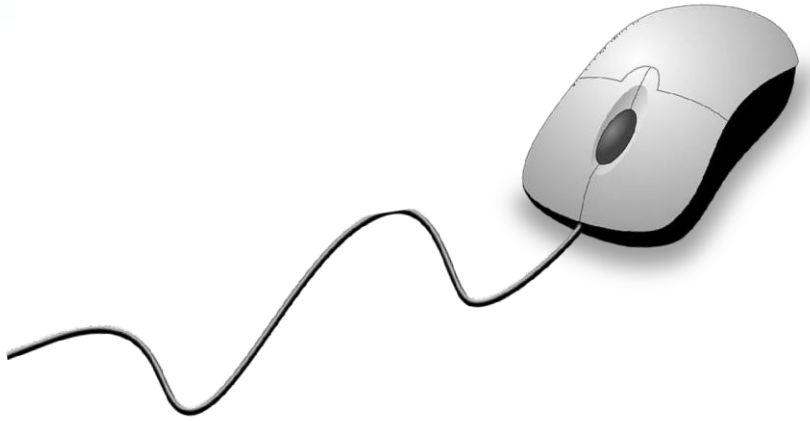
공개SW 솔루션 설치 & 활용 가이드

시스템SW > 데이터관리



mongoDB

제대로 배워보자

How to Use Open Source Software

Open Source Software Installation & Application Guide

오픈소스 소프트웨어 통합지원센터
Open Source Software Support Center

CONTENTS

1. 개요

소개	• MongoDB는 크로스 플랫폼 도큐먼트 지향 데이터베이스 시스템 • NoSQL 데이터베이스로 분류되는 MongoDB는 JSON과 같은 동적 스키마형 문서들(MongoDB는 이러한 포맷을 BSON이라 부름)을 선호함에 따라 전통적인 테이블 기반 관계형 데이터베이스 구조의 사용 꺼림		
주요기능	• MongoDB는 데이터를 유연한 JSON 형식의 문서로 저장, 즉 필드마다 문서마다 다를 수 있으며 시간이 지남에 따라 데이터 구조가 변경 될 수 있음 • MongoDB는 핵심적인 분산 데이터베이스이므로 고가용성, 수평 확장 및 지리적 분포가 내장되어 사용하기 쉬움		
대분류	• 시스템 SW	**소분류**	• 데이터관리
라이선스 형태	• GNU AGPL v3.0	**사전설치 솔루션**	• N/A
실행 하드웨어	• N/A	**버전**	• 4.0.2(2018년 10월 기준)
특징	• MongoDB는 Replica set으로 고가용성 제공 • MongoDB는 Sharding을 사용하여 매우 큰 데이터와 높은 처리량의 작업 지원		
보안취약점	• 취약점 ID : CVE-2016-10572 • 심각도 : 8.1 HIGH(V3) • 취약점 설명 : mongodb-instance 암호화되지 않은 HTTP 연결을 통해 실행 파일을 안전하지 않게 다운로드 • 대응방안 : 0.0.3 이상으로 업데이트 • 참고 경로 : https://www.npmjs.com/advisories/235		
개발회사/커뮤니티	• MongoDB / MongoDB Community		
공식 홈페이지	• https://www.mongodb.org		

2. 기능요약

- MongoDB의 주요 기능

주요기능	지원여부
Community / Enterprise	지원
Replication 기능	지원
Sharding 기능	지원
Transaction 기능	지원
Security 기능	지원
Backup / Restore 기능	지원

3. 실행환경

- 타 NoSQL Database 비교

구분	Redis	Cassandra	MongoDB
저장 방식	Key/value Store	column/column family data Store	Document-oriented Store
사용 언어	C	JAVA	C++
라이선스	BSD	Apache	Apache
운영체제	크로스 플랫폼	크로스 플랫폼	크로스 플랫폼
최신버전	4.0.11	3.11.3	4.0.3

4. 설치 및 실행

세부 목차

1. 사전 작업
   1. SELinux 비활성화
   2. Ulimits settings
   3. Transparent Huge Pages 비활성화
2. MongoDB 설치
   1. MongoDB Community를 수동으로 설치하기 위한 dependencies 설치
   2. Download and extract the MongoDB Community package
   3. Directory 생성한 후에 압축을 푼 file들을 directory에 복사
   4. MongoDB config file 생성
   5. Ensure that the MongoDB binaries are in your PATH
3. 설치 완료
   1. Start MongoDB daemon
   2. Begin using MongoDB

