**目录**

[1. 使用操作 1](#_Toc465784631)

[1.1常见的启动选项 1](#_Toc465784632)

[1.2关闭mongod服务 1](#_Toc465784633)

[1.2.1几种安全关闭方式 1](#_Toc465784634)

[1.2.3异常关闭问题 1](#_Toc465784635)

[2. CURD操作 1](#_Toc465784636)

[2.1几种对象的常用操作 1](#_Toc465784637)

[2.3文档操作 1](#_Toc465784638)

[2.3.1插入文档 1](#_Toc465784639)

[2.3.2查询集合中的文档 1](#_Toc465784640)

[2.3.3文档更新 1](#_Toc465784641)

[3. 一些高级操作 1](#_Toc465784642)

[3.1投影 1](#_Toc465784643)

[3.2limit显示n条文档 1](#_Toc465784644)

[3.3skip跳过n条显示 1](#_Toc465784645)

[3.4sort文档排序 1](#_Toc465784646)

[3.5索引 1](#_Toc465784647)

[3.5.1索引创建 1](#_Toc465784648)

[4. 常用的 1](#_Toc465784649)

[5. 参考资料 1](#_Toc465784650)

MongoDB是面向文档的数据库，工作在集合和文档的概念上。

集合：是一组MongoDB文档（这些文档结构可以不同），集合与关系数据库中的表是类似的；

文档：是一组键值对，相当于关系数据库中的行，一个json行数据；

**jar cvfm MyWordCount.jar manifest.data \*.class**

# 使用操作

安装：sudo apt-get install mongodb

启动服务：sudo service mongodb start

停止：sudo service mongodb stop

客户端连接：mongo，远程连接mongo 10.58.127.56:7333

以用户名密码的形式登录到某个数据库：

mongo -u mongodb -p zhangsan 10.58.127.56:7333/admin

在Windows下的启动服务：（1）在服务中启动mongod（2）命令行执行mongod --dbpath “J:/MongoDB/data/db”，其他操作基本都是相同的。

注意：通常建议使用命令行mongod --config mongod.conf或者mongod --[option --…]的方式启动mongod这样更清楚mongod使用的配置及其相应的路径等，做到自知。

