华北水利水电大学

**课 程 设 计**

（数据库系统）

**专 业： 软件工程**

**班 级： 2017208**

**姓 名： 李乐飞**

**学 号： 201720812**

**指导教师： 翟利红**

二○一九年 十二月 二十二日

**目 录**

第一章 系统概述………………………………………………………1

第二章 系统分析………………………………………………………

第三章 功能分析………………………………………………………

第四章 系统实现………………………………………………………

第五章 调试过程中的问题及系统测试情况…………………………

第六章 总结………………………………………………………

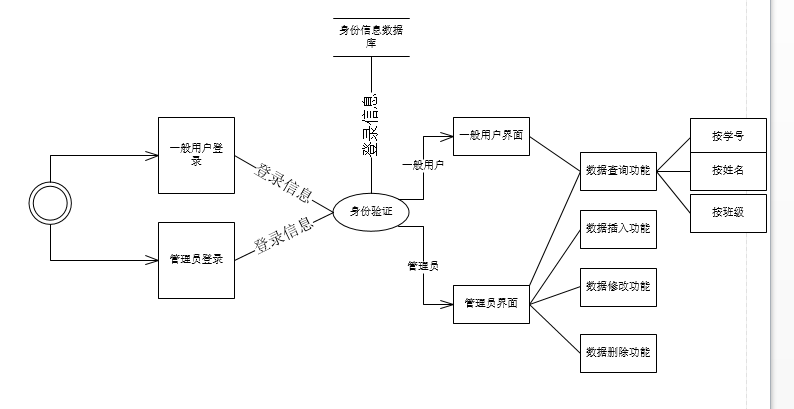
第一章 概述

前端界面采用html、css、JavaScript等，后台在linux上使用node.js，数据库用mysql8。前端页面有登录页面（包括管理员登录和普通用户登录）和数据管理界面共三个界面，基本功能模块包括：数据库连接模块、用户登录模块、数据操作模块。

用户在登录界面登录后，登录信息（用户名、密码、是否是管理员）被送到服务器，再被送到数据库，将登录信息和数据库中的登录信息进行比较，登录成功后，普通用户可进行数据查询功能，管理员除了可以进行查询外还可以，直接向数据库中插入数据，修改数据，删除数据。

第二章 系统分析

数据流图：



数据字典：

登录信息：用户名和密码。

**数据存储**

名称：管理员信息表 组成：姓名、密码 关键字：姓名 说明：存储管理员信息

名称：学生基本信息表 组成：学号、姓名、性别、出生日期、班级、专业、系别 关键字：学号、姓名

说明：登记学生信息

名称：课程信息表 组成：序号、课程号、课程名、学时、学分。

关键字：课程号、课程名 说明：存储课程信息

名称：成绩信息表 组成：学号、课程号、课程名、成绩、补考成绩 关键字：学号、课程号 说明：存储学生各科成绩信息

**管理员信息表（admin）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 含义 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| name | 姓名 | varchar（20） |  | 否 |  | 主键 |
| password | 密码 | varchar（20） |  | 否 |  |  |

**学生基本信息表（studentIformation）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 含义 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| studentID | 学号 | varchar(9) |  | 否 |  | 唯一标识学生，不能重复，主键 |
| gender | 性别 | enum |  | 否 |  |  |
| dateOfBirth | 出生日期 | YEAR |  | 是 |  |  |
| class | 班级 | varchar（7） |  | 是 |  |  |
| major | 专业 | varchar(20) |  | 是 |  |  |
| department | 系别 | varchar(20) |  | 是 |  |  |

