MongoDB - Deep Dive In the Config Database (Shardings)

Adamo Tonete & Vinodh Krishnaswamy

Support Engineer - MongoDB & MySQL
Percona

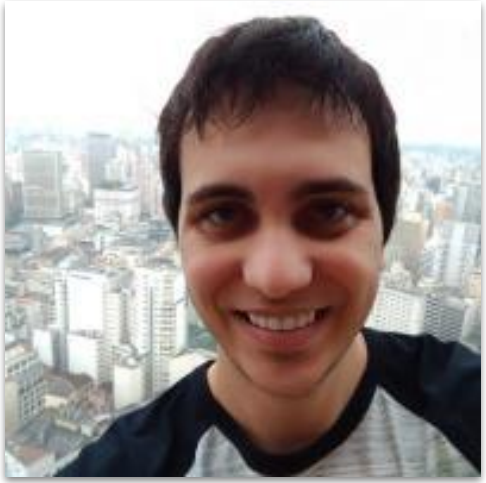
Who is Vinodh Krish!

- Started as MySQL DB DBA
- Support for MongoDB
- Trainer
- Scripting, Reading, Driving

- *"Guruji"* - who shares knowledge
- Now here I am - Perconian!!!

Who is Adamo!

- Senior Support Engineer
- São Paulo / Brazil
- @adamotonete



Agenda

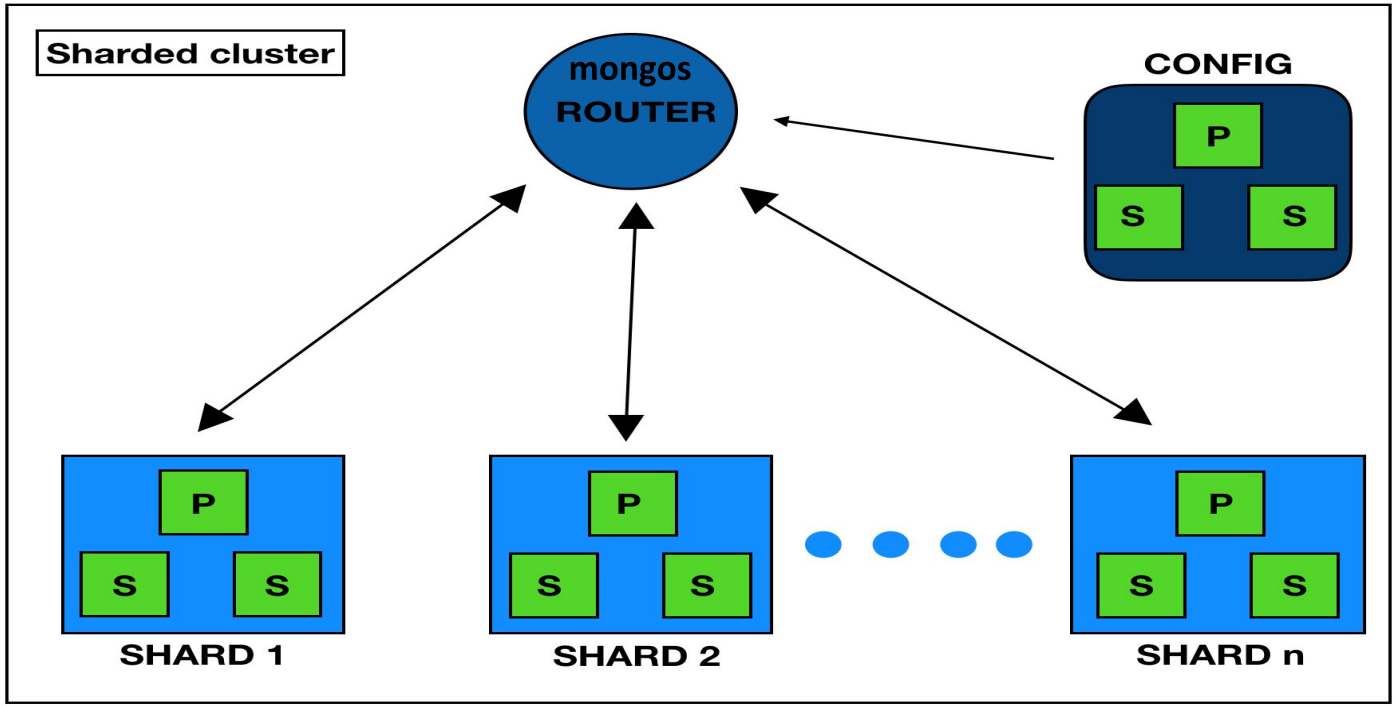
- Sharded Cluster
- Config server internals
  - actionlog
  - changelog
  - chunks
  - collections
  - Databases
  - Lockpings
  - locks
  - migrations

