# mongoDB增删改查篇

## 数据库的建立与查看

客户端连接上数据库之后使用

### 查看当前所有数据库

show dbs //查看当前所有数据库，mongoDB 3.0以上版本默认不显示数据为空的数据库

### 新建/使用数据库

mongoDB没有新建数据库的操作

use newdatabasename //若有该数据库，则直接进入使用；若没有，建立后进入使用

如：

use xfntest

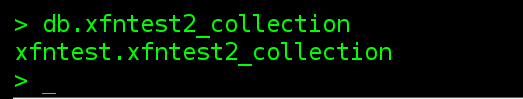
若此时show dbs ，仍然显示为空，因为数据库xfntest里面什么都没有

### 删除数据库

db.dropDatabase() //use databasename之后才能使用

### 新建表（数据集合）

db.collectionname

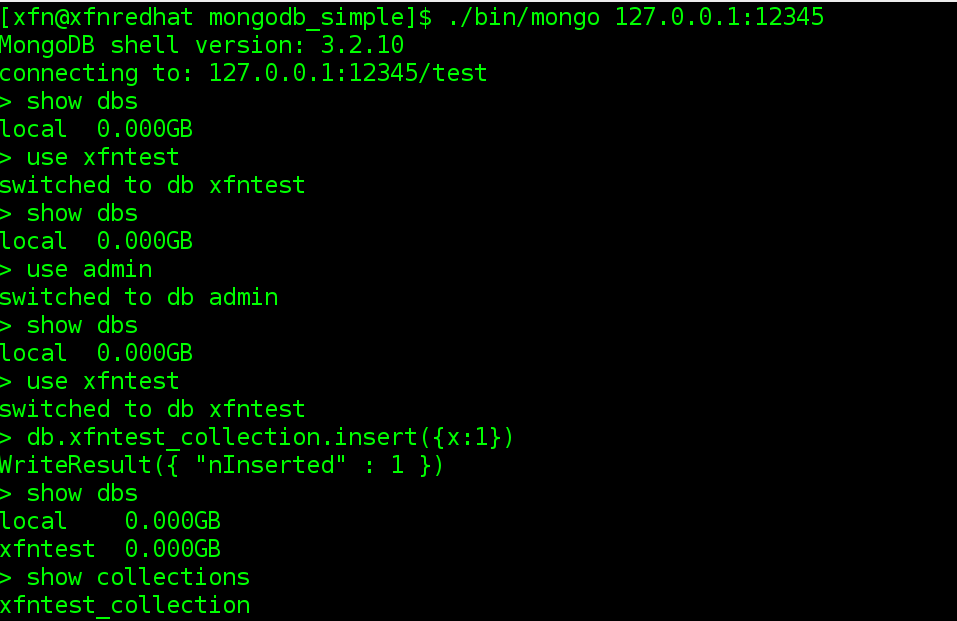


### 插入操作

db. collectionname.insert(xxx)