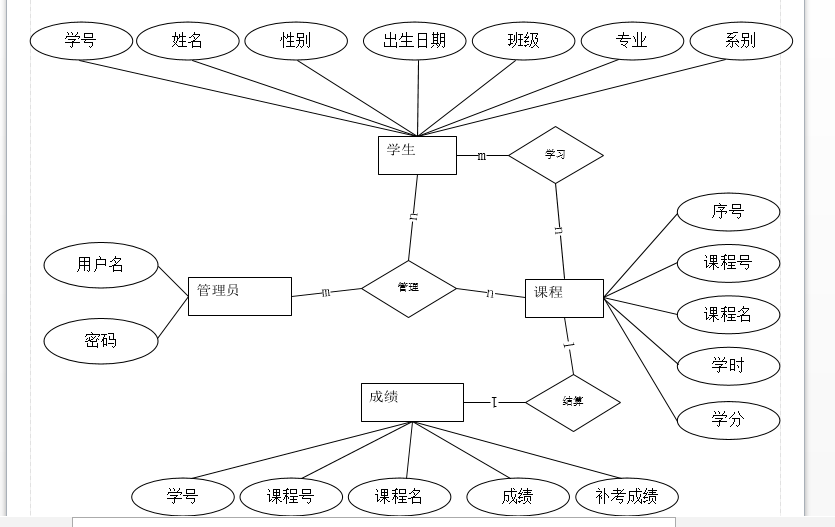
**课程信息表（course）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 含义 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| serialNumber | 序号 | varchar(5) |  | 否 |  |  |
| courseNumber | 课程号 | varchar(5) |  | 否 |  | 主键 |
| courseName | 课程名 | varchar(30) |  | 否 |  | 主键 |
| classHour | 学时 | varchar(5) |  | 是 |  |  |
| credit | 学分 | varchar(5) |  | 是 |  |  |

**成绩信息表（score）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 含义 | 数据类型 | 默认值 | 允许非空 | 自动递增 | 备注 |
| studentID | 学号 | varchar(9) |  | 否 |  | 主键 |
| courseNumber | 课程号 | varchar(5) |  | 否 |  |  |
| courseName | 课程名 | varchar(30) |  | 否 |  | 主键 |
| score | 成绩 | varchar(5) |  |  |  |  |
| makeUpScore | 补考成绩 | varchar(5) |  |  |  |  |

ER图：



关系模式：

管理员（姓名、密码）

学生（学号、姓名、性别、出生日期、班级、专业、系别）

课程（序号、课程号、课程名、学时、学分）

成绩（学号、课程号、课程名、成绩、补考成绩）

成绩信息表是学生信息表和课程信息表的字表，学号、课程号、课程名是外键。

第三章 功能分析

前端

用网页实现，实现三个界面，包括登录界面、一般用户管理界面、管理员管理界面。

登录界面

用单选框来区分一般用户和管理员，网页用按钮、文本域等实现，用css美化相关控件，用JavaScript实现输入信息的发送到服务器等。

一般用户查询页面

一般用户和管理员登录之后的功能不同，所以显示的页面也有差异，在这里将管理员的管理界面去除掉增删改功能的按钮之后就是一般用户的管理界面。页面只给一般用户提供查询的功能，用户先选择需要查询的表，再选择查询条件并输入相应参数，之后点击查询按钮。

本页面会把查询参数传到服务器，之后由服务器处理。页面在创建的时候会与服务器用socket.io建立一个tcp连接，当用户点击查询按钮之后，会触发对应的事件，将参数从这个连接发到服务器，服务器处理后返回结果，前端重新渲染表格，实现查询功能。

管理员管理页面

###### 查询功能

管理员的查询功能和一般用户的查询功能一样。

###### 插入功能

用户点击插入按钮之后，会在表头下面产生一个空行，让用户输入插入的数据，之后数据会被封装成一个对象传到服务器。

###### 修改功能

用户可以直接在查询结果的表格里修改数据，之后将表的名称、这条数据的主键、改变的数据提交到服务器。

###### 删除功能

直接删除用户光标所在的行，用户点击删除之后会将表的名称、这条数据的主键、改变的数据发送到服务器。

服务器

服务器语言使用node.js。

###### 登录功能

当服务器收到前端发来的登录数据之后，会将数据和服务器中的数据对比，如果登录信息正确，就返回对应的界面，否则给出错误提示。

##### 查询功能

根据参数中表的名称、需要查询的参数生成对应的sql语句，对数据库进行操作，将从数据库返回的数据封装成对象，传递到前端。

##### 插入功能

当服务器收到需要插入的数据之后，会生成sql语句，并在数据库中执行插入操作。

##### 修改功能

根据参数中表的名称、这条数据的主键、改变的数据生成对应的sql语句，实现修改操作，目前实现一次只能修改一行。

###### 删除功能

根据参数中表的名称、这个字段的主键来生成对应的sql语句，之后对数据库进行操作。

第四章 系统实现

#### 数据库实现：

重置数据库的sql语句如下：

DROP DATABASE studentDataManager;