Agenda

- Config server internals
  - mongos
  - shards
  - tags
  - transactions
  - Version
- Other *collections* in *config* database
- Do’s and Don’ts
- Q&A

Sharded Cluster

Sharded Cluster



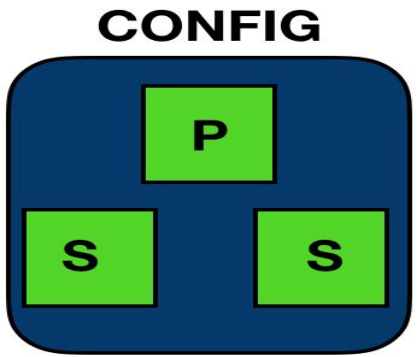
Sharded Cluster - Manage things

- App <-> mongos (router) <-> Shards (data)
- Shard details (Primary Shard)
- Database / Collection details
- Data Distribution :

  Data -> Shard -> chunks -> Balancing
- Zones

Sharded Cluster - metadata

- Who holds them?
- config server holds the metadata!



Sharded Cluster - Config

- Database “config” has the information

For Example,

config DB in psmdb 3.6 ->

```
mongos> use config
switched to db config
mongos> show collections
actionlog
changelog
chunks
collections
databases
lockpings
locks
migrations
mongos
settings
shards
tags
testData
transactions
version
mongos>
```

Sharded Cluster - mongos

- Cache the data from config server
- Use it to route READs and WRITEs to respective Shard
- Helps - Targeted Operation vs Broadcast Operation
- Updates the cache when chunk splits, shard changes etc.

Sharded Cluster - mongos

- What when there is metadata changes like movePrimary, jumbo clearing etc.
  - Restart to refresh the metadata
  - Refresh Cache while online

