**Java大作业——学籍管理系统设计说明书**

Readme!!!!:登录账户为admin密码为password，编写用的jdbc是老师在群里面发的jdbc。如果老师想测试，记得修改sqlServer和sqlServer\_m两个里面jdbc的数据,jdbc我也打包在项目里面了。

# 组员信息

|  |  |  |
| --- | --- | --- |
|  | 姓名 | 唐维鑫（小组长） |
| 学号 | 5120173864 |
| 班级 | 信管1701 |

|  |  |  |
| --- | --- | --- |
|  | 姓名 | 赵月平 |
| 学号 | 5120173857 |
| 班级 | 信管1701 |

|  |  |  |
| --- | --- | --- |
|  | 姓名 | 郑博予 |
| 学号 | 5120173858 |
| 班级 | 信管1702 |

|  |  |  |
| --- | --- | --- |
|  | 姓名 | 方浩文 |
| 学号 | 5120173865 |
| 班级 | 信管1702 |

|  |  |  |
| --- | --- | --- |
|  | 姓名 | 毛瑞 |
| 学号 | 5120173871 |
| 班级 | 信管1702 |

# 组长评分

|  |  |
| --- | --- |
| 姓名 | 得分 |
| 唐维鑫 | 9.9 |
| 赵月平 | 9.8 |
| 郑博予 | 9.8 |
| 方浩文 | 9.9 |
| 毛瑞 | 9.8 |

# 数据库设计

## 数据库系统规划与分析

* 1. 任务陈述

该学籍管理系统能实现存储学生基本信息、选修课程信息，并进行对学生信息的查询、插入、修改、删除的功能。

* 1. 系统范围

通过登陆管理员账号，按学号、按姓名、按班级三种方式查询得到学生信息后，得到特定学号学生的基本信息及选修课程信息的视图，并通过点击修改可修改部分学生基本信息。

## 数据库系统需求分析

* 1. 数据需求分析

学生基本信息数据：学号、姓名、性别、身份证号码、出生年月日、民族、籍贯、政治面貌、家庭通讯地址、邮编、家庭电话、身高、体重，血型、照片；班级信息数据：班级号、班名、班主任；所在系信息数据：系号、系名、系主任；课程信息数据：课程号、课程名、学分；选修课程信息：课程成绩。

## 数据库系统概念结构设计

* 1. E-R图设计

实体-属性列表：

实体集名：学生

属性列表：学号、姓名、性别、身份证号码、出生年月日、民族、籍贯、政治面貌、家庭通讯地址、邮编、家庭电话、身高、体重、血型、照片

实体集名：班级

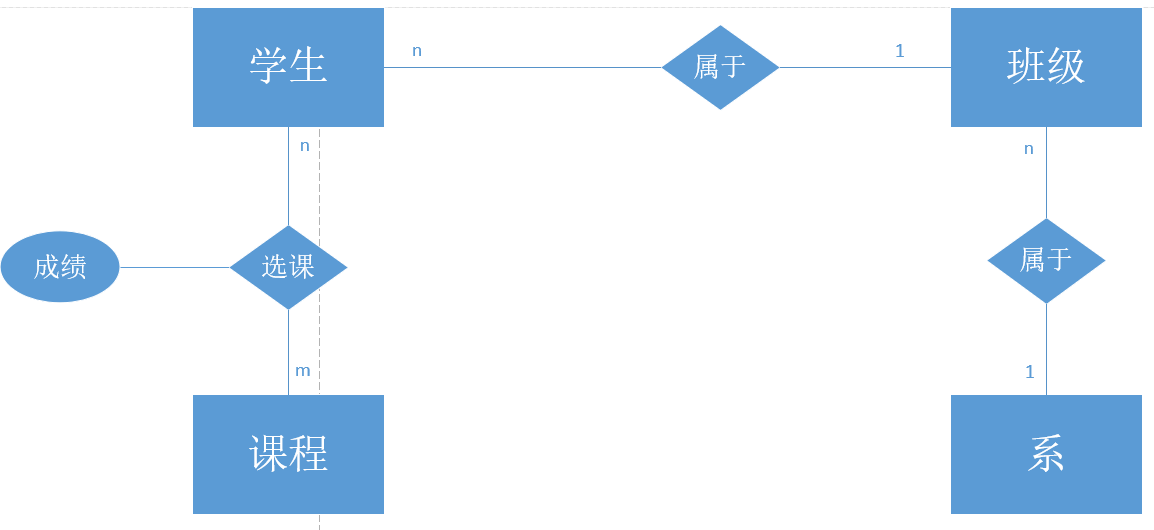
属性列表：班级号、班名、班主任

实体集名：系

属性列表：系号、系名、系主任

实体集名：课程

属性列表：课程号、课程名、学分



## 数据库系统逻辑结构设计

* 1. 关系模式设计

根据E-R图，可得出如下5个关系模式：

关系1：学生（学号，姓名，性别，身份证号码，出生年月日，民族，籍贯，政治面貌，家庭通讯地址，邮编，家庭电话，身高，体重，血型，照片，班级号）主码：学号；外码：班级号

关系2：班级（班级号，班名，班主任，系号）主码：班级号；外码：系号

关系3：系（系号，系名，系主任）主码：系号

关系4：课程（课程号，课程名，学分）主码：课程号

关系5：选课（学号，课程号，成绩）主码：（学号，课程号）；外码：（学号，课程号）

## 创建数据库

具体代码如下：

USE [master]

GO

/\*\*\*\*\*\* Object: Database [学生信息管理] Script Date: 2019/6/29 21:10:41 \*\*\*\*\*\*/

CREATE DATABASE [学生信息管理]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'学生信息管理', FILENAME = N'E:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\学生信息管理.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'学生信息管理\_log', FILENAME = N'E:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\学生信息管理\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [学生信息管理].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [学生信息管理] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [学生信息管理] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [学生信息管理] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [学生信息管理] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [学生信息管理] SET ARITHABORT OFF

GO

ALTER DATABASE [学生信息管理] SET AUTO\_CLOSE OFF

GO

ALTER DATABASE [学生信息管理] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [学生信息管理] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [学生信息管理] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [学生信息管理] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [学生信息管理] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [学生信息管理] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [学生信息管理] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [学生信息管理] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [学生信息管理] SET DISABLE\_BROKER

GO

ALTER DATABASE [学生信息管理] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [学生信息管理] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [学生信息管理] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [学生信息管理] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [学生信息管理] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [学生信息管理] SET READ\_COMMITTED\_SNAPSHOT OFF

GO

ALTER DATABASE [学生信息管理] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [学生信息管理] SET RECOVERY FULL

GO

ALTER DATABASE [学生信息管理] SET MULTI\_USER

GO

ALTER DATABASE [学生信息管理] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [学生信息管理] SET DB\_CHAINING OFF

GO

ALTER DATABASE [学生信息管理] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [学生信息管理] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [学生信息管理] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [学生信息管理] SET QUERY\_STORE = OFF

GO

USE [学生信息管理]

GO

ALTER DATABASE SCOPED CONFIGURATION SET LEGACY\_CARDINALITY\_ESTIMATION = OFF;

GO

ALTER DATABASE SCOPED CONFIGURATION SET MAXDOP = 0;

GO

ALTER DATABASE SCOPED CONFIGURATION SET PARAMETER\_SNIFFING = ON;

GO

ALTER DATABASE SCOPED CONFIGURATION SET QUERY\_OPTIMIZER\_HOTFIXES = OFF;

GO

ALTER DATABASE [学生信息管理] SET READ\_WRITE

GO

## 创建数据表

具体代码如下：

* 1. 学生表

USE [学生信息管理]

GO

/\*\*\*\*\*\* Object: Table [dbo].[学生表] Script Date: 2019/6/29 21:13:34 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[学生表](

