《数据库系统概论》实验指导书

华 中 科 技 大 学

网络空间安全学院

2023 年 11 月

实验前的准备工作

实验的操作系统环境为 Windows 或者 Linux。

数据库使用 Microsoft SQL Server/MySQL 免费开源版/SQLite/openGauss/人大金仓 KingbaseES/达梦数据库 DM 等均可。

实验前请自己下载相关的数据库管理系统(DBMS)，可以是最新版或者是任意可运行的版本。

请首先熟练掌握数据库管理系统的安装过程。Microsoft SQLServer 安装时建议选择混合模式的身份验证方式。记住系统管理员的密码。

针对 Microsoft SQLServer，要熟练掌握服务管理平台和查询分析器的界面操作。针对 MySQL 需要同时安装其可视化客户端管理工具，例如 navicat 或者 MSQL-Front 等，并熟练掌握其操作。需要先熟练掌握数据库创建、创建表、创建用户、使用不同的用户进行登录等基本操作。

最后一个实验需要进行数据库应用开发，需要先熟悉一门编程语言，例如 Java、Python，C/C++等，或者各种.net 环境。

如果你准备使用 B/S 模式，还需要先熟练掌握 Tomcat、WebLogic 或者 IIS 等应用服务器的安装和应用。

如果使用 MySQL 开发 C/S 程序还需要安装其 ODBC 接口。

如果使用 Java 开发应用程序还需要熟悉 Java 的集成开发环境例如 Eclipse 或 MyEclipse 等，并安装 Java 插件。使用其他语言开发，也需要熟悉类似的集成开发环境。如果开发安卓类移动应用，需要安装安卓模拟器。

实验一　数据库定义与基本操作（4 学时）

1、实验目的

(1) 掌握 DBMS 的数据定义功能
(2) 掌握 SQL 语言的数据定义语句
(3) 掌握 DBMS 的数据单表查询功能
(4) 掌握 SQL 语言的数据单表查询语句

2、实验内容

(1) 创建数据库
(2) 创建、删除表
(3) 查看、修改表的定义
(4) 理解索引的特点
(5) 创建和删除索引
(6) SELECT 语句的基本用法
(7) 使用 WHERE 子句进行有条件的查询
(8) 使用 IN，NOT IN，BETWEEN AND 等谓词查询
(9) 利用 LIKE 子句实现模糊查询
(10)利用 ORDER BY 子句为结果排序
(11)用 SQL Server/MySQL 的聚集函数进行统计计算
(12)用 GR0UP BY 子句实现分组查询的方法

3、实验要求

(1) 熟练掌握 SQL 的数据定义语句 CREATE、ALTER、DROP、Select
(2) 写出实验报告

4、实验步骤

4.1 安装数据库

(1) 安装数据库管理系统 DBMS；
(2) 基于可视化界面或者命令行窗口创建数据库，命名为 CSEDB_学号；

4.2 基本表操作

设有一个学生-课程数据库，包括学生关系 Student、课程关系 Course 和选修关系 SC：

学生表：Student(Sno，Sname，Ssex，Sage，Sdept, Scholarship)

课程表：Course(Cno，Cname，Cpno，Ccredit)

学生选课表：SC(Sno，Cno，Grade)

利用 SQLServer 的查询分析器或 MySQL 的查询编辑器进行如下操作，不同的 DBMS 有少许差别，如果存在错误，需要根据错误提示自己排除

(1)　创建、删除表，例如：

```
CREATE TABLE Student
        (Sno          CHAR(5)   NOT NULL   UNIQUE,
         Sname   CHAR(20)   UNIQUE,
         Ssex          CHAR(1) ,
         Sage          INT,
         Sdept       CHAR(15),
         Scholarship CHAR(2))
CREATE TABLE SC(
             Sno CHAR(5),
             Cno CHAR(3),
             Grade     int,
             Primary key (Sno, Cno));
DROP TABLE   Student
```

(2) 查看、修改表的定义，例如：

```
ALTER TABLE Student ADD Scome DATETIME
ALTER TABLE Student ALTER COLUMN Sage SMALLINT
```

(3) 创建和删除索引

```
CREATE UNIQUE INDEX  Stusno ON Student(Sno)；
CREATE UNIQUE INDEX Coucno ON Course(Cno)；
CREATE UNIQUE INDEX SCno ON SC(Sno ASC，Cno DESC)；
DROP INDEX Stusno
```

