## 一 数据库操作

|  |  |
| --- | --- |
| 操作 | 命令 |
| 查看所有数据库 | show dbs |
| 查看当前使用的数据库 | db |
| 切换或创建数据库 | use 数据库名 |
| 删除数据库 | db.dropDatabase() |

## 二 数据常用类型

|  |  |
| --- | --- |
| 类型 | 说明 |
| Object ID | 文档ID（12字节的十六进制数）  1-4 当前时间戳  5-7 机器的ID  8-9 MongoDB服务进程的id  10-12 增量值 |
| String | 字符串（utf-8） |
| Boolean | true或者flase |
| integer | 整数 |
| Double | 浮点数 |
| Arrays | 数组 |
| Object | 嵌入式文档 |
| Null | Null |
| Timestamp | 时间戳 |
| Date | 时间 |

## 三 集合操作

#### 3.1 创建集合 createCollection

|  |
| --- |
| 格式  db.createCollection(name,{options}) |

option可选参数：

|  |  |  |
| --- | --- | --- |
| 参数 | 类型 | 说明 |
| capped | Boolean | true：固定集合大小，超过则删除最早的文档  （如果capped:true-->必须设置集合大小size）  （无法remove删除<document>） |
| autoIndexId | Boolean | true：自动在\_id字段创建索引 |
| size | integer | 指定集合的最大大小 |
| max | integer | 指定包含文档的最大数量 |

例子：

|  |
| --- |
| use mydb  db.createCollection('c1',{capped:true,size:10,max:3,autoIndexId:true})  db.c1.drop()  db.c1.createCollection('c1') |

#### 3.2 插入 insert

|  |
| --- |
| 格式：  db.集合名字.insert(<document>) |

|  |  |
| --- | --- |
| insertOne() | 插入一条文档 |
| insertMany() | 插入多条文档（insert其实等同于insertMany） |

例如：

|  |
| --- |
| db.c1.insert({book:'红楼梦',price:30})  db.c1.insert([  {book:'红与黑',price:40},  {book:'方与圆',price:50}  ])  等同于：  item = [{book:'红与黑',price:40},{book:'方与圆',price:50}]  db.c1.insert(c1) |

#### 3.3 保存 save

|  |
| --- |
| 格式：  db.集合名字.save(<document>) |

save和insert的不同：

如果<document>指定\_id，且这个\_id的值在集合中已经存在

insert(<document>)报错

save(<document>)替原数据

#### 3.4 删除 drop remove

|  |
| --- |
| 格式：  db.集合名字.drop() 删除集合  db.集合名字.remove( 删除集合中指定的文档  <query>,  <justOne>  ) |

justOne默认是false，表示删除匹配的所有文档，如果为true，删除匹配的第一个文档

例如：

|  |
| --- |
| db.c1.remove(  {book:'红楼梦'},  {justOne:true}  )  db.c1.remove(  {book:'方与圆'},  ) |

#### 3.5 更改 update

|  |
| --- |
| 格式：  db.collection.update(  <query>,  <update>,  {  upsert:<boolean>,  multi:<boolean>  }  ) |

参数说明

|  |  |
| --- | --- |
| 关键字 | 说明 |
| upsert | true：如果找不到<query>，就插入<update>  默认是flase |
| multi | true：替换所有满足<query>的文档为<update>，必须配合$set  默认是flase，只替换满足<query>的第一个 |

例如：

|  |
| --- |
| db.c1.update(  {book:'红楼梦'},  {book:'红楼梦',price:43}  )  db.c1.update(  {book:'红与黑'},  {book:'红与黑',price:54},  {multi:true}  )  db.c1.update(  {book:'笑傲江湖'},  {book:'笑傲江湖',price:35}  )  db.c1.update(  {book:'笑傲江湖'}，  {$set:{price:34}}  ) |

注意：$set{key:value}：只更改某一个键的值，如果不存在这个key，就添加

## 四 集合查找

### 4.1 find和findOne

|  |
| --- |
| 格式：  db.集合名字.find(  <query>  )  db.集合名字.findOne(  <query>  ) |

例如

|  |
| --- |
| db.c1.find()  db.c1.findOne()  db.c1.find({book:'红楼梦'}) |

### 4.2 比较运算符

|  |  |
| --- | --- |
| 运算符 | 说明 |
| $lt | 小于 |
| $lte | 小于等于 |
| $gt | 大于 |
| $gte | 大于等于 |
| $ne | 不等于 |
| $eq | 等于 |

例如：

|  |
| --- |
| bd.c1.find({price:{$gt:40}}) |

### 4.3 逻辑运算符

|  |  |
| --- | --- |
| 运算符 | 说明 |
| $or | 或者 |
| ， | and |

|  |
| --- |
| $or:[  {.....},{.....}  ] |

例如：

|  |
| --- |
| db.c1.find(  {price:{$gt:10,$lt:50}}  )  db.c1.find(  {$or:[  {book:'红楼梦'},  {book:'方与圆'}  ]  }  ) |

