设计文档

# 数据库系统名称

学生学籍管理系统（校园教务管理系统）

# 系统简介

本系统采用模块化设计方式，在学生信息管理、教师信息管理、课程管理、选课管理、成绩管理、课表管理等模块中实现新建、删除、修改、查询等功能。

用户分为学生、教师、管理员三种，分别具有不同权限，均不能跨越权限进行操作。

系统采用C/S模式设计，在服务器中运行SQL Server，在客户机中登陆进行各种操作。

# 运行环境及使用的开发工具

开发工具：

服务器端：Microsoft Visual Studio(2012)

客户机端：Microsoft SQL Server(2012)

概念图绘制：Microsoft Visio(2013)

开发语言：

服务器端：SQL

客户机端：VC

运行平台：

服务器端要求能够正常运行Microsoft SQL Server 2003（或以上）

客户机端要求安装Microsoft .NET Framework 3.5（或以上）

无其他特设软硬件要求

注：目前的C/S连接配置尚比较繁琐，会在之后改进

# 概要设计说明

用户权限管理模块：实现学生、教师、管理员的权限管理，各用户利用自身ID和密码登陆系统。

学生、教师信息管理模块：由管理员初始化学生、教师的最基本信息（姓名、编号），由学生、教师本人进行其它信息的更新。

课程管理模块：管理员与教师协商确定可选课程，学生进行选课和退选。

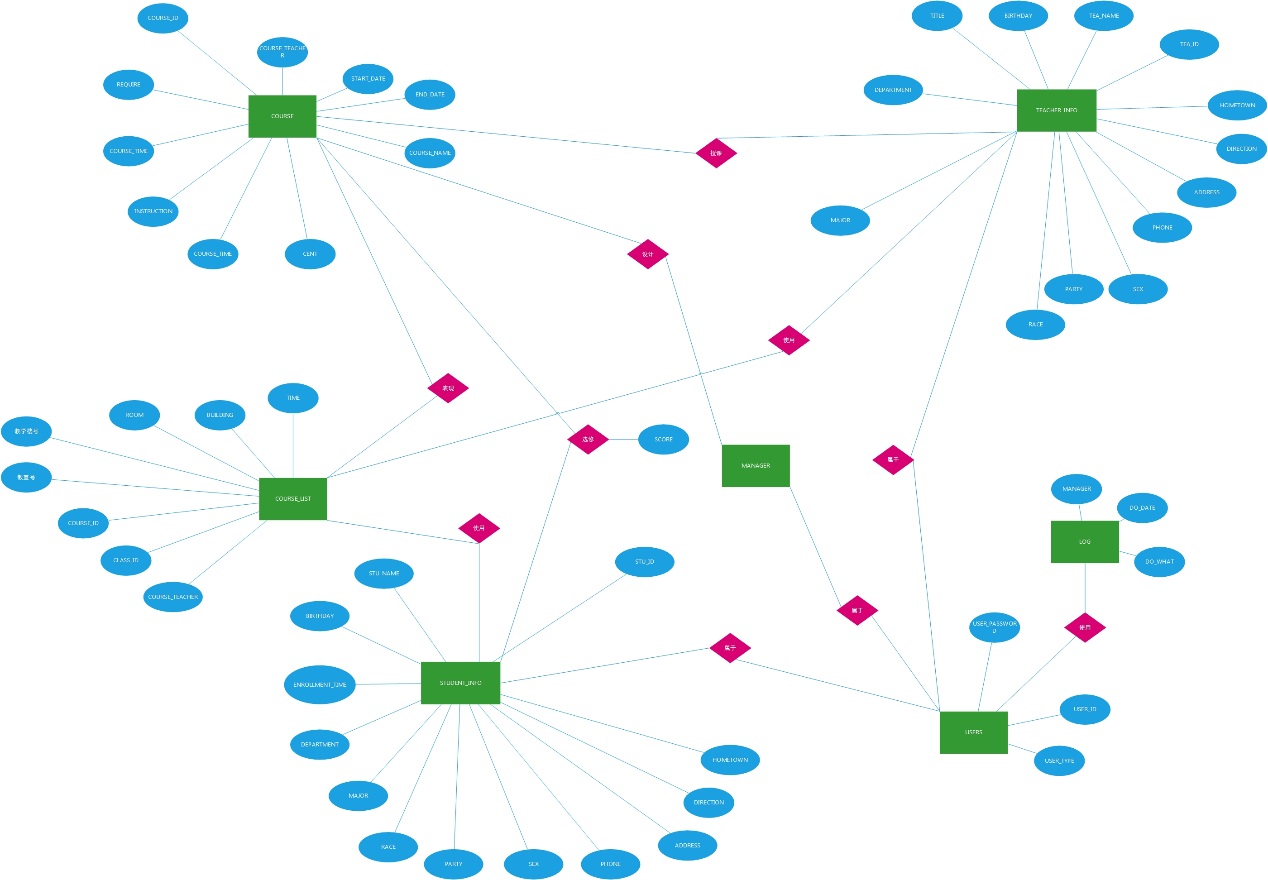
课表管理模块：管理员根据课程内容、学生选课情况确定上课具体信息（上课地点等），学生可进行查看。

