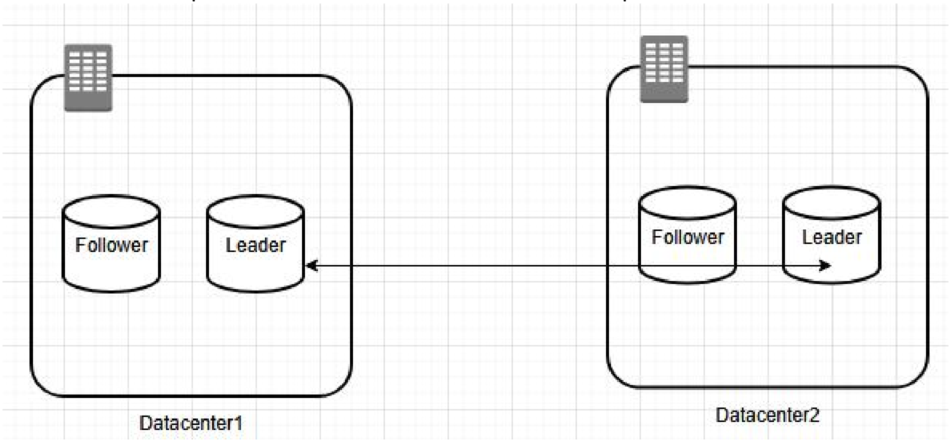
Replication4

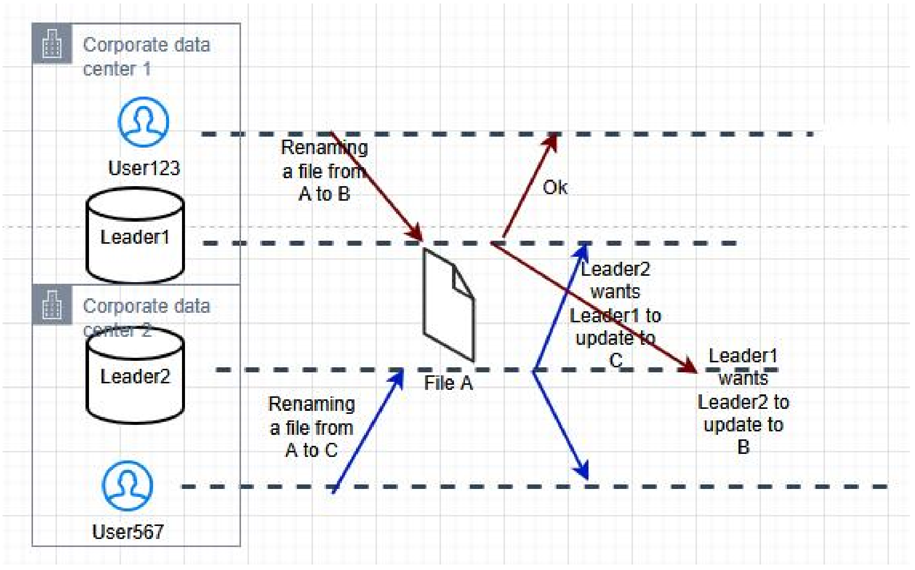
Multi-leader replication

We have multiple datacenters and each datacenter has a Leader and Follower. Now the two Leaders need to co-ordinate and sync up.
All Writes will be performed on Leader and Reads will be performed on Follower



1. Performance will be improved drastically due to multiple Leaders
2. Tolerance goes up
3. Offline operation can be performed easily
4. Collaborative editing

Conflict in Leaders:
Lets say User123 is renaming File A to B on Leader 1. At the same time, User567 is renaming the same File A to C. Leader1 will ask Leader2 to update to B while Leader2 asks to update to C. So we have a conflict.



