Node.js 企业级实战在线课(从入门到精通) - 第五章

概述

1. 认识数据库的概念，了解关系型数据库和非关系型数据库
2. 认识非关系型数据库MongoDB数据库
3. MongoDB数据库的基本使用
4. MongoDB数据库命令详解

认识数据库

数据库，简而言之，存储数据的仓库。

关系型数据库

指数据结构是固定的，数据表是二维表结构
常见关系型数据库：
    MySQL、Oracle、SQLServer等

非关系型数据库

数据结构不固定，结构易横向扩展
常见非关系型数据库：
    MongoDB、Redis等

MongoDB数据库

介绍

1. MongoDB数据库是非关系型数据库,NoSQL的一种
2. MongoDB旨在为WEB应用提供可扩展性的高性能数据存储解决方案
3. MongoDB数据库是基于分布式存储的开源数据库系统
4. MongoDB数据库是文档式数据库，数据结构为键值对类型，BSON结构

MongoDB数据库结构介绍

```
数据库系统        DataBase System
    数据库          DataBase
        集合            Collection
            文档            document
                字段            field
```

下载安装

> MongoDB官网
>
> MongoDB安装包下载地址

添加mongod路径到环境变量

```
// 查找到mongod所在的目录

// 添加环境变量
我的电脑 -> 右键属性 -> 高级系统设置 -> 高级 -> 环境变量 path
```

MongoDB的基本使用

```
// 创建存储数据库的目录
data/db          存储数据的目录
data/log             存储日志的目录

// 启动数据库
mongod --dbpath=d:/data/db --logpath=d:/data/log/mongodb.log --storageEngine=mmapv
1

// 连接数据库
mongo

// 创建数据库/切换数据库
use dbName

// 显示当前数据系统下所有的数据库
show dbs

// 显示当前数据库下所有的集合
show collections

// 创建集合
db.createCollection('collectionName')

// 插入数据
db.stu.insert()

// 查询数据
db.stu.find()        // 查询所有的数据

// 更新
db.stu.update()

// 删除
db.remove()
```

MongoDB中允许存储的值类型

```
ObjectId
Number
String
Boolean
Object
Array
Date
Binary
Reg 正则表达式
JS代码
null
undefined
内嵌文档
```

MongoDB命令详解

插入

```
// 插入一条或多条数据
db.stu.insert()
```

查询

```
// 查询多条
db.stu.find()

// 查询一条数据
db.stu.findOne()

// 格式化查询
.pretty()
```

更新

```
/*
    upsert  描述：当更新的数据不存在时是否允许创建
    multi       描述：是否允许更新多条
*/
db.stu.update({条件},{新数据},{upsert:true/false},{multi:true/false})

// $set操作符，修改指定的字段
db.collectionName.update(con,{$set:{field:value}})

// $inc操作符，数据的增加与减少
db.collectionName.update(con,{$inc:{field:value}})

// 数组修改器
    $push操作符 向数组中压入一个单元
    db.collectionName.update(con,{$push:{field:value}})

    // $push和$each结合操作
    db.users.update({uname:'小白龙'}, {$push:{book:{$each:['西游记','水浒传']}}}),

    // $addToSet 数据已经存在不再进行数据添加

    // 删除元素
    $pop 在数组开头或者末尾删除一个
    $pull 根据条件删除

// $rename字段重命名
```

删除

```
// 删除数据
db.stu.remove({条件})

// 删除集合
db.stu.drop()

// 切换到对应的数据库
db.dropDatabase()
```

条件

```
// 与 $and
{age:20,sex:1}                    // 查询年龄20的男性
{$and:[{age:20},{sex:1}]}         // 同上

// 或 $or
{$or:[{age:20},{sex:1}]}          // 查询年龄20或者是男性

// 范围查询
{uname:{$in:['玉帝','王母','孙悟空']}} // 在范围
{uname:{$nin:['玉帝','王母','孙悟空']}} // 不在范围

// 大于 $gt
{age:{$gt:20}}

// 小于 $lt
{age:{$lt:20}}

// 等于 $eq
{age:10}
{age:{$eq:10}}

// 不等于 $ne
{sex:{$ne:1}}

// 查询值为null的数据
    db.users.find({$and:[{age:null},{age:{$exists:true}}]})
    db.users.find({age:null,x:{$exists:true}})

// 查询数组的值
    // 查询包含某个值
    db.users.find({book:'论持久战'}).pretty()
    $all
        db.users.find({book:{$all:['论持久战','雅典娜']}})

    // 精确匹配
        db.users.find({'book.1':'论持久战'}).pretty()

    // 查询指定长度
        db.users.find({book:{$size:3}}).pretty()
```

