

实验 1 数据库

实验 1.1 创建数据库

3. 使用 Transact-SQL 指定参数创建数据库

(4) 创建指定数据库 testbase1

```
CREATE DATABASE testbase1
ON
( NAME=testbase1_data,
  FILENAME='e:\张小山数据库\testbase1_data.mdf')
LOG ON
( NAME=testbase1_log,
  FILENAME='e:\张小山数据库\testbase1_log.ldf')
GO
```

(5) 指定多个参数创建数据库 testbase2

```
USE master
GO
CREATE DATABASE testbase2
ON
PRIMARY
(NAME=prim_sub_dat1,
  FILENAME='E:\张小山数据库\prim_sub1_dat.mdf',
  SIZE=5MB,
  MAXSIZE=50MB,
  FILEGROWTH=20% ),
(NAME=prim_sub_dat2,
  FILENAME='E:\张小山数据库\prim_sub2_dat.ndf',
  SIZE=5MB,
  MAXSIZE=50MB,
  FILEGROWTH=20% ),
FILEGROUP Grouptest1
(NAME=group1_sub1,
  FILENAME='E:\张小山数据库\group1_sub1_dat.ndf',
  SIZE=5MB,
  MAXSIZE=50MB,
  FILEGROWTH=5MB ),
(NAME=group1_sub2,
  FILENAME='E:\张小山数据库\group1_sub2_dat.ndf',
  SIZE=5MB,
  MAXSIZE=50MB,
  FILEGROWTH=5MB ),
FILEGROUP Grouptest2
(NAME=group2_sub1,
  FILENAME='E:\张小山数据库\group2_sub1_dat.ndf',
  SIZE=5MB,
  MAXSIZE=50MB,
  FILEGROWTH=15% ),
(NAME=group2_sub2,
```

```

FILENAME='E:\张小山数据库\group2_sub2_dat.ndf',
SIZE=5MB,
MAXSIZE=50MB,
FILEGROWTH=15% )
LOG ON
(NAME=testbase2_log,
FILENAME='E:\张小山数据库\testbase2_log_file.ldf',
SIZE=5MB,
MAXSIZE=25MB,
FILEGROWTH=5MB )
GO

```

实验 1.2 分离数据库

2. 使用系统存储过程分离数据库 testbase1

```
exec sp_detach_db testbase1, true
```

实验 1.4 附加数据库

2. 使用系统存储过程安装数据库 testbase1

```
EXEC sp_attach_single_file_db @dbname='testbase1',
@physname='e:\张小山数据库\testbase1_data.mdf'
```

实验 1.5 删除数据库

2. 使用 Transact-SQL 删除数据库 testbase1

```
DROP DATABASE testbase1
```

实验 2 数据库表

实验 2.1 创建数据库表

3. 用 SQL 语句创建数据表 C

```

CREATE TABLE C(CNO CHAR(2),
CN CHAR(10),
CT TINYINT )

```

4. 用 SQL 语句创建数据表 SC

```

CREATE TABLE SC(SNO CHAR(2),
CNO NCHAR(2),
SCORE TINYINT)

```

5. 用 SQL 语句创建数据表 TC

```

CREATE TABLE TC(TNO CHAR(2),
CNO CHAR(2))

```

实验 2.2 修改数据库表结构

1. 用 SQL 向数据表 S 中添加列 NATIVE、修改列 NATIVE

(2) 向 S 表中添加新列。新列定义为, 列名: NATIVE, 数据类型: NCHAR, 长度: 40, NULL。

```
ALTER TABLE S
```

```
ADD NATIVE NCHAR(40) NULL
```

(6) 修改表 S 中的列名为 NATIVE 的定义。新定义为, 列名: NATIVE, 数据类型: CHAR, 长度: 16, NULL。

```
ALTER TABLE S
```

```
ALTER COLUMN NATIVE CHAR(16) NULL
```

实验 2.3 删除数据库表

2. 用 SQL 删除数据表 test2。

```
USE jxsk
```

```
DROP TABLE test2
```

```
GO
```

实验 3 数据操作

3. 使用 INSERT INTO 语句插入数据至数据表 C 中

(2) 插入如图 3-8 所示中的第一行记录。在查询窗口中, 输入下面 INSERT-INTO 语句:

```
USE jxsk
```

```
INSERT INTO C VALUES('C1','程序设计','60')
```

实验 3.2 修改数据

2. 用 SQL 语句修改数据表 T 中的数据

(3) 修改数据表 T 中“张兰”教师的职称。

```
USE jxsk
```

```
UPDATE T SET PROF='教授' WHERE TN='张兰'
```

2. 用 SQL 语句删除数据表 T 中的数据

(2) 删除数据表 T 中“张兰”教师的记录。

```
USE jxsk
```

```
DELETE FROM T WHERE TN='张兰'
```

实验 3.4 复制数据库表

1. 用 SQL 复制数据表 S 生成一新表 test1

(1) 创建复制数据表的 SQL 命令。

```
USE jxsk
```

```
SELECT * INTO test1 FROM S
```

```
GO
```

