

目录

1	下载与安装	2
2	启动	2
3	基本概念	3
4	常用命令	3
4.1	调用命令	3
4.2	use	4
4.3	插入数据	4
4.4	查询数据	5
4.4.1	查询返回document	6
4.4.2	查询返回field	6
4.4.3	结果排序	7
4.4.4	分页查询	7
4.4.5	查询数量	7
4.5	删除数据	7
4.6	更新数据（与关系型差异较大）	7
4.6.1	增加field	8
4.6.2	upset	8
4.6.3	批量更新	8
4.7	索引	8
5	其他	9
5.1	使用web获得mongodb的信息	9
5.2	数据备份和恢复	9
5.3	导入导出数据	9
6	使用MongoDB	9

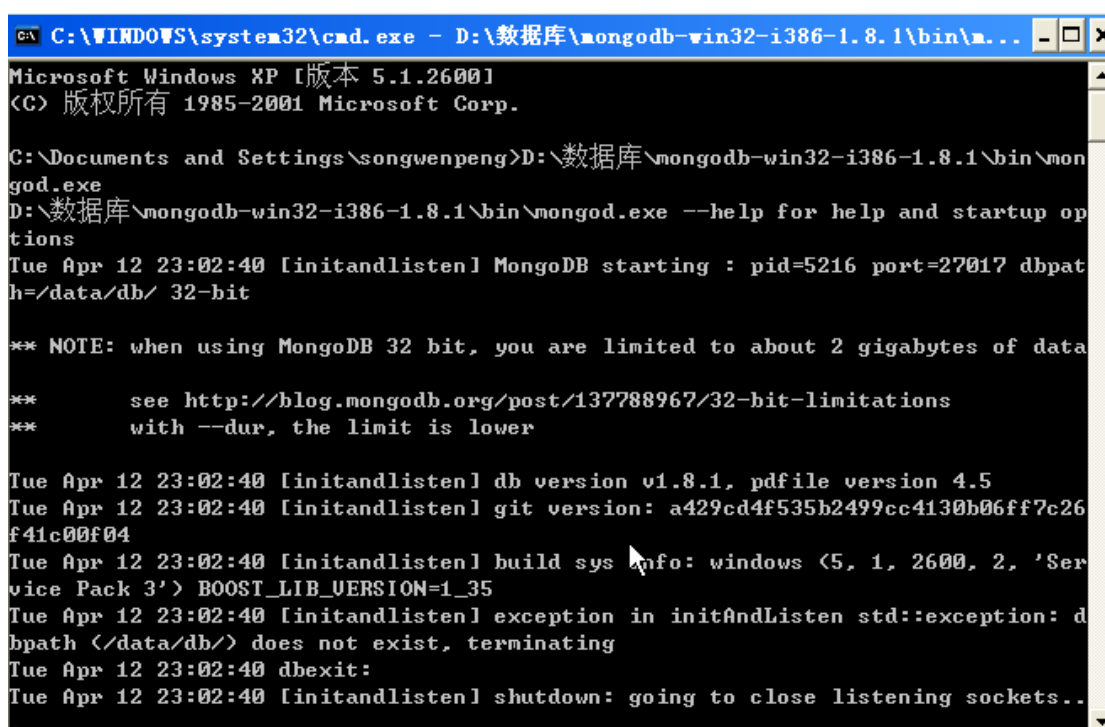
1 下载与安装

下载: <http://www.mongodb.org/downloads>

安装: 直接解压即可。

2 启动

使用 `mongo.exe run` 启动即可, 如下图所示。



```
C:\WINDOWS\system32\cmd.exe - D:\数据库\mongodb-win32-i386-1.8.1\bin\mongo.exe
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\songwenpeng>D:\数据库\mongodb-win32-i386-1.8.1\bin\mongo.exe
D:\数据库\mongodb-win32-i386-1.8.1\bin>mongo.exe --help for help and startup options
Tue Apr 12 23:02:40 [initandlisten] MongoDB starting : pid=5216 port=27017 dbpath=/data/db/ 32-bit

** NOTE: when using MongoDB 32 bit, you are limited to about 2 gigabytes of data
**       see http://blog.mongodb.org/post/137788967/32-bit-limitations
**       with --dur, the limit is lower

Tue Apr 12 23:02:40 [initandlisten] db version v1.8.1, pdfile version 4.5
Tue Apr 12 23:02:40 [initandlisten] git version: a429cd4f535b2499cc4130b06ff7c26f41c00f04
Tue Apr 12 23:02:40 [initandlisten] build sys info: windows (5, 1, 2600, 2, 'Service Pack 3') BOOST_LIB_VERSION=1_35
Tue Apr 12 23:02:40 [initandlisten] exception in initAndListen std::exception: dbpath (/data/db/) does not exist, terminating
Tue Apr 12 23:02:40 dbexit:
Tue Apr 12 23:02:40 [initandlisten] shutdown: going to close listening sockets..
```

