

Best Practice for Designing and Implementing MySQL Geographic Distributed, High Availability Solutions

Marco Tusa
Percona



PERCONA
LIVEONLINE

About Me

- Open source enthusiast
- MySQL tech lead; principal architect
- Working in DB/development world over 33 years (yes, I am that old)
- Open source developer and community contributor



Why Are We Talking About Geographic Distribution?

The need to have geographic distribution for...



Why Are We Talking About Geographic Distribution?

The need to have geographic distribution for...

Disaster
Recovery



Why Are We Talking About Geographic Distribution?

The need to have geographic distribution for...



Data close to consumer/user

Disaster
Recovery



Why Are We Talking About Geographic Distribution?

The need to have geographic distribution for...

Disaster
Recovery



Data close to
consumer/user



Security by
regulations/laws



Why Are We Talking About Geographic Distribution?

The need to have geographic distribution for...

Disaster
Recovery



Data close to
consumer/user



Security by
regulations/laws



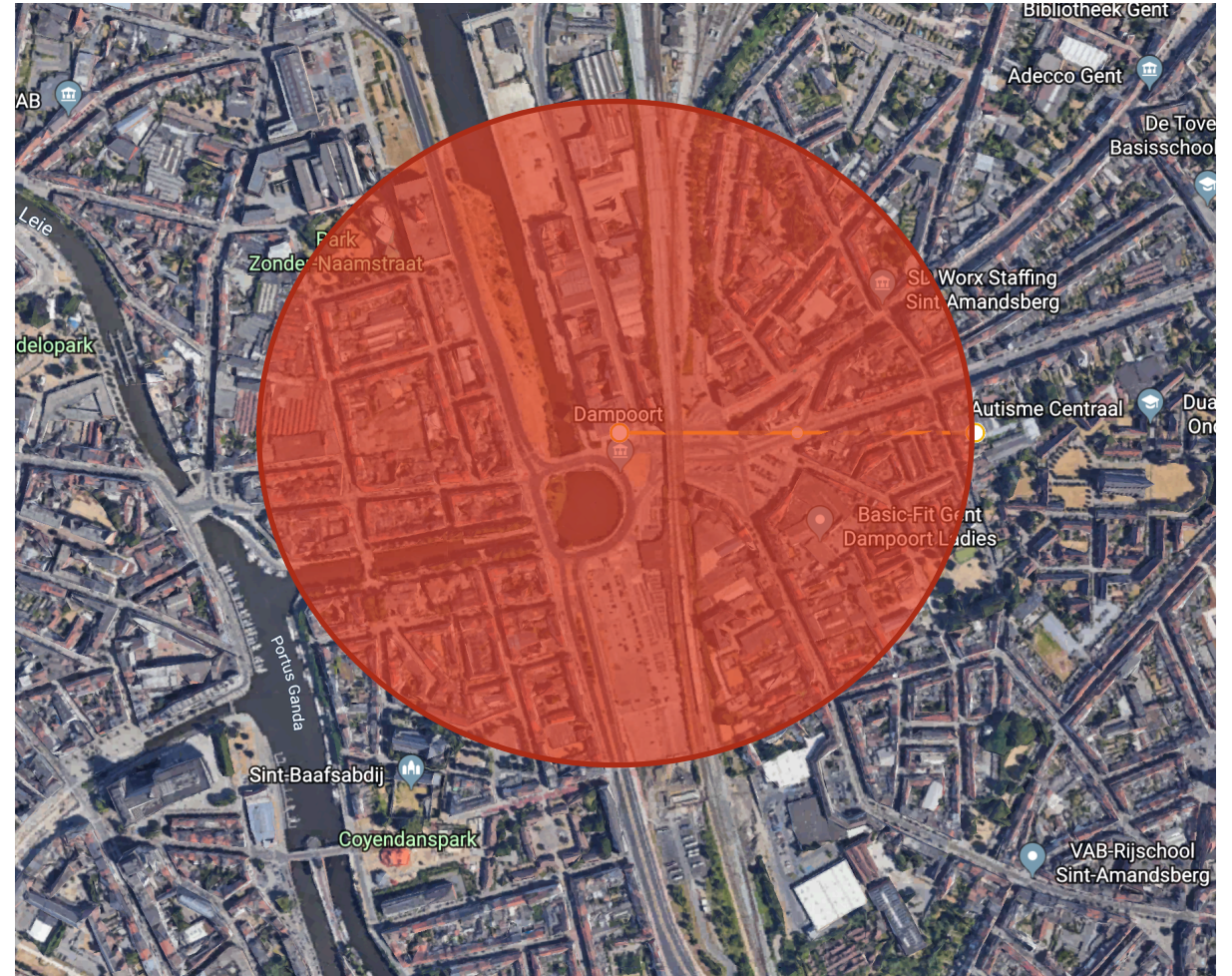
What is Geographic Distribution?

- What is geographic distribution?
 - Near → High Availability
 - Far → Disaster Recovery



What is HA and What is DR?

- **Geographic distribution in High Availability** is to cover service availability in a location
 - **10 Gb Ethernet best case scenario**
400 metre distance max



What is HA and What is DR?

- **Geographic distribution in Disaster Recovery** is to assure we can restore service, in a geographically distributed location
 - Real speed may vary
 - Linear distance ~1000Km



The Impact of Distance on Geographic Distribution



A Real-Life Example

I worked on many cases where a customer had two data centers (DC). I will use information from one of the case as example.

The DC were at a distance of approximately 400 km, connected with a “fiber channel”.

Server 1 and Server 2 were hosted in the same DC, while Server 3 was in the secondary DC.

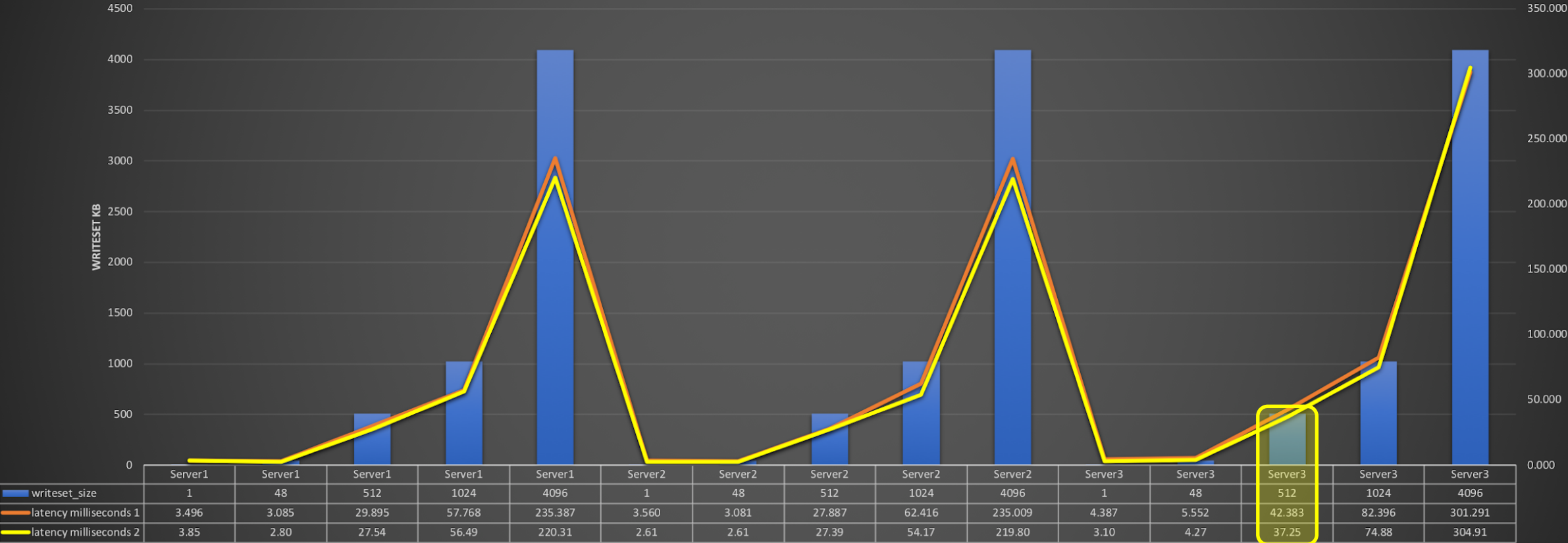
Their ping to Server3 was ~3ms. Not bad at all, right?

