目录

[引言](https://blog.csdn.net/m0_58153897/article/details/141997914#%E5%BC%95%E8%A8%80)

[一.需求设计说明书](https://blog.csdn.net/m0_58153897/article/details/141997914#%E4%B8%80.%E9%9C%80%E6%B1%82%E8%AE%BE%E8%AE%A1%E8%AF%B4%E6%98%8E%E4%B9%A6)

[1．需求分析](https://blog.csdn.net/m0_58153897/article/details/141997914#1%EF%BC%8E%E9%9C%80%E6%B1%82%E5%88%86%E6%9E%90)

[2.系统背景](https://blog.csdn.net/m0_58153897/article/details/141997914#2.%E7%B3%BB%E7%BB%9F%E8%83%8C%E6%99%AF)

[3.系统目标](https://blog.csdn.net/m0_58153897/article/details/141997914#3.%E7%B3%BB%E7%BB%9F%E7%9B%AE%E6%A0%87)

[4.人员分配](https://blog.csdn.net/m0_58153897/article/details/141997914#4.%E4%BA%BA%E5%91%98%E5%88%86%E9%85%8D)

[5.数据流程图（DFD）](https://blog.csdn.net/m0_58153897/article/details/141997914#5.%E6%95%B0%E6%8D%AE%E6%B5%81%E7%A8%8B%E5%9B%BE%EF%BC%88DFD%EF%BC%89)

[二.概念结构设计](https://blog.csdn.net/m0_58153897/article/details/141997914#%E4%BA%8C.%E6%A6%82%E5%BF%B5%E7%BB%93%E6%9E%84%E8%AE%BE%E8%AE%A1)

[1.局部E-R图](https://blog.csdn.net/m0_58153897/article/details/141997914#1.%E5%B1%80%E9%83%A8E-R%E5%9B%BE)

[（1）供应商](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%881%EF%BC%89%E4%BE%9B%E5%BA%94%E5%95%86)

[（2）货物](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%882%EF%BC%89%E8%B4%A7%E7%89%A9)

[（3）客户](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%883%EF%BC%89%E5%AE%A2%E6%88%B7)

[（4）仓库](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%884%EF%BC%89%E4%BB%93%E5%BA%93)

[（5）管理员](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%885%EF%BC%89%E7%AE%A1%E7%90%86%E5%91%98)

[2.全局E-R图](https://blog.csdn.net/m0_58153897/article/details/141997914#2.%E5%85%A8%E5%B1%80E-R%E5%9B%BE)

[3.逻辑结构设计](https://blog.csdn.net/m0_58153897/article/details/141997914#3.%E9%80%BB%E8%BE%91%E7%BB%93%E6%9E%84%E8%AE%BE%E8%AE%A1)

[4.物理结构设计](https://blog.csdn.net/m0_58153897/article/details/141997914#4.%E7%89%A9%E7%90%86%E7%BB%93%E6%9E%84%E8%AE%BE%E8%AE%A1)

[表结构](https://blog.csdn.net/m0_58153897/article/details/141997914#%E8%A1%A8%E7%BB%93%E6%9E%84)

[数据模型](https://blog.csdn.net/m0_58153897/article/details/141997914#%E6%95%B0%E6%8D%AE%E6%A8%A1%E5%9E%8B)

[三.数据库](https://blog.csdn.net/m0_58153897/article/details/141997914#%E4%B8%89.%E6%95%B0%E6%8D%AE%E5%BA%93)

[1.数据库的实施](https://blog.csdn.net/m0_58153897/article/details/141997914#1.%E6%95%B0%E6%8D%AE%E5%BA%93%E7%9A%84%E5%AE%9E%E6%96%BD)

[（1）创建数据库](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%881%EF%BC%89%E5%88%9B%E5%BB%BA%E6%95%B0%E6%8D%AE%E5%BA%93)

[（2）创建表](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%882%EF%BC%89%E5%88%9B%E5%BB%BA%E8%A1%A8)

[2.数据初始化](https://blog.csdn.net/m0_58153897/article/details/141997914#2.%E6%95%B0%E6%8D%AE%E5%88%9D%E5%A7%8B%E5%8C%96)

[供应商](https://blog.csdn.net/m0_58153897/article/details/141997914#%E4%BE%9B%E5%BA%94%E5%95%86)

