# 一 1 Node连接MySql

连接：let db=mysql.createConnection({host, port, user, password, database});

查询：db.query('干啥', (err, data)=>{});

let db=mysql.createConnection({配置});

let db=mysql.createPool({配置});

如果只有一个连接，那么如果有一个数据库操作很慢，会影响项目的速度，所以需要使用连接池：

const express = require('express');  
const mysql = require('mysql');  
  
let mysqlDB = mysql.createPool({  
 host:'localhost',  
 post:3306,  
 user:'root',  
 password:'root',  
 database:'suibian'  
});  
  
let app = express();  
  
app.get('/api',function (req,res) {  
 let query = 'select \* from students';  
 mysqlDB.query(query,function (err,data) {  
 let name = data[0].name;  
 res.send(name);  
 });  
});  
  
app.listen(3000);

## 2 代码示例

前台：

<form action="http://localhost:3000/api">  
 <!-- 输入名字，点击查询，就会显示他的年纪 -->  
 <input type="text" name="name" id="name">  
 <input type="submit" value="查询"><br>  
</form>

后台：

const express = require('express');  
const mysql = require('mysql');  
  
let mysqlDB = mysql.createPool({  
 host:'localhost',  
 post:3306,  
 user:'root',  
 password:'root',  
 database:'suibian'  
});  
  
let app = express();  
  
app.get('/api',function (req,res) {  
  
 let query = 'select age from students where name="' + req.query.name + '"';  
 let newQ = `select age from students where name='${req.query.name}'`;  
 console.log(query);  
 mysqlDB.query(newQ,function (err,data) {  
 let age = data[0].age + '';  
 res.send(age);  
 });  
});  
  
app.listen(3000);

## 4完整的用户登录注册

### 4.1 前台

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <title></title>  
 <script src="jquery.js" charset="utf-8"></script>  
 <script>  
 $(function () {  
*//注册* $('#btn\_reg').click(function () {  
 $.ajax({  
 url: '/reg',  
 data: {user: $('#user').val(), pass: $('#pass').val()},  
 dataType: 'json',  
 success(json) {  
 if (json.err) {  
 alert('注册失败' + json.msg);  
 } else {  
 alert('注册成功');  
 }  
 },  
 error(err) {  
 alert('失败');  
 }  
 });  
 });  
  
*//登陆* $('#btn\_login').click(function () {  
 $.ajax({  
 url: '/login',  
 data: {user: $('#user').val(), pass: $('#pass').val()},  
 dataType: 'json',  
 success(json) {  
 if (json.err) {  
 alert('登陆失败' + json.msg);  
 } else {  
 alert('登陆成功');  
 }  
 },  
 error(err) {  
 alert('失败');  
 }  
 });  
 });  
 });  
 </script>  
</head>  
<body>  
用户：<input type="text" id="user"/><br>  
密码：<input type="password" id="pass"/><br>  
<input type="button" value="注册" id="btn\_reg">  
<input type="button" value="登陆" id="btn\_login">  
</body>  
</html>

### 4.2 后台

const http = require('http');  
const mysql = require('mysql');  
const fs = require('fs');  
const url = require('url');  
const zlib = require('zlib');  
const crypto = require('crypto');  
  
const \_key = 'sadfslekrtuew5iutoselgdtjiypoydse4ufhs.edtyo;s8te4arfeliawkfhtsie5tlfia;sefdshroiupeoutwyeli5gurse;ihf';  
  
function md5(str) {  
 let obj = crypto.createHash('md5');  
  
 obj.update(str);  
  
 return obj.digest('hex');  
}  
  
function md5\_2(str) {  
 return md5(md5(str) + \_key);  
}  
  
let db = mysql.createPool({host: 'localhost', port: 3309, user: 'root', password: '', database: '20180127'});  
  
let server = http.createServer((req, res) => {  
 let {pathname, query} = url.parse(req.url, true);  
 let {user, pass} = query;  
  
 switch (pathname) {case '/reg':if (!user) {  
 res.write('{"err": 1, "msg": "username can\'t be null"}');  
 res.end();  
 } else if (!pass) {  
 res.write('{"err": 1, "msg": "password can\'t be null"}');  
 res.end();  
 } else if (!/^\w{4,16}$/.test(user)) {  
 res.write('{"err": 1, "msg": "username is invaild"}');  
 res.end();  
 } else if (/['|"]/.test(pass)) {  
 res.write('{"err": 1, "msg": "password is invaild"}');  
 res.end();  
 } else {  
 db.query(`SELECT \* FROM user\_table WHERE username='${user}'`, (err, data) => {  
 if (err) {  
 res.write('{"err": 1, "msg": "database error"}');  
 res.end();  
 } else if (data.length > 0) {  
 res.write('{"err": 1, "msg": "this username exsits"}');  
 res.end();  
 } else {  
 db.query(`INSERT INTO user\_table (ID,username,password) VALUES(0,'${user}','${md5\_2(pass)}')`, (err, data) => {  
 if (err) {  
 res.write('{"err": 1, "msg": "database error"}');  
 res.end();  
 } else {  
 res.write('{"err": 0, "msg": "success"}');  
 res.end();  
 }  
 });  
 }  
 });  
 }  
 break;  
 case '/login':if (!user) {  
 res.write('{"err": 1, "msg": "username can\'t be null"}');  
 res.end();  
 } else if (!pass) {  
 res.write('{"err": 1, "msg": "password can\'t be null"}');  
 res.end();  
 } else if (!/^\w{4,16}$/.test(user)) {  
 res.write('{"err": 1, "msg": "username is invaild"}');  
 res.end();  
 } else if (/['|"]/.test(pass)) {  
 res.write('{"err": 1, "msg": "password is invaild"}');  
 res.end();  
 } else {  
 db.query(`SELECT \* FROM user\_table WHERE username='${user}'`, (err, data) => {  
 if (err) {  
 res.write('{"err": 1, "msg": "database error"}');  
 res.end();  
 } else if (data.length == 0) {  
 res.write('{"err": 1, "msg": "no this user"}');  
 res.end();  
 } else if (data[0].password != md5\_2(pass)) {  
 res.write('{"err": 1, "msg": "username or password is incorrect"}');  
 res.end();  
 } else {  
 res.write('{"err": 0, "msg": "success"}');  
 res.end();  
 }  
 });  
 }  
 break;  
 default:  
let rs = fs.createReadStream(`www${pathname}`);  
 let gz = zlib.createGzip();  
 res.setHeader('content-encoding', 'gzip');  
 rs.pipe(gz).pipe(res);  
 rs.on('error', err => {  
 res.writeHeader(404);  
 res.write('Not Found');  
 res.end();  
 });  
 }  
});  
server.listen(8080);