[学号] [nchar](10) NOT NULL,

[姓名] [nchar](10) NOT NULL,

[性别] [char](2) NULL,

[班级号] [nchar](10) NULL,

[身份证号码] [nchar](20) NULL,

[出生年月日] [date] NULL,

[民族] [nvarchar](4) NULL,

[籍贯] [nvarchar](30) NULL,

[政治面貌] [nvarchar](30) NULL,

[家庭通讯地址] [nvarchar](50) NULL,

[邮编] [nvarchar](10) NULL,

[家庭电话] [nvarchar](20) NULL,

[身高] [nvarchar](10) NULL,

[体重] [nvarchar](10) NULL,

[血型] [nvarchar](3) NULL,

[照片] [image] NULL,

CONSTRAINT [PK\_学生表] PRIMARY KEY CLUSTERED

(

[学号] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

* 1. 班级表

USE [学生信息管理]

GO

/\*\*\*\*\*\* Object: Table [dbo].[班级表] Script Date: 2019/6/29 21:15:16 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[班级表](

[班级号] [nchar](10) NOT NULL,

[班名] [nchar](20) NULL,

[班主任] [nchar](10) NULL,

[系号] [nchar](10) NULL,

CONSTRAINT [PK\_班级表] PRIMARY KEY CLUSTERED

(

[班级号] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

* 1. 系表

USE [学生信息管理]

GO

/\*\*\*\*\*\* Object: Table [dbo].[系表] Script Date: 2019/6/29 21:16:05 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[系表](

[系号] [nchar](5) NOT NULL,

[系名] [nchar](20) NULL,

[系主任] [nchar](10) NULL,

CONSTRAINT [PK\_系表] PRIMARY KEY CLUSTERED

(

[系号] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

* 1. 课程表

USE [学生信息管理]

GO

/\*\*\*\*\*\* Object: Table [dbo].[课程表] Script Date: 2019/6/29 21:17:07 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[课程表](

[课程号] [nchar](10) NOT NULL,

[课程名] [nvarchar](20) NULL,

[学分] [tinyint] NULL,

CONSTRAINT [PK\_课程表] PRIMARY KEY CLUSTERED

(

[课程号] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

* 1. 课程表

USE [学生信息管理]

GO

/\*\*\*\*\*\* Object: Table [dbo].[选课表] Script Date: 2019/6/29 21:17:46 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[选课表](

[学号] [nchar](10) NOT NULL,

[课程号] [nchar](10) NOT NULL,

[成绩] [nchar](10) NULL,

CONSTRAINT [PK\_选课表] PRIMARY KEY CLUSTERED

(

[学号] ASC,

[课程号] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

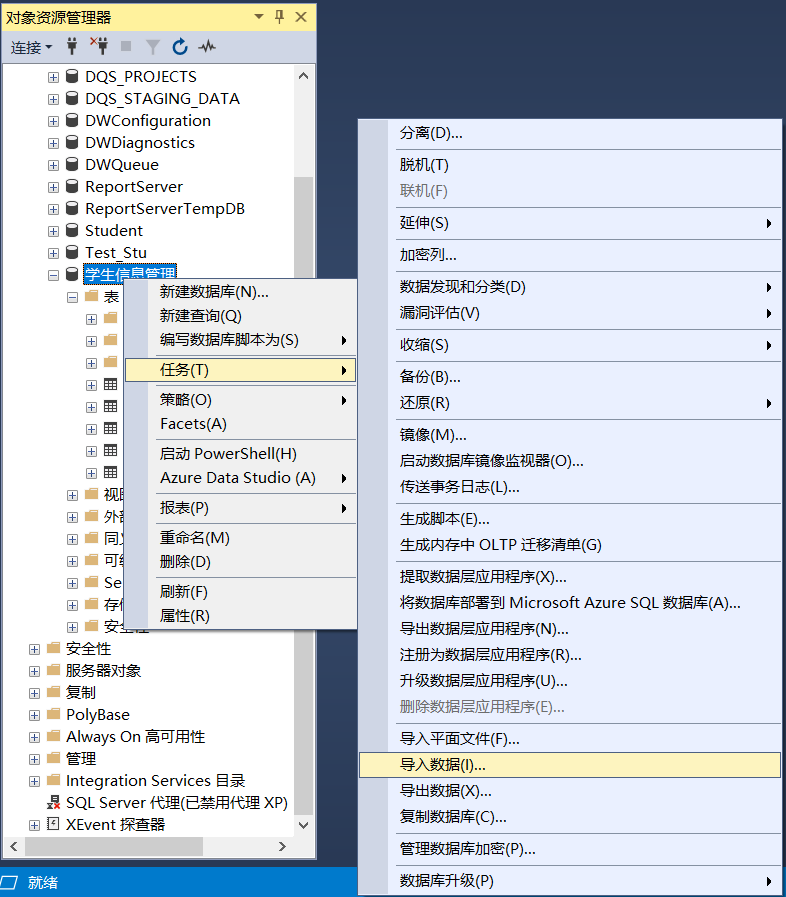
) ON [PRIMARY]

GO

## 数据装载

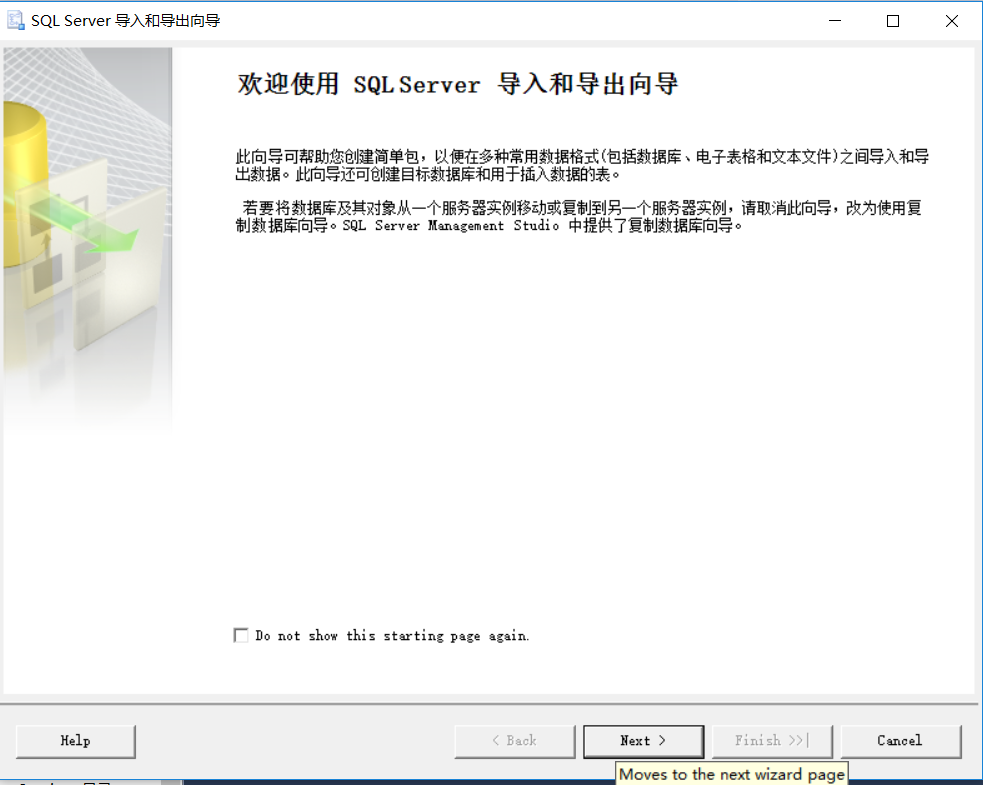
事先准备从网上下载的5个excel工作簿，从excel工作簿中将数据导入数据库。