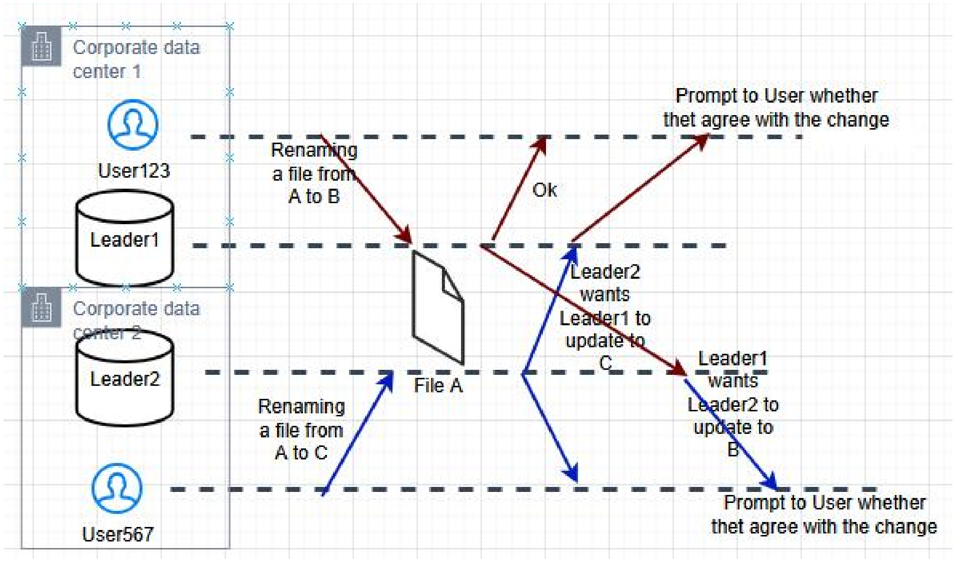
Corporate data center 1
User123
Leader1
Corporate data center 2
Leader2
User567
Renaming a file from A to B
Renaming a file from A to C
File A
Ok
Leader2 wants Leader1 to update to C
Leader1 wants Leader2 to update to B

How to resolve conflicts? Different strategies

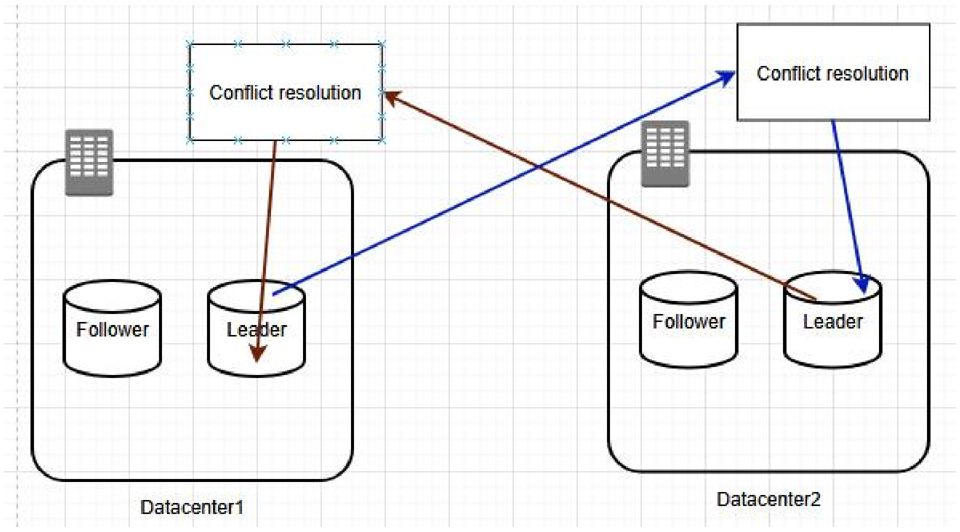
1. Last Write Wing (LWW):
- Picking the Write from higher timestamp
Drawback is there is a chance of Data loss. The updates done on the particular file will all be vanished or replaced by the latest update. Latest update could even override the previous updates.

2. Replica with Higher ID is picked:
For each Leader, we will assign a unique ID to them
We will only pick data from the replicas where ID is the largest
- Assign IDs to Leader transaction
- Pick data from highest ID
- Possibility of data loss

3. Record the conflict and Prompt User to resolve:
In Git, if a Write operation is performed on the same line, there will be PR conflict and ultimately the User has to resolve it. Before Leader2 overwrites updates from Leader1 automatically, it will prompt the user.