2. 用 SQL 复制数据表 T 中“男”同学记录, 生成一新表 test2

(1) 创建复制数据表 T 的 SQL 命令, 复制数据表 T 中男同学的列: TN, SEX, AGE, PROF, 生成一个新表 test2:

```
use jxsk
```

```
Select TN, SEX, AGE, PROF into test2 From T
```

```
where sex ='男'
```

```
go
```

实验 4 完整性约束

实验 4. 1 实体完整性约束

3. 用 SQL 为现有表 T 在 TNO 列上创建 PRIMARY KEY 约束

(3) 为表 T 在列 TNO 上创建主键:

```
ALTER TABLE T
ADD CONSTRAINT PK_TNO PRIMARY KEY(TNO)
```

4. 用 SQL 创建新表 TEST_C, 并以列约束形式创建 PRIMARY KEY 约束

(2) 创建新表及其主键约束。创建数据表 TEST_C, 并以列约束的形式在列 CNO 上创建主键:

```
USE jxsk
CREATE TABLE TEST_C(
    CNO CHAR(2) CONSTRAINT PK_TEST_C PRIMARY KEY,
    CN CHAR(10),
    CT TINYINT)
```

5. 用 SQL 创建新表 TEST_TC, 并以表约束形式创建 PRIMARY KEY 约束

(2) 创建新表及其主键约束。创建数据表 TEST_TC, 并以表约束的形式在列 TNO 和 CNO 上创建主键, 主键约束名为 PK_TEST_TC:

```
USE jxsk
CREATE TABLE TEST_TC(
    TNO CHAR(2),
    CNO CHAR(2),
    CONSTRAINT PK_TEST_TC PRIMARY KEY(TNO, CNO))
```

7. 用 SQL 为现有表 C 中的 CN 列创建 “UNIQUE 约束”

(2) 为数据表 C 在课程名 CN 列上创建 UNIQUE 约束, 约束名为 UNIQUE_C:

```
USE jxsk
ALTER TABLE C
ADD CONSTRAINT UNIQUE_C UNIQUE(CN)
```

9. 用 SQL 为现有表 TEST_TC 增加新列 ID_TC, 并创建此列属性为 IDENTITY

(2) 为数据表 TEST_TC 增加一个新列 ID_TC, 并设置此列为 IDENTITY:

```
USE jxsk
ALTER TABLE TEST_TC
ADD ID_TC INT IDENTITY
```

11. 用 SQL 删除表 C 中 CN 列的 UNIQUE 约束 UNIQUE_C

(2) 删除数据表 C 中的唯一约束 “UNIQUE_C”:

```
USE jxsk
ALTER TABLE C DROP UNIQUE_C
```

实验 4. 2 域完整性约束

1. 用 SQL 给现有表 S 中的年龄列 AGE 创建取值范围在 14 至 40 岁之间的 CHECK 约束, 并检查表中的现有数据是否符合此 CHECK 约束

(2) 约束名为 CHECK_AGE:

```
USE jxsk
ALTER TABLE S WITH CHECK
ADD CONSTRAINT CHECK_AGE CHECK( AGE>=14 AND AGE<=40)
```

GO

3. 用 SQL 创建一新表 TEST_S, 包含 DEFAULT 和 CHECK 约束。

(2) 约束名为 CHECK_AGE:

```
USE jxsk
CREATE TABLE TEST_S(
    SNO CHAR(2) NOT NULL,
    SN CHAR(8) NOT NULL,
    SEX CHAR(2) NULL CONSTRAINT DEFAULT_SEX DEFAULT '男'
    CONSTRAINT CHECK_SEX CHECK(SEX='男' OR SEX='女'),
    AGE TINYINT NULL CONSTRAINT DEFAULT_AGE DEFAULT 18,
    CONSTRAINT CHECK_S_AGE CHECK(AGE>=14 AND AGE<=40))
GO
```

4. 用 SQL 删除表 T 中的 PROF 和 COMM 之间的 CHECK 约束 CHECK_T

```
USE jxsk
ALTER TABLE T
    DROP CONSTRAINT CHECK_T
GO
```

实验 4.3 引用完整性约束

2. 用 SQL 创建表 T 与表 TC 之间的引用关系

(2) 创建表 T 与表 TC 的关联关系的 SQL 语句。

```
ALTER TABLE TC WITH NOCHECK
    ADD CONSTRAINT FK_T_TC FOREIGN KEY (TNO) REFERENCES T(TNO)
    ON DELETE CASCADE
```

实验 4.4 规则

1. 用 SQL 创建、绑定、解除规则

(2) 创建规则 RU_SEX:

```
USE jxsk
GO
CREATE RULE RU_SEX AS @SEX IN('男','女')
GO
```

(4) 绑定规则 RU_SEX 到表 T 中的 SEX 列。

```
USE jxsk
EXEC sp_bindrule RU_SEX, 'T.SEX'
GO
```