CREATE DATABASE IF NOT EXISTS studentDataManager DEFAULT CHARACTER SET 'UTF8';

USE studentDataManager;#创建并使用数据库

#drop TABLE admin;

CREATE TABLE admin(

name VARCHAR(20) NOT NULL,

password VARCHAR(20) NOT NULL,

isAdmin CHAR(1) NOT NULL,

PRIMARY KEY(name)

);#创建管理员信息表（admin）

INSERT INTO admin VALUES ('admin1','123456','1');#创建一个管理员用户账户

INSERT INTO admin VALUES ('user1','123456','0');#创建一个非管理员用户账户

#DROP TABLE studentIformation

CREATE TABLE studentIformation(

studentID VARCHAR(9) NOT NULL,

gender ENUM('男','女','保密') NOT NULL,

dateOfBirth varchar(8),

class VARCHAR(7),

major VARCHAR(20),

department VARCHAR(20),

PRIMARY KEY(studentID)

);#创建学生基本信息表（studentIformation）

#DROP TABLE course

CREATE TABLE course (

serialNumber SMALLINT UNIQUE,

courseNumber SMALLINT,

courseName VARCHAR(30) NOT NULL ,

classHour varchar(5),

credit VARCHAR(5),

PRIMARY KEY(courseNumber,courseName)

);#创建课程信息表（course）

#DROP TABLE score;

create TABLE score(

studentID VARCHAR(9) NOT NULL ,

courseNumber SMALLINT NOT NULL,

courseName VARCHAR(30) NOT NULL,

score VARCHAR(5),

makeUpScore VARCHAR(5),

#PRIMARY KEY(studentID,courseName),

foreign KEY (studentID) REFERENCES studentIformation (studentID),

foreign KEY (courseNumber,courseName) REFERENCES course (courseNumber,courseName)

);#创建成绩信息表（score）

INSERT INTO studentIformation

VALUES ('201720812', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation

VALUES ('201720813', '男', '19980221','2017208','计科','信息工程');

INSERT INTO studentIformation

VALUES ('201720817', '男', '19980221','2017208','软件工程','信息工程');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit)

VALUES(2,2,'计算机网络','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit)

VALUES(1,1,'java程序设计','16','1');#创建课程信息表的一些初始数据

INSERT INTO score

VALUES('201720812','2','计算机网络','100','');

INSERT INTO score

VALUES('201720812','1','java程序设计','90','');#创建一些成绩表的基本数据

为数据库添加大量数据的sql语句如下：

INSERT INTO studentIformation VALUES ('201720701', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720702', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720703', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720704', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720705', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720706', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720707', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720708', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720709', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720710', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720711', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720712', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720713', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720714', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720715', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720716', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720717', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720718', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720719', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720720', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720721', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720722', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720723', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720724', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720725', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720726', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720727', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720728', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720729', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO studentIformation VALUES ('201720730', '男', '19980221','2017208','软件工程','信息工程学院');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(3,3,'数据库2','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(4,4,'数据库3','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(5,5,'数据库4','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(7,7,'c语言1','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(8,8,'c语言2','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(9,9,'c语言3','48','3');

INSERT INTO course(serialNumber,courseNumber,courseName,classHour,credit) VALUES(10,0,'c语言4','48','3');

INSERT INTO score VALUES('201720711','1','java程序设计','100','');

INSERT INTO score VALUES('201720712','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720713','1','java程序设计','100','');

INSERT INTO score VALUES('201720714','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720715','1','java程序设计','100','');

INSERT INTO score VALUES('201720716','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720717','1','java程序设计','100','');

