**目录**

[1. 概述 1](#_Toc454972209)

[2. 升级前注意 2](#_Toc454972210)

[3. 升级单个mongod实例 2](#_Toc454972211)

[4. 升级副本集 2](#_Toc454972212)

[5. 升级分片集群 4](#_Toc454972213)

[5.1更新配置数据库元数据 8](#_Toc454972214)

[5.1.1集群更新前的注意事项 8](#_Toc454972215)

[5.1.2手动备份config库 9](#_Toc454972216)

[5.1.3检查磁盘空间，确保所有索引为{v:1 9](#_Toc454972217)

[5.1.4关闭均衡器 9](#_Toc454972218)

[5.1.5关闭40002端口的mongos 10](#_Toc454972219)

[5.1.6启动一个2.4的mongos开始升级 10](#_Toc454972220)

[5.1.7升级失败如何处理 12](#_Toc454972221)

[5.1.8重启均衡器 13](#_Toc454972222)

[5.2更新集群中的组件（mongos、mongod、mongod-config） 13](#_Toc454972223)

[5.2.1更新剩下的mongos 13](#_Toc454972224)

[5.2.2更新mongod-config 13](#_Toc454972225)

[5.2.3更新分片中的mongod实例 15](#_Toc454972226)

[5.2.4重新开启均衡器 18](#_Toc454972227)

[5.3总结 18](#_Toc454972228)

[6. 再说两句 18](#_Toc454972229)

[6.1MongoDB各个版本的数据文件版本号 18](#_Toc454972230)

[6.2在2.2版本运行2.4的数据会有什么问题 18](#_Toc454972231)

[7. 参考资料 19](#_Toc454972232)

**jar cvfm MyWordCount.jar manifest.data \*.class**

**关键数据升级务必进行升级安全评估，避免升级过程出现问题对业务造成致命影响！！！**

# 概述

根据应用场合的划分，将升级划分为三种类型单个mongod实例、副本集、分片集群，升级难度、繁琐程度依次递增。当然了，不同版本之间的升级过程也是不同的，mongodb升级必须逐步进行（比如从2.2升级到2.6，不能一步升级，必须先从2.2到2.4，然后从2.4到2.6），我会挑选几个代表性的版本进行升级示例，下面几篇会按照版本由低到高的顺序依次升级示例。

本篇介绍**如何从2.2.1+升级到2.4**。如下所介绍的都是升级到2.4内容。

# 升级前注意

升级之前有一些需要注意的地方如下：

* 如果现有版本使用了访问认证（必须用用户名密码访问），为了避免升级对客户端造成影响，在更新mongod或者mongos之前请先为客户端升级lib；
* 分片集群升级，必须最先升级元数据；
* 如果system.users文档中有内容，必须确认没有数据库使用它们。如果存在某一个库使用了它们，请现将其删除；

# 升级单个mongod实例

（1）在<https://www.mongodb.com/download-center#community>下载2.4版本的MongoDB；

（2）关闭旧的mongod，然后用2.4版本的MongoDB直接重启；

**J:\mongodb-64-2.2.4\bin\mongo ip:port**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --port=port --dbpath="…"**

# 升级副本集

为了方便描述，我们假定升级如下模型的副本集：



1. 首先更新各个secondary结点（这是必须的），我们先关闭Secondary-2结点的mongod，然后用mongod2.4替换重启。等待Secondary-2恢复到Secondary状态，再继续进行；

**J:\mongodb-64-2.2.4\bin\mongo ip:port**

**use admin;**

**db.shutdownServer(); # 如果不放心，可以登录到Primay结点执行rs.status()查看结点状态确认已经停掉**

**J:\mongodb-64-2.4.8\bin\mongod --port=port --dbpath="…" --replSet <replSetName>**

1. 等到Secondary-2恢复到Secondary状态之后，我们用同样的方法升级一下Secondary-1结点；
2. 此时，我们切换Primary到另外两个结点中的一个。登录到Primary上，执行切换动作，然后用2.4重启该节点；