4.3 删除数据库

4.4 创建（定义）示例数据库 S_T_学号

4.5 在数据库 S_T_学号中创建学生表 Student、课程表 Course 和选修表 SC

（1）创建 3 个表

① 利用 SQL 语句中的 Create Table 命令/或者可视化环境创建表

```
create table Student
  (Sno CHAR(9) PRIMARY KEY,
   Sname CHAR(20) UNIQUE,
   Ssex CHAR(2),
   Sage SMALLINT,
   Sdept CHAR(20),
   Scholarship char(2)
  );
go
/*表 Student 的主码为 Sno，属性列 Sname 取唯一值*/
create table Course
  (Cno CHAR(4) PRIMARY KEY,
   Cname CHAR(40),
   Cpno CHAR(4),
   Ccredit SMALLINT,
   FOREIGN KEY (Cpno) REFERENCES Course(Cno)
  );
go
/*表 Course 的主码为 Cno，属性列 Cpno(先修课)为外码，被参照表为 Course，被参照列是 Cno*/
create table SC
  (Sno CHAR(9),
   Cno CHAR(4),
   Grade SMALLINT,
   primary key (Sno, Cno),
   FOREIGN KEY (Sno) REFERENCES Student(Sno),
   FOREIGN KEY (Cno) REFERENCES Course(Cno)
  );
  go
  /*表 SC 的主码为(Sno, Cno), Sno 和 Cno 均为外码，被参照表分别为 Student 和 Course，被参照列分别为 Student.Sno 和 Course.Cno*/
```

（2）在 3 个表中添加示例数据（任选一种数据添加方法）

表 Student

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept	奖学金 Scholarship
200215121	李勇	男	20	CS	否
200215122	刘晨	女	19	CS	否
200215123	王敏	女	18	MA	否

200215125	张立	男	19	IS	否

表 Course

课程号 Cno	课程名 Cname	现行课 Cpno	学分 Ccredit
1	数据库	5	4
2	数学		2
3	信息系统	5	4
4	操作系统	6	3
5	数据结构	7	4
6	数据处理		2
7	PASCAL 语言	6	4

表 SC

学号 Sno	课程号 Cno	成绩 Grade
200215121	1	92
200215121	2	85
200215121	3	88
200215122	2	90
200215122	3	80

① 用 SQL 语句中的更新语句（Insert 语句、Update 语句和 Delete 语句）往 3 个表输入示例数据。

```
use S_T;/*将 S_T 设为当前数据库*/
insert into student values('200215121','李勇','男',20,'CS'，'否');
insert into student values('200215122','刘晨','女',19,'CS'，'否');
insert into student values('200215123','王敏','女',18,'MA'，'否');
insert into student values('200215125','张立','男',19,'IS'，'否');
go
/*为表 Student 添加数据*/
insert into course values('1', '数据库', NULL,4);
insert into course values('2', '数学',  NULL,2);
insert into course values('3', '信息系统', NULL,4);
insert into course values('4', '操作系统', NULL,3);
insert into course values('5', '数据结构', NULL,4);
insert into course values('6', '数据处理', NULL, 2);
insert into course values('7', 'java',  NULL,4);
go
update Course set Cpno = '5' where Cno = '1';
update Course set Cpno = '1' where Cno = '3';
update Course set Cpno = '6' where Cno = '4';
update Course set Cpno = '7' where Cno = '5';
update Course set Cpno = '6' where Cno = '7';
/*为表 Course 添加数据*/
go
insert into SC values('200215121', '1',92);
insert into SC values('200215121', '2',85);
insert into SC values('200215121', '3',88);
insert into SC values('200215122', '2',90);
insert into SC values('200215122', '3',80);
/*为表 SC 添加数据*/
go
```

② 利用可视化环境交互式输入数据。

4.6 对学生关系 Student、课程关系 Course 和选修关系 SC 进行查询。

4.6.1 基本练习

(1) SELECT 语句的基本用法

例如：查询全体学生的详细记录。

```
SELECT  Sno,Sname,Ssex,Sage,Sdept FROM Student
```

(2) 使用 WHERE 子句进行有条件的查询

例如：查询选修 2 号课程且成绩在 90 分以上的所有学生的学号、姓名

```
SELECT Student.Sno, student.Sname
FROM     Student, SC
WHERE Student.Sno = SC.Sno AND  SC.Cno= ' 2 ' AND  SC.Grade > 90
```

