《数据库系统原理》大作业
系统设计报告

题目名称：教务管理系统

学号及姓名：78066009 苏嘉娴

78066014 邓奇恩

78066015 谢宇隆

2020 年 12 月 5 日

组内同学承担任务说明

	负责学生
子任务 1：系统功能设计与数据库设计	邓奇恩 谢宇隆 苏嘉娴
子任务 2：系统服务器端开发	谢宇隆 邓奇恩
子任务 3：系统客户端开发	苏嘉娴 邓奇恩

一. 需求分析

1.1 需求描述

该作业做的是一款小型的教务管理系统。用户可在这款管理系统中以相应的用户类型来使用。每个用户使用前都需要输入自己的用户号和密码，如密码正确就可登入系统，系统会自动辨别用户的类型。所有用户登入之后都可自由更改密码与登出。
所有用户都有权限进行的操作

a) 退出登录
b) 更改密码

系统中用户类型一共有三类：

- 学生

a) 可查看自己的个人信息。
b) 可查看自身已选的所有课程，如列表为空则该学生未选课。
c) 在已选课程列表中可做退课操作。
d) 可查看目前可选的所有课程，如列表为空则该表示没有课程开放或所有课程都已选择。
e) 在可选课程列表中可选课。

- 教师

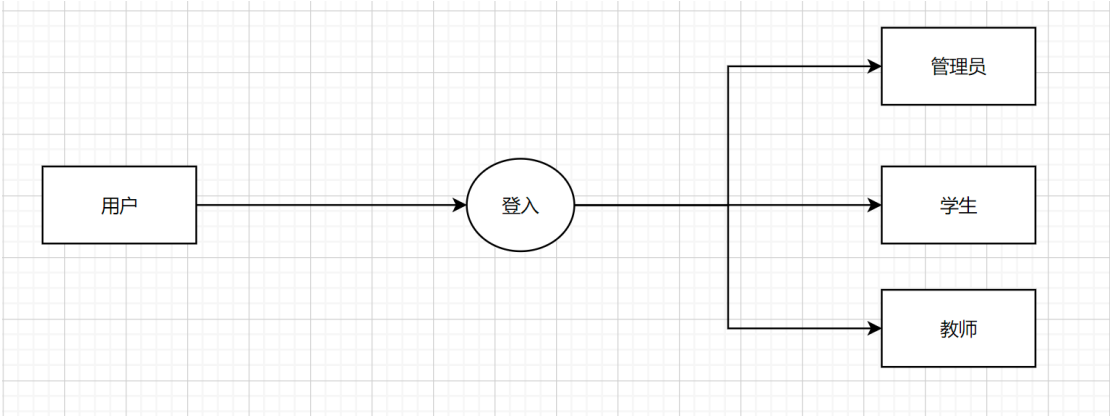
a) 可查看自己的个人信息。
b) 可查看自己负责教学的课程。
c) 可查看选择自己负责教学的课程的学生列表。
d) 可以给选择自己负责教学的课的学生打分数。

