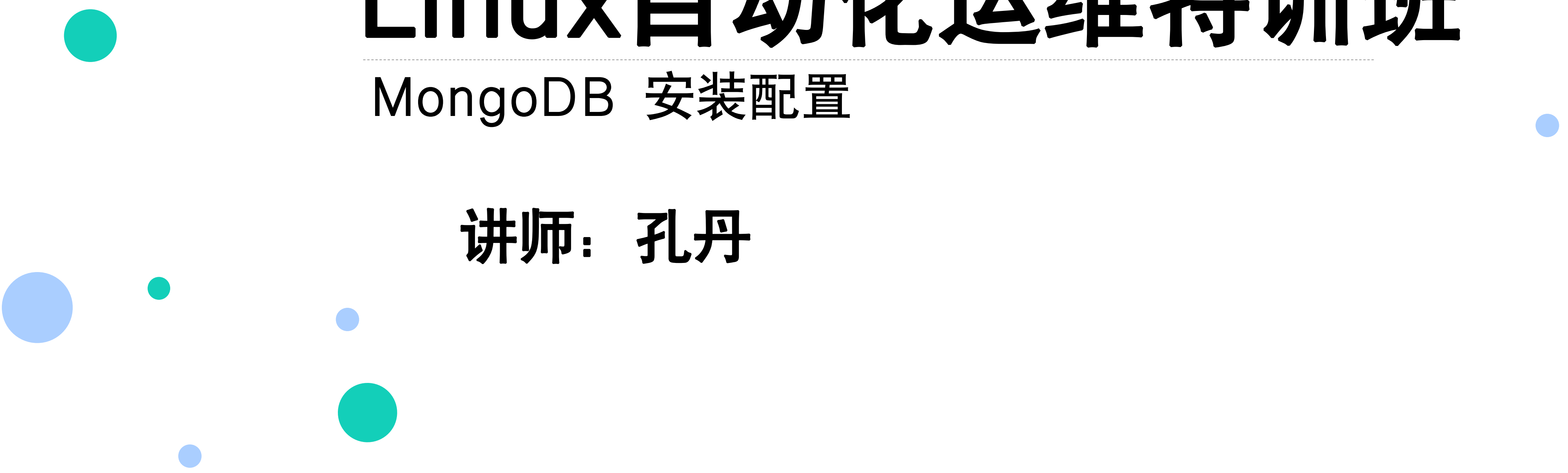




# Linux自动化运维特训班

MongoDB 基本操作

讲师：孔丹



# 大纲

- 数据库操作
- 集合操作
- 插入文档
- 数据查询
- 更新文档
- 删除数据

# 数据库操作

帮助:

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

创建数据库

```
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。

```
MongoDB Enterprise > use MyDB
```

```
switched to db MyDB
```

```
MongoDB Enterprise > db
```

```
MyDB
```

```
MongoDB Enterprise > db.getName()
```

```
MyDB
```

# 数据库操作

查询所有数据库

```
show dbs;
```

```
db.stats() //当前库状态
```

```
db.help() //帮助
```

```
db //当前库, 等价于 db.getName()
```

删除数据库

```
db.dropDatabase()
```

删除当前数据库, 默认为 test, 你可以使用 db 命令查看当前数据库名。

```
MongoDB Enterprise > db.dropDatabase()
```

```
{ "ok" : 1 }
```

注意: 命令严格区分大小写

# Collection操作

## 1、创建一个聚集集合 (table)

```
db.createCollection(name, {capped: <Boolean>, autoIndexId:  
<Boolean>, size: <number>, max <number>} )
```

name:集合的名字

capped:是否启用集合限制, 如果开启需要制定一个限制条件, 默认为不启用, 这个参数没有实际意义

size:限制集合使用空间的大小, 默认为没有限制

max:集合中最大条数限制, 默认为没有限制

autoIndexId:是否使用\_id作为索引, 默认为使用(true或false)

size的优先级比max要高

```
MongoDB Enterprise > db.createCollection("books")  
{ "ok" : 1 }
```

## 2、得到指定名称的聚集集合 (table)

```
db.getCollection("account");
```

# Collection操作

3、得到当前db的所有聚集集合

```
MongoDB Enterprise > db.getCollectionNames()  
[ "books", "student" ]
```

4、显示当前db所有聚集索引的状态