[货物](https://blog.csdn.net/m0_58153897/article/details/141997914#%E8%B4%A7%E7%89%A9)

[客户](https://blog.csdn.net/m0_58153897/article/details/141997914#%E5%AE%A2%E6%88%B7)

[仓库](https://blog.csdn.net/m0_58153897/article/details/141997914#%E4%BB%93%E5%BA%93)

[管理员](https://blog.csdn.net/m0_58153897/article/details/141997914#%E7%AE%A1%E7%90%86%E5%91%98)

[入库](https://blog.csdn.net/m0_58153897/article/details/141997914#%E5%85%A5%E5%BA%93)

[出库](https://blog.csdn.net/m0_58153897/article/details/141997914#%E5%87%BA%E5%BA%93)

[3.可行性分析](https://blog.csdn.net/m0_58153897/article/details/141997914#3.%E5%8F%AF%E8%A1%8C%E6%80%A7%E5%88%86%E6%9E%90)

[（1）技术可行性：](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%881%EF%BC%89%E6%8A%80%E6%9C%AF%E5%8F%AF%E8%A1%8C%E6%80%A7%EF%BC%9A)

[（2）经济可行性：](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%882%EF%BC%89%E7%BB%8F%E6%B5%8E%E5%8F%AF%E8%A1%8C%E6%80%A7%EF%BC%9A)

[（3）运行可行性：](https://blog.csdn.net/m0_58153897/article/details/141997914#%EF%BC%883%EF%BC%89%E8%BF%90%E8%A1%8C%E5%8F%AF%E8%A1%8C%E6%80%A7%EF%BC%9A)

[4.系统功能描述](https://blog.csdn.net/m0_58153897/article/details/141997914#%C2%A04.%E7%B3%BB%E7%BB%9F%E5%8A%9F%E8%83%BD%E6%8F%8F%E8%BF%B0)

[5.数据库实施与维护](https://blog.csdn.net/m0_58153897/article/details/141997914#5.%E6%95%B0%E6%8D%AE%E5%BA%93%E5%AE%9E%E6%96%BD%E4%B8%8E%E7%BB%B4%E6%8A%A4)

[（1）仓库管理](https://blog.csdn.net/m0_58153897/article/details/141997914#%C2%A0%EF%BC%881%EF%BC%89%E4%BB%93%E5%BA%93%E7%AE%A1%E7%90%86)

[（2）入库管理](https://blog.csdn.net/m0_58153897/article/details/141997914#%C2%A0%C2%A0%EF%BC%882%EF%BC%89%E5%85%A5%E5%BA%93%E7%AE%A1%E7%90%86)

[（3）出库管理](https://blog.csdn.net/m0_58153897/article/details/141997914#%C2%A0%EF%BC%883%EF%BC%89%E5%87%BA%E5%BA%93%E7%AE%A1%E7%90%86)

[四、课程设计心得体会](https://blog.csdn.net/m0_58153897/article/details/141997914#%E5%9B%9B%E3%80%81%E8%AF%BE%E7%A8%8B%E8%AE%BE%E8%AE%A1%E5%BF%83%E5%BE%97%E4%BD%93%E4%BC%9A)

## 引言

随着高校信息化的不断推进，学籍管理工作已成为高校管理的核心内容之一。传统的学籍管理模式往往依赖于手工记录或分散的软件系统，导致信息难以集中管理、查询效率低下、数据易出错且维护成本高。为了解决这些问题，构建一个基于SQL Server的高校学籍管理系统显得尤为重要。本系统旨在实现高校学籍信息的全面管理，包括学生基本信息、课程信息、成绩信息、奖惩信息等模块化功能，提升高校学籍管理工作的效率与准确性。

## 一.需求设计说明书

1．需求分析

**需求分析**

**功能需求**

1. **学生基本信息管理**
   * 实现学生个人基本信息的录入、修改、删除及查询功能。
   * 管理学生的班级、院系和专业信息。
2. **课程与成绩管理**
   * 实现课程信息的录入、修改及查询功能。
   * 管理学生的成绩信息，包括课程成绩的录入、修改及查询。
3. **奖惩信息管理**
   * 实现学生奖惩记录的管理功能，包括奖惩信息的新增、修改、删除及查询。
4. **数据完整性与约束**
   * 创建规则用于限制性别字段只能输入“男”或“女”。
   * 确保数据库中各表数据的参照完整性约束。
5. **数据查询功能**
   * 创建视图，用于查询每个学生的学号、姓名、班级、专业及院系信息。