- 管理员

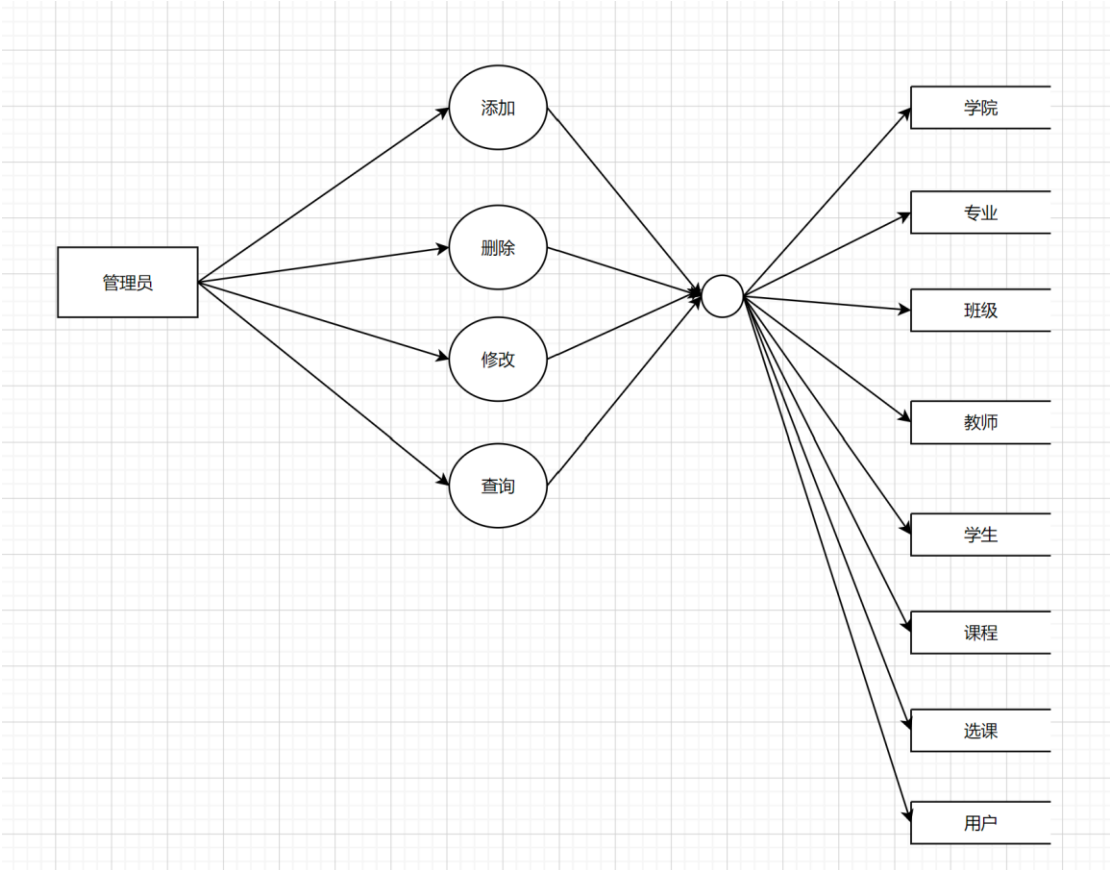
可对如下所有信息做查看、增加、删除与更改操作。
a) 用户的信息
b) 课程
c) 选课情况
d) 学院的信息
e) 专业的信息
f) 班级信息

1.2 数据流图

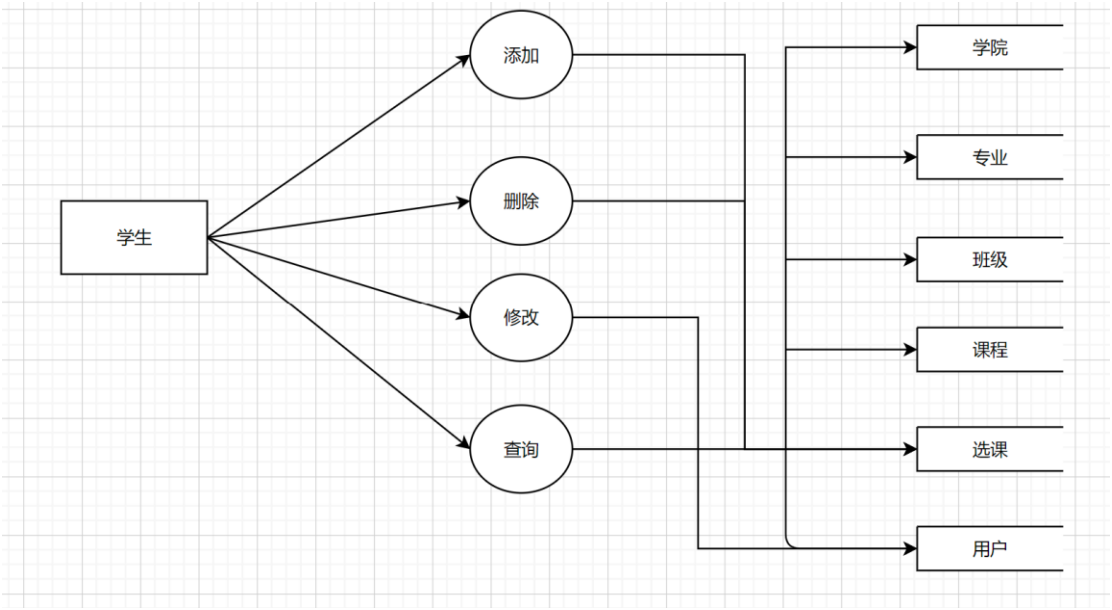
首先设定整个系统必须有一个管理员，既开发人员，之后无论是任何类型用户都需要由管理员添加，经过管理员添加之后其他用户才能以已添加的用户的用户名和密码登入。