4. 설치 및 실행

4.1.1 SELinux 비활성화

- SELinux 설정을 disabled로 설정하여 비활성화
- root계정에서 실행
- 변경 사항을 적용하려면 반드시 시스템 재부팅

  → SELINUX=disabled

```
[root@localhost ~]# cat /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

4. 설치 및 실행

4.1.2 Ulimits settings

- Ulimits는 단일 사용자가 너무 많은 시스템 자원을 사용하는 것 방지
- 한계 값이 낮으면 정상적인 MongoDB 작업 중에 많은 문제 발생
- root계정에서 실행
- 한계 값을 수정한 후에 반드시 재부팅을 해야 적용

  → vi /etc/security/limits.conf

  ```
  *      hard    nproc     64000
  *      soft    nproc     64000
  *      hard    nofile    64000
  *      soft    nofile    64000
  ```

  → ulimit –a

  => -f (file size): unlimited
  -t (cpu time): unlimited
  -v (virtual memory): unlimited
  -n (open files): 64000
  -m (memory size): unlimited
  -u (processes/threads): 64000

```
[root@localhost ~]# cat /etc/security/limits.conf
# /etc/security/limits.conf

#<domain>      <type>  <item>         <value>
#
*      hard    nproc     64000
*      soft     nproc     64000
*      hard    nofile     64000
*      soft     nofile     64000
#*               soft    core            0
#*               hard    rss             10000
#@student        hard    nproc           20
#@faculty        soft    nproc           20
#@faculty        hard    nproc           50
```

```
[root@localhost ~]# ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 15053
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 64000
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 64000
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

4. 설치 및 실행

4.1.3 Transparent Huge Pages 비활성화

- 이 작업을 하지 않고 MongoDB에 접속하면 WARNING이 뜨는 것 확인

```
WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
        We suggest setting it to 'never

WARNING: /sys/kernel/mm/transparent_hugepage/defrag is 'always'.
        We suggest setting it to 'never'
```

- 해당 경로에 있는 enabled와 defrag에 있는 속성이 'always'로 설정되어 있는데, 이것을 'never'로 변경하라는 의미
- 아래와 같이 /etc/rc.local에 들어가 never로 변경하는 명령 줄을 추가한 후에 저장을 하고 실행권한 줌
- root계정에서 실행
- 그리고 반드시 재부팅을 해야 시스템에서 변경사항이 영구적으로 적용

→ vi /etc/rc.local
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag
chmod +x /etc/rc.local

```
[root@localhost ~]# cat /etc/rc.local
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag
[root@localhost ~]#
[root@localhost ~]# chmod +x /etc/rc.local
[root@localhost ~]# reboot
Connection closing...Socket close.
```

4. 설치 및 실행

4.2.1 MongoDB Community를 수동으로 설치하기위한dependencies설치

- MongoDB Community를 수동으로 설치하려면 우선적으로 운영체제에 맞는 dependencies 설치
- root계정에서 실행
- MongoDB 사이트에서 운영체제 버전에 맞는 dependencies 찾아 설치

  → yum install libcurl openssl

```
[root@localhost ~]# yum install libcurl openssl
Loaded plugins: fastestmirror, langpacks
Repodata is over 2 weeks old. Install yum-cron? Or run: yum makecache fast
base
extras
updates
(1/2): extras/7/x86_64/primary_db
(2/2): updates/7/x86_64/primary_db
Determining fastest mirrors
 * base: mirror.kakao.com
 * extras: mirror.kakao.com
 * updates: mirror.kakao.com
```

4. 설치 및 실행

4.2.2 Download and extract the MongoDB Community package

- https://www.mongodb.com/download-center에서 file을 받은 후 압축을 품
- root계정에서 실행

→ wget https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-rhel70-4.0.3.tgz

```
[root@localhost ~]# wget https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-rhel70-4.0.3.tgz
--2018-10-15 16:47:41--  https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-rhel70-4.0.3.tgz
Resolving fastdl.mongodb.org (fastdl.mongodb.org)... 54.230.181.56, 54.230.181.77, 54.230.181.223,
Connecting to fastdl.mongodb.org (fastdl.mongodb.org)|54.230.181.56|:443... connected.
HTTP request sent, awaiting response... 200 OK
```

→ tar -zxvf mongodb-linux-*-4.0.3.tgz

```
[root@localhost ~]# tar -zxvf mongodb-linux-x86_64-rhel70-4.0.3.tgz
mongodb-linux-x86_64-rhel70-4.0.3/README
mongodb-linux-x86_64-rhel70-4.0.3/THIRD-PARTY-NOTICES
mongodb-linux-x86_64-rhel70-4.0.3/MPL-2
mongodb-linux-x86_64-rhel70-4.0.3/GNU-AGPL-3.0
mongodb-linux-x86_64-rhel70-4.0.3/LICENSE-Community.txt
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongodump
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongorestore
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongoexport
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongoimport
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongostat
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongotop
mongodb-linux-x86_64-rhel70-4.0.3/bin/bsondump
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongofiles
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongoreplay
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongod
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongos
mongodb-linux-x86_64-rhel70-4.0.3/bin/mongo
mongodb-linux-x86_64-rhel70-4.0.3/bin/install_compass
```