在 LINUX 和 WINDOWS 系统下的使用大同小异, 不同的地方主要是默认的数据存储目录。LINUX 类系统下存放在 `/data/db` 下, 而 WINDOWS 会存放在 `C:\data\db` 下。可以在启动时使用 `--dbpath` 参数指定存储目录并启动。如: `bin\mongo.exe --dbpath d:\data\mongo`

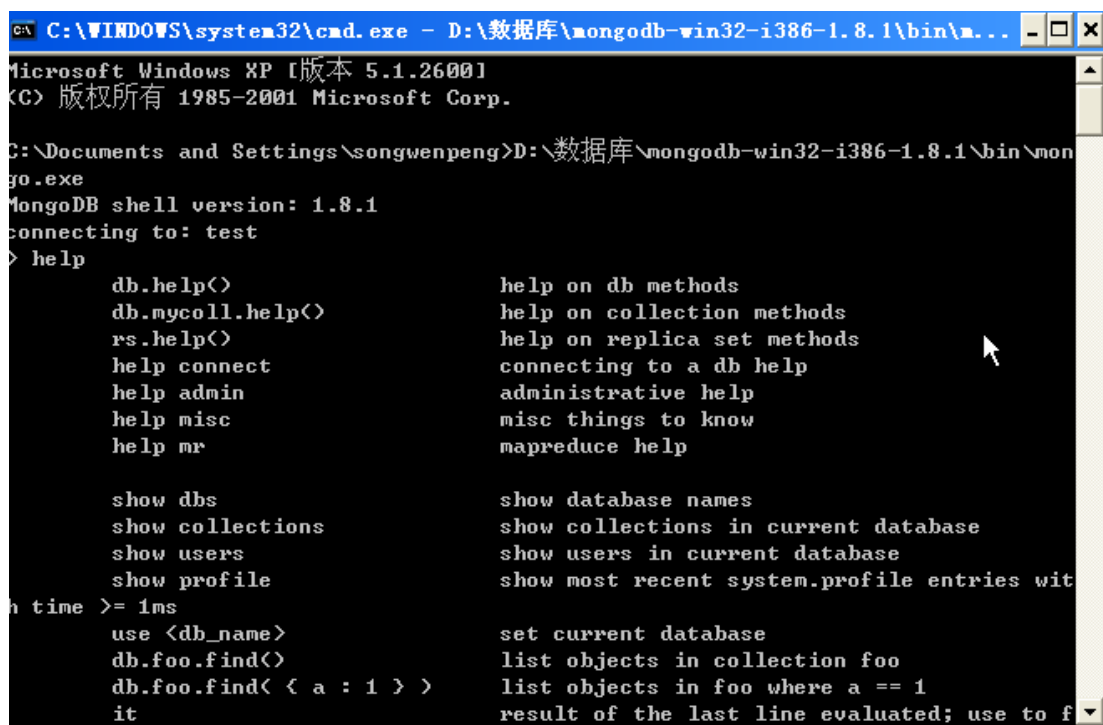
常用启动参数:

`run` 直接启动。例: `./mongo run`

`--dbpath` 指定特定存储目录启动, 若目录不存在则创建。例: `./mongod --dbpath /var/data/mongo`

`--port` 指定端口启动。例: `./mongod --port 12345`

控制台的启动使用 `mongo.exe`, 如下图所示。



```
C:\WINDOWS\system32\cmd.exe - D:\数据库\mongodb-win32-i386-1.8.1\bin\mon...
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\songwenpeng>D:\数据库\mongodb-win32-i386-1.8.1\bin\mon
go.exe
MongoDB shell version: 1.8.1
connecting to: test
> help

      db.help()                help on db methods
      db.mycoll.help()         help on collection methods
      rs.help()                help on replica set methods
      help connect             connecting to a db help
      help admin               administrative help
      help misc                misc things to know
      help mr                  mapreduce help

      show dbs                 show database names
      show collections          show collections in current database
      show users               show users in current database
      show profile             show most recent system.profile entries wit
n time >= 1ms
      use <db_name>            set current database
      db.foo.find()            list objects in collection foo
      db.foo.find( { a : 1 } ) list objects in foo where a == 1
      it                       result of the last line evaluated; use to f
```

