**18CS621: Distributed Systems**

**Lab Assignment: Coordination**

**Date :- 29/11/2020**

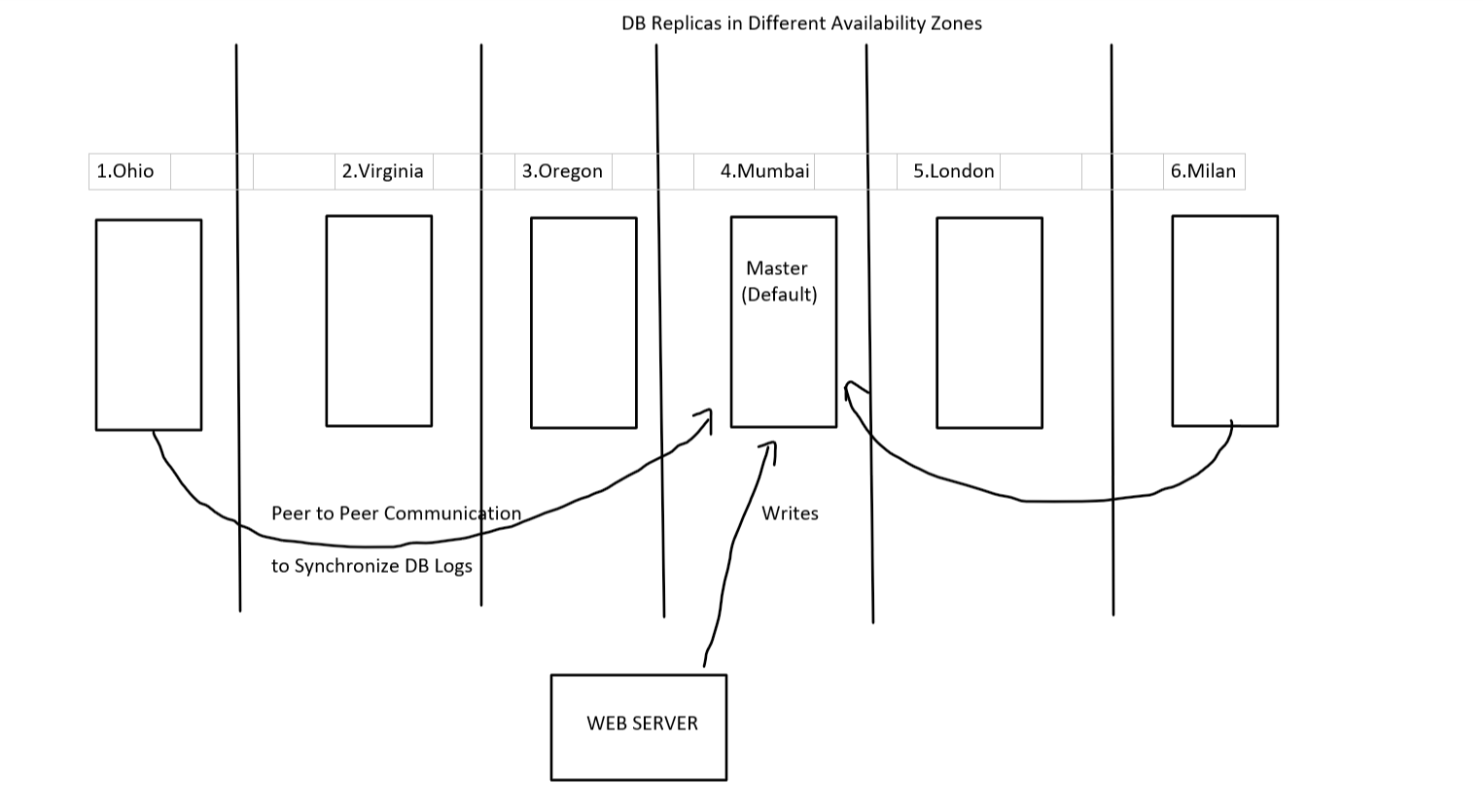
**Individual Contributions to Assignment**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Name** | **Contributions** |
| 20010 | C H Vamshi | Mutual Exclusion (Token Ring) |
| 20017 | K Koushik | Leader Election (Ring Election) |
| 20031 | 1. Aathith Bhalaje | Vector Clocks Synchronization |