MongDB中的原子操作在记录级别，即我们对于一条记录的CURD操作是原子性的。MongoDB不保证多表之间复杂操作的一致性。

## 1.1常见的参数选项

|  |  |  |
| --- | --- | --- |
| 服务 | 参数选项 | 说明 |
| mongod | **普通选项** | |
| -h/--help | 查看帮助信息  mongod --help |
| --version | 查看版本信息  mongod --version |
| **-f/--config** | 配置文件  mongod --config=”/home/work/mongo/conf/mongod.conf” |
| **-v/--version** | 打开日志冗余信息开关  vv~vvvv依次日志信息越详细 |
| --port | 端口  mongod --port=22220 |
| --bind\_ip | 监听的ip列表，只能是本地IP。通常我们使用**--host**来配置需要监听的IP，host可以配置本地IP或者远程机器的IP。 |
| **--maxConns** | 最大连接数量，默认是1000000  mongod -maxConns=2000 |
| **--logpath** | 日志文件全路径名称，注意syslog和该选项不能同时使用，同时使用会报错Can’t use both a logpath and syslog。syslog的含义是将日志信息发送到主机的syslog系统，而不是输出到指定的日志文件。  mongod --logpath=/home/work/mongo/log/mongod.log |
| **--logappend** | 让日志按照追加的方式进行，否则将会覆盖重写  mongod --logappend= true |
| **--logRotate reopen** | 设置每个日志文件大小，避免单个日志文件过大，该选项必须和--logappend一同使用，可以使rename或者reopen。  mongod --logRoate reopen |
| --pidfilepath | pid文件的全路径，如果不设置pidfile默认不创建 |
| --keyfile | 集群认证的私有key文件 |
| --httpinterface | 使能http接口，默认端口为28017，实际是dbport+1000。该选项通常和--rest选项一同使用。 |
| --clusterAuthMode | 开启集群认证模式，可以通过keyFile、sendKeyFile、sendX509等方式 |
| **--fork** | 以fork形式启动mongod以后台方式运行。 |
| **--auth** | 开启认证模式，需要用户名密码才能进行相应操作 |
| **--upgrade** | 该选项用于将磁盘中数据文件的格式更新到最新版，与--dbpath同用。  绝大多数情况下不要使用该选项，如果要升级，最好参考升级文档进行，不要贸然使用--upgrade选项。 |
| **--noauth** | 以非认证模式运行 |
| **--****oplogSize** | 副本集操作日志大小，以MB为单位，默认为磁盘空间的5%，越大越好。 |
| **--cpu** | 设置为true，会强制每4s将cpu和io等待写到日志文件，默认为false |
| **--diaglog** | 创建一个非常详细的故障排查和各种错误诊断日志记录，默认为0。设置为1，在dbpath下面会产生一个diaglog.开头的日志文件，可以取如下值：   |  |  | | --- | --- | | 值 | 含义 | | 0 | 关闭，不做记录。 | | 1 | 写操作记录 | | 2 | 读从左记录 | | 3 | 读写操作记录 | | 7 | 写和一些读操作记录 |   产生的日志可以通过mongosniff来查看（查看之前执行apt-get install libpcap-dev），如下：  sudo mongosniff --source DIAGLOG diaglog.51d434a9 |
| **主从模式选项** | |
| --master | master结点 |
| --slave | slave结点 |
| --source | 为slave模式时，使用source指定master结点的ip:port |
| --only | 为slave模式时，可以指定数据库为副本模式 |
| --salvedelay | 从主库到从库的操作延迟，单位s |
| **副本集选项** | |
| --replSet | 副本集选项  mongod --replSet=lt/10.54.129.12:7898,…… |
| **分片选项** | |
| --configsvr | 声明该服务为配置服务，默认端口27019 |
| --shardsvr | 声明该服务为分片服务，默认端口27018 |
| **存储选项** | |
| --storageEngine | 使用什么存储引擎，3.+，推荐使用wiredTiger引擎 |
| --dbpath | 数据库存储的路径 |
| --syncdelay | 默认为60s，每隔60s写操作的内存数据才会被刷新到磁盘中。 |
| --directoryperdb | 设置为true，将每个数据库的数据存储在不同的目录下。  PS：数据目录变化，会导致mongod找不到数据信息，因此该选项选择开启或者关闭需要提前确定好，否则需要通过移动数据文件结构来配置该选项的开关（未测！）。 |
| --noprealloc | 禁用数据文件与分配功能，禁用通常会有损性能。 |
| --nssize | 命名空间文件的大小，默认16M，最大2G。小数据库不浪费太多空间，大数据又连续磁盘空间。 |
| --profile | 数据库分析等级设置。0关闭，1仅包括慢操作（通过slowms来设置ms），2所有操作。 |
| --quota | 每个数据库的文件数量限制，开启为ture  --quotafile=8，开启配置显示默认限制为8个。 |
| --smallfiles | 是否使用较小默认文件。默认为false，开启后数据文件初始大小限制为512M，日志文件限制为128M。数据库较大不要使用该选项，大数据库开启该选项会造成大量的小文件。 |
| --repair | 启动时修复，需要关闭journal选项，否则报错。 |
| --repairpath | 修复路径，默认是dbpath下的\_tmp目录。 |
| --journal | 记录redo日志，64bit系统默认开启，32bit系统默认关闭。64bit系统要关闭必须使用--nojournal选项。 |
| --journalCommitInterval | 刷新到磁盘的时间间隔，可以在2-300ms之间，默认为30ms（不同块设备）或者100ms（单块设备）。 |
| **WiredTiger引擎选项** | |
| --wiredTigerCacheSizeGB | 最大缓存大小，默认为1/2物理内存大小。 |
| --wiredTigerJournalCompressor | 压缩日志记录，可以为none、snappy、zlib。 |
| --wiredTigerDirectoryForIndexes | 将索引和数据放在不同的目录。 |
| --wiredTigerCollectionBlockCompressor | 数据块压缩，可以为none、snappy、zlib。 |
| --wiredTigerIndexPrefixCompression | 索引前缀压缩，设置后只影响之后建立的索引。为节省磁盘空间和内存，标志性的索引key前缀，只存储一份。 |

