

**数据库系统课程设计**

学生姓名： 宋楚嘉

班 学 号： 11D203 20201000830

指导教师： 郑坤老师

**中国地质大学信息工程学院**

**2021年 2月 18 日**

**目录**

**一、需求分析....................................................................................................................3**

**（一）模块分析 .................................................................................................................3**

**（二）成绩模型分析..................................................................................................................3**

**（三）用户体验分析..................................................................................................................3**

**二、概念设计....................................................................................................................4**

**（一）概念模型（E-R图）..................................................................................................4**

**（二）数据字典.....................................................................................................................6**

**三、逻辑结构设计........................................................................................................8**

**（一）关系模式设计..................................................................................................................8**

**（二）关系模式分析................................................................................................................8**

**四、物理设计.....................................................................................................................11**

**五、系统实施.....................................................................................................................12**

**（一）系统架构........................................................................................................................12**

**（二）客户端系统结构............................................................................................................12**

**（三）连接实现....................................................................................................................13**

**（三）测试 ............................................................................................................................16**

**六、运行维护......................................................................................................................18**

**（一）运行环境.......................................................................................................................18**

**（二）系统维护........................................................................................................................19**

**七、系统操作指南........................................................................................................20**

**八、附录.................................................................................................................................30**

**（一）项目摘要及地址............................................................................................................30**

**（二）演示地址........................................................................................................................30**

**九、总结...................................................................................................................................30**

**实习题目**

设计并开发大学生课程学习管理与成绩评价系统，设计绩点模型，开发地图交互功能。

### 一、需求分析

**1）模块分析**

我认为系统可以分解成四个主要模块：选课管理、成绩管理、用户信息管理维护和地图交互，为学生、教师和管理员提供强大且简便的服务。下面从四个模块进行需求分析：

首先是**选课模块**。学生在选课时首先要能够看到可供选择的教学班信息，包含了课程信息以及教学班的时间、地点、授课教师以及计划上限和当前人数等信息。并且能按照不同的属性，诸如：课程类型、课程名、授课教师等进行条件筛选。在此基础上进一步优化：对用户已经选择的教学班进行标识，提示已选。我们需要明确的是，同一门课程可以开设多个教学班。如果已经选择了某一个教学班，就不能选择其他教学班了，要避免重复选课。

对于**成绩管理模块**，我们首先需要明确的是：从一条成绩记录出发，必须要能“回溯”到相应的教学班信息，最重要的是要能“回溯”到授课教师。如果成绩记录仅能找到课程信息，那么授课教师就难以对他的学生进行成绩管理。因为课程和教师是N:N的关系。

**用户信息的管理维护**首先要规定好哪些是可以改的，哪些是不能改的。比如：行政班实际上决定了学生的专业、系、学院，行政班的变动会使学籍发生异动。而手机号、密码、头像、生日等信息是独属于用户个体的基本信息，与其他实体并无联系，则可以修改。

**地图交互模块**要能够辅助排课，这是基本需求，我们还应当思考的是：如何使得该模块充分服务于学生、教师和管理员。从用户的角度考虑，假设有一张功能足够强大的电子地图，用户会拿它做些什么？1、前往一间教室，学生和教师都希望能够快速把握教室的空间（相对）位置，那么选中教学班信息，地图上要能够指示教室位置。2、学生想去教室自习，希望找到一间空教室，那么楼层中教室的使用情况是怎样的，哪些教室可用，哪些教室被占用了要能够直观体现。3、学生下课后想在教室继续自习，点击地图上的教室，希望能够看到这间教室在不同时间段的占用情况以便安排自习时间。4、管理员排课（教学班）的一种逻辑是：先找一间空教室，再安排占用这间教室的教学班。那么楼层中教室占用情况的总览、某间教室不同时间段的占用情况都应得以体现。由此可见，地图交互模块绝不是一个孤立的模块，它是“增删改查”的强力辅助工具，与其他模块协作可以极大地提高便利性和可视性。

**2）成绩模型分析**

在学校这样一个体系中，具有明确的“层级”结构：学校->学院->系->专业->行政班->学生（教师）。虽然我们不设计层次模型，但是我们可以在用户管理的时候引入树形结构，按照层级进行用户个人信息和学生成绩信息的检索，得到更清晰、更有条理的呈现。以学情排名查询为例：设置绩点计算规则，系统自动对参加学习的所有学生进行平均学分绩点的统计。在不同的级别下，我们应当看到不同的排名视图（系排、专排、班排不应该是“死”数据，而应当根据选择的条件和统计范围动态生成）。

**3）用户体验分析**

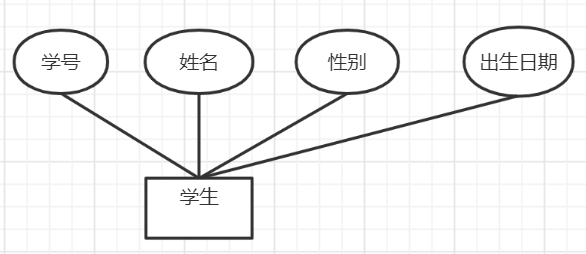
最后，让我们回归用户体验进行需求分析。DataGridView和TreeView可以直观地展示查询结果，那么修改和删除操作要是能够在查询的基础上，在用户看到的表格直接进行操作就可以极大地提升便利性。用户更希望自己的行为是连续的，比如教师登分：在表格上批量登分并可以暂时保存，随时撤销、修改，敲定确认后再一并提交数据库，这样的成绩录入功能可以更好地满足实际需求。