We decided to perform some serious tests, running multiple sets of tests with netperf for many days collecting data. We also used the data to perform additional fine-tuning on the TCP/IP layer AND at the network provider.



A Real-Life Example

Comparing Latency by Host and writeset dimension



Observations

37ms latency is not very high. If that had been the top limit, it would have worked.

But, it was not.

In the presence of the optimized channel, with fiber and so on, when the tests were hitting heavy traffic, the congestion was enough to compromise the data transmitted.

It hit a latency $>200\text{ms}$ for Server 3. Note those were spikes, but if you are in the presence of a tightly coupled database cluster, those events can become failures in applying the data, and can create a lot of instability.



Facts About Server 3

The connection between the two was with fiber.

Distance Km ~400 (~800), we need to double because of the round trip, we also receive packages.

Theoretical time at light-speed =2.66ms (2 ways)

Ping = 3.10ms (signal traveling at ~80% of the light speed) as if the signal had traveled ~930Km (full round trip 800 Km)

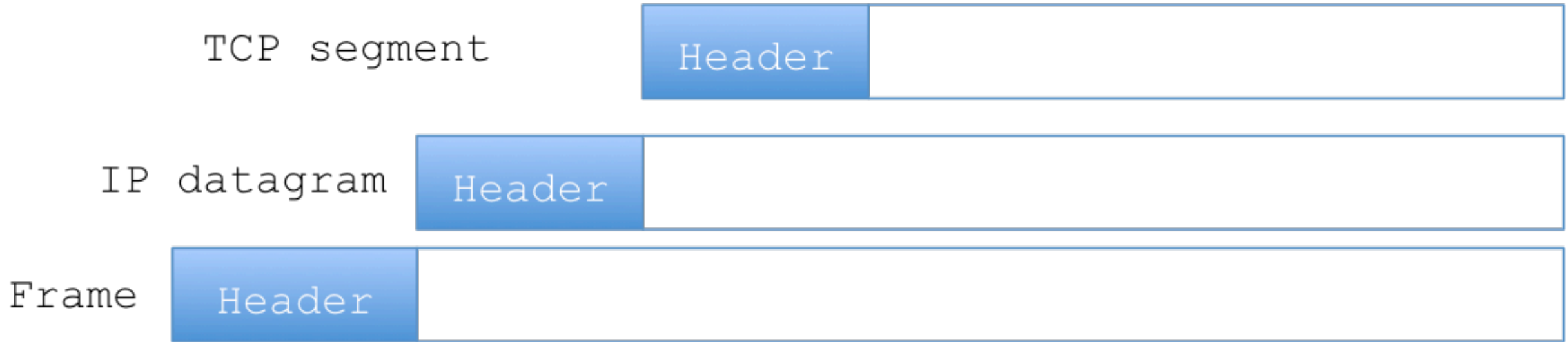
TCP/IP best at 48K = 4.27ms (~62% light speed) as if the signal had traveled ~1,281km

TCP/IP best at 512K =37.25ms (~2.6% light speed) as if the signal had traveled ~11,175km

Given the above, we have from ~20%-~40% to ~97% loss from the theoretical transmission rate.



TCP Encapsulation



Max transportable 1500 MTU – IP Header – TCP Header 1500 – ~40 = 1460 bytes



TCP Sliding Window

Sliding window initial position



Sliding window pointers



Octets up to 2
dispatched
& acknowledge

Octets up to 5
dispatched
not acknowledge

Octets up to 7
still to dispatch



Some Numbers

- With 8KB we need 6 IP Frames
- With 40KB we need 28 IP Frames
- With 387KB we need 271 IP Frames
- With 1MB we need 718 IP Frames
- With 4MB we need ~2,800 Frames

All this if we use the full TCP capacity

Which Solutions Are Available?

2 Models

Tightly coupled database clusters

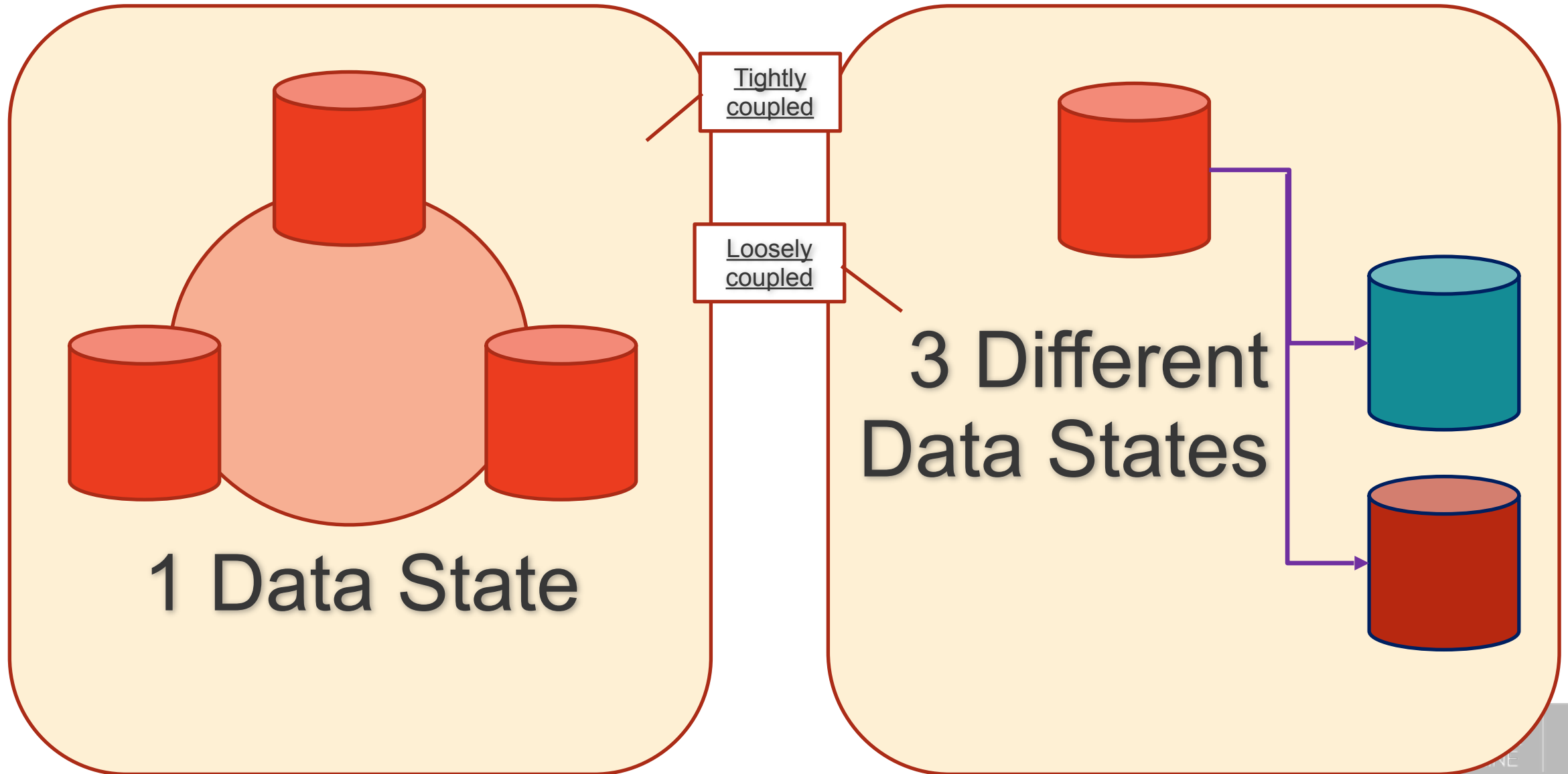
- Datacentric approach (single state of the data, distributed commit)
- Data is consistent in time cross nodes
- **Replication requires high performant link**
- **Geographic distribution is forbidden**
- **DR is not supported**

Loosely coupled database clusters

- **Single node approach (local commit)**
- **Data state differs by node**
- Single node state does not affect the cluster
- Replication link doesn't need to be high performance
- Geographic distribution is allowed
- DR is supported



