print(‘

MongoDB数据库-基础篇

一使用mongodb

1.常用的命令

1)mac下打开终端 执行 brew services start mongodb 启动 MongoDB服务，执行 mongo 开启shell操作；

1.5) mongo 进入客户端

2)显示所有db： show dbs

3)启用某个db： use mydb （如果没有mydb会自动创建此库）

4)显示库中的表：show tables 或 show collections （在mongodb中，table称之为collection）

5)插入一条数据：db.myCollection.insertOne( { x: 1 } ); （如果没有myCollection会自动创建此表）

6)查询表中所有数据：db.myCollection.find() (mongodb的所有数据以文档（json）的形式存放于collection中)

7)查询表中所有数据，优美排版显示：db.myCollection.find().pretty()

8)使用tab键代码提示：输入 db.my 点击tab键，会自动完成代码；输入 db.myCollection.c 点击tab键会提示所有模糊指令，手动敲指令完成即可。

9)退出shell操作： quit() 或 <control-C>快捷键于表格

show dbs 显示数据库列表

use dbname 进入dbname数据库，大小写敏感，没有这个数据库也不要紧

show collections 显示数据库中的集合，相当

2.创建&新增

db.users.save({"name":"lecaf"}) 创建了名为users的集合，并新增了一条{"name":"lecaf"}的数据

db.users.insert({"name":"ghost", "age":10}) 在users集合中插入一条新数据，，如果没有users这个集合，mongodb会自动创建

save()和insert()也存在着些许区别：若新增的数据主键已经存在，insert()会不做操作并提示错误，而save() 则更改原来的内容为新内容。

存在数据：{ \_id : 1, " name " : " n1 "} ，\_id是主键

insert({ \_id : 1, " name " : " n2 " }) 会提示错误

save({ \_id : 1, " name " : " n2 " }) 会把 n1 改为 n2 ，有update的作用。

3.删除

db.users.remove({}) 删除users集合下所有数据

db.users.remove({"name": "lecaf"}) 删除users集合下name=lecaf的数据

db.users.drop()或db.runCommand({"drop","users"}) 删除集合users

db.runCommand({"dropDatabase": 1}) 删除当前数据库

db.users.delateOne({})

db.users.delateMany({})

4.查找

db.users.find() 查找users集合中所有数据

db.users.findOne() 查找users集合中的第一条数据

5.修改

db.users.update({"name":"lecaf"}, {"age":10}) 修改name=lecaf的数据为age=10，第一个参数是查找条件，第二个参数是修改内容，除了主键，其他内容会被第二个参数的内容替换，主键不能修改

$set修改器

db.users.updateOne({"age":"18"},{$set:{name: }})

db.users.updateMany({"age":"18"},{$set:{name: }})

二、高级应用

1.条件查找

db.collection.find({ "key" : value }) 查找key=value的数据

db.collection.find({ "key" : { $gt: value } }) key > value

db.collection.find({ "key" : { $lt: value } }) key < value

db.collection.find({ "key" : { $gte: value } }) key >= value

db.collection.find({ "key" : { $lte: value } }) key <= value

db.collection.find({ "key" : { $eq: value } }) key = value

db.collection.find({ "key" : { $gt: value1 , $lt: value2 } }) value1 < key <value2

db.collection.find({ "key" : { $ne: value } }) key <> value

db.collection.find({ "key" : { $mod : [ 10 , 1 ] } }) 取模运算，条件相当于key % 10 == 1 即key除以10余数为1的

db.collection.find({ "key" : { $nin: [ 1, 2, 3 ] } }) 不属于，条件相当于key的值不属于[ 1, 2, 3 ]中任何一个

db.collection.find({ "key" : { $in: [ 1, 2, 3 ] } }) 属于，条件相当于key等于[ 1, 2, 3 ]中任何一个

db.collection.find({ "key" : { $all: [ 1, 2, 3 ] } }) 完全包含在数组中就会查到这个数组，但凡任意一个不再数组中就查不出来

db.collection.find({ "key" : { $size: 1 } }) $size 数量、尺寸，条件相当于key的值的数量是1（key必须是数组，一个值的情况不能算是数量为1的数组）

db.collection.find({ "key" : { $exists : true|false } }) $exists 字段存在，true返回存在字段key的数据，false返回不存在字度key的数据

db.collection.find({ "key": /^val.\*val$/i }) 正则，类似like；“i”忽略大小写，“m”支持多行

db.collection.find({ $or : [{a : 1}, {b : 2} ] }) $or或 （注意：MongoDB 1.5.3后版本可用），符合条件a=1的或者符合条件b=2的数据都会查询出来

db.collection.find({ "key": value , $or : [{ a : 1 } , { b : 2 }] }) 符合条件key=value ，同时符合其他两个条件中任意一个的数据

db.collection.find({ "key.subkey" :value }) 内嵌对象中的值匹配，注意："key.subkey"必须加引号

db.collection.find({ "key": { $not : /^val.\*val$/i } }) 这是一个与其他查询条件组合使用的操作符，不会单独使用。上述查询条件得到的结果集加上$not之后就能获得相反的集合。

2.排序

db.collection.find().sort({ "key1" : -1 ,"key2" : 1 }) 这里的1代表升序，-1代表降序

3.其他

db.collection.find().limit(5) 控制返回结果数量，如果参数是0，则当作没有约束，limit()将不起作用

db.collection.find().skip(5) 控制返回结果跳过多少数量，如果参数是0，则当作没有约束，skip()将不起作用，或者说跳过了0条

db.collection.find().skip(5).limit(5) 可用来做分页，跳过5条数据再取5条数据

db.collection.find().count(true) count()返回结果集的条数

db.collection.find().skip(5).limit(5).count(true) 在加入skip()和limit()这两个操作时，要获得实际返回的结果数，需要一个参数true，否则返回的是符合查询条件的结果总数

‘)

Mysql MongoDb

DB DB

Table Collections

Row Document

Column Field

MongoDB数据类型