

电子科技大学

实验报告

课程名称：数据库应用基础（创新学分）

学 院：计算机科学与工程学院

专 业：计算机科学与技术

学生姓名：蒋芷昕

学 号：2017180202005

指导教师：符朝旭

评 分：

日 期： 20 年 月 日

一、每个实验的报告格式见下一页,参照实验大纲填写各部分内容按实验大纲的顺序填写实验步骤、实验数据及结果分析,实验操作写在实验步骤中,操作结果写在结果分析中

二、全部实验写在一份 word 文档中,最后一并提交。提交前请使用 word 或 wps 软件的”另存为”功能,将报告保存为 PDF 格式,之后再提交。

三、实验报告提交到实验平台(主楼 A2-412,413)

实验一：结构化查询语言

实验学时：4 学时

一、实验内容和目的：

通过设置数据约束条件，体会数据库完整性的含义，约束条件下数据修改操作的限制。练习使用单表查询、多表查询、组查询、子查询。

二、实验原理：

数据库的完整性、约束条件、结构化查询语言。

三、实验器材（设备、元器件）

PC 机

操作系统：Windows

数据库：Microsoft SQL server

四、实验步骤：

(1) 恢复数据库 mydb，备份文件为 mydb.bak；

(2) 完成数据操纵和查询，给出操作代码和执行结果截图；

- ① 添加数据库约束条件，要求学生的成绩的取值范围为“0”到“100”之间；
- ② 将你的学生信息添加到学生表中，要求学号姓名为真实数据，其它字段随意；
- ③ 你要选修全部课程，使用一条 SQL 语句实现该功能；
- ④ 使用 update 命令登记你的全部课程分数，分数取值随意；
- ⑤ 查询你的选课记录，返回课程号、课程名、分数；
- ⑥ 将学号为“101”学生的学号改为“020060101”，且同时更改该所有的选课信息；
- ⑦ 查询年龄在指定区间（比如 20—28 之间）的学生姓名（通过出生日期和当前日期计算年龄 $\text{year}(\text{getdate}) - \text{year}(\text{stud.birthd})$ ）；
- ⑧ 查询姓“张”的学生的学号、姓名、邮件地址；
- ⑨ 统计每个学生的选课情况，返回：学号、姓名、选课门数、总学分数；
- ⑩ 统计每门课程的选课情况，返回：课程号、最高分、最低分、平均分；

- 11 查询每门课程获得最高分的学生信息，返回课程号、课程名、最高分、学号、姓名；
 - 12 查询既选修了 1 号课程，又选修了 2 号课程的学生学号和姓名。
 - 13 查询选修了全部课程的学生学号及姓名；
-

五、实验数据及结果分析：

- (1) 恢复数据库 mydb，备份文件为 mydb.bak；
- (2) 完成数据操纵和查询，给出操作代码和执行结果截图；

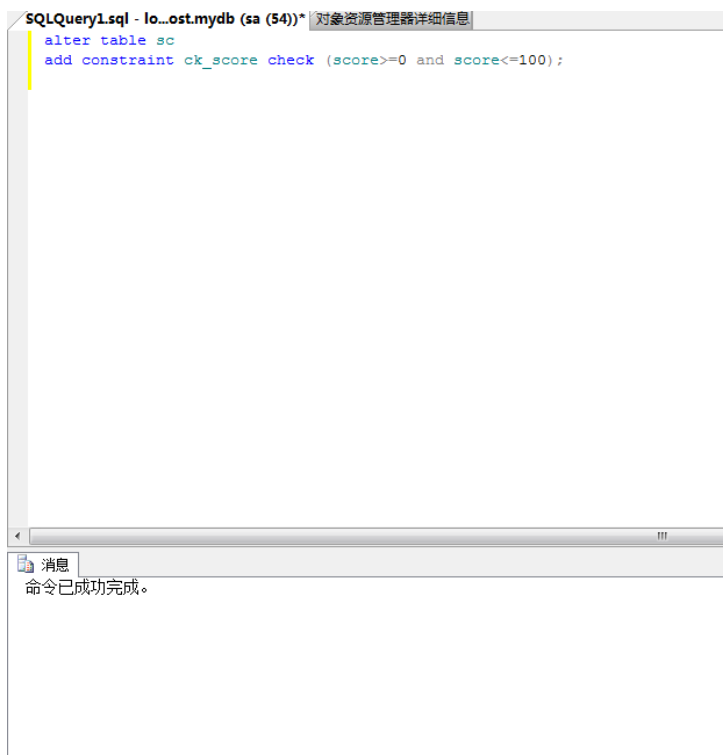
① 添加数据库约束条件，要求学生的成绩的取值范围为“0”到“100”之间；

代码（文本）：

```
>alter table sc
```

```
add constraint ck_score check (score>=0 and score<=100);
```