4. 설치 및 실행

4.2.3 Directory 생성한 후에 압축을 푼 file들을 directory에 복사

- data file, log file이 저장될 directory를 생성하고, 압축을 푼 directory에 있는 file들을 생성한 directory 복사
- root계정에서 실행

  → mkdir -p mongodb/data mongodb/log

  → cp -r mongodb-linux-x86_64-rhel70-4.0.3/* mongodb

```
[root@localhost ~]# mkdir -p mongodb/data mongodb/log
[root@localhost ~]# cp -r mongodb-linux-x86_64-rhel70-4.0.3/* mongodb

[root@localhost ~]#
[root@localhost ~]# ll mongodb
total 132
drwxr-xr-x 2 root root  4096 Oct 15 16:53 bin
drwxr-xr-x 2 root root  4096 Oct 15 16:53 data
-rw-r--r-- 1 root root 34520 Oct 15 16:53 GNU-AGPL-3.0
-rw-r--r-- 1 root root  2149 Oct 15 16:53 LICENSE-Community.txt
drwxr-xr-x 2 root root  4096 Oct 15 16:53 log
-rw-r--r-- 1 root root 16726 Oct 15 16:53 MPL-2
-rw-r--r-- 1 root root  2195 Oct 15 16:53 README
-rw-r--r-- 1 root root 57190 Oct 15 16:53 THIRD-PARTY-NOTICES
[root@localhost ~]#
```

4. 설치 및 실행

4.2.4 MongoDB config file 생성

- 환경설정을 위하여 config file을 생성한 후에 필요한 설정들 작성
- root계정에서 실행

→ systemLog:
  destination: file
  logAppend: true
  path: /root/mongodb/log/mongod.log

Where and how to store data.
storage:
  dbPath: /root/mongodb/data
  journal:
    enabled: true

how the process runs
processManagement:
  fork: true
  pidFilePath: /root/mongodb/config.pid
  timeZoneInfo: /usr/share/zoneinfo

network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0

```
[root@localhost ~]# cat /etc/mongod.conf
# mongod.conf
# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /root/mongodb/log/mongod.log

# Where and how to store data.
storage:
  dbPath: /root/mongodb/data
  journal:
    enabled: true
#  engine:
#  mmapv1:
#  wiredTiger:

# how the process runs
processManagement:
  fork: true  # fork and run in backgroundcd lo
  pidFilePath: /root/mongodb/config.pid  # location of pidfile
  timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0  # Listen to local interface only, comment to listen
```

4. 설치 및 실행

4.2.5 Ensure that the MongoDB binaries are in your PATH

- 현재 MongoDB binary file은 /root/mongodb/bin directory에 있음
- root계정에서 실행
- 사용자가 쉽게 데몬 구동과 접속을 할 수 있도록 alias 생성

  → vi .bashrc

  alias mongod='/root/mongodb/bin/mongod'
  alias mongo='/root/mongodb/bin/mongo'

  source .bashrc

```
[root@localhost ~]# cat .bashrc
# .bashrc

# User specific aliases and functions

alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
alias mongod='/root/mongodb/bin/mongod'
alias mongo='/root/mongodb/bin/mongo'

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
[root@localhost ~]# source .bashrc
```

4. 설치 및 실행

4.3.1 Start MongoDB daemon

- MongoDB 데몬 실행
- root계정에서 실행
- --config 옵션을 사용하여 config file의 설정을 적용한 데몬 실행

  → mongod --config /etc/mongod.conf

```
[root@localhost ~]# mongod --config /etc/mongod.conf
2018-10-15T17:00:01.322+0900 I CONTROL  [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify
about to fork child process, waiting until server is ready for connections.
forked process: 2900
child process started successfully, parent exiting
```

