mongodb 的安装和配置

# mongo的安装

## 下载，解压mongodb文件。

把解压的文件放在/opt/下面（存放目录用自己习惯的位置）

wget <http://fastdl.mongodb.org/linux/mongodb-linux-x86_64-2.6.9.tgz>

mv mongodb-linux-x86\_64-2.6.9.tgz /opt

tar -zxvf mongodb-linux-x86\_64-2.6.9.tgz

mv mongodb-linux-x86\_64-2.6.9 mongodb2.6.9（重命名）

## 创建mongodb存放的数据文件、日志文件

/mongodb/mongodb\_db(存放数据)

/mongodb/mongodb\_logs(存放日志文件)

mkdir -p /mongodb/mongodb\_db

mkdir -p /mongodb/mongodb\_logs

## 启动mongodb服务

第一步：cd /opt/mongodb2.6.9/bin

第二步：./mongod --dbpath=/mongodb/mongodb\_db --logpath=/mongodb/mongodb\_logs/mongodb.log --logappend&

//执行完会出现:

all output going to: /mongodbdata/mongodb\_logs/mongodb.log

## 检查服务是否启动mongodb（服务的端口为：27017）

# netstat -lanp  | grep 27017

tcp 0 0 0.0.0.0:27017 0.0.0.0:\* LISTEN 6509/ ./mongod

unix 2 [ ACC ] STREAM LISTENING 2068952 6509/./mongod /tmp/mongodb-27 017.sock

出现以上提示说明启动成功。

## 通过mongodb客服端程序连接数据库

数据库客户端：/opt/mongodb2.6.9/bin/mongo

/opt/mongodb2.6.9/bin/mongo

MongoDB shell version: 2.6.9

connecting to: test

> use mongo\_test //创建库

switched to db mongo\_test

> db.createCollection("test")//创建库

{ "ok" : 0, "errmsg" : "collection already exists" }

# 配置mongodb

## 创建并且配置mongodb.conf在安装目录

***verbose***：日志信息冗余。默认false。提高内部报告标准输出或记录到logpath配置的日志文件中。要启用verbose或启用verbosity 用vvvv参数，

启动verbose冗长信息，它的级别有 vv~vvvvv，v越多级别越高，在日志文件中记录的信息越详细。如：

**verbose = true  
vvvv = true**

***port***：端口默认27017，MongoDB的默认服务TCP端口，监听客户端连接。要是端口设置小于1024，比如1021，则需要root权限启动，不能用mongodb帐号启动，（普通帐号即使是27017也起不来）否则报错：[mongo --port=1021 连接]

ERROR: listen(): bind() failed errno:13 Permission denied for socket: 127.0.0.1:1021

**port = 27017**

***bind\_ip***：绑定地址。默认127.0.0.1，只能通过本地连接。进程绑定和监听来自这个地址上的应用连接。要是需要给其他服务器连接，则需要注释掉这个或则把IP改成本机地址，如192.168.200.201[其他服务器用 mongo --host=192.168.200.201 连接] ，可以用一个逗号分隔的列表绑定多个IP地址。

**bind\_ip = 127.0.0.1**

***maxConns***：最大连接数。默认值：取决于系统（即的ulimit和文件描述符）。MongoDB中不会限制其自身的连接。当设置大于系统的限制，则无效，以系统限制为准。这对于客户端创建很多“表”，允许连接超时而不关闭“表”的时候很有用。设置该值的高于连接池和总连接数的大小，以防止尖峰时候的连接。注意：不能设置该值大于20000。

**maxConns = 100**

***objcheck***:强制验证客户端请求。2.4的默认设置为objcheck成为true，在早期版本objcheck默认为false。因为它强制验证客户端请求，确保客户端绝不插入无效文件到数据库中。对于嵌套文档的对象，会有一点性能影响。设置noobjcheck 关闭。

