链滴

MongoDB 之 Curd（增改查删）常用命令

作者：zml2015
原文链接：https://ld246.com/article/1495620558274
来源网站：链滴

MongoDB常用命令：

- show dbs 命令可以显示所有数据库的列表
- db 显示当前数据库对象或集合
- use <database> 可以连接到一个指定的数据库

创建数据库

语法

use DATABASE_NAME

如果数据库不存在，则创建数据库，否则切换到指定数据库。

实例

以下实例我们创建了数据库 zml:

```
> use zml
switched to db zml
```

增

文档的数据结构和JSON基本一样。

所有存储在集合中的数据都是BSON格式。

BSON是一种类json的一种二进制形式的存储格式,简称Binary JSON。

语法

MongoDB 使用 insert() 或 save() 方法向集合中插入文档，语法如下：

```
db.COLLECTION_NAME.insert(document)
```

实例

```
> db.zml.insert({"name":"郑明亮"})
WriteResult({ "nInserted" : 1 })
> db.zml.findOne()
{ "_id" : ObjectId("591ea525c664724bd6fc4e1d"), "name" : "郑明亮" }
> db.zml.insert({"name":"张胜凡"})
WriteResult({ "nInserted" : 1 })
> db.zml.find();
{ "_id" : ObjectId("591ea525c664724bd6fc4e1d"), "name" : "郑明亮" }
{ "_id" : ObjectId("591ea60cc664724bd6fc4e1e"), "name" : "张胜凡" }

> db.zml.find().pretty(); # 将查询结果进行格式化
{
```

```
        "_id" : ObjectId("591ea525c664724bd6fc4e1d"),
        "name" : "郑明亮",
        "tel" : "15733100573"
}
{
        "_id" : ObjectId("591ea60cc664724bd6fc4e1e"),
        "name" : "张胜凡",
        "tel" : "15732199366"
}
```

插入文档你也可以使用 db.col.save(document) 命令。如果不指定 _id 字段 save() 方法类似于 insert(方法。如果指定 _id 字段，则会更新该 _id 的数据。

删

语法

remove() 方法的基本语法格式如下所示：

```
db.collection.remove(
   <query>,
   <justOne>
)
```

如果你的 MongoDB 是 2.6 版本以后的，语法格式如下：

```
db.collection.remove(
   <query>,
   {
     justOne: <boolean>,
     writeConcern: <document>
   }
)
```

参数说明：

query :（可选）删除的文档的条件。

justOne :（可选）如果设为 true 或 1，则只删除一个文档。

writeConcern :（可选）抛出异常的级别。

实例

- 移除 title 为 'MongoDB 教程' 的文档：

```
>db.col.remove({'title':'MongoDB 教程'})
WriteResult({ "nRemoved" : 2 })          # 删除了两条数据
>db.col.find()
......                                   # 没有数据
```

- 如果你只想删除第一条找到的记录可以设置 justOne 为 1，如下所示：

```
>db.COLLECTION_NAME.remove(DELETION_CRITERIA,1)
```

- 如果你想删除所有数据，可以使用以下方式（类似常规 SQL 的 truncate 命令）：

```
>db.col.remove({})
>db.col.find()
>
```

更新（改）

update() 方法

update() 方法用于更新已存在的文档。语法格式如下：

```
db.collection.update(
   <query>,
   <update>,
   {
     upsert: <boolean>,
     multi: <boolean>,
     writeConcern: <document>
   }
)
```

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,就把按条件出来多条记录全部更新。
- writeConcern :可选，抛出异常的级别。

实例

只更新第一条记录：

```
db.col.update( { "count" : { $gt : 1 } } , { $set : { "test2" : "OK"} } );
```

全部更新：

```
db.col.update( { "count" : { $gt : 3 } } , { $set : { "test2" : "OK"} },false,true );
```

只添加第一条：

```
db.col.update( { "count" : { $gt : 4 } } , { $set : { "test5" : "OK"} },true,false );
```

全部添加加进去:

```
db.col.update( { "count" : { $gt : 5 } } , { $set : { "test5" : "OK"} },true,true );
```

全部更新：

```
db.col.update( { "count" : { $gt : 15 } } , { $inc : { "count" : 1} },false,true );
```

只更新第一条记录：

```
db.col.update( { "count" : { $gt : 10 } } , { $inc : { "count" : 1} },false,false );
```

查询

find() 方法以非结构化的方式来显示所有文档。

pretty() 方法以格式化的方式来显示所有文档。

findOne() 方法，它只返回一个文档。

语法

MongoDB 查询数据的语法格式如下：

```
db.collection.find(query, projection)
```

参数说明

- query ： 可选，使用查询操作符指定查询条件
- projection ： 可选，使用投影操作符指定返回的键。查询时返回文档中所有键值， 只需省略该参即可（默认省略）。

如果你需要以易读的方式来读取数据，可以使用 pretty() 方法，语法格式如下：

```
>db.col.find().pretty()
```

MongoDB 与 RDBMS Where 语句比较

如果你熟悉常规的 SQL 数据，通过下表可以更好的理解 MongoDB 的条件语句查询：

操作	格式	范例	DBMS的类似语句
等于	{key:value}	db.col.find({"by":"菜鸟教程"}).pretty()	where by = '菜鸟教程'
小于	{key:{$lt:value}}	db.col.find({"likes":{$lt:50}}).pretty()	where likes < 50
小于或等于	{key:{$lte:value}}	db.col.find({"likes":{$lte:50}}).pretty()	where likes <= 50
大于	{key:{$gt:value}}	db.col.find({"likes":{$gt:50}}).pretty()	where likes > 50
大于或等于	{key:{$gte:value}}	db.col.find({"likes":{$gte:50}}).pretty()	where likes >= 50

