Database Replication

Mohammad Reza Khosravian

01

Introduction

What is Replication ?

What is Replication

Replication in computing involves sharing information so as to ensure consistency between redundant resources, such as software or hardware components, to improve reliability, fault-tolerance, or accessibility.

Replication VS Backup

Frequent updates and quickly lose any historical state

Replication

Saves a copy of data and it remains unchanged for a long period of time

Backup

Replication Geographic Strategy

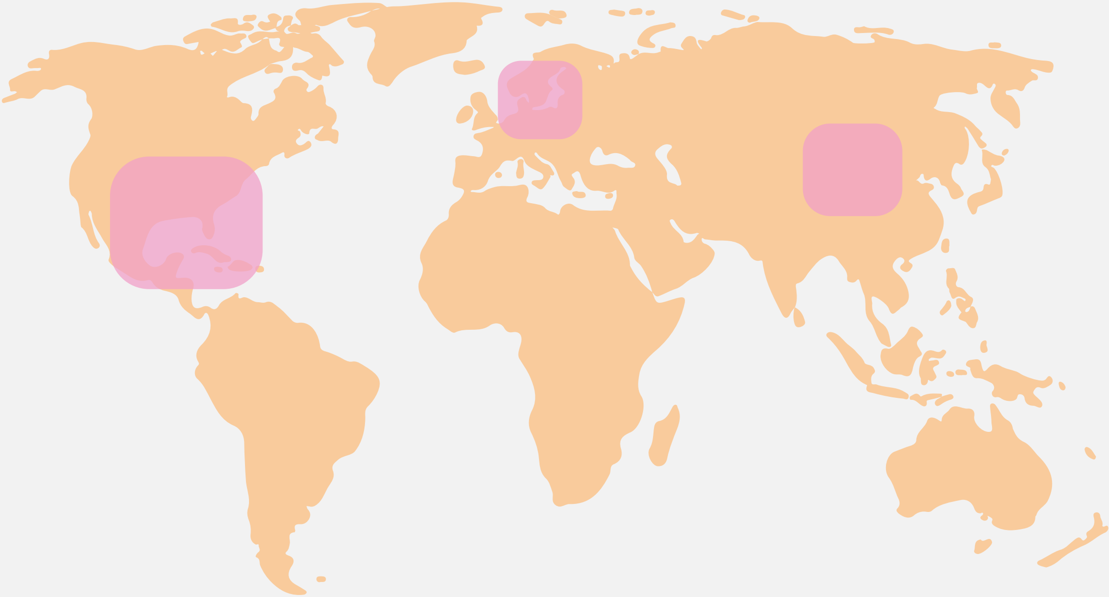
Much more performance Tree-Like Interconnections is the most popular topology within a data center

