THE UNIVERSITY OF SYDNEY

School of Computer Science
Dr. Ying Zhou

COMP5338: Advanced Data Models **Sem. 2/2020**

Week 5: MongoDB Replication

15.09.2020

Learning Objective and Scenario

The focus of this week's tutorial is read and write processing in replicated MongoDB Setting.

Suppose we have a MongoDB collection `users` with each document representing a user's profile. The profile document contains many fields. In this question, we are only interested in two fields: `age` and `status`. The collection is stored in a MongoDB replica set with three members. We assume all members are healthy and there is no problem with network communication among members. This exercise focuses on the following pair of read and write queries:

```
db.users.updateMany(
    { age: { $gt: 18 } },
    { $set: { status: 'A'} })

db.users.find(
    { age: { $gt: 20 },
    { age: 1, status: 1}})
```

For simplicity, we assume that

- All writes prior to t_0 have been successfully replicated to all members
- There is no concurrent write query in the time period of interest.
- Acknowledgment does not require appending to on-disk journal
- The write query affects two documents in the collection.
- The read query also returns two documents in the collection.
- No multi-document transaction is used.

Table 1 shows the most recent in memory value of the two documents at various replica set members between t_0 and t_9. It also shows important events during processing of the write request. At t_0 all members have consistent view of the data. The write query is issued at t_0. By t_1, the primary updates one document, by t_2, it updates both documents that satisfy the age criteria in the write command. This indicates completion of the write on primary member by t_2. The primary sends the write, in the form of `oplog`, at $t3$ to the two secondary members. Both members will apply the write and acknowledge back to the primary. The primary will acknowledge the client in proper time as determined by the write concern level and will also notify each secondary when should they update their snapshots of most recent `w: ''majority''` write.

Table 1: Time line of most recent values at Various Replica Set Members

time	**Primary**	**secondary 1**	**secondary 2**
t_0	{age:21, status:'u'} {age:23, status:'s'}	{age:21, status:'u'} {age:23, status:'s'}	{age:21, status:'u'} {age:23, status:'s'}
t_1	{age:21, status:'A'} {age:23, status:'s'}	{age:21, status:'u'} {age:23, status:'s'}	{age:21, status:'u'} {age:23, status:'s'}
t_2	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'u'} {age:23, status:'s'}	{age:21, status:'u'} {age:23, status:'s'}
t_3	**Primary sends the write oplog to Secondary 1 and 2**		
t_4	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'A'} {age:23, status:'s'}	{age:21, status:'u'} {age:23, status:'s'}
t_5	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'A'} {age:23, status:'s'}	{age:21, status:'A'} {age:23, status:'s'}
t_6	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'A'} {age:23, status:'s'}
t_7	**Secondary 1 acknowledges successful write**		
t_8	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'A'} {age:23, status:'A'}	{age:21, status:'A'} {age:23, status:'A'}
t_9	**Secondary 2 acknowledges successful write**		
t_{10}	**Primary receives Secondary 1's acknowledgement and notify secondary 1**		
t_{11}	**Primary receives Secondary 2's acknowledgement and notify secondary 2**		
t_{12}	**Secondary 1 receives notice to update its snapshot of its most recent w:"majority" write**		
t_{13}	**Secondary 2 receives notice to update its snapshot of its most recent w:"majority" write**		

Question 1: Default Setting

Suppose the write query uses default setting for **write concern**, which requests acknowledgement only from the primary. Suppose the **read preference** is set to default, meaning the read request will be answered by the primary. Suppose also that the the **read concern** is set to default, meaning the query returns data from the instance with no guarantee that the data has been written to a majority of the replica set members.

Answer the following questions:

1. When will the primary acknowledge the write back to the client?
2. What would return if the read happens between t_1 and t_2?
3. What would return if the read happens at t_3?
4. What would return if the read happens at t_7?

Question 2: Write concern: majority; read concern: majority; read preference: default

Suppose the write query sets the write concern to: `majority` and the read concern is also set to `majority`. The read preference is default.

Answer the following questions:

1. Which member will answer the read query?
2. When will the primary acknowledge the write back to the client?
3. What would return if the read happens between t_1 and t_2?
4. What would return if the read happens at t_3?
5. What would return if the read happens at t_7
6. What is the earliest time point for a read query to receive the updated value of both documents?

Question 3: Write concern: majority; read concern: majority; read preference: secondary

Suppose the write query sets the write concern to: `majority` and the read concern is also set to `majority`. The read preference is secondary, which can be any of the two secondaries.

Answer the following questions:

1. If secondary 1 is answering the read query, what is the earliest time point for it to return the updated value of both documents.
2. When would secondary 1 and secondary 2 return the same results for the read query?

Question 4: Other Valid Settings

Is it possible for a read query to return {age:21, status:'A'} {age:23, status:'s'} when write concern is set to `majority`? If yes, give a scenario of when this might happen.