INSERT INTO score VALUES('201720718','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720719','1','java程序设计','100','');

INSERT INTO score VALUES('201720720','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720730','1','java程序设计','100','');

INSERT INTO score VALUES('201720729','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720728','1','java程序设计','100','');

INSERT INTO score VALUES('201720727','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720726','1','java程序设计','100','');

INSERT INTO score VALUES('201720725','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720724','1','java程序设计','100','');

INSERT INTO score VALUES('201720723','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720722','1','java程序设计','100','');

INSERT INTO score VALUES('201720701','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720702','1','java程序设计','100','');

INSERT INTO score VALUES('201720709','1','java程序设计','100','');

INSERT INTO score VALUES('201720708','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720707','1','java程序设计','100','');

INSERT INTO score VALUES('201720706','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720705','1','java程序设计','100','');

INSERT INTO score VALUES('201720704','2','计算机网络','90','');#创建一些成绩表的基本数据

INSERT INTO score VALUES('201720703','1','java程序设计','100','');

其中admin表中负责记录可以登录的用户的信息，和其它三个表没有依赖关系；score表的外键是studentInformation表的studentID列和course表的ourseNumber,courseName列。

#### 服务器实现：

服务器所部属的项目目录树如下：

.

├── connectDB.js

├── index.html

├── indexSendData.html

├── package.json

├── package-lock.json

├── public

│   ├── css

│   │   ├── index.css

│   │   ├── indexSendData.css

│   │   └── manager.css

│   ├── js

│   │   ├── indexSendData.js

│   │   ├── login.js

│   │   └── manager.js

│   ├── manager.html

│   └── managerOnlyQuery.html

└── server.js

文件的详细功能步骤细节请看文件中的注释。

server.js是服务器的主程序，在这里在这里直接收到前端发来的数据，直接与数据库建立连接，直接用socket与前端通信。在这个文件中，先用express将路由设置为静态路由，之后对从前端收到的data对象的operation的值的不同，转入不同的操作，包括对数据库的增删查改和用户的登录，并将操作执行的时间记录下来。

connectDB.js：这个文件里有五个函数，分别是对数据库的增删查改和对用户登录信息的验证的具体细节。这个文件里写了一些对数据库进行增删查改操作的函数和登录操作，对数据库操作的函数：包括拼接数据库的sql语句，处理操作之后返回的结果，操作数据库的函数都需要传递三个参数：在server.js中与数据库建立的连接，与客户端建立的socket连接，客户端发来的data。

package开头的两个文件记录了项目的初始化信息，和模块之间的依赖等等，主要用于项目部署到不同的地方的初始化。

#### 前端实现：

index.html：主页。实现了登录的功能，用户在点击登录后，会将登录信息封装成一个data对象，发送到后端进行验证，验证成功后，前端的window.location会发生变化。

indexSendData.html：这个文件用于在写后端的时候对接口的测试。

manager.html：管理员管理界面，有表名的单选按钮，和四种操作的四个按钮。

managerOnlyQuery.html:一般用户的管理页面，相比管理员的管理页面，少了增删改的功能。

主要的前端文件都放在public这个目录下面，css文件设置一些简单的样式。

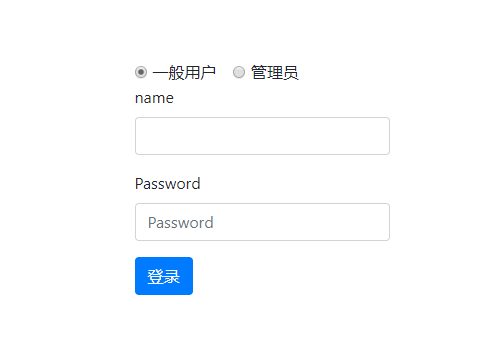
indexSendData.js：是indexSendData.html的js文件。

login.js：绑定主页的登录按钮，当用户点击登录之后，将data发往服务器并监听服务器的各种响应，比如url重定向，提示密码错误。

manager.js：这个文件是普通用户和管理员共用的js文件，产生和发送增删查改的数据，这个文件分别设置了管理页面四个按钮的单击事件，记录了界面的焦点发生变化是表格中的数据的改变和焦点的位置，同时监听服务器发送给自己的各种事件，比如：查询的结果，自己异常操作的提示，同时为了让页面在适当的时候刷新，还允许服务器让前端执行函数。