```
db.printCollectionStats()
```

# Collection操作

其他:

1、查询之前的错误信息

`db.getPrevError();`

2、清除错误记录

`db.resetError();`

查看聚集集合基本信息

1、查看帮助

`db.books.help()`

2、查询当前集合的数据条数

`db.books.count()`

3、查看数据空间大小

`db.books.dataSize()`

4、得到当前聚集集合所在的db

`db.books.getDB()`

5、得到当前聚集的状态

`db.books.stats()`

# Collection操作

其他:

6、得到聚集集合总大小

`db.books.totalSize()`

7、聚集集合储存空间大小

`db.books.storageSize()`

8、Shard版本信息

`db.books.getShardVersion()`

9、聚集集合重命名

`db.books.renameCollection("BOOK")`

10、删除当前聚集集合

`db.BOOK.drop()`



# 插入文档

## 插入文档

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)。

3.2 版本后还有以下几种语法可用于插入文档：

- ◇ db.collection.insertOne():向指定集合中插入一条文档数据
- ◇ db.collection.insertMany():向指定集合中插入多条文档数据

### 1、插入单条文档

```
MongoDB Enterprise > db.student.insert({id:201,name:'张三',sex:'男',age:18})
WriteResult({ "nInserted" : 1 })
```

```
MongoDB Enterprise > db.student.insertOne({id:202,name:'李四',sex:'女',age:19})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("599196a75aa1de0dfdd680b2")
}
```

```
MongoDB Enterprise > db.student.save({id:203,name:'王五',sex:'男',age:19})
WriteResult({ "nInserted" : 1 })
```

# 插入文档

## 插入文档

查看:

```
MongoDB Enterprise > db.student.find()
```

```
{ "_id" : ObjectId("5991966a5aa1de0dfdd680b1"), "id" : 201, "name" : "张三", "sex" : "男",  
  "age" : 18 }
```

```
{ "_id" : ObjectId("599196a75aa1de0dfdd680b2"), "id" : 202, "name" : "李四", "sex" : "女",  
  "age" : 19 }
```

```
{ "_id" : ObjectId("599196d95aa1de0dfdd680b3"), "id" : 203, "name" : "王五", "sex" : "男",  
  "age" : 19 }
```

## 2、插入多条数据

使用insertMany

```
MongoDB Enterprise > db.student.insertMany(  
  ... [{id:204,name:'赵六',sex:'女',age:21},  
  ... {id:205,name:'钱七',sex:'男',age:20}]  
  ... )  
  {  
    "acknowledged" : true,  
    "insertedIds" : [  
      ObjectId("599198025aa1de0dfdd680b4"),  
      ObjectId("599198025aa1de0dfdd680b5")  
    ]  
  }
```

# 插入文档

## 插入文档

### 3、插入多维数据

```
MongoDB Enterprise > db.student.insertMany( [{id:206,name:'孙八',sex:'男',age:21,address:{province:'Hebei',city:'tangshan'}}, {id:207,name:'侯九',sex:'男',age:20,address:{province:'Zhejiang',city:'hangzhou'}},{id:207,name:'熊十',sex:'女',age:21,address:{province:'Fujian',city:'xiamen'}}] )
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("599199e55aa1de0dfdd680b6"),
    ObjectId("599199e55aa1de0dfdd680b7"),
    ObjectId("599199e55aa1de0dfdd680b8")
  ]
}
```

查看所有的记录:

```
MongoDB Enterprise > db.student.find()
```

# 插入文档

## 4、数组信息添加

```
MongoDB Enterprise > db.student.insertMany(
... [{id:208,name:'吴京',sex:'男
',age:43,address:{province:'beijing',city:'beijing'},hobdy:['mil','Fight','sing']},
... {id:209,name:'杨幂',sex:'女
',age:31,address:{province:'HongKong',city:'HongKong'},hobdy:['dance','shoppin
g','sing']},
... {id:210,name:'黄渤',sex:'男
',age:43,address:{province:'Shandong',city:'qingdao'},hobdy:['dance','game','sing
']}]})
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("59919dabea892d3424c01998"),
    ObjectId("59919dabea892d3424c01999"),
    ObjectId("59919dabea892d3424c0199a")
  ]
}
```

# 数据查询

## 1、笼统方式查询

db.数据表.find() //查询全部 , 相当于: select \* from 表;  
db.数据表.findOne() //查询全部