Replicate Data is the Key - Sync vs Async



Which Implementations?: **Primary - Secondary**

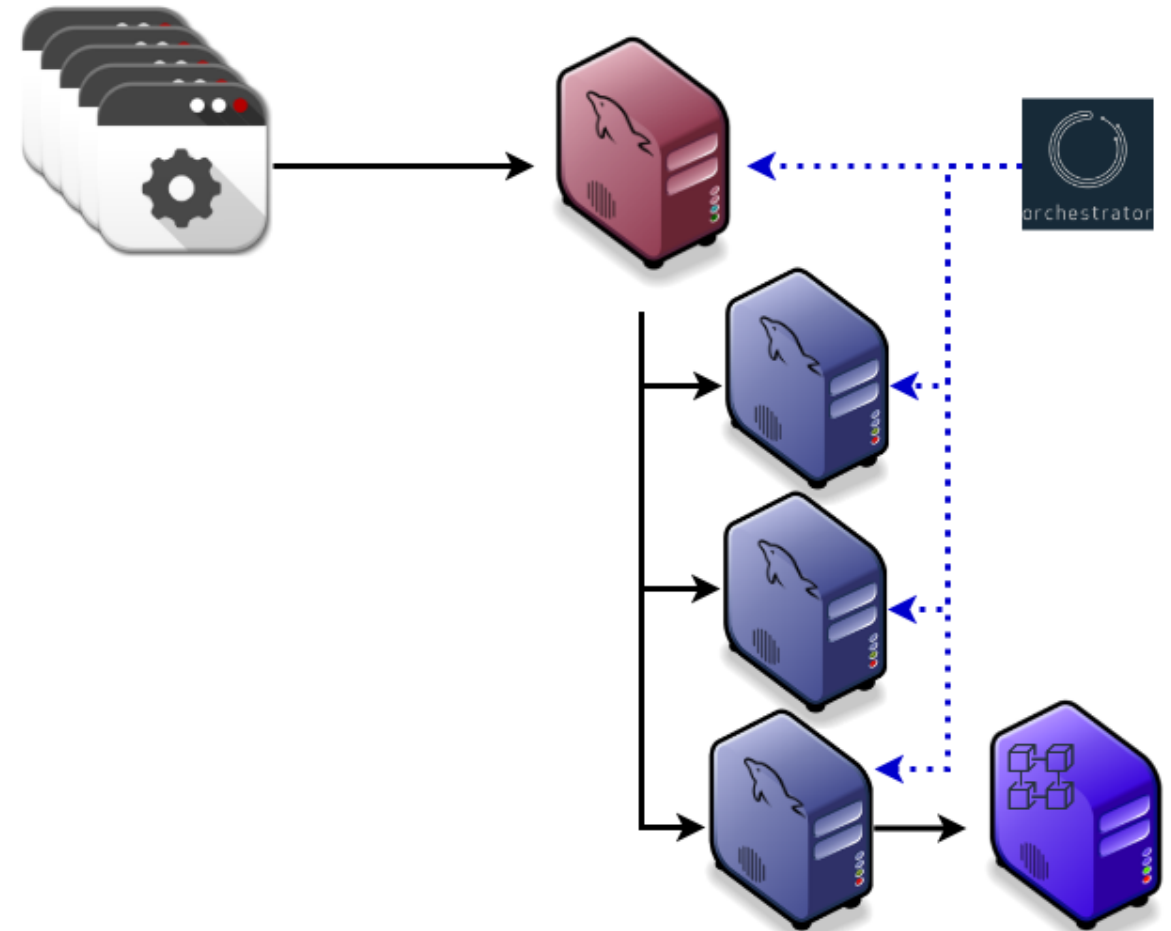
Replication by internal design

Positive things:

- Each node is independent
- Read scale
- Low network impact

Negative things:

- Stale reads
- Low HA
- Consistent only in the Primary
 - Each node has its own data state



Loosely coupled

Which Implementations?: PXC

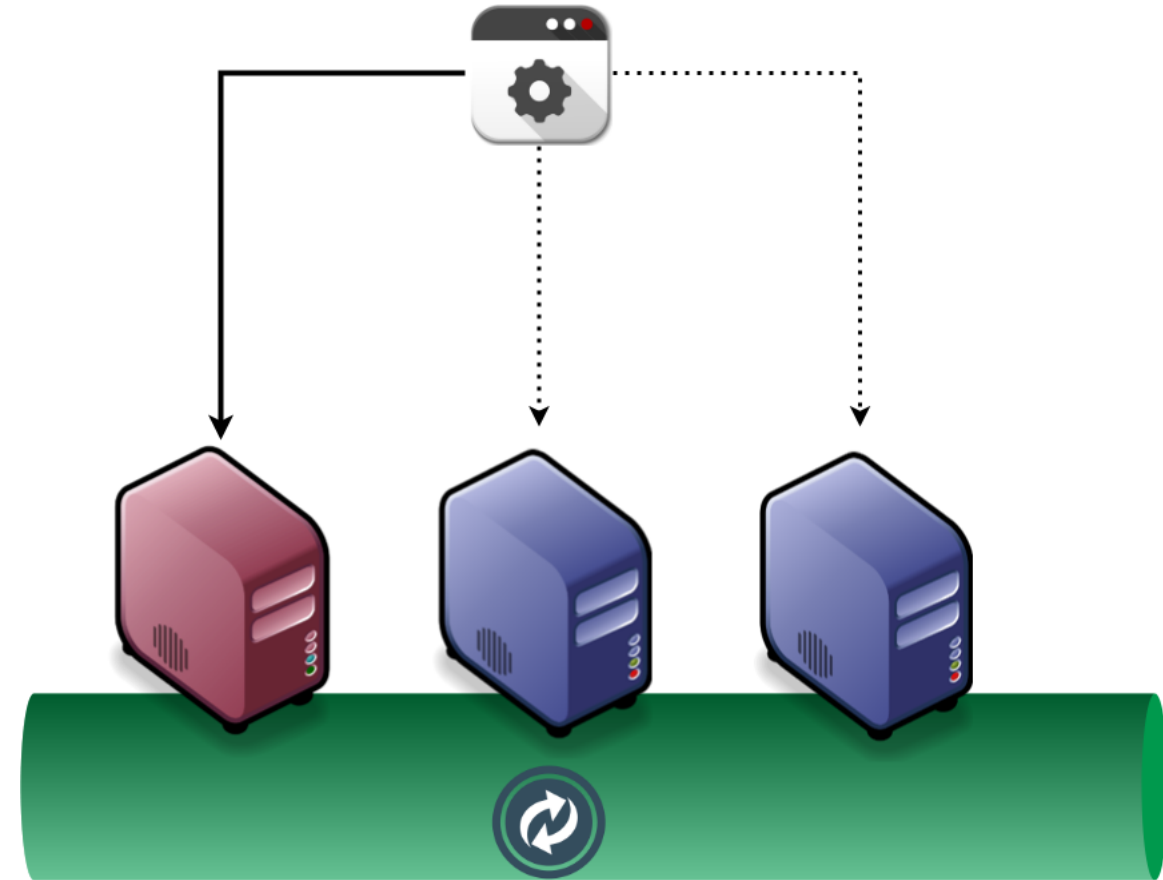
Replication by internal design

Positive things:

- Highly available
- Read scale
- Data is in almost in sync all of the time

Negative things:

- Doesn't scale writes
- More nodes = more internal overhead
- Network is a very impacting factor
- 1 Writer only (PLEASE!!!!!!)



