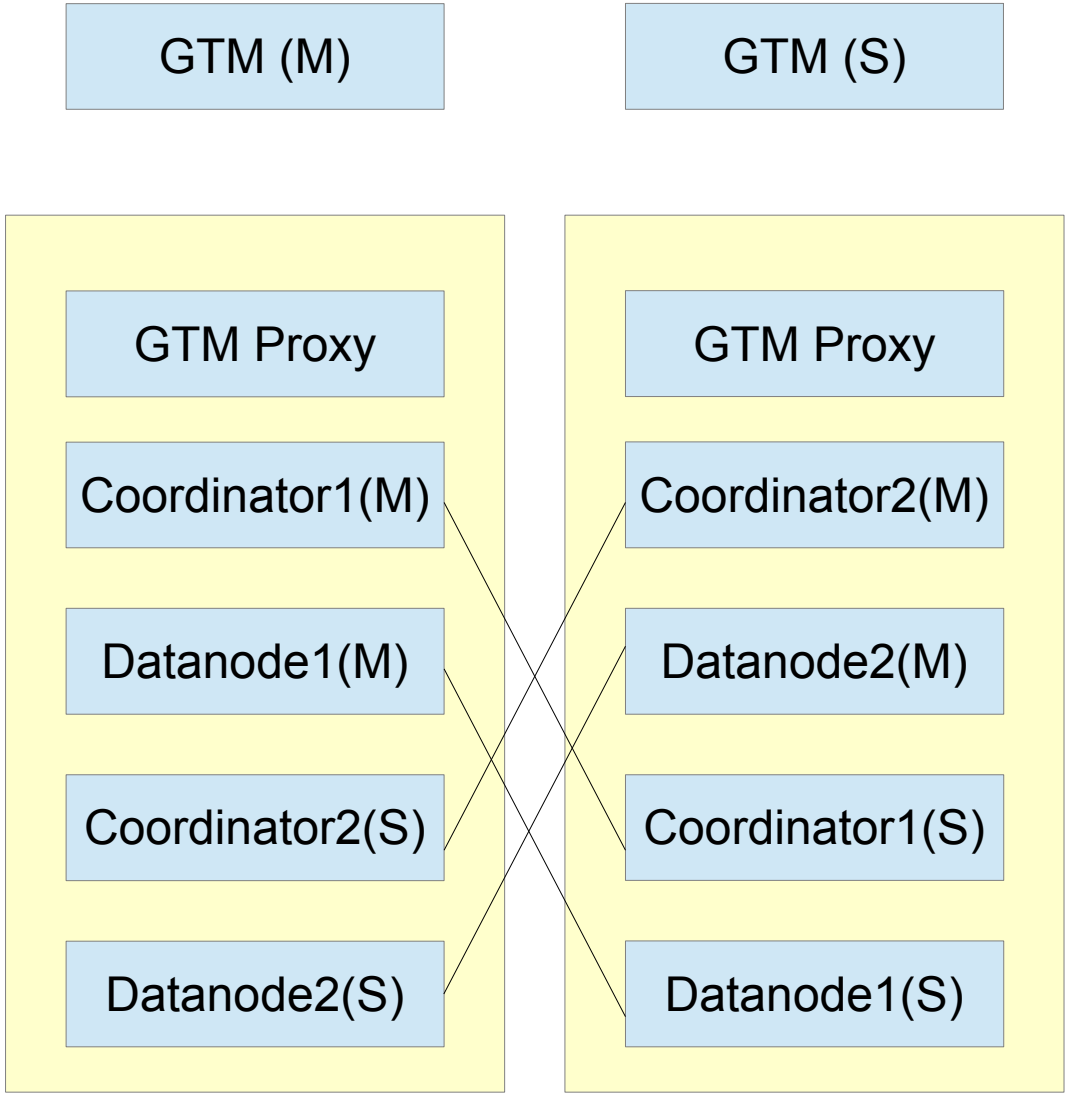
- XC is not a simple replication
  - When a component fails, other components is still alive and can continue to provide database service.
  - Remaining components may need reconfiguration to accommodate new master of the failed component.
- This section shows what to do to prepare slaves and failover each component
- Useful shell scripts will be found in `pgxc_ctl` directory in `https://github.com/koichi-szk/PGXC-Tools`