第一步，如图。



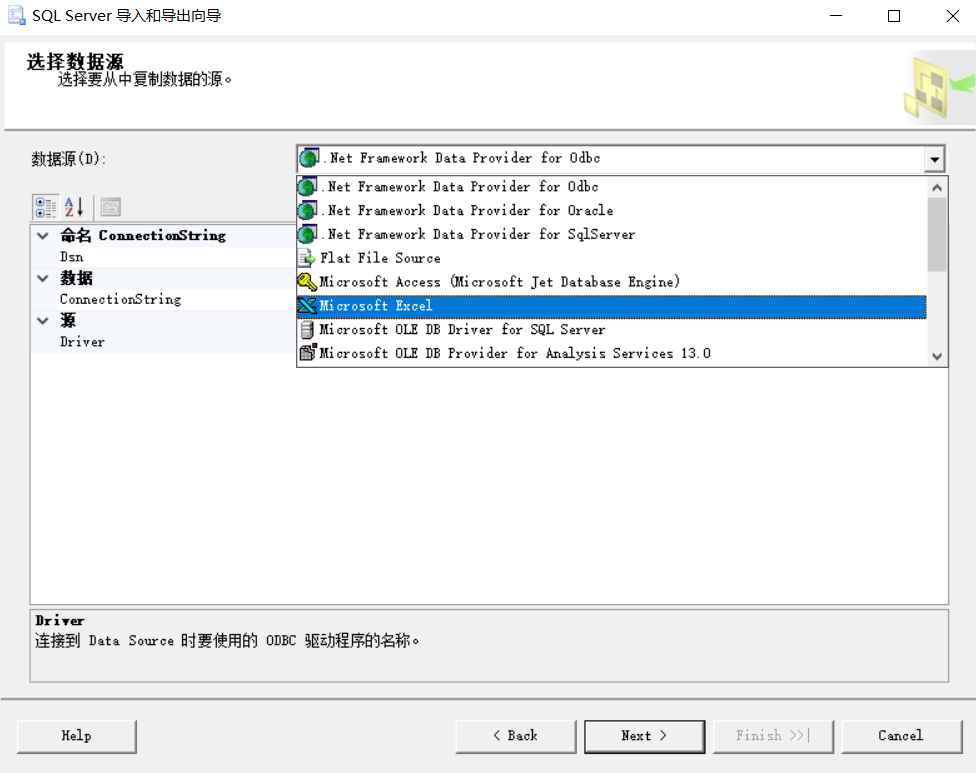
点击导入数据。

第二步，如图。



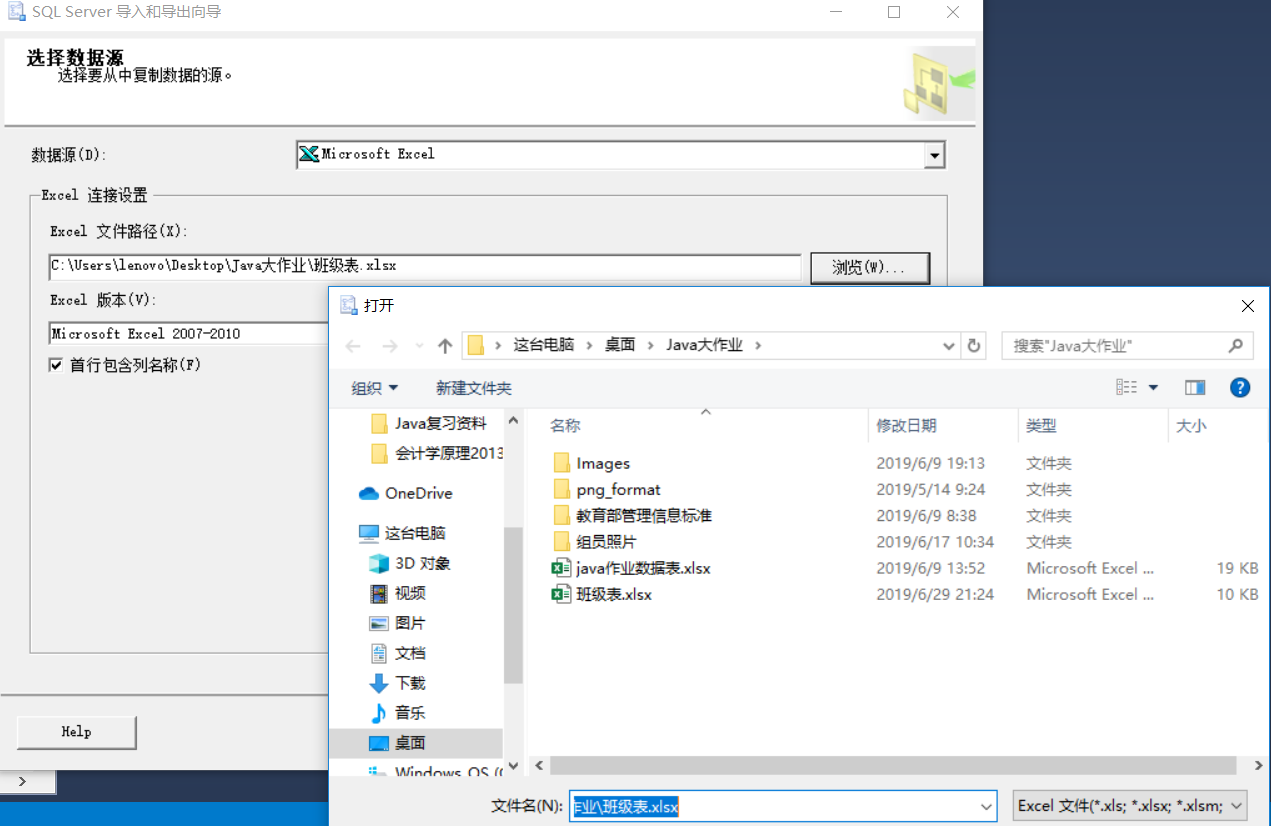
点击Next。

第三步，如图。



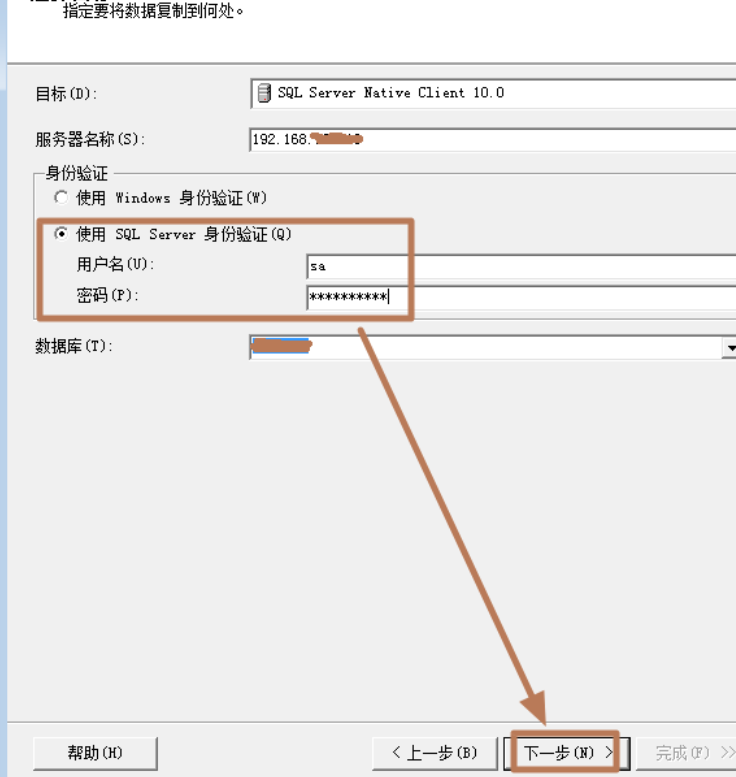
选择Microsoft Excel。

第四步，选择需要导入的EXCEL文件。点击浏览，找到导入的文件确定。如图。

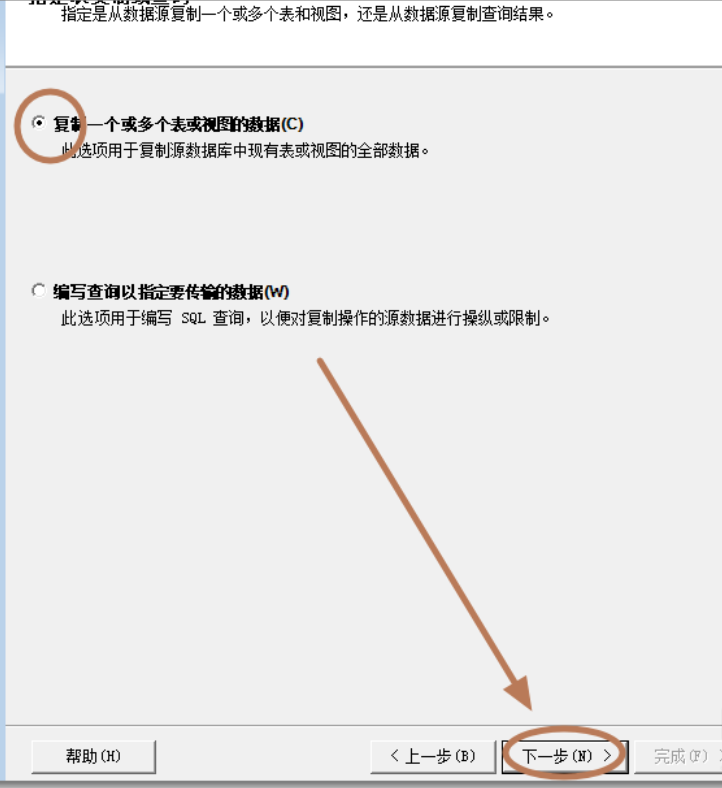


确认路径没有问题后点击下一步。

第五步，默认为是使用的WINODWS身份验证，改为使用SQL身份验证。输入用户名和密码。



第六步，如图。



接下来选择我们想要导入的文件后点击下一步至完成即可。导入数据完成后可以使用select \* from 班级表，测试导入结果。

至此，数据库部分建立完成。

# 系统设计

## 首先进行功能的拆分与模块的拆分

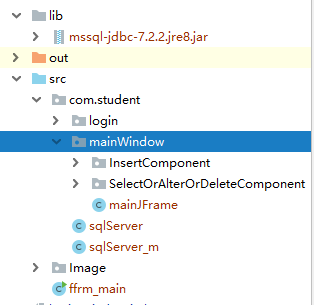
经过思考，我们将功能拆分成：

1. 学生信息的查询，包括按学号查询，按姓名查询。
2. 学生信息的修改与删除。
3. 学生信息的整条插入。
4. 登陆学籍系统的实现。

对应功能，我们进行模块划分：

1. 学生信息的查询，修改与删除模块。
2. 录入整体信息模块。
3. 登陆模块。
4. 整体框架模块

最后我们的工程目录如下：



接下来我们进行了整体UI的大体设计，画出了草图，对里面的组件大小进行了约定

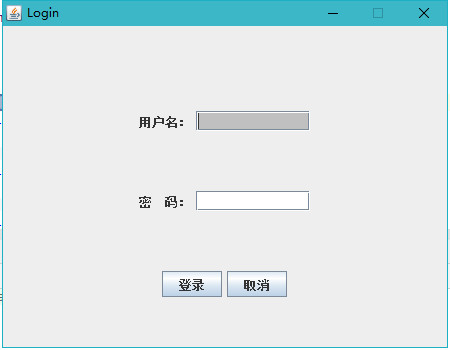


将任务划分完毕后，我们进行了分工，每人负责一个模块的代码设计，然后将这些模块组合起来。