(7) 解除规则在表 T 上的绑定。

```
USE jxsk
EXEC sp_unbindrule 'T.SEX'
GO
```

实验 5.1 创建索引

1. 用 SQL 为表 T 在 TNO 列上按降序创建聚簇索引 IND_TNO

(2) 创建 SQL 语句, 为表 T 在的 TNO 列上按降序创建聚簇索引 IND_TNO:

```
USE jxsk
CREATE CLUSTERED INDEX IND_TNO ON _T(TNO DESC)
GO
```

4. 用 SQL 为表 C 在 CN 列上按升序创建唯一索引 IND_CN

(2) 创建 SQL 语句, 为表 C 在的 CN 列上按降序创建聚簇索引 IND_CN:

```
USE jxsk
CREATE UNIQUE INDEX IND_CN ON C(CN)
GO
```

实验 5.2 删除索引

2. 用 SQL 删除表 T 中的索引 IND_SN_AGE

(3) 创建 SQL 语句, 删除表 T 中的索引 IND_SN_AGE:

```
USE jxsk
DROP INDEX T.IND_SN_AGE
GO
```

实验 6 视图

1. 使用 SQL 创建课程表视图 VIEW_CTABLE

```
USE jxsk
GO
CREATE VIEW VIEW_CTABLE
AS SELECT TN,CN FROM T,C,TC
WHERE T.TNO=TC.TNO AND C.CNO=TC.CNO
GO
```

2. 对视图 VIEW_S 执行 INSERT 语句

```
USE jxsk
INSERT INTO VIEW_S VALUES ('古明','男',18,'自动化')
GO
```

3. 修改视图 VIEW_S 的定义, 使其包含 S 表的主键, 再执行插入操作

(2) 创建修改视图的 SQL 语句, 使其包含表 S 中的字段: SNO, SN, SEX, DEPT

```
USE jxsk
GO
ALTER VIEW VIEW_S
AS SELECT SNO, SN, SEX, DEPT FROM S
GO
```

(5) 创建 SQL 语句, 向视图中插入数据:

```
USE jxsk
INSERT INTO VIEW_S VALUES ('S8', '古明','男','自动化')
```

4. 对视图 VIEW_S 执行 UPDATE 语句, 查看此视图的基本表 S 中数据的变化

```
USE jxsk
UPDATE VIEW_S SET DEPT='计算机' WHERE SN='古明'
GO
```

实验 6.3 修改视图

2. 使用 SQL 给视图 VIEW_CTABLE 增加一个课时字段: CT TINYINT

```
USE jxsk
GO
ALTER VIEW VIEW_CTABLE
AS SELECT TN,CN,CT FROM T,C,TC
WHERE T.TNO=TC.TNO AND C.CNO=TC.CNO
```

```
GO
```

2. 使用 SQL 删除视图 VIEW_CTABLE

(3) 创建删除视图的 SQL 语句。

```
USE jxsk
GO
DROP VIEW VIEW_CTABLE
GO
```

实验 7 数据查询

实验 7.1 单表查询

1. 指定列或全部列查询。

(1) 查询 S 表中全体学生的详细记录；

```
USE jxsk
SELECT * FROM S
GO
```

(2) 查询所有学生的姓名及其出生年份。

```
USE jxsk
SELECT SN, year(date)-AGE FROM S
GO
```

2. 按条件查询及模糊查询

(1) 查询考试成绩有不及格学生的学号。

```
USE jxsk
SELECT DISTINCT SNO FROM SC WHERE SCORE < 60
GO
```

(2) 查询年龄在 20~23 岁之间的学生姓名、系名、年龄。

```
USE jxsk
SELECT SN, DEPT, AGE FROM S
WHERE AGE BETWEEN 20 AND 23
GO
```

(3) 查询姓李的学生的姓名、学号和性别。

```
USE jxsk
SELECT SN, SNO, SEX FROM S
WHERE SN LIKE '李%'
GO
```

(4) 查询名字中第 2 个字为“明”字的男学生的姓名和系名。

```
USE jxsk
SELECT SN AS 姓名, DEPT AS 系名 FROM S
WHERE SN LIKE '_明%' AND SEX = '男'
GO
```

3. 对查询结果排序

(1) 查询信息系、计算机系学生的姓名、系名，结果按系名升序，姓名降序排序。

```
USE jxsk
SELECT SN AS 姓名, DEPT AS 系名 FROM S
```

```
WHERE DEPT IN ('信息' , '计算机' )
```

```
ORDER BY DEPT , SN DESC
```

```
GO
```

- (2) 查询所有有课程号 C2 成绩的学生的学号、课程号和成绩，并按成绩降序排序。

```
USE jxsk
```

```
SELECT SNO AS 学号, CNO AS 课号, SCORE AS 成绩 FROM SC
```

```
WHERE CNO='C2' AND SCORE IS NOT NULL
```

```
ORDER BY SCORE DESC
```

```
GO
```

4. 使用聚集函数的查询

- (1) 查询计算机系学生总人数。

```
USE jxsk
```

```
SELECT COUNT(*) FROM SC
```

```
WHERE DEPT='计算机'
```

```
GO
```