第五章 调试过程中的问题及系统测试情况

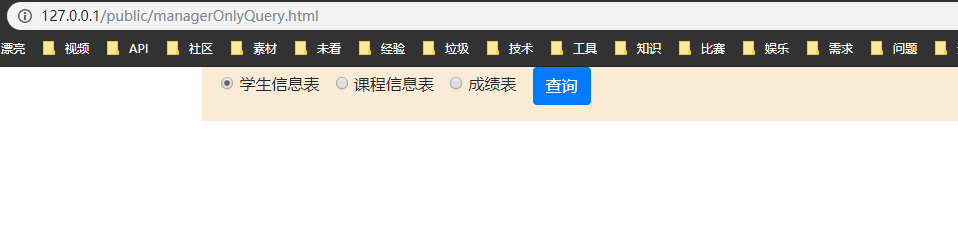
#### 系统测试

 如果服务器是在本地开的，在浏览器中输入127.0.0.1，浏览器返回如下界面：

如果输入的信息有误会进行如下提示：



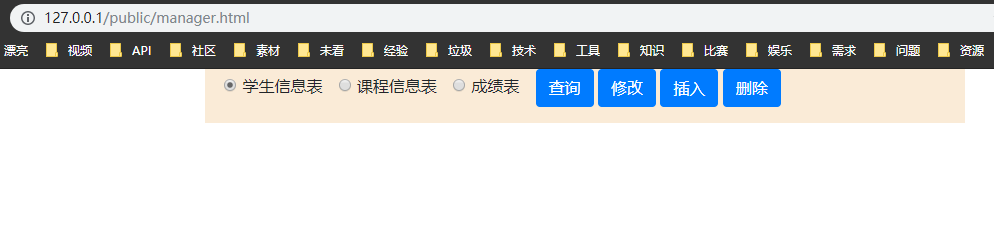
如果一般用户登录成功会进入如下界面：



点击查询后得到如下结果：

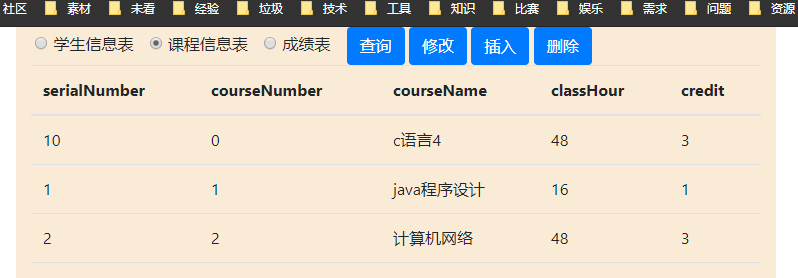


如果是以管理员的身份进入管理页面，管理页面如下：



点击查询：

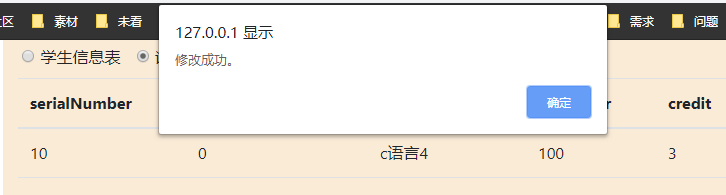


选择课程信息表，点击查询：

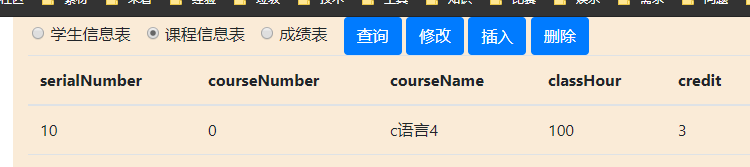
不修改数据的情况下点击修改：



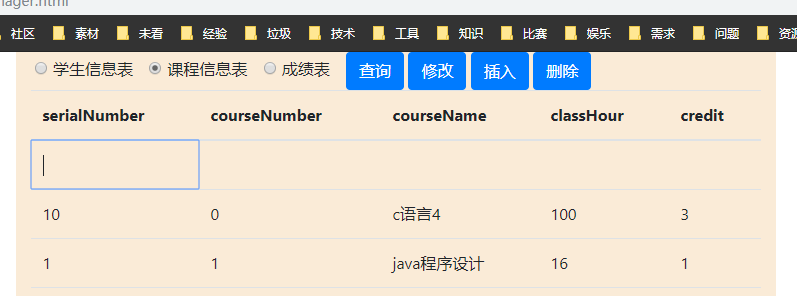
点击c语言4的学时单元格，将c语言4的学时修改为100，之后点击修改：



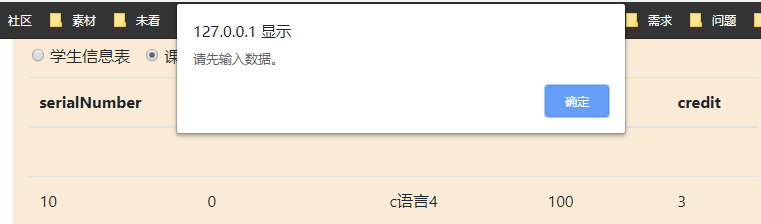
再次点击查询，发现修改成功：



之后点击插入按钮，表头下面多出一个空行，点击空行里面的单元格，显示如下：

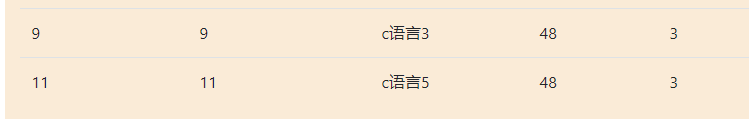


