MongoDB常用操作命令大全

成功启动MongoDB后，再打开一个命令行窗口输入mongo，就可以进行数据库的一些操作。输入help可以看到基本操作命令，只是MongoDB没有创建数据库的命令，但有类似的命令。

如：如果你想创建一个“myTest”的数据库，先运行use myTest命令，之后就做一些操作（如：db.createCollection('user')）,这样就可以创建一个名叫“myTest”的数据库。

# 数据库常用命令

## Help查看命令提示

复制代码代码如下:

help  
db.help();  
db.yourColl.help();  
db.youColl.find().help();  
rs.help();

## Help查看命令提示

复制代码代码如下:

use yourDB; 当创建一个集合(table)的时候会自动创建当前数据库

## 查询所有数据库

复制代码代码如下:

show dbs;

## 删除当前使用数据库

复制代码代码如下:

db.dropDatabase();

## 从指定主机上克隆数据库

复制代码代码如下:

db.cloneDatabase(“127.0.0.1”); 将指定机器上的数据库的数据克隆到当前数据库

## 从指定的机器上复制指定数据库数据到某个数据库

复制代码代码如下:

db.copyDatabase("mydb", "temp", "127.0.0.1");将本机的mydb的数据复制到temp数据库中

## 修复当前数据库

复制代码代码如下:

db.repairDatabase();

## 查看当前使用的数据库

复制代码代码如下:

db.getName();  
db; db和getName方法是一样的效果，都可以查询当前使用的数据库

## 显示当前db状态

复制代码代码如下:

db.stats();

## 当前db版本

复制代码代码如下:

db.version();

## 查看当前db的链接机器地址

复制代码代码如下:

db.getMongo();

# Collection聚集集合

## 创建一个聚集集合（table）

复制代码代码如下:

db.createCollection(“collName”, {size: 20, capped: 5, max: 100});

//创建成功会显示{“ok”:1}  
//判断集合是否为定容量db.collName.isCapped();

## 得到指定名称的聚集集合（table）

复制代码代码如下:

db.getCollection("account");

## 得到当前db的所有聚集集合

复制代码代码如下:

db.getCollectionNames();

## 显示当前db所有聚集索引的状态

复制代码代码如下:

db.printCollectionStats();

# 用户相关

## 添加一个用户

复制代码代码如下:

db.addUser("name");  
db.addUser("userName", "pwd123", true); 添加用户、设置密码、是否只读

## 数据库认证、安全模式

复制代码代码如下:

db.auth("userName", "123123");

## 显示当前所有用户

复制代码代码如下:

show users;

## 删除用户

复制代码代码如下:

db.removeUser("userName");

# 聚集集合查询

## 查询所有记录

复制代码代码如下:

db.userInfo.find();  
相当于：select\* from userInfo;

默认每页显示20条记录，当显示不下的情况下，可以用it迭代命令查询下一页数据。注意：键入it命令不能带“；”但是你可以设置每页显示数据的大小，用DBQuery.shellBatchSize= 50;这样每页就显示50条记录了。

## 查询去掉后的当前聚集集合中的某列的重复数据

复制代码代码如下:

db.userInfo.distinct("name");  
会过滤掉name中的相同数据  
相当于：select distict name from userInfo;

## 查询age = 22的记录

复制代码代码如下:

db.userInfo.find({"age": 22});  
相当于： select \* from userInfo where age = 22;

## 查询age > 22的记录

复制代码代码如下:

db.userInfo.find({age: {$gt: 22}});  
相当于：select \* from userInfo where age >22;

## 查询age < 22的记录

复制代码代码如下:

db.userInfo.find({age: {$lt: 22}});  
相当于：select \* from userInfo where age <22;

## 查询age >= 25的记录

复制代码代码如下:

db.userInfo.find({age: {$gte: 25}});  
相当于：select \* from userInfo where age >= 25;

## 查询age <= 25的记录

复制代码代码如下:

db.userInfo.find({age: {$lte: 25}});

## 查询age >= 23 并且 age <= 26

复制代码代码如下:

db.userInfo.find({age: {$gte: 23, $lte: 26}});

## 查询name中包含 mongo的数据

复制代码代码如下:

db.userInfo.find({name: /mongo/});  
//相当于%%  
[code]select \* from userInfo where name like ‘%mongo%';