**J:\mongodb-64-2.2.4\bin\mongo ip:port**

**use admin;**

**rs.stepDown(); # 切换Primay结点，这个动作会强迫执行一个故障转移failover**

**db.shutdownServer(); # 如果不放心，执行rs.status()查看结点状态确认已经关闭**

**J:\mongodb-64-2.4.8\bin\mongod --port=port --dbpath="…" --replSet <replSetName>**

由此可见，副本集的更新到2.4时是允许热更新的，而且注意到允许存在不同版本的mongod实例运行。

副本集的故障转移（failover，失效备援、故障转移）并不是立即完成的，在故障转移完成之前是不能进行写操作的，通常需要等待30s或者更多。

# 升级分片集群

为了演示分片集群更新，准备几个简单的集群大致是：2个mongos服务、3个mongo-config服务、2个分片服务（每个分片服务3个结点组成）。如下准备机器目录：

G:\mongodb-mongos1-40001

G:\mongodb-mongos2-40002

G:\mongodb-config1-30001

G:\mongodb-config2-30002

G:\mongodb-config3-30003

G:\mongodb-shard01repl-mongod1-10001

G:\mongodb-shard01repl-mongod2-10002

G:\mongodb-shard01repl-mongod3-10003

G:\mongodb-shard02repl-mongod1-20001

G:\mongodb-shard02repl-mongod2-20002

G:\mongodb-shard02repl-mongod3-20003

这个按照“05 MongoDB分片集群存储篇.doc”进行配置即可。

1. 配置2个分片

**J:\mongodb-64-2.2.4\bin\mongod --port=10001 --dbpath="G:\mongodb-shard01repl-mongod1-10001\data" --replSet=lt01 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongod --port=10002 --dbpath="G:\mongodb-shard01repl-mongod2-10002\data" --replSet=lt01 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongod --port=10003 --dbpath="G:\mongodb-shard01repl-mongod3-10003\data" --replSet=lt01 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:10001**

**use admin;**

**config = {\_id:"lt01", members:[{\_id:0, host:"127.0.0.1:10001"}, {\_id:1, host:"127.0.0.1:10002"}, {\_id:2, host:"127.0.0.1:10003"}]};**

**rs.initiate(config);**

**rs.status();**

**J:\mongodb-64-2.2.4\bin\mongod --port=20001 --dbpath="G:\mongodb-shard02repl-mongod1-20001\data" --replSet=lt02 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongod --port=20002 --dbpath="G:\mongodb-shard02repl-mongod2-20002\data" --replSet=lt02 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongod --port=20003 --dbpath="G:\mongodb-shard02repl-mongod3-20003\data" --replSet=lt02 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:20001**

**use admin;**

**config = {\_id:"lt02", members:[{\_id:0, host:"127.0.0.1:20001"}, {\_id:1, host:"127.0.0.1:20002"}, {\_id:2, host:"127.0.0.1:20003"}]};**