**非功能需求**

1. **系统性能**
   * 确保系统在高并发访问情况下能够保持较高的响应速度。
2. **数据安全性**
   * 实现用户分级权限管理，确保敏感数据的访问与修改仅限授权用户。
3. **扩展性**
   * 系统需具有良好的扩展性，以便未来根据需求增加新功能模块。
4. **易用性**
   * 提供友好的用户界面及简便的操作流程，降低用户的学习成本。

**用户需求**

* **管理员**
  + 管理所有学生的基础信息、成绩信息及奖惩记录。
  + 配置系统规则及数据约束，维护系统正常运行。
* **教职工**
  + 查询和管理学生的成绩、课程及相关信息。
* **学生**
  + 查询个人基本信息、课程成绩及奖惩记录。

**系统功能要求**

1. **实现学生信息、班级、院系、专业等的管理**
   * 系统需支持对学生个人信息、所属班级、院系及专业的增删改查操作。
2. **实现课程、学生成绩信息管理**
   * 系统需支持课程信息的录入、修改与查询，并对学生的成绩信息进行管理。
3. **实现学生的奖惩信息管理**
   * 系统需支持学生奖惩记录的管理，包括新增、编辑、删除和查询功能。
4. **性别字段的输入限制**
   * 系统需创建数据规则，限制性别字段仅能输入“男”或“女”，确保数据录入的规范性。
5. **创建视图进行数据查询**
   * 创建一个视图，能够查询所有学生的学号、姓名、班级、专业及院系信息，便于快速查看。
6. **建立数据库表的参照完整性约束**
   * 在数据库设计中，确保相关表之间的参照完整性，如学生与班级、院系、专业的关联，学生与课程、成绩的关联等。

2. 系统背景

高校学籍管理系统的主要任务是对学生从入学到毕业期间的相关数据进行全面、准确的管理与维护。随着高校规模的扩大和学生数量的增加，手工管理方式已难以满足实际需求，亟需一个信息化、集成化的管理平台。利用SQL Server数据库技术，系统将提供稳定、安全、高效的数据存储与管理功能，实现多用户操作、数据完整性约束及高效的查询支持。

系统的设计与实现以满足高校日常学籍管理的实际需求为目标，结合当下数据库技术的发展趋势，开发出具有较高实用价值的学籍管理信息系统。通过对学生数据的集中管理，系统可支持多维度的数据查询与分析，为高校的教学决策提供重要的数据支撑。

3.系统目标

本系统的目标是通过信息化手段提升高校学籍管理的效率和准确性，为管理人员和学生提供便捷的功能支持，具体目标如下：

1. **提升数据管理效率**
   * 通过集中化管理学生、课程、成绩、奖惩等信息，减少传统手工记录的工作量和出错率。
2. **实现数据的高效查询与分析**
   * 提供多维度的数据查询功能，为管理决策提供数据支持。
3. **确保数据的完整性和安全性**
   * 通过参照完整性约束和权限管理，确保数据的一致性与访问的安全性。
4. **提高用户体验**
   * 提供简洁直观的界面和便捷的操作流程，满足不同用户群体的使用需求。
5. **支持未来扩展**
   * 设计灵活的数据结构和功能模块，便于后续新增功能或适配不同高校的具体需求。

4.人员分配

* **管理员**
  + 管理所有学生的基础信息、成绩信息及奖惩记录。
  + 配置系统规则及数据约束，维护系统正常运行。
* **教职工**
  + 查询和管理学生的成绩、课程及相关信息。
  + 分配并管理教职工与学生、课程的对应关系。
* **学生**
  + 查询个人基本信息、课程成绩及奖惩记录。

5.数据流程图（DFD）

**数据流程图说明：**

1. **外部实体**
   * **学生**：输入和查看个人信息。
   * **教职工**：管理学生信息、课程、成绩和奖惩信息。
   * **管理员**：维护系统规则和参照完整性。
2. **主要处理过程**
   * **学生信息管理**：录入、修改、删除和查询学生信息。
   * **课程和成绩管理**：课程录入与查询，成绩录入与修改。
   * **奖惩信息管理**：新增、查询奖惩记录。
   * **人员分配管理**：分配教职工与课程、学生的关系。
   * **视图查询**：通过视图快速查询学生的综合信息。
3. **数据存储**
   * **学生表**：存储学生基本信息。
   * **课程表**：存储课程信息。
   * **成绩表**：存储学生成绩。
   * **奖惩表**：存储学生奖惩记录。
   * **教职工表**：存储教职工信息。