如下是一个配置文件的示例：

bind\_ip=10.240.127.46

port=7017

dbpath=/home/hadoop/software/mongodb\_64-3.2.8/dbdir

logpath=/home/hadoop/software/mongodb\_64-3.2.8/log/mongo.log

logappend=true

logRotate=rename

pidfilepath=/home/hadoop/software/mongodb\_64-3.2.8/pidfile\_path/mongod.pid

# 开启Web接口，便于查看

httpinterface=true

rest=true

# 记录所有操作，慢操作时间以1000ms为界

profile=2

slowms=1000

directoryperdb=true

storageEngine=wiredTiger

fork=true

## 1.2关闭mongod服务

### 1.2.1几种安全关闭方式

* 方式1

**mongo ip:port/admin   --eval "db.shutdownServer()"**

* 方式2

**use admin;**

**db.shutdownServer();**

{shutdown:1, force:true}或者延迟执行{shutdown:1, timeoutSecs:60}

* 方式3

mongod --shutdown --dbpaht…..

* 方式4

kill -15可以使用，这也是官方推荐的kill方式之一。

### 1.2.3异常关闭问题

* kill -9导致的问题

由于被同事kill -9后导致MongoDB瘫痪，错误：Error: error doing query: failed。

首先MongoDB使用mmap方式进行数据文件管理，也就是说写操作基本是在内存中进行，写操作会被每隔60秒(syncdelay设定)的flush到磁盘里。如果在这60秒内flush处于停止事情我们进行kill -9那么从上次flush之后的写入数据将会全部丢失。如果在flush操作进行时执行kill -9则会造成文件混乱，可能导致数据全丢了，启动时加了repair也无法恢复。

**如何解决，暂时还不清楚，有待测试！**

* 关闭无法重启

之前听说如果用kill -9 mongod之后mongodb服务无法开启问题的

解决方法是删除/data下的mongod.lock就能够正常使用了。

# CURD操作

## 2.1几种对象的常用操作

表2.1-1mongo数据库常用操作

|  |  |  |  |
| --- | --- | --- | --- |
| 操作类型 | 命令 | 含义 |  |
| db操作 | db | 查看当前所在的数据库 |  |
| show dbs | 查看/创建所有的数据库，数据库中有collection才可显示 |  |
| use db\_name | 切换到指定数据库 |  |
| show collections | 查看当前数据库中有哪些collection（相当于表的概念） |  |
| db.stats() | 获取数据库存储大小等信息 |  |
| db.dropDatabases() | 删除当前数据库 |  |
| db.help() | 查看db类型对象的所有操作方法 |  |
| db.getCollection("name") | 获取某个collection。通常用在collection\_name包含特殊字符无法使用db.collection\_name操作数据时， 该方法就可以用了。 |  |
|  |  |  |
| collection操作 | db.collection\_name.find() | 查询集合中的所有文档 | （1）注意collection的名称中最好不要包含特殊字符比如/等，如果包含了可用类似于此类方式对表进行数据操作：db.getCollection(name).find()。 |
| db.collection\_name.count() | 查询当前数据库中文档的总数据量 |
| db.collection.dataSize() | collection中的数据大小 |
| db.collection.storageSize() | 为collection分配的空间大小，包括未使用的空间 |
| db.collection.totalIndexSize() | collection中索引数据大小 |
| db.collection.totalSize() | collection中索引+data所占空间 |
| db.createCollection(name, options) | 创建集合，options可选如下： |
| db.collection\_name.drop() | 删除集合 |
| db.collection.help() | 查看collection类型对象的所有操作方法 |
|  |  |
|  |  |
| 文档操作 |  |  |  |
|  |  |  |
|  |  |  |  |