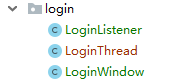
## 各模块代码设计

### 登陆框部分代码

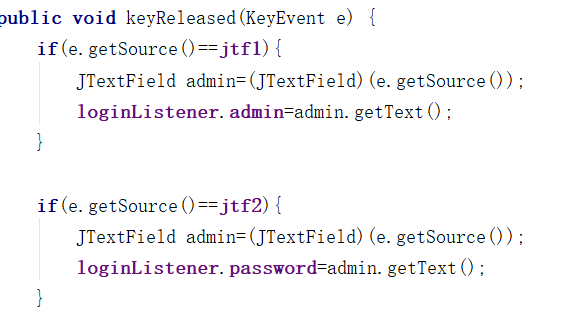
实现界面如图



登录框的文件分为三个类：



1. loginWindow:主要负责登陆框的UI，以及密码和账号进行更新。UI部分用了表格布局模式。密码和账号进行更新用了一个KeyListener接口，当键盘松开的时候，获取到密码与账号的值，然后对LoginListener中的admin以及password进行更新。



1. LoginListener:主要负责对账号和密码是否对应进行判断。它监听了UI部分的确认与取消按钮。

确认部分的响应：



逻辑：当点击确认后，查询数据库账号密码表中对应输入admin的password，如果未查到，提示该账号不存在。如果存在，对查询结果与password对比，如果相等，修改这个类里面的IsLogin为true。当点击取消，设置IsOut为ture。

1. LoginThread:考虑到性能的原因，这里写了LoginThread代码，作用是不停判断是否登录，登录就关闭登录窗口，让主窗口出现。程序所有的类初始化都放在这个类里面。

核心代码：



### 主框架代码设计

如下：主框架是放置核心组件的容器



红色部分为主框架写好的部分，白色为对数据库进行操作的组件。

代码为mainJFrame类，两个按钮响应

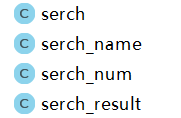


逻辑：当点击查询，修改，删除按钮时，将学生信息插入组件设置为不可见，将查询，修改，删除组件设置为看见。当点击学生信息插入按钮时，将查询，修改，删除组件设置为不可见，将学生信息插入组件设置为看见。