4. 설치 및 실행

4.3.2 Begin using MongoDB

- 정상적으로 데몬이 구동되었다면 MongoDB 접속
- root계정에서 실행

  → mongo

```
[root@localhost ~]# mongo
MongoDB shell version v4.0.3
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("ef7da849-5d77-45b0-81e3-0869906a429b") }
MongoDB server version: 4.0.3
Server has startup warnings:
2018-10-15T17:00:01.480+0900 I STORAGE  [initandlisten]
2018-10-15T17:00:01.480+0900 I STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2018-10-15T17:00:01.480+0900 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/prodnotes-filesystem
2018-10-15T17:00:02.287+0900 I CONTROL  [initandlisten]
2018-10-15T17:00:02.287+0900 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-10-15T17:00:02.287+0900 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2018-10-15T17:00:02.287+0900 I CONTROL  [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2018-10-15T17:00:02.287+0900 I CONTROL  [initandlisten]
2018-10-15T17:00:02.287+0900 W CONTROL  [initandlisten]
2018-10-15T17:00:02.287+0900 W CONTROL  [initandlisten]
2018-10-15T17:00:02.287+0900 I CONTROL  [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

>
```

5. 기능소개

세부 목차

5. 기능소개

5.1.1 Database 생성 및 조회

- Database 생성
  - use database_name 으로 생성
  - 1개 이상의 collection이 존재해야 database 리스트에서 확인
- Database 조회
  - show dbs : database 리스트 확인
  - db : 현재 사용 중인 database 확인
  - db.stats() : database 상태 확인

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
>
>
> use testDB
switched to db testDB
>
> db.createCollection("testCollection")
{ "ok" : 1 }
>
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
testDB  0.000GB
>
> db
testDB
```

```
> db.stats()
{
        "db" : "testDB",
        "collections" : 1,
        "views" : 0,
        "objects" : 0,
        "avgObjSize" : 0,
        "dataSize" : 0,
        "storageSize" : 4096,
        "numExtents" : 0,
        "indexes" : 1,
        "indexSize" : 4096,
        "fsUsedSize" : 7936765952,
        "fsTotalSize" : 44254150656,
        "ok" : 1
}
```

5. 기능소개

5.1.2 Database 제거

- db.dropDatabase()로 database 제거
- use로 해당 database에 스위치하고 나서 실행해야 제거

```
> use testDB
switched to db testDB
> db.dropDatabase()
{ "dropped" : "testDB", "ok" : 1 }
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

5. 기능소개

5.2.1 Collection 생성

- db.createCollection(name, [option]) 으로 collection 생성
- name은 collection 이름이고, option은 document 타입으로 구성된 해당 collection의 설정값
- option 객체의 속성들
  - capped : Boolean타입, 이 값을 true로 설정하면 capped collection을 활성화, Capped collection 이란 고정된 크기를 가진 collection으로, size가 초과되면 가장 오래된 데이터를 덮어씀, 이 값을 true로 설정하면 size 값을 꼭 설정
  - size : number 타입, capped collection을 위해 해당 collection 최대 사이즈를 ~bytes로 지정
  - max : number 타입, 해당 collection에 추가할 수 있는 최대 document 개수를 설정

```
> use testDB
switched to db testDB
>
> db.createCollection("emp", {
... capped : true,
... size : 6142800,
... max : 10000
... })
{ "ok" : 1 }
>
```

5. 기능소개

5.2.2 Collection 조회 및 제거

- Collection 조회

  - show collections로 collection 리스트 확인

- Collection 제거

  - db.collection명.drop()으로 collection 제거

```
> show collections
emp
testCollection
>
> db.emp.drop()
true
>
> show collections
testCollection
>
```

5. 기능소개

5.3.1 Document 생성

- Document 생성

  - db.collection명.insert(document)로 document 추가
  - 배열형식으로 전달하면 여러 document를 bulk 형식으로 추가

```
> use testDB
switched to db testDB
>
> db.emp.insert([
... {"name" : "jane", "age" : 20},
... {"name" : "john", "nationality" : "Korea"}
... ]);
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 2,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```

5. 기능소개

5.3.2 Document 조회(2/1)

- Document 조회

  - db.collection명.find([query, projection]) 로 collection의 document 리스트 확인
  - 한 줄이 너무 길어 불편할 때는 끝에 .pretty()를 붙이면 사용자가 보기 편하게 결과 출력

```
> db.emp.find()
{ "_id" : ObjectId("5bc5a9cc1c5e1becb5f8002f"), "name" : "jane", "age" : 20 }
{ "_id" : ObjectId("5bc5a9cc1c5e1becb5f80030"), "name" : "john", "nationality" : "Korea" }
>
> db.emp.find().pretty()
{
        "_id" : ObjectId("5bc5a9cc1c5e1becb5f8002f"),
        "name" : "jane",
        "age" : 20
}
{
        "_id" : ObjectId("5bc5a9cc1c5e1becb5f80030"),
        "name" : "john",
        "nationality" : "Korea"
}
>
```

5. 기능소개

5.3.2 Document 조회(2/2)

- Document 조회

  - db.collection명.find([query, projection])의 매개변수로 아래와 같은 것들이 들어갈 수 있음

    → query : document 타입, optional이며, document를 조회할 때 기준을 정함, 기준이 없이 collection에 있는 모든 document를 조회할 때는 이 매개변수를 비우거나, {} 전달

    → projection : document 타입, optional이며, document를 조회할 때 보여질 field 정함