// 此处的collectionname可以理解成表名，把数据插入数据库的数据集合

db. collectionname.insert({表列:列值}) 插入数据表



db. collectionname.insert({表列:列值｝) 插入数据表

如：db.xfntest\_collection.insert({x:1}) //本例插入json键值类型数据

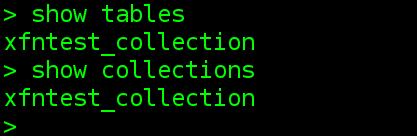
此时再show dbs，就会出现xfntest数据库

### 显示表（数据集合）

collection等价于table

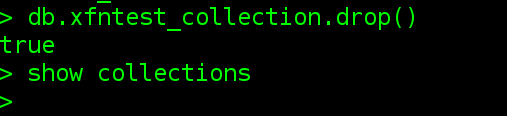
show collections //亦可理解成显示当前数据库下的所有表

show tables //效果相同



### 删除表（某个数据库下的某个指定数据集合）

db.collectionname.drop() //相当于删除表的操作

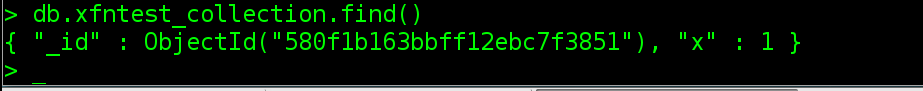


### 查询数据

#### 查询全体

db.collectionname.find() // 相当于db.表名.find();若find中不加数据，显示整个表内容

如：db.xfntest\_collection.find()则显示了数据集合中的所有数据

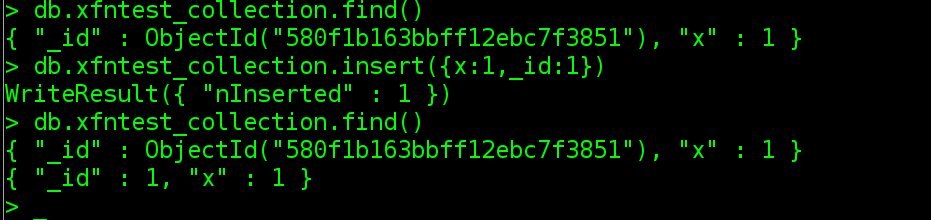


可理解成，逗号分割开表的不同字段，冒号是键值之间的连接符号

\_id是mongoDB默认生成字段，\_id相当于主键，全局唯一不能重复，ObjectId为其值

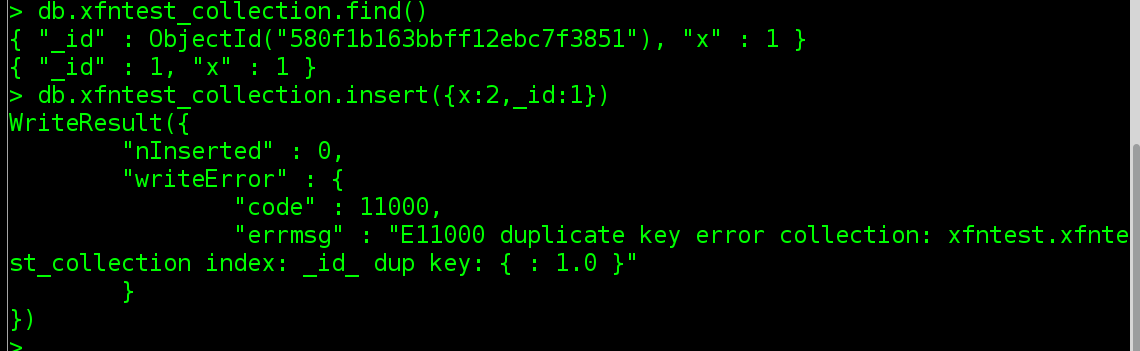
\_id虽然不能重复，但是其值用户可自行设置比如

db.xfntest\_collection.insert({x:2,\_id:1}) //插入一行：x为2，id为1



验证\_id不能重复

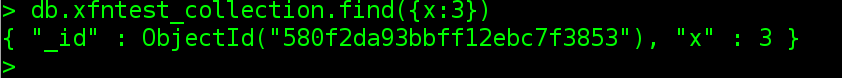
db.xfntest\_collection.insert({x:3,\_id:1})



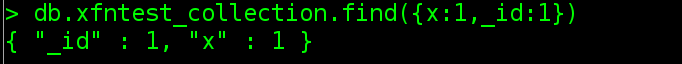
报错

#### 指定字段查询

db.collectionname.find({字段名:字段值})