Tightly coupled

Which Implementations?: PS – Group Replication

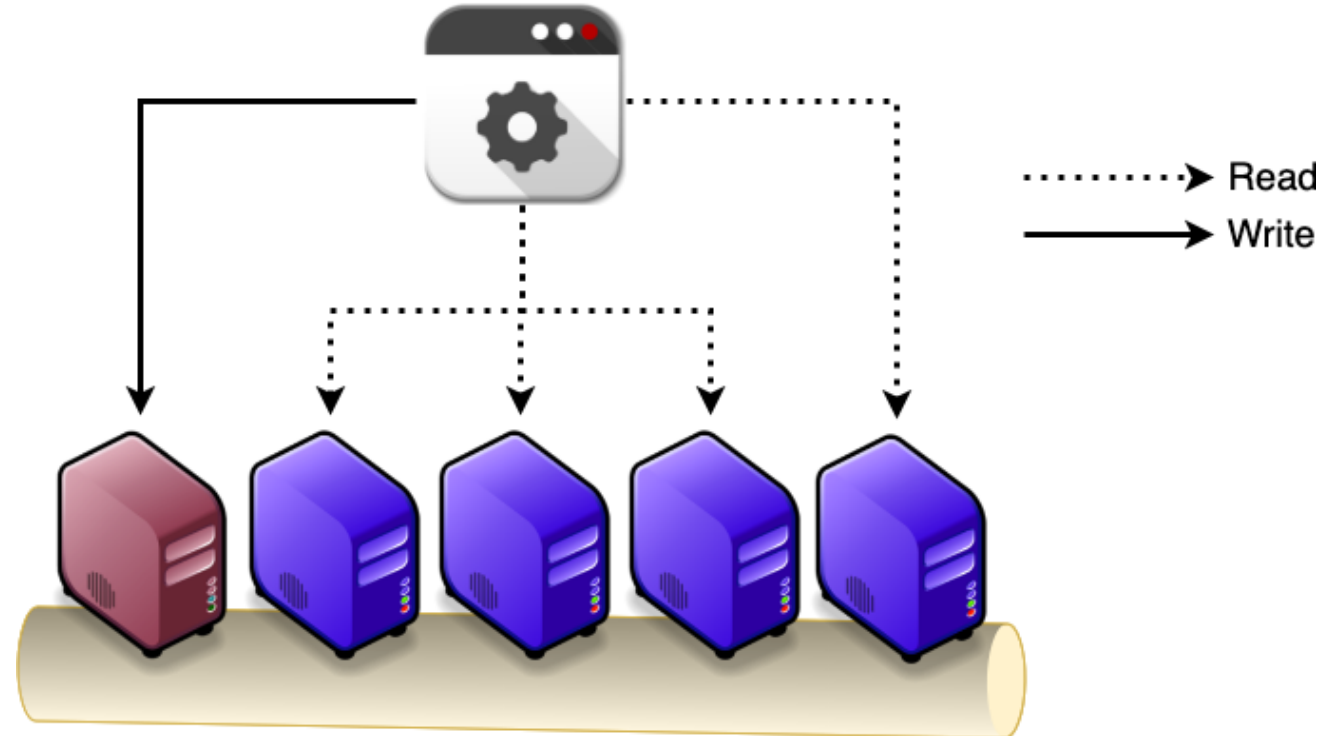
Replication by standard MySQL

Positive things:

- Highly available
- Read scale
- Data is almost in sync all the time
- More network tolerant

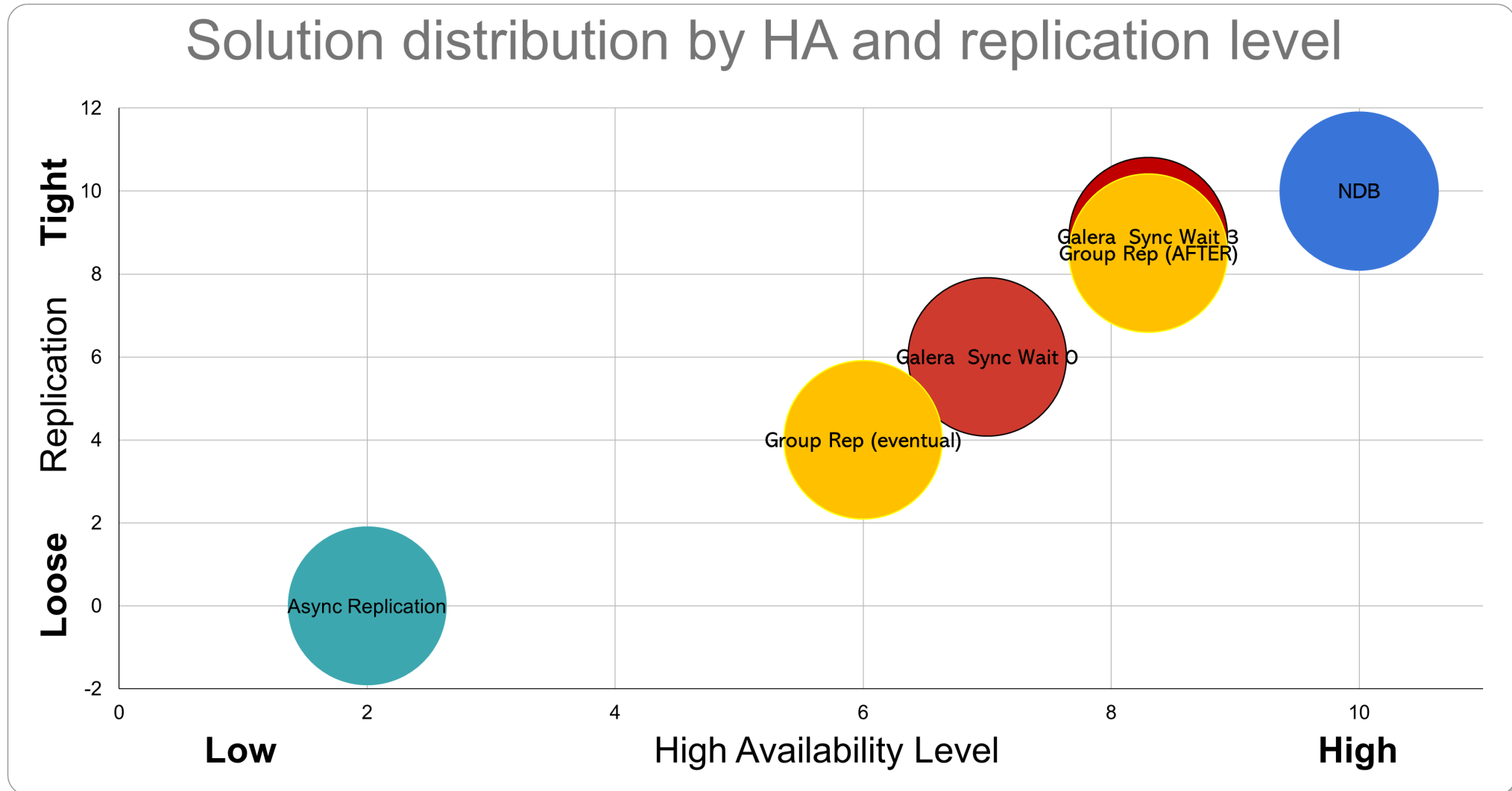
Negative things:

- Doesn't scale writes
- 1 Writer only (PLEASE!!!!!!)