3 基本概念

MongoDB 由 databases 组成, databases 由 collections 组成, collections 由 documents (相当于行) 组成, 而 documents 有 fields (相当于列) 组成。

MongoDB 是异步写数据。

4 常用命令

4.1 调用命令

调用命令的方法 `db.help()`; 若漏掉 `()` 则直接显示方法体。如下图所示。

```
C:\WINDOWS\system32\cmd.exe - D:\数据库\mongodb-win32-i386-1.8.1\bin\m...
> db.help
function <> {
  print("DB methods:");
  print("\tdb.addUser(username, password[, readOnly=false])");
  print("\tdb.auth(username, password)");
  print("\tdb.cloneDatabase(fromhost)");
  print("\tdb.commandHelp(name) returns the help for the command");
  print("\tdb.copyDatabase(fromdb, todb, fromhost)");
  print("\tdb.createCollection(name, { size : ..., capped : ..., max : ... } >");
  print("\tdb.currentOp() displays the current operation in the db");
  print("\tdb.dropDatabase()");
  print("\tdb.eval(func, args) run code server-side");
  print("\tdb.getCollection(cname) same as db['cname'] or db.cname");
  print("\tdb.getCollectionNames()");
  print("\tdb.getLastError() - just returns the err msg string");
  print("\tdb.getLastErrorObj() - return full status object");
  print("\tdb.getMongo() get the server connection object");
  print("\tdb.getMongo().setSlaveOk() allow this connection to read from the n
onmaster member of a replica pair");
  print("\tdb.getName()");
  print("\tdb.getPrevError()");
  print("\tdb.getProfilingLevel() - deprecated");
  print("\tdb.getProfilingStatus() - returns if profiling is on and slow thres
hold >");
  print("\tdb.getReplicationInfo()");
  print("\tdb.getSiblingDB(name) get the db at the same server as this one");
  print("\tdb.isMaster() check replica primary status");
  print("\tdb.killOp(opid) kills the current operation in the db");
  print("\tdb.listCommands() lists all the db commands");
  print("\tdb.printCollectionStats()");
  print("\tdb.printReplicationInfo()");
  print("\tdb.printSlaveReplicationInfo()");
  print("\tdb.printShardingStatus()");
  print("\tdb.removeUser(username)");
  print("\tdb.repairDatabase()");
  print("\tdb.resetError()");
  print("\tdb.runCommand(cmdObj) run a database command. if cmdObj is a strin
g, turns it into { cmdObj : 1 }");
  print("\tdb.serverStatus()");
  print("\tdb.setProfilingLevel(level,<slowms> 0=off 1=slow 2=all");
  print("\tdb.shutdownServer()");
```

可以看到方法的具体实现。是 JavaScript 程序。

4.2 use

use demodb

创建 demodb, 不用担心 demodb 不会创建, 当创建第一个 collection 时, demodb 会自动创建。

4.3 插入数据

```
db.unicorns.insert({name: 'demo', sex: 'm', weight: 70})
```

插入一个数据, collection 为 unicorns

使用 db.getCollectionNames(), 会得到 unicorns 和 system.indexes。system.indexes 对每个 DB 都会有, 用于记录 index。

db.unicorns.find()会看到 document。

如下所示

```
> use demodb
switched to db demodb
> db.unicorns.insert({name: 'demo', sex: 'm', weight: 70})
> db.getCollectionNames()
[ "system.indexes", "unicorns" ]
> db.unicorns.find()
{ "_id" : ObjectId("4da6eea3a8d5cd3b72081cf2"), "name" : "demo", "sex" : "m", "weight" : 70 }
> db.system.indexes.find()
{ "name" : "_id_", "ns" : "demodb.unicorns", "key" : { "_id" : 1 }, "v" : 0 }
>
> db.unicorns.insert({name: 'Leto', gender: 'm', home: 'Arrakeen', worm: false})

> db.unicorns.find()
{ "_id" : ObjectId("4da6eea3a8d5cd3b72081cf2"), "name" : "demo", "sex" : "m", "weight" : 70 }
{ "_id" : ObjectId("4da6f09ea8d5cd3b72081cf3"), "name" : "Leto", "gender" : "m", "home" : "Arrakeen", "worm" : false }
>
```