### 二、概念设计

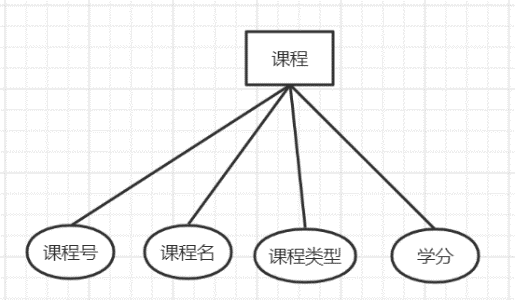
（1）**概念模型（E-R图）**

局部ER图

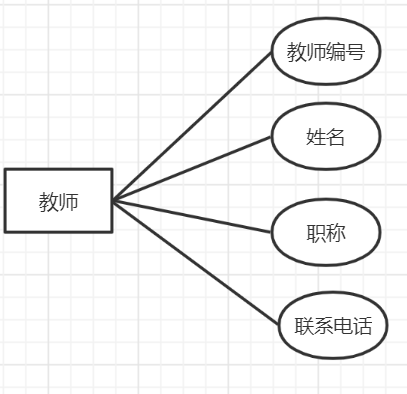
①学生



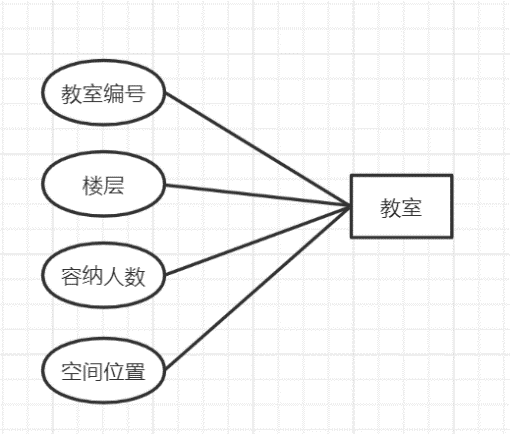
②课程



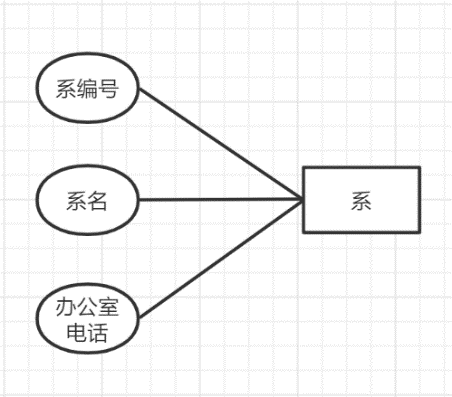
③教师



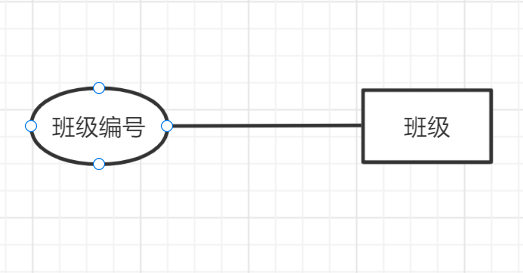
④教室



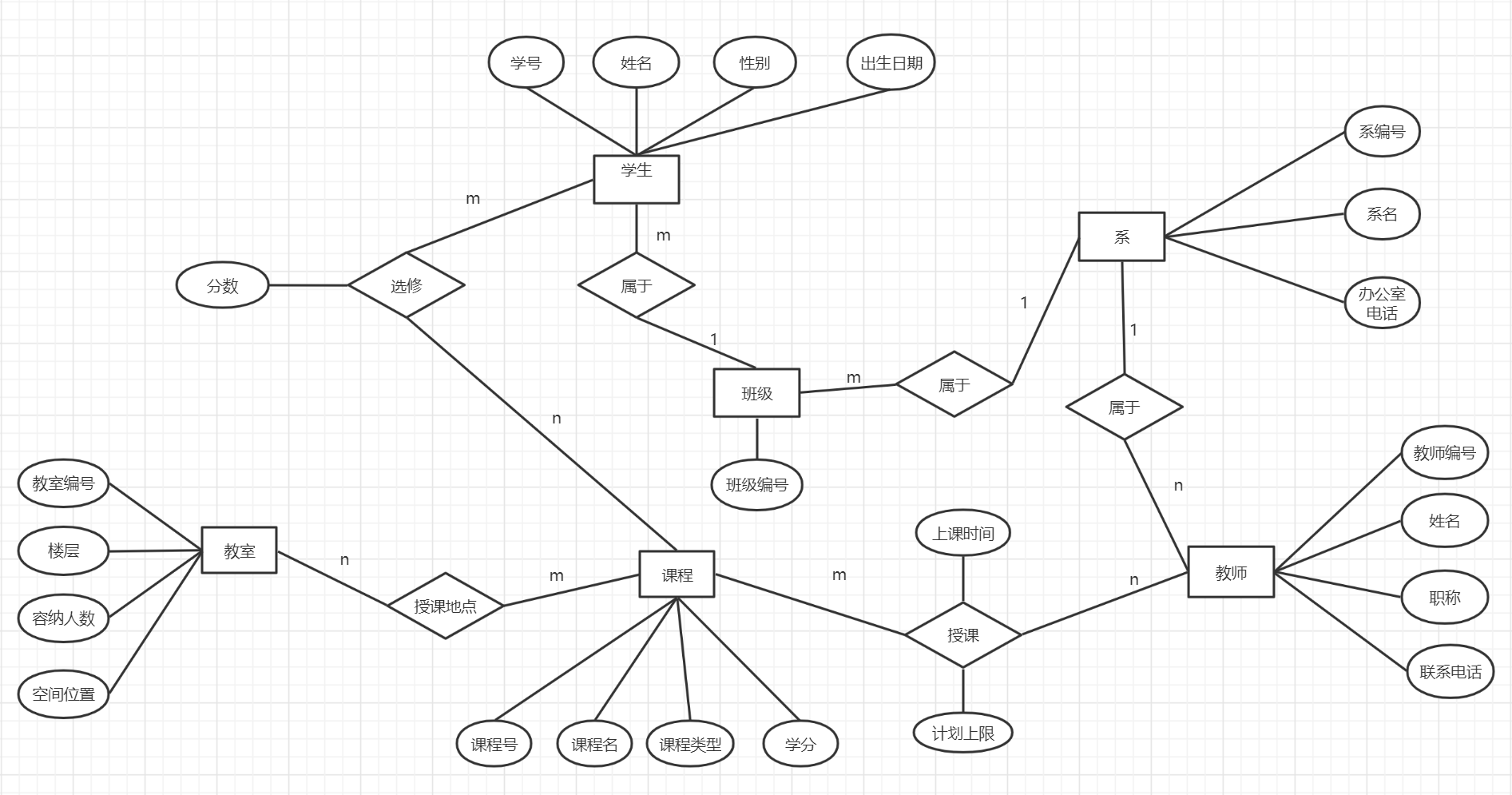
⑤系



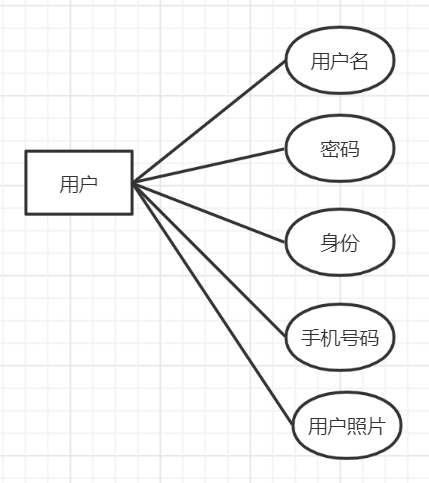
⑥班级



总ER图：



（对学生、教师、管理员实体进行抽象后可以得到用户类。之后建立用户安全信息表，可以将三者的安全信息存储在这一张表上）



（2）**数据字典**：

表1 数据项列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 数据项 | 数据类型 | 码 | 备注 |
| 1 | 系编号 | varchar(30) | 主码 | 系 |
| 2 | 系名 | varchar(40) |  |  |
| 3 | 系办公室电话 | varchar(50) |  |  |
| 4 | 学院 | varchar(40) |  |  |
| 5 | 教师编号 | varchar(20) | 主码 | 教师 |
| 6 | 姓名 | varchar(20) |  |  |
| 7 | 职称 | varchar(20) |  |  |
| 8 | 联系方式 | varchar(50) |  |  |
| 9 | 系编号 | varchar(30) | 外码 |  |
| 10 | 班级编号 | varchar(10) | 主码 | 班 |
| 11 | 系编号 | varchar(30) | 外码 |  |
| 12 | 专业 | varchar(40) |  |  |
| 13 | 学号 | varchar(20) | 主码 | 学生 |
| 14 | 姓名 | varchar(20) |  |  |
| 15 | 性别 | varchar(10) |  |  |
| 16 | 出生日期 | smalldatetime |  |  |
| 17 | 班级编号 | varchar(10) | 外码 |  |
| 18 | 教室 | varchar(10) | 主码 | 教室 |
| 19 | 楼层 | varchar(5) |  |  |
| 20 | 容纳人数 | tinyint |  |  |
| 21 | 空间位置 | varchar(10) |  |  |
| 22 | 课程代号 | varchar(20) | 主码 | 课程 |
| 23 | 课程名 | varchar(40) |  |  |
| 24 | 课程类型 | varchar(20) |  |  |
| 25 | 学分 | tinyint |  |  |
| 26 | 课程教学ID | varchar(20) | 主码 | 教学班 |
| 27 | 上课时间 | varchar(40) |  |  |
| 28 | 教室编号 | varchar(10) |  |  |
| 29 | 教师编号 | varchar(20) | 外码 |  |
| 30 | 计划上限 | tinyint |  |  |
| 31 | 已选人数 | tinyint |  |  |
| 32 | 课程代号 | varchar(20) | 外码 |  |
| 33 | 学号 | varchar(20) | 外码 | 课程学生SC |
| 34 | 课程教学ID | varchar(20) | 外码 | (学号,课程教学ID)为主码 |
| 35 | 成绩 | float |  |  |
| 36 | 用户名（id） | varchar(20) | 主码 | 用户安全信息 |
| 37 | 密码 | varchar(max) |  |  |
| 38 | 身份 | varchar(20) |  |  |
| 39 | 手机号码 | varchar(11) |  |  |
| 40 | 用户照片 | image |  |  |
| 41 | 学号 | varchar(20) | 外码 | 绩点模型 |
| 42 | 课程代号 | varchar(20) | 外码 | (学号,课程代号)为主码 |
| 43 | 等级 | varchar(10) |  |  |
| 44 | 绩点 | float |  |  |
| 45 | 权重 | float |  |  |

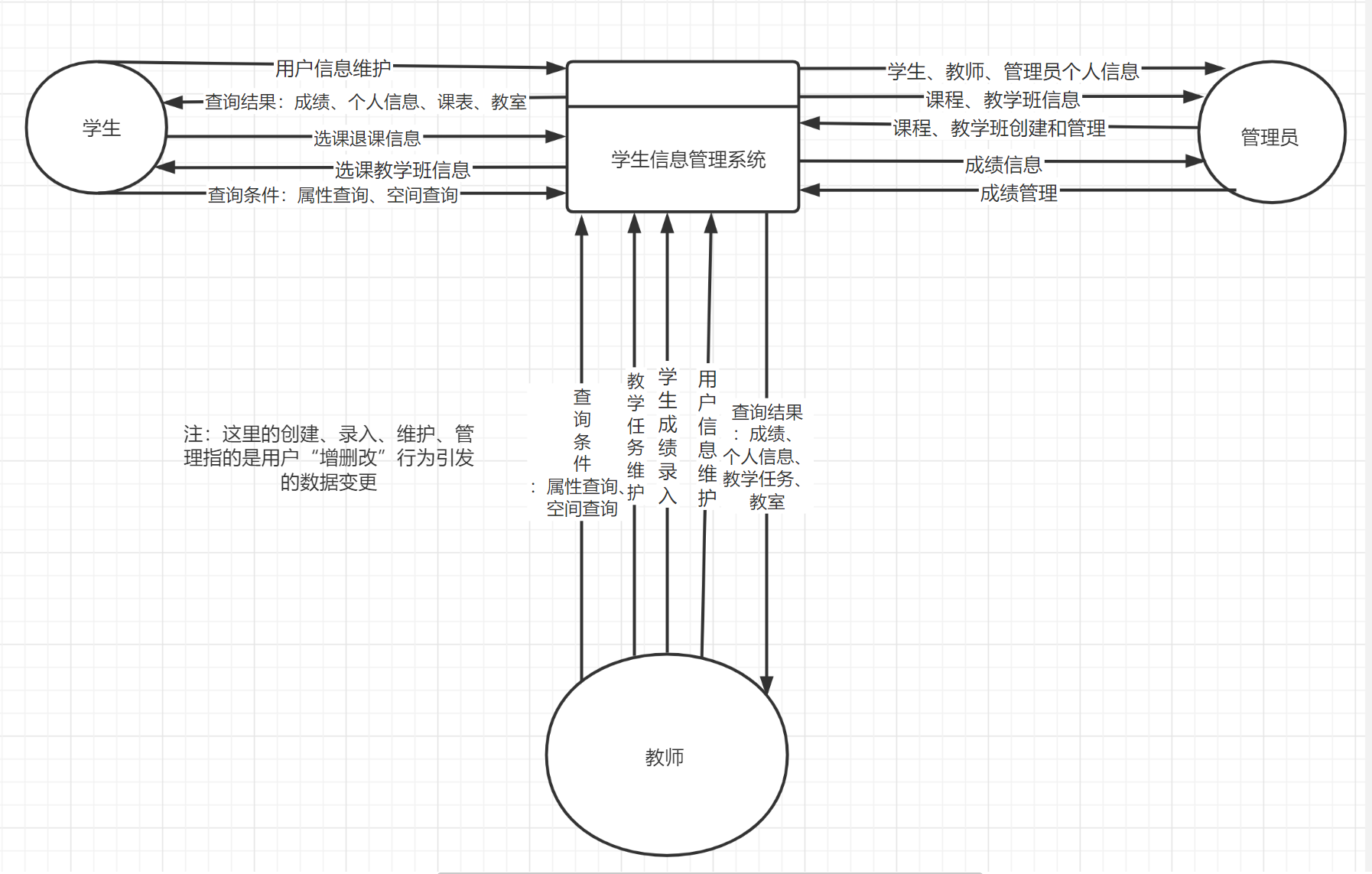
表2 数据结构列表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 数据结构名 | 含义说明 | 组成数据项 |
| 1 | 系 | 系信息 | 系编号、系名、系办公室电话、（所属）学院 |
| 2 | 教师 | 教师信息 | 教师编号、姓名、职称、联系方式、系编号 |
| 3 | 班 | 行政班信息 | 班级编号、系编号、（对应）专业 |
| 4 | 学生 | 学生信息 | 学号、姓名、性别、出生日期、班级编号 |
| 5 | 教室 | 教室信息 | 教室编号、楼层、容纳人数、空间位置 |
| 6 | 课程 | 课程信息 | 课程代号、课程名、课程类型、学分 |
| 7 | 教学班 | 教学班信息 | 课程教学ID、上课时间、教室编号、教师编号、计划上限、已选人数、课程代号 |
| 8 | 课程学生SC | 选课记录 | 学号、课程教学ID、成绩 |
| 9 | 用户安全信息 | 用户安全相关 | 用户名、密码、身份、手机号码、照片 |

