MongoDB Quick Start - Replica Sets and Sharding

Submitted by alex on Fri, 04/13/2012 - 00:40

Scaling out 1 node with Replica Set

If you already have a running mongod instance you first need restart it with --replSet

$ mongod --fork --rest --replSet mySetName --port 27017 --dbpath ~/Projects/db/data/mongo/r0 --logpath ~/Library/Logs/mongod-r0.log

Initialize the node via the mongo client

$ mongo

> rs.initiate()

{

        "info" : "Config now saved locally.  Should come online in about a minute.",

        "ok" : 1

}

This node is our Primary Node where read/write is done initially.

> db.messages.insert({City: "Tokyo"})

> db.messages.find()

{ "\_id" : ObjectId("4f86e502b5b1999b3ba7a748"), "City" : "Tokyo" }

Add backup/secondary nodes

Running on localhost hence different portnames, data dir and log

$ mongod --fork --rest --replSet mySetName --port 27018 --dbpath ~/Projects/db/data/mongo/r1 --logpath ~/Library/Logs/mongod-r1.log

Another backup

$ mongod --fork --rest --replSet mySetName --port 27019 --dbpath ~/Projects/db/data/mongo/r2 --logpath ~/Library/Logs/mongod-r2.log

Add secondary nodes to the Replica Set

Connect to the Primary Node.

$ mongo localhost:27017

> rs.add("localhost:27018")

{"ok" : 1}

> rs.add("localhost:27019")

{"ok" : 1}

Make a secondary node readable

Backup nodes are not readable as default

$ mongo localhost:27018

SECONDARY> db.messages.find()

error: { "$err" : "not master and slaveok=false", "code" : 13435 }

SECONDARY> db.getMongo().setSlaveOk()

not master and slaveok=false

SECONDARY> db.messages.find()

{ "\_id" : ObjectId("4f86e502b5b1999b3ba7a748"), "City" : "Tokyo" }

setSlaveOk() is per session...

Cannot insert to backup nodes

SECONDARY> db.messages.insert({Country: "Japan"})

not master

Add an Arbiter if the number of nodes are even

Adding an Arbiter is not needed in our example since we have 3 nodes

$ mongod --fork --rest --replSet mySetName --port 27020 --dbpath ~/Projects/db/data/mongo/ab --logpath ~/Library/Logs/mongod-arbiter.log

or

$ mongo localhost:27017

> rs.addArb("localhost:27020")

{"ok" : 1}

Replica Set with 3 nodes

$ mongo localhost:27017

> config = {\_id: 'subs', members: [

    {\_id: 0, host: 'localhost:27017'},

    {\_id: 1, host: 'localhost:27018'},

    {\_id: 2, host: 'localhost:27019'}]

  }

> rs.initiate(config)

Sharding with Replica Sets

Min req:

- 1 to 1000 shards

- 1 to 3 config servers (3 in prod)

- 1 or more mongos (db/shard router)

2 Shard servers, 1 shard with 2 RS nodes, 1 shard with 1 RS enabled node

--- Shard server 1 ---

$ mongod --fork --rest --shardsvr --replSet shard0001 --port 37017 --dbpath ~/Projects/db/data/mongo/r0 --logpath ~/Library/Logs/mongod-r0.log

$ mongod --fork --rest --replSet shard0001 --port 37018 --dbpath ~/Projects/db/data/mongo/r1 --logpath ~/Library/Logs/mongod-r1.log

--- Arbiter ---

$ mongod --fork --rest --replSet shard0001 --port 47018 --dbpath ~/Projects/db/data/mongo/ab --logpath ~/Library/Logs/mongod-arbiter.log

--- Initiate Shard 1 ---

$ mongo localhost:37018

> config = {\_id: 'subs', members: [

      {\_id: 0, host: 'localhost:37017'},

      {\_id: 0, host: 'localhost:37018'},

      {\_id: 1, host: 'localhost:47018', arbiterOnly: true}]

  }

> rs.initate(config)

--- Shard server 2 ---

$ mongod --fork --rest --shardsvr --replSet shard0002 --port 37019 --dbpath ~/Projects/db/data/mongo/r2 --logpath ~/Library/Logs/mongod-r2.log