- (2) 查询选修了微机原理课程的学生人数、平均成绩、最高成绩。

```
USE jxsk
```

```
SELECT COUNT(*) AS 人数, AVG(SCORE) AS 平均分数, MAX(SCORE) AS 最高分数
```

```
FROM C, SC
```

```
WHERE CN='微机原理' AND C.CNO=SC.CNO
```

```
GO
```

5. 对查询结果分组

- (1) 查询各个课程号及相应的选课人数。

```
USE jxsk
```

```
SELECT CNO AS 课程号, COUNT(SNO) AS 人数,
```

```
FROM SC
```

```
GROUP BY CNO
```

```
GO
```

- (2) 查询选修了 2 门以上课程的学生姓名和平均成绩。

```
USE jxsk
```

```
SELECT SN AS 姓名, AVG(SCORE) AS 平均成绩
```

```
FROM S, SC
```

```
WHERE S.SNO = SC.SNO
```

```
GROUP BY S.SN
```

```
HAVING COUNT(*)>2
```

```
GO
```

实验 7.2 连接查询

1. 连接查询

- (1) 查询所有选课学生的学号、姓名、选课名称及成绩。

```
USE jxsk
```

```
SELECT S.SNO, SN, CN, SCORE
```

```
FROM S, C, SC
```

```
WHERE S.SNO=SC.SNO AND C.CNO=SC.CNO
```

```
GO
```


- (2) 查询每门课程的课程号、任课教师姓名及其选课人数。

```
USE jxsk
SELEC  C.CNO,TN, COUNT(SC.SNO) AS 学生人数
FROM  T, TC, C, SC
WHERE T.TNO=TC.TNO AND C.CNO=TC.CNO AND  C.CNO=SC.CNO
GROUP BY C.CN,T.TN
GO
```

2. 自身连接

- (1) 查询所有比“刘伟”工资高的教师姓名、工资和刘伟的工资。

```
USE jxsk
SELECT  X.TN AS 姓名, X.SAL AS 教师工资,Y.SAL AS 刘伟工资
FROM  T AS X, T AS Y
WHERE X.SAL>Y.SAL AND Y.TN='刘伟'
GO
```

- (2) 查询同时选修了“程序设计”和“微机原理”的学生姓名、系名。

```
USE jxsk
SELECT  DISTINCT(SN) AS 姓名,C1.CN AS 课程名 1,C2.CN AS 课程名 2
FROM  C AS C1, C AS C2,SC AS SC1 ,SC AS SC2,S
WHERE  C1.CNO = SC1.CNO AND C2.CNO=SC2.CNO AND
        C1.CN='程序设计' AND C2.CN='微机原理' AND
        SC1.SNO=SC2.SNO AND SC1.SNO=S.SNO
GO
```

3. 外连接

查询所有学生的学号、姓名、选课名称及成绩（没有选课的同学的选课信息显示为空）

```
USE jxsk
SELECT  S.SNO, SN,CN,SCORE FROM  S
LEFT OUTER JOIN SC  ON S.SNO=SC.SNO
LEFT OUTER JOIN C  ON C.CNO=SC.CNO
GO
```

实验 7.3 嵌套查询

1. 返回一个值的子查询

- (1) 查询与“刘伟”教师职称相同的教师号、姓名和职称。

```
USE jxsk
SELECT TNO, TN, PROF FROM T
WHERE PROF= (SELECT PROF FROM T
              WHERE TN='刘伟' )
GO
```

2. 返回一组值的子查询

- (1) 使用 ANY 谓词查询讲授课程号为 C5 的教师姓名

```
USE jxsk
SELECT  TN FROM T
WHERE  (TNO=ANY (SELECT TNO FROM TC
                  WHERE CNO='C5' ))
```

- (2) 使用“IN”谓词查询讲授课程号为 C5 的教师姓名

```
USE jxsk
SELECT  TN FROM T
```

```
WHERE (TNO IN (SELECT TNO FROM TC WHERE CNO='C5'))
GO
```

- (3) 使用“ALL”谓词查询其他系中比计算机系所有教师工资都高的教师的姓名、工资和所在系。

```
USE jxsk
SELECT TN, SAL, DEPT FROM T
WHERE (SAL>ALL (SELECT SAL FROM T
                WHERE DEPT='计算机') AND (DEPT<>'计算机'))
GO
```

- (4) 使用“EXISTS”谓词查询没有讲授课程号为 C5 的教师姓名、所在系。

```
USE jxsk
SELECT TN, DEPT FROM T
WHERE (NOT EXISTS (SELECT * FROM TC
                  WHERE TNO=T.TNO AND CNO='C5'))
GO
```

- (5) 使用“NOT EXISTS”谓词查询至少选修了学生 S2 选修的全部课程的学生学号。

```
USE jxsk
SELECT DISTINCT SNO FROM SC SCX
WHERE NOT EXISTS (SELECT * FROM SC SCY WHERE SCY.SNO='S2' AND
                 NOT EXISTS (SELECT * FROM SC SCZ
                             WHERE CZ.SNO=SCX.SNO
                             AND SCZ.CNO=SCY.CNO))GO
GO
```

