**Experiment:** Out-of-Sync Database Replicas  
**Type:** Network Chaos

## **Introduction**

Network Chaos is a resource in Chaos Mesh, a chaos engineering tool for Kubernetes. Chaos Mesh allows the injection of faults like latency, packet loss, and network partitioning into a Kubernetes cluster to test how applications handle network disruptions.

In this case, the Network Chaos configuration introduces network latency to a specific database pod. This is useful for testing how a system behaves under network delays, particularly in distributed databases where latency can lead to out-of-sync database replicas or replication lag.

## **Purpose: Simulating Out-of-Sync Database Replicas**

By adding network latency to the database pod (**ledger-db-0**), we aim to evaluate how the system handles delayed database replication. This helps in:

* Identifying issues with eventual consistency.
* Observing how microservices react to delayed database updates.
* Understanding the resilience of the system under real-world network fluctuations.

## **Execution**

cd /root/hymavathi/out\_sync\_database\_replicas.yaml

. /script

**Expected Outcome**

Check UI for timeouts or error messages  
Run kubectl logs <UI\_POD> to see if requests are failing

**How to Overcome Out-of-Sync Database Replicas**

To mitigate and recover from network-induced replication issues, consider the following approaches:

**1. Monitor Replication Lag**

* Use monitoring tools like Prometheus, Grafana, or built-in database replication metrics.

## **2. Increase Heartbeat Frequency**

* Shorter intervals help detect replication delays faster.

## **3. Configure Retry Mechanisms**

* Databases like PostgreSQL and MySQL support delayed retries to handle temporary network issues.

## **4. Enable Conflict Resolution Strategies**

* Some databases use **Multi-Version Concurrency Control (MVCC)** to manage conflicting updates efficiently.

## **5. Network Optimization**

* Implement **Quality of Service (QoS)** policies to prioritize database traffic and reduce latency