运行结果（截图）：

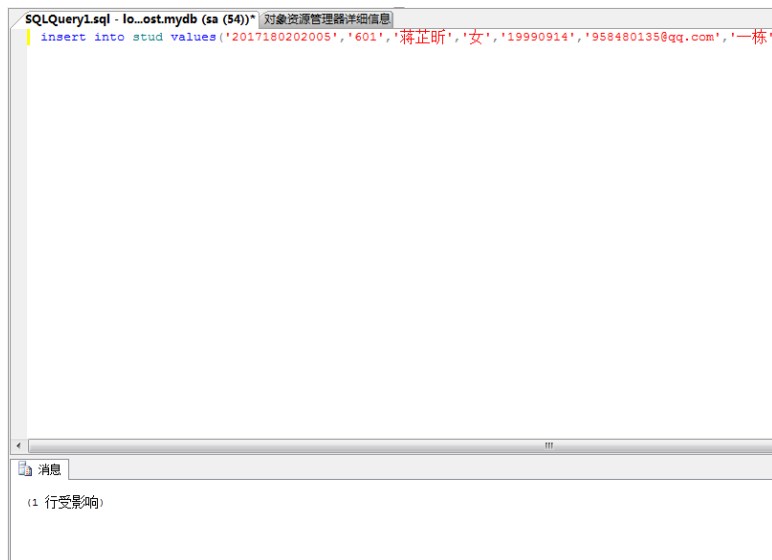


② 将你的学生信息添加到学生表中，要求学号姓名为真实数据，其它字段随意；

代码（文本）：

```
>insert into stud values('2017180202005','601','蒋芷昕','女','19990914','958480135@qq.com','一栋');
```

运行结果（截图）：



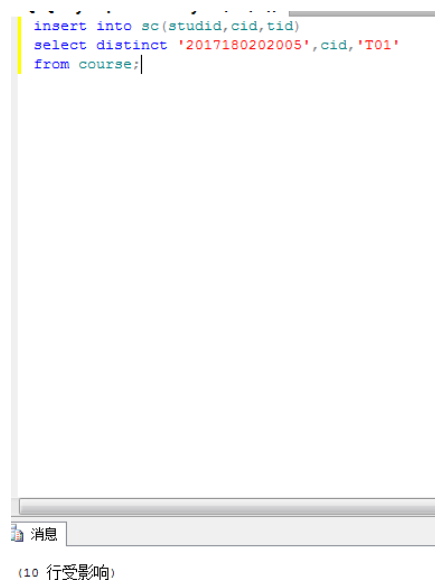
	StudID	DEPID	SNAME	sex	BIRTHD	EMAIL	HOMEADDR
1	2017180202005	601	蒋芷昕	女	1999-09-14 00:00:00.000	958480135@qq.com	一栋

③ 你要选修全部课程，使用一条 SQL 语句实现该功能；

代码（文本）：

```
>insert into sc(studid,cid,tid)
select distinct '2017180202005',cid,'T01'
from course;
```

运行结果（截图）：



	StudID	CID	TID	SCORE
1	2017180202005	6001	T01	NULL
2	2017180202005	6002	T01	NULL
3	2017180202005	6003	T01	NULL
4	2017180202005	6004	T01	NULL
5	2017180202005	6005	T01	NULL
6	2017180202005	6006	T01	NULL
7	2017180202005	6007	T01	NULL
8	2017180202005	6008	T01	NULL
9	2017180202005	6009	T01	NULL
10	2017180202005	6011	T01	NULL

④ 使用 update 命令登记你的全部课程分数，分数取值随意；
代码（文本）：

```
>update sc
```

```
set score=100
```

```
where studid='2017180202005'
```

运行结果（截图）：

```
update sc
set score=100
where studid='2017180202005'
```

消息

(10 行受影响)