**rs.initiate(config);**

**rs.status();**

1. 启动mongos和配置服务

**J:\mongodb-64-2.2.4\bin\mongod --dbpath="G:\mongodb-config1-30001\data" --port=30001 --configsvr**

**J:\mongodb-64-2.2.4\bin\mongod --dbpath="G:\mongodb-config2-30002\data" --port=30002 --configsvr**

**J:\mongodb-64-2.2.4\bin\mongod --dbpath="G:\mongodb-config3-30003\data" --port=30003 --configsvr**

**J:\mongodb-64-2.2.4\bin\mongos --port=40001 --configdb="127.0.0.1:30001,127.0.0.1:30002,127.0.0.1:30003"**

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:40001**

**use admin;**

**db.runCommand({ addshard : "lt01/127.0.0.1:10001,127.0.0.1:10002,127.0.0.1:10003"});**

**db.runCommand({ addshard : "lt02/127.0.0.1:20001,127.0.0.1:20002,127.0.0.1:20003"});**

**db.runCommand({ listshards : 1 });**

**J:\mongodb-64-2.2.4\bin\mongos --port=40002 --configdb="127.0.0.1:30001,127.0.0.1:30002,127.0.0.1:30003"**

（3）创建片键，分片数据库，添加数据

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:40001**

**use admin;**

**db.runCommand({"enablesharding":"school"});**

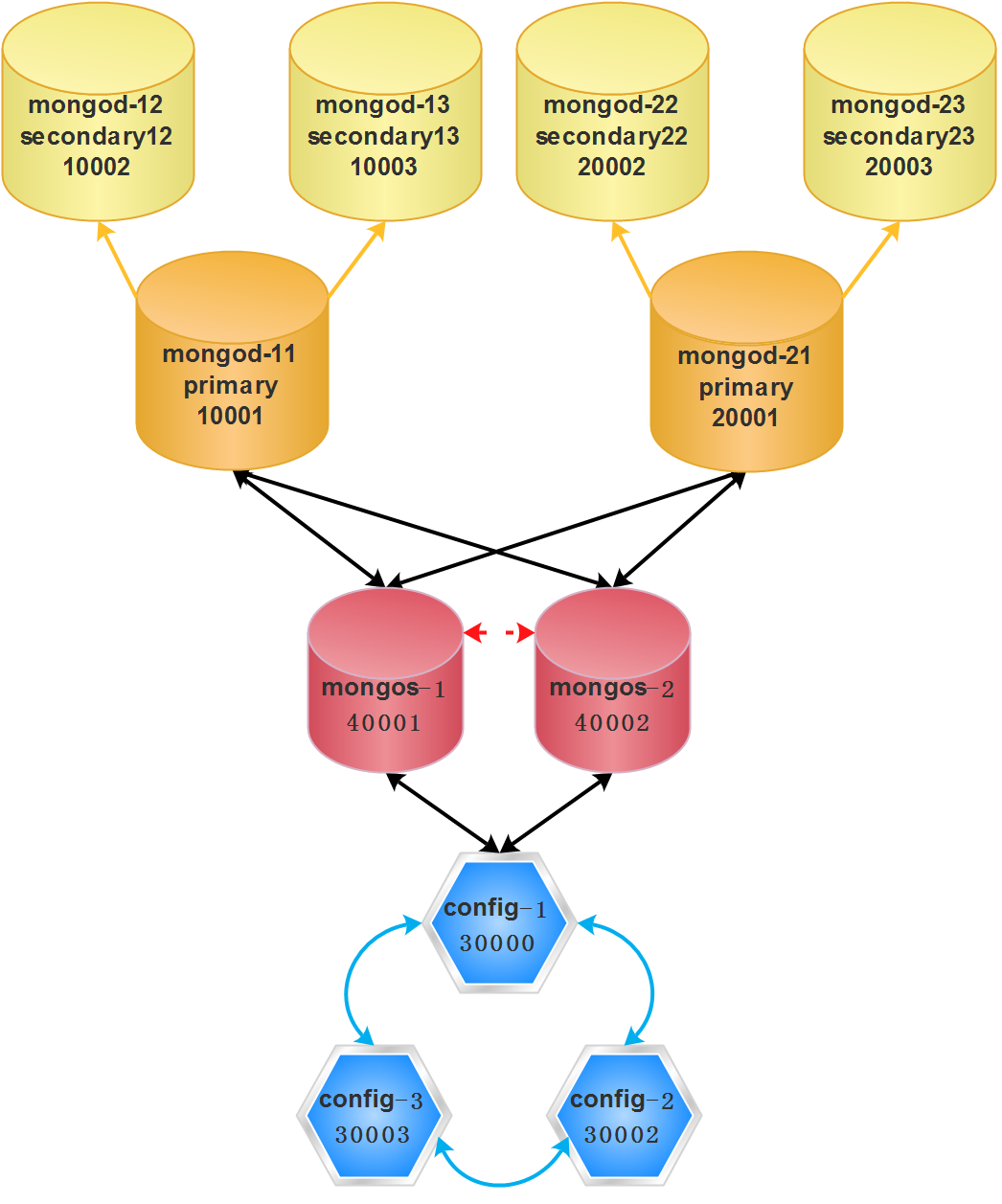
**db.runCommand({"shardcollection":"school.student", "key":{"\_id":1}});**