Circular Configuration



October 24th, 2012

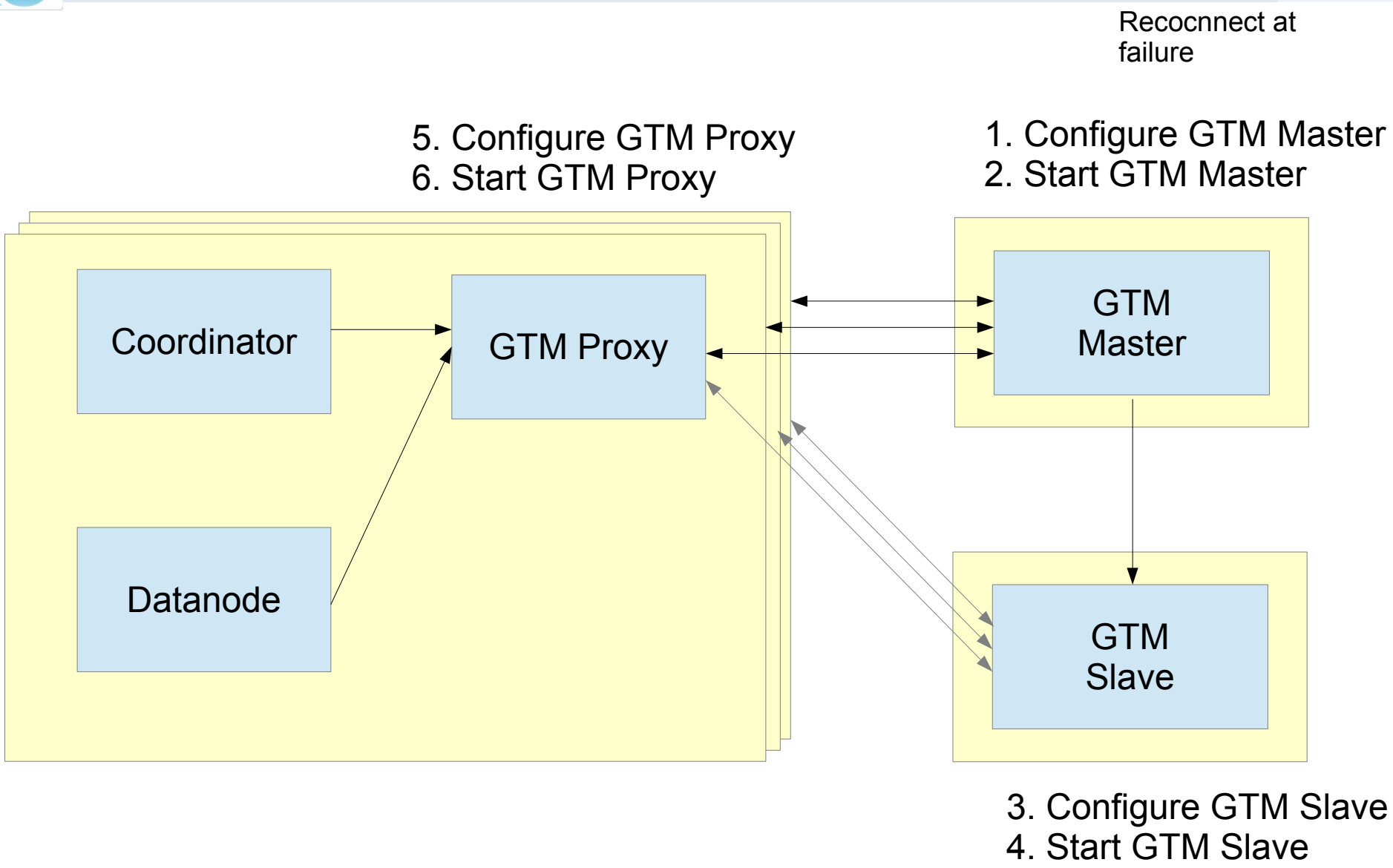