```
> db.emp.find( { } , { "_id" : false, "name" : true, "nationality" : true })
{ "name" : "jane" }
{ "name" : "john", "nationality" : "Korea" }
>
> db.emp.find({"name" : "john"})
{ "_id" : ObjectId("5bc5a9cc1c5e1becb5f80030"), "name" : "john", "nationality" : "Korea" }
>
```

5. 기능소개

5.3.3 Document 제거

- Document 제거

  - db.collection명.remove(criteria[, justOne])로 document 제거
  - 매개변수로 들어가는 객체의 속성들은 아래와 같음
    → criteria : document 타입, 데이터의 기준 값으로 일치하면 기본적으로 다 삭제, 이 값이 { } 이면 collection의 모든 데이터를 제거, 반드시 넣어야 함
    → justOne : Boolean 타입, optional 매개변수이며, 이 값이 true이면 1개의 document만 제거, 이 매개변수가 생략되면 기본값을 false이고, criteria에 해당되는 모든 document 제거

```
> db.emp.remove({ "name" : "john" })
WriteResult({ "nRemoved" : 1 })
>
> db.emp.find()
{ "_id" : ObjectId("5bc5a9cc1c5e1becb5f8002f"), "name" : "jane", "age" : 20 }
> db.emp.find()
{ "_id" : ObjectId("5bc5a9cc1c5e1becb5f8002f"), "name" : "jane", "age" : 20 }
>
> db.emp.remove({ })
WriteResult({ "nRemoved" : 1 })
>
> db.emp.find()
>
```

5. 기능소개

5.4.1 MongoDB Backup

- MongoDB Backup

  - MongoDB에서 backup / restore 방식이 크게 2가지

    1. 특정한 node에서 물리적인 장애가 발생할 경우(disk crash 등) 서비스 중인 다른 secondary node 에서 데이터 파일을 물리적으로 copy하여 backup하는 방법
    2. mongodump 커맨드를 이용하여 주기적으로 backup하여 file을 보관하고, 장애가 발생할 경우 dump file을 이용하여 restore하는 방법, mongodump는 MongoDB 설치경로/bin에 있음

    → backup 명령어

    mongodump --host "IP" --port "port번호" --db "백업할 DB이름" -o "백업파일 생성할 경로"

    ex] /root/mongodb/bin/mongodump --host "192.168.71.101:27017" -o /root/mongodb/backup

    → 현재 MongoDB에 있는 모든 DB를 backup하는 명령

    replica set으로 구축한 경우에는 backup 수행시 성능에 영향을 미치기 때문에 secondary node에서 수행 특정 DB만을 backup 받으려면 –db 옵션을 사용하면 되고, DB 전체를 backup 받으려면 생략

5. 기능소개

5.4.2 MongoDB Restore

- MongoDB Restore

  - mongorestore 커맨드로 backup dump file을 이용하여 restore, mongorestore는 MongoDB설치 경로/bin에 있음

    → restore 명령어

    /root/mongodb/bin/mongorestore --db DB명 "백업파일경로"

    ex] /root/mongodb/bin/mongorestore --drop /root/mongodb/backup

    → 현재 모든 DB를 restore하는 명령

    --drop 옵션은 복구할 때, 기존에 동일한 collection이 존재할 경우 삭제하고 할 것인지에 대한 설정, 그리고 backup과 마찬가지로 특정 DB만 선택적으로 복구를 하려면 --db 옵션 사용

5.5 MongoDB 종료

- MongoDB 종료

  - ctrl + c 로 종료하면 불완전한 종료이므로, mongo shell에서 데몬 종료

