

**数据库系统开发实验**

**班 号：**

**学 号：1120190699**

**姓 名：**

**日 期：2021年9月15日**

1. **实验方法**
   1. **开发环境**

操作系统：Windows10

数据库管理系统：Microsoft SQL Server Management Studio 18

编程工具：Visual Studio2019

编程语言：C#

* 1. **回答问题**

1. **使用哪种数据提供程序？**

来自数据库的数据源。使用数据集类型的数据库模型。

1. **使用的数据连接对象是哪一个？连接对象是如何建立的？最后生成的连接对象中的连接字符串是什么？代表什么含义？**

使用的数据连接对象：SQLConnection类的对象

连接对象如何建立：使用命名空间System.Data.SqlClient，新建SqlConnection的实例，初始化连接字符串之后调用Open方法。

连接字符串：server=LAPTOP-7O733H1T\BITPC; database=LessonChoice ; uid=sa ; pwd=supreme,

代表的含义：数据源（数据库服务器）是LAPTOP-7O733H1T\BITPC；默认数据库是LessonChoice；用户名和密码是sa和supreme,

1. **使用的数据适配器对象是什么？其中的查询或更新语句是什么？如果有参数则参数是如何处理的？**

使用的数据适配器对象：SqlDataAdapter类的对象，它是通过SqlCommand对象来执行Sql语句，从而实现查询、更新功能的

查询语句举例：SqlDataAdapter da = new SqlDataAdapter;

da.SelectCommand = new SqlCommand("select \* from user", conn);

参数处理：为每一个参数创建一个SqlParameter类对象，然后将其添加到该SqlCommand的参数集合中。

SqlParameter parameter = new SqlParameter("@Student", StudentID);

da.SelectCommand.Parameters.Add(parameter);

1. **使用的数据集对象是什么？数据集中有哪些数据表？数据表是由哪些适配器对象生成的？（或采用其它方法）。**

使用的数据集对象：DataSet类的对象，数据表是DataTable类的对象。但是由于整个窗体中只需要“可选课程列表”、“已选课程表”两个DataTable，就没有为它们再创建数据集。

数据表由哪些适配器对象生成：在函数SelectDataTable中，SqlDataAdapter类的对象sda生成了“可选课程列表”、“已选课程表”两个DataTable，使用的是sda.Fill(dt);

* 1. **数据库搭建**

学生表Student

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段代码 | 数据类型 | 主键/外键 | 备注 |
| 学号 | ID | int | PK | NOT NULL |
| 学生姓名 | name | varchar(255) |  |  |
| 学生性别 | gender | varchar(10) |  | female/male |
| 学生班级 | class | int |  | NOT NULL |

教师表Teacher

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段代码 | 数据类型 | 主键/外键 | 备注 |
| 教师工号 | ID | int | PK | NOT NULL |
| 教师姓名 | name | varchar(255) |  | NOT NULL |

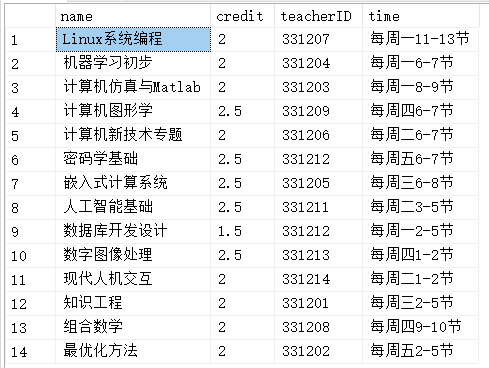
课程表Lessons

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段代码 | 数据类型 | 主键/外键 | 备注 |
| 课程名称 | name | varchar(1024) | PK | NOT NULL |
| 学分 | credit | float |  | NOT NULL |
| 授课教师 | teacherID | int | FK | 引用教师.教师工号 |
| 授课时间 | time | varchar(1024) |  |  |

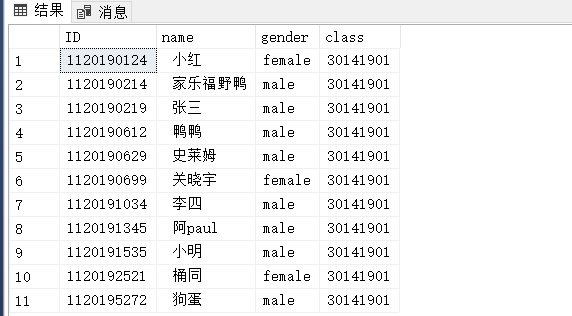
学生选课表TakeLesson

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段代码 | 数据类型 | 主键/外键 | 备注 |
| 学生学号 | StudentID | int | FK | NOT NULL  引用学生.学号 |
| 课程名称 | Lessonname | varchar(1024) | FK | NOT NULL  引用课程.课程名称 |