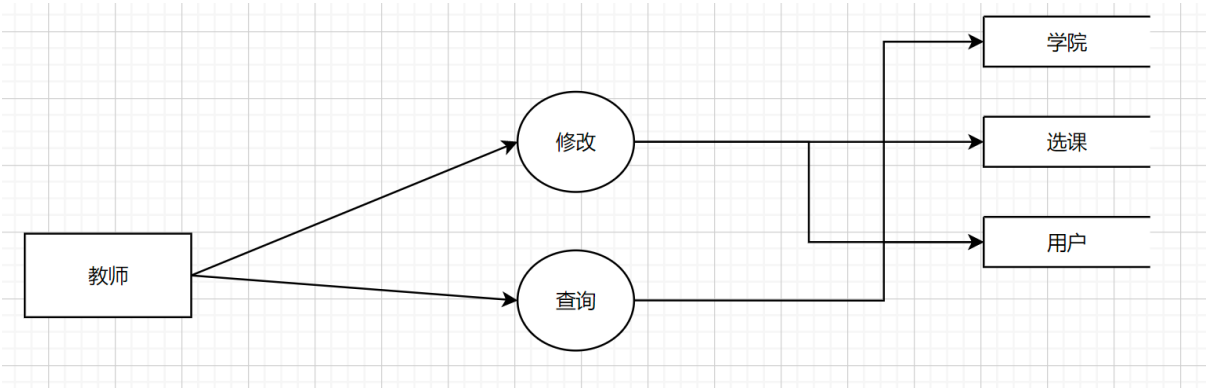
有了用户之后就分别说明每个用户类型的设定，管理员是整个系统中权限最高的用户类型，管理员可对任意表进行添加、删除、修改和查询。



学生可查询自己的资料，如所在学院、专业、班级等。而学生涉及的添加和删除操作是在选课表中，学生的选课和退课既对应选课表的添加和删除，除此之外学生还可更改自身用户密码，既为对用户的修改操作。



教师类型用户无任何添加和删除操作，教师类型用户可查询自身资料如所在学院和用户信息，还有更改选课表中学生所选科目的分数以及更改自身用户密码。



1.3 数据元素表

数据名	含义	数据类型	允许空值	码	其他
class_id	班号	varchar（100）	否	主码	NULL
year	年级	varchar（100）	是	-	NULL
major_id	专业号	varchar（100）	是	外码	NULL

表 1 班级 class

数据名	含义	数据类型	允许空值	码	其他
course_id	课程号	varchar（100）	否	主码	NULL
name	课程名	varchar（100）	是	-	NULL
teacher_id	教师号	varchar（100）	否	主码	NULL

department_id	学院号	varchar（100）	是	外码	NULL

表 2 **课程** course

数据名	含义	数据类型	允许空值	码	其他
course_id	课程号	varchar（100）	否	主码	NULL
student_id	学号	varchar（100）	否	主码	NULL
grade	成绩	varchar（100）	是	-	NULL

表 3 **选修** courseselection

数据名	含义	数据类型	允许空值	码	其他
department_id	学院号	varchar（100）	否	主码	NULL
name	学院名	varchar（100）	是	-	NULL

表 4 **学院** department

数据名	含义	数据类型	允许空值	码	其他
major_id	专业号	varchar（100）	否	主码	NULL
name	专业名	varchar（100）	是	-	NULL
department_id	学院号	varchar（100）	是	外码	NULL

表 5 **专业** major

数据名	含义	数据类型	允许空值	码	其他
student_id	学号	varchar（100）	否	主码	NULL
name	学生姓名	varchar（100）	是	-	NULL
class_id	班级号	varchar（100）	是	外码	NULL
course_cnt	已选课数目	varchar（100）	是	-	NULL

表 6 **学生** student

数据名	含义	数据类型	允许空值	码	其他
teacher_id	教师号	varchar（100）	否	主码	NULL
name	教师姓名	varchar（100）	是	-	NULL
department_id	学院号	varchar（100）	是	外码	NULL