### 查询修改删除代码设计

#### 查询界面部分

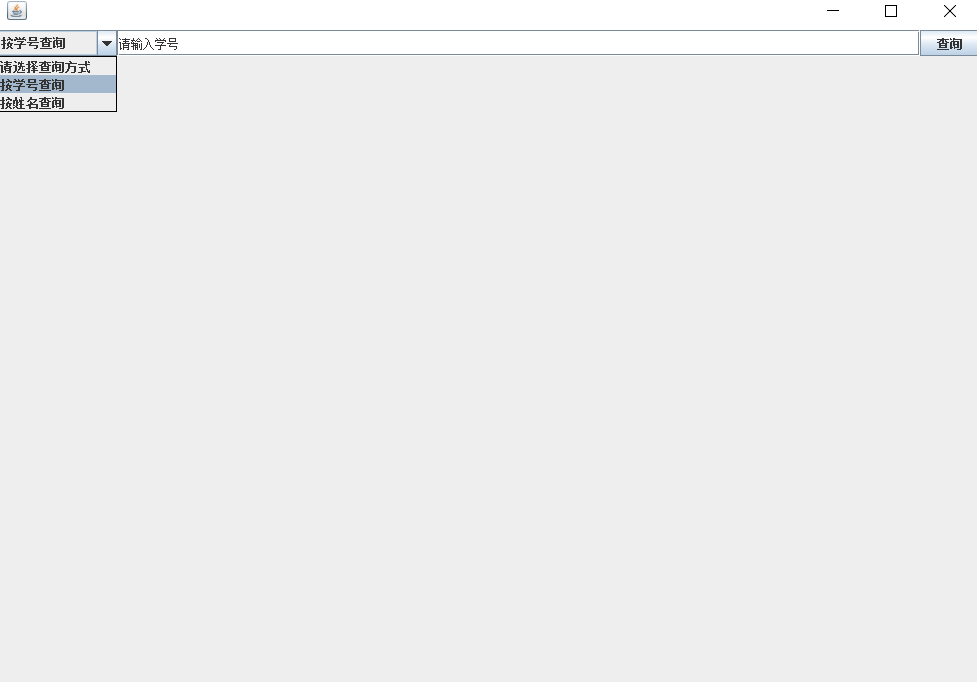
代码设计五个类，如下。

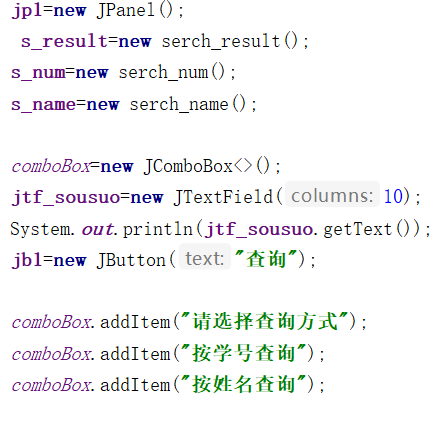


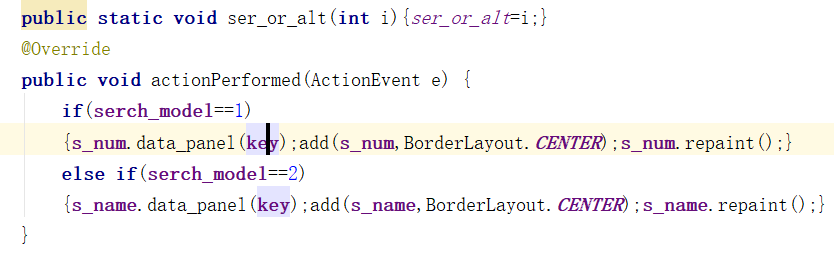


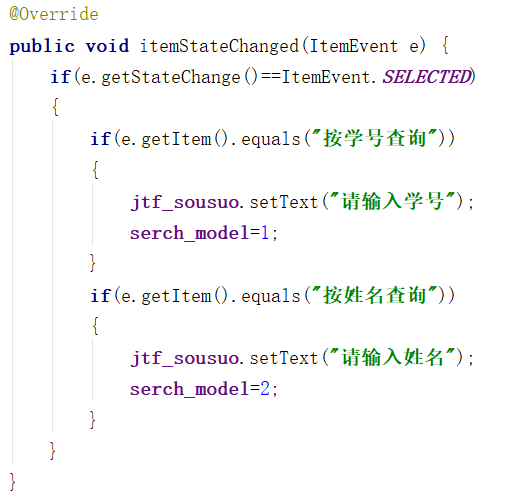
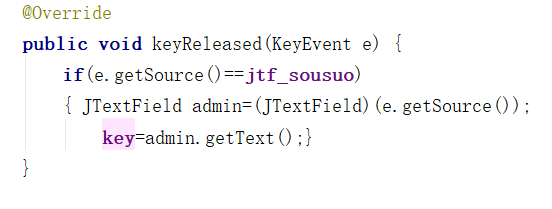
大致逻辑为通过serch类进入查询的初始界面，后选择查询方式为按学号查询或按姓名查询，在搜索框输入学号或姓名后，点击查询按钮，进入仅有学号、姓名、班级信息的主要信息界面；点击主要信息列后的查询按钮，进入详细信息的界面；点击课程学习情况按钮，查看选修课程的成绩及学分。

1. Serch类完成查询的初始界面，外观及主要代码如下。

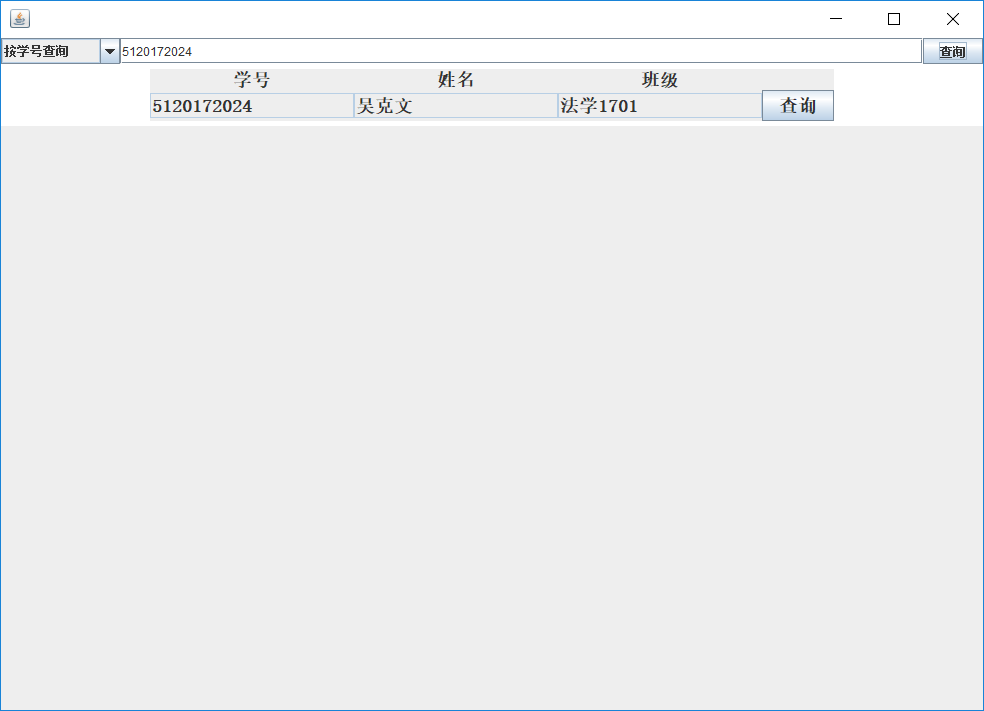


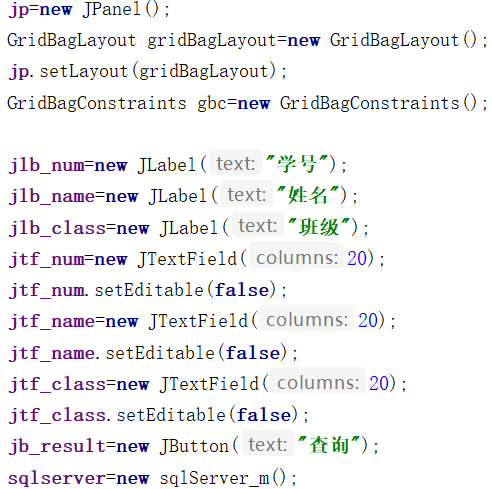


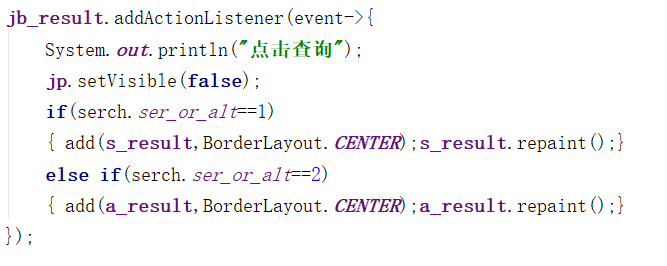




2、serch\_num类查询按学号查询出的主要信息，通过点击查询按钮，使用SQL语句**"SELECT A.学号,A.姓名,B.班名 FROM 学生表 A join 班级表 B on A.班级号=B.班级号 where A.学号 = "**+**"'"**+key+**"'"**，连接学生表和班级表，外观及主要代码如下。