**use school;**

**for(var i=11; i<100000; i++){db.student.insert({"name":i})}**

**sh.status();**

模型如下：



将分片集群从2.2更新到2.4大致分为如下几步：

1. 更新配置数据库元数据；
2. 更新集群中的组件（mongos、mongod、mongod-config）；

## 5.1更新配置数据库元数据

### 5.1.1集群更新前的注意事项

（1）在配置服务的结点上，需要有**足够的未用的磁盘空间**free-space，free-space的大小至少是当前数据大小的4-5倍（升级配置数据库元数据的过程中，会备份config.chunks和config.collections）。另外，**需要确认所有库中的索引类型是{v:1}**，如果有些索引是{v:0}索引，数据块在切分的时候可能会出错，因为{v:0}索引是早些MongoDB版本的，如果有这样的情况，可以将这类索引删除。还有就是配置服务结点之间要确保配置服务结点之间的网络延迟较小。

查看索引：db.my\_student.getindexes();

删除索引：db.my\_student.dropIndexes(“insert\_time\_1”);

（2）在升级过程元数据的过程中，不能对元数据做修改，尤其是如下动作：

* [sh.enableSharding()](https://docs.mongodb.com/manual/reference/method/sh.enableSharding/#sh.enableSharding)；
* [sh.shardCollection()](https://docs.mongodb.com/manual/reference/method/sh.shardCollection/#sh.shardCollection)；
* [sh.addShard()](https://docs.mongodb.com/manual/reference/method/sh.addShard/#sh.addShard)；
* [db.createCollection()](https://docs.mongodb.com/manual/reference/method/db.createCollection/#db.createCollection)；
* [db.collection.drop()](https://docs.mongodb.com/manual/reference/method/db.collection.drop/#db.collection.drop)；
* [db.dropDatabase()](https://docs.mongodb.com/manual/reference/method/db.dropDatabase/#db.dropDatabase)；
* 任何创建数据库的动作；
* <https://docs.mongodb.com/manual/reference/sharding/>这个地方的所有命令，注意并不是所有命令都会导致修改元数据，但是为了确保最好在升级元数据的过程不要使用这个链接出的所有命令；

（3）警告：一旦升级到了2.4版本，就**不要使用2.4之前的mongos和mongod了**，如果使用mongos或者mongod**可能会**重建老的元数据格式。

说完注意事项，下面正式更新配置元数据：

### 5.1.2手动备份config库

虽然官方文档说在备份升级过程中会备份相关config库中的信息，但是我们还是手动先备份一下元数据（以防万一），我们手动执行：

**mongodump -host 127.0.0.1:30001 -d config -o "G:\mongodb\_upgrade\_backup"**

5.1.3检查磁盘空间，确保所有索引为{v:1}

检查磁盘空间是已经使用空间的4-5倍，查看config库中的所有索引是否存在{v:0}索引，如果有的话那么登录到mongos进行重建索引的操作db.collection.reIndex()。可能其他库中也有{v:0}索引，用同样的方式搞定。

### 5.1.4关闭均衡器

登录mongos（二者都可以），关闭均衡器。

**mongos> sh.getBalancerState();**

**true**

**mongos> sh.setBalancerState(false);**

**mongos> sh.getBalancerState();**

**false**

**mongos>**

### 5.1.5关闭40002端口的mongos

注意，升级元数据的时候，mongod-config的服务一定要处于开启状态，否则升级谁呢，对吧！

我们的集群中有2个mongos，我们逐个升级，这样可以避免业务受牵连（前提是业务采取了轮询mongos的方式访问）。在此我们先关闭40002端口的mongos服务，利用40002端口的mongos服务来升级元数据，如下进行。