成绩管理模块：教师负责登记课程成绩，学生只能查看。

日志管理模块：对某时刻某用户进行的某操作进行记录，以便日后查看（目前只是服务器端支持，客户端不具该功能）

# 详细设计说明

## 概念结构设计：



## 逻辑结构设计：

USERS(USER\_ID,USER\_PASSWORD,USER\_TYPE)

STUDENT\_INFO(STU\_ID,STU\_NAME,ENROLLMENT,DEPARTMENT,MAJOR,SEX,BIRTHDAY,HOMETOWN,ADDRESS,PHONE,RACE,PARTY,DIRECTIN)

TEACHER\_INFO(TEA\_ID,TEA\_NAME,TITLE,DEPARTMENT,MAJOR,SEX,BIRTHDAY,HOMETOWN,ADDRESS,PHONE,RACE,PARTY,DIRECTIN)

COURSE(COURSE\_ID,COURSE\_NAME,COURSE\_TIME,START\_TIME,END\_TIME,CENT,COURSE\_TEACHER,REQUIRE,INSTRUCTION)

COURSE\_SCORE(STU\_ID,COURSE\_ID,COURSE\_NAME,SCORE)

COURSE\_LIST(COURSE\_ID,COURSE\_NAME,CLASS\_ID,COURSE\_TEACHER,WEEKDAY,DAYHOUR,BUILDING,ROOM)

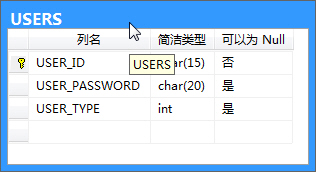
LOG(MANAGER,DO\_DATE,DO\_WHAT)

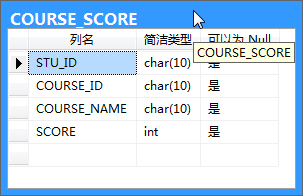
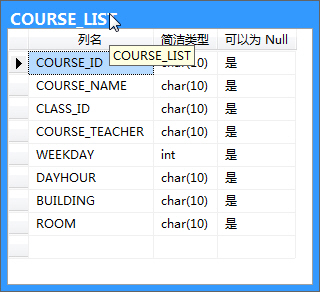
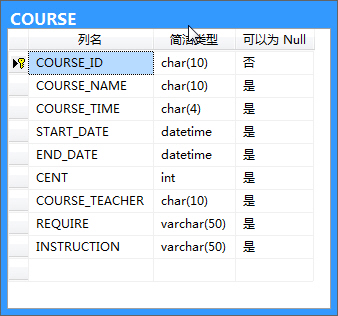
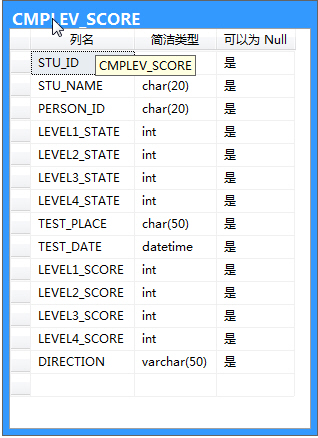
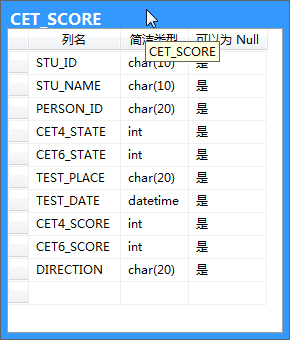
注：