	StudID	CID	TID	SCORE
1	2017180202005	6001	T01	100
2	2017180202005	6002	T01	100
3	2017180202005	6003	T01	100
4	2017180202005	6004	T01	100
5	2017180202005	6005	T01	100
6	2017180202005	6006	T01	100
7	2017180202005	6007	T01	100
8	2017180202005	6008	T01	100
9	2017180202005	6009	T01	100
10	2017180202005	6011	T01	100

⑤ 查询你的选课记录，返回课程号、课程名、分数：

代码（文本）：

```
>select course.cid 课程号,cname 课程名,sc.score 分数
from course,sc
where course.cid=sc.cid
and studid='2017180202005';
```

运行结果（截图）：

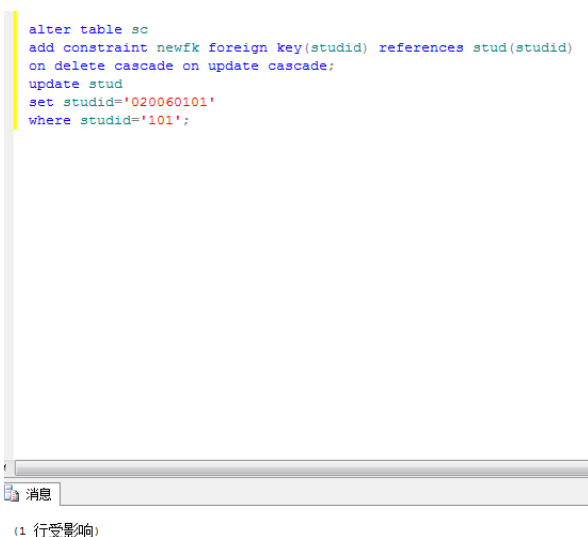
	课程号	课程名	分数
1	6001	计算机组成原理	100
2	6002	操作系统	100
3	6003	数据结构	100
4	6004	数据库原理	100
5	6005	Computer Network	100
6	6006	Objected JAVA	100
7	6007	Software Engeneering	100
8	6008	UNIX Basic	100
9	6009	UNIX OS Design	100
10	6011	数据库应用开发	100

⑥ 将学号为“101”学生的学号改为“020060101”，且同时更改该所有的选课信息；

代码（文本）：

```
>alter table sc
add constraint newfk foreign key(studid) references stud(studid)
on delete cascade on update cascade;
update stud
set studid='020060101'
where studid='101';
```

运行结果（截图）：



⑦ 查询年龄在指定区间（比如 20—28 之间）的学生姓名（通过出生日期和当前日期计算年龄
`year(getdate())-year(stud.birthd)`）；

代码（文本）：

```
>select sname
from stud
where year(getdate())-year(stud.birthd) between 20 and 28;
```

运行结果（截图）：



	sname
1	陈刚
2	陈海霞
3	楚刚
4	楚国梁
5	楚海霞
6	楚敏
7	冯国梁
8	冯敏
9	韩刚
10	韩国梁
11	韩海霞
12	蒋刚
13	蒋海霞
14	蒋敏
15	李刚

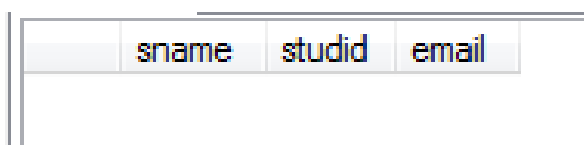
查询已成功执行。

⑧ 查询姓“张”的学生的学号、姓名、邮件地址；

代码（文本）：

```
>select sname,studid,email
from stud
where sname like '张%';
```

运行结果（截图）：



sname	studid	email
-------	--------	-------

⑨ 统计每个学生的选课情况，返回：学号、姓名、选课门数、总学分数；

代码（文本）：

```
>select sc.studid 学号,sname 姓名,count(sc.cid) 选课门数,sum(course.credits) 总学分
from sc,stud,course
where stud.studid=sc.studid and sc.cid=course.cid
group by sc.studid,stud.sname;
```

运行结果（截图）：

	学号	姓名	选课门数	总学分
1	020060101	陈刚	10	24.0
2	102	陈国梁	5	14.0
3	103	陈海霞	5	14.0
4	104	陈敏	5	14.0
5	105	楚刚	5	14.0
6	106	楚国梁	5	14.0
7	107	楚海霞	5	14.0
8	108	楚敏	5	14.0
9	109	冯刚	5	14.0
10	110	冯国梁	5	14.0
11	111	冯海霞	5	14.0
12	112	冯敏	5	14.0
13	113	韩刚	5	14.0
14	114	韩国梁	5	14.0
15	115	韩海霞	5	14.0
16	116	韩敏	5	14.0
17	117	蒋刚	5	14.0
18	118	蒋国梁	5	14.0
19	119	蒋海霞	5	14.0
20	120	蒋敏	5	14.0

