

数据库操作

MySQL

mySQL的安装配置

请自行查找相关资料

在Nodejs中使用mySQL

- 安装mysql模块

```
npm install mysql
```

- 连接数据库

```
var mysql = require('mysql');

//创建连接池
var pool = mysql.createPool({
  host      : 'localhost',
  user      : 'root',
  password  : '',
  port: 3306,
  database: 'edusys',
  multipleStatements: true
```

```
});  
  
pool.query('select * from student', function(error, rows){  
    console.log(rows);  
});
```

- 数据库操作

格式: <数据库>.query(sql,callback)

- 增 insert into <表名> [(<字段名1>[,...<字段名n >])] values (值1)[, (值n)];

```
insert into MyGuests (firstname, lastname, email)  
values ('John', 'Doe', 'john@example.com');
```

- 删 delete from <表名> where <条件>

```
--删除MyGuests表中id为1的数据  
DELETE FROM MyGuests where id=1;  
  
--删除所有数据  
DELETE FROM MyGuests
```

- 改 update <表名> set 字段=新值,... where 条件

```
update MyGuests set name='Mary' where id=1;
```

- 查 select <字段1, 字段2, ...> from <表名> where < 表达式 >

```
--查看表 MyGuests 中所有数据
select * from MyGuests;

--查看表 MyGuests 中前10行数据:
select * from MyGuests order by id limit 0,10;
```

- 条件控制语句

- WHERE 语句:

```
SELECT * FROM tb_name WHERE id=3;
```

- 相关条件控制符:

- =、>、<、<>、IN(1,2,3.....)、BETWEEN a AND b
 - AND、OR、NOT
 - LIKE用法中
 - % 匹配任意、
 - _ 匹配一个字符 (可以是汉字)

- LIMIT idx,qty: 数量控制

- SELECT * FROM goods LIMIT 2,5

- IS NULL 空值检测

- 排序ORDER BY

- asc 升序 (默认)
 - desc 降序

封装模块

```
module.exports = {
  select: function(tsql, callback){
    pool.query(tsql, function(error, rows){
```

```
        if(rows.length > 1){
            callback({rowCount: rows[1][0]['rowCount'], data: rows[0]});
        } else {
            callback(rows);
        }
    });
}
```

MongoDB

MongoDB是一个基于分布式文件存储的数据库，由C++语言编写，旨在为WEB应用提供可扩展的高性能数据存储解决方案，是一个介于关系数据库和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的。它支持的数据结构非常松散，是类似json的bson格式

bson:是一种类json的一种二进制形式的存储格式，简称Binary JSON

下载与安装

- 下载地址: <https://www.mongodb.com/download-center/community>
- 安装路径尽量简单，不要有中文

配置数据库

- 配置数据库保存目录

```
mongod.exe --dbpath D:\data\mongodb\db
```

启动成功后，可通过 <http://127.0.0.1:27017> 访问

- 配置为 windows 服务

```
systemLog:
  destination: file
  path: d:\data\mongodb\log\mongod.log
storage:
  dbPath: d:\data\mongodb\db
```

- 命令行执行

```
mongod.exe --config c:\mongodb\mongod.cfg --service --serviceName MongoDB --install
```

- 配置环境变量

安装mongodb默认自动配置环境变量，方便在命令行中使用相关命令

连接数据库

- mongo 连接到数据库并进行操作
- mongod 显示数据库信息

常用命令

输入help可以看到基本操作命令

数据库操作(Database)

- 查看所有数据库: `show dbs`
- 查看当前使用的数据库
 - `db.getName()`
 - `db`
- 显示当前db状态: `db.stats()`

- 查看当前db的连接地址: `db.getMongo()`
- 创建/切换数据库: `use DBNAME`
如果数据库不存在, 则创建数据库, 否则切换到指定数据库。
- 删除当前使用数据库: `db.dropDatabase()`

集合操作(Collection)

利用`use DBNAME` 切换到当前数据库后, 可以进行集合与文档的操作

- 查询所有集合:
 - `db.getCollectionNames()`
 - `show collections`
- 创建集合:
 - `db.createCollection('集合名');`
PS:只有创建了集合, 数据库才能成功创建
- 删除集合:
 - `db.集合.drop();`

文档操作(Document)

文档就是数据, 这里的所有操作都是针对数据 格式: `db.集合.方法()`

- 增 (插入数据) :
 - `insert()`
 - `insertOne()`
 - `insertMany()`
- 删 (删除数据)