表 7 **教师** teacher

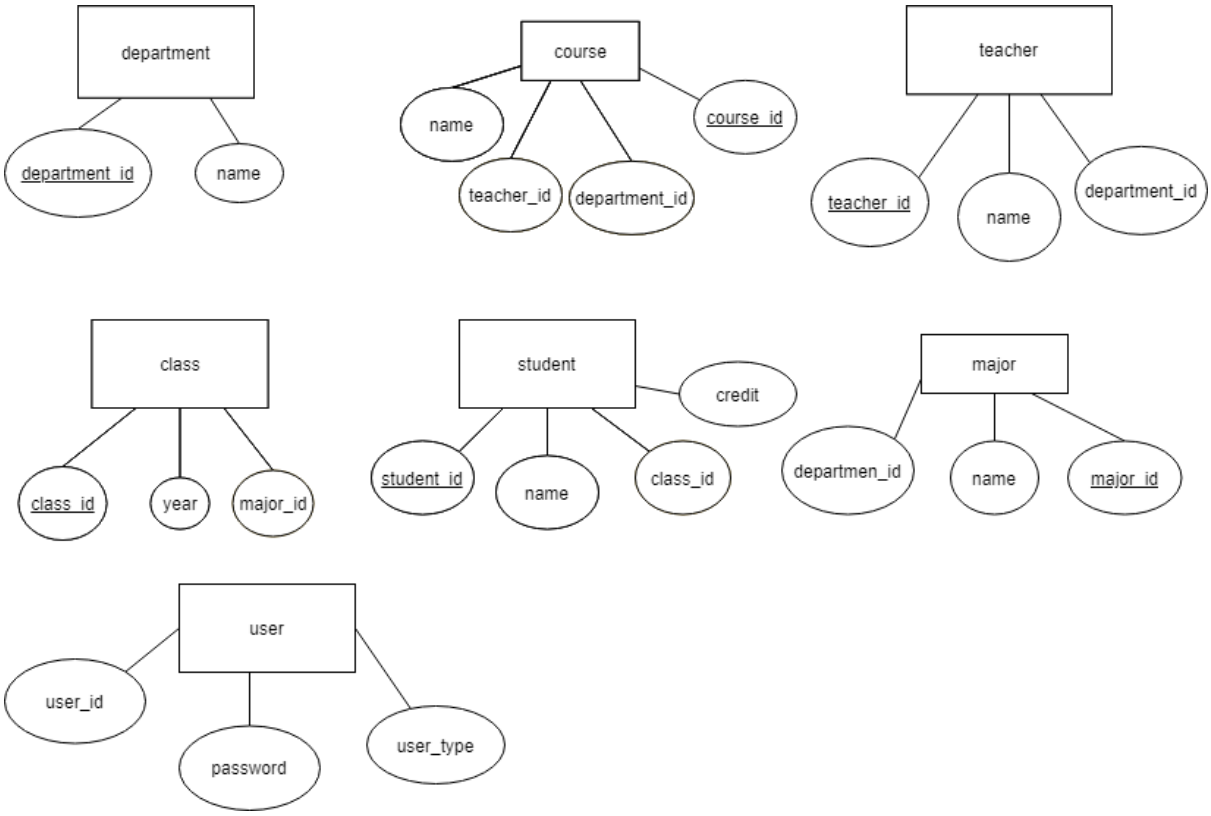
数据名	含义	数据类型	允许空值	码	其他
user_id	用户号	varchar（100）	否	主码	NULL
password	用户密码	varchar（100）	是	-	NULL
user_type	用户类型	varchar（100）	是	-	NULL