Corporate data center 1
User123
Leader1
Corporate data center 2
Leader2
User567
Renaming a file from A to B
Renaming a file from A to C
File A
Ok
Leader2 wants Leader1 to update to C
Leader1 wants Leader2 to update to B
Prompt to User whether thet agree with the change
Prompt to User whether thet agree with the change

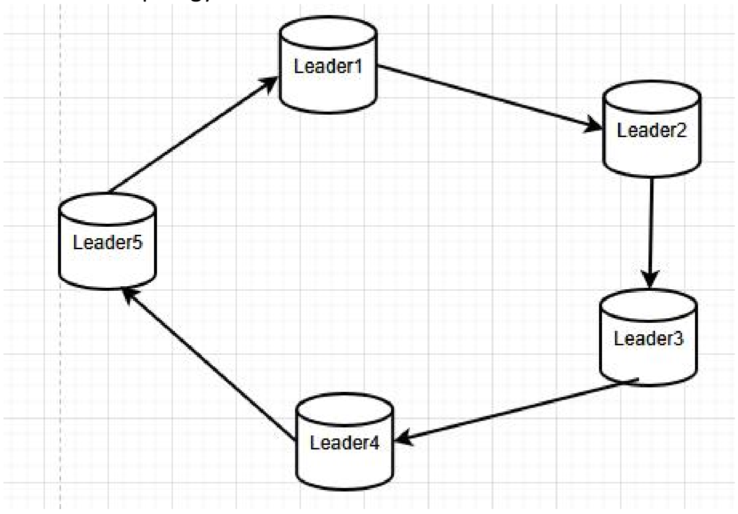
- we can get prompts from Application code
- can merge A, can merge B, or can have both the changes. We have a choice



Any changes first go to Conflict resolution from there we can resolve and update Leaders

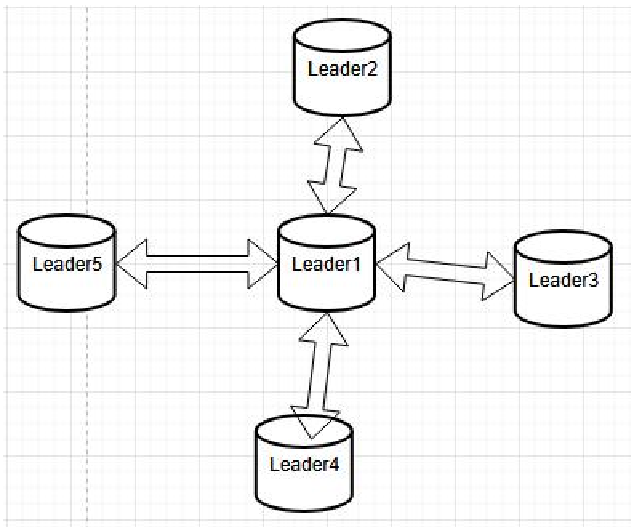
Topologies:
We can have different topologies, which means how different Leaders will connect to or talk to each other. How do we connect different Leaders?

1. Circular topology:



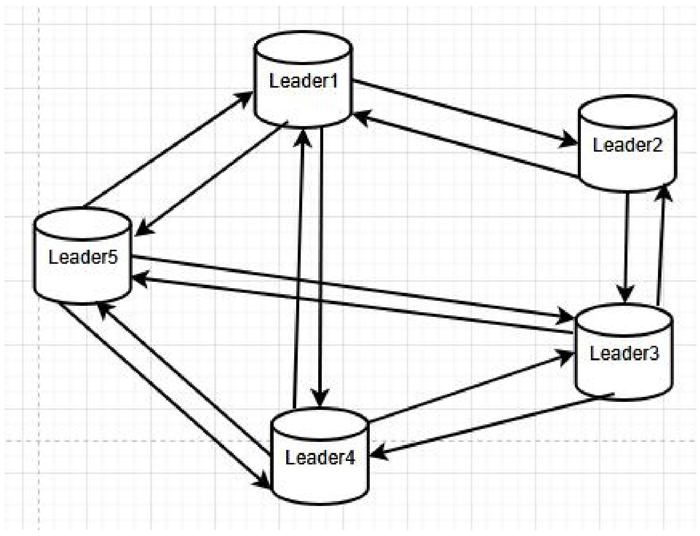
It has to go through a cycle to update data. There will be Latency, and conflict resolution will be complex. Also there will be scability issues.
- One replica connected to one/two replicas
- Slower in terms of updation so high latency
- If one node fails, we will have to reassign everything, lot of work
- In case of failure of even one node, it will disrupt the flow
- if the failed node has the most recent data, that will also be gone

2. Star topology:



- Common Node pathway
- Center node
- if Center node fails, it is a big issue, we will have to appoint a new node as Center node, will lead to High latency

3. All-To-All topology:
This is the most popular topology



Every node is directly connect to every other node. All nodes are directly connected.
- since every node is connected to every other, Latency will be low
- If Leader2 stops working completely
- Data loss is also reduced
- The structure of the system will be complicated
- Complex to handle and configure