数据字典

**数据字典**

1. **学生表（Student）**
   * **StudentID**: 学生编号，主键，唯一标识学生（类型：INT）
   * **Name**: 学生姓名（类型：VARCHAR(50)）
   * **Gender**: 性别，仅限“男”或“女”（类型：VARCHAR(10)）
   * **DOB**: 出生日期（类型：DATE）
   * **ClassID**: 所属班级编号，外键（类型：INT）
   * **MajorID**: 专业编号，外键（类型：INT）
   * **DepartmentID**: 院系编号，外键（类型：INT）
2. **班级表（Class）**
   * **ClassID**: 班级编号，主键（类型：INT）
   * **ClassName**: 班级名称（类型：VARCHAR(50)）
   * **DepartmentID**: 院系编号，外键（类型：INT）
3. **专业表（Major）**
   * **MajorID**: 专业编号，主键（类型：INT）
   * **MajorName**: 专业名称（类型：VARCHAR(50)）
   * **DepartmentID**: 院系编号，外键（类型：INT）
4. **院系表（Department）**
   * **DepartmentID**: 院系编号，主键（类型：INT）
   * **DepartmentName**: 院系名称（类型：VARCHAR(50)）
5. **课程表（Course）**
   * **CourseID**: 课程编号，主键（类型：INT）
   * **CourseName**: 课程名称（类型：VARCHAR(50)）
   * **Credits**: 学分（类型：INT）
   * **MajorID**: 专业编号，外键（类型：INT）
6. **成绩表（Grade）**
   * **GradeID**: 成绩编号，主键（类型：INT）
   * **StudentID**: 学生编号，外键（类型：INT）
   * **CourseID**: 课程编号，外键（类型：INT）
   * **Score**: 成绩分数（类型：DECIMAL(5, 2)）
7. **奖惩表（RewardPunishment）**
   * **RecordID**: 记录编号，主键（类型：INT）
   * **StudentID**: 学生编号，外键（类型：INT）
   * **Type**: 类型（奖或惩，类型：VARCHAR(10)）
   * **Reason**: 原因（类型：TEXT）
   * **Date**: 日期（类型：DATE）
8. **教职工表（Staff）**
   * **StaffID**: 教职工编号，主键（类型：INT）
   * **Name**: 教职工姓名（类型：VARCHAR(50)）
   * **DepartmentID**: 所属院系编号，外键（类型：INT）
9. **人员分配表（Allocation）**
   * **AllocationID**: 分配编号，主键（类型：INT）
   * **StaffID**: 教职工编号，外键（类型：INT）
   * **StudentID**: 学生编号，外键（类型：INT）
   * **CourseID**: 课程编号，外键（类型：INT）

二.概念结构设计

        数据库设计主要是进行数据库的逻辑设计，即将数据按一定的分类、分组系统和逻辑层次组织起来，是面向用户的。数据库设计时需要综合用户之间的存档数据和数据需求，分析各个数据之间的关系，按照DBMS提供的功能和描述工具，设计出规模适当、正确反映数据关系、数据冗余少、存取效率高、能满足多种查询要求的数据模型。简而言之，数据库就是信息的集合。关系数据库中包含了多个数据表的信息。因此可分一下步骤进行设计:

1.局部E-R图

1、专业：



2、班级



3、学院



4、职工



5、奖惩



6、课程



7、学生



全局er图：



3.逻辑结构设计

关系模式

1. Student(StudentID, Name, Gender, DOB, ClassID, MajorID, DepartmentID)

2. Class(ClassID, ClassName, DepartmentID)

3. Major(MajorID, MajorName, DepartmentID)

4. Department(DepartmentID, DepartmentName)

5. Course(CourseID, CourseName, Credits, MajorID)

6. Grade(GradeID, StudentID, CourseID, Score)

7. RewardPunishment(RecordID, StudentID, Type, Reason, Date)

8. Staff(StaffID, Name, DepartmentID)

9. Allocation(AllocationID, StaffID, StudentID, CourseID)

4.物理结构设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| StudentID | INT | - | 否 | 学生编号 (主键) |
| Name | VARCHAR | 50 | 否 | 学生姓名 |
| Gender | CHAR | 1 | 否 | 性别 ("男" 或 "女") |
| DOB | DATE | - | 是 | 出生日期 |
| ClassID | INT | - | 否 | 班级编号 (外键) |
| MajorID | INT | - | 否 | 专业编号 (外键) |
| DepartmentID | INT | - | 否 | 院系编号 (外键) |