### 5.1.6启动一个2.4的mongos开始升级

注意更新时，config-mongod必须打开，否则元数据将无法更新。

在此重新开启一个端口启动mongos2.4的一个实例，配置数据库依然使用当前集群中的配置数据库，使用更新的方式如下：

**J:\mongodb-64-2.4.8\bin\mongos.exe --port=40002 --configdb="127.0.0.1:30001,127.0.0.1:30002,127.0.0.1:30003" --upgrade**

如果升级正常，这个mongos会正常启动完成，正常启动完成即就意味着升级已经完成了。在控制台输出日志信息中会有如下信息：

**Wed Jun 22 20:06:51.115 [mongosMain] MongoS version 2.4.8 starting: pid=9580 port=40002 64-bit host=LIULEI26 (--help for usage)**

**… …**

**Wed Jun 22 20:06:51.388 [mongosMain] starting upgrade of config server from v3 to v4**

**Wed Jun 22 20:06:51.388 [mongosMain] starting next upgrade step from v3 to v4**

**Wed Jun 22 20:06:51.389 [mongosMain] about to log new metadata event: { \_id: "LIULEI26-2016-06-22T12:06:51-576a7f5b9d195e7e586ceada", server: "LIULEI26", clientAddr: "N/A", time: new Date(1466597211389), what: "starting upgrade of config database", ns: "config.version", details: { from: 3, to: 4 } }**

**… …**

**Wed Jun 22 20:06:54.638 [mongosMain] about to log new metadata event: { \_id: "LIULEI26-2016-06-22T12:06:54-576a7f5e9d195e7e586ceade", server: "LIULEI26", clientAddr: "N/A", time: new Date(1466597214638), what: "finished upgrade of config database", ns: "config.version", details: { from: 3, to: 4 } }**

**Wed Jun 22 20:06:54.774 [mongosMain] upgrade of config server to v4 successful**

**Wed Jun 22 20:06:54.911 [mongosMain] distributed lock 'configUpgrade/LIULEI26:40002:1466597211:41' unlocked.**

**Wed Jun 22 20:06:55.454 [Balancer] about to contact config servers and shards**

**Wed Jun 22 20:06:55.455 [Balancer] starting new replica set monitor for replicaset lt01 with seed of 127.0.0.1:10001,127.0.0.1:10002,127.0.0.1:10003**

**Wed Jun 22 20:07:01.824 [mongosMain] connection accepted from 127.0.0.1:12934 #8 (2 connections now open)**

看到Wed Jun 22 20:06:54.774 [mongosMain] upgrade of config server to v4 successful就表示我们的元数据升级成功了。

### 5.1.7升级失败如何处理

* 普通过程失败

如果是普通地方升级失败，你可以尝试重新升级。重新升级之前需要连接到之前的mongos查看一下之前的升级是否有升级锁存在：

**db.getMongo().getCollection("config.locks").findOne({ \_id : "configUpgrade" })**

如果这个地方有锁存在，我们可以手动将其重置，执行如下命令：

**db.getMongo().getCollection("config.locks").update({ \_id : "configUpgrade" }, { $set : { state : 0 } })**

注意：如果是存在文档锁或者其他非升级锁，不要进行重置锁的操作，这些锁只需等待其完成即可。

* critical section升级失败

如果在关键地方实际失败（升级时会看到entered critical section for config upgrade），需要做一些处理如下来同步3个配置服务器之间的数据（当然如果是1个那就不用了，按照普通失败处理然后重新升级即可）。重新同步各个配置服务数据如下：

注意：同步配置服务数据的时候，分片的数据依然是可以访问的，但是要避免修改元数据。不要进行这些操作：添加分片、删除数据库、删除文档。

1. 关闭均衡器，停止修改元数据；
2. 关闭2个配置服务，通常关闭后2个，比如你的配置服务是configA:27019、configB:27019、configC:27019，关闭configB、configC即可；
3. 用mongodump导出configA的所有数据；
4. 备份configB、configC的数据，然后各自重新创建一个数据目录；
5. 重启configB、configC，dbpath用新的空目录，port使用一个新的端口；
6. 用mongorestore将mongodump导出的数据，重新导入到新的configB、configC中，现在数据已经同步了；
7. 用老端口、新dbpath重新启动configB、configC；
8. 现在配置已经同步了，要重新升级元数据还需要一些操作：