(3) 使用 IN，NOT IN，BETWEEN 等谓词查询

例如：查询信息系（IS）、数学系（MA）和计算机科学系（CS）学生的姓名和性别。

```
SELECT Sname,Ssex
FROM  Student
WHERE Sdept IN ( 'IS','MA','CS' )
```

例如：查询年龄在 20~23 岁（包括 20 岁和 23 岁）之间的学生的姓名、系别和年龄。

```
SELECT Sname,Sdept,Sage
FROM    Student
WHERE Sage BETWEEN 20 AND 23
```

(4)利用 LIKE 子句实现模糊查询

例如：查询所有姓刘学生的姓名、学号和性别。

```
SELECT Sname,Sno,Ssex
FROM Student
WHERE  Sname LIKE '刘%'
```

(5)利用 ORDER 子句为结果排序

例如：查询选修了 3 号课程的学生的学号及其成绩，查询结果按分数降序排列。

```
SELECT Sno,Grade
FROM  SC
WHERE  Cno= '3'
ORDER BY Grade DESC
```

(6)用 SQL Server 的统计函数进行统计计算

例如：计算 1 号课程的学生平均成绩。

```
SELECT AVG(Grade)
FROM SC
WHERE Cno= '1'
```

(7)用 GR0UP BY 子句实现分组查询的方法

例如：查询选修了 3 门以上课程的学生学号。

```
SELECT Sno
FROM  SC
```

```
GROUP BY Sno
HAVING    COUNT(*) >3
```

4.4.2 扩展练习（要求写出并执行 SQL 语句来完成以下各种操作，记录查询结果）

（1）查询全体学生的学号、姓名和年龄；

（2）查询所有计算机系学生的详细记录；

（3）找出考试成绩为优秀（90 分及以上）或不及格的学生的学号、课程号及成绩；

（4）查询年龄不在 19~20 岁之间的学生姓名、性别和年龄；

（5）查询数学系（MA）、信息系（IS)的学生的姓名和所在系；

（6）查询名称中包含“数据”的所有课程的课程号、课程名及其学分；

（7) 找出所有没有选修课成绩的学生学号和课程号；

（8）查询学生 200215121 选修课的最高分、最低分以及平均成绩；

（9）查询选修了 2 号课程的学生的学号及其成绩，查询结果按成绩升序排列；

（10）查询每个系名及其学生的平均年龄。

（思考：如何查询学生平均年龄在 19 岁以下（含 19 岁）的系别及其学生的平均年龄？）

实验二　SQL 的复杂操作（4 学时）

1、实验目的

掌握 SQL 语言的数据多表查询语句和更新操作

2、实验内容

(1) 等值连接查询（含自然连接查询）与非等值连接查询
(2) 自身连接查询
(3) 外连接查询
(4) 复合条件连接查询
(5) 嵌套查询（带有 IN 谓词的子查询）
(6) 嵌套查询（带有比较运算符的子查询）
(7) 嵌套查询（带有 ANY 或 ALL 谓词的子查询）
(8) 嵌套查询（带有 EXISTS 谓词的子查询）
(9) 集合查询
(10) update 语句用于对表进行更新
(11) delete 语句用于对表进行删除
(12) insert 语句用于对表进行插入

3、实验要求

(1) 熟练掌握 SQL 的连接查询语句
(2) 熟练掌握 SQL 的嵌套查询语句
(3) 掌握表名前缀、别名前缀的用法
(4) 掌握不相关子查询和相关子查询的区别和用法
(5) 掌握不同查询之间的等价替换方法（一题多解）及限制
(6) 熟练掌握 SQL 的数据更新语句 INSERT、UPDATE、DELETE
(7) 记录实验结果，认真完成实验报告

4、实验步骤

4.1 使用上次实验室的数据库，如果没有保存，则重新建立，并输入数据。

4.2 对学生关系 Student、课程关系 Course 和选修关系 SC 进行多表查询

4.2.1 基本练习

（1）等值连接查询与自然连接查询

例如：查询每个学生及其选修课的情况。

```
SELECT Student.*, SC.*
FROM Student, SC
WHERE Student.Sno = SC.Sno; /*  一般等值连接  */
```