- deleteOne()
- deleteMany()
- 改 (更新数据)
 - updateOne(query,newData)
 - updateMany()

```
//更新指定字段
//查找name属性为tiantian的数据，并更新age属性为27
db.user.updateOne({name:'tiantian'},{$set:{age:27}})
```

- 查 (查询数据) :
 - 查询所有: `db.集合.find()`
 - 按条件查询 (支持多条件) : `db.集合.find(条件)`
 - 查询第一条 (支持条件) : `db.集合.findOne(条件)`

```
//查询user下所有数据
db.user.find();

// 查询user下年龄为38的
db.user.find({age:38})

// 查询user下年龄大于38的
db.user.find({age:{gt:38}})
```

条件与筛选

- 限制数量: `db.表名.find().limit(数量);`

- 跳过指定数量: `db.表名.find().skip(数量)`
- 比较查询
 - 大于: `$gt`
 - 小于: `$lt`
 - 大于等于: `$gte`
 - 小于等于: `$lte`
 - 非等于: `$ne`

NodeJS中使用mongodb

- 安装mongodb模块

```
npm install mongodb
```

- 连接mongoDB
 - 默认地址: `mongodb://localhost:27017`
如果数据库不存在, MongoDB 将创建数据库并建立连接。

```
//引入模块
const mongodb = require('mongodb');
const MongoClient = mongodb.MongoClient;

//连接MongoDB并连接数据库laoxie, 无则自动创建
MongoClient.connect("mongodb://localhost:27017/laoxie", function(err, database) {
  if(err) throw err;

});
```

- 使用/创建数据库


```
const mongodb = require('mongodb');
const MongoClient = mongodb.MongoClient;

//使用数据库也可以使用以下方式
//1.连接mongoDB
MongoClient.connect("mongodb://localhost:27017", function(err, database) {
  if(err) throw err;
  // 连接数据库，无则自动创建
  let db = database.db('laoxie');
});
```

- 创建集合:createCollection()

格式: db.createCollection(name, options)

- 使用集合collection()

db.collection(name)

- 删除集合: drop()

格式: db.COLLECTION_NAME.drop(callback)

```
dbase.createCollection('site', function (err, res) {
  if (err) throw err;
  console.log("创建集合!");
  db.close();
});
```

MongoDB的导入导出

- 导出mongoexport 把一个collection导出成JSON格式或CSV格式的文件。可以通过参数指定导出的数据项，也可以根据指定的条件导出数据

- 格式: `mongoexport -d dbname -c collectionname -o file --type json/csv -f field`

- 参数说明:

- -d : 数据库名
- -c : collection名
- -o : 输出的文件名
- --type : 输出的格式, 默认为json
- -f : 输出的字段, 如果-type为csv, 则需要加上-f "字段名"

```
mongoexport -d mytest -c goods -o D:/data/goods.json --type json -f "_id,name,price,img_url,add_time"
```

- 导入mongoimport

- 格式: `mongoimport -d dbname -c collectionname --file filename --headerline --type json/csv -f field`

- 参数说明:

- -d : 数据库名
- -c : collection名
- --type : 导入的格式默认json
- -f : 导入的字段名
- --headerline : 如果导入的格式是csv, 则可以使用第一行的标题作为导入的字段
- --file : 要导入的文件

```
mongoimport -d mongotest -c goods --file D:/data/goods.json --type json
```

MongoDB备份与恢复

- 备份

- 格式: `mongodump -h dbhost -d dbname -o dbdirectory`

- 参数说明:

- -h: MongoDB所在服务器地址, 例如: 127.0.0.1, 当然也可以指定端口号: 127.0.0.1:27017
- -d: 需要备份的数据库实例, 例如: test
- -o: 备份的数据存放位置, 例如: D:/mongodump/

当然该目录需要提前建立, 这个目录里面存放该数据库实例的备份数据。

```
mongodump -h 127.0.0.1:27017 -d mytest -o D:/mongodump/
```

- 恢复

- 格式: `mongorestore -h dbhost -d dbname --dir dbdirectory`

- 参数或名:

- -h: MongoDB所在服务器地址
- -d: 需要恢复的数据库实例, 例如: test, 当然这个名称也可以和备份时候的不一样, 比如test2
- --dir: 备份数据所在位置, 例如: D:/mongodump/
- --drop: 恢复的时候, 先删除当前数据, 然后恢复备份的数据。就是说, 恢复后, 备份后添加修改的数据都会被删除, 慎用!

```
mongorestore -h 192.168.17.129:27017 -d mytest --dir D:/mongodump/
```

【案例】

- 封装数据的增删改查
- 登录/注册页面的实现

- 利用token保持登录状态