人工插入数据后，表格类似下图：







* 1. **C#代码编写**

在Visual Studio中，使用“WindowsFormsApp”项目模板，一共制作了四个.cs文件和一个窗体设计界面。其中四个.cs文件分别是程序主运行函数文件、窗体设计的对应代码逻辑文件、专门用来存放sql语句相关操作的类文件、专门用来存放控件相关操作函数的类文件。下面将对他们分别进行介绍。

**（1）程序主运行函数文件ProgramRun.cs**

此文件是应用程序的主入口点，会在每次运行程序时生成一个新的主窗口。

**（2）专门用来存放sql语句相关操作的类文件SqlWarehouse.cs**

此文件中封装了三个函数。

StringtoSqlCommand用于将输入的字符串转化成SqlCommand类的对象并返回。其中主要是设置SqlCommand的Connection属性、CommandType属性和CommandText属性。

SelectDataTable利用输入的SqlCommand对象获得数据适配器，返回与该数据适配器绑定的数据源。

ExecuteScalarStringSql用于执行输入的字符串这一查询类sql语句，并返回一个object类的对象，其中存放的是该sql语句的查询结果。执行方法是首先调用本文件中的StringtoSqlCommand函数返回一个sqlcommand，然后使用sqlcommand.ExecuteScalar()，返回此object对象。

**（3）专门用来存放控件相关操作函数的类文件Fun.cs**

此文件中封装了四个函数。

SetDataGridViewStyle用于在指定的DataGridView中添加第x列“选课”或者添加第x列“退课”，并且设置此列的字体、居中等样式属性。其中主要是使用了dgv.Columns.Add(列代码,HeaderText)这个函数，设置了dgv.Columns[x].DefaultCellStyle的Alignment、NullValue、ForeColor、Font属性，再使用dgv.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill用数据表填满DataGridView控件。

MoveRow用于在一个DataTable中删除第x行，并将此行移动到另一个DataTable中。使用dtfrom.Rows[x].ItemArray获得一个object类的对象，将这个object对象添加到另一个DataTable中。

Count用于计算指定DataGridView中第x列每一行的总和。实现逻辑是通过foreach (DataGridViewRow dgvr in dgv.Rows)遍历此DataGridView的每一行，每一次循环都累加。

Admit用于将遍历指定的DataGridView中的每一行，将相应的数据插入到数据库的学生选课表TakeLesson中。主要是通过调用SqlWarehouse类中的StringtoSqlCommand函数，foreach循环，SqlParameter类和sqlcommand.ExecuteNonQuery()方法实现的。

**（4）窗体设计的对应代码逻辑文件MainWindow.cs**

此文件中包含了MainWindow主窗体的初始化代码，实现了“确认学号”、“提交选课”、“退出”三个Button控件的代码逻辑，“可选课程列表”、“已选课程表”两个DataGridView控件的代码逻辑。此外还封装了一个ClearAll()函数，用于将所有控件的属性设置为初始值；封装了一个SetLabel()函数，使用DataGridView的RowCount()方法以及调用Fun中的Count函数获取“已选课程表”这个DataGridView控件的总行数和“学分”列的总数，通过两个label控件分别显示出来。主要实现逻辑如下。

点击“确认”学号Button，如果textbox为空，代表没有输入学号，则弹出MessageBox提示输入学号。如果textbox不为空，则先通过调用SqlWarehouse的ExecuteScalarStringSql函数查找学生表中该学号的学生的姓名。如果查找没有结果，弹出MessageBox提示学号不存在或输入错误，并使用ClearAll将所有控件还原成初始值；如果获得了该学生的姓名，改变label控件的Text属性显示学生姓名，调用SetLabel()，且分别执行数据库中已经写好的“查找学号为x的学生未选课程”、“查找学号为x的学生已选课程”这两个存储过程，调用SqlWarehouse的SelectDataTable 函数返回相应的数据源，也就是“可选课程列表”、“已选课程表”两个DataGridView控件的数据源，调用Fun中的SetDataGridViewStyle()函数在“可选课程列表”的第一列增加“选课”列、“已选课程表”的最后一列增加“退课”列。