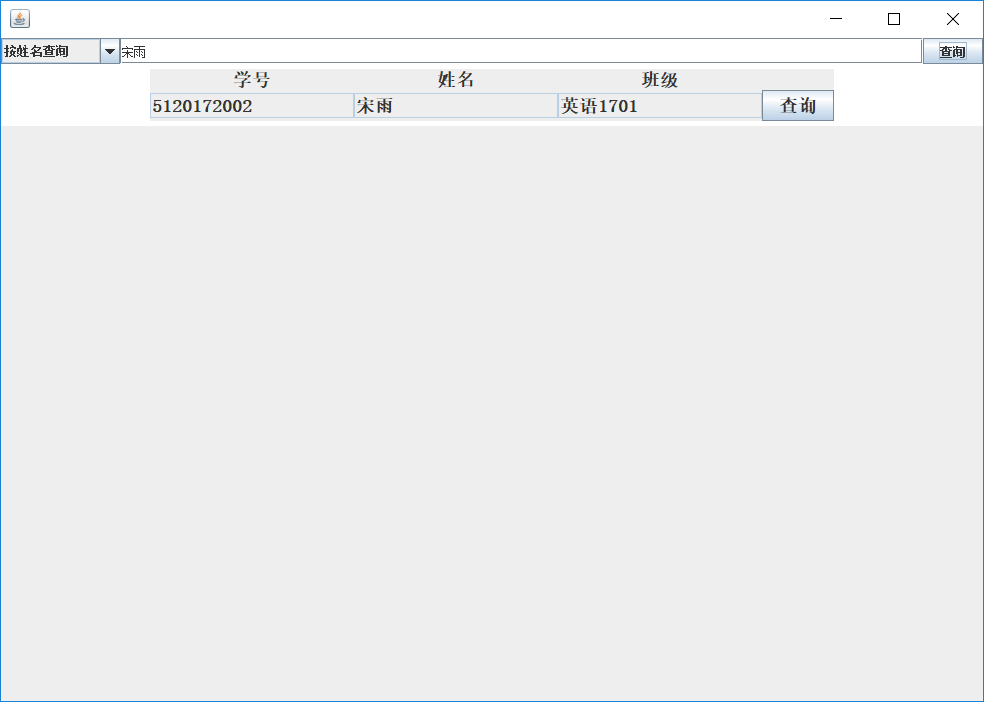


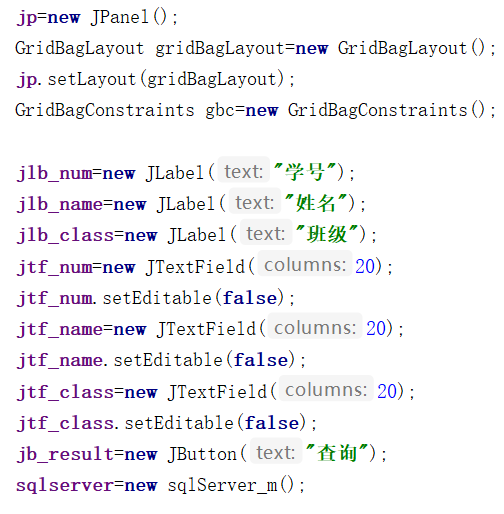


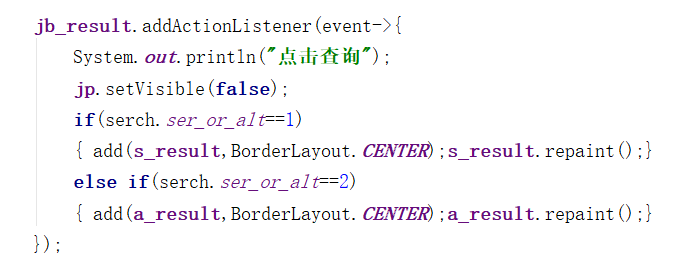


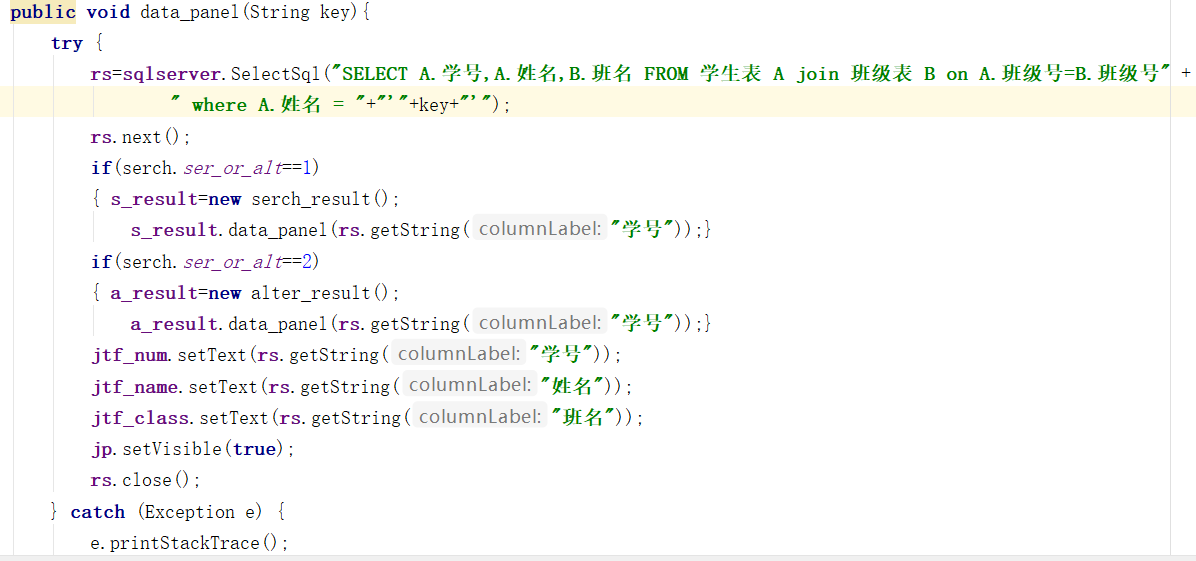


3、serch\_name类查询按姓名查询出的主要信息，通过点击查询按钮，使用SQL语句**"SELECT A.学号,A.姓名,B.班名 FROM 学生表 A join 班级表 B on A.班级号=B.班级号 where A.姓名 = "**+**"'"**+key+**"'"**，连接学生表和班级表，外观及主要代码如下。

