Very Large, Distributed Data Volumes (TDT4225)

Course Coordinator : Svein Erik Bratsberg

Assignment 4

Submitted by
Name : Md Anwarul Hasan
Student ID : 583233

1. Kleppmann Chap 5

a) When should you use multi-leader replication, and why should you use it in these cases?

When is leader-based replication better to use?

Answer:

It is better to use multi leader replication when,

i) When there is multi data centre operation

When there is a database that has several replicas, if it uses single leader based replication then one of the datacentre should be the leader and other would go through that datacentre to write. But in multi leader approach, each of the datacentre individually operates in leader-follower approach and also replicates it's changes to other datacentre. Advantages are,

a) Performance: As in single leader approach, every follower need to go through the leaders datacentre, user may face the latency for update. Usually multi datacentre are build to support faster operation, but in such cases (single leader) the
b) Fail tolerance: in single leader approach when the leader fails, a follower in other data center become leader. But in multi leader approach each and every data center are independent and during fail in one leader cause no disruption or change in other leaders. Whenever the failed datacenter is online the replication take place in it
c) Data traverse between data center go though public net and single leader may face problem during writing as it writes synchronously. On the other hand multi leader datacenter writes asynchronously and thus may face less problem.

ii) When there is clients with offline operations

When the user has something to do with the data during offline, and when comes back online replicates the changes made during the user's device was in offline. In this case the devices have local databases.

Leader based replication is better to use when all the replication need ensure that data is properly copied in all the instances of all database. One data center atc as leader and other data center act as follower. Data write always initially occurs in leader and followers reads from the leader or follower in read only mode.

b) Why should you use log shipping as a replication means instead of replicating the SQL statements?

Answer:

Write the log ahead means writing the transaction log before writing the other database files. So this is usually very low-level language and reconstruction can be done properly. IT ensures good backup for every entry and can be used to restore the data. Log shipping ensures shipping of log, and those logs will help to recover data quickly from very large tables also.

2. Kleppmann Chap 6

a) What is the best way of supporting re-partitioning? And why is this the best way? (According to Kleppmann).

Answer:

b) Explain when you should use local indexing, and when you should use global indexing?

Answer:

It is beneficial to use local indexing when the number of partitions in a database are small. In such a case the local index can be easily arranged and also searched. Suppose, in a database of car if the partition is less then the local index can be useful to search a “red” car. And although it will require to search the car in all the partitions but there will be small number of partitions and search will not be difficult.

In a global index, the indexed value from all the partitions can be found in one place. But it can create data bottleneck and hamper the purpose of partitioning. So the global indexing also needs to be partitioned, but not the way like the way of database partitioned but in a special way. For example, in a global indexing of car colours, cars of colour names starting by letters between a to m in a partition and cars of colour names starting by letters between n to z in a different partition. When it is possible to arrange the data in this way then the global indexing is beneficial.

3. Kleppmann Chap 7

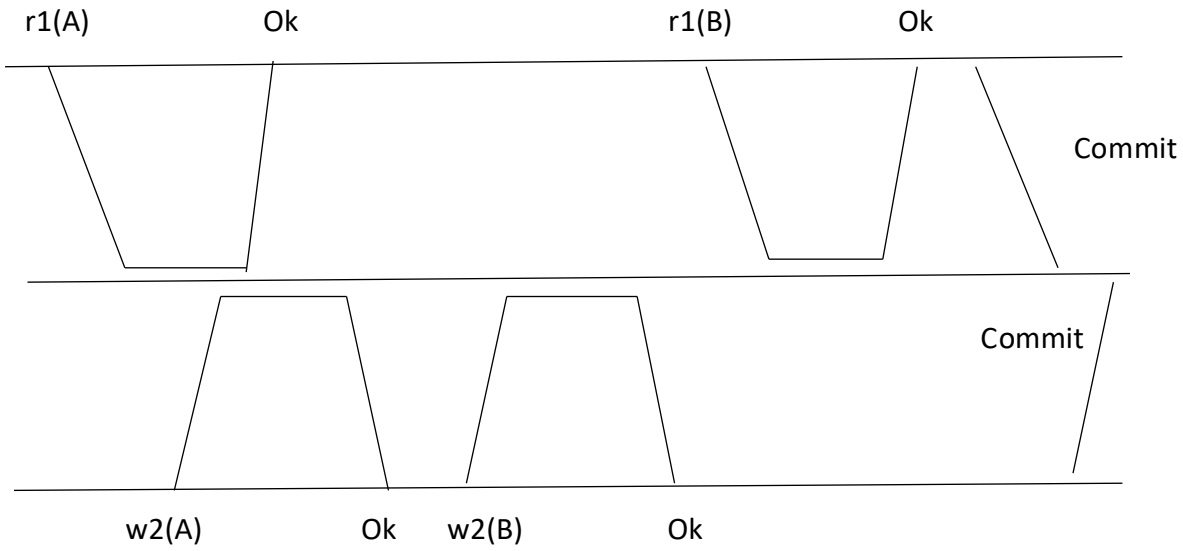
a) Read committed vs snapshot isolation. We want to compare read committed with snapshot isolation. We assume the traditional way of implementing read committed, where write locks