表3 数据流列表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 数据流名 | 来源 | 去向 |
| 1 | 登录 | 三方 | 无 |
| 2 | 查询 | All | 无 |
| 3 | 学生信息修改 | 学生 | Student |
| 4 | 教师信息修改 | 教师 | Teacher |
| 5 | 学生选课 | 教学班表（读） | 课程学生SC（写） |
| 6 | 学生成绩录入 | 课程学生SC | 课程学生SC |
| 7 | 地图选课、  排课 | 课程表、教学班表 | 课程表、教学班表 |

数据流图



### 三、逻辑结构设计

**1）关系模式设计**

①系(系编号, 系名, 系办公室电话, 学院)

②教师(教师编号, 姓名, 职称, 联系方式, 系编号) 外码：系编号

③班(班级编号 ,系编号, 专业) 外码：系编号

④学生(学号, 姓名, 性别, 出生日期, 班级编号) 外码：班级编号

⑤教室(教室编号, 楼层, 容纳人数, 空间位置)

⑥课程(课程代号, 课程名, 课程类型, 学分)

⑦课程学生SC(学号, 课程教学ID, 成绩) 外码：学号、课程教学ID

⑧教学班(课程教学ID, 上课时间, 教室编号, 教师编号, 计划上限, 已选人数, 课程代号)

外码：教室编号、教师编号、课程代号

⑨用户安全信息(用户名, 密码, 身份, 手机号码, 用户照片)

**2）关系模式分析**

完成概念设计后，我们要结合各级E-R图、总E-R图和实体间的相互关系进行实体-联系图向**关系模式转化**。

“一对一联系,不必要单独成为一个关系，可以将它与联系中的任何一方实体转化成的关系合并，一般与元祖较少的关系合并；一对多联系,不必要单独作为一个关系模式，可将其与联系中的N方实体转化成的关系合并；多对多联系,必须单独成为一个关系模式，不能与任何一方实体联系如果转换为一个独立的关系模式，则与该联系相连的各实体的码以及联系本身的属性均转换为关系的属性。”

根据该理论我们将多对多关系的实体进组合，生成诸如：课程学生SC、课程教师TC、课程教室CC等关系模式。

结合需求分析可知：1、同一门课程同一位老师可以开设多个教学班2、每一个教学班对应一位老师3、每一条成绩信息都要能够找到对应的授课教师。由此可见，以课程号为课程学生SC的外键已经**不能满足我们的实际需求**了。课程学生SC的外键应当是教学班的主码。故我认为：有必要**引入“教学班”这一实体**，为其规定唯一标识ID，并将其作为成绩的参照对象。我们可以借助面向对象的思想进一步优化：

以**课程**为父类，派生出**教学班**

1. 课程与选修行为无关，是该门课程所有开设的教学班的高级抽象，仅保留共有的基本特征
2. 教学班通过包含课程代号继承了课程的属性，并因选修行为联系了诸多实体派生出了时间、空间、对象的属性。
3. 课程以课程代号为主键，教学班以课程教学ID为主键，课程学生SC表以课程教学ID为外键

我们可以得到如下关系模式：

课程(课程代号, 课程名, 课程类型, 学分)

课程学生SC(学号, 课程教学ID, 成绩) 外码：学号、课程教学ID

教学班(课程教学ID, 上课时间, 教室编号, 教师编号, 计划上限, 已选人数, 课程代号)

外码：教室编号、教师编号、课程代号

需要明确的是：选课界面的基础是**教学班表**，用户看到的是开设的教学班信息。个人课表和成绩管理的基础是**课程学生SC**，是相应选课记录的集合。这样，教务管理系统的核心功能得以保障。

图一为利用脚本在dbdiagram.io中生成的关系图，图二在management studio自动生成的关系图。可见，management studio的关系图虽然不算精美，但是对实体间的参照关系有着更加清晰、直观的反映。在构建复杂视图的时候，我们可以参考这样的关系图，更快地完成设计。（见下一页）

图1

图2

### 四、物理设计

**1）存储安排**：对基本数据表按照索引方法存放在磁盘的某一具体区域，而对于经常进行连接操作的多个关系则建立聚簇存取的方法，并存放在磁盘的另一区域。

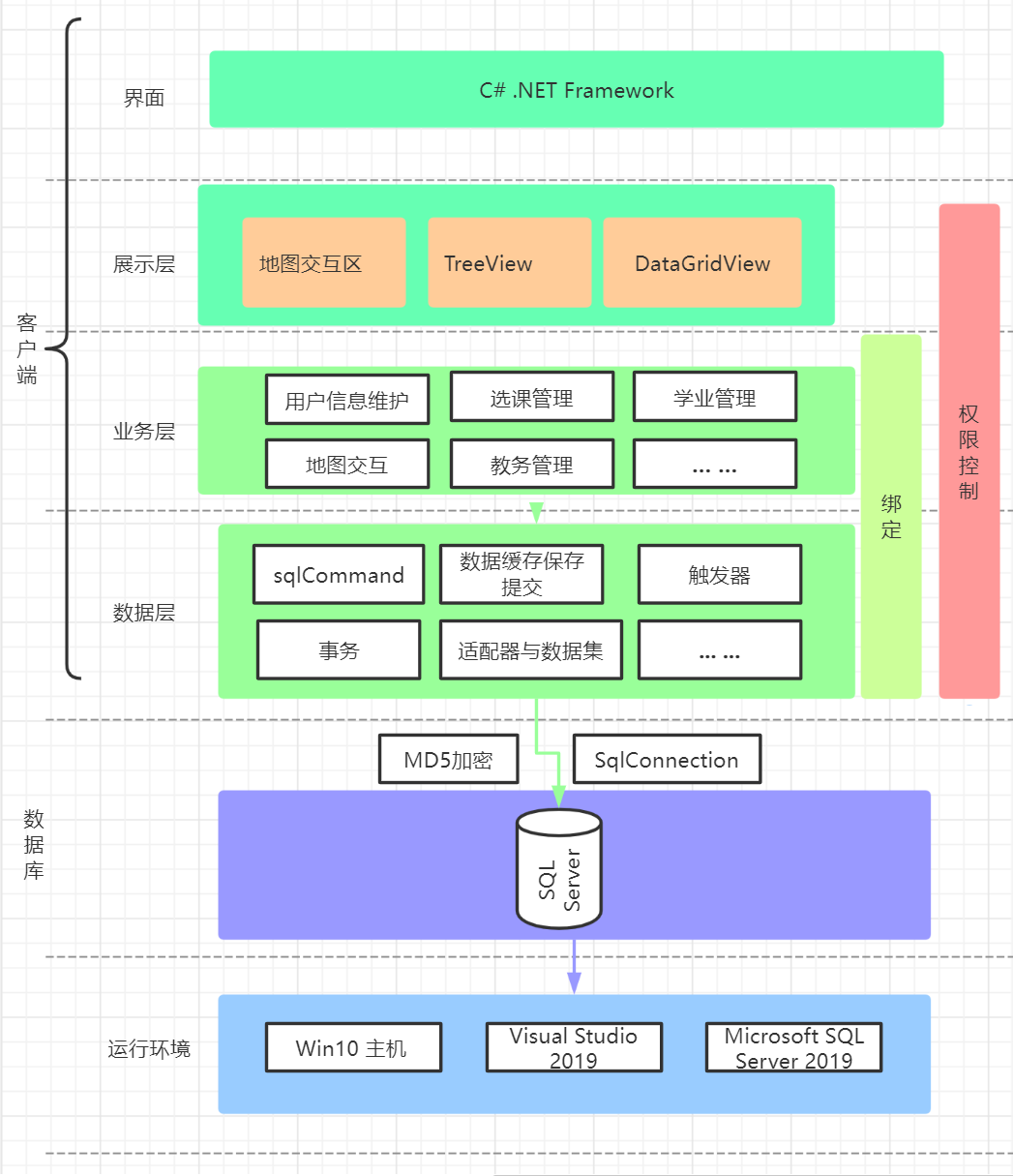
**2）存储方法**：我们可以采用聚簇存取的方法完善物理设计。如果将同一个系的学生元组集中存放，则每读一个物理块可以得到多个满足条件的元组，从而显著地减少了访问磁盘的次数。聚簇可以提高部分性能，但建立和维护聚簇也会增加开销。

**3）存储路径优化**： 数据库的两个重要文件：主要数据文件（.mdf）和事务日志文件（.ldf），一般情况下都会默认存储在Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\DATA这条路径下。结合数据库存储结构优化的思想，我们可以将较大的课程学生SC表分放在两个磁盘上，以加快存取速度。我们还可以将日志文件和数据库对象（表、索引等）放在不同的磁盘上，以改善系统性能。

### 五、系统实施

【系统架构】

教务管理系统2.0版本采用C/S架构实现，开发工具的版本信息见运行维护部分，系统架构如下图所示：



【客户端系统结构】



（无缩放的结构图见附录文件）

【连接实现】

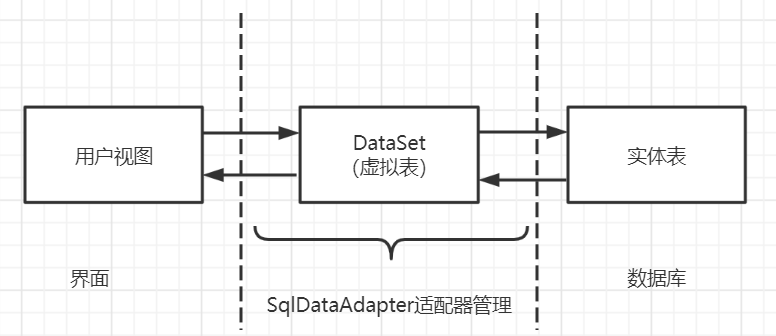
1）数据库连接采用SqlConnection类提供的字符串连接方法，须在命名空间追加：

using System.Data.SqlClient;