**db.getMongo().getCollection("config.version").update({ \_id : 1 }, { $unset : { upgradeState : 1 } })**

### 5.1.8重启均衡器

连接已升级的mongos服务，执行sh.setBalancerState(“true”);重新打开均衡器。**此时也没有必要重启负载均衡器，因为整个升级还没有完成呢！**

## 5.2更新集群中的组件（mongos、mongod、mongod-config）

前面我们更新完了元数据，下面更新其他组件。同样的依然先要关闭均衡器，按照如下顺序更新。

### 5.2.1更新剩下的mongos

我们已经升级了40002端口的mongos了，下面我们升级其他端口的mongos服务，我们这里只有40001端口的mongos。首先我们关闭这个端口的mongos，然后按照新版本的mongo2.4进行重启（直接用新的mongos2.4启动，加上原来的配置数据库的信息即可）：

**J:\mongodb-64-2.4.8\bin\mongos.exe --port=40001 --configdb="127.0.0.1:30001,127.0.0.1:30002,127.0.0.1:30003"**

这样，我们就完成了所有（2个）mongos的升级。

### 5.2.2更新mongod-config

提示：注意必须更新完成所有的mongos再更新mongo-config服务。注意对于mongod-config的升级，官方给出如下建议：upgrading the first system in the mongos --configdb argument last.貌似是说，从--configdb参数的最后一个先开始更新。我的集群中mongos的--configdb选项的参数依次是："127.0.0.1:300 01,127.0.0.1:30002,127.0.0.1:30003"，所以我从30003端口的开始更新，关闭2.2.4的mongod-config，用2.4的重新启动。

* 关闭并重新启动30003端口的mongod-config服务

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:30003**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --dbpath="G:\mongodb-config3-30003\data" --port=30003 --configsvr**

* 关闭并重新启动30002端口的mongod-config服务

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:30002**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --dbpath="G:\mongodb-config2-30002\data" --port=30002 --configsvr**

* 关闭并重新启动30001端口的mongod-config服务

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:30001**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --dbpath="G:\mongodb-config1-30001\data" --port=30001 --configsvr**

至此，所有的mongod-config配置服务器已经更新完毕。

### 5.2.3更新分片中的mongod实例

至此我们的更新操作已经完成了一大半了，下面我们来更新分片集群。我们依次更新lt01、lt02分片。

关键点：先切换Secondary结点，然后更替Primary结点，最后再切换之前的Primary结点。

* 更新lt01分片

更新分片时稍微麻烦一点，有一个Primary结点更替的命令需要执行。因为我们的分片Primary承担着数据写入的任务，所以Primay必须时刻存在才能不影响业务。我们按照：lt01-10003、lt01-10002、lt01-10001的顺序依次更新，因为当前10001为Primay结点，因此不能第一个更新它。

1. 停止lt01-10003端口并以mongod2.4重启

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:10003**

**use admin;**

**db.shutdownServer(); # 如果不放心，可以登录到10001结点执行sh.status()查看结点状态确认已经停掉**

**J:\mongodb-64-2.4.8\bin\mongod --port=10003 --dbpath="G:\mongodb-shard01repl-mongod3-10003\data" --replSet=lt01 --shardsvr**

1. 停止lt01-10002端口并以mongod2.4重启

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:10002**

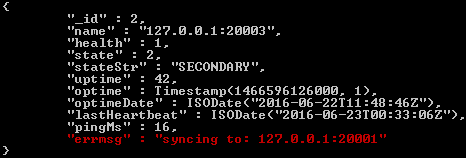
**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --port=10002 --dbpath="G:\mongodb-shard01repl-mongod2-10002\data" --replSet=lt01 --shardsvr**

1. 切换Primary结点

一定要注意！！！在切换之前查看另外两个已经启动的副本结点状态，rs.status()查看状态之后，不要出现errmsg的信息为“syncing to: \*\*\*\*\*\*“如下，