4.4 查询数据

先插入测试数据

```
db.unicorns.insert({name: 'Horny', dob: new Date(1992,2,13,7,47), loves: ['carrot','papaya'],
weight: 600, gender: 'm', vampires: 63});
db.unicorns.insert({name: 'Aurora', dob: new Date(1991, 0, 24, 13, 0), loves:['carrot', 'grape'],
weight: 450, gender: 'f', vampires: 43});
db.unicorns.insert({name: 'Unicrom', dob: new Date(1973, 1, 9, 22, 10), loves:['energon',
'redbull'], weight: 984, gender: 'm', vampires: 182});
db.unicorns.insert({name: 'Roooooodles', dob: new Date(1979, 7, 18, 18, 44),loves: ['apple'],
weight: 575, gender: 'm', vampires: 99});
db.unicorns.insert({name: 'Solnara', dob: new Date(1985, 6, 4, 2, 1), loves:['apple', 'carrot',
'chocolate'], weight:550, gender:'f', vampires:80});
db.unicorns.insert({name:'Ayna', dob: new Date(1998, 2, 7, 8, 30), loves: ['strawberry',
'lemon'], weight: 733, gender: 'f', vampires: 40});
db.unicorns.insert({name:'Kenny', dob: new Date(1997, 6, 1, 10, 42), loves: ['grape', 'lemon'],
weight: 690, gender: 'm', vampires: 39});
db.unicorns.insert({name: 'Raleigh', dob: new Date(2005, 4, 3, 0, 57), loves:['apple', 'sugar'],
weight: 421, gender: 'm', vampires: 2});
db.unicorns.insert({name: 'Leia', dob: new Date(2001, 9, 8, 14, 53), loves: ['apple',
'watermelon'], weight: 601, gender: 'f', vampires: 33});
db.unicorns.insert({name: 'Pilot', dob: new Date(1997, 2, 1, 5, 3), loves: ['apple',
```

```
'watermelon'], weight: 650, gender: 'm', vampires: 54});
db.unicorns.insert({name: 'Nimue', dob: new Date(1999, 11, 20, 16, 15), loves:['grape',
'carrot'], weight: 540, gender: 'f'});
db.unicorns.insert({name: 'Dunx', dob: new Date(1976, 6, 18, 18, 18), loves: ['grape',
'watermelon'], weight: 704, gender: 'm', vampires: 165});
```

4.4.1 查询返回 document

使用查询

```
db.unicorns.find({name: 'Dunx'})
```

其他说明:

- \$lt, \$lte, \$gt, \$gte and \$ne 分别表示小于、小于等于、大于、大于等于、不等于

```
db.unicorns.find({gender: {$ne: 'f'}, weight: {$gte: 701}})
```

- \$exists 用于表示 field 是否存在

```
db.unicorns.find({vampires: {$exists: false}})
```

- or 和 and

```
db.unicorns.find({gender: 'f', $or: [{loves: 'apple'}, {loves: 'orange'}, {weight: {$lt: 500}}]})
```

4.4.2 查询返回 field

```
db.unicorns.find(null, {name: 1})
```

只返回 name 这个 field, 具体如下所示:

```
> db.unicorns.find(null, {name: 1})
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cf7"), "name" : "Aurora" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cf6"), "name" : "Horny" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cf8"), "name" : "Unicrom" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cf9"), "name" : "Rooodooles" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cfa"), "name" : "Solnara" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cfb"), "name" : "Ayna" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cfc"), "name" : "Kenny" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cfd"), "name" : "Raleigh" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cfe"), "name" : "Leia" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081cff"), "name" : "Pilot" }
{ "_id" : ObjectId("4da6f22da8d5cd3b72081d00"), "name" : "Nimue" }
{ "_id" : ObjectId("4da6f231a8d5cd3b72081d01"), "name" : "Dunx" }
> db.unicorns.find(null, {name: 1, _id: 0})
{ "name" : "Aurora" }
{ "name" : "Horny" }
{ "name" : "Unicrom" }
{ "name" : "Rooodooles" }
{ "name" : "Solnara" }
{ "name" : "Ayna" }
```

```
{ "name" : "Kenny" }  
{ "name" : "Raleigh" }  
{ "name" : "Leia" }  
{ "name" : "Pilot" }  
{ "name" : "Nimue" }  
{ "name" : "Dunx" }  
>
```