2）自定义类Door类，封装了SqlCommand的Connection(),ExecuteNonQuery(),ExecuteReader()方法。将SQL语句传给Door的实例对象即可执行。执行指定的SQL语句是数据库与平台间交互的基本方法，每一个行为都为其指定SQL语句。

3）我们还可以利用适配器（SqlDataAdapter），将数据集（DataSet）与实体表绑定。一旦绑定，界面上呈现的表与数据库中的实体表就实现了数据上的关联了。

运行机理如图所示：



对于简单表来说，我们甚至不需要关心“增删改查”具体执行了怎样的SQL语句，因为它们是同步的。但是如果数据库侧绑定的是视图，而且是多表连接的视图，就会遇到技术障碍。我们都知道：**对于多表连接的视图，DBMS难以通过简单的消解法对相应的表进行修改。**（因为Update的对象、字段是不明确的）

绑定，相当于自动生成了SQL语句。既然绑定多表连接的视图不能自动生成Update语句，我们必须**自行指定UpdateCommand**。方法如下：

1、建立好DataGridView（用户视图）的索引，可以是字段名

2、写好基本表的Update语句，将允许修改的字段的修改值用参数替代

3、将参数与索引绑定

4、SqlCommandBuilder执行适配器的重构

完成后适配器就可以正确Update了！

重新指定适配器Update语句的代码实现：

**绑定代码的实现：**



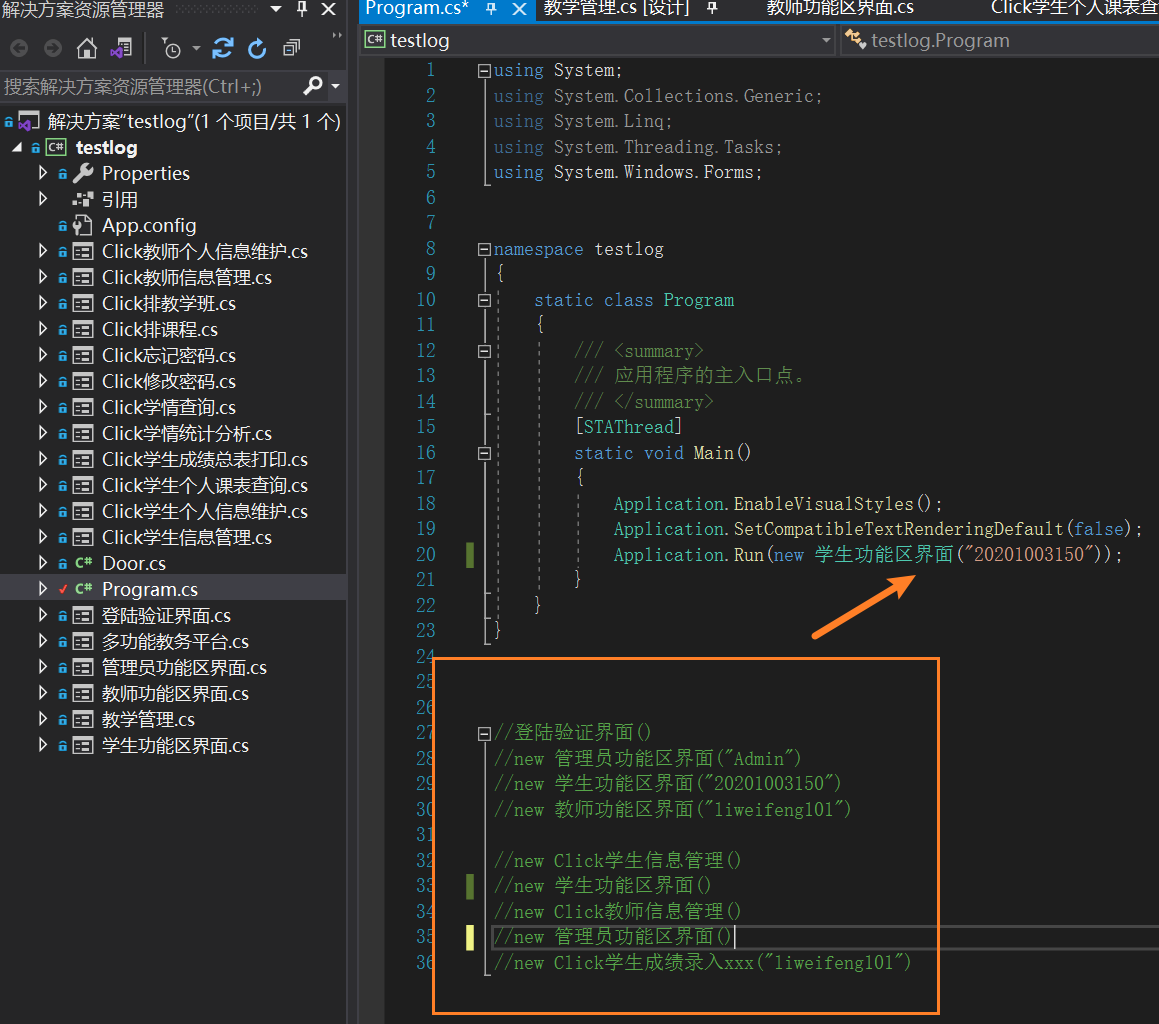
快速绑定



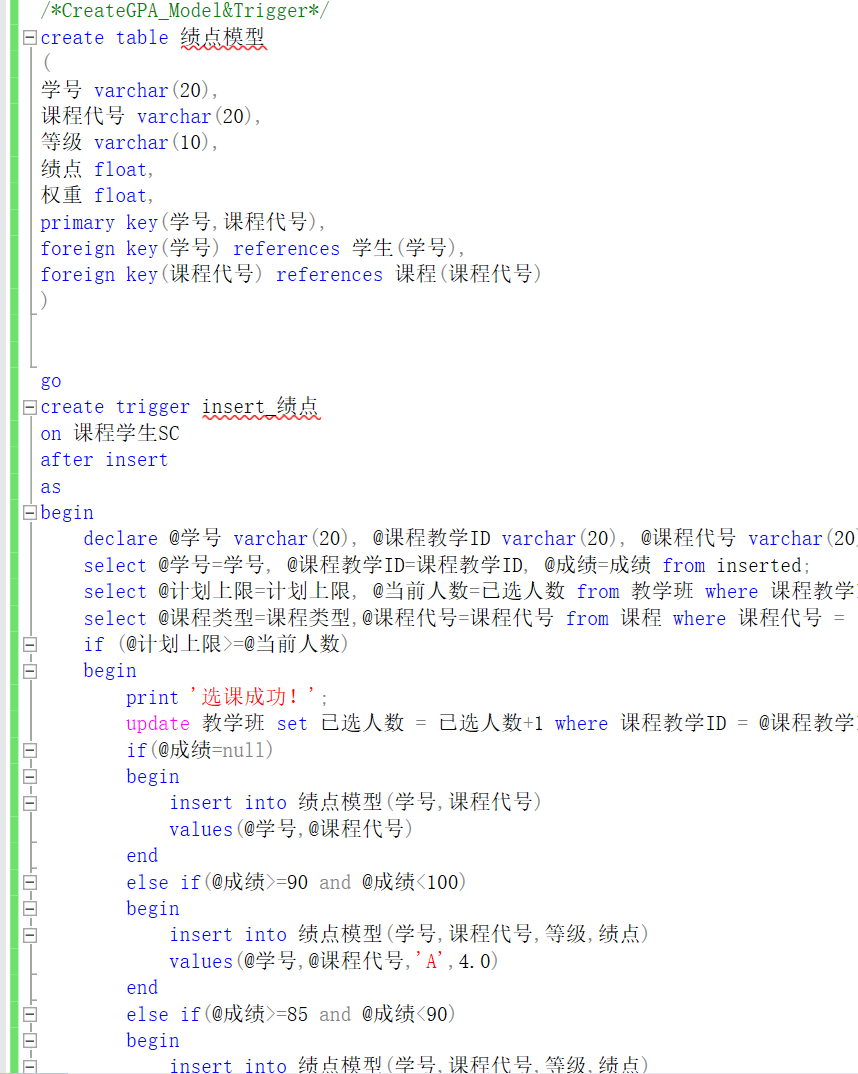
指定SqlCommand，设定参数，重构适配器

【测试】

1）实施了全流程测试和针对性测试



2）模拟生成了32位同学的数据，每人10-15条左右的选课记录，为绩点模型提供了较为丰富的测试样本。绩点模型的数据由课程学生SC表配合触发器间接导入



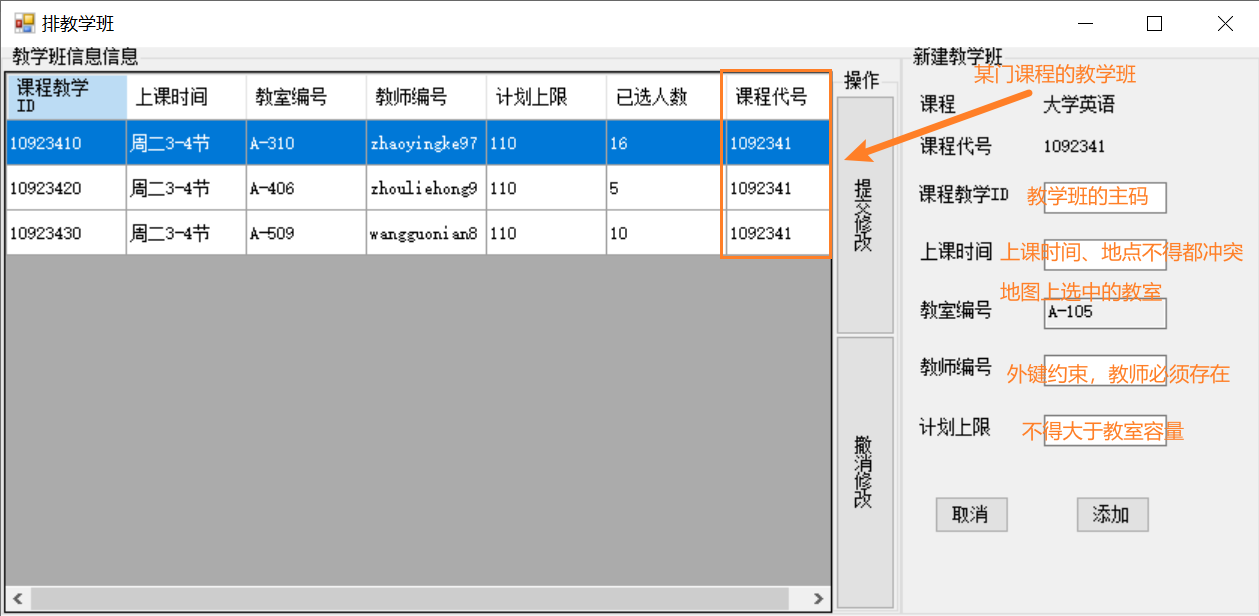
个人成绩样例：

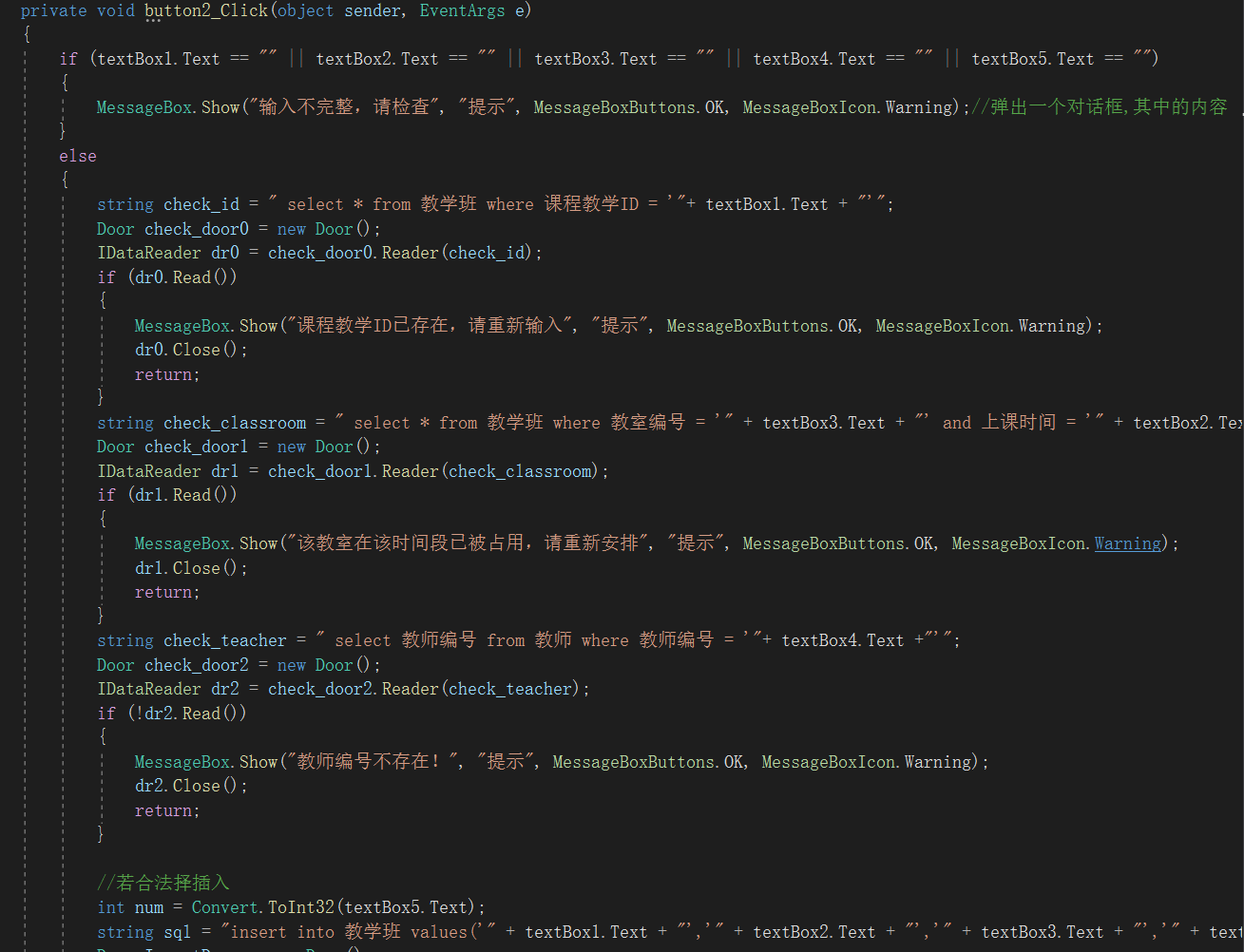