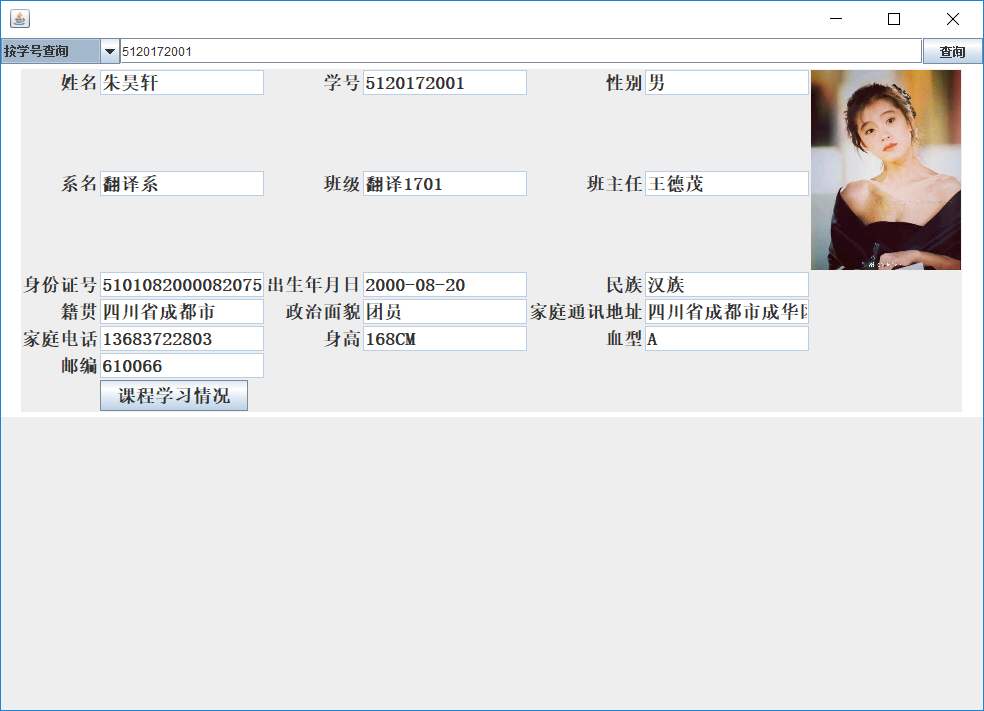




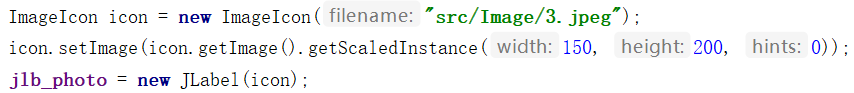




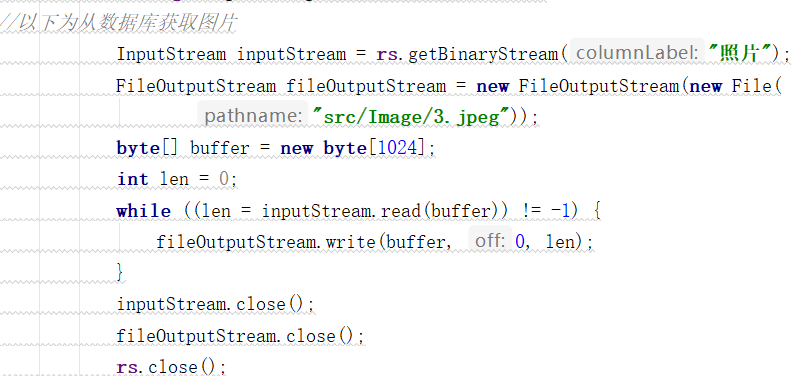
4、serch\_result类为通过点击主要信息界面后的查询按钮进行再次查询，使用SQL语句**"SELECT A.姓名,A.性别,A.身份证号码,A.出生年月日,A.民族,A.籍贯,"** + **"A.政治面貌,A.家庭通讯地址,A.邮编,A.家庭电话,A.身高,A.血型,A.照片,B.班名,B.班主任,C.系名"** +**" FROM 学生表 A join 班级表 B on A.班级号=B.班级号 join 系表 C on B.系号=C.系号"**+**" where A.学号="**+**"'"**+num+**"'"**，连接学生表、班级表、系表得到详细的学生基本信息，外观及主要代码如下。





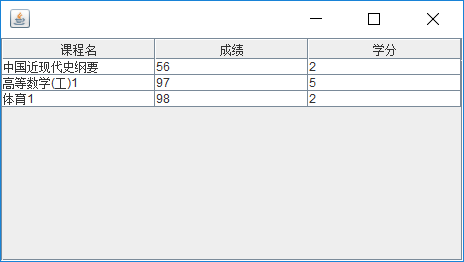


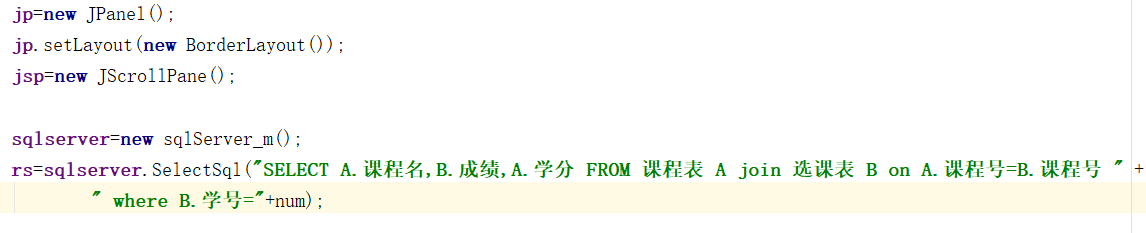


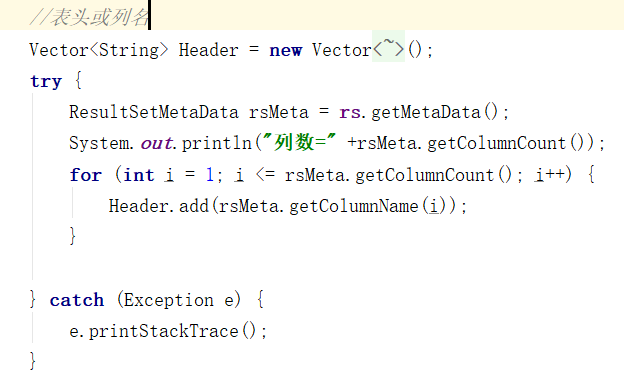


5、Course类是通过点击详细信息界面的课程学习情况按钮，使用SQL语句**"SELECT A.课程名,B.成绩,A.学分 FROM 课程表 A join 选课表 B on A.课程号=B.课程号 where B.学号="**+num

，连接课程表和选课表，进一步查询出该同学选修的课程及成绩，并且使用表格展示课程信息，外观及主要代码如下。







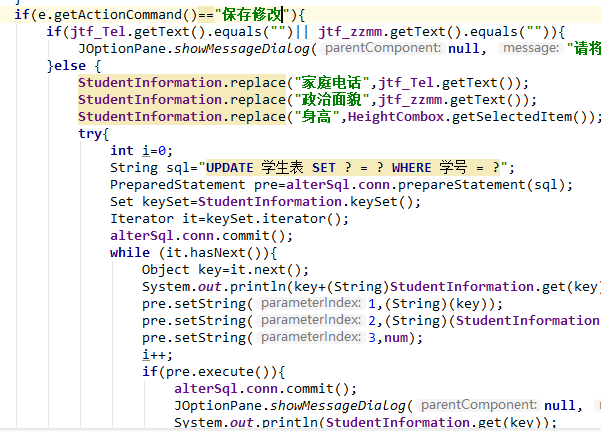


#### 修改删除部分

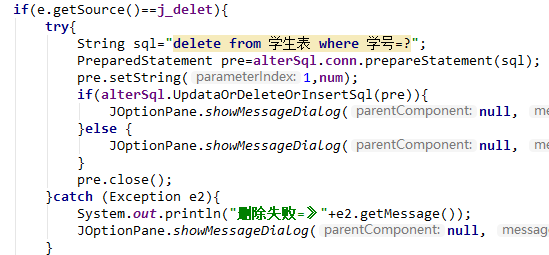
修改部分代码和前面查询代码一样，只有最后展示的结果界面不一样。

修改部分逻辑是通过文本框或选择框获取到数据后，存储在一个map数据类型里面，然后将map里面的数据依次向数据库更新。

当点击保存修改按钮后，先判断输入值是否为空，不为空的话，向map存入信息，遍历map进行进行数据库更新。代码如下：



当点击删除按钮时，获取当前的信息的学号，进行删除，代码如下：



界面如截图，可以编辑的地方是可以修改数据的部分



### 4.学生信息插入部分代码

整体逻辑：用户输入完数据后，当点击提交的时候，首先检查数据是否填写完整，完整的话将输入信息保存在一个map里面，然后添加进数据库。

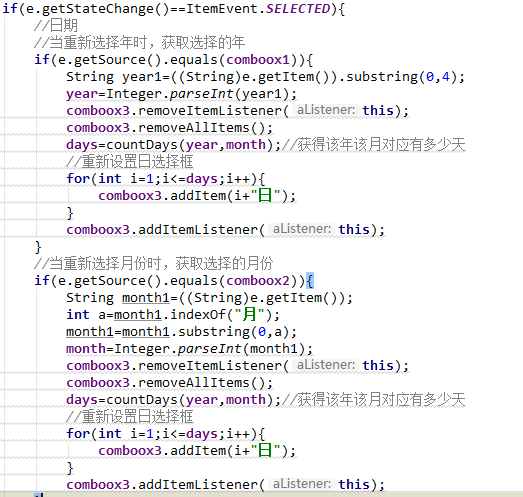
#### UI部分：

考虑到输入数据的正确性，对性别，地址，日期等设计为选择框输入或复选框输入，界面如下:

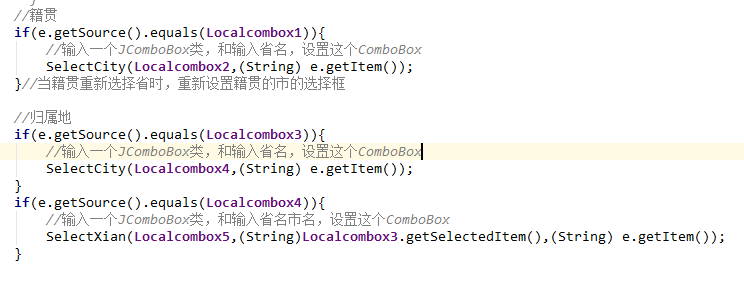


其中一些重要的设计：

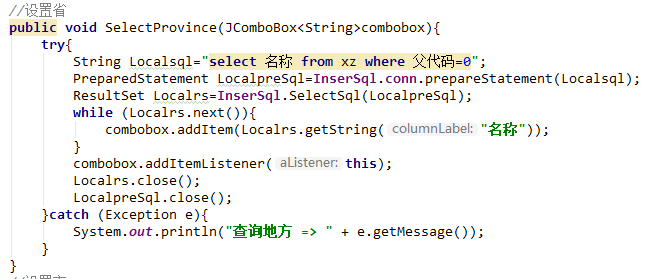
1. 出生年月部分：逻辑是每当用户选择一次年或月选项，对日进行更新。相应代码代码如下：



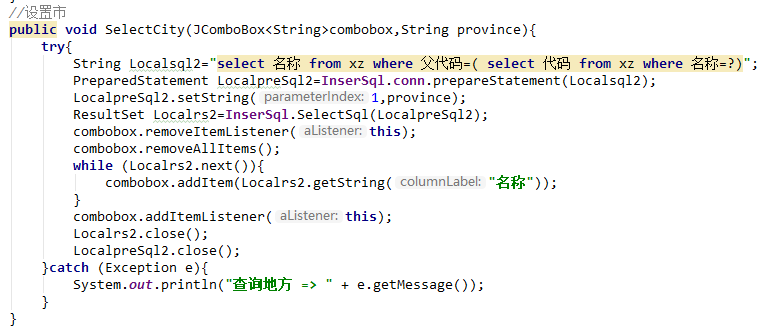
1. 地址选择框：逻辑当重新选择省时，对市和县的数据通过查询数据库后更新。当重新选择市市，对县的数据通过查询数据库后更新。代码如下：



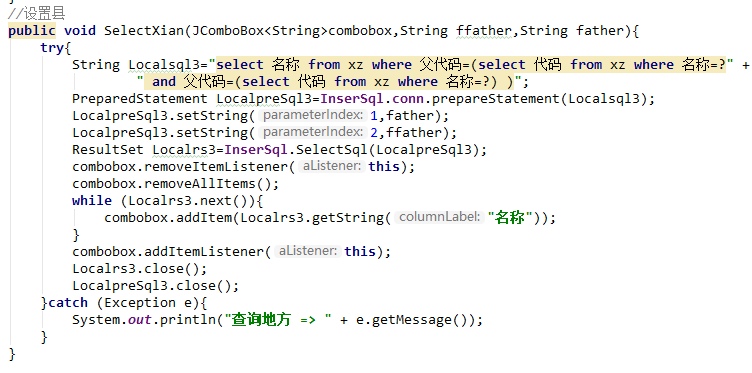
SelectPrince()函数代码：



SelectCity()函数代码：



SelectXian()函数代码：

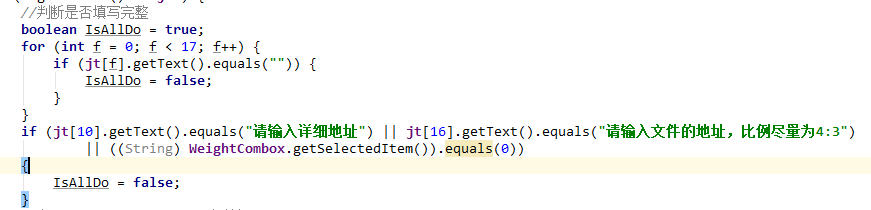


#### 对重填按钮的相应：



#### 对提交按钮的响应：

##### 先检查是否填写完



如果填写完，开始向map写入数据：其中对一些关键数据的处理：

##### 数据处理

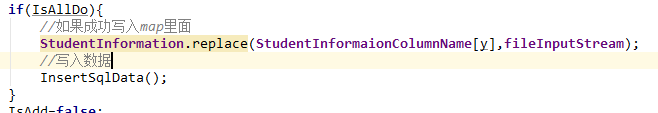
* 1. 日期类



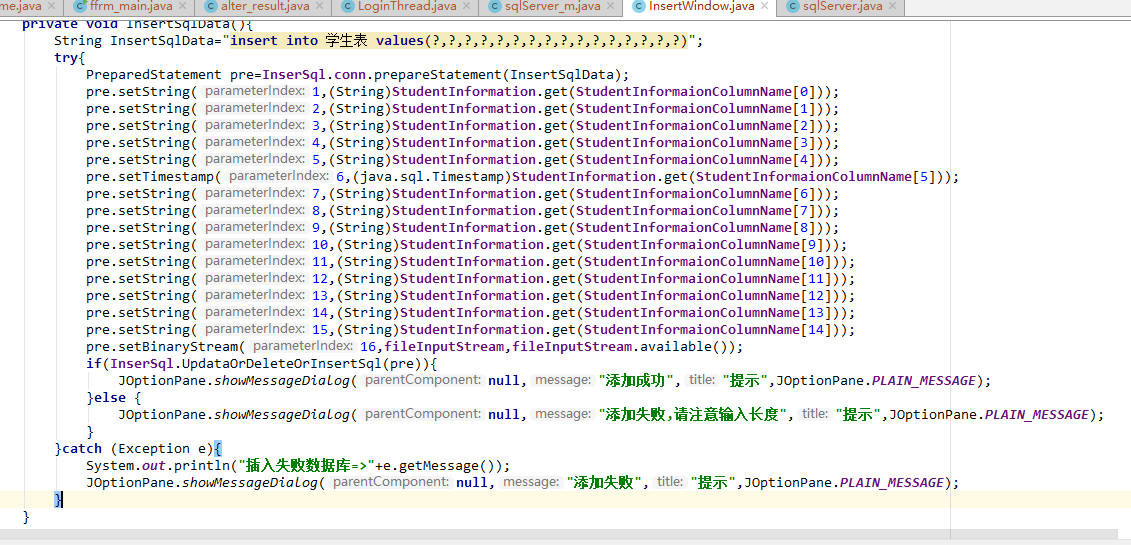
2图片类



##### 数据进行写入



InsertSqlData:因为之前已经将数据保存在一个map里面，所以这里只需要将map中数据依次写入数据库，代码如下：



## （三）其他

SqlSever, SqlSever\_m:这两个类是我们写代码时建立的数据库连接类，方便连接数据和对数据库信息进行操作。老师如果想进入软件，记得改里面的相应的信息哟