# 《数据库系统》上机报告：

# 学籍管理数据库系统

姓名： 赵宇阳

学号： 22009200439

指导老师： 赵亮 马智

1. **需求分析**
2. 系统数据结构
3. 专业（Major）
4. 班级（Class）
5. 学生（Student）
6. 课程（Course）
7. 教学计划（Curriculum）
8. 成绩（Grades）
9. 教师（Teacher）
10. 授课安排（Teaching）
11. 系统功能
12. 专业管理

添加、修改、删除专业信息。

查询专业列表及其详细信息。

1. 班级管理

添加、修改、删除班级信息。

查询班级列表及其详细信息，包括所属专业和入学年份。

1. 学生管理

添加、修改、删除学生信息。

查询学生列表及其详细信息，包括所属班级、性别和出生日期。

学生学籍状态管理，包括正常、留级、退学等。

1. 课程管理

添加、修改、删除课程信息。

查询课程列表及其详细信息，包括课程名称和学分。

1. 教学计划管理

添加、修改、删除教学计划信息。

查询教学计划列表及其详细信息，包括所属专业、课程、授课学期和课程性质。

1. 成绩管理

录入、修改、删除学生成绩。

查询学生成绩列表及其详细信息，包括学生、课程和分数。

计算学生的总学分、必修课和选修课的不及格学分，并根据规则判断是否需要开除。

1. 教师管理

添加、修改、删除教师信息。

查询教师列表及其详细信息，包括教师姓名。

1. 课程安排管理

添加、修改、删除授课安排信息。

查询授课安排列表及其详细信息，包括教师、班级和课程。

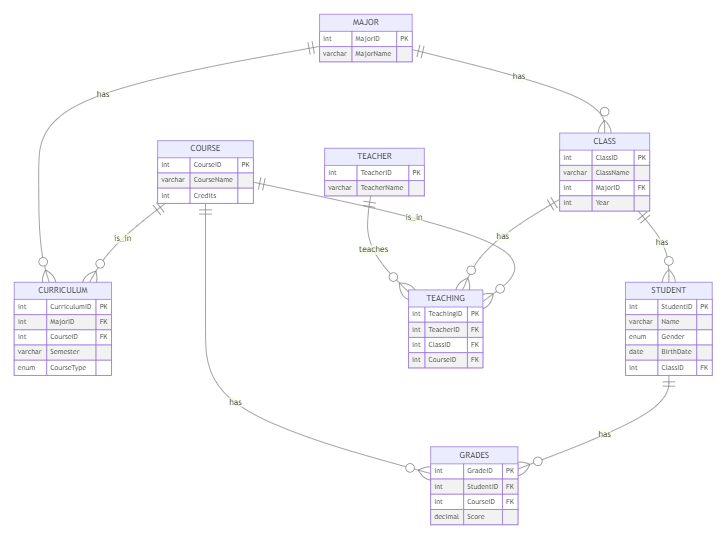
1. 系统设置

用户权限管理。

数据备份与恢复。

1. **概念结构设计**

E-R图设计：



1. **逻辑结构设计**
2. 实体

(1) 专业（Major）

MajorID：专业编号，主键。

MajorName：专业名称。

(2) 班级（Class）

ClassID：班级编号，主键。

ClassName：班级名称。

MajorID：所属专业的编号，外键。

Year：入学年份。

(3) 学生（Student）

StudentID：学生编号，主键。

Name：学生姓名。

Gender：性别。

BirthDate：出生日期。

ClassID：所属班级的编号，外键。

(4) 课程（Course）

CourseID：课程编号，主键。

CourseName：课程名称。

Credits：学分。

(5) 教学计划（Curriculum）

CurriculumID：教学计划编号，主键。

MajorID：所属专业的编号，外键。

CourseID：课程编号，外键。

Semester：授课学期。

CourseType：课程性质（必修或选修）。

(6) 成绩（Grades）

GradeID：成绩编号，主键。

StudentID：学生编号，外键。

CourseID：课程编号，外键。

Score：分数。

(7) 教师（Teacher）

TeacherID：教师编号，主键。

TeacherName：教师姓名。

(8) 授课安排（Teaching）

TeachingID：授课安排编号，主键。

TeacherID：教师编号，外键。

ClassID：班级编号，外键。

CourseID：课程编号，外键。

1. 关系
2. Class-Major（班级与专业的关联）

一个班级属于一个专业（1:N）

1. Curriculum-Major（课程计划与专业的关联）

一个专业有多个课程计划（1:N）

1. Curriculum-Course（课程计划与课程的关联）

一个课程计划包含一个课程（1:1）

1. Grades-Student（成绩与学生的关联）

一个学生成绩对应一个学生（1:1）

1. Grades-Course（成绩与课程的关联）

一个学生成绩对应一个课程（1:1）

1. Student-Class（学生与班级的关联）

一个学生属于一个班级（1:1）

1. Teaching-Teacher（教学安排与教师的关联）

一个教学安排对应一个教师（1:1）

1. Teaching-Class（教学安排与班级的关联）

一个教学安排对应一个班级（1:1）

1. Teaching-Course（教学安排与课程的关联）

一个教学安排对应一个课程（1:1）

1. **功能实现**
2. 选用的DBMS：

DBMS: MySQL (版本 9.0.1)