又如：查询每个学生及其选修课的情况（去掉重复列）。

```
SELECT Student.Sno, Sname, Ssex, Sage, Cno, Grade
FROM Student, SC
WHERE Student.Sno = SC.Sno;    /*  自然连接--特殊的等值连接  */
```

（2）自身连接查询

例如：查询每一门课的间接先修课。

```
SELECT FIRST.Cno, SECOND.Cpno
FROM Course FIRST, Course SECOND
WHERE FIRST.Cpno = SECOND.Cno;
```

（3）外连接查询

例如：查询每个学生及其选修课的情况（要求输出所有学生--含未选修课程的学生的情况）

```
SELECT Student.Sno, Sname, Ssex, Sage, Sdept, Cno, Grade
FROM Student LEFT OUTER JOIN SC ON(Student.Sno = SC.Sno);
```

（4）复合条件连接查询

例如：查询选修了 2 号课程而且成绩在 90 以上的所有学生的学号和姓名。

```
SELECT Student.Sno, Sname
FROM Student, SC
WHERE Student.Sno = SC.Sno AND
        SC.Cno = '2' AND SC.Grade >= 90;
```

又如：查询每个学生的学号、姓名、选修的课程名及成绩。

```
SELECT Student.Sno, Sname, Cname, Grade
FROM Student, SC, Course
WHERE Student.Sno = SC.Sno AND
            SC.Cno = Course.Cno;
```

（5）嵌套查询（带有 IN 谓词的子查询）

例如：查询与“刘晨”在同一个系学习的学生的学号、姓名和所在系。

```
SELECT Sno, Sname, Sdept
FROM Student
WHERE Sdept IN
            (SELECT Sdept
              FROM Student
              WHERE Sname = '刘晨');     /*  解法一*/
```

可以将本查询中的 IN 谓词用比较运算符‘=’来代替：

```
SELECT Sno, Sname, Sdept
FROM Student
WHERE Sdept =
            (SELECT Sdept
              FROM Student
              WHERE Sname = '刘晨');     /*  解法二*/
```

也可以使用自身连接完成以上查询：

```
SELECT s1.Sno, s1.Sname, s1.Sdept
FROM Student s1, Student s2
WHERE s1.Sdept = S2.Sdept AND
              s2.Sname = '刘晨';          /*  解法三*/
```

还可以使用 EXISTS 谓词完成本查询：

```
SELECT Sno, Sname, Sdept
FROM Student S1
WHERE EXISTS
      (SELECT *
        FROM Student S2
        WHERE S2.Sdept=S1.Sdept AND S2.Sname='刘晨'); /*  解法四*/
```

又如：查询选修了课程名为“信息系统”的学生号和姓名。

```
SELECT Sno, Sname
FROM Student
WHERE Sno IN
            (SELECT Sno
              FROM SC
              WHERE Cno IN
                          (SELECT Cno
                            FROM Course
                            WHERE Cname = '信息系统'
                          )
            );
```

也可以使用连接查询来完成上述查询：

```
SELECT Student.Sno, Sname
FROM Student, SC, Course
WHERE Student.Sno = SC.Sno AND
        SC.Cno = Course.Cno AND
```

```
        Course.Cname = '信息系统';
```

（6）嵌套查询（带有比较运算符的子查询）

例如：找出每个学生超过他所选修课程平均成绩的课程号。

```
SELECT Sno, Cno
FROM SC x
WHERE Grade >= ( SELECT AVG(Grade)
                 FROM SC y
                 WHERE y.Sno = x.Sno);
```

（7）嵌套查询（带有 ANY 或 ALL 谓词的子查询）

例如：查询其他系中比计算机系某个学生年龄小的学生的姓名和年龄。

```
SELECT Sname, Sage
FROM   Student
WHERE Sage <ANY (SELECT Sage
                 FROM Student
                 WHERE Sdept = 'CS')
      AND Sdept <> 'CS';
```

本查询也可以使用聚集函数来实现：

```
SELECT Sname, Sage
FROM   Student
WHERE Sage < (SELECT MAX(Sage)
              FROM Student
              WHERE Sdept = 'CS')
      AND Sdept <> 'CS';
```

又如：查询其他系中比计算机系所有学生年龄都小的学生的姓名和年龄。

```
SELECT Sname, Sage
FROM   Student
WHERE Sage <ALL (SELECT Sage
                 FROM Student
                 WHERE Sdept = 'CS')
      AND Sdept <> 'CS';
```