2.2MongoDB的数据类型

String：这是最常用的数据类型来存储数据。在MongoDB中的字符串必须是有效的UTF-8；

Integer：这种类型是用来存储一个数值。整数可以是32位或64位，这取决于您的服务器；

Boolean：此类型用于存储一个布尔值 (true/ false) ；

Double：这种类型是用来存储浮点值；

Min/ Max keys：这种类型被用来对BSON元素的最低和最高值比较；

Arrays：使用此类型的数组或列表或多个值存储到一个键；

Timestamp：时间戳。这可以方便记录时的文件已被修改或添加；

Object：此数据类型用于嵌入式的文件；

Null：这种类型是用来存储一个Null值；

Symbol：此数据类型用于字符串相同，但它通常是保留给特定符号类型的语言使用；

Date：此数据类型用于存储当前日期或时间的UNIX时间格式。可以指定自己的日期和时间，日期和年，月，日到创建对象；

Object ID：此数据类型用于存储文档的ID；

Binary data：此数据类型用于存储二进制数据；

Code：此数据类型用于存储到文档中的JavaScript代码；

Regular expression：此数据类型用于存储正则表达式。

## 2.3文档操作

### 2.3.1插入文档

插入文档的时候，MongDb会为我们的文档计算一个ObjectId，这个ObjectId由12个字节组成，如下：

ObjectId(4 bytes timestamp, 3 bytes machine id, 2 bytes process id, 3 bytes incrementer)

另外，插入时如果key两侧有引号，那么连引号都会插入作为key。

* insert方法

db.collection\_name.insert({文档})