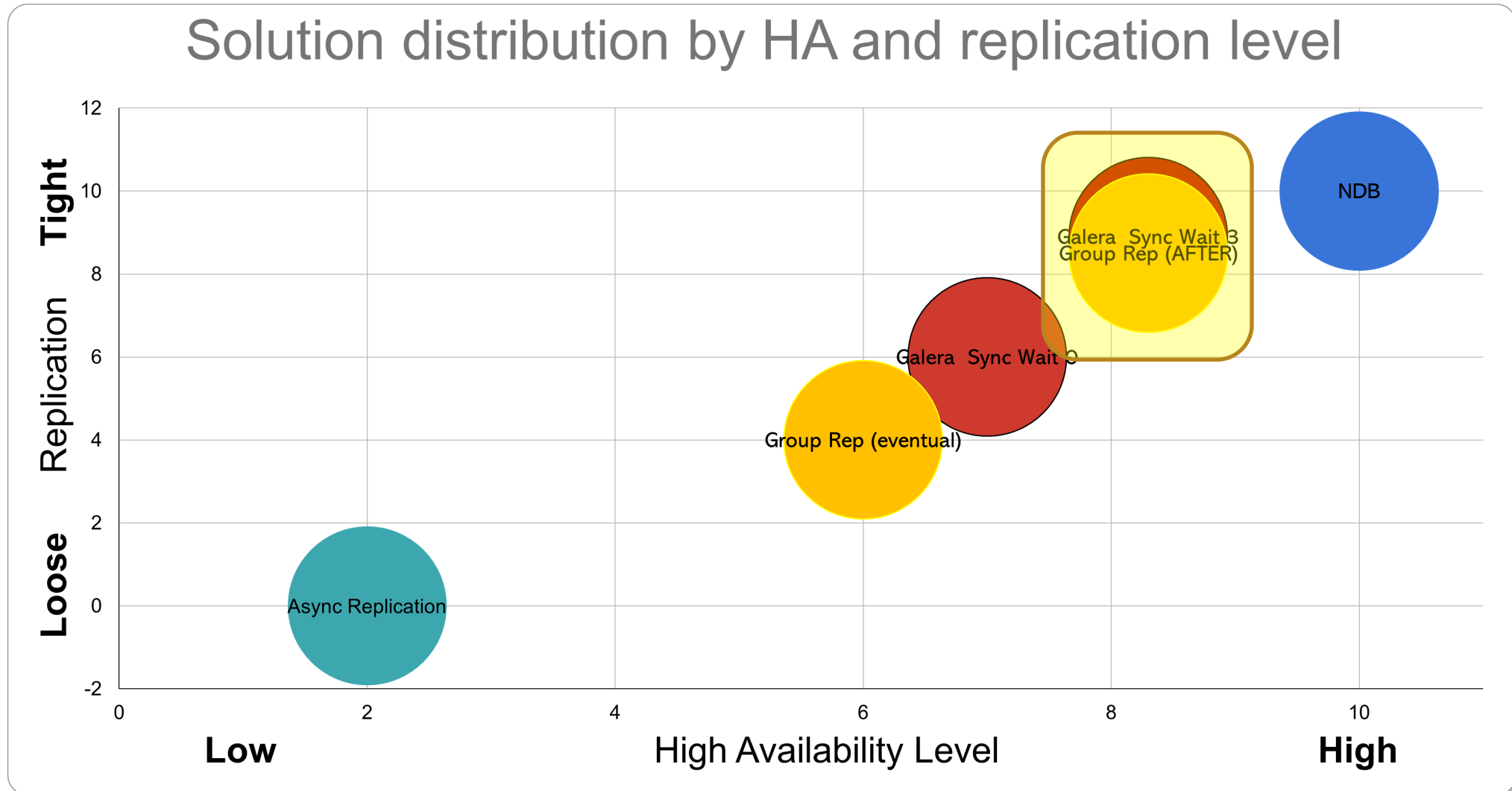


Tightly coupled

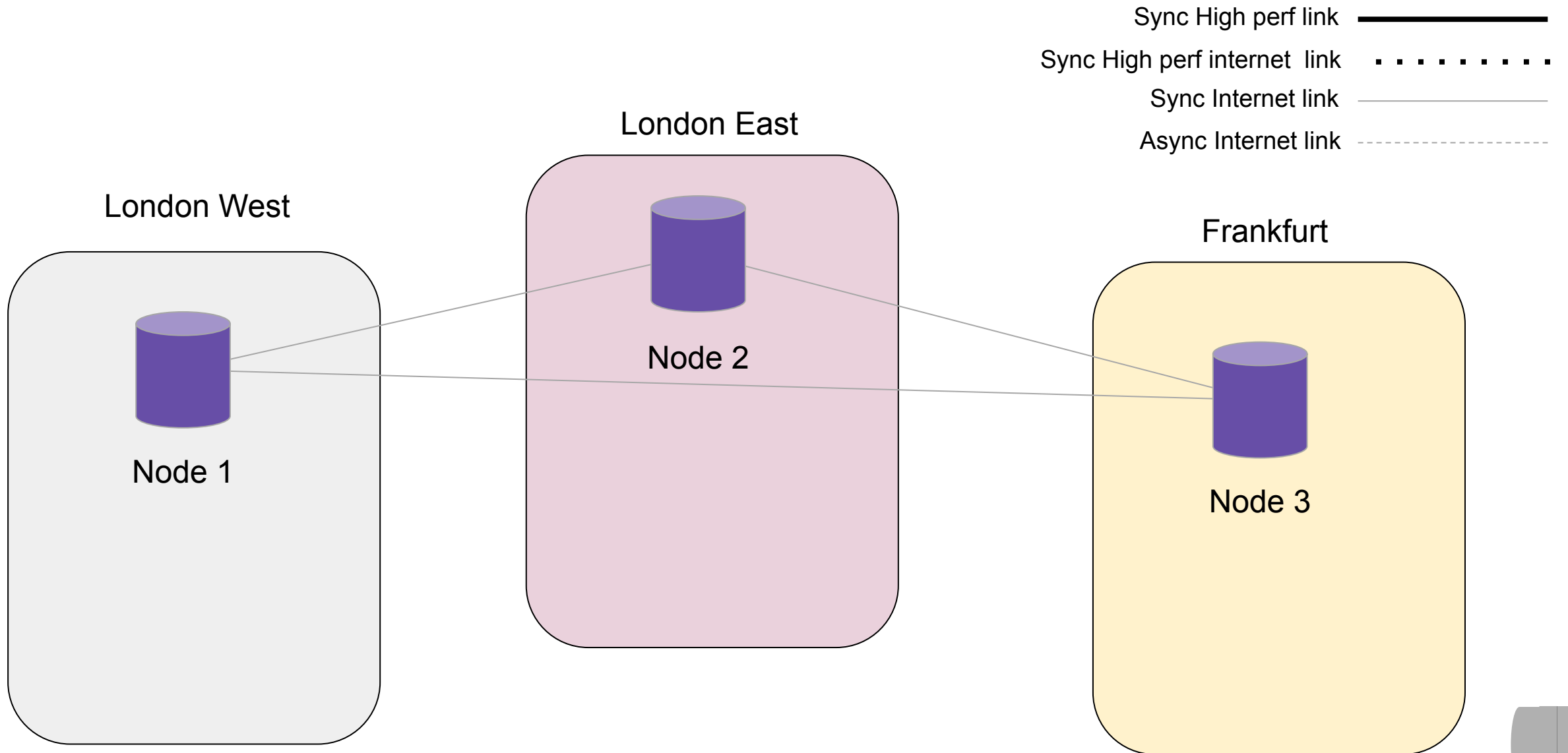
Implementations by HA Level and Tight/Loose Relation



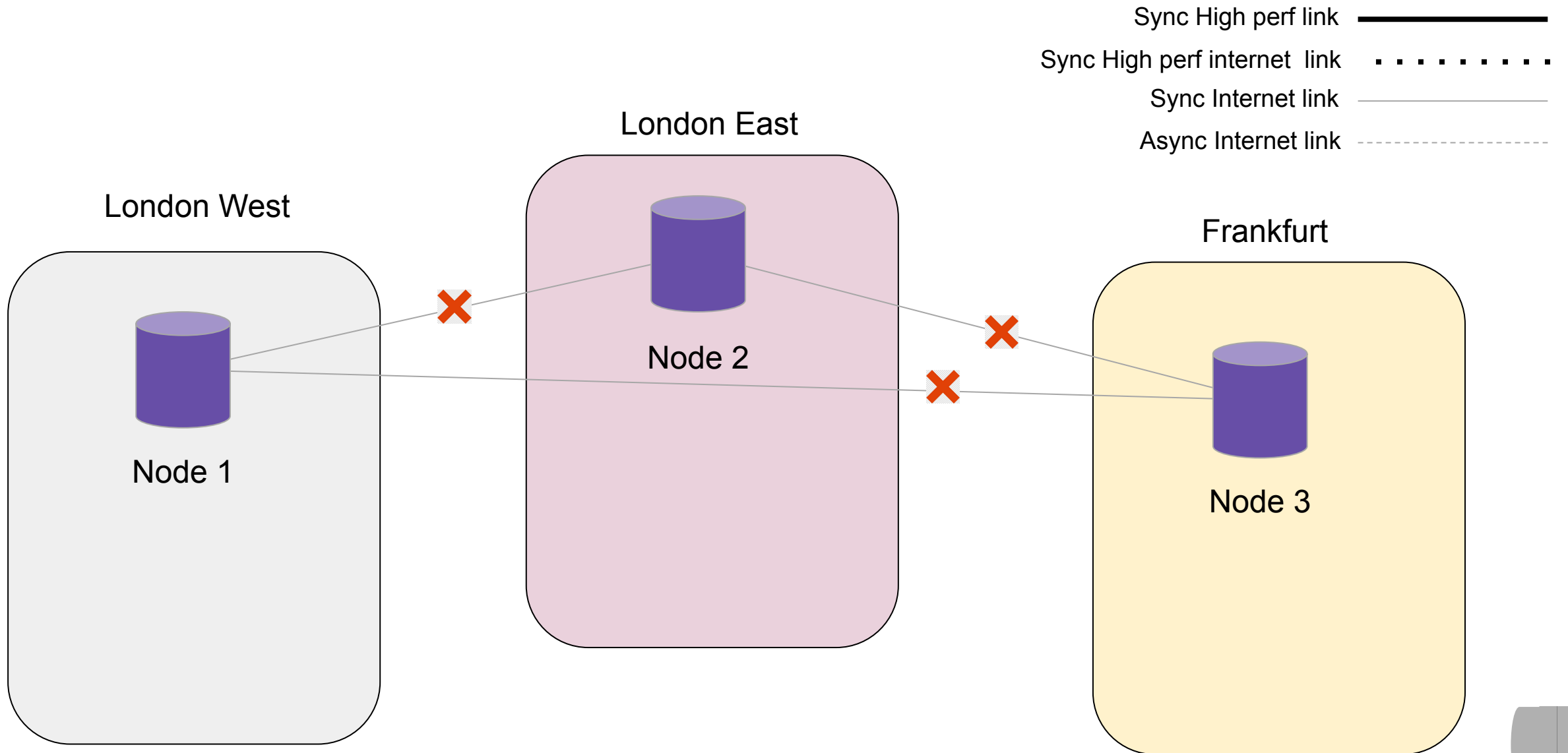
Implementations by HA Level and Tight/Loose Relation