## 查询name中以mongo开头的

复制代码代码如下:

db.userInfo.find({name: /^mongo/});  
select \* from userInfo where name like ‘mongo%';

## 查询指定列name、age数据

复制代码代码如下:

db.userInfo.find({}, {name: 1, age: 1});  
相当于：select name, age from userInfo;

当然name也可以用true或false,当用ture的情况下河name:1效果一样，如果用false就是排除name，显示name以外的列信息。

## 查询指定列name、age数据, age > 25

复制代码代码如下:

db.userInfo.find({age: {$gt: 25}}, {name: 1, age: 1});  
相当于：select name, age from userInfo where age >25;

## 按照年龄排序

复制代码代码如下:

升序：db.userInfo.find().sort({age: 1});  
降序：db.userInfo.find().sort({age: -1});

## 查询name = zhangsan, age = 22的数据

复制代码代码如下:

db.userInfo.find({name: 'zhangsan', age: 22});  
相当于：select \* from userInfo where name = ‘zhangsan' and age = ‘22';

## 查询前5条数据

复制代码代码如下:

db.userInfo.find().limit(5);  
相当于：selecttop 5 \* from userInfo;

## 查询10条以后的数据

复制代码代码如下:

db.userInfo.find().skip(10);  
相当于：select \* from userInfo where id not in (  
selecttop 10 \* from userInfo  
);

## 查询在5-10之间的数据

复制代码代码如下:

db.userInfo.find().limit(10).skip(5);

//可用于分页，limit是pageSize，skip是第几页\*pageSize

## or与 查询

复制代码代码如下:

db.userInfo.find({$or: [{age: 22}, {age: 25}]});  
相当于：select \* from userInfo where age = 22 or age = 25;

## 查询第一条数据

复制代码代码如下:

db.userInfo.findOne();  
相当于：selecttop 1 \* from userInfo;  
db.userInfo.find().limit(1);

## 查询某个结果集的记录条数

复制代码代码如下:

db.userInfo.find({age: {$gte: 25}}).count();  
相当于：select count(\*) from userInfo where age >= 20;

## 按照某列进行排序

复制代码代码如下:

db.userInfo.find({sex: {$exists: true}}).count();  
相当于：select count(sex) from userInfo;

# 索引

## 创建索引

复制代码代码如下:

db.userInfo.ensureIndex({name: 1});  
db.userInfo.ensureIndex({name: 1, ts: -1});

## 查询当前聚集集合所有索引

复制代码代码如下:

db.userInfo.getIndexes();

## 查看总索引记录大小

复制代码代码如下:

db.userInfo.totalIndexSize();

## 读取当前集合的所有index信息

复制代码代码如下:

db.users.reIndex();

## 删除指定索引

复制代码代码如下:

db.users.dropIndex("name\_1");

## 删除所有索引索引

复制代码代码如下:

db.users.dropIndexes();

# 修改、添加、删除集合数据

## 添加

复制代码代码如下:

db.users.save({name: ‘zhangsan', age: 25, sex: true});

添加的数据的数据列，没有固定，根据添加的数据为准

## 修改

复制代码代码如下:

db.users.update({age: 25}, {$set: {name: 'changeName'}}, false, true);  
相当于：update users set name = ‘changeName' where age = 25;  
db.users.update({name: 'Lisi'}, {$inc: {age: 50}}, false, true);  
相当于：update users set age = age + 50 where name = ‘Lisi';  
db.users.update({name: 'Lisi'}, {$inc: {age: 50}, $set: {name: 'hoho'}}, false, true);  
相当于：update users set age = age + 50, name = ‘hoho' where name = ‘Lisi';

## 删除

复制代码代码如下:

db.users.remove({age: 132});

## 查询修改删除

复制代码代码如下:

db.users.findAndModify({  
    query: {age: {$gte: 25}},   
    sort: {age: -1},   
    update: {$set: {name: 'a2'}, $inc: {age: 2}},  
    remove: true  
});  
db.runCommand({ findandmodify : "users",   
    query: {age: {$gte: 25}},   
    sort: {age: -1},   
    update: {$set: {name: 'a2'}, $inc: {age: 2}},  
    remove: true  
});

update 或 remove 其中一个是必须的参数; 其他参数可选。  
参数    详解     默认值   
query    查询过滤条件    {}   
sort    如果多个文档符合查询过滤条件，将以该参数指定的排列方式选择出排在首位的对象，该对象将被操作    {}   
remove    若为true，被选中对象将在返回前被删除    N/A   
update    一个 修改器对象  
N/A   
new    若为true，将返回修改后的对象而不是原始对象。在删除操作中，该参数被忽略。    false   
fields    参见Retrieving a Subset of Fields (1.5.0+)   
All fields   
upsert    创建新对象若查询结果为空。 示例 (1.5.4+)   
false