也可以使用聚集函数来实现：

```
SELECT Sname, Sage
FROM   Student
WHERE Sage < (SELECT MIN(Sage)
              FROM Student
              WHERE Sdept = 'CS')
      AND Sdept <> 'CS';
```

（8）嵌套查询（带有 EXISTS 谓词的子查询）

例如：查询所有选修了 1 号课程的学生姓名。

```
SELECT Sname
FROM Student
WHERE EXISTS
            (SELECT *
              FROM SC
              WHERE Sno=Student.Sno AND Cno='1');
```

又如：查询所有未选修 1 号课程的学生姓名。

```
SELECT Sname
FROM Student
WHERE NOT EXISTS
            (SELECT *
              FROM SC
              WHERE Sno=Student.Sno AND Cno='1');
```

可以使用带有 EXISTS 谓词的子查询实现全称量词或蕴涵逻辑运算功能：

例如：查询选修了全部课程的学生姓名。

```
SELECT Sname
FROM Student
WHERE NOT EXISTS
        (SELECT *
         FROM Course
         WHERE NOT EXISTS
                 (SELECT *
                  FROM SC
                  WHERE Sno=Student.Sno AND
                                Cno=Course.Cno));
```

又如：查询至少选修了学生 200215122 选修的全部课程的学生号码。

```
SELECT DISTINCT Sno
FROM SC SCX
WHERE NOT EXISTS
        (SELECT *
         FROM SC SCY
         WHERE SCY.Sno='200215122' AND
                 NOT EXISTS
                     (SELECT *
                      FROM SC SCZ
                      WHERE SCZ.Sno=SCX.Sno AND
                                SCZ.Cno=SCY.Cno));
```

（9）集合查询

例如：查询计算机系的学生以及年龄不大于 19 岁的的学生。

```
SELECT *
FROM Student
WHERE Sdept='CS'
UNION        /*并集运算*/
SELECT *
FROM Student
WHERE Sage<=19;
```

可以改用多重条件查询：

```
SELECT *
FROM Student
WHERE Sdept='CS' OR Sage<=19;
```

又如：查询既选修了课程 1 又选修了课程 2 的学生（交集运算）。

```
SELECT Sno
FROM SC
WHERE Cno='1'
INTERSECT            /*交集运算*/
SELECT Sno
FROM SC
WHERE Cno='2';
```

可以使用嵌套查询：

```
SELECT Sno
FROM SC
WHERE Cno='1' AND Sno IN
                        (SELECT Sno
                         FROM SC
                         WHERE Cno='2');
```

思考：能不能改用多重条件查询？

```
SELECT Sno
FROM SC
WHERE Cno='1' AND Cno='2';
```

再如：查询计算机系的学生与年龄不大于 19 岁的学生的差集。

```
SELECT *
FROM Student
WHERE Sdept='CS'
EXCEPT              /*差集运算*/
SELECT *
FROM Student
WHERE Sage<=19;
```

可以改用多重条件查询：

```
SELECT *
FROM Student
WHERE Sdept='CS' AND Sage>19;
```

(10) update 语句用于对表进行更新

例如：将信息系所有学生的年龄增加 1 岁。

```
UPDATE Student
SET Sage= Sage+1
WHERE Sdept=' IS '
```

(11) delete 语句用于对表进行删除

例如：删除学号为 95019 的学生记录。

```
DELETE
FROM Student
WHERE Sno='95019'
```

(12) insert 语句用于对表进行插入

例如：插入一条选课记录('95020'，'1 ')。

```
INSERT
 INTO SC(Sno，Cno)
 VALUES (' 95020 '，' 1 ')
```