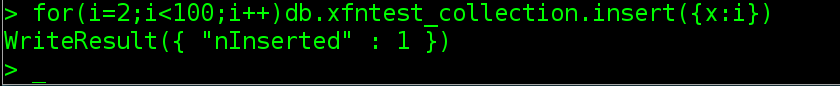
亦支持多字段查询



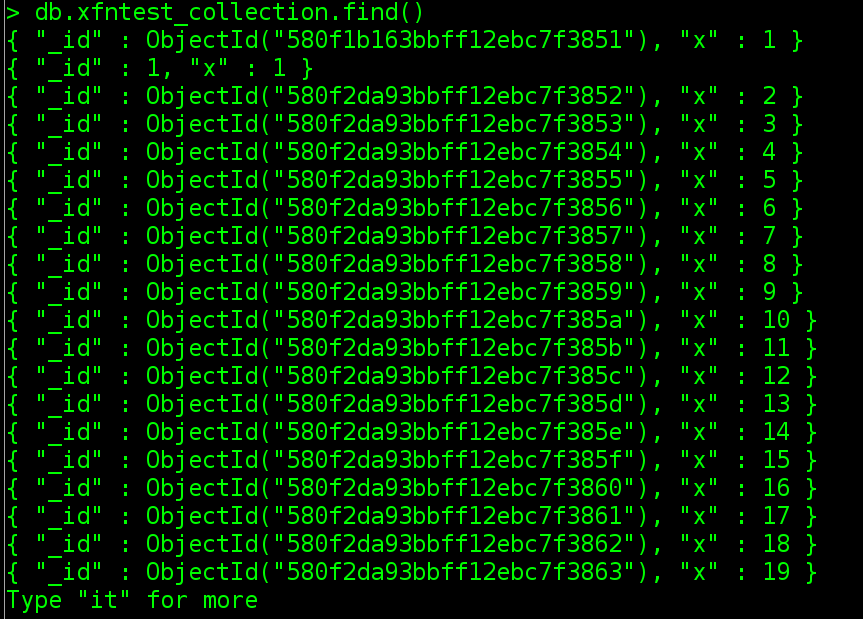
#### 插入多条数据

插入操作亦支持for循环，比如

for(i=3;i<100;i++)db. collectionname.insert({表列:i}); //插入97条数据



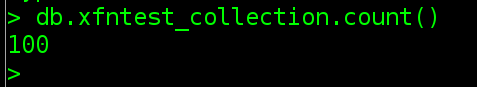
插入98条数据



显示前20条数据，输入it，显示更多

#### 计数

db. collectionname.find().count(); //进行计数



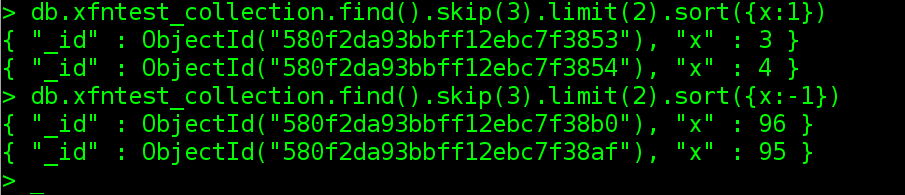
#### 查询结果限制

db.collectionname.find().skip(3).limit(2).sort({x:1})

skip(n):过滤调前n条数据

limit(m):限制返回条数为m

sort({字段名:1/-1}):使用字段为1时递增排序ASC，-1时递减排序DESC



注意一下，根据x:-1的输出结果知：db.collectionname.find().skip(3).limit(2).sort({x:1})

的skip,limit,sort虽然是从左至右的，但实际结果却是对find()的结果从右到左使用skip,limit,sort

### 数据修改

使用update()

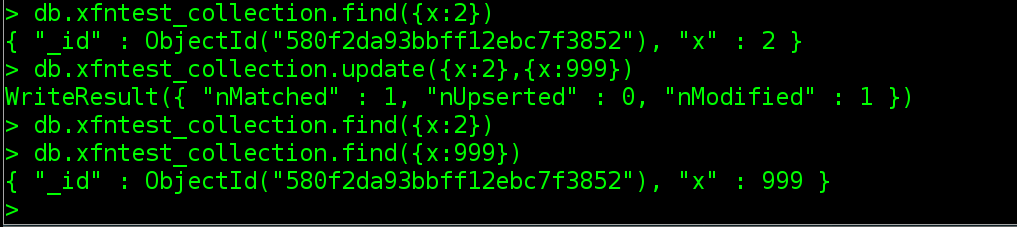
update()接收两个参数，第一个参数用于确定需要更新的记录，第二个参数用于指定新的数据值

db.collectionname.update({原字段名:原字段值},{原字段名:想修改的字段值})

#### 单字段修改

比如

db.xfntest\_collection.update({x:2},{x:999})



原来的x:2的数据被更新成x:999

#### 多字段下的部分字段修改

使用 { $set : { } }做为第二个参数值

比如我先插入

db.xfntest\_collection.insert({x:100,y:100,z:100}) //$set外围亦有{}，$set后有:

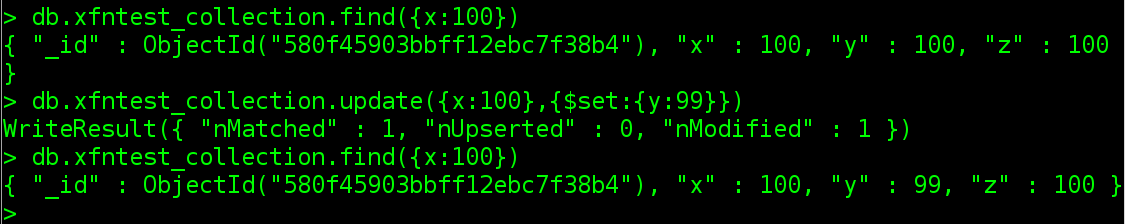
现在我只想修改y为99,不想动x和z的值怎么办？

db.xfntest\_collection.update({x:100},{$set:{y:100}})

或者

db.xfntest\_collection.update({z:100},{$set:{y:100}})

第一个参数{x:100}或者{z:100}为了确定要修改的那一行，使用 { $set : { } }做为第二个参数值，是为了部分修改，只修改y字段的数据



#### 更新一条不存在的数据

有时我们存在这样的需求：当更新的数据不在数据库时，自动添加到数据库

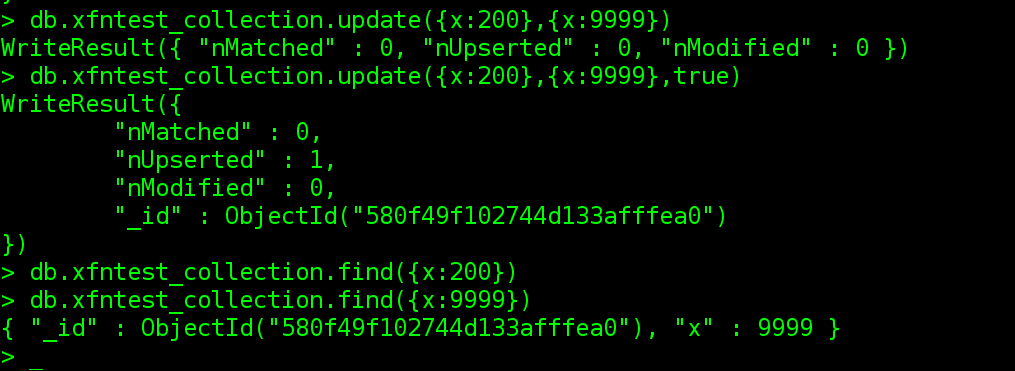
比如

db.xfntest\_collection.update({x:200},{x:9999}) //其实不建议这么做

此时数据库中不存在x:200的值，我们又想插入x:9999怎么办？

只执行上面的操作，x:9999是不会出现的

此时在update()后添加第三个参数为true即可实现自动添加不存在的数据



#### 更新多条数据

默认情况下update()只会更新找到的第一条数据

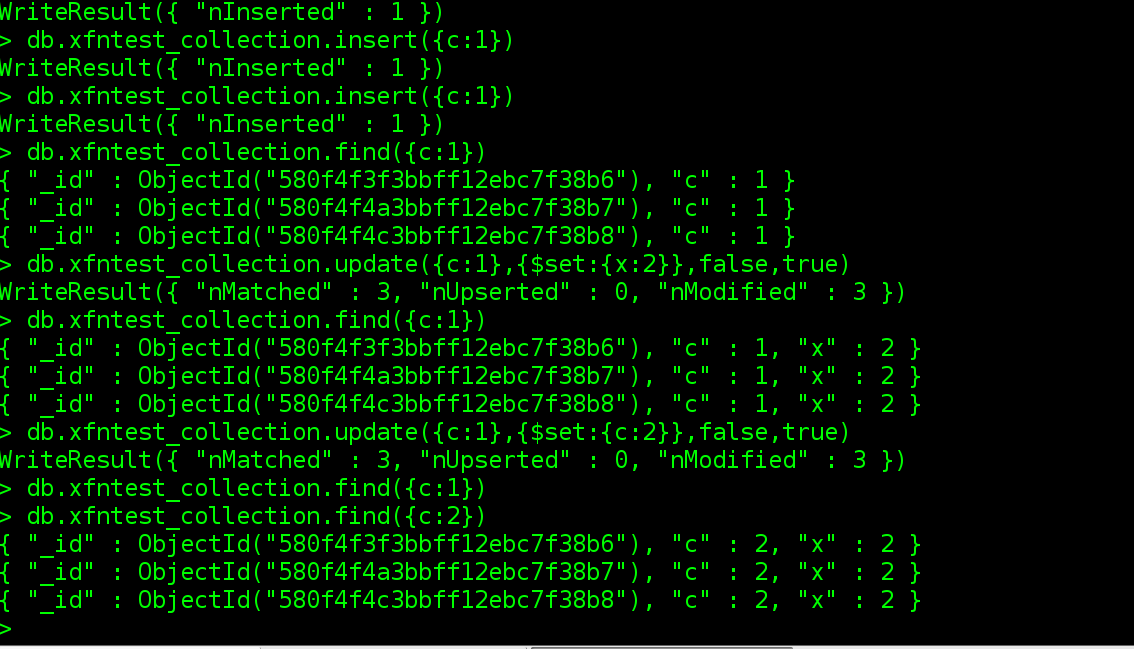
比如n条数据都含一个字段x:100，想更新x:100为x:200

使用db.collectionname.update({x:100},{x:200})发现只更新了find({x:100})的第一条数据

此时使用update()的第四个参数multi，设置为true

现在同时插入3条c:1的数据

db.xfntest\_collection.update({c:1},{$set:{c:2}},false,true)——注意：安全起见，要多条更新的字段必须使用$set操作符



update参数说明：

query : update的查询条件，类似sql update查询内where后面的。