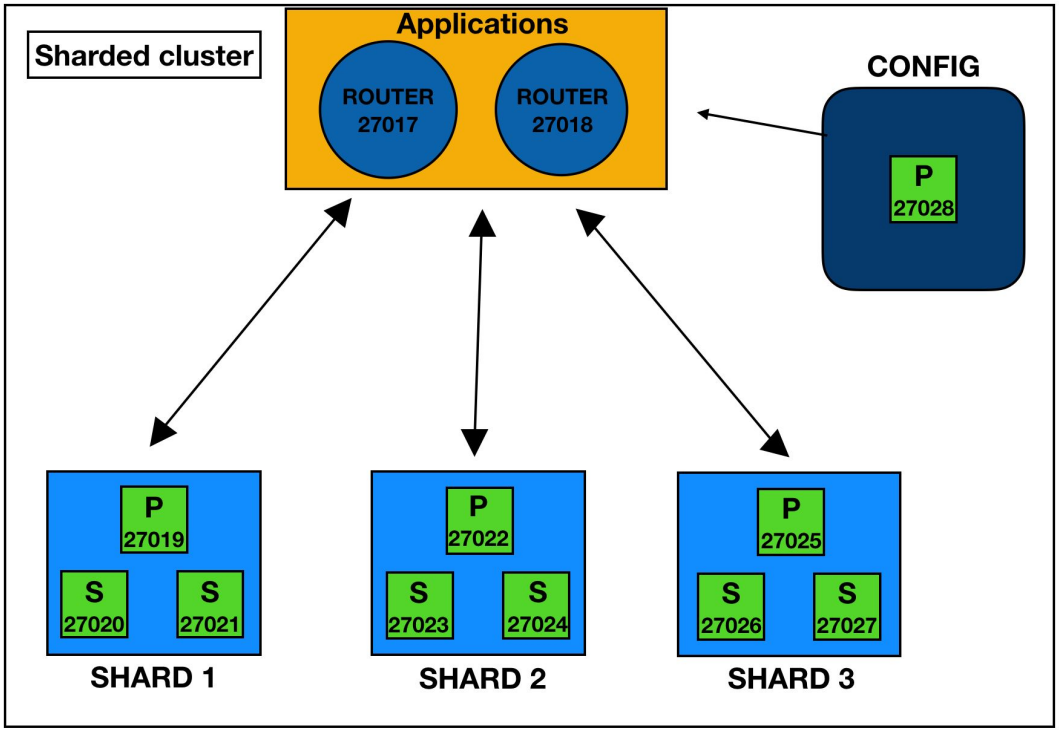
Sharded Cluster - mongos

- Restart to refresh the metadata
- Refresh Cache while online
  - Collection: **db.adminCommand({ flushRouterConfig: "<db.collection>" })**
  - Database: **db.adminCommand({ flushRouterConfig: "<db>" })**
  - Instance level: **db.adminCommand({ flushRouterConfig: 1})**

Config database - dive in

Config database - Test Cluster

- Architecture of our Test Sharded Cluster



PERCONA

Config database - Test Cluster

- Architecture of our Sharded Cluster - Database details

```
mongos> db.version()
3.6.11-3.1
mongos> show dbs
admin   0.000GB
config  0.002GB
vinodh  0.004GB
mongos> use vinodh
switched to db vinodh
mongos> show collections
testData
testData2
```

Config database - actionlog

- Actions of Balancer

```
mongos> db.actionlog.find().pretty()
{
        "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1-2019-04-21T08:28:55.431+0000-5cbc29c743d1e33d99187078",
        "server" : "vinodh-krishnaswamy-vinodh-test-standalone-1",
        "clientAddr" : "",
        "time" : ISODate("2019-04-21T08:28:55.431Z"),
        "what" : "balancer.round",
        "ns" : "",
        "details" : {
                "executionTimeMillis" : 501,
                "errorOccured" : false,
                "candidateChunks" : 1,
                "chunksMoved" : 1
        }
}
{
        "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1-2019-04-21T08:28:56.930+0000-5cbc29c843d1e33d991870a5",
        "server" : "vinodh-krishnaswamy-vinodh-test-standalone-1",
        "clientAddr" : "",
        "time" : ISODate("2019-04-21T08:28:56.930Z"),
        "what" : "balancer.round",
```

Config database - changelog

- All metadata change Operations
- Useful in Troubleshooting
- Check this first
- Example shows - *split* operation

```
mongos> db.changelog.find().sort({time:-1}).limit(1).pretty()
{
        "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1-2019-04-22T18:17:32.262+0000-5cbe053c430df567bede019c",
        "server" : "vinodh-krishnaswamy-vinodh-test-standalone-1",
        "clientAddr" : "127.0.0.1:49660",
        "time" : ISODate("2019-04-22T18:17:32.262Z"),
        "what" : "split",
        "ns" : "vinodh.testData2",
        "details" : {
                "before" : {
                        "min" : {
                                "id" : NumberLong("-179059785235328755")
                        },
                        "max" : {
                                "id" : NumberLong("4451356721936257132")
                        },
                        "lastmod" : Timestamp(3, 1),
                        "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6")
                },
                "left" : {
                        "min" : {
                                "id" : NumberLong("-179059785235328755")
                        },
                        "max" : {
                                "id" : NumberLong("2108209780126097837")
                        },
                        "lastmod" : Timestamp(3, 2),
                        "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6")
                },
                "right" : {
                        "min" : {
                                "id" : NumberLong("2108209780126097837")
                        },
                        "max" : {
                                "id" : NumberLong("4451356721936257132")
                        },
                        "lastmod" : Timestamp(3, 3),
                        "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6")
                }
        }
}
mongos>
```