4.2.2 扩展练习（要求写出并执行 SQL 语句完成以下各种操作，记录查询结果）

（1）查询每门课程及其被选情况（输出所有课程中每门课的课程号、课程名称、选修该课程的学生学号及成绩--如果没有学生选择该课，则相应的学生学号及成绩为空值）。

（2）查询与“张立”同岁的学生的学号、姓名和年龄。（要求使用至少 3 种方法求解）

（3）查询选修了 3 号课程而且成绩为良好（80~89 分）的所有学生的学号和姓名。

（4）查询学生 200215122 选修的课程号、课程名

（思考：如何查询学生 200215122 选修的课程号、课程名及成绩？）

（5）找出每个学生低于他所选修课程平均成绩 5 分以上的课程号。（输出学号和课程号）

（6）查询比所有男生年龄都小的女生的学号、姓名和年龄。

（7）查询所有选修了 2 号课程的学生姓名及所在系。

（8）使用 update 语句把平均成绩为良的学生的年龄增加 2 岁，并查询出来。

（9）使用 insert 语句增加两门课程：C 语言和人工智能，并查询出来

（10）使用 delete 语句把人工智能课程删除，并查询出来。

实验三 SQL 的高级实验（4 学时）

1、实验目的

(1) 掌握 SQL 语言的视图、触发器、存储过程、安全等功能

2、实验内容

(1) 创建表的视图

(2) 利用视图完成表的查询

(3) 删除表的视图

(4) 创建触发器

(5) 创建存储过程

(6) 对用户进行授权和查询

(7) 用户定义完整性

3、实验要求

(1) 掌握视图的定义与操作

(2) 掌握对触发器的定义

(3) 掌握对存储过程的定义

(4) 掌握如何对用户进行授权和收回权限

(5) 掌握用户定义完整性的方法

(6) 写出实验报告

4、实验步骤（要求写出并执行 SQL 语句完成以下各种操作，记录查询结果）

使用上次实验室的数据库，如果没有保存，则重新建立，并输入数据。

(1) 创建 CS 系的视图 CS_View

(2) 在视图 CS_View 上查询 CS 系选修了 1 号课程的学生

(3) 创建 IS 系成绩大于 80 的学生的视图 IS_View

(4) 在视图 IS_View 查询 IS 系成绩大于 80 的学生

(5) 删除视图 IS_View

(6) 利用可视化窗口创建 2 个不同的用户 U1 和 U2,利用系统管理员给 U1 授予 Student 表的查询和更新的权限，给 U2 对 SC 表授予插入的权限。然后用 U1 登录，分别 1）查询学生表的信息；2）把所有学生的年龄增加 1 岁，然后查询；3）删除 IS 系的学生；4）查询 CS 系的选课信息。用 U2 登录，分别 1）在 SC 表中插入 1 条记录（‘200215122’，‘1’，75）；2）查询 SC 表的信息，3）查询视图 CS_View 的信息。

(7) 用系统管理员登录，收回 U1 的所有权限

(8) 用 U1 登录，查询学生表的信息

(9) 用系统管理员登录

(10) 对 SC 表建立一个更新触发器，当更新了 SC 表的成绩时，如果更新后的成绩大于等于 95，则检查该成绩的学生是否有奖学金，如果奖学金是“否”，则修改为“是”。如果修改后的成绩小于 95，则检查该学生的其他成绩是不是有大于 95 的，如果都没有，且修改前的成绩

是大于 95 时，则把其奖学金修改为”否”。然后进行成绩修改，并进行验证是否触发器正确执行。1）首先把某个学生成绩修改为 98，查询其奖学金。2）再把刚才的成绩修改为 80，再查询其奖学金。

（11）删除刚定义的触发器

（12）定义一个存储过程计算 CS 系的课程的平均成绩和最高成绩，在查询分析器或查询编辑器中执行存储过程，查看结果。

（13）定义一个带学号为参数的查看某个学号的所有课程的成绩，查询结果要包含学生姓名。进行验证。

（14）把上一题改成函数。再进行验证。

（15）在 SC 表上定义一个完整性约束，要求成绩再 0-100 之间。定义约束前，先把某个学生的成绩修改成 120，进行查询，再修改回来。定义约束后，再把该学生成绩修改为 120，然后进行查询。

实验四 数据库设计（4 学时）

1 实验目的：掌握数据库设计和开发技巧

2 实验内容：通过一个数据库具体设计实例，掌握数据库设计的方法。

3 实验要求：

熟练掌握使用 SQL 语句设计数据库的方法，实现前述实验的学生管理系统，完成实验报告。

4 系统功能要求：

1）新生入学信息增加，学生信息修改。

2）课程信息维护（增加新课程，修改课程信息，删除没有选课的课程信息）。

3）录入学生成绩，修改学生成绩。

4）按系统计学生的平均成绩、最好成绩、最差成绩、优秀率、不及格人数。

5）按系对学生成绩进行排名，同时显示出学生、课程和成绩信息。

6）输入学号，显示该学生的基本信息和选课信息。