排序查询

> sort() 排序 1 正序 -1 倒序

限制查询

> limit() 限制条数

跳过查询

> skip() 跳过条数

服务器启动与客户端连接的完整模式

服务器启动完整模式

```
mongod --dbpath=d:/data/db --logpath=d:/data/log/mongodb.log --storageEngine=mmapv
1 --port=27017 --auth

mongod                启动服务器的命令
--dbpth               指定数据存储的位置
--logpath                  指定日志文件存储的位置
--storaageEngine      指定数据存储的引擎
--port                     指定服务器的端口号
--auth                     开启权限验证
```

> 注：在生产环境下开启mongodb服务器一定要开启权限验证

客户端连接完整模式

```
mongo ip:port/dbname -u username -p password
```

> 注：在mongodb服务器开启权限验证后，连接时才需要验证用户

MongoDB的权限管理

> 注：在mongodb中一定要先创建管理员身份，再开启权限验证

创建用户

> 注：在MongoDB中的有admin数据库，存储的管理员权限相关信息，对admin数据库拥有了管理员权限就对整个数据库都有了管理员权限

管理员

```
// 切换到admin数据库
use admin

// 创建管理员账户
> db.createUser({
... user : 'huxiaoshuai',
... pwd : 'memeda',
... roles:[{role:'__system',db:'admin'}]
... })
Successfully added user: {
        "user" : "huxiaoshuai",
        "roles" : [
                {
                        "role" : "__system",
                        "db" : "admin"
                }
        ]
}

roles 权限组
    role 角色
        __system        系统级别管理员
        read            读权限
        readWrite       读写权限

    db  针对的数据库
```

其他普通用户

在创建用户时必须要切换到对应的数据库位置，执行db.createUser()方法

```
// 对于school数据库创建读写的权限的用户
> db.createUser({
... user : 'schoolUser',
... pwd : '123',
... roles:[{role:'readwrite',db:'school'}]
... })
Successfully added user: {
        "user" : "schoolUser",
        "roles" : [
                {
                        "role" : "readwrite",
                        "db" : "school"
                }
        ]
}
>
```

用户登录验证

```
切换到该用户对应的数据库，进行权限验证

// 成功返回1，失败返回0
db.auth(username,pwd)
```

删除用户

```
db.dropUser(username)
```

显示当前数据库下的用户

```
show users
```

MongoDB的索引操作

> 概述：索引是数据库创建的一种便捷查询的目录，有序的查询，提升查询速度

创建索引

```
// 普通索引
> db.stu.ensureIndex({uname:1})

// 唯一性索引
> db.stu.ensureIndex({uname:1},{unique:true})

// 创建唯一性索引，删除数据
db.test.ensureIndex({"userid":1},{"unique":true,"dropDups":true})
```

查询索引

```
> db.stu.getIndexes()
```

删除索引

```
> db.stu.dropIndex({uname:1})
```

查询语句的执行效率

```
> db.stu.find().explain()
```

MongoDB的数据备份与恢复

数据备份

```
// 退出数据库客户端
mongodump -h localhost --port 27017 -d school -o d:/memeda

-h              数据库服务器IP地址
--port      端口号
-d              数据库(导出所有就取消该项)
-u              用户名
-p              密码
```

数据恢复

```
// 退出数据库
mongorestore -h localhost --port 27017 -d 数据库 --drop 文件存在路径
--drop 删除原有的数据
-u      用户
-p      密码

// 恢复所有数据到数据库
mongorestore -h --port 文件存储路径
```

MongoDB的数据导出与导入

数据导出

```
mongoexport -h localhost --port 27017 -d school -c stu -q {} -f _id,uname --type=c
sv -o d:/memeda/demo.csv

-c          对应集合
-q          查询条件
-f          指定要获取的字段
--type  指定导出的文件类型
    csv
    json
-o          指文件存储的路径
```

数据导入

```
mongoimport -h IP --port 端口 -u 用户名 -p 密码 -d 数据库 -c 表名 --type 类型 --headerl
ine --upsert --drop 文件名
```

将MongoDB服务设置系统级服务的方法

安装服务

```
--install --serviceName=MongoDB
```

启动服务

```
net start mongodb
```

关闭服务

```
net stop mongodb
```

删除服务

```
sc delete mongodb
```