Config database - changelog

Some more examples

```
mongos> db.changelog.find().sort({time:1}).skip(12).limit(1).pretty()
{
        "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1-2019-04-21T08:28:55.424+0000-5cbc29c7e99e5ff459cf8ba7",
        "server" : "vinodh-krishnaswamy-vinodh-test-standalone-1",
        "clientAddr" : "127.0.0.1:55680",
        "time" : ISODate("2019-04-21T08:28:55.424Z"),
        "what" : "moveChunk.from",
        "ns" : "vinodh.testData",
        "details" : {
                "min" : {
                        "id" : { "$minKey" : 1 }
                },
                "max" : {
                        "id" : 3001
                },
                "step 1 of 6" : 0,
                "step 2 of 6" : 3,
                "step 3 of 6" : 18,
                "step 4 of 6" : 94,
                "step 5 of 6" : 349,
                "step 6 of 6" : 11,
                "to" : "shard03",
                "from" : "shard02",
                "note" : "success"
        }
}
```


PERCONA

Config database - chunks

- Information about each ***chunk*** of the cluster
- Where they live
- Real data distribution about chunks

Config database - chunks

```
mongos> db.chunks.findOne()
{
        "_id" : "config.system.sessions-_id_MinKey",
        "ns" : "config.system.sessions",
        "min" : {
                "_id" : { "$minKey" : 1 }
        },
        "max" : {
                "_id" : { "$maxKey" : 1 }
        },
        "shard" : "shard01",
        "lastmod" : Timestamp(1, 0),
        "lastmodEpoch" : ObjectId("5cbc288343d1e33d99186a92")
}
mongos> db.chunks.find()
{ "_id" : "config.system.sessions-_id_MinKey", "ns" : "config.system.sessions", "min" : { "_id" : { "$minKey" : 1 } }, "max" : { "_id" : { "$maxKey" : 1 } }, "shard" : "shard01", "lastmod" : Timestamp(1
, 0), "lastmodEpoch" : ObjectId("5cbc288343d1e33d99186a92") }
{ "_id" : "vinodh.testData-id_MinKey", "lastmod" : Timestamp(2, 0), "lastmodEpoch" : ObjectId("5cbc291043d1e33d99186cc9"), "ns" : "vinodh.testData", "min" : { "id" : { "$minKey" : 1 } }, "max" : { "id"
: 3001 }, "shard" : "shard03" }
{ "_id" : "vinodh.testData-id_3001.0", "lastmod" : Timestamp(3, 0), "lastmodEpoch" : ObjectId("5cbc291043d1e33d99186cc9"), "ns" : "vinodh.testData", "min" : { "id" : 3001 }, "max" : { "id" : 7001 }, "sh
ard" : "shard01" }
{ "_id" : "vinodh.testData-id_7001.0", "lastmod" : Timestamp(3, 1), "lastmodEpoch" : ObjectId("5cbc291043d1e33d99186cc9"), "ns" : "vinodh.testData", "min" : { "id" : 7001 }, "max" : { "id" : { "$maxKey"
 : 1 } }, "shard" : "shard02" }
{ "_id" : "vinodh.testData2-id_MinKey", "lastmod" : Timestamp(2, 0), "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6"), "ns" : "vinodh.testData2", "min" : { "id" : { "$minKey" : 1 } }, "max" : { "id
" : NumberLong("-179059785235328755") }, "shard" : "shard01" }
{ "_id" : "vinodh.testData2-id_-179059785235328755", "lastmod" : Timestamp(3, 2), "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6"), "ns" : "vinodh.testData2", "min" : { "id" : NumberLong("-17905978
5235328755") }, "max" : { "id" : NumberLong("2108209780126097837") }, "shard" : "shard03" }
{ "_id" : "vinodh.testData2-id_4451356721936257132", "lastmod" : Timestamp(3, 1), "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6"), "ns" : "vinodh.testData2", "min" : { "id" : NumberLong("445135672
1936257132") }, "max" : { "id" : { "$maxKey" : 1 } }, "shard" : "shard02" }
{ "_id" : "vinodh.testData2-id_2108209780126097837", "lastmod" : Timestamp(3, 3), "lastmodEpoch" : ObjectId("5cbe041f430df567beddfaa6"), "ns" : "vinodh.testData2", "min" : { "id" : NumberLong("210820978
0126097837") }, "max" : { "id" : NumberLong("4451356721936257132") }, "shard" : "shard03" }
mongos>
```