Within a data center

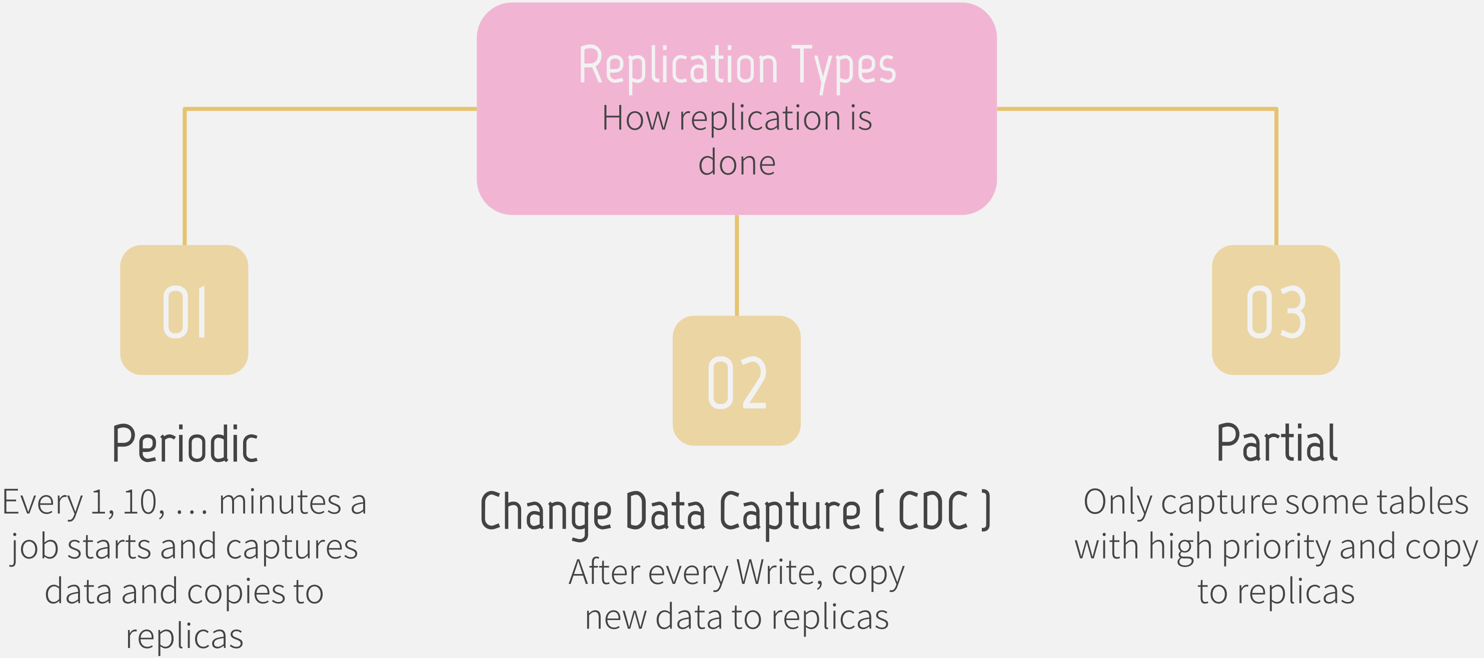
Data centers that are not in one region to survive interruption that something like Storms can occure.

Across data centers

Across Data Centers



Replication Types



Replication Types

How replication is done

01

Periodic

Every 1, 10, … minutes a job starts and captures data and copies to replicas

02

Change Data Capture [CDC]

After every Write, copy new data to replicas

03

Partial

Only capture some tables with high priority and copy to replicas

02

Why

we use database replication ?

Database failures

- Node increasing causes failure possibility
- Node internal errors
- Power supply
- Network switch failures
- Earthquake, Storm, …

Advantages

- Do not lose any Data
- Increase system uptime
- Data consistency
- Decrease delay
- Increase performance (Separate Reads & Writes)
- Fault tolerance

Disadvantages

- Increase complexity
- Increase costs

03

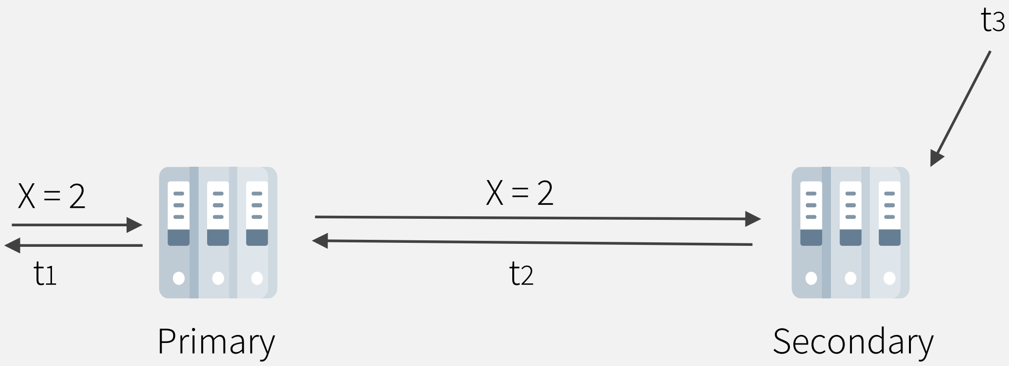
Methods

Which methods are
available for replication ?

Replication Lag (r)

Time that takes to copy data from master to all slaves.