are held to the end of the transaction, while read locks are set and released when doing the read itself. Show how the following schedule is executed using these two approaches:

r1(A); w2(A); w2(B); r1(B);c1;c2;

Answer:

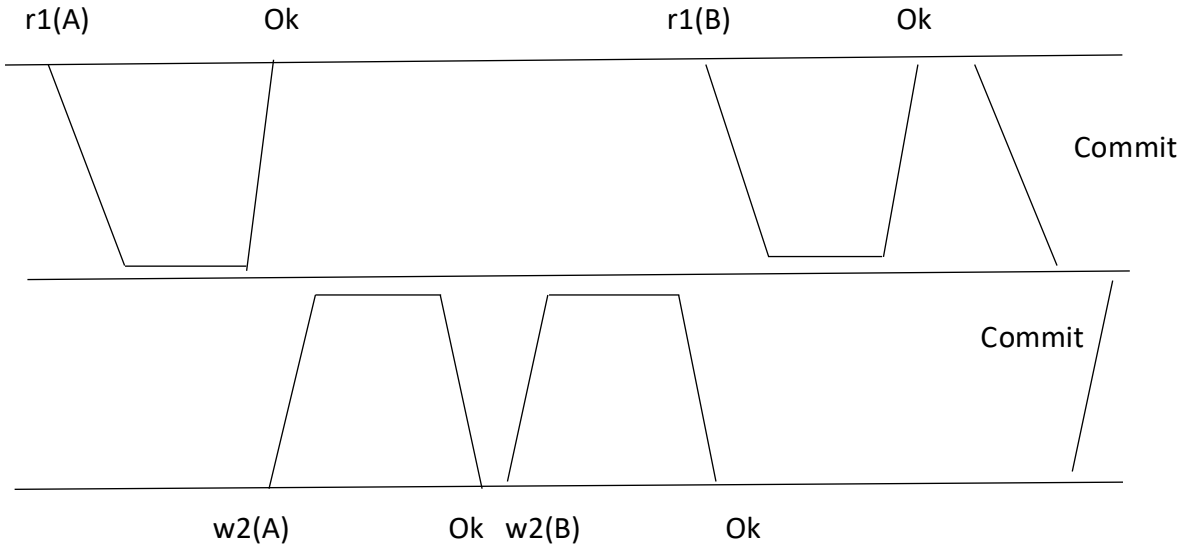
Read Comitted –



b) Also show how this is executed using serializable with 2PL (two-phase locking).

Answer:

2PL (two-phase locking) –



4. Kleppmann Chap 8

a) If you send a message in a network and you do not get a reply, what could have happened? List some alternatives.

Answer:

If a message was send in a network but there is no reply of that, then there may be several reasons for that. Som of them are given below,

1. The request did not reach the destination or it is lost
2. The request is waiting in a queue to be delivered
3. The destination node is already in failed state
4. The destination node stopped sending response for some time
5. The destination node has received and processed the request but the response itself lost
6. The destination node has received and processed the request but the response is delaying

Some of the alternatives are,

1. Detecting the faults by –
    a. Employing a load balancer that will stop sending requests to a node that is dead
    b. Promoting a follower into leader in a single leader distributed database where leader fails
2. Through observing some feedback, like –
    a. If a server process cashes then the OS will close or refuse TCP connection by sending FIN or RST reply
    b. By a script, when the node process crashes but node OS active, then the script will inform other nodes about the crash
3. Timeouts

b) Explain why and how using clocks for last write win (LWW) could be dangerous.

