**Sharding, Replica Set**

* sharding : 데이터를 나눠서
* replica set: 복제 셋(복구용도)

=> 둘은 거의 같이 쓰임

* 샤드는 홀수로 7개이상 하는 것을 권장.

shard key) index와 비슷.

- chunk 단위(하나의 온전한 단위)로 나누기 쉬워야 함. 샤드에 64mb까지 가능. 일반적인 도큐먼트라고 생각.

랜덤키를 이용하여 모든 문서가 균등하게 나뉘어야. (부하가 걸리는 서버가 없어야)

-\_id키로 하면 데이터가 한곳에 몰릴 가능 성이 있음(왜? 시간에 따라서 id가 되기 때문에)

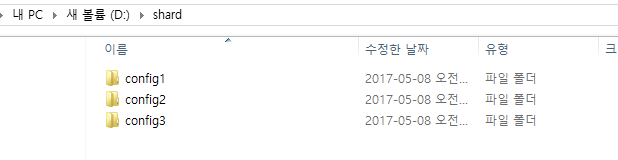
-농도(Cardinality) : 필드의 유일성을 정의.

----------- 샤드실습 --------

Config 서버를 ReplicaSet으로 만들 것.

3.4 버전부터는 반드시 config 서버를 ReplicaSet으로 만들어야 한다.

1. **config 및 Replicaset 만들기**
2. 폴더만들기



2) config 의 RelicaSet 설정

* cmd : mogod --configsvr --replSet <세트이름> --dbpath <config 서버경로> -port

→ config서버로 만들겠다.

* <Config 서버 포트>
* 서버 절때 끄지 말기 !
* 

3) 서버접속

cmd 하다 더 키고

만들어진 ConfigServer중 하나에 접속한다 - 접속한 config 서버가 Primary가 된다.

mongo --host <hostname> --port <port>



4) 묶어주는 작업

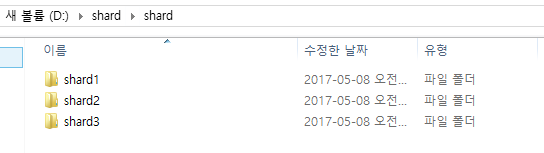
그 다음엔 샤딩을 위한 ConfigServer들을 등록시킨다. (컴퓨터 다를 때, localhst부분 ip로 바꾸어야 함.)

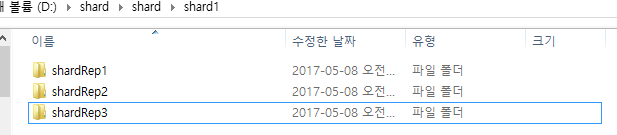
|  |
| --- |
| rs.initiate(  {  \_id: "configRepl",  configsvr: true,  members: [  { \_id : 0, host : "localhost:20001" },  { \_id : 1, host : "localhost:20002" },  { \_id : 2, host : "localhost:20003" }  ]  }  ) |

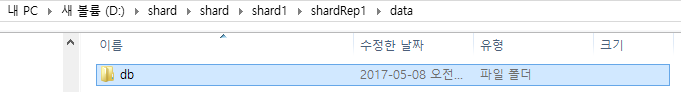
이후 exit로 나가기

**2. Shard 만들기(3개-복제 셋 도)**

1. 폴더만들기







나머지 shard Repl2, 3도 data의 db폴더까지 만들기.

다된걸 shard2, 3에 똑같이 복사

2) Replicaset설정 (총 9번 실행. shard3개 x Replicaset3개)

cmd창 9개 띄워야함 !! 끄지말기. 번호주의 (9개를 경로다르게 하여 설정)

|  |
| --- |
| mongod --shardsvr --replSet shardRepl3 --dbpath D:\shard\shard\shard3\shardRep3\data\db --port 30033 |

3. 묶어주기

3개의 서버를 묶어주기(secondary로 지정되는건 신경안써줘도됨)

mongo --host localhost --port 30031

|  |
| --- |
| mongo --host localhost --port 30011  rs.initiate(  {  \_id : "shardRepl1",  members: [  { \_id : 0, host : "localhost:30011" },  { \_id : 1, host : "localhost:30012" },  { \_id : 2, host : "localhost:30013" }  ]  }  )    mongo --host localhost --port 30021  rs.initiate(  {  \_id : "shardRepl2",  members: [  { \_id : 0, host : "localhost:30021" },  { \_id : 1, host : "localhost:30022" },  { \_id : 2, host : "localhost:30023" }  ]  }  )    mongo --host localhost --port 30031  rs.initiate(  {  \_id : "shardRepl3",  members: [  { \_id : 0, host : "localhost:30031" },  { \_id : 1, host : "localhost:30032" },  { \_id : 2, host : "localhost:30033" }  ]  }  ) |

**3. mongos 연결하기**(클라이언트를 위한 통로 만들기)

mongos --configdb <configReplSetName>/<host:port>,<host:port>,<host:port>

|  |
| --- |
| mongos --configdb configRepl/localhost:20001,localhost:20002,localhost:20003 -port 20000 |

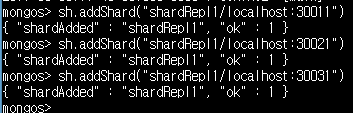
20000번 포트로 몽고스 연결. config로 가는 mongos를 연결해놨다. 직접적으로 가진 못해서 2만번 포트를 통해 연결.



mongos로 연결 됨

샤드 붙이기

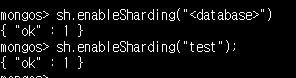
sh.addShard(“<replSetName>/s1-mongo1.example.net:27017”)

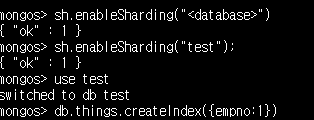


status로 상태확인



1. 샤딩 가능한 데이터베이스 등록시키기





인덱스를 만들고 실행해야함. 단일 인덱스만 가능.

2) 샤딩 설정 전에 컬랙션의 인덱스를 먼저 걸어줘야 한다.

3) 인덱스를 통해 샤드 키를 만들기(해쉬 방법 이용)

권리자 권한으로 실행





샤딩 가능한 컬렉션 등록시키기(해쉬를 통해 적절히 분배될 수 있도록)

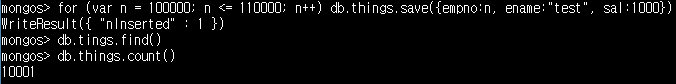
sh.shardCollection(“<satabase>.<collection>”, {<key> : <diriection> })

샤드확인



대략 1만건의 데이터 넣기

for (var n = 100000; n <= 110000; n++) db.things.save({empno:n, ename:"test", sal:1000})



exit로 나간 후

잘분배됐는지 확인) 10000개를 3개나눠서 하니깐. 3천대로.되면 잘 된 것.(primary)



나머지도 확인해보기.

키에 대하여 )

Ranged Shard Key : 데이터가 한곳에 몰릴 수 도 있음.(0~1000은 shard1, 1001~3000은 shard2…)

Hashed Shard Key: 유일한 값으로 키값을 해야 데이터가 몰리지 않음. 키를 기준으로 분배. (ex. id % 3). 검색속도 줄임. 아이템이 저장된 그룹을 지정. 검색에 대한 범위를 줄이는 것이 핵심.