Pair Configuration



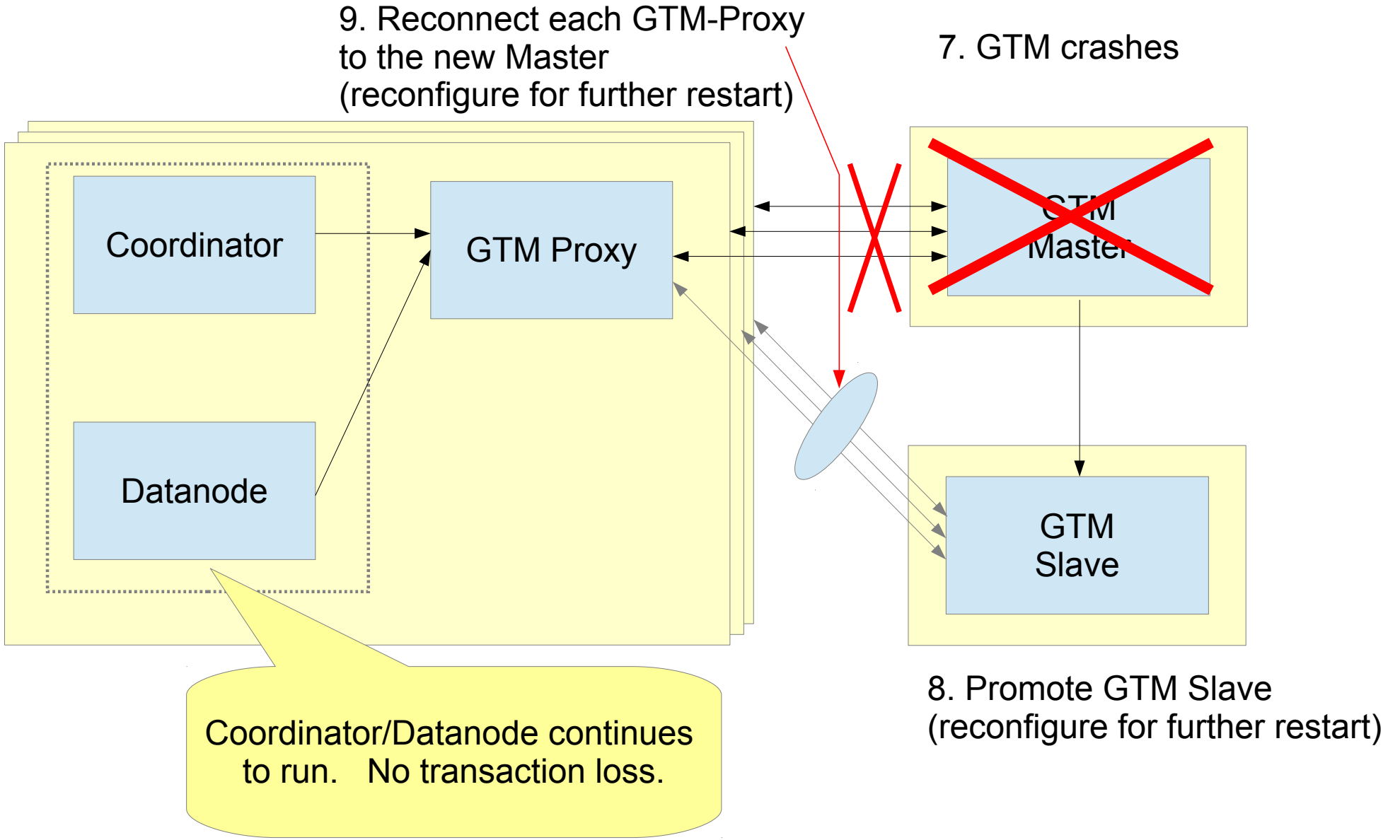
• • •

Easier integration with Pacemaker