如果errmsg有如上信息说明此时正在从primary同步数据，此时最好等待完成，在进行结点的切换（官方没有，测试时个人发现，觉得应该这么做）。

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:10001**

**rs.stepDown()**

**rs.status();**

另外，还要注意，切换完结点之后要确认查看一下10001当前的状态，确认已经不是PRIMAYR结点了并且另外两个结点有一个已经是PRIMARY结点了，并且rs.status()之后没有errmsg的相关信息。

1. 停止lt01-10001端口并以mongod2.4重启

Primary结点已经切换，此时我们可以更新lt01-10001端口的结点了：

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:10001**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --port=10001 --dbpath="G:\mongodb-shard01repl-mongod1-10001\data" --replSet=lt01 --shardsvr**

至此lt01分片更新完成。

* 更新lt02分片

更新分片lt02同lt01，操作如下：

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:20003**

**use admin;**

**db.shutdownServer(); # 如果不放心，可以登录到20001结点执行sh.status()查看结点状态确认已经停掉**

**J:\mongodb-64-2.4.8\bin\mongod --port=20003 --dbpath="G:\mongodb-shard02repl-mongod3-20003\data" --replSet=lt02 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:20002**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --port=20002 --dbpath="G:\mongodb-shard02repl-mongod2-20002\data" --replSet=lt02 --shardsvr**

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:20001**

**rs.stepDown()**

**rs.status(); # 查看一下20001当前的状态，确认已经不是PRIMAYR结点了**

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:20001**

**use admin;**

**db.shutdownServer();**

**J:\mongodb-64-2.4.8\bin\mongod --port=20001 --dbpath="G:\mongodb-shard02repl-mongod1-20001\data" --replSet=lt02 --shardsvr**

### 5.2.4重新开启均衡器

重新打开集群中的均衡器。

**J:\mongodb-64-2.2.4\bin\mongo 127.0.0.1:40001**

**sh.setBalancerState(true);**

## 5.3总结

（1）元数据的更新至关重要，集群中其他组件的更新稍微简单一点；

（2）分片的更新，注意切换Primary结点；

# 再说两句

## 6.1MongoDB各个版本的数据文件版本号

|  |  |  |
| --- | --- | --- |
| MongoDB版本 | 数据文件Major | 数据文件Minor |
| 2.2 | 4 | 5 |
| 2.4 | 4 | 6 |
| 2.6 |  |  |
| 2.8 |  |  |
| 3.0 |  |  |
| 3.2 |  |  |
|  |  |  |

查看当前文档的数据格式版本db.getSiblingDB(database\_name).stats().dataFileVersion。

## 6.2在2.2版本运行2.4的数据会有什么问题

如果已经完成了升级，但是你需要用2.4中的数据在2.2下面运行，需要考虑如下问题：

* hash索引

如果你没有使用hash索引可以忽略该点。如果2.4版本的数据使用hash索引，那么2.2是无法进行查询的。此外，mongos2.2也无法进行插入操作，因为hashed索引2.2中还不支持，当2.2开启后根据hash片键根本无法将数据插入到正确的分片上。

* 2dsphere/text索引

没有使用该俩索引可以忽略该点。如果你创建了2dspherre/text索引，你可以在运行2.2的时候加上--upgrade并来删除2dsphere/text索引。如下操作：

**J:\mongodb-64-2.2.4\bin\mongod --dbpath=… --upgrade**

然后删除2dsphere索引和text索引（必须删除所有的这两类索引），删除索引使用：

**db.collection.dropIndex({loc:"2dsphere"});**

**db.collection.dropIndex("records\_text");**

如果你没有加--upgrade选项，直接运行mongod2.2那么会出现错误信息：'need to upgrade database index\_plugin\_upgrade with pdfile version 4.6, new version: 4.5 Not upgrading, exiting'。

如果你没有上面2个问题，那么你基本可以直接替换2.4为2.2直接运行。

# 参考资料

1. https://docs.mongodb.com/manual/release-notes/2.4-upgrade/