学院学情样例：



3）功能的健壮性除了数据库的完整性约束保障，还需要我们自行补充：





### 六、运行维护

**（一）运行环境**

|  |  |
| --- | --- |
| SQL Server Management Studio | 15.0.18206.0 |
| Microsoft Analysis Services 客户端工具 | 15.0.1567.0 |
| Microsoft 数据访问组件 (MDAC) | 10.0.19041.1 |
| Microsoft MSXML | 3.0 6.0 |
| Microsoft Internet Explorer | 9.11.19041.0 |
| Microsoft .NET Framework | 4.8.04084 |
| 操作系统 | 10.0.19042 |
| Microsoft Visual Studio Community 2019 | 16.11.9 |

**（二）系统维护**

**1）界面系统维护**

实现更多功能、测试改进以增强系统健壮性、优化布局提升用户体验等都是系统维护的任务。

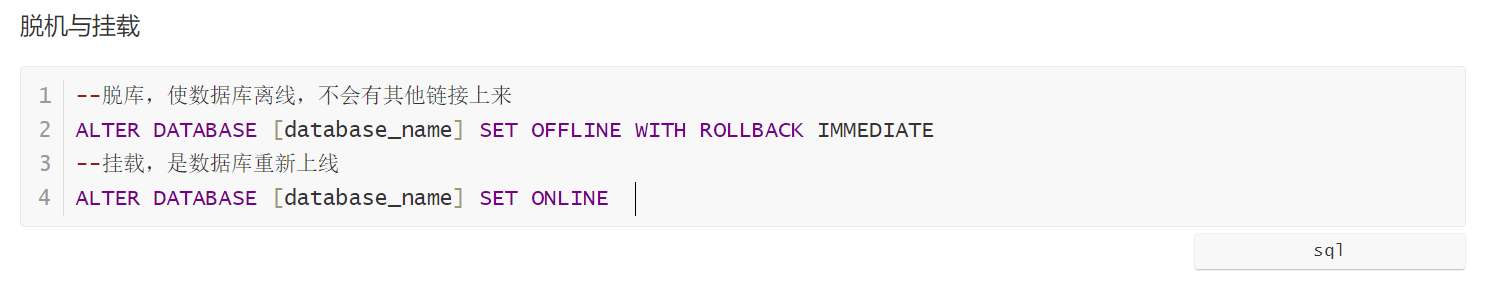
界面系统的维护可以借助Github等作为版本控制的工具，定期更新优化，合理采纳他人的request，积极解决issues，发扬开源精神……我们可以做的更好。

**2）数据库维护**

数据库的维护包括了备份系统数据、恢复数据库系统、产生用户信息表，并为信息表授权、监视系统运行状况，及时处理系统错误、保证系统数据安全，周期更改用户口令等。

将存储的Mdf与Ldf文件从数据库分离出去备份可以实现静态、海量转储。分离mdf和ldf文件需要先脱机

【脱机】

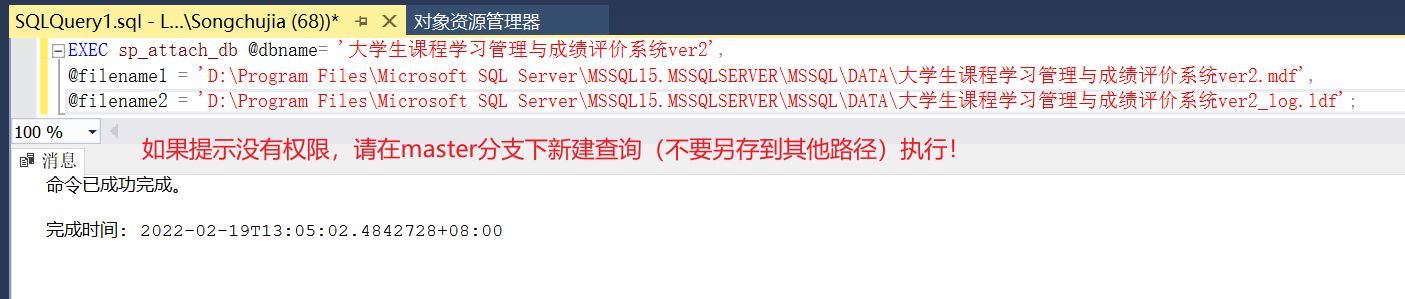


【附加Mdf和ldf】

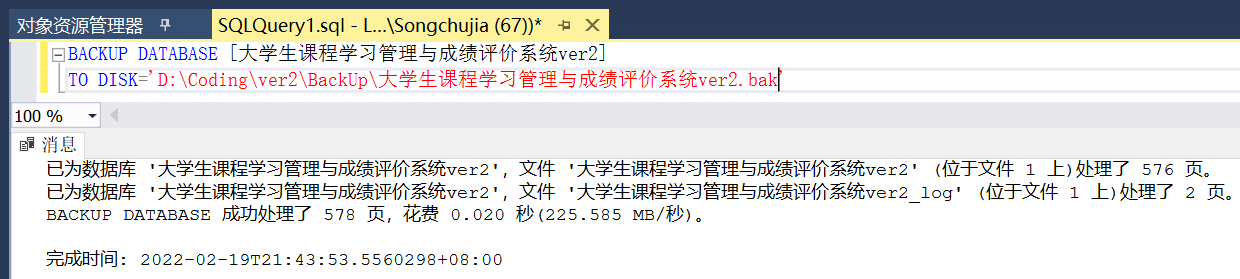
附加Mdf与Ldf文件可以是我们重新连接数据库的第一步。附加过程往往会出现系统对文件的权限不够，按照以下两点操作即可解决：