Answer:

There are several times when using clocks for last write wins could be dangerous. Those are given below,

1. If a node has lagging clock, it will always fail to overwrite values written by a previous node that has faster clock. In this scenario, there will be no error reported, but may cause huge amount of data to be loss.
2. LWW can not distinguish writes that happened sequentially but in quick succession and when the later writes has lagging clock. Even though a later write try to wite, but LWW detects it as not latest write and ignores wite for lagging in clock.

3. LWW may let two node to write independently with same time stamp. This may occur even when the clock has millisecond resolution.

5. Kleppmann Chap 9

a) Explain the connection between ordering, linearizability and consensus.

Answer:

The connection between the ordering, linearizability and consensus are quite simple and straightforward. For example whenever an operation happens in a proper order then it creates a situation where we can say that in a situation where there is several replica of database, in a single point in time there is only one single copy of data present. From the wherever replica the data is being read, it will generate same result. This requires some consensus between the replicas of database. Otherwise the total concept would not stand and may create confusion to its users.

b) Are there any distributed data systems which are usable even if they are not linearizable? Explain your answer.

Answer:

Yes there is a data system that is usable even though it is not linearizable. That is multi-leader replication distributed database. Here the basic construction or the formation of multi-leader replication guarantees that each and every node will have the same value in a certain instance. In a multi-leader setup, there is a leader in each datacenter and every write is processed in local datacenter and it is replicated asynchronously to the other datacenters. For this reason the network delay is not present to the users and there will be a better performance. This is why this is not linearizable but this is usable.

6. Coulouris Chap 14

a) Given two events e and f. Assume that the logical (Lamport) clock values L are such that L (e) < L (f). Can we then deduce that e "happened before" f? Why? What happens if one uses vector clocks instead? Explain.

Answer:

For the events e and f the logical clock vales are provided L(e) and L(f).

If we denote “happened before” or “happened earlier” by “→” then we can say that L (e) < L (f) indicated that logical clock value L(f) is greater than logical clock vale L(e). but this do not implies that e →f.

It may happen or may not happen as well, depending on several assumptions. For example, if e and f are in the same process then we can say that. Again, if through series of events we

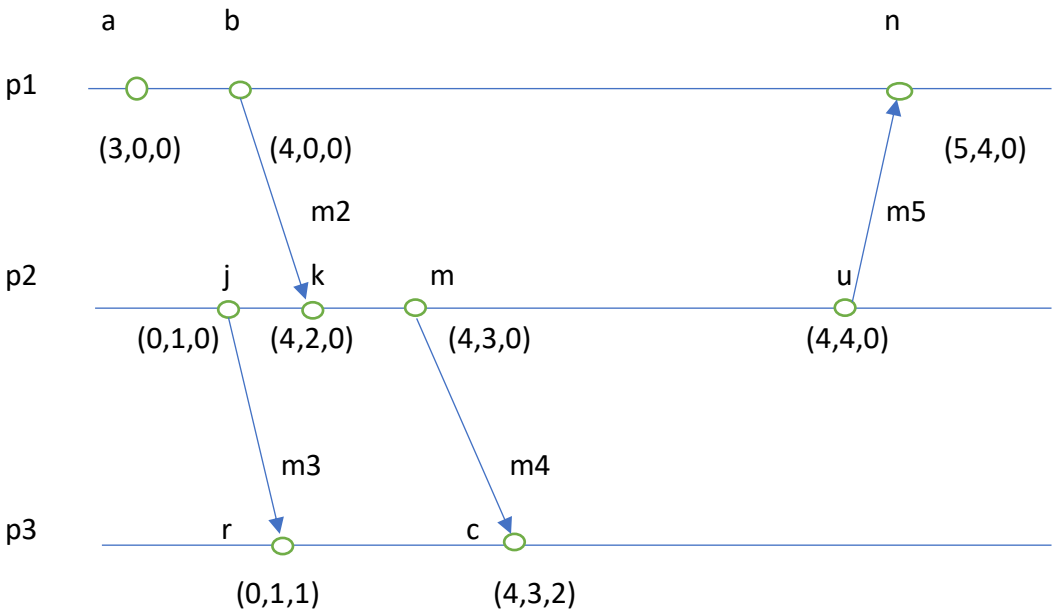
can deduce that e is earlier than f then we can say that. Like, if we know that e→a, a→c and c→f. then we can say that e→f, else we can not say that.