```
MongoDB Enterprise > db.student.find()
```

## 2 限制条件查询

db.数据表.find(条件)  
db.数据表.findOne(条件)

查询id为203的记录

```
MongoDB Enterprise > db.student.find({id:203})
```

```
{ "_id" : ObjectId("599196d95aa1de0dfdd680b3"), "id" : 203, "name" : "王五", "sex" : "男", "age" : 19 }
```

```
MongoDB Enterprise > db.student.find({id:203}).pretty()
```

```
{  
  "_id" : ObjectId("599196d95aa1de0dfdd680b3"),  
  "id" : 203,  
  "name" : "王五",  
  "sex" : "男",  
  "age" : 19  
}
```

# 数据查询

## 3 范围条件查询

\$gt \$lt \$gte \$lte 表示大于、小于、大于等于、小于等于

操作	格式	范例	RDBMS中的类似语句
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:50}}).pretty()	where likes != 50



# 数据查询

## 3 范围条件查询

\$gt \$lt \$gte \$lte 表示大于、小于、大于等于、小于等于

查询年龄大于40岁

```
MongoDB Enterprise > db.student.find({age:{$gt:40}})
```

```
{ "_id" : ObjectId("59919dabea892d3424c01998"), "id" : 208, "name" : "吴京", "sex" : "男", "age" : 43, "address" : { "province" : "beijing", "city" : "beijing" }, "hobdy" : [ "mil", "Fight", "sing" ] }
```

```
{ "_id" : ObjectId("59919dabea892d3424c0199a"), "id" : 210, "name" : "黄渤", "sex" : "男", "age" : 43, "address" : { "province" : "Shandong", "city" : "qingdao" }, "hobdy" : [ "dance", "game", "sing" ] }
```

# 数据查询

## 4 设置多个查询条件

并且（与） `db.数据表.find({条件,条件,条件}).pretty()`

OR 条件语句使用了关键字 `$or`,语法格式如下:

`db.数据表.find({ $or: [ {条件},{条件},{条件}]}).pretty()`

查询年龄大于等于20岁小于等于30岁

MongoDB Enterprise > `db.student.find({age: {$gte:20,$lte:30}})`

查看姓王的记录

MongoDB Enterprise > `db.student.find({name: /^王/})`

查看姓名里面有王的记录

MongoDB Enterprise > `db.student.find({name: /王/})`

查询指定列name、age数据

MongoDB Enterprise > `db.student.find({}, {name: 1, age :1})`

注意: `_id`是自动生成的, 如果不要显示使用下面

MongoDB Enterprise > `db.student.find({}, {name: 1, age :1, _id: 0})`

按照年龄排序

MongoDB Enterprise > `db.student.find().sort({age: 1})` //升序

MongoDB Enterprise > `db.student.find().sort({age: -1})` //降序



# 数据查询

## 4 设置多个查询条件

查询前5条数据

```
MongoDB Enterprise > db.student.find().limit(5)
```

查询7条以后的数据

```
MongoDB Enterprise > db.student.find().skip(7)
```

查询在5-8之间的数据

```
MongoDB Enterprise > db.student.find().skip(4).limit(4)
```

查询20岁以下男性

```
MongoDB Enterprise > db.student.find({age: {$lte:20},sex:'男'})
```

查询年龄小于20岁或女性

```
MongoDB Enterprise > db.student.find({$or:[{age:{$lt:20} },{sex:'女'}]})
```

统计男性个数

```
MongoDB Enterprise > db.student.find({sex:'男'}).count()
```

# 数据查询

## 5 多维字段查询

db.表.find({'key':值})

查询地址为北京

MongoDB Enterprise > db.student.find({'address.city': 'beijing'})

## 6 数组条件限制

db.表.find({字段(数组):val}) //数组元素值，有val即可，存在一个元素

查找爱好军事片的

MongoDB Enterprise > db.student.find({hobdy:'mil'})

db.表.find({字段(数组):{'\$all':[v1,v2]}}) //数组元素值，存在v1和v2即可，顺序不做要求

示例：查找同时喜欢唱歌跳舞

MongoDB Enterprise > db.student.find({hobdy:{\$all:['dance','sing']}})

## 7 限制查询字段

db.表.find({条件},{字段 :1/0,字段 :1/0})

1:查询此字段

0:排除此字段

示例：查询age为20岁的学生姓名和年龄

MongoDB Enterprise > db.student.find({age:20},{name:1,age:1})

#注意，设置字段要么都是0，要么都是1，不能既有0又有1，\_id除外，可以任意设置。

# 数据查询

## 8 其他查询

\$in查询包含的值

这个与\$where不一样，查询的值在\$in给出的范围之内就都可以查出来

示例：查询年龄在20-23岁

MongoDB Enterprise >