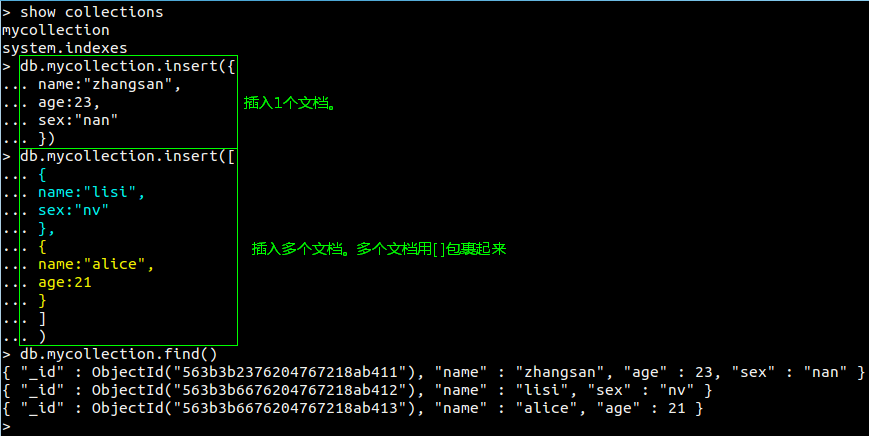


图2.3.1-1插入文档

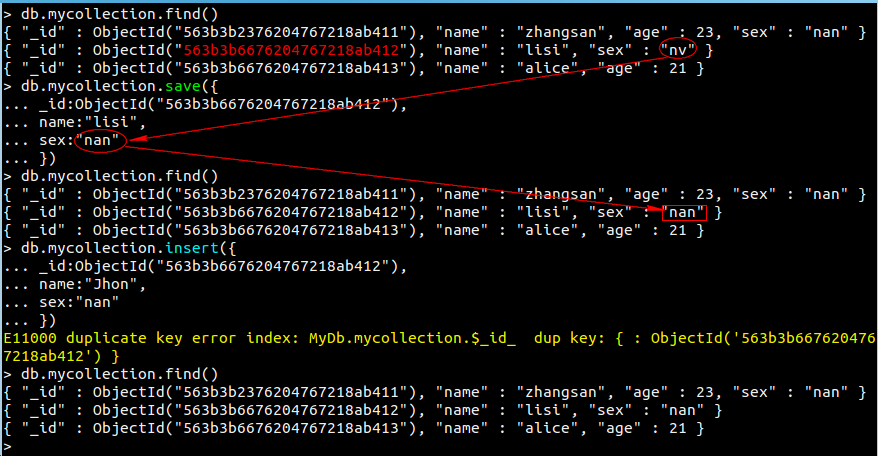
* save方法

 db.post.save(document)即可。

* insert和save的区别

1. save函数，当插入的数据ObjectId不存在时，回向集合中插入数据。如果数据存在，则会调用update更新数据；

insert插入时，当数据ObjectId存在，insert将会忽略本条插入。

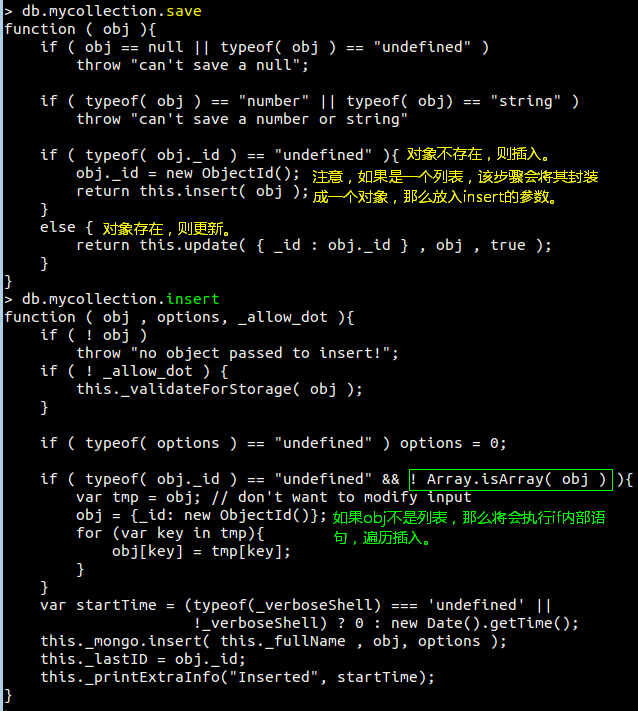


2.3.1-2save和insert的不同

1. save函数，插入列表是需要进行遍历，一个一个插入；

insert函数，不需要遍历，直接插入，所以效率略高。

输入函数名查看实现（**PS：如图属个人理解，不知正确与否**）：



2.3.1-3save和insert代码实现

* $addToSet、$each插入数组数据

db.cswuyg.update({"\_id": "wyg"}, {"$set": {"a": "c"}, "$addToSet": {"add": {"$each" :["a", "c"]}}}, true)

$each是让list中每个元素单独插入，如果没有each，list将作为整体插入。

### 2.3.2查询集合中的文档

* 整体查询

db.collection\_name.find()

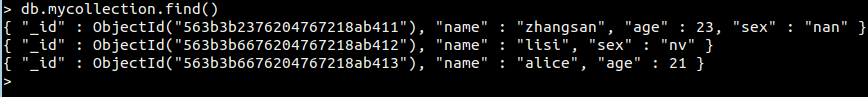


图2.3.2-1查询文档

带有格式的查询db.collection\_name.find().pretty()

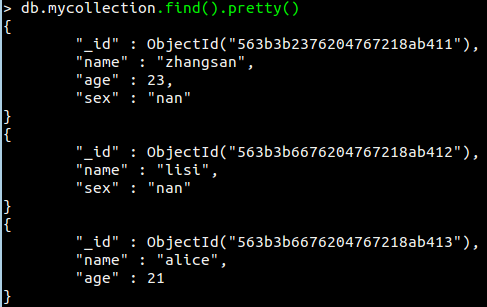


图2.3.2-2格式化查询文档

* 模糊查询

默认查询通过正则表达式实现，示例如下：

1. 寻找包含指定字段的文档

**>db.posts.find({post\_text:{$regex:"w3cschool.cc"}})**

**或者**

**>db.posts.find({post\_text:/w3cschool.cc/})**

1. 寻找包含指定字段的文档并不区分大小写

**>db.posts.find({post\_text:{$regex:"w3cschool.cc",$options:"$i"}})**

**或者**

**>db.posts.find({post\_text:/w3cschool.cc/i}})**

**集合中会返回所有包含字符串 w3cschool.cc 的数据，且不区分大小写：**

**{**

**"\_id" : ObjectId("53493d37d852429c10000004"),**

**"post\_text" : "hey! this is my post on W3Cschool.cc",**

**"tags" : [ "tutorialspoint" ]**

**}**

* 条件查询

a．单条件子语句



图2.3.2-3MongoDB的条件化查询



图2.3.2-4MongoDB条件子句使用示例

b．AND条件语句

db.mycol.find({key1:value1, key2:value2})

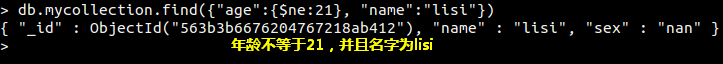


图2.3.2-5AND复合条件子句

c．OR条件子句

db.mycol.find(

{

$or:[

{ key1:value1},

{ key2:value2}

]

}

)