It can make data incosistency.

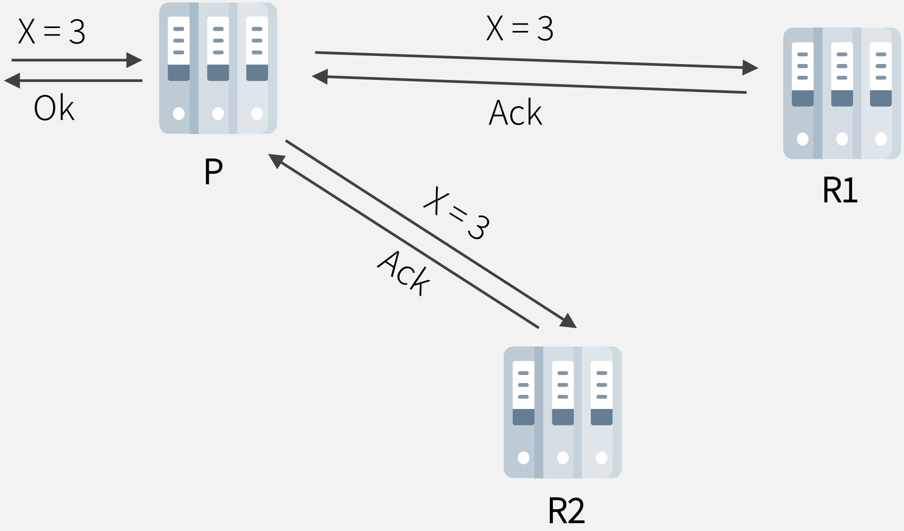


Good: $r < t_3 - t_1$

Bad: $r > t_3 - t_1$

01. Synchronous

After every write request, master (primary) waits for all slaves to acknowledge then finishes the write operation.



Pros

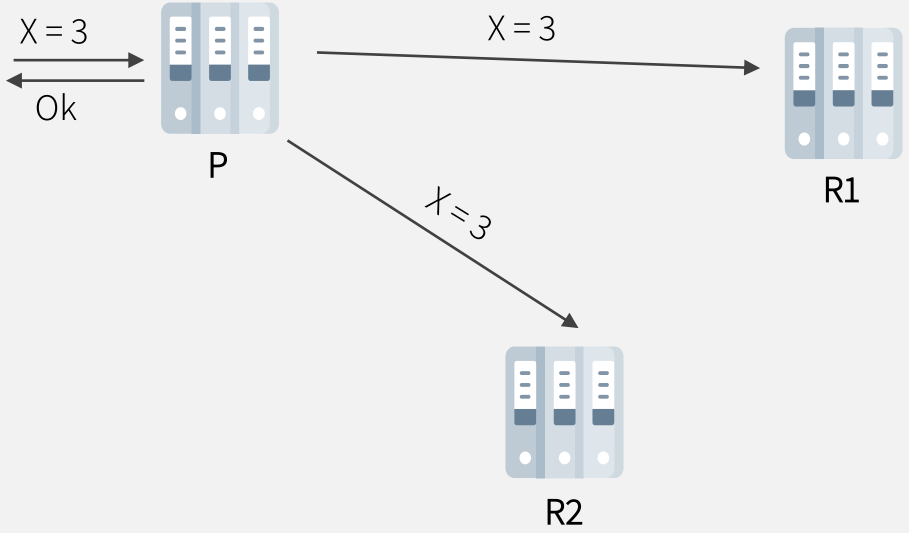
- Replication lag = 0
- Data consistency

Cons

- Low performance
- One node failure causes write fail

02. Asynchronous

Every write is done by master immediately, then sends to replicas (secondaries) but does not wait for replica acks.