```
db.student.find({age:{$in:[20,21,22,23]}},{name:1,age:1,_id:0})
```

\$exists判断字段是否存在

可以用\$exists判断某一字段是否存在

查询存在age字段的记录

MongoDB Enterprise > db.student.find({age:{\$exists:true}})

查询不存在age字段的记录

MongoDB Enterprise > db.student.find({age:{\$exists:false}})

\$mod取模运算

这个操作可以进行模运算。

age除5余3的记录

MongoDB Enterprise > db.student.find({age:{\$mod:[5,3]}})

# 数据查询

## 8 其他查询

\$ne不等于操作

可以查询不等于某一字段的数据

age不是43岁

MongoDB Enterprise > db.student.find({age:{\$ne:43}})

\$nin不包含操作

这个与\$in相反，查询不包含某一字段的记录

# 更新文档

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。
- writeConcern :可选，抛出异常的级别。

# 更新文档

示例:

```
MongoDB Enterprise > db.books.find()
```

```
{ "_id" : ObjectId("599026a4124716992d85a2ed"), "title" : "MongoDB教程",  
"price" : 75, "num" : 50 }
```

将title更新为MySQL教程

```
MongoDB Enterprise > db.books.update({"title" : "MongoDB教程"},{$set:{"title" :  
"MySQL教程"}})
```

price增加1块

```
MongoDB Enterprise > db.books.update({"title" : "MySQL教程"},{$inc:{"price" :  
1}})
```

```
db.表.update({条件},{'$set' :{字段:值,字段:值}})
```

```
db.表.update({条件},{字段:值,字段:值})
```

有\$set的修改: 只修改设置的字段, 其他字段不变

没有\$set的修改: 只修改设置的字段, 没有修改的字段就删除了 (\_id除外)

注意: 字段有则修改没有则添加新字段

## 6、删除数据

删除记录: db.表.remove(条件)

删除字段: db.表.update({条件},{ '\$unset' :{字段:1/字段:0}})



# 总结

- 数据库操作
- 集合操作
- 插入文档
- 数据查询
- 更新文档
- 删除数据

# 作业

□ 1. 创建一个数据库 名字grade

□ 2. 数据库中创建一个集合名字 class

□ 3. 集合中插入若干数据 文档格式如下

```
{name:'zhang',age: 10,sex:'m',hobby:['a','b','c']}
```

hobby: draw sing dance basketball football pingpong computer

□ 4. 查找

查看班级所有人信息

查看班级中年龄为8岁的学生信息

查看年龄大于10岁的学生信息

查看年龄在 4---8岁之间的学生信息

找到年龄为6岁且为男生的学生

找到年龄小于7岁或者大于10岁的学生

找到年龄是8岁或者11岁的学生

找到兴趣爱好有两项的学生

找到兴趣爱好有draw的学生

找到既喜欢画画又喜欢跳舞的学生

统计爱好有三项的学生人数

找出本班年龄第二大的学生

查看学生的兴趣范围

将学生按年龄排序找到年龄最大的三个

删除所有 年级大于12或者小于4岁的学生



# 作业

## □5. 增加、更新、删除、统计

1. 将小红的年龄变为8岁 兴趣爱好变为 跳舞 画画
2. 追加小明兴趣爱好 唱歌
3. 小王兴趣爱好增加 吹牛 打篮球
4. 小李增加爱好, 跑步和唱歌, 但是不要和以前的重复
5. 该班所有同学年龄加1
6. 删除小明的sex属性
7. 删除小李兴趣中的第一项
8. 将小红兴趣中的画画爱好删除

增加分数域 `score: {'chinese': 88, 'english': 78, 'math': 98}`

1. 按照性别分组统计每组人数
2. 按照姓名分组, 过滤出有重名的同学
3. 统计每名男生的语文成绩
4. 将女生按照英语分数降序排列



# 谢谢观看

更多好课，请关注[万门大学APP](#)