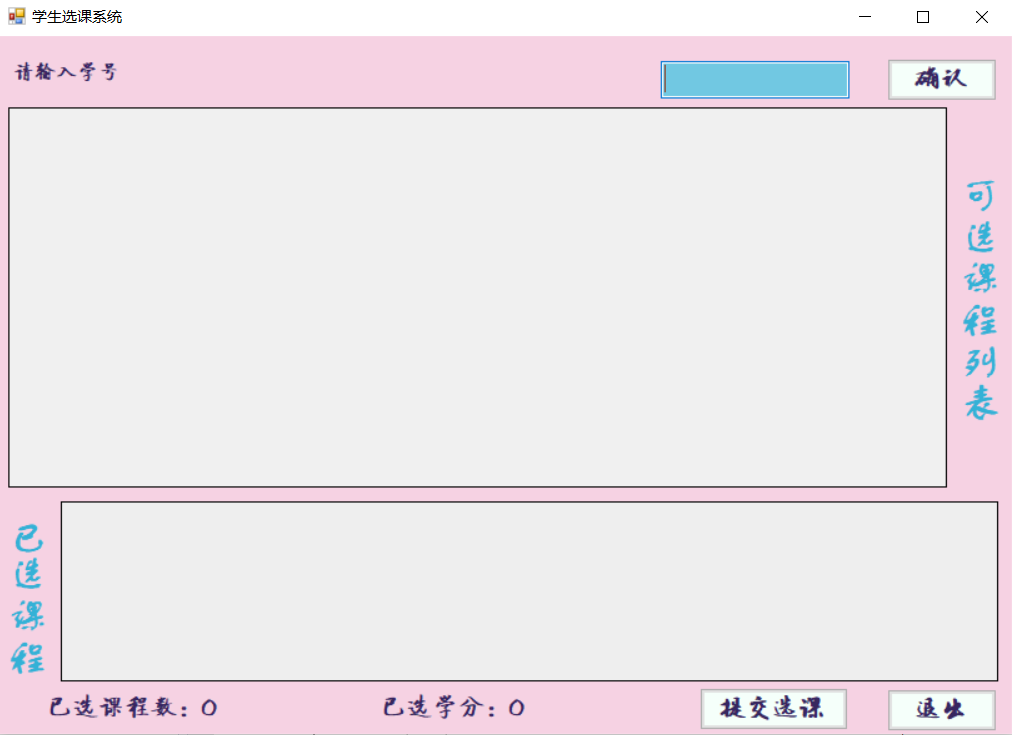
若点击“可选课程列表”或“已选课程表”的“选课”或“退课”列，则调用Fun中的MoveRow方法将被点击的行从一个表移动到另一个表中。这主要是通过对它们的数据源进行操作实现的。具体操作方法见上文。

若点击“提交选课”Button，会首先使用DataGridView的RowCount()方法确认已选课程数是否在3-5门之间，否则弹出MessageBox提示。然后，调用Fun中的Count函数确认已选课程的总学分是否介于8-12之间，否则弹出MessageBox提示。若满足选课要求，则调用SqlWarehouse的StringtoSqlCommand函数，在数据库的学生选课表TakeLesson中删除所有该学号同学的已有选课；然后调用Fun中的Admit函数将目前“已选课程表”中的每一行都插入到数据库的学生选课表TakeLesson中；最后调用本文件中的SetLabel()函数，并弹出MessageBox提示选课成功。