--- Initiate Shard 2 ---

$ mongo localhost:37019

> config = {\_id: 'services', members: [{ \_id: 0, host: 'localhost:37019' } ]}

> rs.initate(config)

-- Config DB node ---

$ mongod --fork --rest --configsvr --dbpath ~/Projects/db/data/mongo/c0 --port 27019 --logpath ~/Library/Logs/mongo-configdb.log

--- DB/Shard router (def port 27017) ---

Connect to Config DB node

$ mongos --fork --configdb localhost:27019 --logpath ~/Library/Logs/mongos-1.log

Connect the dots...add the shards

Connect to mongos

$ mongos

mongos> use admin

mongos> db.runCommand( {addShard: 'localhost:37017'} )

{

        "ok" : 0,

        "errmsg" : "host is part of set: subs use replica set url format <setname>/<server1>,<server2>,...."

}

mongos> db.runCommand( {addShard: 'shard0001/localhost:37017'} )

{ "shardAdded" : "subs", "ok" : 1 }

mongos> db.runCommand( {addShard: 'shard0002/localhost:37019'} )

{ "shardAdded" : "services", "ok" : 1 }

Enable sharding for the specified DB

mongos> db.runCommand( { enablesharding : "people" } );

Import some tests data

$ curl http://api.twitter.com/1/statuses/user\_timeline.json?screen\_name=alyu | mongoimport --jsonArray -d people -c twitter

$ curl http://api.twitter.com/1/statuses/user\_timeline.json?screen\_name=severalnines | mongoimport --jsonArray -d people -c twitter

$ curl http://api.twitter.com/1/statuses/user\_timeline.json?screen\_name=cnn | mongoimport --jsonArray -d people -c twitter

Add unique index to the collection

mongos> db.twitter.ensureIndex({"\_id": 1})

Enable sharding on the collection

mongos> db.runCommand( { shardcollection : "people.twitter" , key : { \_id : 1 } , unique : true } );

{ "collectionsharded" : "people.twitter", "ok" : 1 }

Sharding misc

mongos> use config

mongos> show collections

chunks

databases

lockpings

locks

mongos

settings

shards

system.indexes

version

mongos> db.mongos.find()

{ "\_id" : "hyperion.local:27017", "ping" : ISODate("2012-04-12T11:57:35.104Z"), "up" : 951 }

mongos> db.printShardingStatus()

--- Sharding Status ---

  sharding version: { "\_id" : 1, "version" : 3 }

  shards:

        {  "\_id" : "shard0001",  "host" : "shard0001/localhost:37017,localhost:37018" }

        {  "\_id" : "shard0002",  "host" : "shard0002/localhost:37019" }

  databases:

        {  "\_id" : "admin",  "partitioned" : false,  "primary" : "config" }

...

mongos> db.printShardingStatus()

--- Sharding Status ---

  sharding version: { "\_id" : 1, "version" : 3 }

  shards:

        {  "\_id" : "shard0001",  "host" : "shard0001/localhost:37017,localhost:37018" }

        {  "\_id" : "shard0002",  "host" : "shard0002/localhost:37019" }

  databases:

        {  "\_id" : "admin",  "partitioned" : false,  "primary" : "config" }

        {  "\_id" : "people",  "partitioned" : true,  "primary" : "subs" }

                people.twitter chunks:

                               shard0001    2

                        { "\_id" : { $minKey : 1 } } -->> { "\_id" : ObjectId("4f86cc60fa08cbf3838451f5") } on : shard0001 { "t" : 1000, "i" : 1 }

                        { "\_id" : ObjectId("4f86cc60fa08cbf3838451f5") } -->> { "\_id" : { $maxKey : 1 } } on : shard0001 { "t" : 1000, "i" : 2 }

        {  "\_id" : "twitter",  "partitioned" : false,  "primary" : "shard0002" }

mongos> db.runCommand({ismaster:1});

mongos> db.runCommand({ listShards : 1});

config = db.getSisterDB("config")

mongos> config.databases.find()

mongos> use config

mongos> db.locks.find( { \_id : "balancer" } )

{ "\_id" : "balancer", "process" : "hyperion.local:27017:1334230904:16807", "state" : 0, "ts" : Obje