**objcheck = true**

**noobjcheck = false**

***logpath***：指定日志文件，该文件将保存所有的日志记录、诊断信息。除非另有指定，mongod将所有的日志信息输出到标准输出。如果没有指定logappend，重启则日志会进行覆盖操作。

**logpath** =**/mnt/mongodb\_logs/mongodb.log**

***logappend***：写日志的模式：设置为true为追加。默认是覆盖。如果未指定此设置，启动时MongoDB的将覆盖现有的日志文件。

**logappend=true**

***syslog***：日志输出都发送到主机的syslog系统，而不是标准输出到logpath指定日志文件。syslog和logpath不能一起用，会报错：.Cant use both a logpath and syslog。

**syslog = true**

***pidfilepath***：进程ID，没有指定则启动时候就没有PID文件。默认缺省。

**pidfilepath = /var/run/mongo.pid**

***keyFile***：指定存储身份验证信息的密钥文件的路径。默认缺省。详情见："

word-spacing: 0px; display: inline; white-space: normal; orphans: 2;

widows: 2; font-size-adjust: none; font-stretch: normal; background-color:

#ffffff; -webkit-text-size-adjust: auto; -webkit-text-stroke-width:

0px;">Replica Set Security" and “Replica Set Administration.”

**keyFile = /srv/mongodb/keyfile**

***nounixsocket***：套接字文件，默认为false，有生成socket文件。当设置为true时，不会生成socket文件。

**nounixsocket = false**

***unixSocketPrefix***：套接字文件路径，默认/tmp

**unixSocketPrefix = /tmp**

***fork***：是否后台运行，设置为true 启动 进程在后台运行的守护进程模式。默认false。

**fork = true**

***auth***：用户认证，默认false。不需要认证。当设置为true时候，进入数据库需要auth验证，当数据库里没有用户，则不需要验证也可以操作。直到创建了第一个用户，之后操作都需要验证。

**auth = true**

**noauth = false**

比如：通过db.addUser('sa','sa')  在admin库下面创建一个超级用户，只能在在admin库下面先认证完毕了：ab.auth('sa','sa') ，才能去别的库操作，不能在其他库验证。这样连接数据库也需要指定库：

1.mongo -usa -psa admin     #sa 帐号连接admin

1.mongo -uaa -paa test      #aa 帐号连接test

***cpu***：设置为true会强制mongodb每4s报告cpu利用率和io等待，把日志信息写到标准输出或日志文件。默认为false。开启日志会出现：Mon Jun 10 10:21:42.241 [snapshotthread] cpu: elapsed:4000  writelock: 0%

**cpu = true**

***dbpath***：数据存放目录。默认： word-spacing: 0px; white-space: normal; orphans: 2; widows: 2; font-size-adjust: none; font-stretch: normal; background-color: #ffffff; -webkit-text-size-adjust: auto; -webkit-text-stroke-width: 0px;">/data/db/

**dbpath=/mnt/mongodb\_db**

**diaglog**：创建一个非常详细的故障排除和各种错误的诊断日志记录。默认0。设置为1，为在dbpath目录里生成一个diaglog.开头的日志文件，他的值如下：

1.Value Setting

2.0 off. No logging. #关闭。没有记录。

3.1 Log write operations. #写操作

4.2 Log read operations. #读操作

5.3 Log both read and write operations. #读写操作

6.7 Log write and some read operations. #写和一些读操作

***directoryperdb***：设置为true，修改数据目录存储模式，每个数据库的文件存储在DBPATH指定目录的不同的文件夹中。使用此选项，可以配置的MongoDB将数据存储在不同的磁盘设备上，以提高写入吞吐量或磁盘容量。默认为false。  
注意：要是在运行一段时间的数据库中，开启该参数，会导致原始的数据都会消失（注释参数则会回来）。因为数据目录都不同了，除非迁移现有的数据文件到directoryperdb产生的数据库目录中，如：  
root@m3:/var/lib/mongodb# mv test.\* test/  
把test数据文件迁移到directoryperdb产生的数据库test目录中。 所以需要在规划好之后确定是否要开启。

**directoryperdb = ture**

原始数据结构：  
02.journal  
03.mongod.lock  
04.local.0  
05.local.1  
06.local.ns  
07.test.0  
08.test.1  
09.test.ns  
  