```
> use admin
switched to db admin
>
> db.shutdownServer()
server should be down...
2018-10-17T11:13:14.126+0900 I NETWORK  [js] trying reconnect to 127.0.0.1:27017 failed
2018-10-17T11:13:14.126+0900 I NETWORK  [js] reconnect 127.0.0.1:27017 failed failed
>
[root@localhost ~]# ps -ef | grep mongo
root      3189  2561  0 11:13 pts/0    00:00:00 grep --color=auto mongo
[root@localhost ~]#
```

6. 활용예제

세부 목차

1. Replication 정의
2. Replica set 구축
   1. 방화벽 설정
   2. config file 생성
   3. replica set 구성
   4. replica set 상태 확인
   5. bulk data insert
   6. auto-failover 동작 확인
3. Sharding 정의

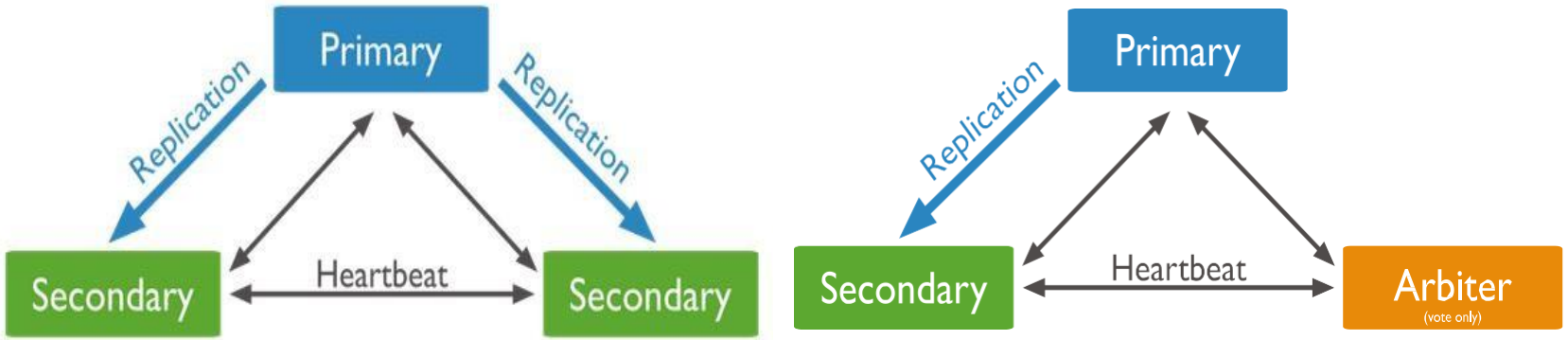
6. 활용예제

6.1 Replication 정의

- Replication

- MongoDB에서는 기본적으로 H/A를 위해 Replica Set 구성의 Replication 제공, 이로 인해, 데이터의 일관성 및 안정성과 높은 가용성 보장
- Read에 대한 높은 성능 보장
- Replica Set의 경우, 기본적으로 Master-Slave와 같은 동작 방식을 보여주지만, 'Arbiter'등을 이용하여 사용자 임의로 새로운 primary를 정하는 것이 가능, 또한 재해 복구를 위한 Auto fail-over 기능 지원
- Replica set의 경우 최소 3 node(1 primary, 2 secondary)로 구성하는 것을 기본적으로 권고하며, 'Arbiter' 라는 secondary node를 이용하여 구성하는 것 가능

  (단, Arbiter node는 별도의 데이터 파일을 가지고 있지 않음)



6. 활용예제

6.2.1 방화벽 설정

- 방화벽 설정

  - 3개의 node로 replication 구축을 테스트하기 위해 node 간 통신을 주고 받으려면 시스템 방화벽 차단
  - root계정에서 실행

    → systemctl stop firewalld.service
    → systemctl disable firewalld.service

  - 모든 node 적용
  - 단, 예제에서는 단순 구축 및 구동 테스트를 위하여 시스템 방화벽을 해제한 것이므로, 실제 운영상에서는 보안 등의 문제로 방화벽을 활성화해야 하므로 노드간 통신을 위한 IP 및 포트 설정 필요

```
[root@localhost ~]# systemctl stop firewalld.service
[root@localhost ~]# systemctl disable firewalld.service
Removed symlink /etc/systemd/system/basic.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@localhost ~]#
[root@localhost ~]# systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)

Oct 18 10:17:30 localhost.localdomain systemd[1]: Starting firewalld - dynamic firewall daemon...
Oct 18 10:17:30 localhost.localdomain systemd[1]: Started firewalld - dynamic firewall daemon.
Oct 18 10:44:09 localhost.localdomain systemd[1]: Stopping firewalld - dynamic firewall daemon...
Oct 18 10:44:09 localhost.localdomain systemd[1]: Stopped firewalld - dynamic firewall daemon.
```

6. 활용예제

6.2.2 config file 생성

- config file 생성

  - replica set으로 구성하기 위해서 설정 파일 생성

  - root계정에서 실행

  → systemLog:
  destination: file
  logAppend: true
  path: /root/mongodb/log/replica.log

  storage:
  dbPath: /root/mongodb/data/
  journal:
  enabled: true

  processManagement:
  fork: true # fork and run in background
  pidFilePath: /root/mongodb/mongod.pid
  timeZoneInfo: /usr/share/zoneinfo

  # network interfaces
  net:
  port: 27017
  bindIp: 0.0.0.0

  replication: replSet
  Name: "rs1"

  - 모든 node에 적용