实验 7.4 集合查询

查询计算机系的学生及年龄不大于 19 岁的学生。

```
USE jxsk
SELECT * FROM S WHERE DEPT='计算机'
UNION
SELECT * FROM S WHERE AGE<=19
GO
```

实验 8 存储过程

实验 8.1 创建并执行存储过程

1. 用企业管理器创建并执行一存储过程

- (1) 创建存储过程 Pro_Qsinf: 通过学生学号来查询学生的姓名、年龄、系名。

```
USE jxsk
GO
Create Procedure Pro_Qsinf
@sno_in char(8)= 'S2', @sname_out char(8) output, @sage_out int output, @dept_out char(10) output
As select @sname_out=sn, @sage_out=age, @dept_out=dept
From S where sno=@sno_in
GO
```

- (2) 执行存储过程 Pro_Qsinf。查询并显示出默认学号（即 S2）和学号为 S4 学生的姓名和年龄。

```
USE jxsk
declare @sno_in char(8),
@sname_out char(8),
@sage_out int,
```

```

        @sdept_out char(10)
exec Pro_Qsinf default, @sname_out output, @sage_out output,
        @sdept_out output

print @sname_out
print @sage_out
print @sdept_out
select @sno_in='S4'
exec Pro_Qsinf @sno_in, @sname_out output, @sage_out output,
        @sdept_out output

print @sname_out
print @sage_out
print @sdept_out
go

```

2. 用 SQL 创建一存储过程

(1) 创建存储过程 **Pro_Qscore**：通过学生姓名和课程名查询该生该课程的成绩。

```

USE jxsk
go
CREATE Procedure Pro_Qscore
        @sname_in char(8), @cname_in char(10) ,
        @score_out tinyint output
As select @score_out=score from s, c, sc
        where s.sno=sc.sno and c.cno=sc.cno and sn=@sname_in and cn=@cname_in
go

```

(2) 执行存储过程 **Pro_Qscore**。查询并显示学生“李思”的“程序设计”课程的成绩。

```

USE jxsk
declare @sname_in char(8),
        @cname_in char(8),
        @score_out tinyint
select @sname_in='李思'
select @cname_in='程序设计'
exec Pro_Qscore @sname_in, @cname_in,
        @score_out output
print Rtrim ( @sname_in ) +'='+Ltrim( str(@score_out ) )
go

```

实验 8.2 修改存储过程

2. 用 SQL 修改存储过程 **Pro_Qsinf**

```

USE jxsk
GO
Alter Procedure Pro_Qsinf
        @sno_in char(2)='S1', @sname_out char(8) output,
        @ssex_out char(2) output, @dept_out char(10) output
As
        select @sname_out=sn, @ssex_out=sex , @dept_out=dept
        from s where sno=@sno_in
GO

```

实验 8.3 删除存储过程

2. 用 SQL 删除存储过程 Proc_Qscore

```
USE jxsk
Drop procedure Pro_Qscore
Go
```

实验 9 触发器

1. 用企业管理器为表 S 创建一级联更新触发器 TRIGGER_S。

```
CREATE TRIGGER TRIGGER_S
ON S
FOR UPDATE AS
IF UPDATE(SNO)
BEGIN
    DECLARE @SNO_NEW CHAR(2),@SNO_OLD CHAR(2)
    SELECT @SNO_NEW=SNO FROM INSERTED
    SELECT @SNO_OLD=SNO FROM DELETED
    UPDATE SC SET SNO=@SNO_NEW WHERE SNO=@SNO_OLD
END
```

2. 企业管理器为表 SC 创建一限制更新触发器 TRIGGER_SC

要求：若修改 SC 表中一记录的学号，则要检查表 S 中是否存在与该学号相同的记录，若有则不许修改，若没有则可修改。

(1) 创建触发器

```
CREATE TRIGGER TRIGGER_SC
ON SC
FOR UPDATE AS
IF UPDATE(SNO)
BEGIN
    DECLARE @SNO_NEW CHAR(2), @SNO_OLD CHAR(2),SNO_CNT INT
    SELECT @SNO_OLD=SNO FROM DELETED
    SELECT @SNO_CNT=COUNT(*) FROM S WHERE SNO=@SNO_OLD
    IF @SNO_CNT<>0
        ROLLBACK TRANSACTION
END
```

3. 用 SQL 为表 SC 创建一触发器 Score_sc_tri

(1) 要求：当插入一个记录或修改成绩时，确保此记录的成绩在 0~100 之间。

```
Create trigger Score_sc_tri
On SC For Insert,update
As declare @scroe_read tinyint
Select @scroe_read=scroe from inserted
If @scroe_read>=0 and @scroe_read<=100
begin
    Print ' 操作完成！'
    Return
End
Print ' 成绩超出 0~100 之间！请重新输入。'
Rollback Transaction

GO
```