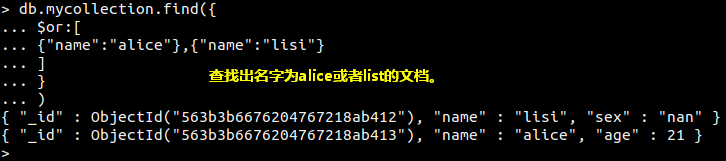


图2.3.2-6OR条件子句

d．AND和OR组合子句

{{key:value}, $or:[{kev:value},{key:value}]}

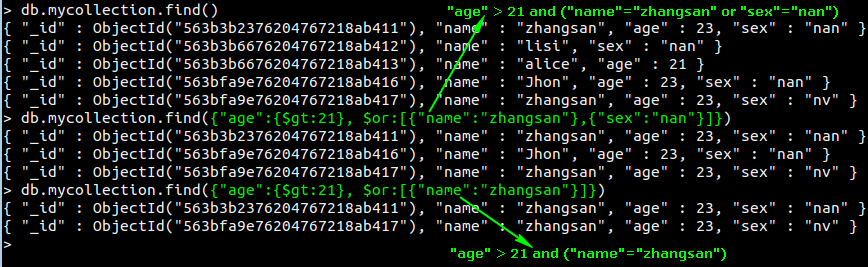


图2.3.2-7AND和OR复合子句

e．查看某条记录的某个字段

在find方法中将添加一个{}将某个字段设置为1，比如{“date”:1, “data”:1}，这样就会只显示该字段。

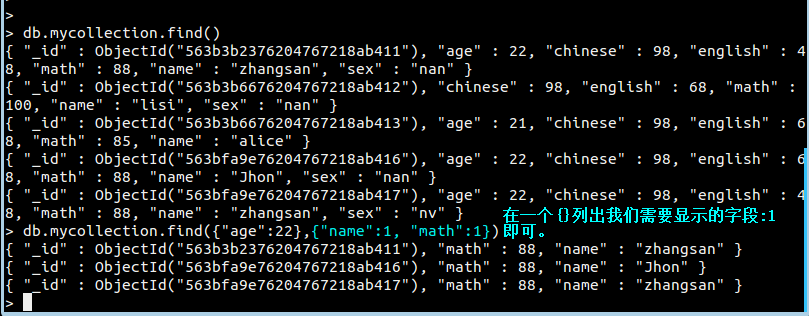


图2.3.2-8查询某个字段

### 2.3.3文档更新

* 修改key-value值

更新文档值采用update方法。

db.collection\_name.update({},{$set:{}},{**multi:true**})如果不设置multi为true，Mongo将只更新一个文档。



图2.3.3-1update更新键值

* 添加一列

db.collection\_name.update({},{$set:{“index”:0}}, {multi:true})，为所有文档添加index列。

# 一些高级操作

## 3.1投影

投影是选择指定的key列进行显示，如：db.collection\_name.find({},{key:1[,key:0]})，为1显示，0不显示。

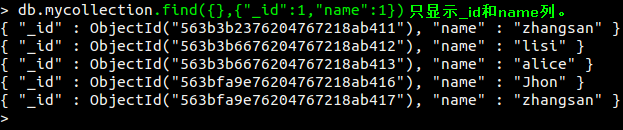


图3.1-1选择指定列显示

## 3.2limit显示n条文档

db.collection\_name.find().limit(2)显示2条记录。

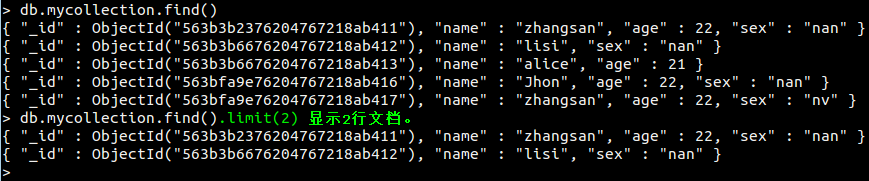


图3.2-1limit限制显示文档条数

## 3.3skip跳过n条显示

db.collection\_name.find().limit(2).skip(1)