```
[root@localhost ~]# cat /etc/mongod.conf
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /root/mongodb/log/replica.log

# Where and how to store data.
storage:
  dbPath: /root/mongodb/data/
  journal:
    enabled: true
#  engine:
#  mmapv1:
#  wiredTiger:

# how the process runs
processManagement:
  fork: true  # fork and run in background
  pidFilePath: /root/mongodb/mongod.pid  # location of pidfile
  timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0  # Listen to local interface only, comment to liste

replication:
  replSetName: "rs1"
```

6. 활용예제

6.2.3 replica set 구성

- replica set 구성

  - MongoDB 데몬을 띄우고 mongoshell에 접속하여 replica set 구성

  → rs.initiate(
{
_id : "rs1",
members: [
{_id : 0, host : "192.168.71.107:27017"},
{_id : 1, host : "192.168.71.106:27017"},
{_id : 2, host : "192.168.71.108:27017"}
]
})

```
> rs.initiate(
... {
... _id : "rs1",
... members: [
... {_id : 0, host : "192.168.71.107:27017"},
... {_id : 1, host : "192.168.71.106:27017"},
... {_id : 2, host : "192.168.71.108:27017"}
... ]
... })
{
        "ok" : 1,
        "operationTime" : Timestamp(1539829560, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1539829560, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
```

  - 하나의 node에만 적용

6. 활용예제

6.2.4 replica set 상태 확인

- replica set 상태 확인

  - rs.status() 명령어를 통해 현재 replica set 구성 상태 확인
  - replica set 구성을 하고 일정 시간이 지나면 >에서 replica_set_name:[primary, secondary]>로 변경 확인

```
rs1:PRIMARY> rs.status()
{
        "set" : "rs1",
        "date" : ISODate("2018-10-18T02:48:11.485Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "syncingTo" : "",
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1539830880, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1539830880, 1),
                        "t" : NumberLong(1)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1539830880, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1539830880, 1),
                        "t" : NumberLong(1)
                }
        },
```

```
rs1:SECONDARY>
```

```
rs1:SECONDARY>
```

6. 활용예제

6.2.5 bulk data insert

- bulk data insert

  - 동기화가 제대로 되는지 확인하기 위해서 primary node에 대량의 데이터 저장
  - secondary node에서 데이터가 동기화 확인

```
rs1:PRIMARY> for (i=1; i<=100000; i=i+1) { db.testCollection.insert( { x : i } ) }
WriteResult({ "nInserted" : 1 })
rs1:PRIMARY>
```

```
rs1:PRIMARY> db.testCollection.count()
100000
rs1:PRIMARY>
```

```
rs1:SECONDARY> rs.slaveOk()
rs1:SECONDARY>
rs1:SECONDARY> use test
switched to db test
rs1:SECONDARY>
rs1:SECONDARY> db.testCollection.count()
100000
rs1:SECONDARY>
```

6. 활용예제

6.2.6 auto-failover 동작 확인

- auto-failover 동작 확인

  - primary node의 mongodb 데몬을 종료한 후 secondary node가 primary로 auto-failover확인

```
rs1:PRIMARY> use admin
switched to db admin
rs1:PRIMARY>
rs1:PRIMARY> db.shutdownServer()
server should be down...
2018-10-18T16:05:00.505+0900 I NETWORK  [js] trying reconnect to 127.0.0.1:27017 failed
2018-10-18T16:05:00.724+0900 I NETWORK  [js] reconnect 127.0.0.1:27017 failed failed
2018-10-18T16:05:00.730+0900 I NETWORK  [js] trying reconnect to 127.0.0.1:27017 failed
2018-10-18T16:05:00.731+0900 I NETWORK  [js] reconnect 127.0.0.1:27017 failed failed
[root@localhost ~]# mongod --config /etc/mongod.conf
2018-10-18T16:05:46.064+0900 I CONTROL  [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify
about to fork child process, waiting until server is ready for connections.
forked process: 7090
child process started successfully, parent exiting
[root@localhost ~]#
[root@localhost ~]# mongo
MongoDB shell version v4.0.3
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("f518aba1-2883-4c4c-b82b-75f7b54e48d3") }
MongoDB server version: 4.0.3
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