(2) 验证触发器 Score_sc_tri 的作用

```
INSERT INTO SC VALUES('S1','C5',190)
GO
INSERT INTO SC VALUES('S1','C5',100)
GO
UPDATE sc SET Score=130 WHERE sno='S2' AND cno='C5'
GO
UPDATE sc SET Score=60 WHERE sno='S2' AND cno='C5'
GO
```

4. 用 SQL 为表 C 创建一个级联删除触发器 TRIGGER_DC：通过课程名从 C 表中删除某课程信息，同时删除 SC 表中与此课程相关的选课记录

```
USE jxsk
GO
Create trigger TRIGGER_DC
On SC For DELETE
As declare @CNO_DEL CHAR(2)
Select @CNO_DEL=CNO From DELETED
DELETE FROM SC WHERE CNO=@CNO_DEL
GO
```

(2) 验证触发器的作用。

```
USE jxsk
Delete from C where CNO='C1'
GO
```

实验 9.2 修改触发器

1. 用企业管理器修改表 S 的触发器 TRIGGER_S

```
Create trigger TRIGGER_S
On S For DELETE
As declare @SNO_DEL CHAR(2)
Select @SNO_DEL=SNO From DELETED
DELETE From SC Where SNO=@SNO_DEL
```

2. 用 SQL 修改表 C 的触发器 TRIGGER_DC

要求：通过课程名从 C 表中删除某课程信息，同时删除 SC 表和 TC 表中与此课程相关的记录。

```
USE jxsk
GO
Alter trigger TRIGGER_DC
On SC For DELETE
As declare @CNO_DEL CHAR(2)
Select @CNO_DEL=CNO from DELETED
DELETE from SC WHERE CNO=@CNO_DEL
DELETE from TC WHERE CNO=@CNO_DEL
GO
```

实验 9.3 删除触发器

2. 用 SQL 删除表 C 的触发器 TRIGGER_DC

```
USE jxsk
```

```
DROP TRIGGER TRIGGER_DC
GO
```

实验 10 T-SQL 程序设计

1. 计算 1~100 之间所有能被 3 整除的数的个数和总和

```
DECLARE @SUM SMALLINT,
        @I SMALLINT, @NUMS SMALLINT
SET @SUM=0
SET @I=1
SET @NUMS=0
WHILE (@I<=100)
BEGIN
    IF (@I%3=0)
    BEGIN
        SET @SUM=@SUM+@I
        SET @NUMS=@NUMS+1
    END
    SET @I=@I+1
END
PRINT '总和是: ' + STR( @SUM )
PRINT '个数是: ' + STR( @NUMS )
```

2. 从学生表中选取 SNO, SN, SEX, 如果为“男”则输出“M”, 如果为“女”则输出“F”

```
USE jxsk
SELECT SNO AS 学号, SN AS 姓名, 性别 =
    CASE SEX
        WHEN '男' THEN 'M'
        WHEN '女' THEN 'F'
    END
FROM _S
GO
```

实验 10.2 面向复杂 T-SQL 程序设计

1. 从教学数据库 jxsk 中查询所有同学选课成绩情况: 姓名、课程名、成绩。要求: 凡成绩为空者输出“未考”、小于 60 分的输出“不及格”; 60 分至 70 分的输出“及格”; 70 分至 80 分的输出“中”; 80 分至 90 分的输出“良好”; 90 分至 100 分的输出“优秀”。并且输出记录按下列要求排序: 先按 SNO 升序, 再按 CNO 号升序, 最后按成绩降序。

```
USE jxsk
SELECT SN AS 姓名,
       CN AS 课程名,
       成绩 =
       CASE
           WHEN SCORE IS NULL THEN '未考'
           WHEN SCORE<60 THEN '不及格'
           WHEN SCORE>=60 AND SCORE<70 THEN '及格'
           WHEN SCORE>=70 AND SCORE<80 THEN '中'
           WHEN SCORE>=80 AND SCORE<90 THEN '良好'
           WHEN SCORE>=90 THEN '优'
```

```

END
FROM SC, S, C
WHERE S.SNO=SC.SNO AND C.CNO=SC.CNO
ORDER BY S.SNO, C.CNO, SCORE DESC
GO

```

2. 现给教师增加工资的操作

要求：必须任 2 门以上课程且涨幅按总收入分成三个级别：4000 元以上涨 300；3000 元以上涨 200；3000 以下涨 100。只任 1 门课程的不涨。其他情况的不涨。

```

UPDATE t SET sal = sal +
CASE
    WHEN t.tno IN
        (SELECT TC.tno
         FROM T,TC
         WHERE T.tno = TC.tno AND (SAL+COMM) >=4000
         GROUP BY TC.TNO HAVING COUNT(cno)>=2) THEN 300
    WHEN T.tno IN
        (SELECT TC.tno
         FROM T,TC
         WHERE T.tno = TC.tno AND (SAL+COMM)>=3000 AND
        (SAL+COMM)<4000
         GROUP BY TC.Tno HAVING COUNT(cno)>=2) THEN 200
    WHEN T.Tno IN
        (SELECT Tc.Tno
         FROM T, TC
         WHERE T.Tno = TC.Tno AND (T.Sal+t.COMM<3000)
         GROUP BY TC.Tno HAVING COUNT(cno)>=2) THEN 100
    WHEN T.Tno in
        (SELECT TC.Tno
         FROM T,TC
         GROUP BY TC.Tno HAVING COUNT(*)=1) THEN 50
END