**4). Leader Election Algorithm :-**

**How does The Ring algorithm work :-**

1. If process P1 detects a coordinator failure, it creates new active list which is empty initially. It sends election message to its neighbor on right and adds number 1 to its active list.
2. If process P2 receives message elect from processes on left, it responds in 3 ways:

3. If message received does not contain 1 in active list then P1 adds 2 to its active list and forwards the message.  
  
 4. If this is the first election message it has received or sent, P1 creates new active list with numbers 1 and 2. It then sends election message 1 followed by 2.  
  
   
  
(III) If Process P1 receives its own election message 1 then active list for P1 now contains numbers of all the active processes in the system. Now Process P1 detects highest priority number from list and elects it as the new coordinator.  
  
   
  
   
  
**Diagram :-**  
  
  
  
  
**Assuming the situation with AWS Aurora DB: -**  
  
There will be 6 Data Copies each in different zones and there will be 6 availability zones (Data centers).  
  
If there is any request like (Write into Database from the Server), the request is sent to the Leader (Coordinator) which has the higher priority. In our case the Leader is MUMBAI (assuming).  
  
The Leader will reflect that transaction to all those remaining database replicas by peer-to-peer communication   
  
In usual cases: -   
The Server will be responsible for sending write request to all the DB’s.

But if we do that, there will great delay.

So, Amazon has optimized its operation in Aurora DB as;

* + - One writes request is generated by web server which is sent to leader.
    - Later on, P2P communication takes place between leader and other DB Replicas
    - So, whenever a DB Replica wants to update its logs it communicates with the leader (which is in Mumbai in our case) where the first transaction is performed
    - So, after every transaction of P2P communication takes place, the leader will give transaction logs to other DB instances which requested leader
    - Actual write operations, will be done in leader. Later, the REPLICAS communicate with the leader to get the write logs

1). Everyone communicates with the leader but what if a flood or disaster occurs in the leader zone and the DB got crashed (This zone will be DEAD).

2). Now, among these a leader must be elected based upon the ID’s given to them (i.e., maintained by AWS) the leader will be finally selected.

3). After, the leader Election Algorithm one will become leader & then the write operation will be sent to the new leader.  
  
 4). So, when there are new write logs, they are sent to new leader

5). And, all those remaining DB replicas will communicate to the newly elected Leader.

**Instructions to Run Code: -**

* **Files Structure: -**

-Co-Ordination Assignment

|-Problem1(Vector Clock)

|- vectorClock.go

|- VectorClock.docx (Documentation)

|-Problem3(Mutual Exclusion)

|-tokenRingMuTex.go

|-Token Ring Mutual Exclusion.docx (Documentation)

|-Problem4(Leader Election)

|-ringLeaderElection.go

|-Election Leader.docx (Documentation)

* **How to give Inputs: -**
  + It is a menu driven Application
  + First it shows Different Replicas which are situated in each Availability zone
  + The User is provided Four options
    - 1.Process a P2P communication (to exchange logs with leader)
    - 2.Fail the DB Replica
    - 3.Display the Leader ()
    - 4.Exit Application
  + Upon selecting 1st option, you are prompted to enter a availability zone where you wanted to initiate transaction (communicates with leader)
  + If the Leader is Active- Our Request is fulfilled
  + If we notice that Leader is failed, we start leader Election algorithm(Ring Election) and select new Leader
  + Upon selecting 2nd Option it prompts us to enter the zone where the Replica has failed
  + The 3rd Option displays the current Leader
  + 4th Option Exits the Application

**Note : -**

Fail DB Replica – It cannot fail upon itself, so we are creating a situation to fail it.

**Output Screenshots** :-