区分大小写: 普通形式=lower，分隔形式=lower

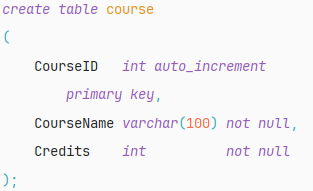
驱动程序: MySQL Connector/J (版本 mysql-connector-j-8.2.0

(Revision: 06a1f724497fd81c6a659131fda822c9e5085b6c)，JDBC4.2)

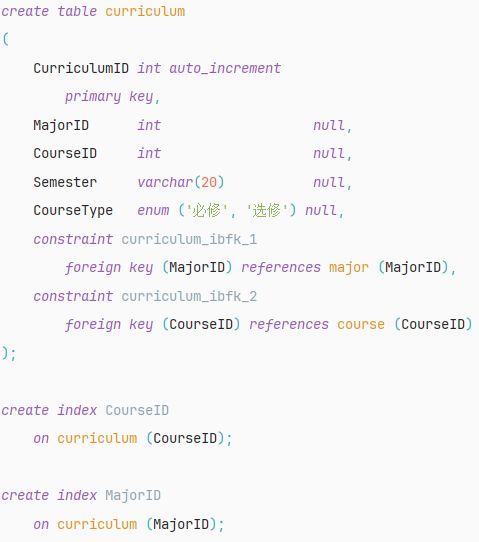
1. 建表语句：
2. Class



1. Course



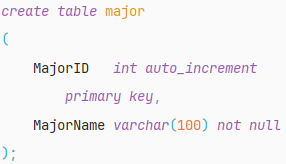
1. Curriculum



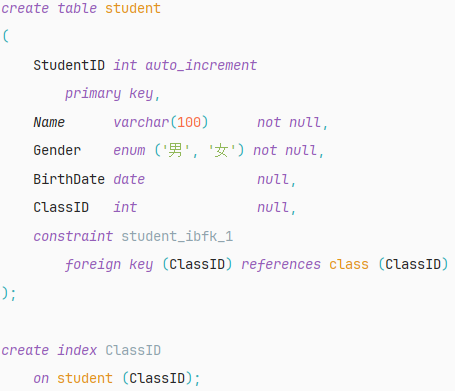
1. Grades



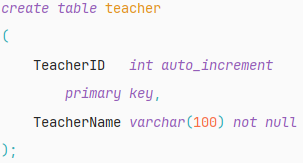
1. Major



1. Student



1. Teacher



1. Teaching



1. 针对问题编写的SQL以及对应的运行结果
2. 录入一位学生，应包含学号、姓名、性别、出生年月、班级等信息

SQL语句:

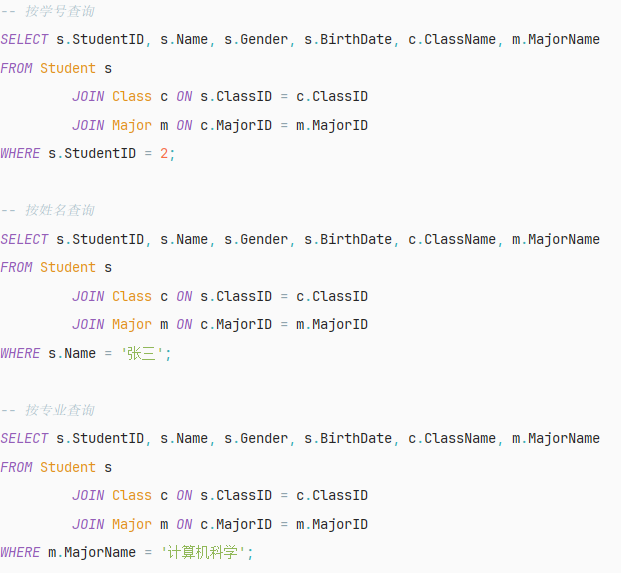


执行结果：



1. 按学号、姓名、专业三种方式查询学生基本信息

SQL语句：



执行结果：

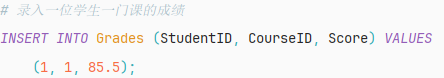






1. 录入一位学生一门课的成绩

SQL语句：

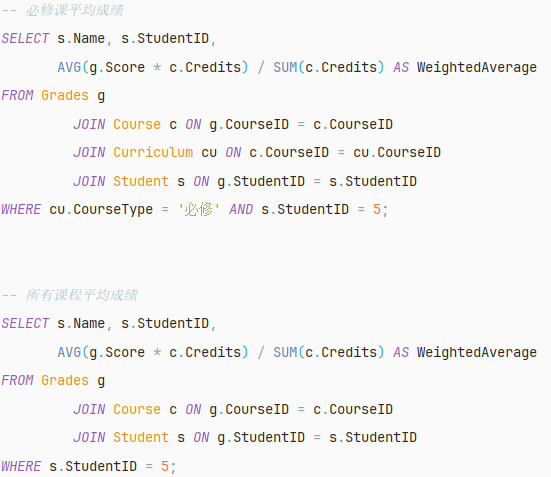


执行结果：



1. 查询一位学生所修的课程、性质（必修或选修）、学期、学分及成绩；查询他的必修课平均成绩、所有课程平均成绩（平均成绩应按学分加权）

SQL语句：

  
执行结果：

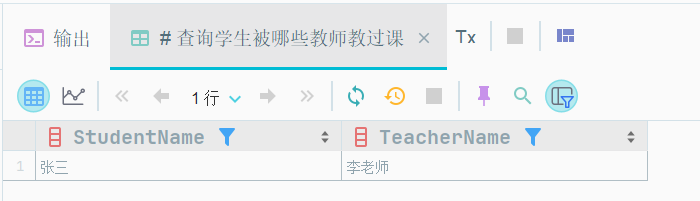




1. 查询一位学生被哪些教师教过课

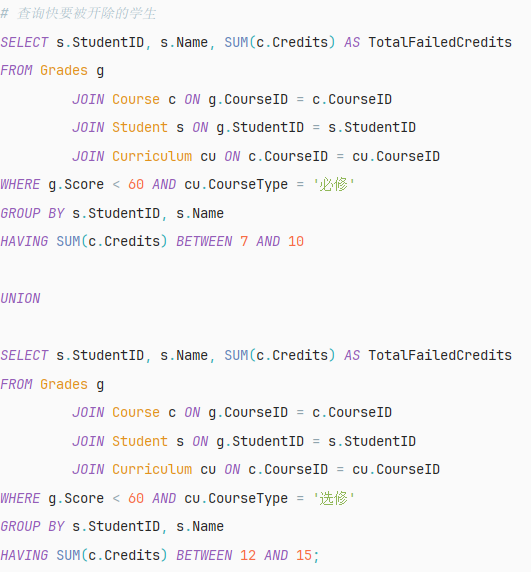
SQL语句：

  
执行结果：

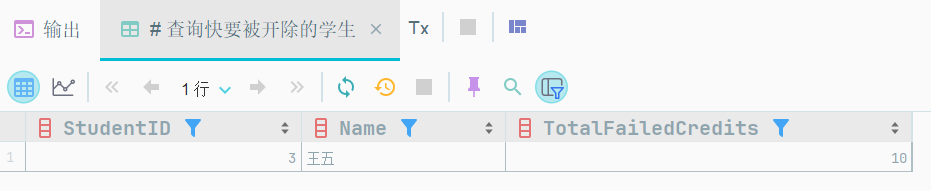


1. 查询快要被开除的学生（距被开除差3学分之内）

SQL语句：



执行结果：



1. **遇到的主要问题及解决方法**

问题描述：

在编写查询快要被开除的学生的SQL语句过程中，最初尝试直接通过Grades表与Course表、Student表以及Curriculum表进行连接，并筛选出成绩低于60分且课程类型为“必修”或“选修”的记录。然而，在计算学生的累计失败学分时发现结果不准确，具体表现为某些学生的累计失败学分数值偏低或偏高。经过初步排查，怀疑是由于某些学生在多个学期选修了同一门课程，导致成绩记录重复计算，进而影响了最终的统计结果。

解决方法：

为了解决上述问题，采取了以下措施：

1. **去重处理**：在计算每个学生的累计失败学分之前，先对成绩记录进行去重处理，确保每门课程的成绩只被计算一次。
2. **分组统计**：分别针对“必修”和“选修”课程进行分组统计，确保不同类型的课程按照不同的标准进行累计。
3. **条件筛选**：在HAVING子句中增加了具体的条件限制，确保只有累计失败学分在指定范围内的学生才会被选中。
4. **总结**

上机主要围绕学籍管理数据库系统的开发与实现展开。通过需求分析、概念结构设计、逻辑结构设计以及功能实现等多个环节，我们成功构建了一个功能完备、操作便捷的学籍管理系统。以下是本次实验的主要成果和收获：

需求分析：明确了系统需要管理的专业、班级、学生、课程、教学计划、成绩、教师和授课安排等八大模块，为后续的设计和实现奠定了基础。

概念结构设计：通过E-R图的设计，清晰地展示了各个实体之间的关系，为数据库的逻辑结构设计提供了直观的参考。

逻辑结构设计：定义了各个实体的属性及其关系，确保了数据库的规范化和一致性。特别是通过外键约束，保证了数据的完整性和准确性。

功能实现：选择了MySQL作为数据库管理系统，并编写了相应的建表语句和SQL查询语句。实现了学生信息的录入、查询、成绩管理、教师管理等功能，并解决了在查询快要被开除的学生时出现的数据重复计算问题。

问题解决：在遇到数据重复计算的问题时，通过去重处理、分组统计和条件筛选等方法，有效解决了问题，确保了查询结果的准确性。

通过本次实验，不仅加深了对数据库系统理论知识的理解，还提升了实际操作能力和问题解决能力。未来将进一步优化系统性能，增加更多的功能模块，以满足更复杂的应用需求。