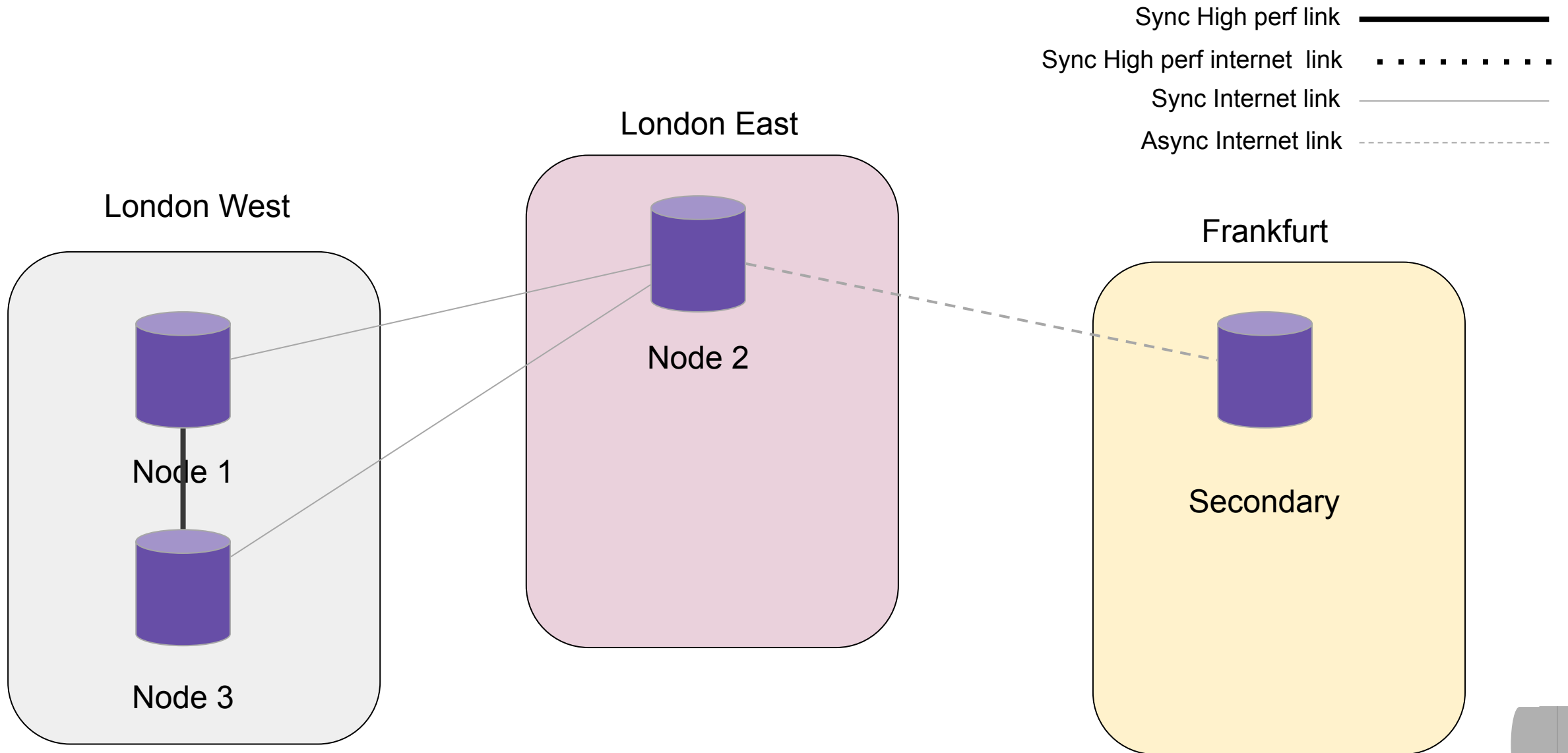
Architectures: What Should NOT Be Done 1



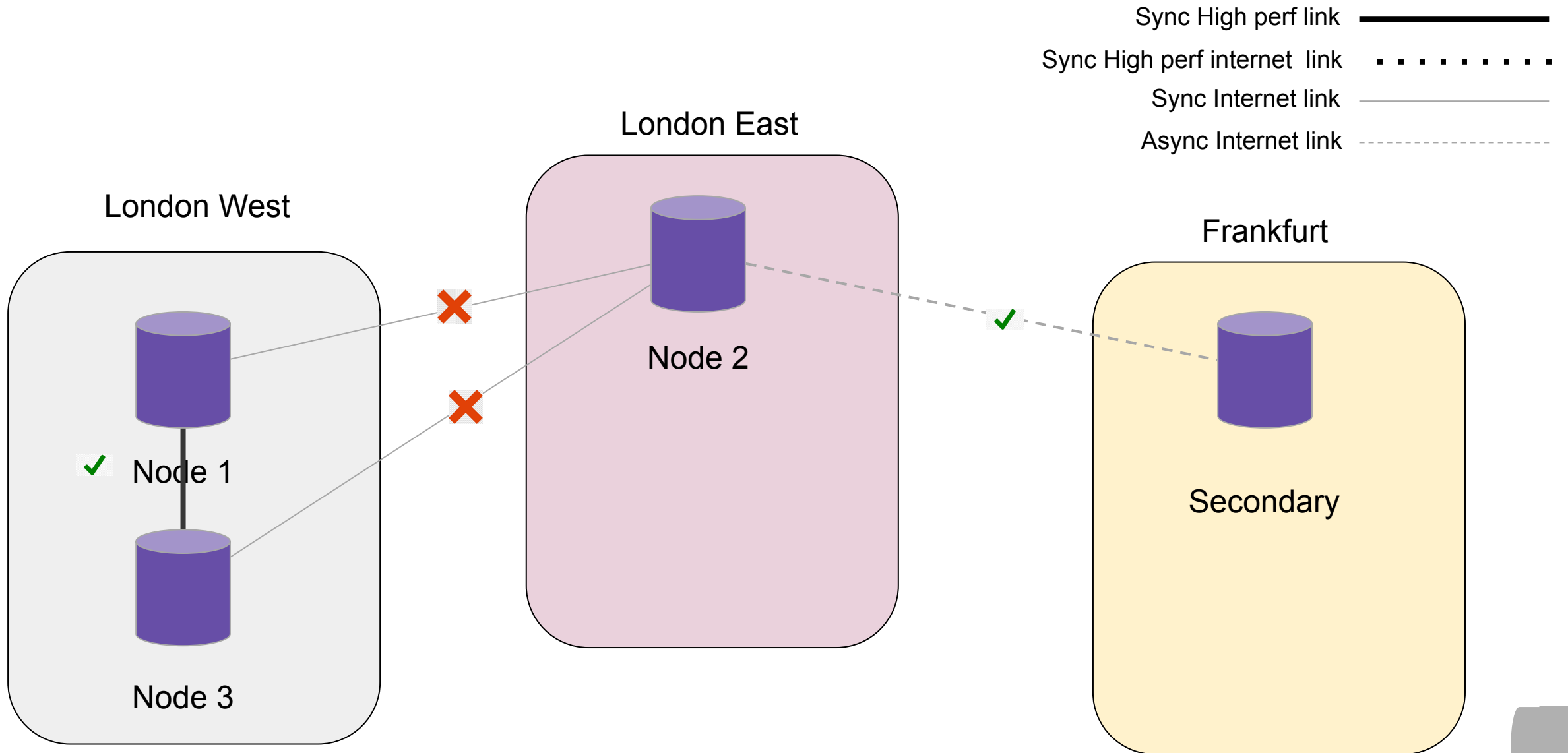
Architectures: What Should NOT Be Done 1