### 4.4 范围运算符

|  |  |
| --- | --- |
| 运算符 | 说明 |
| $in | 在... ...之中 |

例如：

|  |
| --- |
| db.c1.find(  {book:{$in:['红与黑','方与圆']}}  ) |

### 4.5 正则表达式

|  |  |
| --- | --- |
| 关键字 | 说明 |
| $regex | 正则 |

例如：

|  |
| --- |
| db.c1.find({book:{$regex:'.\*红.\*'}}) |

### 4.6 自定义查询

|  |  |
| --- | --- |
| 格式 | 说明 |
| $where:function(){....} | 自定义查询 |

例如：

|  |
| --- |
| db.c1.find({$where:function(){return this.price>10}}) |

### 4.7 读指定数量文档 limit

例如：

|  |
| --- |
| db.c1.find().limit(2) |

### 4.8 跳过指定数量的文档skip

|  |
| --- |
| db.c1.find().skip(2) |

### 4.9 排序sort

|  |
| --- |
| 格式：  db.数据库名字.find().sort({字段名字:1,字段名字:-1}) |

1表示升序，-1降序

例如：

|  |
| --- |
| db.c1.find().sort({price:1,book:-1}) |

### 4.10 投影

|  |
| --- |
| 格式：  db.数据库名字.find({字段:1,字段:-1}) |

1表示显示，0表示不显示，\_id默认为显示

例如：

|  |
| --- |
| db.c1.find({},{price:1,book:0}) |

### 4.11 去重 distinct

|  |
| --- |
| 格式：  db.集合名字.distinct('去重字段',{条件}) |

例如：

|  |
| --- |
| db.c1.distinct('book',{price:{$gt:10}}) |

## 五 聚合

### 5.1 格式

|  |
| --- |
| db.集合名字.aggregate([{管道:{表达式}}]) |

### 5.2 管道

#### 5.2.1 修改文档结构 $project

例子：

|  |
| --- |
| db.c1.aggregate([{$project:{\_id:0,book:1,price:'null'}}]) |

0表示不显示，1表示显示，字符串表示输出结果中替换的内容

只能\_id：0，其他字段不行

#### 5.2.2 过滤 $match

例子：

|  |
| --- |
| db.c1.aggregate([{$match:{price:{$gt:30}}}]) |

#### 5.2.3 显示管道返回的文档书$limit

例子：

|  |
| --- |
| db.c1.aggregate([{$limit:2}]) |

#### 5.2.4 跳过指定的文档$skip

例子：

|  |
| --- |
| db.c1.aggregate([{$skip:2}]) |

#### 注意：$limit和$skip有先后顺序

#### 5.2.5 拆分集合 $unwind

例子：

|  |
| --- |
| db.createCollection('c2')  item = {  h1:'中国文学',  h2:'小说',  book\_name:['第一本书','第二本书'],  price:[30,40]  }  db.c2.insert(item)  db.c2.aggregate([  {$unwind:{path:'$book\_name'}},  {$unwind:{path:'$price'}}  ])  item2 = {h1:'中国文学',h2:'小说',book\_name:'第三本书'}  db.c2.insert(item)  db.c2.aggregate([{$unwind:{path:'$price'}}])  db.c2.aggregate([{$unwind:{path:'$price',preserveNullAndEmptyArrays:true}}]) |

注意：preserveNullAndEmptyArrays:true，避免缺失字段的文档丢失

#### 5.2.6 分组 $group

例子：

|  |
| --- |
| db.c1.aggregate([  {$group:{\_id:'$book'},price\_avg:{$avg:'$price'}}  ]) |

\_id表示筛选的字段

#### 5.2.7 排序 $sort

例子：

|  |
| --- |
| db.c1.aggregate([  {$sort:{book:1,price:-1}}  ]) |

### 5.3 常用表达式

|  |  |
| --- | --- |
| 表达式 | 说明 |
| $sum | 求和 |
| $avg | 平均值 |
| $min | 最小值 |
| $max | 最大值 |
| $push | 把结果文档插入到一个数组中 |
| $addToset | 在结果文档中插入值到一个数组中，但不创建副本 |
| $first | 根据资源文档的排序获取第一个文档数据 |
| $last | 根据资源文档排序获取最后一个文档数据 |

## 六 python操作mongodb

### 6.1 安装pymongo

|  |
| --- |
| pip install pymongo |

### 6.2 入门案例

|  |
| --- |
| *import* pymongo  mongo\_client = pymongo.MongoClient('localhost',27017) # 使用mydb这个数据库 mydb = mongo\_client.mydb # 得到集合对象 c1 = mydb.c1 # 插入 c1.insert({'book':'三个火枪手','price':30}) # 保存 c1.save({'book':'基督山恩仇记','price':40}) # 删除 c1.remove({'book':'红楼梦'},{'justOne':*True*}) # 更改 c1.update({'book':'方与圆'},{'$set':{'price':100}})  # 查找 ret1 = c1.find() *for* i *in* range(ret1.count()):  print(ret1.next()) ret2 = c1.find().limit(2).skip(2) *for* i *in* range(2):  print(ret2.next()) ret3 = c1.find\_one() print(ret3.next())  # 聚合  ret4 = c1.aggregate([{'$project':{'\_id':'book','count':{'$sum':1}}}]) print(ret4) print(ret4.next()) |

注意：驼峰式命名：采用\_分开