update : update的对象和一些更新的操作符（如$,$inc...）等，也可以理解为sql update查询内set后面的

upsert : 可选，这个参数的意思是，如果不存在update的记录，是否插入objNew,true为插入，默认是false，不插入。

multi : 可选，mongodb 默认是false,只更新找到的第一条记录，如果这个参数为true,就把按条件查出来多条记录全部更新

只更新第一条记录：

db.col.update({c:1},{$set:{c:2}})

全部更新：

db.col.update({c:1},{$set:{c:2}},false,true)

只添加第一条：

db.col.update({c:1},{$set:{c:2}},true,false)

全部添加加进去:

db.col.update({c:1},{$set:{c:2}},true,true)

只更新第一条记录：

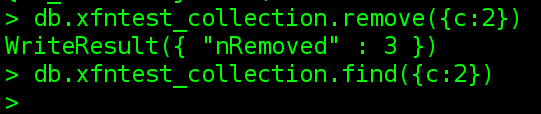
db.col.update({c:1},{$inc:{c:2}},false,false)

### 数据删除

使用remove()删除数据

与update()不同，remove()不能部分字段的删除，即不能使用{set:{}}，但是它可以同时删除多条数据

db.collectionname.remove({c:2})



remove,insert,update方法参数都不能为空

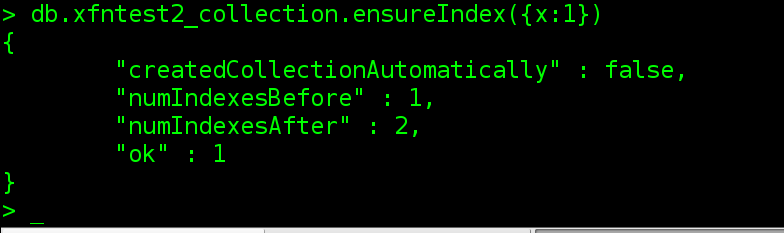
## 索引篇小引子

数据量很大时，查询使用索引可大幅度提高效率

1.db.collectionname.getIndexes() //获取当前数据集的所有索引信息，初始情况下有\_id一个索引



2.db. collectionname.ensureIndex({x:1}) #创建x字段索引，+1为升序，-1为降序，这里的1不再代表值



3.如果数据量非常大，创建索引需要消耗一定的时间，这时需要在使用数据库之前就将索引创建完毕，否则会对数据库性能造成较大影响