不输入数据的情况下点击插入：

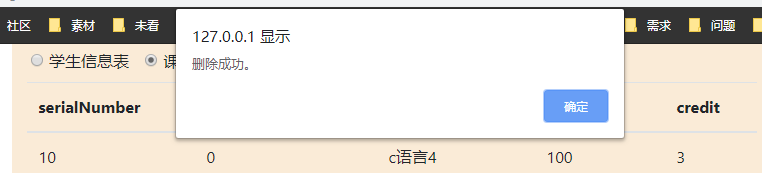


如果再空白单元格中输入了我们自定义的c原因5的信息，点击插入，结果如下：

点击确定，可以看到，c语言5的信息被插入了：



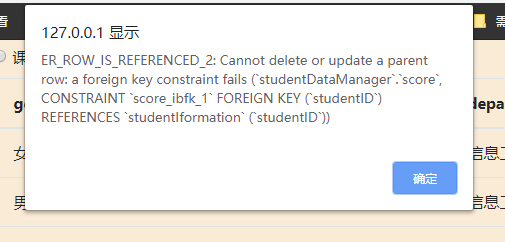
在c语言5的字段上单击，点击删除，提示如下：



点击确定后，发现c语言5的信息已经被删除了：



如果上面进行的操作不符合数据库设计的要求（数据的唯一性，外键依赖等），会返回数据库的报错信息，例如：我尝试删除学号为201720812学生信息，因为分数表里有这个学生的分数信息依赖这个字段，所以报错如下：



###### 调试过程中遇到的问题：

没有考虑到的操作方式导致服务器异常，现在修复了一部分。

多个用户同时执行操作时，其它用户收到本应发给特定用户的信息，解决方法可以是让没必要广播的信息不进行广播。

第六章 结束语

基本的增删查改和登录功能已经实现，不过感觉自己花了五分之二的时间在写前端的逻辑，处理用户的异常操作。设计数据库的时间花的不多。

这个文档中系统实现之前的内容都是在编码完成的，但是编码完成后在回过头来看自己当初的需求、想要实现的目标，发现现在只是实现了简单的功能，比如：修改、删除操作一次只能操作一行。

参考文献:无

[1]

[2]

[3]