表 8 **用户** user

二. 数据库概念模式设计

2.1 系统初步 E-R 图

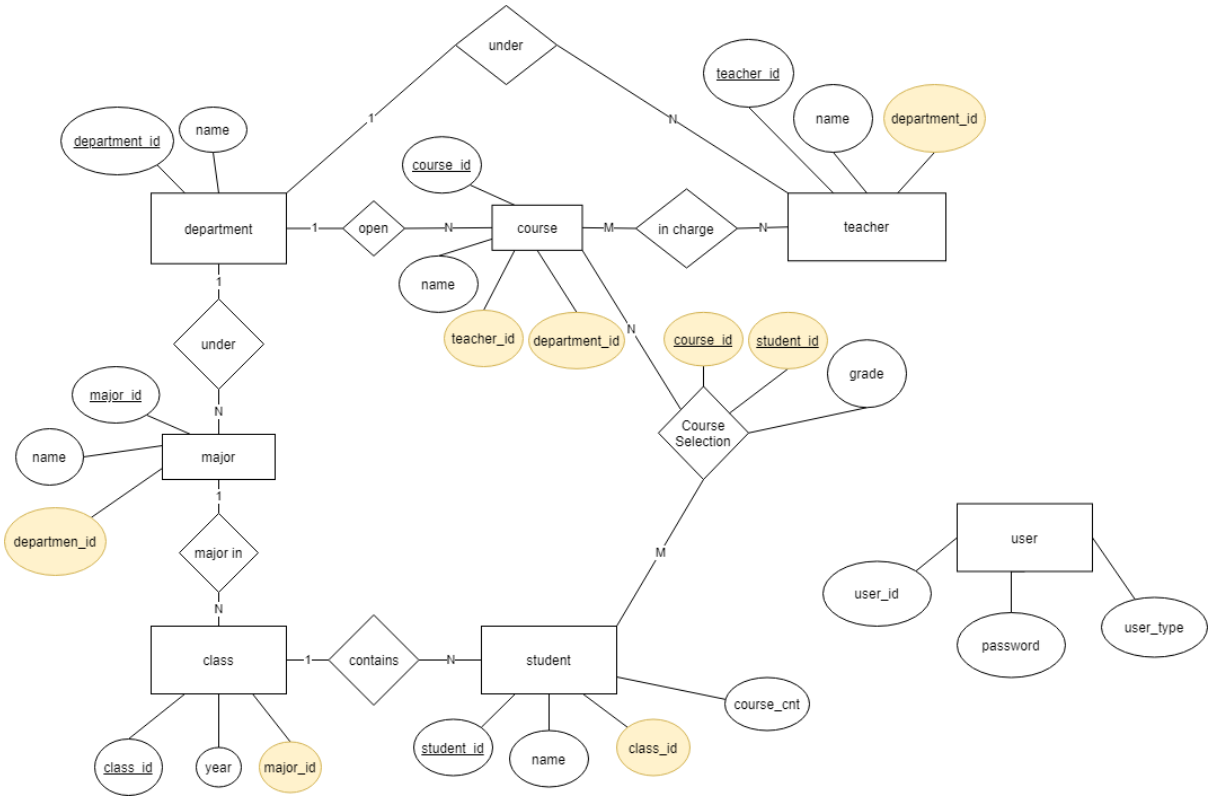
根据需求分析、数据流图和数据元素表得到下图所示的 6 个实体分别是 department、course、teacher、major、class、student。



根据需求描述，可以得出实体间的联系如下：

1. 1 个学院可以拥有多个老师，1 个老师只能隶属一个学院。
2. 1 个学院可以开多门课程，但 1 门课程只能由 1 个学院开课。
3. 1 个学院可以开多个专业，1 个专业只能由 1 个学院开。
4. 1 个专业可以拥有多个班级，1 个班级只属于一个专业。
5. 1 个班级能拥有多个学生，1 个学生只能存在于 1 个班级。
6. 1 个教师可以负责多个课程，1 门课程可由多位教师同时授课。
7. 1 个学生可选修多门课程，1 门课程可由多名学生选修。

2.2 系统基本 E-R 图



三. 数据库逻辑模式设计

1. 数据库关系模式

1. department (department_id,name)
2. major (major_id , name, departent_id)
3. class (class_id, year ,major_id)
4. student (student_id, name, class_id, course_cnt)
5. course (course_id, name, teacher_id, deparment_id)
6. teacher(teacher_id, name, department_id)
7. course selection(course_id, student_id, grade)
8. user(user_id, password, user_type)

2. 关系模式范式等级的判定与规范化

1. department (department_id,name)
   函数依赖：department_id→name
   关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。
2. major (major_id , name, departent_id)
   函数依赖：major_id→name , major_id→department_id

关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。

3. class (class_id, year ,major_id)
   函数依赖：class_id→major_id , class_id→year
   关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。
4. student (student_id, name, class_id, course_cnt)
   函数依赖：student_id→name, student_id→class_id , student_id→course_cnt
   关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。
5. course (course_id, name, teacher_id, deparment_id)
   函数依赖：course_id→name, course_id→teacher_id , course_id→department_id
   关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。
6. teacher(teacher_id, name, department_id)
   函数依赖：teacher_id→name , teacher_id→department_id
   关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。
7. course selection(course_id, student_id, grade)
   函数依赖：(course_id,student_id)→grade
   关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于3NF。
8. user(user_id, password, user_type)
   函数依赖： user_id→password, user_id →user_type 关系模式里的每一个非主属性即不传递依赖于码，也不部分依赖于码，因此关系属于 3NF。