If, there was vector clock present and would denote event e and f like V(e) and V(f) and mention that V(e)<V(f), then we could say that e→f, because the relation V(e)<V(f) is true only and if e <= f and e (not) f. From this relation we can say that the using vector clock and mentioned L (e) < L (f) indicates that e→f.

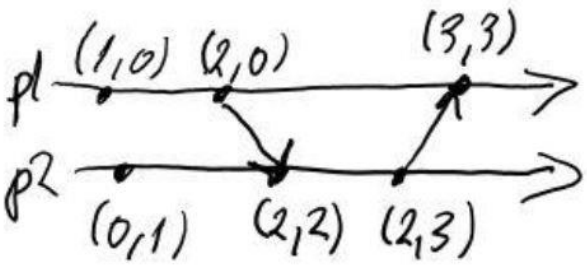
b) The figure below shows three processes and several events. Vector clock values are given for some of the events. Give the vector clock values for the remaining events.

Answer:

The vector clock values for the remaining evens are given below,

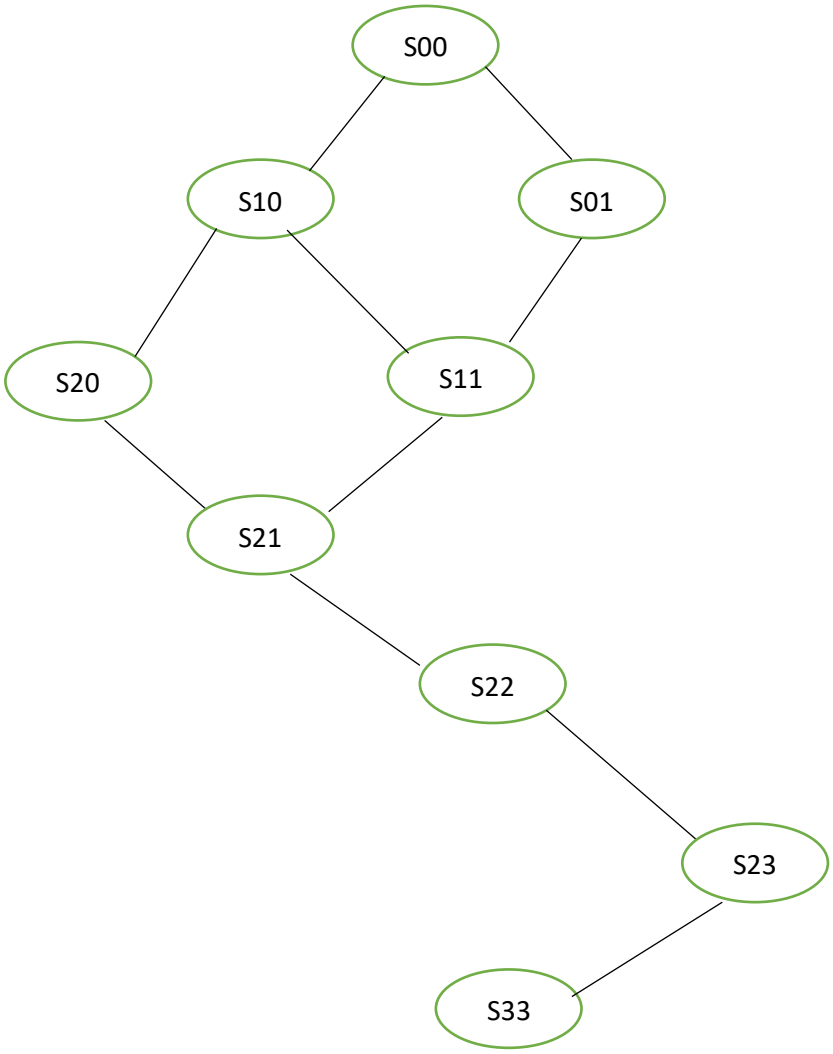


d) The figure below shows the events that occur at two processes P1 and P2. The arrows mean sending of messages. Show the alternative consistent states the system can have had. Start from state S00. (Sxy where x is p1's state and y is p2's state)



Answer:

The alternative consistent states are,



7. RAFT

RAFT has a concept where the log is replicated to all participants. How does RAFT ensure that the log is equal on all nodes in case of a crash and a new leader?

Answer:

Whenever there is a crash, the candidates nodes organizes an election to elect a new leader.