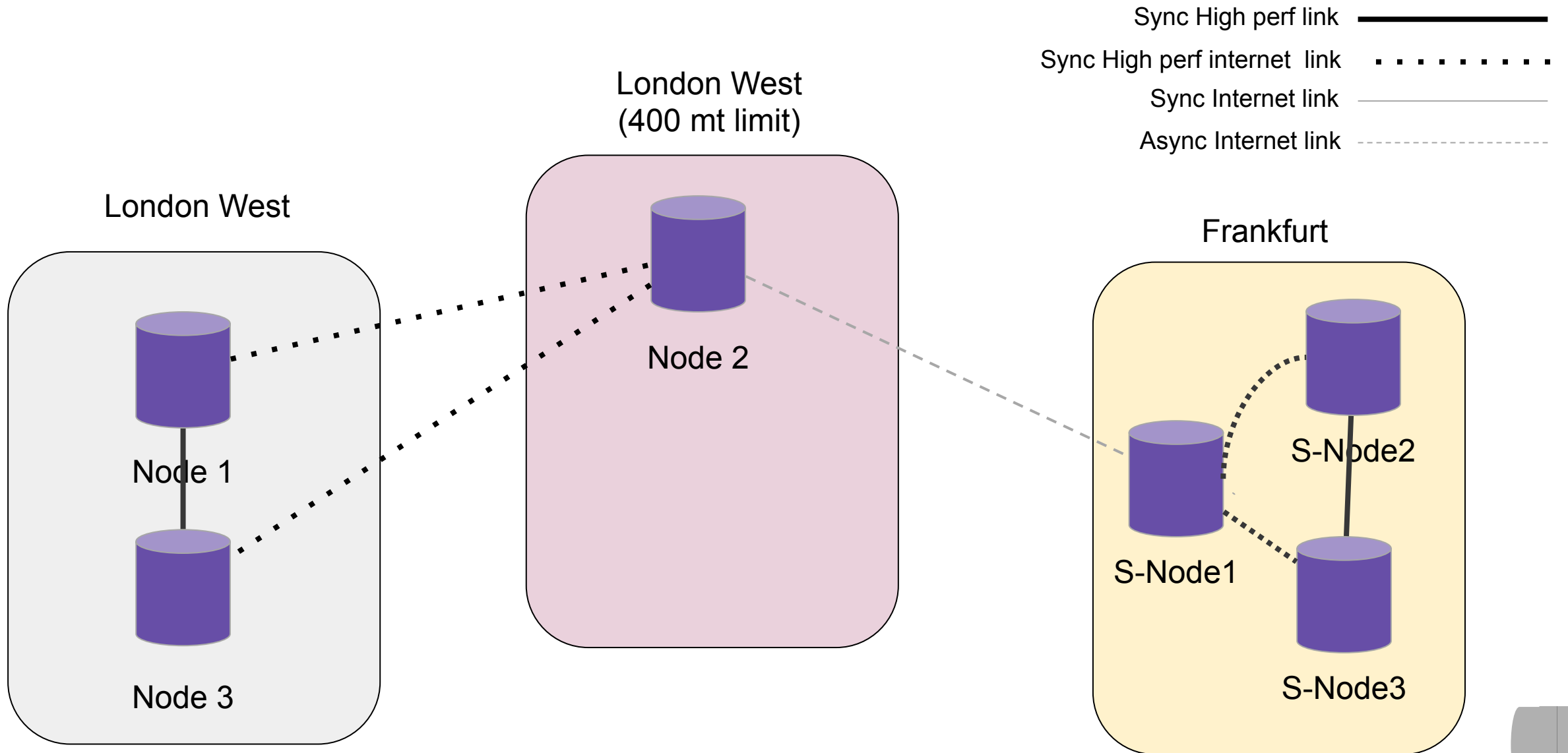
Architectures: What Should NOT Be Done 2



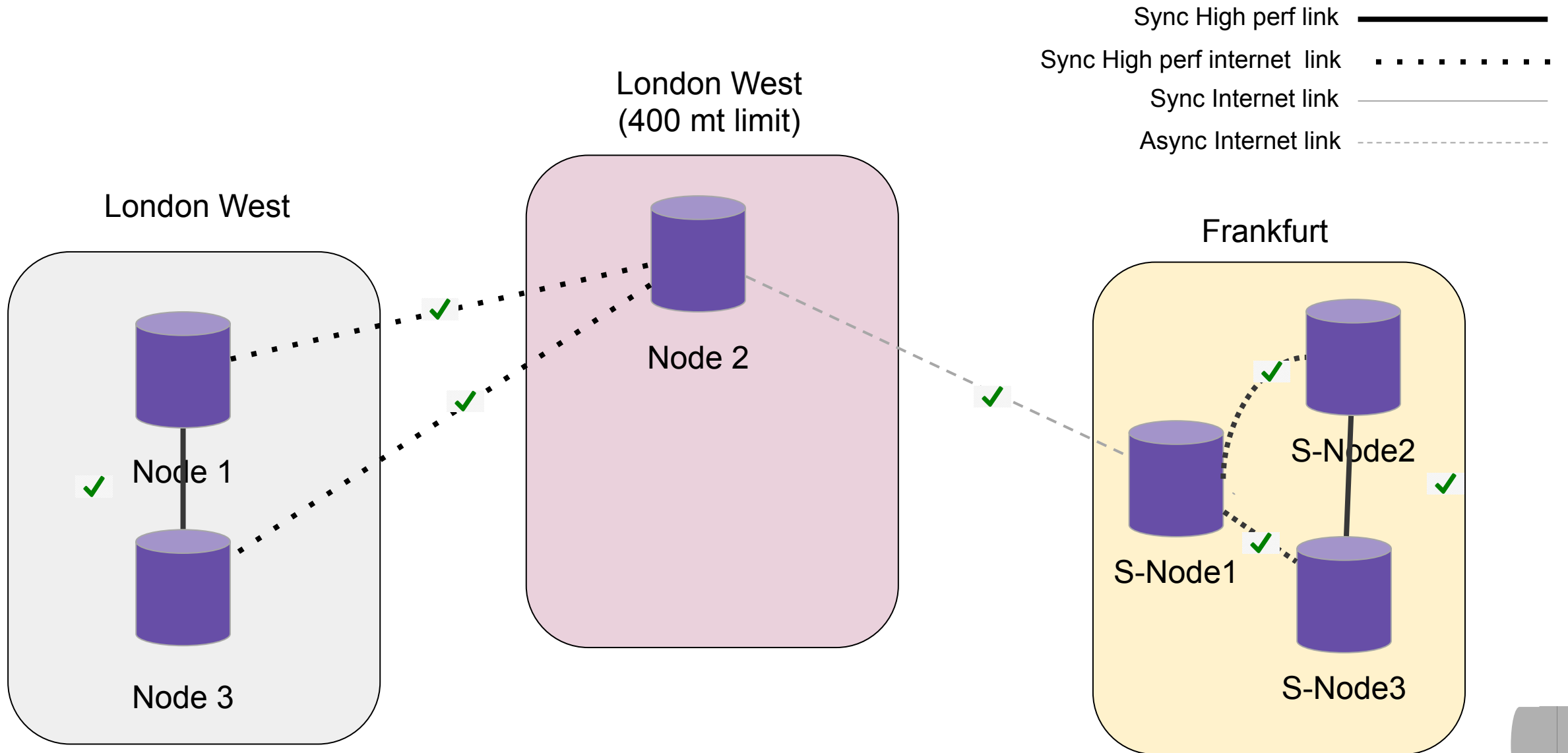
Architectures: What Should NOT Be Done 2



