**mongodb与mysql命令对比**

传统的关系[数据库](http://www.itokit.com/tag.php?tag=%CA%FD%BE%DD%BF%E2" \t "_blank)一般由[数据库](http://www.itokit.com/tag.php?tag=%CA%FD%BE%DD%BF%E2" \t "_blank)（database）、表（table）、记录（record）三个层次概念组成，MongoDB是由**数据库（database）、集合（collection）、文档对象（document）**三个层次组成。MongoDB对于关系型数据库里的表，但是集合中没有列、行和关系概念，这体现了模式自由的特点。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MySQL** | **MongoDB** | **Sqlite3** | | **说明** |
| mysqld | mongod |  | | [服务器](http://www.itokit.com/tag.php?tag=%B7%FE%CE%F1%C6%F7" \t "_blank)守护进程 |
| mysql | mongo | Sqlite3 | | 客户端工具 |
| mysqldump | mongodump |  | | 逻辑备份工具 |
| mysql | mongorestore |  | | 逻辑恢复工具 |
|  | db.repairDatabase() |  | | 修复数据库 |
| mysqldump | mongoexport |  | | 数据导出工具 |
| source | mongoimport |  | | 数据导入工具 |
| grant \* privileges on \*.\* to … | Db.addUser()  Db.auth() |  | | 新建用户并权限 |
| show databases | show dbs | .databases | | 显示库列表 |
| Show tables | Show collections | .tables | | 显示表列表 |
| Show slave status | Rs.status |  | | 查询主从状态 |
| Create table users(a int, b int) | db.createCollection("mycoll", {capped:true,  size:100000}) 另：可隐式创建表。 |  | | 创建表 |
| Delete from table douban |  |  | |  |
| Create INDEX idxname ON users(name) | db.users.ensureIndex({name:1}) |  | | 创建索引 |
| Create INDEX idxname ON users(name,ts DESC) | db.users.ensureIndex({name:1,ts:-1}) |  | | 创建索引 |
| Insert into users values(1, 1) | db.users.insert({a:1, b:1}) |  | | 插入记录 |
| Select a, b from users | db.users.find({},{a:1, b:1}) |  | | 查询表 |
| Select \* from users | db.users.find() |  | | 查询表 |
| Select \* from users where age=33 | db.users.find({age:33}) |  | | 条件查询 |
| Select a, b from users where age=33 | db.users.find({age:33},{a:1, b:1}) |  | | 条件查询 |
| select \* from users where age<33 | db.users.find({'age':{$lt:33}}) |  | | 条件查询 |
| select \* from users where age>33 and age<=40 | db.users.find({'age':{$gt:33,$lte:40}}) |  | | 条件查询 |
| select \* from users where a=1 and b='q' | db.users.find({a:1,b:'q'}) |  | | 条件查询 |
| select \* from users where a=1 or b=2 | db.users.find( { $or : [ { a : 1 } , { b : 2 } ] } ) |  | | 条件查询 |
| select \* from users limit 1 | db.users.findOne() |  | | 条件查询 |
| select \* from users where name like "%Joe%" | db.users.find({name:/Joe/}) |  | | 模糊查询 |
| select \* from users where name like "Joe%" | db.users.find({name:/^Joe/}) |  | | 模糊查询 |
| select count(1) from users | Db.users.count() |  | | 获取表记录数 |
| select count(1) from users where age>30 | db.users.find({age: {'$gt': 30}}).count() |  | | 获取表记录数 |
| select DISTINCT last\_name from users | db.users.distinct('last\_name') |  | | 去掉重复值 |
| select \* from users ORDER BY name | db.users.find().sort({name:-1}) |  | | 排序 |
| select \* from users ORDER BY name DESC | db.users.find().sort({name:-1}) |  | | 排序 |
| EXPLAIN select \* from users where z=3 | db.users.find({z:3}).explain() |  | | 获取存储路径 |
| update users set a=1 where b='q' | db.users.update({b:'q'}, {$set:{a:1}}, false, true) |  | | 更新记录 |
| update users set a=a+2 where b='q' | db.users.update({b:'q'}, {$inc:{a:2}}, false, true) |  | | 更新记录 |
| delete from users where z="abc" | db.users.remove({z:'abc'}) |  | | 删除记录 |
|  | db. users.remove() |  | | 删除所有的记录 |
| drop database IF EXISTS test; | use test  db.dropDatabase() |  | | 删除数据库 |
| drop table IF EXISTS test; | db.mytable.drop() |  | | 删除表/collection |
|  | db.addUser(‘test’, ’test’) |  | | 添加用户  readOnly-->false |
|  | db.addUser(‘test’, ’test’, true) |  | | 添加用户  readOnly-->true |
|  | db.addUser("test","test222") |  | | 更改密码 |
|  | db.system.users.remove({user:"test"})  或者db.removeUser('test') |  | | 删除用户 |
|  | use admin | |  | 超级用户 |
|  | db.auth(‘test’, ‘test’) | |  | 用户授权 |
|  | db.system.users.find() | |  | 查看用户列表 |
|  | show users | |  | 查看所有用户 |
|  | db.printCollectionStats() | |  | 查看各collection的状态 |
|  | db.printReplicationInfo() | |  | 查看主从复制状态 |
|  | show profile | |  | 查看profiling |
|  | db.copyDatabase('mail\_addr','mail\_addr\_tmp') | |  | 拷贝数据库 |
|  | db.users.dataSize() | |  | 查看collection数据的大小 |
|  | db. users.totalIndexSize() | |  | 查询索引的大小 |