1、修改文件属性-系统完全控制

2、在DBMS系统默认的路径下新建master分支的查询，再执行SQL语句（在其他路径下可能依然会提示没有权限）

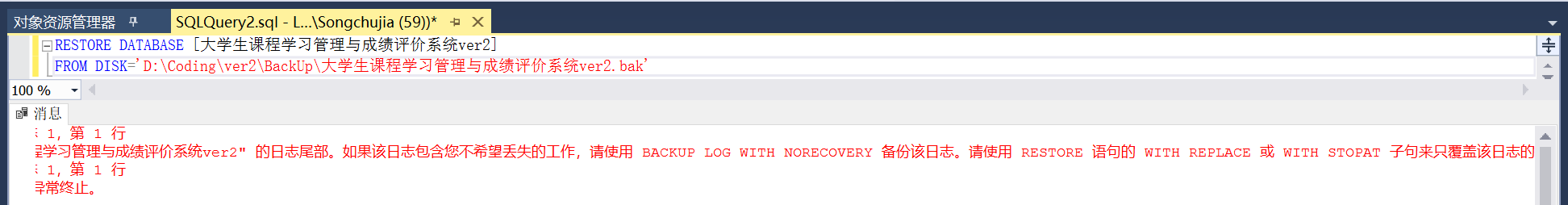


【生成备份文件】



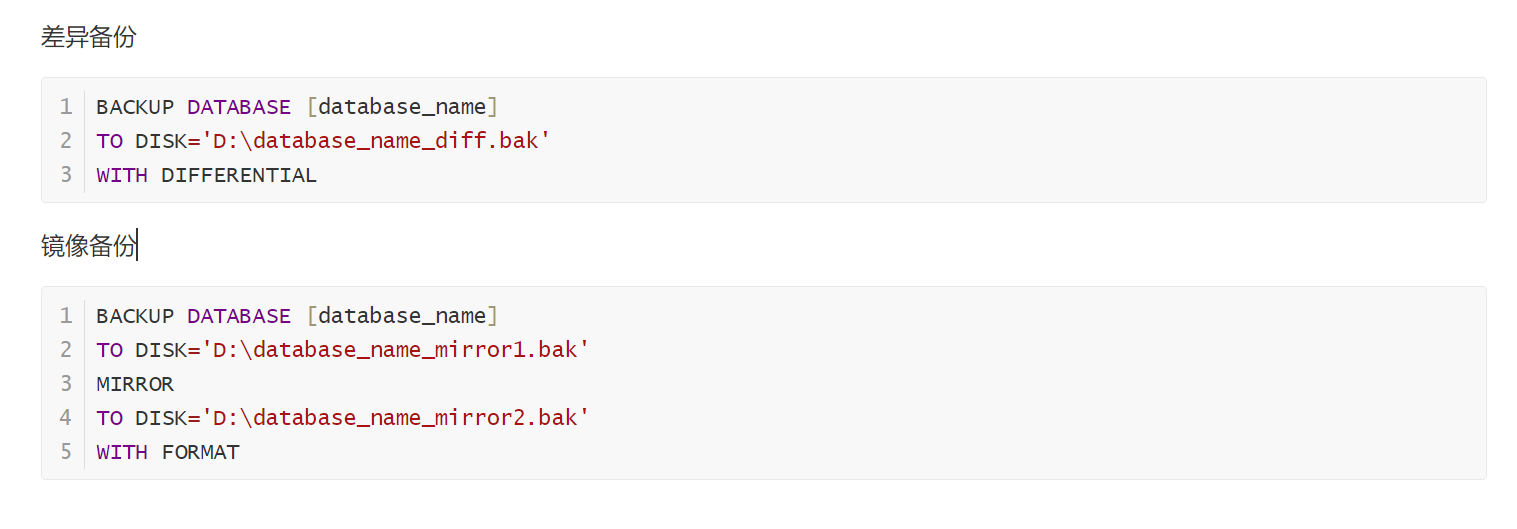
【载入备份文件】

数据库载入备份前需要BACKUP LOG WITH NORECOVERY备份日志，否则会丢失工作记录



【其他备份方式】

我们还可以进行差异备份和镜像备份：

（图片摘自我的数据库学习笔记）

### 七、系统操作指南

系统架构详见附录，该系统实现了：用户信息的管理维护，课程及教学班的管理维护，选课系统及教学管理，构建绩点模型进行GPA计算并服务于学情统计，地图交互空间查询以及空间对象的属性查询。