# 语句块操作

## 简单Hello World

复制代码代码如下:

print("Hello World!");

这种写法调用了print函数，和直接写入"Hello World!"的效果是一样的；

## 将一个对象转换成json

复制代码代码如下:

tojson(new Object());  
tojson(new Object('a'));

## 循环添加数据

复制代码代码如下:

> for (var i = 0; i < 30; i++) {  
... db.users.save({name: "u\_" + i, age: 22 + i, sex: i % 2});  
... };

这样就循环添加了30条数据，同样也可以省略括号的写法

复制代码代码如下:

> for (var i = 0; i < 30; i++) db.users.save({name: "u\_" + i, age: 22 + i, sex: i % 2});

也是可以的，当你用db.users.find()查询的时候，显示多条数据而无法一页显示的情况下，可以用it查看下一页的信息；

## find 游标查询

复制代码代码如下:

>var cursor = db.users.find();  
> while (cursor.hasNext()) {   
    printjson(cursor.next());   
}

这样就查询所有的users信息，同样可以这样写：

复制代码代码如下:

var cursor = db.users.find();  
while (cursor.hasNext()) { printjson(cursor.next); }

同样可以省略{}号

## forEach迭代循环

复制代码代码如下:

db.users.find().forEach(printjson);

forEach中必须传递一个函数来处理每条迭代的数据信息

## 将find游标当数组处理

复制代码代码如下:

var cursor = db.users.find();  
cursor[4];

取得下标索引为4的那条数据  
既然可以当做数组处理，那么就可以获得它的长度：cursor.length();或者cursor.count();

那样我们也可以用循环显示数据

复制代码代码如下:

for (var i = 0, len = c.length(); i < len; i++) printjson(c[i]);

## 将find游标转换成数组

复制代码代码如下:

> var arr = db.users.find().toArray();  
> printjson(arr[2]);

用toArray方法将其转换为数组

## 定制我们自己的查询结果

只显示age <= 28的并且只显示age这列数据

复制代码代码如下:

db.users.find({age: {$lte: 28}}, {age: 1}).forEach(printjson);  
db.users.find({age: {$lte: 28}}, {age: true}).forEach(printjson);

排除age的列

复制代码代码如下:

db.users.find({age: {$lte: 28}}, {age: false}).forEach(printjson);

## forEach传递函数显示信息

复制代码代码如下:

db.things.find({x:4}).forEach(function(x) {print(tojson(x));});

# 其他

## 查询之前的错误信息

复制代码代码如下:

db.getPrevError();

## 清除错误记录

复制代码代码如下:

db.resetError();

## 查看聚集集合基本信息

* 查看帮助  db.yourColl.help();
* 查询当前集合的数据条数  db.yourColl.count();
* 查看数据空间大小 db.userInfo.dataSize();
* 得到当前聚集集合所在的db db.userInfo.getDB();
* 得到当前聚集的状态 db.userInfo.stats();
* 聚集集合储存空间大小 db.userInfo.storageSize();
* 得到聚集集合总大小 db.userInfo.totalSize();
* Shard版本信息  db.userInfo.getShardVersion()
* 聚集集合重命名 db.userInfo.renameCollection("users"); 将userInfo重命名为users
* 删除当前聚集集合 db.userInfo.drop();

复制代码代码如下:

show dbs:显示数据库列表   
show collections：显示当前数据库中的集合（类似关系数据库中的表）   
show users：显示用户   
use <db name>：切换当前数据库，这和MS-SQL里面的意思一样   
db.help()：显示数据库操作命令，里面有很多的命令   
db.foo.help()：显示集合操作命令，同样有很多的命令，foo指的是当前数据库下，一个叫foo的集合，并非真正意义上的命令   
db.foo.find()：对于当前数据库中的foo集合进行数据查找（由于没有条件，会列出所有数据）   
db.foo.find( { a : 1 } )：对于当前数据库中的foo集合进行查找，条件是数据中有一个属性叫a，且a的值为1