```

实验 11 用户定义数据类型与自定义函数

实验 11.1 创建和使用用户定义数据类型

1. 创建和使用一用户定义的数据类型 Idnum

(1) 用 SQL 语句创建一个用户定义的数据类型 Idnum。

```

USE jxsk
EXEC sp_addtype Idnum, 'char(6)', 'NOT NULL'
GO

```

(2) 使用用户定义的数据类型 Idnum，创建一个学生表 STUDENT 和一个教师表 TEACHER。

```

USE jxsk
CREATE TABLE STUDENT(
    SNO Idnum ,
    SN CHAR(11) ,
    SSEX CHAR(2),
    SAGE TINYINT )
CREATE TABLE TEACHER(

```

```

TNO Idnum ,
TN CHAR(11),
TSEX CHAR(2),
TAG TINYINT,
TPROF CHAR(11) )

```

(2) 使用用户定义的数据类型 **Nameperson**。修改学生表 **STUDENT** 中的姓名类型为 **Nameperson** 和教师表中的教师姓名类型为 **Nameperson**。

```

USE jxsk
ALTER TABLE STUDENT
    ALTER COLUMN SN Nameperson
ALTER TABLE TEACHER
    ALTER COLUMN TN Nameperson

```

实验 11.2 删除用户定义数据类型

1. 使用系统存储过程删除用户定义的数据类型 **Nameperson**

```

USE jxsk
ALTER TABLE student ALTER COLUMN SNO CHAR(6) NOT NULL
ALTER TABLE teacher ALTER COLUMN TNO CHAR(6) NOT NULL
EXEC Sp_droptype Nameperson
GO

```

实验 11.3 创建和使用用户定义的函数

③ 创建用户定义函数 **Score_FUN**。

```

CREATE FUNCTION Score_FUN( @SNAME_IN CHAR(11), @CNAME_IN CHAR(11) )
RETURNS TINYINT
AS
BEGIN
    DECLARE @SCORE_OUT TINYINT
    SELECT @SCORE_OUT=score FROM sc, s, c
        where s.sno=sc.sno and c.cno=sc.cno and
            Sn=@SNAME_IN and Cn=@CNAME_IN
    RETURN(@SCORE_OUT)
END

```

(2) 使用用户定义的函数 **Score_FUN**，查询学生钱尔的编译原理课程的成绩。

```

DECLARE @S_Score tinyint
USE jxsk
EXEC @S_Score=dbo.Score_FUN '钱尔', '编译原理'
Print ' 钱尔的编译原理成绩是 '+ STR( @S_Score )
GO

```

2. 用 SQL 创建一个内嵌函数

(1) 用 SQL 创建一个内嵌函数 **S_Score_FUN**

```

USE jxsk
GO
CREATE FUNCTION S_Score_FUN( @SNAME_IN CHAR(10) )
RETURNS TABLE
AS
    RETURN (SELECT CN, SCORE FROM SC, S, C
        Where S.Sno=SC.Sno and

```



```
S.cno=SC.Cno and Sn=@SNAME_IN )
```

```
GO
```

(2) 使用用户定义的函数 S_Score_FUN，查询学生钱尔所有课程的成绩。

```
USE jxsk
SELECT * FROM S_Score_FUN ( '钱尔' )
GO
```

3. 用 SQL 创建一个多语句函数

(1) 用 SQL 创建一个多语句函数 ALL_Score_FUN。

```
USE jxsk
GO
CREATE FUNCTION ALL_Score_FUN ( @CNAME_IN CHAR(10) )
RETURNS @ALL_SCORE_TAB TABLE ( SNO CHAR(6) PRIMARY KEY,
SN CHAR (10) NOT NULL ,
SEX CHAR (2),
SCORE TINYINT )
AS
BEGIN
    INSERT @ALL_SCORE_TAB
    SELECT S.SNO,SN,SEX,SCORE
    FROM SC, S, C
    WHERE S.Sno = SC.Sno and C.Cno=SC.Cno and Cn=@CNAME_IN
    RETURN
END
GO
```

(2) 使用用户定义的函数 ALL_Score_FUN，查询选择微机原理课程的学生的成绩。

```
USE jxsk
SELECT * FROM ALL_Score_FUN ( '微机原理' )
GO
```