2. 班级表 (Class Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| ClassID | INT | - | 否 | 班级编号 (主键) |
| ClassName | VARCHAR | 50 | 否 | 班级名称 |
| MajorID | INT | - | 否 | 专业编号 (外键) |

3. 专业表 (Major Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| MajorID | INT | - | 否 | 专业编号 (主键) |
| MajorName | VARCHAR | 50 | 否 | 专业名称 |
| DepartmentID | INT | - | 否 | 院系编号 (外键) |

4. 院系表 (Department Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| DepartmentID | INT | - | 否 | 院系编号 (主键) |
| DepartmentName | VARCHAR | 50 | 否 | 院系名称 |

5. 课程表 (Course Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| CourseID | INT | - | 否 | 课程编号 (主键) |
| CourseName | VARCHAR | 50 | 否 | 课程名称 |
| Credit | INT | - | 否 | 学分 |
| DepartmentID | INT | - | 否 | 开课院系编号 (外键) |

6. 成绩表 (Grades Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| GradeID | INT | - | 否 | 成绩编号 (主键) |
| StudentID | INT | - | 否 | 学生编号 (外键) |
| CourseID | INT | - | 否 | 课程编号 (外键) |
| Score | DECIMAL(5,2) | - | 否 | 分数 |

7. 奖惩表 (Rewards Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| RewardID | INT | - | 否 | 奖惩编号 (主键) |
| StudentID | INT | - | 否 | 学生编号 (外键) |
| Type | VARCHAR | 10 | 否 | 奖惩类型 ("奖" 或 "惩") |
| Description | TEXT | - | 是 | 奖惩描述 |
| Date | DATE | - | 否 | 奖惩日期 |

8. 教职工表 (Staff Table)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| StaffID | INT | - | 否 | 教职工编号 (主键) |
| Name | VARCHAR | 50 | 否 | 教职工姓名 |
| DepartmentID | INT | - | 否 | 所属院系编号 (外键) |
| Position | VARCHAR | 50 | 否 | 职位 |

三.数据库

1.数据库的实施

（1）创建数据库

CREATE DATABASE UniversityManagementSystem;

GO

USE UniversityManagementSystem;

GO

（2）创建表

**1. 学生表（Student）**

CREATE TABLE Student (

StudentID INT PRIMARY KEY IDENTITY(1,1),

Name VARCHAR(50) NOT NULL,

Gender VARCHAR(10) CHECK (Gender IN ('男', '女')),

DOB DATE NOT NULL,

ClassID INT NOT NULL,

MajorID INT NOT NULL,

DepartmentID INT NOT NULL

);

**2. 班级表（Class）**

CREATE TABLE Class (

ClassID INT PRIMARY KEY IDENTITY(1,1),

ClassName VARCHAR(50) NOT NULL,

DepartmentID INT NOT NULL

);

**3. 专业表（Major）**

CREATE TABLE Major (

MajorID INT PRIMARY KEY IDENTITY(1,1),

MajorName VARCHAR(50) NOT NULL,

DepartmentID INT NOT NULL

);

**4. 院系表（Department）**

CREATE TABLE Department (

DepartmentID INT PRIMARY KEY IDENTITY(1,1),

DepartmentName VARCHAR(50) NOT NULL

);

**5. 课程表（Course）**

CREATE TABLE Course (

CourseID INT PRIMARY KEY IDENTITY(1,1),

CourseName VARCHAR(50) NOT NULL,

Credits INT NOT NULL,

MajorID INT NOT NULL

);

**6. 成绩表（Grade）**

CREATE TABLE Grade (

GradeID INT PRIMARY KEY IDENTITY(1,1),

StudentID INT NOT NULL,

CourseID INT NOT NULL,

Score DECIMAL(5, 2) CHECK (Score >= 0 AND Score <= 100)