Architectures: What Can Be Done



Architectures: What Can Be Done




Architectures: What You Should Do



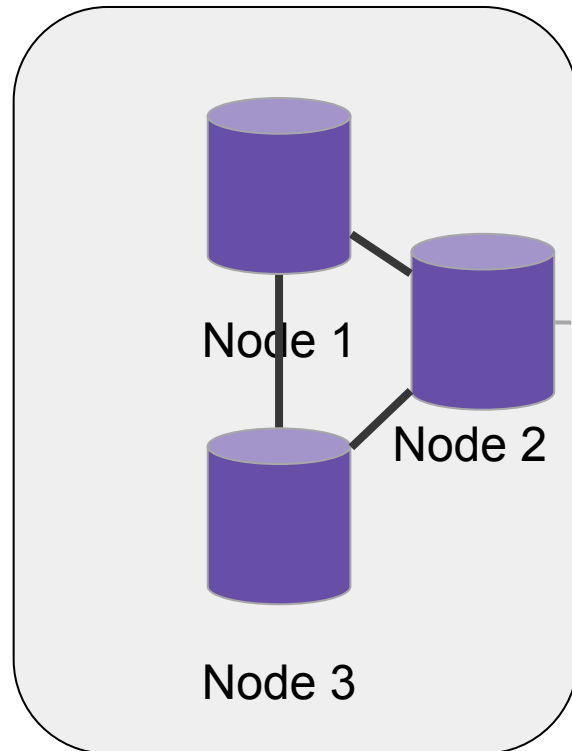
Sync High perf link 

Sync High perf internet link 

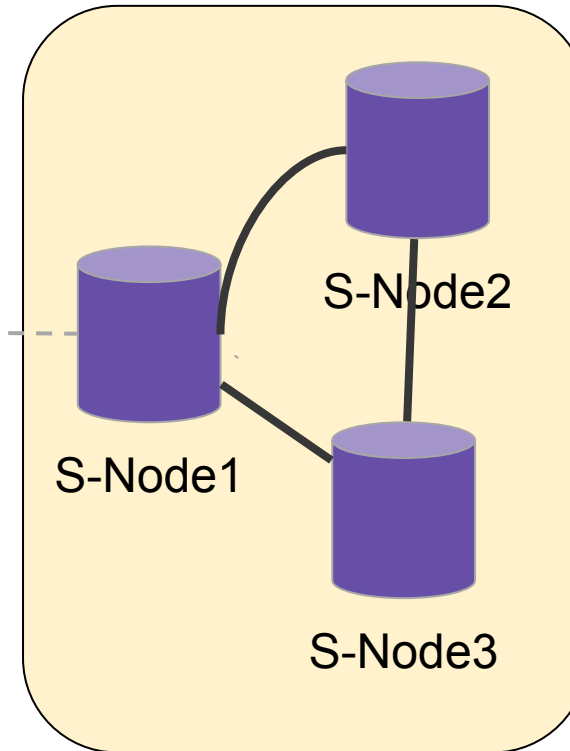
Sync Internet link 

Async Internet link 

London



Frankfurt



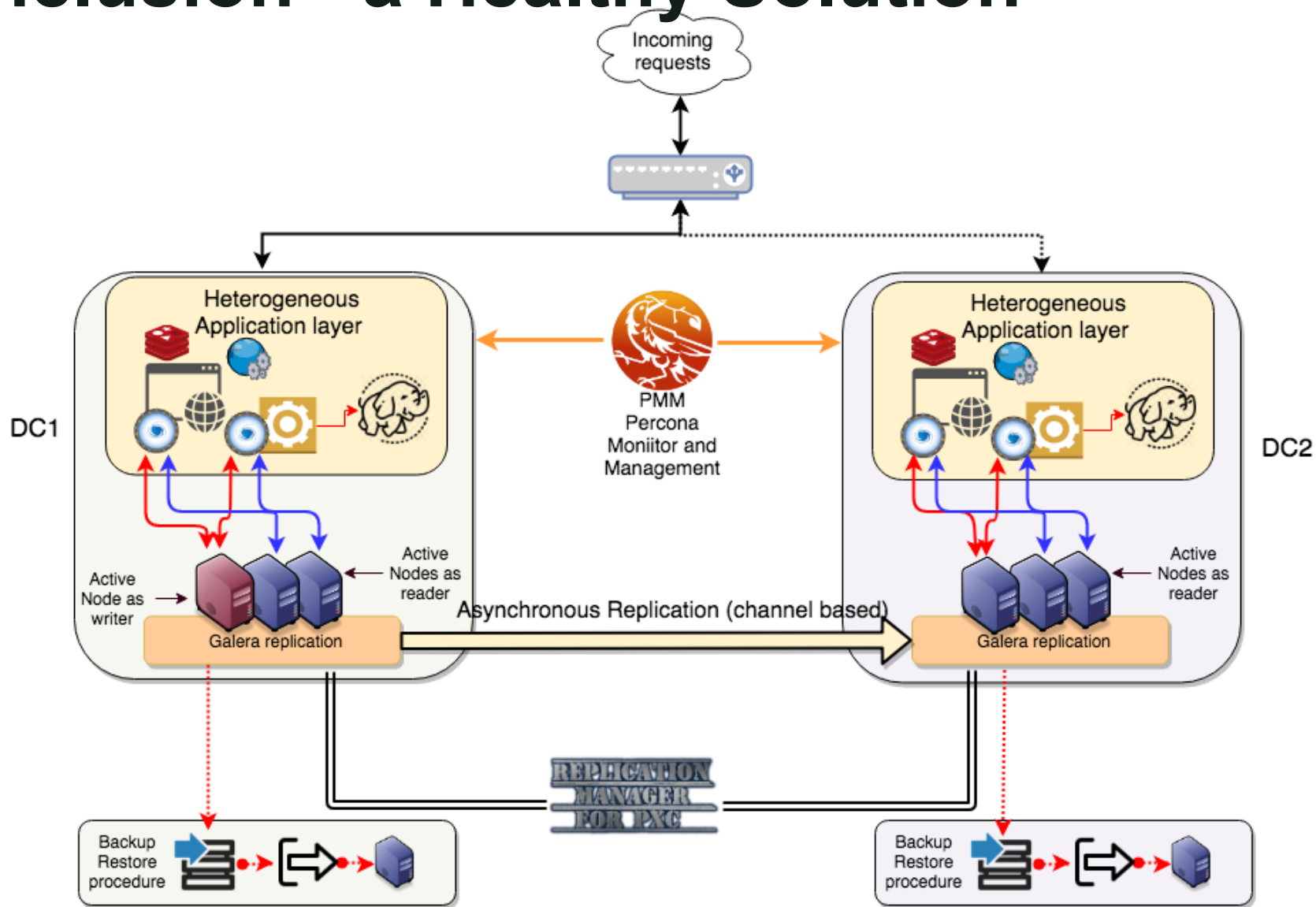
Conclusion - a Healthy Solution

Must have a business continuity plan and cover at least:

- HA
- DR (RTO)
- Backup/restore (RPO)
- Load distribution
- Correct monitoring/alerting



Conclusion - a Healthy Solution*



**I am showing PXC/Galera replication by convenience, but it could also be Percona Server with Group Replication*

Conclusion - a Healthy Solution

Must have a business continuity plan and cover at least:

- HA → (PXC OR PS-GR)
- DR → (PXC OR PS-GR) with Asynchronous replication and RMP (replication manager for PXC/PS-GR)
- Backup/restore → Backup/restore policy
- Load distribution → ProxySQL with Query rules
- Correct monitoring/alerting → Percona Monitoring and Management (PMM)



Useful References

- <https://www.percona.com/blog/2018/11/15/mysql-high-availability-on-premises-a-geographically-distributed-scenario/>
- <https://dev.mysql.com/doc/mysql-ha-scalability/en/ha-overview.html>
- <https://www.percona.com/blog/2014/11/17/typical-misconceptions-on-galera-for-mysql/>
- <http://galeracluster.com/documentation-webpages/limitations.html>
- <http://tusacentral.net/joomla/index.php/mysql-blogs/170-geographic-replication-and-quorum-calculation-in-mysqलगalera.html>
- <http://tusacentral.net/joomla/index.php/mysql-blogs/167-geographic-replication-with-mysql-and-galera.html>
- <http://tusacentral.net/joomla/index.php/mysql-blogs/164-effective-way-to-check-the-network-connection-when-in-need-of-a-geographic-distribution-replication-.html>
- <http://tusacentral.net/joomla/index.php/mysql-blogs/183-proxysql-percona-cluster-galera-integration.html>
- <https://github.com/sysown/proxysql/wiki>
- <https://www.percona.com/blog/2018/11/15/how-not-to-do-mysql-high-availability-geographic-node-distribution-with-galera-based-replication-misuse/>
- <https://github.com/y-trudeau/Mysql-tools/tree/master/PXC>