Config database - collections

```
mongos> db.chunks.find().count()
8
mongos> sh.status()
--- Sharding Status ---
  sharding version: {
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("5cbc275943d1e33d9918665c")
  }
  shards:
        {  "_id" : "shard01",  "host" : "shard01/localhost:27019,localhost:27020,localhost:27021",  "state" : 1 }
        {  "_id" : "shard02",  "host" : "shard02/localhost:27022,localhost:27023,localhost:27024",  "state" : 1 }
        {  "_id" : "shard03",  "host" : "shard03/localhost:27025,localhost:27026,localhost:27027",  "state" : 1 }
  active mongoses:
        "3.6.11-3.1" : 2
  autosplit:
        Currently enabled: yes
  balancer:
        Currently enabled:  yes
        Currently running:  no
        Failed balancer rounds in last 5 attempts:  0
        Migration Results for the last 24 hours:
                2 : Success
  databases:
        {  "_id" : "config",  "primary" : "config",  "partitioned" : true }
                config.system.sessions
                        shard key: { "_id" : 1 }
                        unique: false
                        balancing: true
                        chunks:
                                shard01 1
                        { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" : 1 } } on : shard01 Timestamp(1, 0)
        {  "_id" : "vinodh",  "primary" : "shard02",  "partitioned" : true }
                vinodh.testData
                        shard key: { "id" : 1 }
                        unique: false
                        balancing: true
                        chunks:
                                shard01 1
                                shard02 1
                                shard03 1
                        { "id" : { "$minKey" : 1 } } -->> { "id" : 3001 } on : shard03 Timestamp(2, 0)
                        { "id" : 3001 } -->> { "id" : 7001 } on : shard01 Timestamp(3, 0)
                        { "id" : 7001 } -->> { "id" : { "$maxKey" : 1 } } on : shard02 Timestamp(3, 1)
                vinodh.testData2
                        shard key: { "id" : "hashed" }
                        unique: false
                        balancing: true
                        chunks:
                                shard01 1
                                shard02 1
                                shard03 2
```

Config database - databases

- Shows the databases information
- No system databases (admin, local etc)

```
mongos> db.databases.find()
{ "_id" : "vinodh", "primary" : "shard02", "partitioned" : true }
```

Config database - databases

- Shows even if it is not sharded:

```
mongos> use adamo
switched to db adamo
mongos> db.testData.insert({id:1})
WriteResult({ "nInserted" : 1 })
mongos> show dbs
adamo   0.000GB
admin   0.000GB
config  0.002GB
vinodh  0.004GB
mongos> use config
switched to db config
mongos>
mongos> db.databases.find()
{ "_id" : "vinodh", "primary" : "shard02", "partitioned" : true }
{ "_id" : "adamo", "primary" : "shard03", "partitioned" : false }
```