);

**7. 奖惩表（RewardPunishment）**

CREATE TABLE RewardPunishment (

RecordID INT PRIMARY KEY IDENTITY(1,1),

StudentID INT NOT NULL,

Type VARCHAR(10) CHECK (Type IN ('奖励', '惩罚')),

Reason TEXT NOT NULL,

Date DATE NOT NULL

);

**8. 教职工表（Staff）**

CREATE TABLE Staff (

StaffID INT PRIMARY KEY IDENTITY(1,1),

Name VARCHAR(50) NOT NULL,

DepartmentID INT NOT NULL

);

**9. 人员分配表（Allocation）**

CREATE TABLE Allocation (

AllocationID INT PRIMARY KEY IDENTITY(1,1),

StaffID INT NOT NULL,

StudentID INT NOT NULL,

CourseID INT NOT NULL

);

**1. 限制性别只能输入“男”或“女”**

在创建 Student 表时，已通过 CHECK 约束实现：

Gender VARCHAR(10) CHECK (Gender IN ('男', '女'))

**2. 创建视图查询学生信息**

以下是创建视图的 SQL，用于查询学生的学号、姓名、班级、专业和院系：

CREATE VIEW StudentInfo AS

SELECT

S.StudentID AS 学号,

S.Name AS 姓名,

C.ClassName AS 班级,

M.MajorName AS 专业,

D.DepartmentName AS 院系

FROM

Student S

JOIN

Class C ON S.ClassID = C.ClassID

JOIN

Major M ON S.MajorID = M.MajorID

JOIN

Department D ON S.DepartmentID = D.DepartmentID;

创建表间外键约束

-- Student表

ALTER TABLE Student

ADD FOREIGN KEY (ClassID) REFERENCES Class(ClassID),

FOREIGN KEY (MajorID) REFERENCES Major(MajorID),

FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID);

-- Class表

ALTER TABLE Class

ADD FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID);

-- Major表

ALTER TABLE Major

ADD FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID);

-- Course表

ALTER TABLE Course

ADD FOREIGN KEY (MajorID) REFERENCES Major(MajorID);

-- Grade表

ALTER TABLE Grade

ADD FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

FOREIGN KEY (CourseID) REFERENCES Course(CourseID);

-- RewardPunishment表

ALTER TABLE RewardPunishment

ADD FOREIGN KEY (StudentID) REFERENCES Student(StudentID);

-- Staff表

ALTER TABLE Staff

ADD FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID);

-- Allocation表

ALTER TABLE Allocation

ADD FOREIGN KEY (StaffID) REFERENCES Staff(StaffID),

FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

FOREIGN KEY (CourseID) REFERENCES Course(CourseID);

2.数据初始化

输入插入

**1. 插入院系表数据（Department）**

INSERT INTO Department (DepartmentName)

VALUES ('计算机科学与技术学院'),

('电子信息工程学院'),

('经济管理学院');

**2. 插入专业表数据（Major）**

INSERT INTO Major (MajorName, DepartmentID)

VALUES ('软件工程', 1),

('通信工程', 2),

('市场营销', 3);

**3. 插入班级表数据（Class）**

INSERT INTO Class (ClassName, DepartmentID)

VALUES ('软件工程2021级1班', 1),

('通信工程2021级1班', 2),

('市场营销2021级1班', 3);

**4. 插入学生表数据（Student）**

INSERT INTO Student (Name, Gender, DOB, ClassID, MajorID, DepartmentID)

VALUES ('张三', '男', '2001-05-20', 1, 1, 1),

('李四', '女', '2000-11-15', 2, 2, 2),

('王五', '男', '2002-03-10', 3, 3, 3);

**5. 插入课程表数据（Course）**

INSERT INTO Course (CourseName, Credits, MajorID)

VALUES ('数据库系统', 4, 1),

('信号与系统', 3, 2),

('市场营销学', 3, 3);

**6. 插入成绩表数据（Grade）**

INSERT INTO Grade (StudentID, CourseID, Score)

VALUES (1, 1, 95.5),

(2, 2, 88.0),

(3, 3, 76.5);

**7. 插入奖惩表数据（RewardPunishment）**

INSERT INTO RewardPunishment (StudentID, Type, Reason, Date)