开启 directoryperdb，并把数据文件迁移到相关的数据目录后的结构：  
13.journal  
14.mongod.lock  
15.local/local.0  
16.local/local.1  
17.local/local.ns  
18.test/test.0  
19.test/test.1  
20.test/test.ns

***journal***：日志，（redo log，更多的介绍请看这里和这里）  
默认值：（在64位系统）true。默认值：（32位系统）false。设置为true，启用操作日志，以确保写入持久性和数据的一致性，会在dbpath目录下创建journal目录。设置为false，以防止日志持久性的情况下，并不需要开销。为了减少磁盘上使用的日志的影响，您可以启用nojournal，并设置为true。  
注意：在64位系统上禁用日志必须使用带有nojournal的。

**journal=truejournal=true/false**

**nojournal = true/false**

***journalCommitInterval***：刷写提交机制，默认是30ms或则100ms。较低的值，会更消耗磁盘的性能。此选项接受2和300毫秒之间的值：如果单块设备提供日志和数据文件，默认的日记提交时间间隔为100毫秒。如果不同的块设备提供的日志和数据文件，默认的日记提交的时间间隔为30毫秒。

**journalCommitInterval=100**

***ipv6***：是否支持ipv6，默认false。

***jsonp***：是否允许JSONP访问通过一个HTTP接口，默认false。

***nohttpinterface***：是否禁止http接口，即28017 端口开启的服务。

默认 false，支持。**nohttpinterface = false**

***noprealloc***：预分配方式。默认false：使用预分配方式来保证写入性能的稳定，预分配在后台进行，并且每个预分配的文件都用0进行填充。这让MongoDB始终保持额外的空间和空余的数据文件，从而避免了数据增长过快而带来的分配磁盘空间引起的阻塞。设置noprealloc= true来禁用预分配的数据文件，会缩短启动时间，但在正常操作过程中，可能会导致性能显著下降。

**noprealloc = false**

***noscripting***：是否禁止脚本引擎。默认是false：不禁止。ture：禁止  
要是设置成true：运行一些脚本的时候会出现：

1.JavaScript execution failed: group command failed: { "ok" : 0, "errmsg" : "server-side JavaScript execution is disabled" }

**noscripting = true/false**

***notablescan***：是否禁止表扫描操作。默认false：不禁止，ture：禁止

禁止要是执行表扫描会出现：error: { "$err" : "table scans not

allowed:test.emp", "code" : 10111 }可以动态修改设置：

db.adminCommand({setParameter:1, notablescan:false})

***notablescan = true/false***

***nssize***:命名空间的文件（即NS）的默认大小，默认16M，最大2G。  
所有新创建的默认大小命名空间的文件（即NS）。此选项不会影响现有的命名空间的文件的大小。默认值是16M字节，最大大小为2 GB。让小数据库不让浪费太多的磁盘空间，同时让大数据在磁盘上有连续的空间。

1.-rwxrwxrwx 1 mongodb zhoujy  16M  6月 11 14:44 test.0  
2.-rwxrwxrwx 1 mongodb zhoujy  32M  6月  1 21:36 test.1  
3.-rwxrwxrwx 1 mongodb zhoujy  16M  6月 11 14:44 test.ns  
4.drwxr-xr-x 2 root    root   4.0K  6月 10 11:57 \_tmp

nssize  = 16

***quota***：配额，默认false。是否开启配置每个数据库的最多文件数的限制。当为true则用quotaFiles来配置最多文件的数量。

**quota = true**

***quotaFiles***：配额数量。每个数据库的数据文件数量的限制。此选项需要quota为true。默认为8。

**quotaFiles = 8**

***rest***： 默认false，设置为true，使一个简单的 REST API。

**rest = true**

设置为true，开启后，在MongoDB默认会开启一个HTTP协议的端口提供REST的服务（nohttpinterface = false），这个端口是你Server端口加上1000，即28017，默认的HTTP端口是数据库状态页面，（开启后，web页面的Commands 行中的命令都可以点进去）。mongodb自带的REST，不支持 增、删、改，同时也不支持 权限认证。