Config database - lockpings

- Tracks the pings of the active members in the Cluster

```
mongos> db.lockpings.find()
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27017:1555834711:-6294631489276540355", "ping" : ISODate("2019-04-22T09:43:39.853Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27018:1555834715:-5419806672643232120", "ping" : ISODate("2019-04-22T09:43:40.687Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27019:1555834724:-3820006162981004663", "ping" : ISODate("2019-04-22T09:43:50.944Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27021:1555834724:-8918282113069318315", "ping" : ISODate("2019-04-22T09:43:51.407Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27020:1555834724:-6558449033310927342", "ping" : ISODate("2019-04-22T09:43:51.407Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27024:1555834724:2948689541098604315", "ping" : ISODate("2019-04-22T09:43:51.407Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27023:1555834724:-6277057166590546775", "ping" : ISODate("2019-04-22T09:43:51.407Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27022:1555834724:4754386133603103927", "ping" : ISODate("2019-04-22T09:43:51.407Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27025:1555834725:-8076436883776105936", "ping" : ISODate("2019-04-22T09:43:50.944Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27026:1555834725:7410882278031117983", "ping" : ISODate("2019-04-22T09:43:50.944Z") }
{ "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27027:1555834725:-2102579600528248446", "ping" : ISODate("2019-04-22T09:43:50.944Z") }
{ "_id" : "ConfigServer", "ping" : ISODate("2019-04-22T19:44:50.036Z") }
```

Config database - locks

- Uses to store the distributed locks
- Balancer used to run on ***mongos*** and it acquires ***lock*** in old versions
- Starting in MongoDB 3.4, the balancer runs on the primary of the config server replica set
  - Primary member of config server replicaSet takes a balancer lock by modifying the ***_id: "balancer"*** document in the ***locks*** collection
  - MongoDB can perform parallel chunk migrations.
    - at most n/2 (rounded down) simultaneous chunk migrations. (n shards)
- Starting 3.6, the balancer no longer takes a ***lock***

Config database - locks

```
[mongos> db.locks.find().sort({ts:-1}).limit(1).pretty()
{
        "_id" : "adamo",
        "state" : 0,
        "process" : "ConfigServer",
        "ts" : ObjectId("5cbe1639430df567bede5390"),
        "when" : ISODate("2019-04-22T19:30:01.118Z"),
        "who" : "ConfigServer:conn153",
        "why" : "createDatabase"
}
mongos>
```

Config database - migrations

Config database - mongos

- Details of ***mongos*** instances of the Cluster

```
mongos> db.mongos.find().pretty()
{
        "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27017",
        "advisoryHostFQDNs" : [ ],
        "mongoVersion" : "3.6.11-3.1",
        "ping" : ISODate("2019-04-22T20:58:38.123Z"),
        "up" : NumberLong(10394),
        "waiting" : true
}
{
        "_id" : "vinodh-krishnaswamy-vinodh-test-standalone-1:27018",
        "advisoryHostFQDNs" : [ ],
        "mongoVersion" : "3.6.11-3.1",
        "ping" : ISODate("2019-04-22T20:58:38.680Z"),
        "up" : NumberLong(10393),
        "waiting" : true
}
mongos>
```

Config database - shards

- Shows the shard details of the cluster

```
mongos> db.shards.find().pretty()
{
        "_id" : "shard01",
        "host" : "shard01/localhost:27019,localhost:27020,localhost:27021",
        "state" : 1
}
{
        "_id" : "shard02",
        "host" : "shard02/localhost:27022,localhost:27023,localhost:27024",
        "state" : 1
}
{
        "_id" : "shard03",
        "host" : "shard03/localhost:27025,localhost:27026,localhost:27027",
        "state" : 1
}
```