⑩ 统计每门课程的选课情况，返回：课程号、最高分、最低分、平均分；

代码（文本）：

```
>select cid 课程号,max(score) 最高分,min(score) 最低分,avg(score) 平均分
from sc
group by cid;
```

运行结果（截图）：

	课程号	最高分	最低分	平均分
1	6001	100	48	73
2	6002	100	48	73
3	6003	100	48	73
4	6004	100	48	74
5	6005	100	48	75
6	6006	100	100	100
7	6007	100	100	100
8	6008	100	100	100
9	6009	100	100	100
10	6011	100	48	73

11 查询每门课程获得最高分的学生信息，返回课程号、课程名、最高分、学号、姓名；

代码（文本）：

```
>select course.cid 课程号,cname 课程名,sc.score 最高分,sc.studid 学号,stud.sname 姓名
from stud,sc,course,(select sc.cid,max(sc.score)scores from sc,course where sc.cid=course.cid group by(sc.cid))
maxscore
where stud.studid=sc.studid
and sc.cid=maxscore.cid
and course.cid=sc.cid
and sc.score = maxscore.scores;
```

运行结果（截图）：

	课程号	课程名	最高分	学号	姓名
1	6011	数据库应用开发	100	118	蒋国梁
2	6011	数据库应用开发	100	2017180202005	蒋芷昕
3	6009	UNIX OS Design	100	2017180202005	蒋芷昕
4	6008	UNIX Basic	100	2017180202005	蒋芷昕
5	6007	Software Engineering	100	2017180202005	蒋芷昕
6	6006	Objected JAVA	100	2017180202005	蒋芷昕
7	6005	Computer Network	100	2017180202005	蒋芷昕
8	6005	Computer Network	100	149	杨刚
9	6004	数据库原理	100	2017180202005	蒋芷昕
10	6004	数据库原理	100	113	韩刚
11	6003	数据结构	100	140	王敏
12	6003	数据结构	100	2017180202005	蒋芷昕
13	6002	操作系统	100	163	周海霞
14	6002	操作系统	100	2017180202005	蒋芷昕
15	6002	操作系统	100	110	冯国梁
16	6001	计算机组成原理	100	2017180202005	蒋芷昕
17	6001	计算机组成原理	100	128	钱敏

12 查询既选修了 1 号课程，又选修了 2 号课程的学生学号和姓名。

代码（文本）：

```
>select stud.studid 学号,sname 姓名
from stud,(SELECT first.studid FROM SC first,sc second where first.cid='6001' and second.cid='6002' and
first.studid=second.studid)c
where STUD.StudID=c.StudID;
```

运行结果（截图）：

	学号	姓名
1	020060101	陈刚
2	102	陈国梁
3	103	陈海霞
4	104	陈敏
5	105	楚刚
6	106	楚国梁
7	107	楚海霞
8	108	楚敏
9	109	冯刚
10	110	冯国梁
11	111	冯海霞
12	112	冯敏
13	113	韩刚
14	114	韩国梁
15	115	韩海霞
16	116	韩敏
17	117	蒋刚
18	118	蒋国梁
19	119	蒋海霞
20	120	蒋敏

13 查询选修了全部课程的学生学号及姓名；

代码（文本）：

```
>select sc.studid 学号,sname 姓名
from stud,sc
where stud.studid=sc.studid
group by sc.studid,stud.sname
having count(cid)=(select count(cid) from course);
```

运行结果（截图）：

	学号	姓名
1	020060101	陈刚
2	2017180202005	蒋芷昕

六、实验结论、心得体会和改进建议：

通过实际操作 SQL SERVER，对数据库进行修改、插入、查询等基本的数据库操作，加深了对数据库完整性、约束条件、结构化查询等概念的理解，同时体会到了数据库系统强大的可操作性。针对一项查询，可以构建许多不同的查询操作。

实验二：数据库建模

实验学时：4 学时

一、实验内容和目的：

学习数据库建模工具 PowerDesigner 设计数据库。学习数据库建模工具 PowerDesigner 最基本的使用方法，使用 PDM（物理模型），以图形化界面方式创建表及确定各表之间的关系；通过物理模型生成创建数据库的脚本；