skip语句会被先执行，如上语句表示跳过一条文档，然后显示接着的2条文档。如下：

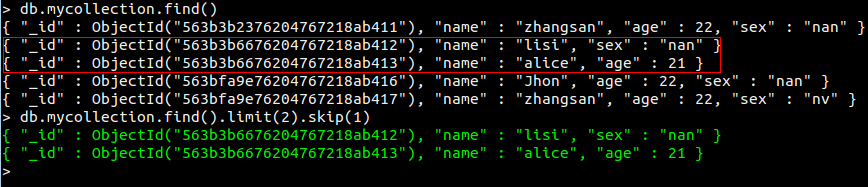


图3.3-1skip方法使用

skip的默认值是0。

## 3.4sort文档排序

db.collection\_name.sort({key:1})，1为升序，-1为降序。

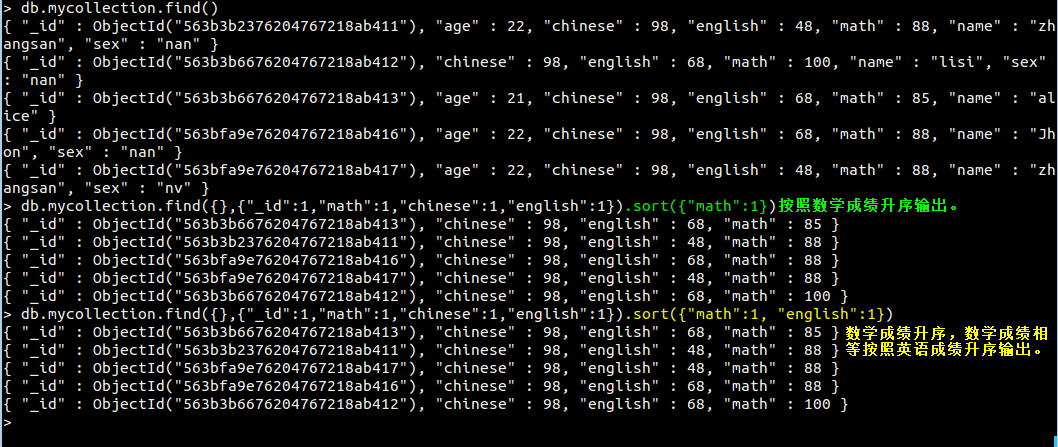


图3.4-1sort排序输出

## 3.5索引

### 3.5.1索引创建

db.collection\_name.ensureIndex()

# 常用的

#### 4.1根据数据插入时间选择数据

db.vdc\_info.findOne({"insert\_time":{"$lt":1363746619}}, {"insert\_time":1});

通常建议，在MongoDB或者其他数据库中所插入到数据一定要有insert\_time这个字段，非常重要。

#### 4.2最大值最小值

最小值：db.vdc\_info.find().sort({"insert\_time":1}).limit(1);

最大值：db.vdc\_info.find().sort({"insert\_time":-1}).limit(1);

#### 4.3统计没有某个字段的item数量

db.vdc\_info.find({insert\_time:{$exists:true}}).count();

db.vdc\_info.find({insert\_time:{$exists:false}}).count();

#### 4.4删除某条记录

删除一条记录或者多条

删除所有：db.Test.remove({\_id:”5464asdf54sda6f54sd5f65sda4f6saf”});

删除一条：db.Test.remove({\_id:”5464asdf54sda6f54sd5f65sda4f6saf”}, true);

# 参考资料

1. http://www.yiibai.com/mongodb/mongodb\_quick\_guide.html
2. http://blog.csdn.net/mcpang/article/details/7714744
3. http://www.educity.cn/wenda/389658.html
4. http://blog.csdn.net/liuxuejin/article/details/7410717
5. http://blog.csdn.net/yuan882696yan/article/details/24048673
6. http://www.cnblogs.com/liyonghui/p/mongodb.html
7. http://my.oschina.net/pwd/blog/399374
8. https://docs.mongodb.com/manual/core/wiredtiger/#storage-wiredtiger-compression
9. http://blog.csdn.net/sasoritattoo/article/details/16363199
10. http://www.cnblogs.com/buro79xxd/archive/2011/03/23/1993220.html
11. https://docs.mongodb.com/v2.2/reference/method/db.shutdownServer/
12. https://docs.mongodb.com/v2.2/reference/command/shutdown/#dbcmd.shutdown
13. http://francs3.blog.163.com/blog/static/405767272012101483936886/