Config database - tags

- Shows if ***tags*** / ***Zones*** enabled
- Example from other shard with tags enabled

```
mongos> db.tags.find()
{ "_id" : { "ns" : "vinodh.example", "min" : { "location" : "EU", "custId" : { "$minKey" : 1 } } }, "ns" : "vinodh.example", "min" : { "location" : "EU", "custId" : { "$minKey" : 1 } }, "max" : { "locat
ion" : "EU", "custId" : { "$maxKey" : 1 } }, "tag" : "EU" }
{ "_id" : { "ns" : "vinodh.example", "min" : { "location" : "ASIA", "custId" : { "$minKey" : 1 } } }, "ns" : "vinodh.example", "min" : { "location" : "ASIA", "custId" : { "$minKey" : 1 } }, "max" : { "l
ocation" : "ASIA", "custId" : { "$maxKey" : 1 } }, "tag" : "ASIA" }
{ "_id" : { "ns" : "vinodh.example", "min" : { "location" : "US", "custId" : { "$minKey" : 1 } } }, "ns" : "vinodh.example", "min" : { "location" : "US", "custId" : { "$minKey" : 1 } }, "max" : { "locat
ion" : "US", "custId" : { "$maxKey" : 1 } }, "tag" : "US" }
mongos> sh.status()
--- Sharding Status ---
  sharding version: {
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("5cbe287b0e64843113dc7493")
  }
  shards:
        {  "_id" : "shard01",  "host" : "shard01/localhost:27019,localhost:27020,localhost:27021",  "state" : 1,  "tags" : [ "US" ] }
        {  "_id" : "shard02",  "host" : "shard02/localhost:27022,localhost:27023,localhost:27024",  "state" : 1,  "tags" : [ "EU" ] }
        {  "_id" : "shard03",  "host" : "shard03/localhost:27025,localhost:27026,localhost:27027",  "state" : 1,  "tags" : [ "ASIA" ] }
```

Config database - settings

- Holds ***chunk*** size - modify here to change the chunk size
- Balancer Status
- Autosplit status

```
mongos> db.settings.find()
{ "_id" : "chunksize", "value" : 64 }
{ "_id" : "balancer", "stopped" : false }
{ "_id" : "autosplit", "enabled" : true }
mongos>
```

Config database - settings

- Change chunk size:

```
mongos> db.settings.find({_id:"chunksize"})
{ "_id" : "chunksize", "value" : 64 }
mongos> db.settings.save({ "_id" : "chunksize", "value" : 32 })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
mongos> db.settings.find({_id:"chunksize"})
{ "_id" : "chunksize", "value" : 32 }
mongos>
```

Config database - settings

- Enable balancer window:

```
mongos> sh.stopBalancer()
{
        "ok" : 1,
        "operationTime" : Timestamp(1555970667, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1555970667, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
mongos> use config
switched to db config
mongos> db.settings.update(
...     { _id: "balancer" },
...     { $set: { activeWindow : { start : "01:00", stop : "06:00" } } },
...     { upsert: true }
... )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
mongos> db.settings.find({_id:"balancer"})
{ "_id" : "balancer", "stopped" : true, "mode" : "off", "activeWindow" : { "start" : "01:00", "stop" : "06:00" } }
mongos> sh.setBalancerState(true)
{
        "ok" : 1,
        "operationTime" : Timestamp(1555970705, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1555970705, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
mongos>
```

Config database - settings

- Disable balancer window:

```
mongos> use config
switched to db config
mongos> db.settings.update({ _id : "balancer" }, { $unset : { activeWindow : true } })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
mongos> db.settings.find({_id:"balancer"})
{ "_id" : "balancer", "stopped" : false, "mode" : "full" }
mongos>
```

Config database - settings

- Enable / Disable autoSplit

  sh.enableAutoSplit()

  sh.disableAutoSplit()

Config database - version

- The version collection holds the current metadata version number. This collection contains only one document
- To access the version collection you must use the ***db.getCollection()*** method.

```
mongos> db.getCollection("version").find().pretty()
{
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("5cbc275943d1e33d9918665c")
}
mongos>
```

Other collections in config DB

Config database - system.sessions

- New in MongoDB 3.6
- This collection is not Cluster specific. Could find in replicaSet as well.
- Used for “Sessions”
- The system.sessions collection stores session records that are available to all members of the deployment.
- For Cluster, primaryShard for this collection is ***config server***

Config database - system.sessions

- From ***sh.status()*** below:
  - "**primary**" field is "**config**"

```
{  "_id" : "config",  "primary" : "config",  "partitioned" : true }
        config.system.sessions
                shard key: { "_id" : 1 }
                unique: false
                balancing: true
                chunks:
                        shard01 1
                { "_id" : { "$minKey" : 1 } } -->> { "_id" : { "$maxKey" : 1 } } on : shard01 Timestamp(1, 0)
{  "_id" : "vinodb",  "primary" : "shard02",  "partitioned" : true }
```