***repair***：修复数据库操作，默认是false。  
设置为true时，启动后修复所有数据库，设置这个选项最好在命令行上，而不是在配置文件或控制脚本。如：  
命令行修复：mongod –repair启动时修复，需要关闭journal

repairpath：修复路径，默认是在dbpath路径下的\_tmp 目录

***smallfiles***：是否使用较小的默认文件。默认为false，不使用。设置为true，使用较小的默认数据文件大小。smallfiles减少数据文件的初始大小，并限制他们到512M，也减少了日志文件的大小，并限制他们到128M。如果数据库很大，各持有少量的数据，会导致mongodb创建很多文件，会影响性能。

**smallfiles = true**

***syncdelay***：刷写数据到日志的频率，通过fsync操作数据。默认60秒。

**syncdelay = 60**

默认就可以，不需要设置。不会对日志文件（journal files）有影响

警告：如果设置为0，SYNCDELAY 不会同步到磁盘的内存映射文件。在生产系统上，不要设置这个值。

***sysinfo***：系统信息，默认false。设置为true，mongod会诊断系统有关的页面大小，数量的物理页面，可用物理??页面的数量输出到标准输出。

1.Tue Jun 11 21:07:15.031 sysinfo:  
2.Tue Jun 11 21:07:15.035   page size: 4096  
3.Tue Jun 11 21:07:15.035   \_SC\_PHYS\_PAGES: 256318  
4.Tue Jun 11 21:07:15.035   \_SC\_AVPHYS\_PAGES: 19895

当开启sysinfo参数的时候，只会打印上面的信息，不会启动mongodb的程序。所以要关闭该参数，才能开启mongodb。

***upgrade***:升级。默认为false。当设置为true，指定DBPATH，升级磁盘上的数据格式的文件到最新版本。会影响数据库操作，更新元数据。大部分情况下，不需要设置该值。

***traceExceptions***：是否使用内部诊断。默认false。

***quiet***：安静模式。quiet = true

***setParameter***：2.4的新参数，指定启动选项配置。想设置多个选项则用一个setParameter选项指定，可以setParameter的参数请见这里，详情请见这里  
声明setParameter设置在这个文件中，使用下面的格式：

**setParameter = <parameter>=<value>**

***Replication Options  复制选项***

* **replSet**：使用此设置来配置复制副本集。指定一个副本集名称作为参数，所有主机都必须有相同的名称作为同一个副本集。
* **oplogSize**：指定的复制操作日志（OPLOG）的最大大小。mongod创建一个OPLOG的大小基于最大可用空间量。对于64位系统，OPLOG通常是5％的可用磁盘空间。一旦mongod第一次创建OPLOG，改变oplogSize将不会影响OPLOG的大小。
* **fastsync**：默认为false。在副本集下，设置为true，从一个dbpath里启用从库复制服务，该dbpath的数据库是主库的快照，可用于快速启用同步，否则的mongod将尝试执行初始同步。注意：如果数据不完全同步，mongod指定fastsync开启，secondary或slave与主永久不同步，这可能会导致显着的一致性问题。
* **replIndexPrefetch**：2.2版本出现的新参数，默认是all。可以设置的值有：all, none, and \_id\_only。只能在副本集（replSet）中使用。默认情况下，secondary副本集的成员将加载所有索引到内存中（从OPLOG之前的操作有关的）。您可以修改此行为，使secondary只会加载\_id索引。指定\_id\_或none，防止mongod的任何索引加载到内存。