rs1:SECONDARY>
```

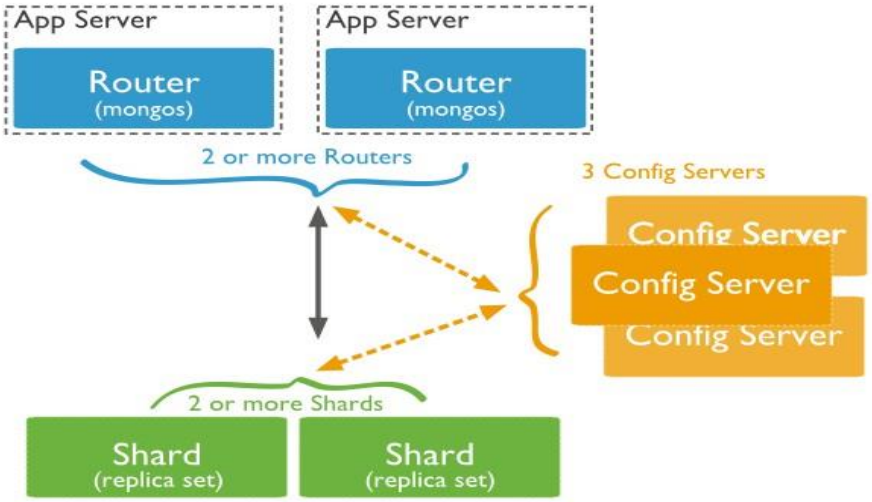
```
rs1:SECONDARY>
rs1:PRIMARY>
```

6. 활용예제

6.3 Sharding 정의

- Sharding

  - Automatic Sharding mechanism을 이용하여 수평 확장(Scale Out) 가능
  - Scale Out을 통한 데이터 분산 처리 가능
  - Write에 대한 높은 성능 보장
  - Sharding 구성 환경에서, Query를 분석해서 해당하는 노드들로 리다이렉트 시켜주는 Query Route(Mongos)라는 특별한 구성 요소(노드) 필요
  - 메타 데이터를 저장하고 있는 Config 서버는 3개의 노드가 별도로 필요로 하며, 이 Config 노드는 다른 노드들과 물리적으로 별도 구성 권장



7. FAQ

Q Collection schema를 정의하거나 변경하려면 어떻게 해야 합니까?

A MongoDB에서 Collection을 위한 schema를 지정할 필요가 없습니다. 즉, 단일 collection의 document에는 동일한 field set가 있을 필요가 없습니다. collection의 document 구조를 변경하려면 document를 새 구조로 업데이트하십시오. 예를 들어, 새 필드를 추가하거나 기존 필드를 제거하거나 필드 값을 새 유형으로 업데이트할 수 있습니다.

Q MongoDB는 트랜잭션을 지원합니까?

A MongoDB의 원자적 single-document 작업은 이미 대다수의 application의 데이터 무결성 요구를 충족시키는 트랜잭션을 제공합니다. multiple sub-document 및 배열의 업데이트를 포함하여 하나 이상의 필드가 단일 작업으로 작성될 수 있습니다.

MongoDB는 document가 업데이트될 때 완벽한 격리를 보장합니다. 오류로 인해 작업이 롤백되어 클라이언트가 문서의 일관된 보기를 수신하게 됩니다. MongoDB는 replica set을 위한 multi-document 트랜잭션을 제공하고, MongoDB 4.2부터는 sharded cluster를 위한 트랜잭션 지원이 예정되어 있습니다.

8. 용어정리

용어	설 명
Collection	Collection은 MongoDB Document 그룹, RDBMS의 table과 비슷한 개념이지만, schema를 따로 가지고 있지 않음
Document	RDBMS의 record와 비슷한 개념, 데이터 구조는 한 개 이상의 key-value 쌍으로 이루어져 있음, Document끼리 다른 schema를 가지고 있을 수 있음, 즉, 서로 다른 데이터들을 가지고 있을 수 있음
HA	서버, 네트워크, 프로그램 등의 정보 시스템이 상당히 오랜 기간동안 지속적으로 정상 운영이 가능한 성질을 의미
Transaction	하나의 논리적인 기능을 수행하기 위한 작업의 단위, 데이터베이스 시스템은 각각의 트랜잭션에 대해 원자성, 일관성, 격리성, 영속성을보장
Atomicity(원자성)	트랜잭션이 데이터베이스에 모두 반영되던가, 아니면 전혀 반영되지 않아야 함
Consistency(일관성)	트랜잭션이 실행을 성공적으로 완료하면 언제나 일관성 있는 데이터베이스 상태로 변환, 즉 트랜잭션 실행의 결과로 데이터베이스 상태가 모순되지 않음
Isolation(격리성)	둘 이상의 트랜잭션이 동시에 병행 실행되고 있을 경우에 어느 하나의 트랜잭션이라도 다른 트랜잭션의 연산에 끼어들 수 없음
Durability(영속성)	트랜잭션이 일단 실행을 성공적으로 완료하면 그 결과는 영속적임, 따라서 시스템은 어떤 경우에도 완료된 결과의 영속성을 보장

Open Source Software Installation & Application Guide