- (1) 使用 PowerDesigner 设计一个数据库物理模型，在实验报告中给出模型图；
- (2) 通过该模型生成数据库脚本，在实验报告中给出生成的脚本；

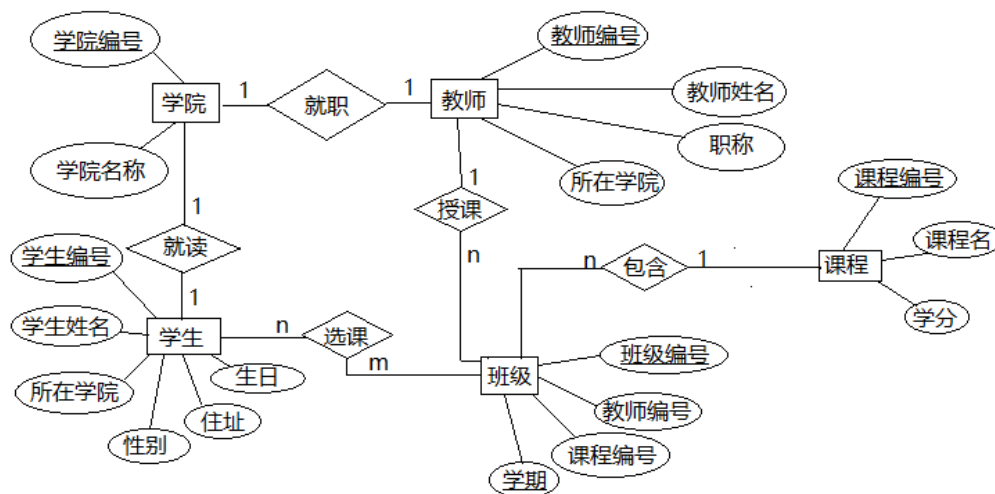
二、实验原理：

1. 使用 PowerDesigner 设计一个数据库物理模型，在实验报告中给出模型图；

2. 通过该模型生成数据库脚本，在实验报告中给出生成的脚本；

分析数据，共建立 6 张表存储数据：

- (1) Department, teacher, student, course 四张表分别记录学院、学生、教师、课程的基本信息，tc 用于连接教师与所授课程的关系，sc 连接学生与所选班级的关系。E-R 图如下：