# 三 Node.js操作MongoDB

需要安装操作MongoDB的驱动：npm install mongodb

const MongoClient = require('mongodb').MongoClient;  
const MongoURL = 'mongodb://localhost:27017/';  
function \_connectDB(callback) {  
 MongoClient.connect(MongoURL,function (err,db) {  
 if(err){  
 console.log('Mongo未连接');  
 callback(err,'Mongo未连接');  
 return;  
 }  
 console.log('Mongo已连接');  
 callback(null, db);  
 });  
}  
  
*//插入一个数据*exports.insertOne = function (dbname,collectionName,json,callback) {  
 \_connectDB(function (err,db) {  
 if(err){  
 throw err;  
 return;  
 }  
 let dbObj = db.db(dbname);  
 dbObj.collection(collectionName).insertOne(json, function (err, result) {  
 if (err) {  
 throw err;  
 db.close();  
 return;  
 }  
 callback(null, result);  
 db.close();  
 });  
 });  
};

使用：

const db = require('./db');  
db.insertOne('testdb','teacher',{'name':'lisi'},function (err,result) {  
 if(err){  
 console.log(err);  
 return;  
 }  
 console.log('sucess');  
});

let mongoose = require('mongoose');  
mongoose.connect('mongodb://localhost:27017/test');  
  
let Person = mongoose.model('Person',{  
 name:String,  
 age:Number  
});  
  
let p = new Person({  
 name:'tom',  
 age:30  
});  
  
/\*p.save(function (err) {  
 console.log('存储成功');  
});\*/  
  
/\*  
Person.find({  
 'name':'tom'  
},function (err,result) {  
 let p1 = result[0];  
 p1.age = 28;  
 p1.save(function (err) {  
 console.log('修改成功');  
 });  
});\*/  
  
*//推荐写法*(async function () {  
 let rs = await Person.find({'name':'tom'});  
 if(rs){  
 let p1 = rs[0];  
 p1.age = 15;  
 p1.save(function (err) {  
 console.log('修改成功');  
 });  
 } else {  
 console.log('没找到数据');  
 }  
})();

schema：

let mongoose = require('mongoose');  
mongoose.connect('mongodb://localhost:27017/test');  
  
let db = mongoose.connection;  
db.once('open', function (callback) {  
 console.log("数据库成功打开");  
});  
  
*//博客的结构*var blogSchema = new mongoose.Schema({  
 title: String,  
 author: String,  
 body: String,  
 comments: [{ body: String, date: Date }],  
 date: { type: Date, default: Date.now },  
 hidden: Boolean,  
 meta: {  
 votes: Number,  
 favs: Number  
 }  
});  
  
blogSchema.methods.showInfo = function(){  
 console.log(this.title);  
};  
  
let Blog = mongoose.model('Blog', blogSchema);  
  
let blog = new Blog({  
 "title" : "博客测试",  
 "author" : "考拉"  
});  
  
blog.save();  
blog.showInfo();

# 四 ORM框架 sequelize

sequelize框架需要额外安装mysql2 包

npm install mysql2 --save

npm install sequelize --save

const Sequelize = require('sequelize');  
const mysqlConfig = {  
 database: 'lib1', *// 使用哪个数据库* username: 'root', *// 用户名* password: 'root', *// 口令* host: 'localhost', *// 主机名* port: 3306 *// 端口号，MySQL默认3306*};  
let sequelize = new Sequelize(

mysqlConfig.database,

mysqlConfig.username,

mysqlConfig.password,

{  
 host: mysqlConfig.host,  
 dialect: 'mysql',  
 pool: {  
 max: 5,  
 min: 0,  
 idle: 30000  
 }  
});  
  
*//sequelize与数据库直连  
// (async ()=>{  
// let table = await sequelize.query("SELECT \* FROM users");  
// console.log(table);  
// })();  
  
//ORM*let User = sequelize.define('user', {

*//sequelize会给user自动添加s后缀* id: {  
 type: Sequelize.STRING(50),  
 primaryKey: true  
 },  
 name: Sequelize.STRING(100),  
 age: Sequelize.STRING(10),  
}, {  
 timestamps: false *//关闭sequelize默认的timestamp的功能*});  
  
let name = 'ha';  
let age = 22;  
(async ()=>{  
 let dog = await User.create({  
 id:0,  
 name:name,  
 age:age  
 });  
})();