Config database - system.sessions

```
mongos> use config
switched to db config
mongos> db.system.sessions.find().pretty()
{
        "_id" : {
                "id" : UUID("fc7886be-9701-4b12-9e8b-9b334ba7d746"),
                "uid" : BinData(0,"47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=")
        },
        "lastUse" : ISODate("2019-04-22T21:09:53.884Z")
}
mongos> Date()
Mon Apr 22 2019 21:12:04 GMT+0000 (UTC)
```

Config database - transactions

- New in MongoDB 3.6
- This collection is not Cluster specific. Could find in replicaSet as well.
- Used for “Sessions”
- The transactions collection stores records used to support retryable writes for replica sets and sharded clusters.
- Do not edit this collection

Config database - transactions

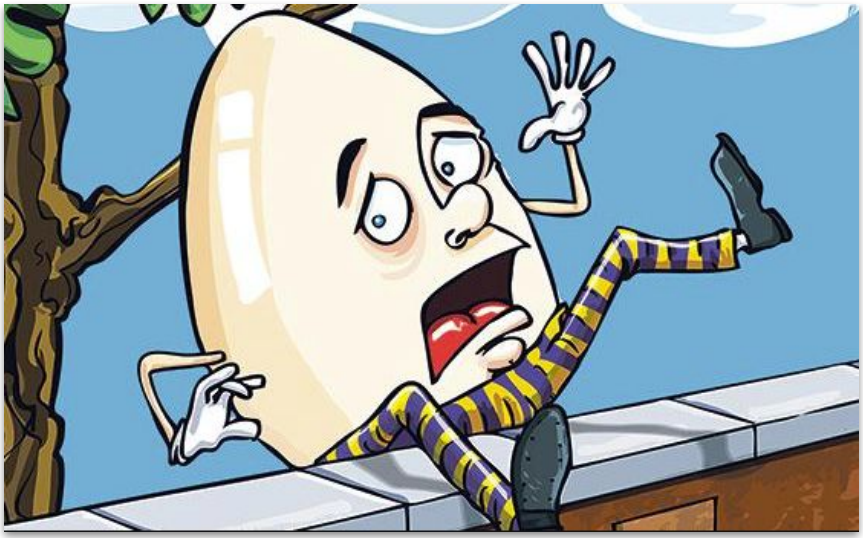
EXAMPLE FROM 4.0:

```
PRIMARY> db.test.insert({id:1})
WriteResult({ "nInserted" : 1 })
PRIMARY>
PRIMARY> session1 = db.getMongo().startSession()
session { "id" : UUID("548bf24f-f9d2-467d-b98f-700c5a932124") }
PRIMARY> session1.startTransaction()
PRIMARY> session1.getDatabase("percona").test.insert({today : new Date()})
WriteResult({ "nInserted" : 1 })
PRIMARY> session1.getDatabase("percona").test.insert({value : "hello"})
WriteResult({ "nInserted" : 1 })
PRIMARY> session1.commitTransaction()
PRIMARY>
PRIMARY> use config
switched to db config
PRIMARY> db.transactions.find()
{ "_id" : { "id" : UUID("548bf24f-f9d2-467d-b98f-700c5a932124"), "uid" : BinData(0,"47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=") }, "txnNum" :
NumberLong(0), "lastWriteOpTime" : { "ts" : Timestamp(1555968218, 1), "t" : NumberLong(2) }, "lastWriteDate" : ISODate("2019-04-22T21:23:38.061Z") }
```

Do’s & Don’ts

Config database - BEWARE!!!

- Any change to ***config*** DB affects the Sharded Cluster
- Make sure WHAT you do, BEFORE you do it!



Config database - Do's

- Stop Balancer
- Backup your *config* server

Question?

Send us your queries

- Percona Forums, JIRA - jira.percona.com
- Adamo

  adamo.tonete@percona.com

  Twitter: @adamotonete

- Vinodh

  vinodh.krishnaswamy@percona.com

  https://in.linkedin.com/in/vinodhkrish

THANK YOU