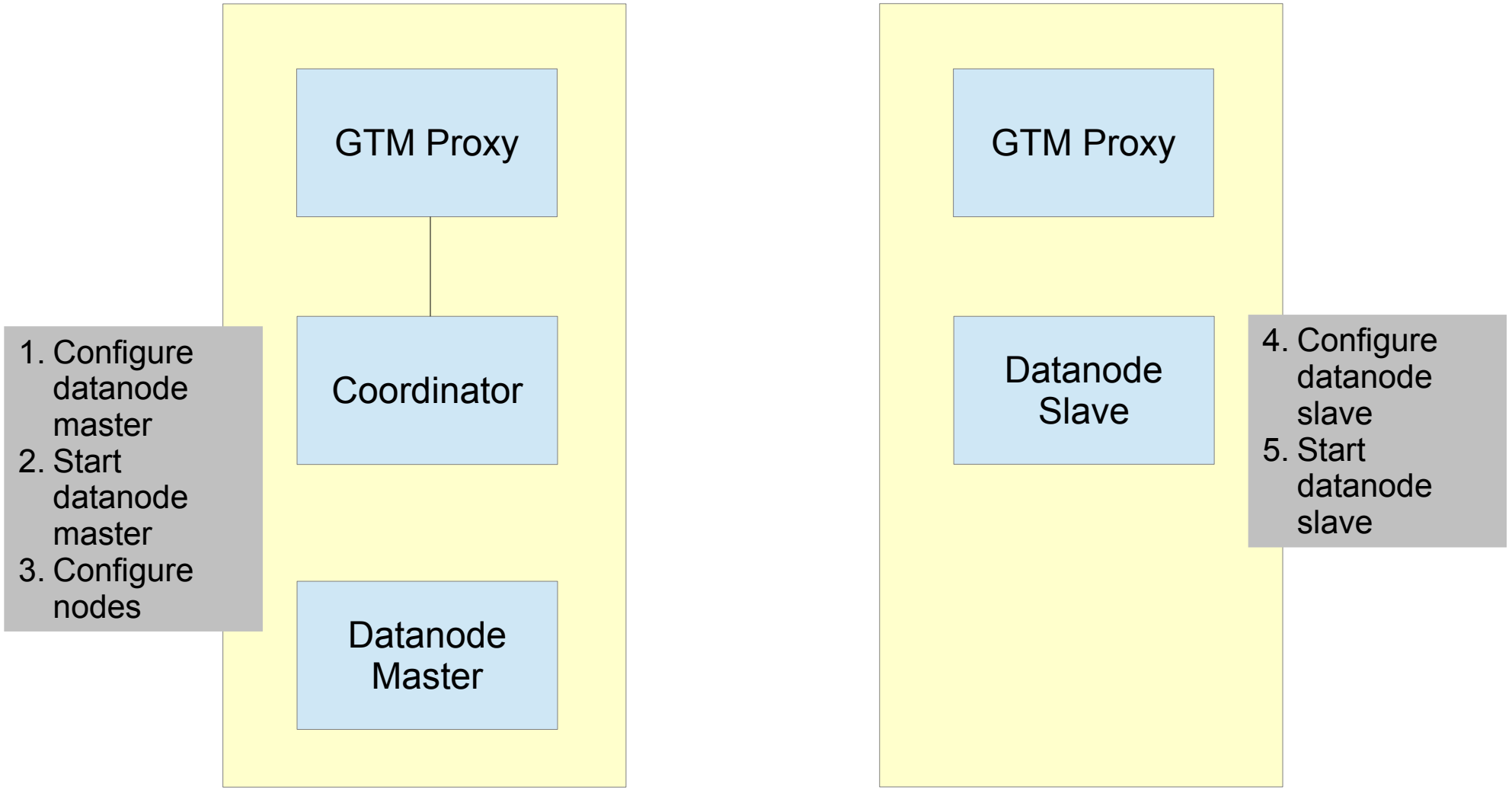
Failure handling outline – GTM(1)