三、实验器材（设备、元器件）

操作系统：Windows

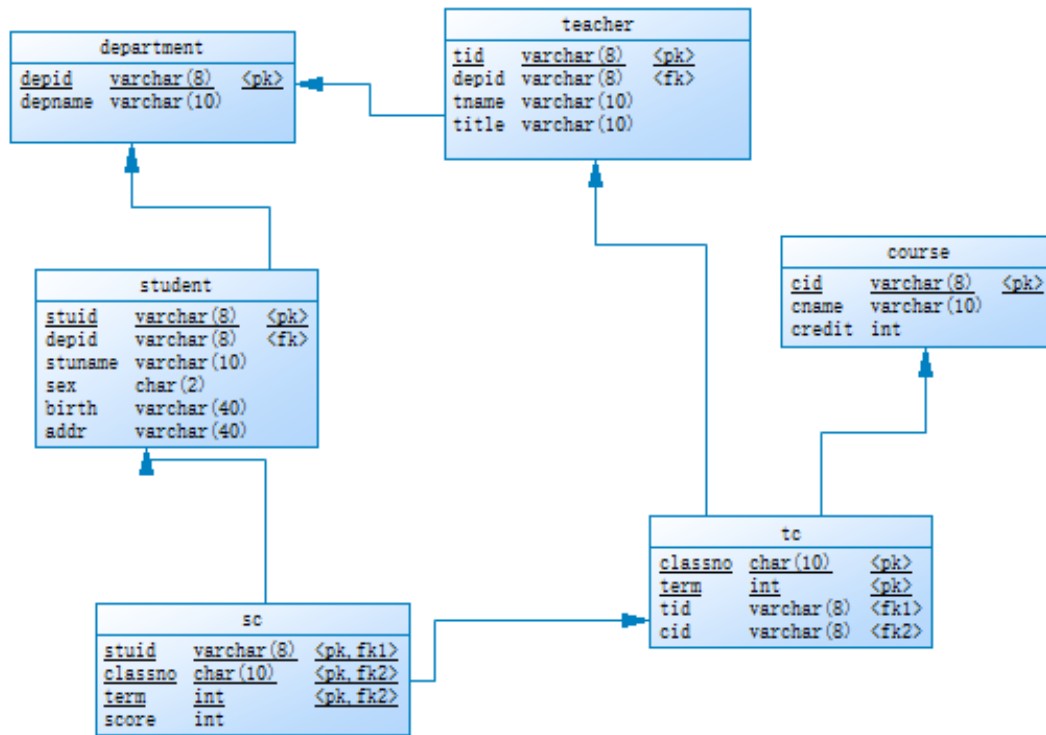
应用软件：Power Designer

四、实验步骤：

1. 使用 PowerDesigner 画出基于 sqlserver 的物理模型图；
2. 生成创建数据库的代码；

五、实验数据及结果分析：

1. 使用 PowerDesigner 画出基于 sqlserver 的物理模型图；
模型图为（贴图）：



>

2. 生成创建数据库的代码；
自动生成的代码（文本）：

```

> /*=====*/
/* Table: course                                     */
/*=====*/

create table course
(
    cid                varchar(8)                not null,
    cname              varchar(10)               null,
    credit             int                       null,
    constraint PK_COURSE primary key clustered (cid)
);

/*=====*/
/* Table: department                                   */
/*=====*/

create table department
(
    depid              varchar(8)                not null,
    depname            varchar(10)               null,

```

```

        constraint PK_DEPARTMENT primary key clustered (depid)
    );

/*=====*/
/* Table: sc                                     */
/*=====*/
create table sc
(
    stuid          varchar(8)          not null,
    classno        char(10)            not null,
    term           int                 not null,
    score          int                 null,
    constraint PK_SC primary key clustered (stuid, classno, term)
);

/*=====*/
/* Table: student                               */
/*=====*/
create table student
(
    stuid          varchar(8)          not null,
    depid          varchar(8)          null,
    stuname        varchar(10)         null,
    sex            char(2)             null,
    birth          varchar(40)          null,
    addr           varchar(40)          null,
    constraint PK_STUDENT primary key clustered (stuid)
);

/*=====*/
/* Table: tc                                     */
/*=====*/
create table tc
(
    classno        char(10)            not null,
    term           int                 not null,
    tid            varchar(8)          not null,
    cid            varchar(8)          not null,
    constraint PK_TC primary key clustered (classno, term)
);

/*=====*/
/* Table: teacher                               */
/*=====*/
create table teacher
(
    tid            varchar(8)          not null,

```

```
    depid          varchar(8)          null,
    tname          varchar(10)         null,
    title          varchar(10)         null,
    constraint PK_TEACHER primary key clustered (tid)
);
```

```
alter table sc
    add constraint FK_SC_FK_SC_TC_TC foreign key (classno, term)
        references tc (classno, term)
        on update restrict
        on delete restrict;
```

```
alter table sc
    add constraint FK_SC_REFERENCE_STUDENT foreign key (stuid)
        references student (stuid)
        on update restrict
        on delete restrict;
```

```
alter table student
    add constraint FK_STUDENT_FK_STUD_D_DEPARTME foreign key (depid)
        references department (depid)
        on update restrict
        on delete restrict;
```

```
alter table tc
    add constraint FK_TC_FK_TC_COU_COURSE foreign key (cid)
        references course (cid)
        on update restrict
        on delete restrict;
```

```
alter table tc
    add constraint FK_TC_FK_TC_TEA_TEACHER foreign key (tid)
        references teacher (tid)
        on update restrict
        on delete restrict;
```

```
alter table teacher
    add constraint FK_TEACHER_FK_TEACHE_DEPARTME foreign key (depid)
        references department (depid)
        on update restrict
        on delete restrict;
```

六、实验结论、心得体会和改进建议：

利用 PowerDesigner 软件，可以直观明了的搭建一个数据库物理模型，帮助学生理解对数据库模型间关系，有利于数据库知识的学习，掌握其应用。

通过本次数据库的设计，学生在实践中巩固并掌握了数据库系统设计的步骤、流程及思想，增长了在数据库设计方面的见识。使学生深刻认识到数据库学习的重要性，也帮助学生发现了未能及时纠正的，对数据库理解上的错误，同时也掌握到了很多新知识。尤其是在课本中难以获得的实践的知识与能力，学生通过本次实验设计能够体会到理论知识和实践相结合的重要性。