***Master/Slave Replication：主从复制的相关设置***

* master：默认为false，当设置为true，则配置当前实例作为主实例。

master = true

* slave：默认为false，当设置为true，则配置当前实例作为从实例。

slave = true

* source：默认为空，格式为：<host><:port>。用于从实例的复制：设置从的时候指定该选项会让从复制指定主的实例

source = 127.0.0.1:30001

* only：默认为空，用于从选项，指定一个数据库进行复制。

only = abc          #只同步abc集合（库）

* slavedelay：设置从库同步主库的延迟时间，用于从设置，默认为0。

slavedelay = 60     #延迟60s同步主数据

* autoresync：默认为false，用于从设置。是否自动重新同步。设置为true，如果落后主超过10秒，会强制从自动重新同步。如果oplogSize太小，此设置可能有问题。如果OPLOG大小不足以存储主的变化状态和从的状态变化之间的差异，这种情况下强制重新同步是不必要的。当设置autoresync选项设置为false，10分钟内从不会进行大于1次的自动重新同步。

autoresync = false

**设置MongoDB数据库**

cd /opt/mongodb2.6.9/ #进入MongoDB安装目录

touch mongodb.conf //创建mongodb配置文件

vi mongodb.conf  #编辑

port=27017 #端口号

dbpath=/mongodb/mongodb\_db #数据库路径

logpath=/mongodb/mongodb\_logs/mongodb.log #日志输出文件路径

fork=true #设置后台运行

logappend=true #日志输出方式

maxConns=10000

## 设置开机启动MongoDB

./mongod --config /opt/mongodb2.6.9/mongodb.conf  #启动MongoDB

/opt/mongodb2.6.9/bin/mongo 127.0.0.1:27017/admin --eval "db.shutdownServer()"#关闭MongoDB

vi /etc/rc.d/init.d/mongod #设置开机启动MongoDB

ulimit -SHn 655350

#!/bin/sh

# chkconfig: - 64 36

# description:mongod

case $1 in

start)

/opt/mongodb2.6.9/bin/mongod  --config /opt/mongodb2.6.9/mongodb.conf

;;

stop)

/opt/mongodb2.6.9/bin/mongo 127.0.0.1:27017/admin --eval "db.shutdownServer()"

;;

status)

/opt/mongodb2.6.9/bin/mongo 127.0.0.1:27017/admin --eval "db.stats()"

;;

esac

chmod +x /etc/rc.d/init.d/mongod #添加脚本执行权限

chkconfig --add mongod

chkconfig mongod on #设置开机启动

service mongod start #启动MongoDB



## 添加环境变量

vi /etc/profile

export PATH=$PATH:/opt/mongodb2.6.9/bin

source /etc/profile #使配置立即生效

mongo  #进入MongoDB控制台

show dbs #查看默认数据库

use admin  #切换到admin数据库

exit #退出MongoDB控制台

# 用户管理

开启动用户权限认证: auth=true

## 添加管理员账号

|  |
| --- |
| [root@localhost zhangy]# mongo  MongoDB shell version: 2.4.6  connecting to: tank  > use admin          //切换到admin数据库  switched to db admin  > show collections;  system.indexes  system.users          //用户表  > db.system.users.find();   //用户表没有数据  > db.addUser('tank','test');  //添加一个管理员账号  {    "user" : "tank",    "readOnly" : false,    "pwd" : "988432606980d0695e4f668f6bbc643a",    "\_id" : ObjectId("529e5d543b6a4608ac833429")  }  用户只能在用户所在数据库登录，管理员需要通过admin认证后才能管理其他数库  [root@localhost zhangy]# mongo  MongoDB shell version: 2.4.6  connecting to: tank  > show dbs; //显示所有数据库失败，因为还没有认证  Wed Dec 4 06:39:50.925 listDatabases failed:{ "ok" : 0, "errmsg" : "unauthorized" } at src/mongo/shell/mongo.js:46  > db.auth('tank','test'); //认证失败，因为这个用户不属于tank这个数据库  Error: 18 { code: 18, ok: 0.0, errmsg: "auth fails" }  0  > use admin //切换到admin数据库  switched to db admin  > db.auth('tank','test'); //在admin数据库认证成功  1  > use tank; //切换到tank数据库  switched to db tank  > show collections; //不会在提示没有权限了  contact  system.indexes  users |