4.4.3 结果排序

```
//heaviest unicorns first  
db.unicorns.find().sort({ weight: -1 })  
//by vampire name then vampire kills:  
db.unicorns.find().sort({ name: 1, vampires: -1 })  
1 表示升序, -1 表示降序
```

4.4.4 分页查询

```
db.unicorns.find().sort({ weight: -1 }).limit(2).skip(1)  
得到第二个和第三个。limit 规定查询个数, skip 规定忽略几个。
```

4.4.5 查询数量

```
db.unicorns.count({ vampires: { $gt: 50 } })  
或者  
db.unicorns.find({ vampires: { $gt: 50 } }).count()
```

4.5 删除数据

```
db.unicorns.remove()  
如下面所示。  
> db.unicorns.remove()  
> db.unicorns.find()  
>
```

4.6 更新数据（与关系型差异较大）

```
db.unicorns.update({ name: 'Rooooooodles' }, { weight: 590 })
```

注意: 此语句执行后, 先查询 name 是 'Rooooooodles' 的所有数据, 然后将 name 是 'Rooooooodles' 的整个 document 都替换为 { weight: 590 }。

即 `db.unicorns.insert({name: 'Roooooodles', dob: new Date(1979, 7, 18, 18, 44), loves: ['apple'], weight: 575, gender: 'm', vampires: 99});`

整个替换为

`{weight: 590}`

执行\$set，不会替换原有数据。

`db.unicorns.update({weight: 590}, {$set: {name: 'Roooooodles', dob: new Date(1979, 7, 18, 18, 44), loves: ['apple'], gender: 'm', vampires: 99}})`

后会还原，但`{weight: 590}`不会被替换，因为没有\$set

因此正确的更新方式如下

`db.unicorns.update({name: 'Roooooodles'}, {$set: {weight: 590}})`

4.6.1 增加 field

`db.unicorns.update({name: 'Pilot'}, {$inc: {vampires: -2}})`

\$inc 增加或减少数字

`db.unicorns.update({name: 'Aurora'}, {$push: {loves: 'sugar'}})`

\$push 增加数组元素

\$pop 减少数组元素

4.6.2 upset

若存在更新，否则添加

`db.hits.update({page: 'unicorns'}, {$inc: {hits: 1}}, true);`

`db.hits.find();`

使用第三个参数设置是否 true (upset)，默认是 false

4.6.3 批量更新

`db.unicorns.update({}, {$set: {vaccinated: true }});`

`db.unicorns.find({vaccinated: true});`

不会将所有的数据的 vaccinated 都更新为 true

若将所有的数据的 vaccinated 都更新为 true，则如下：

`db.unicorns.update({}, {$set: {vaccinated: true }}, false, true);`

`db.unicorns.find({vaccinated: true});`

4.7 索引

创建索引的方式

`db.unicorns.ensureIndex({name: 1})`

删除索引的方式


```
db.unicorns.dropIndex({name: 1})  
创建独立索引  
db.unicorns.ensureIndex({name: 1}, {unique: true})  
创建联合索引  
db.unicorns.dropIndex({name: 1, vampires: -1})
```

5 其他

5.1 使用 web 获得 MongoDB 的信息

使用
<http://localhost:28017/>
获得 MongoDB 的信息。

5.2 数据备份和恢复

使用 mongodump.exe 备份数据库
mongodump --db learn --out backup
使用 mongorestore.exe 恢复数据库
mongorestore --collection unicorns backup/learn/unicorns.bson

5.3 导入导出数据

从 JSON 和 CSV 格式导入导出 mongoexport.exe 和 mongoimport.exe
mongoexport --db learn -collection unicorns
mongoexport --db learn -collection unicorns --csv -fields name,weight,vampires

6 使用 MongoDB

什么时候使用 MongoDB?
针对关系不是很紧密并且数据量较大的数据, 例如股票数据。