不等于	{key:{$ne:value}}	db.col.find({"likes":{$ne:0}}).pretty()
	where likes != 50	

MongoDB AND 条件

MongoDB 的 find() 方法可以传入多个键(key)，每个键(key)以逗号隔开，及常规 SQL 的 AND 条件。

语法格式如下：

```
>db.col.find({key1:value1, key2:value2}).pretty()
```

实例

以下实例通过 by 和 title 键来查询 菜鸟教程 中 MongoDB 教程 的数据

```
> db.col.find({"by":"菜鸟教程", "title":"MongoDB 教程"}).pretty()
{
        "_id" : ObjectId("56063f17ade2f21f36b03133"),
        "title" : "MongoDB 教程",
        "description" : "MongoDB 是一个 Nosql 数据库",
        "by" : "菜鸟教程",
        "url" : "http://www.runoob.com",
        "tags" : [
                "mongodb",
                "database",
                "NoSQL"
        ],
        "likes" : 100
}
```

以上实例中类似于 WHERE 语句：WHERE by='菜鸟教程' AND title='MongoDB 教程'

MongoDB OR 条件

MongoDB OR 条件语句使用了关键字 $or,语法格式如下：

```
>db.col.find(
   {
      $or: [
        {key1: value1}, {key2:value2}
      ]
   }
).pretty()
```

实例

以下实例中，我们演示了查询键 by 值为 菜鸟教程 或键 title 值为 MongoDB 教程 的文档。

```
>db.col.find({$or:[{"by":"菜鸟教程"},{"title": "MongoDB 教程"}]}).pretty()
{
        "_id" : ObjectId("56063f17ade2f21f36b03133"),
        "title" : "MongoDB 教程",
```

```
        "description" : "MongoDB 是一个 Nosql 数据库",
        "by" : "菜鸟教程",
        "url" : "http://www.runoob.com",
        "tags" : [
                "mongodb",
                "database",
                "NoSQL"
        ],
        "likes" : 100
}
>
```

AND 和 OR 联合使用

以下实例演示了 AND 和 OR 联合使用，类似常规 SQL 语句为： 'where likes>50 AND (by = '菜鸟程' OR title = 'MongoDB 教程')'

```
>db.col.find({"likes": {$gt:50}, $or: [{"by": "菜鸟教程"},{"title": "MongoDB 教程"}]}).pretty()
{
        "_id" : ObjectId("56063f17ade2f21f36b03133"),
        "title" : "MongoDB 教程",
        "description" : "MongoDB 是一个 Nosql 数据库",
        "by" : "菜鸟教程",
        "url" : "http://www.runoob.com",
        "tags" : [
                "mongodb",
                "database",
                "NoSQL"
        ],
        "likes" : 100
}
```

MongoDB中条件操作符有：

(>) 大于 - $gt

(<) 小于 - $lt

(>=) 大于等于 - $gte

(<=) 小于等于 - $lte

MongoDB Limit与Skip方法

语法

limit()方法基本语法如下所示：

```
>db.COLLECTION_NAME.find().limit(NUMBER)
```

MongoDB Skip() 方法

我们除了可以使用limit()方法来读取指定数量的数据外，还可以使用skip()方法来跳过指定数量的数据

skip方法同样接受一个数字参数作为跳过的记录条数。

语法

skip() 方法脚本语法格式如下：

```
>db.COLLECTION_NAME.find().limit(NUMBER).skip(NUMBER)
```

实例

以上实例只会显示第二条文档数据

```
> db.zml.find({}).limit(1).skip(1).pretty();
{
        "_id" : ObjectId("591ea60cc664724bd6fc4e1e"),
        "name" : "张胜凡",
        "tel" : "15732199366"
}
> db.zml.find({},{_id:0,name:1}).limit(1).skip(0).pretty();
{ "name" : "郑明亮" }
```

补充说明：

- 第一个 {} 放 where 条件，为空表示返回集合中所有文档。
- 第二个 {} 指定那些列显示和不显示 （0表示不显示 1表示显示)。
- 当查询时同时使用sort,skip,limit，无论位置先后 最后执行顺序 sort再skip再limit。

MongoDB sort()方法

在MongoDB中使用使用sort()方法对数据进行排序，sort()方法可以通过参数指定排序的字段，并使用 1 和 -1 来指定排序的方式，其中 1 为升序排列，而-1是用于降序排列。

语法

sort()方法基本语法如下所示：

```
>db.COLLECTION_NAME.find().sort({KEY:1})
```

实例

```
>db.col.find({},{"title":1,_id:0}).sort({"likes":-1})
{ "title" : "PHP 教程" }
{ "title" : "Java 教程" }
{ "title" : "MongoDB 教程" }
```

补充说明：

- 第一个 {} 放 where 条件，为空表示返回集合中所有文档。
- 第二个 {} 指定那些列显示和不显示 （0表示不显示 1表示显示)。
- 当查询时同时使用sort,skip,limit，无论位置先后 最后执行顺序 sort再skip再limit。