## 添加普通用启

|  |
| --- |
| > use tank;  switched to db tank  > db.addUser('tank1','test');   //为tank数据库添加了一个可读写用户tank1  {    "\_id" : ObjectId("529e5f8474b4c660718a70f3"),    "user" : "tank1",    "readOnly" : false,    "pwd" : "35dd47abff098f5b4f0b567db8edeac5"  }  > db.addUser('tank2','test',true); //为tank数据库添加了一个只读用户tank2  {    "user" : "tank2",    "readOnly" : true,    "pwd" : "1792916c544d247538ded52e6df7b887",    "\_id" : ObjectId("529e67553992b24438d5e315")  }  > exit  //退出  bye  [root@localhost zhangy]# mongo  MongoDB shell version: 2.4.6  connecting to: tank  > db.auth('tank1','test');  //刚添加的用户可以登录。 |

# 其他

## Mongodb的修复

第一步：mongod --repair 或者

rm -f /mnt/mongodb/mongodb\_db/mongod.lock

第二步启动mongodb：./mongod --dbpath=/mnt/mongodb\_db --logpath=/mnt/mongodb\_logs/mongodb.log --logappend&

## 导入命令

mongoimport -h 172.16.8.189 -d fashy -c orderInfo --type csv --file E:\mongodate\1202.csv --headerline --upsert

* -d    指定把数据导入到哪一个数据库中
* -c    指定把数据导入到哪一个集合中
* --type    指定导入的数据类型
* --file       指定从哪一个文件中导入数据
* --headerline    仅适用于导入csv,tsv格式的数据，表示文件中的第一行作为数据头
* --upsert  以新增或者更新的方式来导入数据

倒流程

mongoexport -u tiku7915 -p U8BncEyc7UkH -h 223.252.223.229:7915  --verbose --authenticationDatabase admin -d tiku -c k12\_question\_circulation  -o k12\_question\_circulation.json

## mongoDB3.0

### 警告信息的处理

**echo "never" > /sys/kernel/mm/transparent\_hugepage/enabled**  
**echo "never" > /sys/kernel/mm/transparent\_hugepage/defrag**

//mongoDB需要配置如下

 ulimit -f unlimited  
   ulimit -t unlimited  
   ulimit -v unlimited  
   ulimit -n 64000  
   ulimit -m unlimited  
   ulimit -u 32000

也可以通过文件来控制/etc/security/limits.d/下某个文件实现：  
**#mongodb set limits config**  
**\*  -  fsize        unlimited   # (file size)**  
**\*  -  cpu         unlimited   # (cpu time)**  
**\*  -  as           unlimited   # (virtual memory size)**  
**\*  -  nofile     64000        # (open files)**  
**\*  -  nproc     64000        # (processes/threads)**

## **Mongodb库名更改**

db.copyDatabase('old\_name', 'new\_name');

use old\_name

db.dropDatabase();

mongodump --host 10.161.219.44 --port 20000 -d tiku -c question -o /home/k12/

mongorestore -d tiku -c question\_tmp --directoryperdb /home/k12/tiku/question.bson

mongorestore --dir tiku/ -d tiku(rename)

db.question\_bak.drop()

db.question.renameCollection("question\_bak")

db.question\_tmp.renameCollection("question")

|  |
| --- |
| #!/bin/bash  . /etc/profile  mongodump --host 10.161.219.44 --port 20000 -d tiku -c question -o /home/k12/ &>updateMongo.txt  if [ $? -ne 0 ] ;then  echo "there is an error to mongodump!"  exit  fi  mongorestore -d tiku -c question\_tmp --directoryperdb /home/k12/tiku/question.bson &>>updateMongo.txt  if [ $? -ne 0 ] ;then  echo "there is an error to mongorestore!"  exit  fi  mongo tiku switchMongo.js |
| var collections=db.getCollectionNames();  if(collections.indexOf("question\_bak")!=-1){  db.question\_bak.drop();  }  if(collections.indexOf("question")!=-1){  db.question.renameCollection("question\_bak");  }  if(collections.indexOf("question\_tmp")!=-1){  db.question\_tmp.renameCollection("question");  } |