系统特色：

①系统将部分功能集成在多功能教务管理平台界面，该界面通过重载构造函数和功能函数，以及控件的展示与隐藏，使得不同角色调用同一个界面可以实现相近或完全不同的功能。

②“增删改查”往往在同一个界面甚至在同一张表上实现，可视性和便利性相较1.0版本都有所提升。

③尽量保持界面操作逻辑的一致，用户尝试过一个就能触类旁通了。

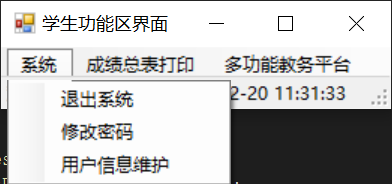
**1）登陆界面**



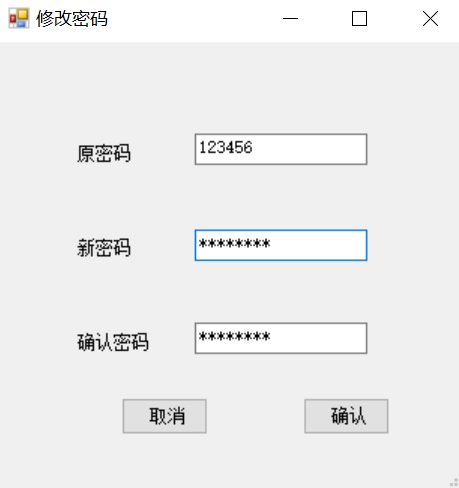
需要验证码登陆，用户密码进行MD5加密保障安全

**2）系统功能区**

学生、教师和管理员的系统功能区提供用户个人信息的维护。



修改密码，加密后存储密文



部分信息可修改，部分信息只读



**3）成绩总表打印**

提供个人信息、成绩信息（分数和绩点）、已修学分核算、平均学分绩点、打印时间



**4）学生多功能教务管理平台**

【选课界面及教室导引】

①点击表中最后一列即可选课，自动刷新，已选教学班浅蓝色标识。

每门课程只能选择一个教学班



②点击退课，自动刷新

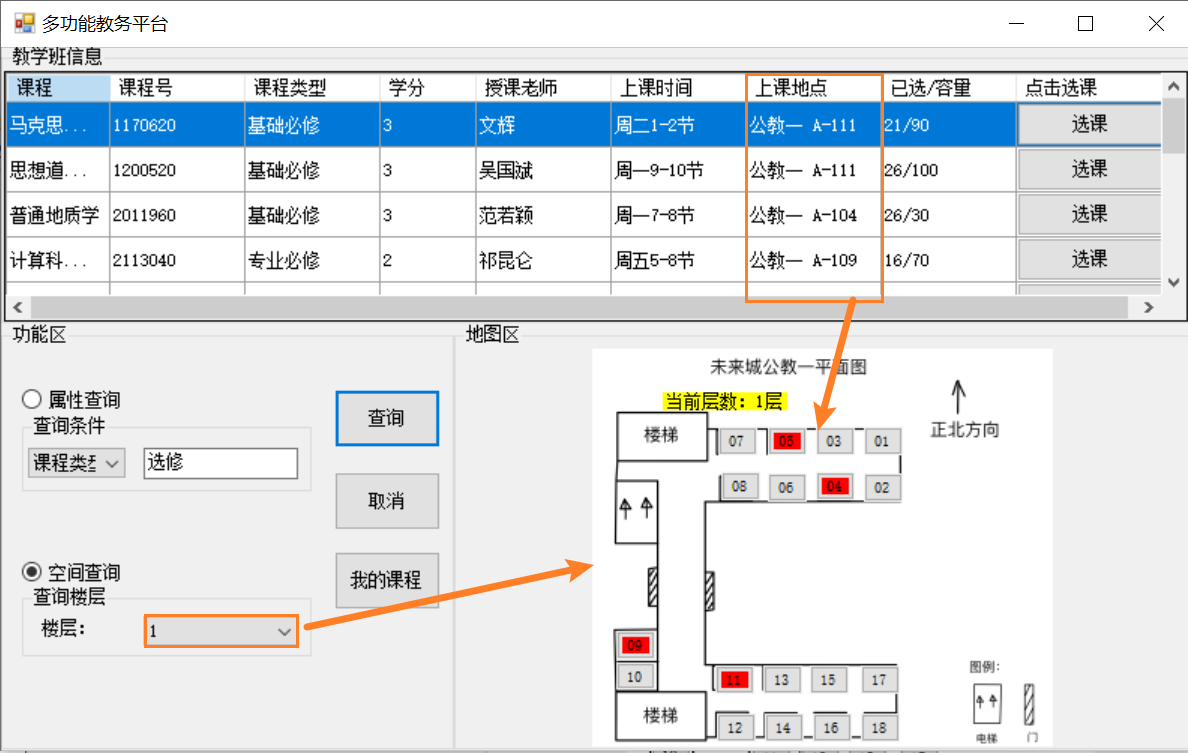


【多条件属性查询】



【空间查询】

①查询楼层教室占用情况



②点击教室，（自动）展示不同时间段教室占用情况



**5）教师多功能教务管理平台**

继承自学生教务管理平台，派生了教务工作区：包括个人教学任务展示，成绩管理模块



**6）管理员多功能教务管理平台**

继承自学生、教师管理平台，派生了课程、教学班管理，排课功能

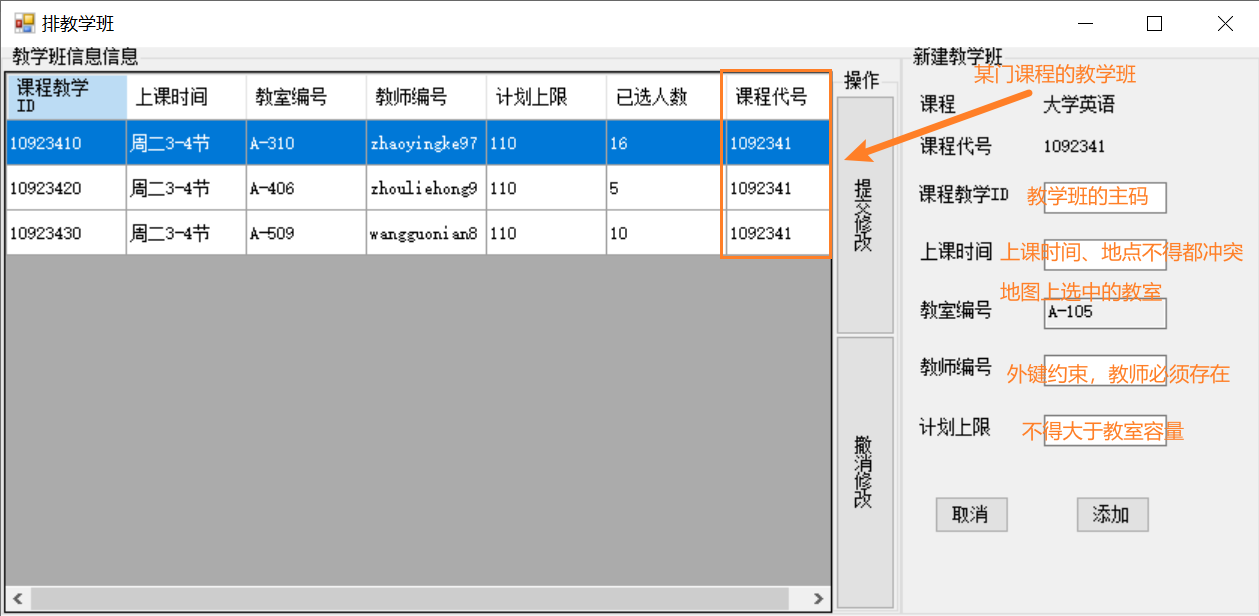
①管理课程、教学班或者选中教室开始排课



②排课程



③排教学班



**7）学情查询**

双击TreeView的不同节点：学院、系、专业、班级会得到不同层级的排名，也可以在条件查询中输入查询



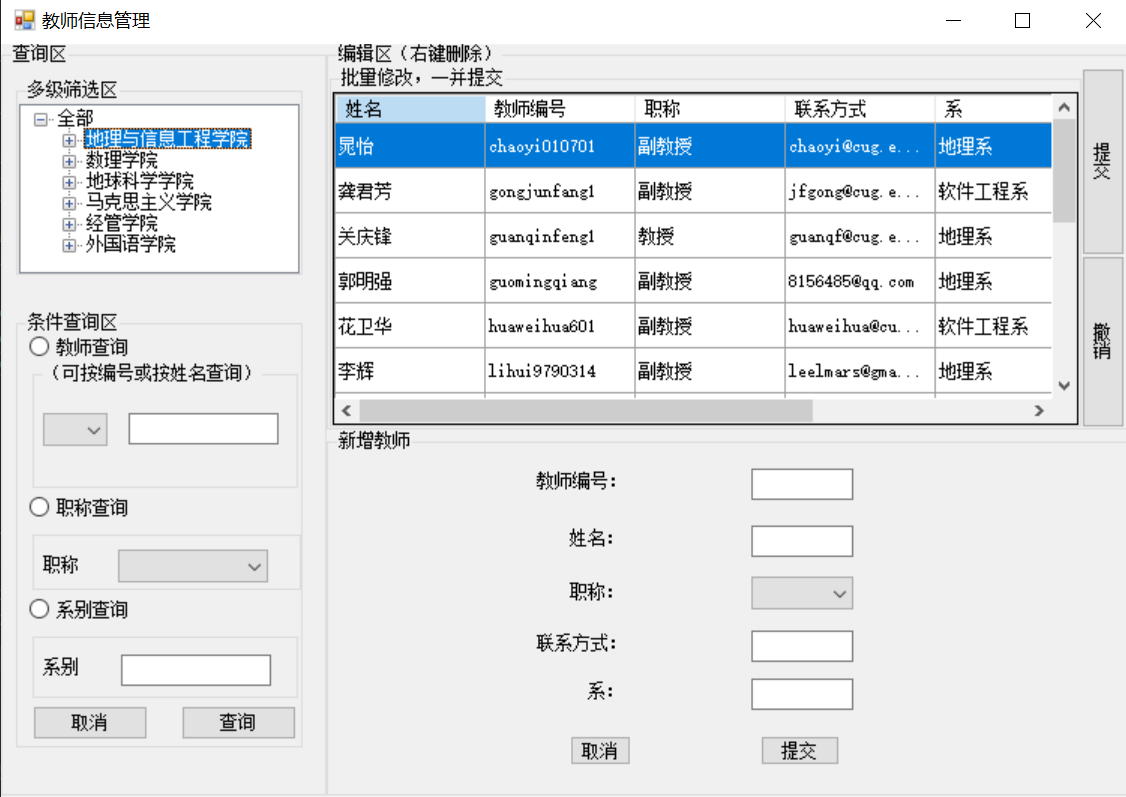


**8）学生信息管理**

继承自学情查询，操作逻辑大致相同。集成“增删改查”，可批量编辑表格修改再提交

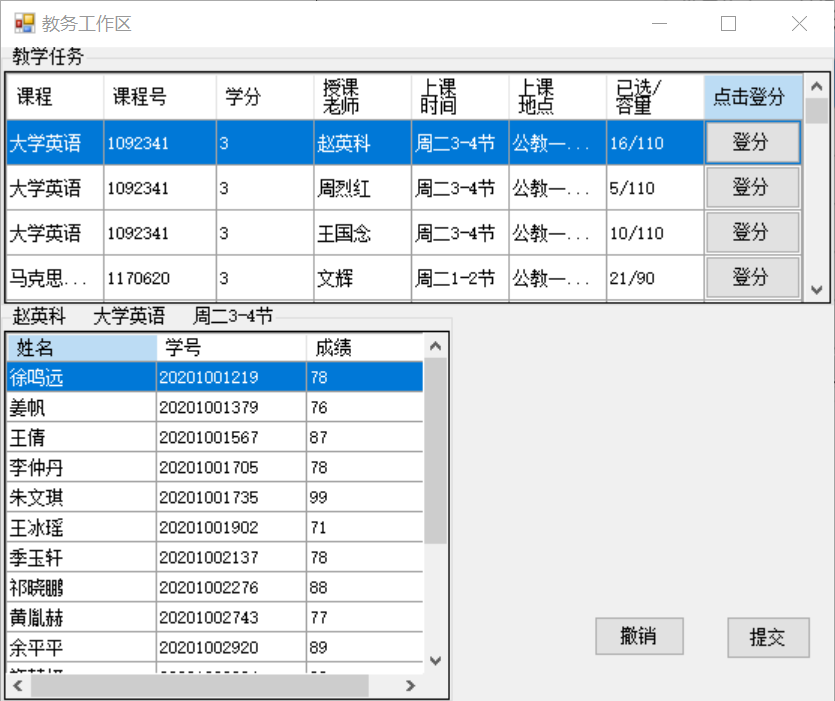
**9）学生信息管理**

继承自学生信息管理，“增删改查”集成。



**10）教学管理**

继承自教务工作区，操作逻辑完全相同。管理员权限更大，可以对全校的教学班的成绩进行管理



### 八、附录

项目地址：[Polumm/DataBaseVer2 (github.com)](https://github.com/Polumm/DataBaseVer2)

2.0版本的改进摘要可查看README.md，压缩包文件概况如下：



### 九、总结

第一次经历这样完整的项目开发，收获很多

1）作为纯新手，最开始的时候不必过于拘泥细节和顶层设计。我们总是要不断地迭代升级的，一遍过不太可能。尤其是面对一个完全没有经验的项目，快速推进有利于在短时间内把握项目，然后结合实践中遇到的问题，就可以给出更优的方案。

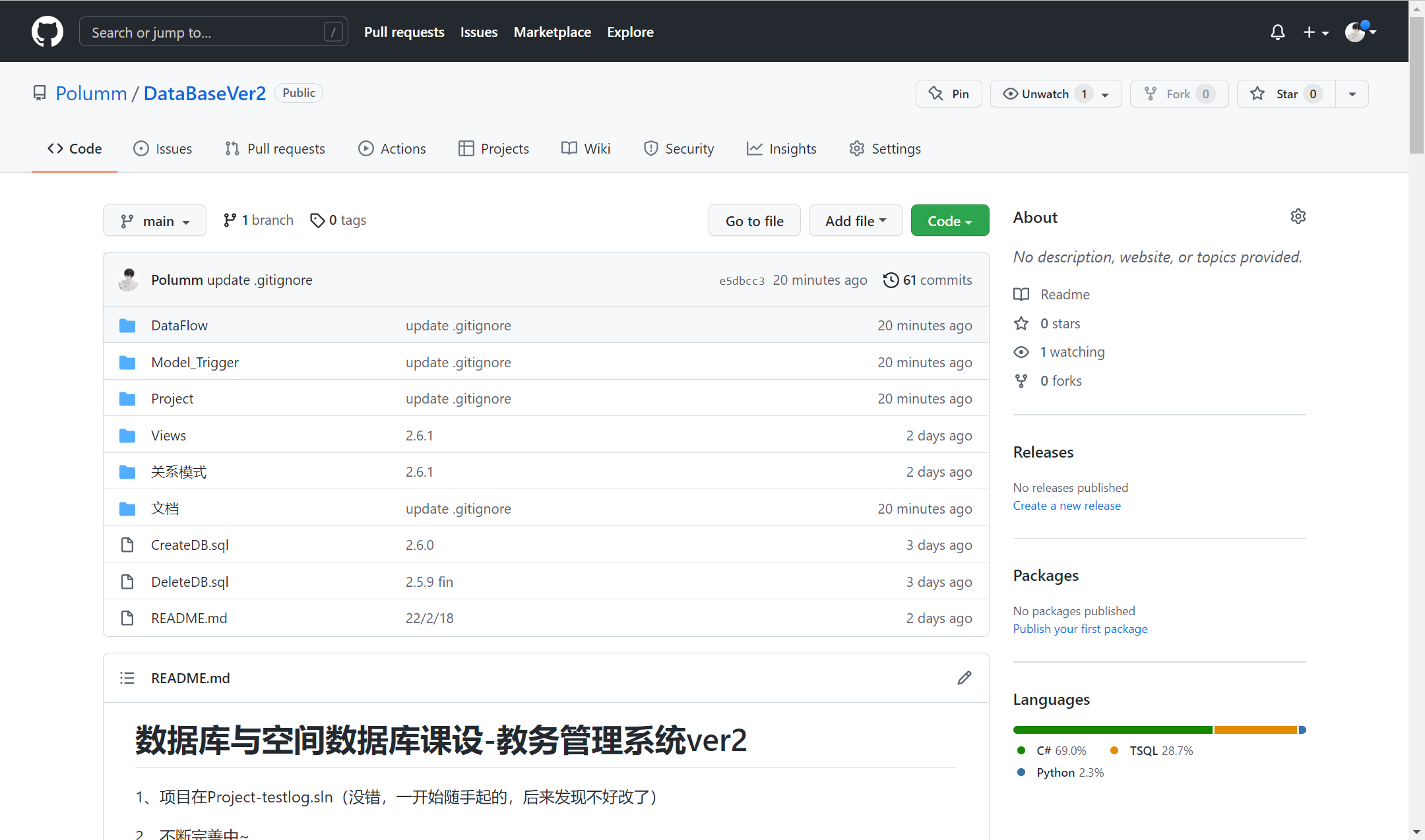
敏捷开发的思想给了我很大的帮助。功能开发前我先进行分类，把功能相近的放在一组，着力先实现一个模型，再将这个模型向它的“兄弟姐妹”推广。比如学生信息管理、教师信息管理、学情查询的需求都是在树状结构中筛选、在表格中展示，并且管理维护类的功能都有批量编辑的需求，操作逻辑也大致相同，那么先着力实现该模型的核心功能，再将其继承派生就把相近的需求也实现了。