VALUES (1, '奖励', '获得国家奖学金', '2024-05-01'),

(2, '惩罚', '违反宿舍管理规定', '2024-06-15'),

(3, '奖励', '优秀学生干部', '2024-07-01');

**8. 插入教职工表数据（Staff）**

INSERT INTO Staff (Name, DepartmentID)

VALUES ('张老师', 1),

('李老师', 2),

('王老师', 3);

**9. 插入人员分配表数据（Allocation）**

INSERT INTO Allocation (StaffID, StudentID, CourseID)

VALUES (1, 1, 1),

(2, 2, 2),

(3, 3, 3);

3.可行性分析

**1. 技术可行性**

本系统基于SQL Server数据库技术开发，SQL Server作为一款成熟的关系型数据库管理系统，具备以下技术优势：

* **高效的数据存储与管理**：支持大规模数据处理，适合高校学籍信息的集中管理。
* **强大的查询能力**：SQL语言提供高效的查询机制，能够快速满足学籍管理中多维度查询的需求。
* **数据完整性与安全性保障**：支持外键、数据约束以及权限管理，确保数据的一致性和访问安全性。
* **易于维护与扩展**：系统设计符合模块化原则，便于后续功能的扩展和维护。

高校现有的技术设备和网络条件能够支持系统的开发与运行，因此从技术角度看，本系统的实现是可行的。

**2. 经济可行性**

系统开发和运行的成本主要包括以下几方面：

* **开发成本**：
  + 人力成本：开发团队的工资支出。
  + 软件成本：SQL Server可选择免费版本（如Express版）或已有高校许可版本，无需额外购置。
* **运行成本**：
  + 服务器及硬件设备：大多数高校已具备相关硬件资源，无需新增设备。
  + 系统维护：维护成本较低，主要为定期更新和数据备份。

考虑到学籍管理信息化的长期效益（如降低管理成本、提升效率、减少错误等），本系统的投入产出比是较高的，具有良好的经济可行性。

**3. 操作可行性**

本系统面向高校的学籍管理部门、教职工和学生，设计时注重操作的简便性和用户体验：

* **管理员**
  + 提供图形化界面，便于学籍信息的集中管理和规则配置。
* **教职工**
  + 简化操作流程，易于上手，便于录入和查询学生成绩、课程信息等。
* **学生**
  + 提供直观的界面，学生能够快速查询个人信息、成绩和奖惩记录。

经过前期调研，大多数用户对系统的功能需求和设计目标表示认可，系统操作难度较低，具备良好的操作可行性。

 4.系统功能描述

1. **学生信息管理功能**
   * 支持学生基本信息（姓名、学号、性别、班级、专业等）的录入、修改、删除和查询。
   * 提供多条件的快速检索功能。
2. **课程与成绩管理功能**
   * 课程信息的录入、修改和查询。
   * 学生成绩的录入、修改、查询及统计分析（如平均成绩、最高分等）。
3. **奖惩信息管理功能**
   * 实现学生奖惩记录的添加、修改、删除和查询。
   * 提供奖惩信息的分类统计（如奖励占比、惩罚原因统计）。
4. **教职工与人员分配管理功能**
   * 记录教职工基本信息，并实现教职工与学生、课程之间的关联管理。
   * 支持按院系或专业查询教职工分配信息。
5. **系统约束与规则管理功能**
   * 实现字段规则的设定，如性别字段只能输入“男”或“女”。
   * 确保所有外键关联的参照完整性。
6. **数据查询与报表生成功能**
   * 提供自定义查询功能，用于快速检索学生、课程、成绩等信息。
   * 自动生成学籍、课程、成绩等相关报表，支持导出为Excel或PDF格式。

5.数据库实施与维护

1. **备份与恢复**
   * 定期执行数据库备份，确保数据安全。
   * 提供数据恢复机制，防止因故障导致的数据丢失。
2. **性能优化**
   * 监控数据库性能，针对查询频繁的表添加索引。
   * 优化SQL查询语句，减少查询延迟。
3. **安全管理**
   * 设置用户权限，确保数据的访问安全性。
   * 记录用户操作日志，防止数据的非法篡改。
4. **系统升级与扩展**
   * 根据实际需求调整数据库表结构。
   * 在系统扩展时，确保与现有数据的兼容性