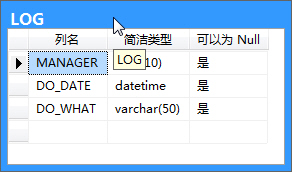
COURSE\_LIST模型中若COURSE\_NAME相同，CLASS\_ID不同，则COURSE\_ID不同，因而也满足第二、三范式

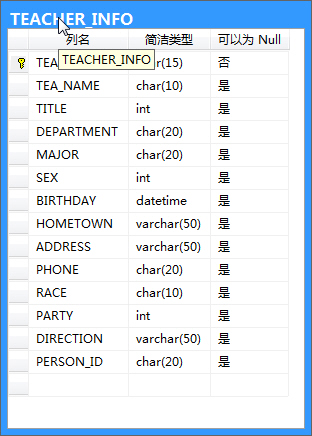
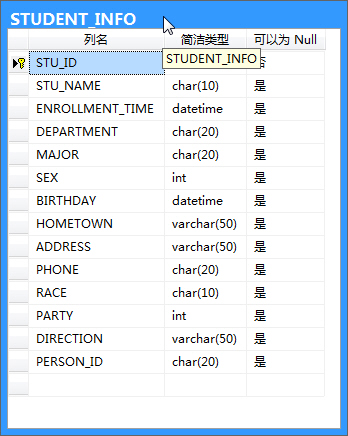
COURSE\_SCORE模型不符合第二范式，COURSE\_NAME仅依赖于COURSE\_ID，但SCORE既依赖于STU\_ID又依赖于COURSE\_ID

## 数据库所包含表名及各数据库表结构:









## 根据客户需求设计相关查询及其SQL语句：

exec sp\_mark

exec sp\_mark @coursename='宏观经济学'

exec sp\_order

exec sp\_order @coursename='宏观经济学'

# 总结：

通过这次校务管理系统的设计我对于数据库理论的理解更加深入了,之前一些课程(特备是E-R图的绘制)因为种种原因没有跟上,在设计数据库是我有查阅了不少资料,可以说基本掌握了。另外通过客户端图形界面的制作我对Visual C++更加熟悉了，可以说这是一个意外的收获。虽然整个设计开发过程话费了不少时间和经历，但真正获得的远比这些要多，所以还是很感激能有这样一个机会的。当然到目前为止数据库还是有一些缺陷，由于期末考试时间挺紧的所以也没有太多时间去优化了，但我在之后的时间会继续改进，因为真如果让我抛弃这样一个我费了这么多时间的东西还真有一些舍不得。现在暂时把一些不足列出来，以备以后的查阅。

* 完善约束条件（如：身份证、学号、教师编号、课程号、入学年份等的验证）
* 改进C/S连接方式，简化连接过程
* 将四六级管理、计算机等级考试管理等模块加入用户端的图形界面
* 增加自动识别功能（如： 根据USER\_ID自动识别类型，不用用户自己选择）
* 进一步优化逻辑模式，在保证效率的同时减少冗余度
* 将课表管理和课程管理合并
* 增加备份和并发控制机制
* 完备信息记录，如学生等的邮箱，课程的考试信息