点击“退出”Button即使用Application.Exit()退出程序。

1. **实验结果（程序运行界面和操作简单说明）**

运行程序，初始化界面如下。已选课程数、已选学分均为0，左上角提示请输入学号。

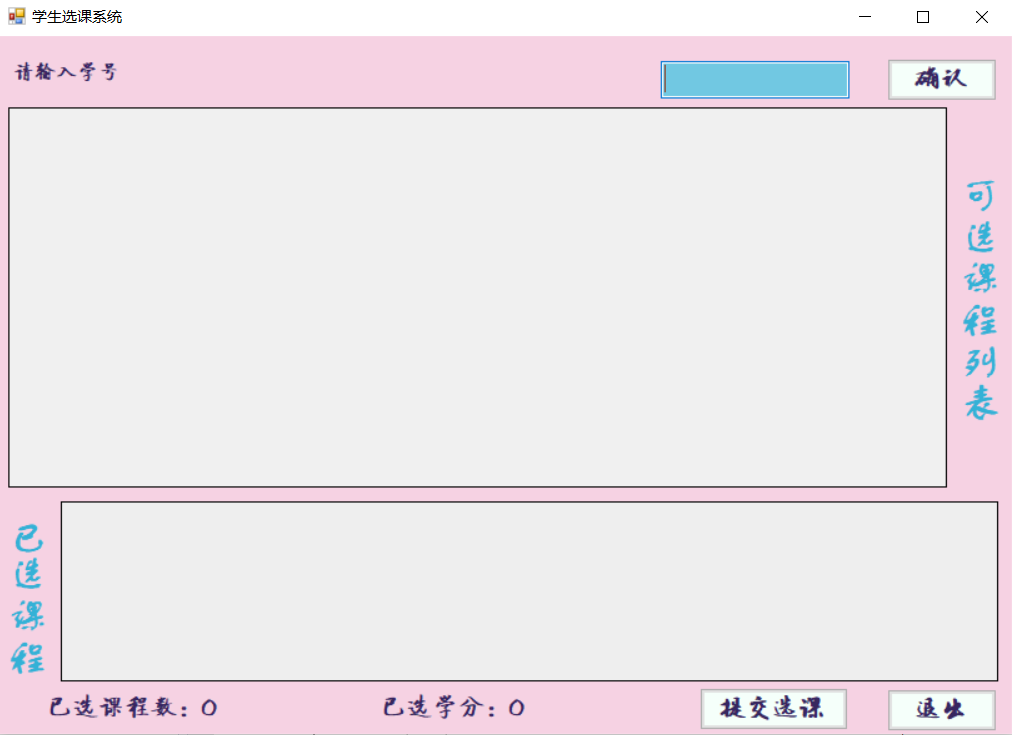
****

在右上角的文本框输入“1120190699”，这是一个数据库中已经存在的学生学号，并且她已经选好了课。点击“确认”按钮之后，文本框清空，“可选课程列表”、“已选课程表”显示出这个学生可选课程和已选课程。已选课程数和已选学分更新。左上角控件显示学生姓名。



在右上角的文本框输入一个数据库中不存在的学生学号。点击“确认”按钮之后，弹出提示框表示“学号不存在或输入错误”。点击提示框的“确定”之后界面恢复成一开始的状态（即第一张图）。



****

在右上角的文本框输入“1120192521”，这是一个数据库中已经存在的学生学号，并且还没有选课。点击“确认”按钮之后，“可选课程列表”、“已选课程表”显示出这个学生可选课程和已选课程。已选课程数和已选学分更新。左上角控件显示学生姓名。

****

点击“可选课程列表”中某一行的“选课”列，该行会从“可选课程列表”中删除，并加入到“已选课程”中，“已选课程数”和“已选学分”相应更新。

****

点击“已选课程”中“计算机新技术专题”的“退课”列，该行会从“已选课程”中删除，并加入到“可选课程列表”中，“已选课程数”和“已选学分”相应更新。

****

此时“已选课程数”和“已选学分”均满足选课要求，点击“提交选课”按钮，会弹出消息框提示“选课成功”。

****

如果“已选课程数”和“已选学分”不满足选课要求，点击“提交选课”按钮后会弹出消息框提示相应的错误信息。



****

随时点击右下角的“退出”按钮会退出程序。

1. **实验总结（结果分析）**
   1. **问题分析**

本次实验学习了与C#、.NET类方法相关的许多知识，学习了Visual Studio项目与数据库通信的基本方法。掌握了SqlCommand类、SqlParameter类、SqlConnection、SqlDataAdapter、DataGridView、DataTable、DataRow类的基本使用。解决了在两个DataTable中移动指定行，通过代码设置Label、DataGridView等控件的样式属性等等问题。

* 1. **程序优缺点**

优点：

1. 程序界面设计采用了较为独特的色彩搭配，界面设计简洁美观。
2. 在一个主程序界面中实现了所有的要求功能。在学号输入正确切换至学号输入错误的页面时，控件发生相应变化，如已选课程数和已选学分地提示在学号输入错误的页面中不会出现。并且在备选课表和拟选课表的第一列、最后一列增加了“选课”、“退课”提示列，用户交互科学。界面设计合理，用户交互科学。

缺点：

目前数据库的课程表中所有的课程在上课时间上都是不冲突的，在本项目开发中没有处理课程时间冲突这一问题。应该可以增加在拟选课的DataTable或者DataGridView中判断学生是否选中了一门时间冲突的课，若课程时间冲突则禁止选择。