**mongodb语法**

MongoDB的好处挺多的，比如多列索引，查询时可以用一些统计函数，支持多条件查询，但是目前多表查询是不支持的，可以想办法通过数据冗余来解决多表查询的问题。

MongoDB对数据的操作很丰富，下面做一些举例说明，内容大部分来自官方文档，另外有部分为自己理解。

查询colls所有数据

db.colls.find() //select \* from colls

通过指定条件查询

db.colls.find({‘last\_name’: ‘Smith’});//select \* from colls where last\_name=’Smith’

指定多条件查询

db.colls.find( { x : 3, y : “foo” } );//select \* from colls where x=3 and y=’foo’

指定条件范围查询

db.colls.find({j: {$ne: 3}, k: {$gt: 10} });//select \* from colls where j!=3 and k>10

查询不包括某内容

db.colls.find({}, {a:0});//查询除a为0外的所有数据

支持<, <=, >, >=查询，需用符号替代分别为$lt，$lte，$gt，$gte

db.colls.find({ “field” : { $gt: value } } );

db.colls.find({ “field” : { $lt: value } } );

db.colls.find({ “field” : { $gte: value } } );

db.colls.find({ “field” : { $lte: value } } );

也可对某一字段做范围查询

db.colls.find({ “field” : { $gt: value1, $lt: value2 } } );

不等于查询用字符$ne

db.colls.find( { x : { $ne : 3 } } );

in查询用字符$in

db.colls.find( { “field” : { $in : array } } );

db.colls.find({j:{$in: [2,4,6]}});

not in查询用字符$nin

db.colls.find({j:{$nin: [2,4,6]}});

取模查询用字符$mod

db.colls.find( { a : { $mod : [ 10 , 1 ] } } )// where a % 10 == 1

$all查询

db.colls.find( { a: { $all: [ 2, 3 ] } } );//指定a满足数组中任意值时

$size查询

db.colls.find( { a : { $size: 1 } } );//对对象的数量查询，此查询查询a的子对象数目为1的记录

$exists查询

db.colls.find( { a : { $exists : true } } ); // 存在a对象的数据

db.colls.find( { a : { $exists : false } } ); // 不存在a对象的数据

$type查询$type值为bsonhttp://bsonspec.org/数 据的类型值

db.colls.find( { a : { $type : 2 } } ); // 匹配a为string类型数据

db.colls.find( { a : { $type : 16 } } ); // 匹配a为int类型数据

使用正则表达式匹配

db.colls.find( { name : /acme.\*corp/i } );//类似于SQL中like

内嵌对象查询

db.colls.find( { “author.name” : “joe” } );

1.3.3版本及更高版本包含$not查询

db.colls.find( { name : { $not : /acme.\*corp/i } } );

db.colls.find( { a : { $not : { $mod : [ 10 , 1 ] } } } );

sort()排序

db.colls.find().sort( { ts : -1 } );//1为升序2为降序

limit()对限制查询数据返回个数

db.colls.find().limit(10)

skip()跳过某些数据

db.colls.find().skip(10)

snapshot()快照保证没有重复数据返回或对象丢失

count()统计查询对象个数

db.students.find({‘address.state’ : ‘CA’}).count();//效率较高

db.students.find({‘address.state’ : ‘CA’}).toArray().length;//效率很低

group()对查询结果分组和SQL中group by函数类似

distinct()返回不重复值