Pros

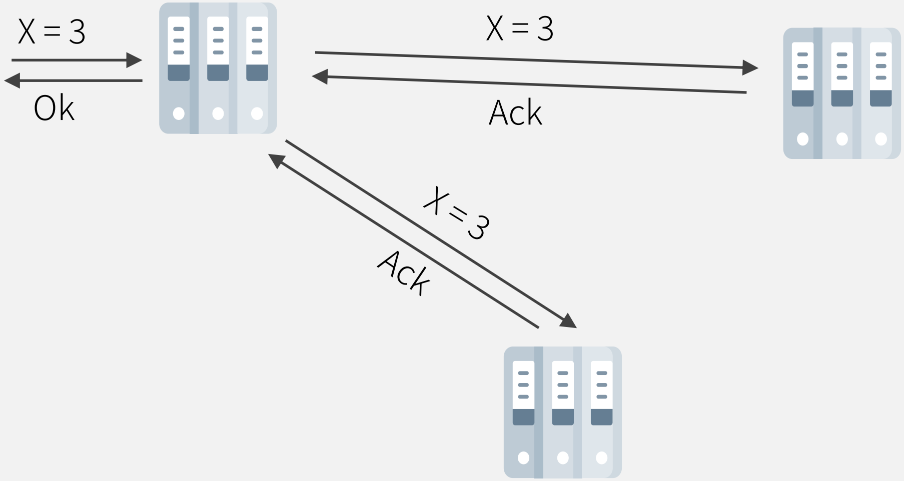
- Fast write operations

Cons

- Complex consistency algorithms

03. Semi Synchronous

Primary just waits for one replica to ack and does the write operation.



Split-Brain

A big problem

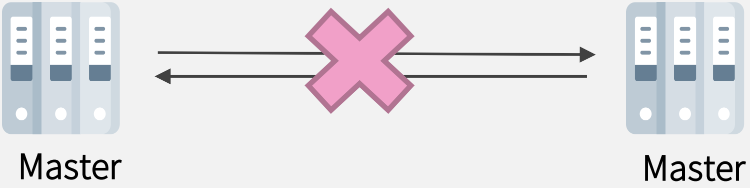
Split-Brain

In distributed systems world, because of network inconsistency, nothing is sure!

Because : Connection Interruption

How does it occur :

When the **router** between this two node gets into trouble, when both master nodes are ok.
In this situation, both nodes thinks that they are the single master and both think they are in sync.

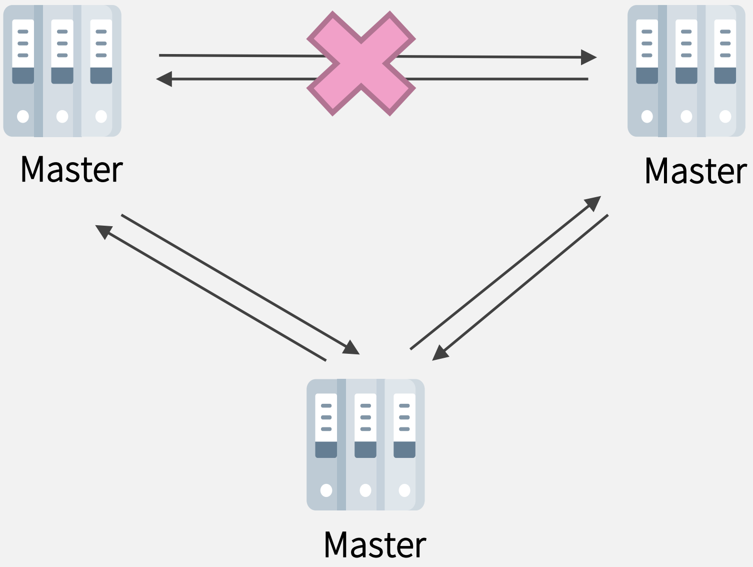


Split-Brain (One solution)

Three Master.

Whenever one router gets into trouble, masters still have a way to communicate and stay synced.

The possibility of failure of two router in this example is much less than earlier example.



RESOURCES

Books

Database System Concepts – 7th edition

Web

https://docs.mongodb.com/manual/replication/

https://www.manageengine.com/device-control/data-replication.html

https://medium.com/@pkostohrys/database-replication-an-overview-f7ade110477#:~:text=Replication%20has%20three%20popular%20algorithms,Leaderless%20replication

https://www.youtube.com/watch?v=RIcNswROzCc