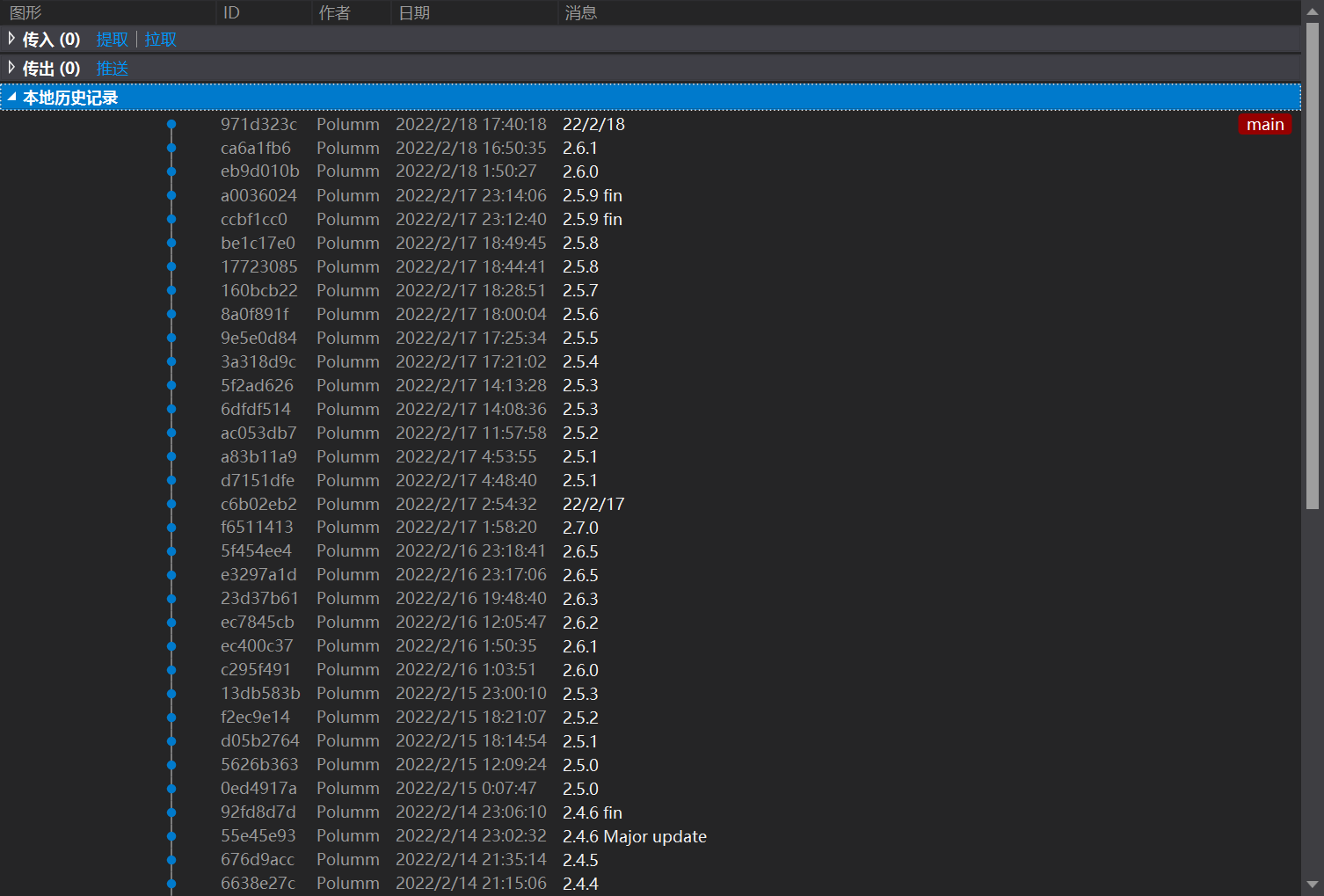
还有就是测试要快速有效。我编写了爬虫，自己又模拟生成了大量的数据，就是为了有更好的测试效果，结果关系模式重新设计了后，又得调整数据，“被数据拖着跑”会很累。最开始的时候，我们应当用少量的数据去检验模式的合理性和系统的健壮性，利用测试的结果指导我们迭代升级，再逐步地扩大数据的规模。

可以使用TAPD这样的项目管理平台，提高开发效率。



使用GitHub版本控制有助于放开手脚，大胆尝试





2）练习了利用通用爬虫搭配聚焦爬虫从导师简介页面和个人页面爬取数据。建立数据库需要有自主收集、处理数据的能力，我会继续积累这方面的技能。



（部分老师个人页面变化较大，后期可采用bs4或者xpath等进行提取）

3）触发器和约束条件的一些思考

本次课程，部分功能我尝试用触发器实现，部分约束效果利用参照完整性约束实现。但是我发现触发器也有它的局限性，比如，部分数据由触发器间接导入，导入效率降低了，有时候会卡很久。外键约束则可能会在意想不到的地方增大某些功能的实现难度。其实约束就是一种保护性法则，有些逻辑简单的约束我们可以直接在用户层的代码中实现，比如：一些“增删改”前先对某些表进性合法性检查。这样更加灵活，可以随时适应需求的变化。

4）在交互系统的设计开发中，考虑用户的体验非常重要。比如选课要有筛选条件，选了一门教学班就应当有所标识。而且最好能够让用户的操作连续起来，“增删改查”不要孤立地进行。

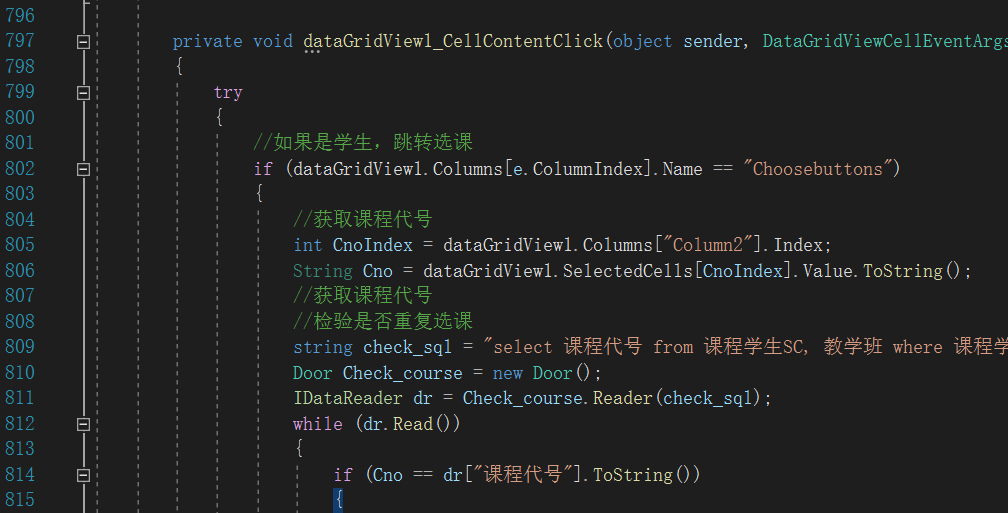
5）通过这次课设，我对数据流和数据“生命周期”中的环节有了更具体的认识。学会利用适配器和数据集绑定用户视图和数据库基本表，并能够解决多表连接视图的同步修改问题。数据集是用户与数据库的桥梁，保存行为是将用户的修改存入数据集（缓冲区），而提交行为是将数据集的修改写入数据库。

6）这次课设中，我接触了ArcEngine，接下来会学习利用ArcEngine实现更强大的交互功能。

7）学会使用保护性措施、设计更健壮的程序：

①危险行为“增删改”最好在try语句中，catch错误提示

②最好不要把程序“写死”，并且最好利用自己指定的索引去寻访（自动生成的数字索引可能会发生意想不到的变化），比如：获取选中行的课程号，如果采用用户看到的视图中列的位置作为下标（假设看起来是第0列） 去寻访，写成SelectedCells[0]。那么当表尾追加新列时，在表刷新前，新列将会是第0列，这时就会获取到错误的数据。



感谢老师一直以来的指导和启发，和老师交流常有茅塞顿开的感觉。在老师的建议下思考和改进，我取得了很大的进步。我会继续努力，不断学习，追求更高的境界。