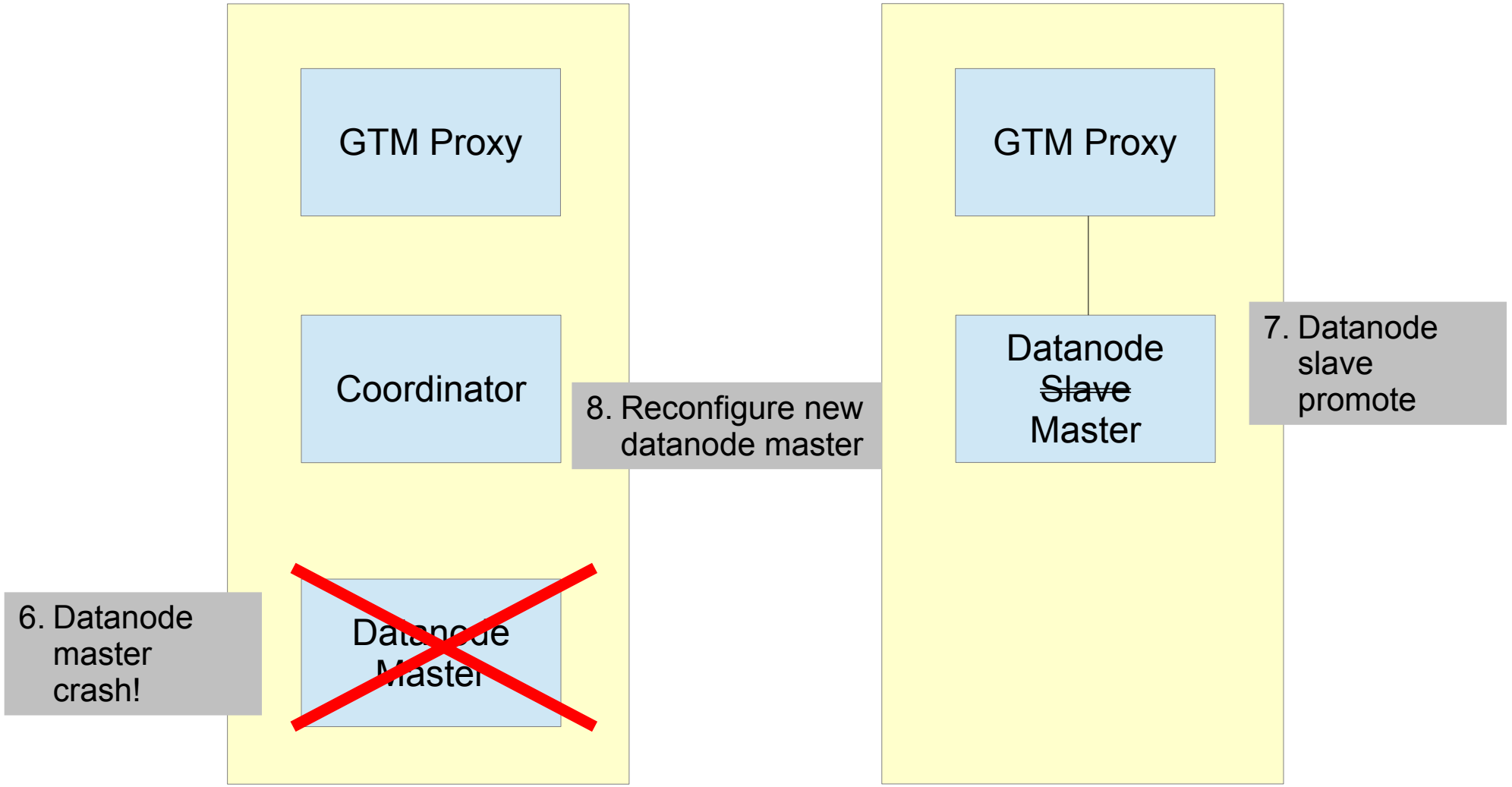
Failure handling outline – GTM(2)



Failure Handling Outline - Datanode(1)



Failure Handling Outline - Coordinator(2)



Useful Materials

- Postgres-XC
  - http://postgres-xc.sourceforge.net/ (project web site)
  - http://sourceforge.net/projects/postgres-xc/ (development site)
  - http://postgresxc.wikia.com/wiki/Postgres-XC_Wiki (Wiki site)
  - https://github.com/koichi-szk/PGXC-Tools (pgxc_ctl, gtm_util tools)
- PostgreSQL resource agents for Pacemaker/Heartbeat
  - sf-ex : mount filesystem exclusively
    - https://github.com/ClusterLabs/resource-agents/blob/master/heartbeat/sfex
  - postgres – streaming replication
    - https://github.com/ClusterLabs/resource-agents/blob/master/heartbeat/pgsql