实验 11.4 修改用户定义的函数

1. 用企业管理器修改函数

(1) 用企业管理器修改函数 Score_FUN

```
CREATE FUNCTION Score_FUN( @SNAME_IN CHAR(10),@CNAME_IN CHAR(10) )
RETURNS CHAR(8)
AS
BEGIN
    DECLARE @SCORE_OUT CHAR(8)
    SELECT @SCORE_OUT=
    CASE
    WHEN SCORE IS NULL THEN '未考'
    WHEN SCORE <60 THEN '不及格'
    WHEN SCORE >=60 AND SCORE <70 THEN '及格'
    WHEN SCORE >=70 AND SCORE <80 THEN '中'
    WHEN SCORE >=80 AND SCORE <90 THEN '良好'
    WHEN SCORE >= 90 THEN '优秀'
    END
    FROM sc, s, c
    WHERE s. sno=sc. sno and c. cno=sc. cno and Sn=@SNAME_IN and Cn=@CNAME_IN
```

```

        RETURN (@SCORE_OUT)
    END
(2) 使用用户定义的函数 Score_FUN，查询学生钱尔的编译原理课程的成绩
    DECLARE @S_Score CHAR(8)
    USE jxsk
    EXEC @S_Score=dbo.Score_FUN '钱尔','编译原理'
    Print ' 钱尔的编译原理成绩是 '+ @S_Score
    GO

```

2. 用 SQL 修改函数

(1) 用 SQL 修改函数 S_Score_FUN，要求增加一输出列对应成绩的等级。

```

USE jxsk
GO
ALTER FUNCTION S_Score_FUN( @SNAME_IN CHAR(10) )
RETURNS TABLE
AS
    RETURN (SELECT CN, SCORE,
        LEVER=
            CASE
                WHEN SCORE IS NULL THEN '未考'
                WHEN SCORE <60 THEN '不及格'
                WHEN SCORE >=60 AND SCORE <70 THEN '及格'
                WHEN SCORE >=70 AND SCORE <80 THEN '中'
                WHEN SCORE >=80 AND SCORE <90 THEN '良好'
                WHEN SCORE >= 90 THEN '优秀'
            END
        FROM SC, S, C
        WHERE S.Sno=SC.Sno and C.cno=SC.Cno and Sn=@SNAME_IN )
GO

```

(2) 使用用户定义的函数 S_Score_FUN，查询学生钱尔所有课程的成绩。

```

USE jxsk
SELECT * FROM S_Score_FUN ( '钱尔' )
GO

```

实验 11.5 删除用户定义的函数

2. 用 SQL 删除函数 S_Score_FUN

```

USE jxsk
DROP FUNCTION S_Score_FUN
GO

```

实验 12.4 对象级许可权限管理

2. 授予用户 SQLUser 对数据库 jiaoxuedb 表 Student 的 INSERT 权限；废除对表 Student 的 UPDATE 权限

```

USE jiaoxuedb
GO
GRANT INSERT ON Student TO SQLUser
REVOKE UPDATE ON Student TO SQLUser

```

GO

实验 13 SQL Server 事务设计

实验 13.1 设计并执行事务

1. 设计并执行事务 1

将学生“陈东辉”的“计算机基础”课程成绩改为 77 分。

```
BEGIN TRANSACTION
USE jiaoxuedb
GO
UPDATE SC SET score=77 WHERE Sno IN (SELECT SNO FROM Student
    WHERE Sname='陈东辉') AND Cno IN (SELECT CNO FROM Course
    WHERE Cname='计算机基础')
GO
COMMIT
GO
```

2. 设计事务 2

将课程“数据结构”的课号与“微机原理”的课号互换。

```
BEGIN TRANSACTION
USE jiaoxuedb
GO
Declare @cno1 char(5),@cno2 char(5)
Select @cno1=cno From Course Where Cname='数据结构'
Select @cno2=cno From Course Where Cname='微机原理'
UPDATE Course SET Cno=@cno1 WHERE Cname='微机原理'
UPDATE Course SET Cno=@cno2 WHERE Cname='数据结构'
GO
COMMIT
GO
```

3. 设计事务 3

教师“许永军”退休，由他讲授的 2 门课程中，课程“微机原理”转给教师“张朋”讲授；“数据库”转给“李英”讲授。

```
BEGIN TRANSACTION
USE jiaoxuedb
GO
Declare @cno char(5),@tno1 char(6),@tno2 char(6)
Select @tno1=Tno From Teacher Where Tname='许永军'
Select @tno2=Tno From Teacher Where Tname='张朋'
Select @cno=Cno From Course Where Cname='微机原理'
UPDATE TC SET Tno=@tno2 WHERE Tno=@tno1 AND Cno=@cno
Select @tno1=Tno From Teacher Where Tname='许永军'
Select @tno2=Tno From Teacher Where Tname='李英'
Select @cno=Cno From Course Where Cname='数据库' UPDATE TC SET Tno=@tno2 WHERE Tno=@tno1 AND
Cno=@cno
COMMIT
GO
```

实验 13.2 设计复杂事务

1. 设计并执行事务 1

学生“王一山”打算选修“计算机网络”课程，根据要求，此门课程选修的人数最多为 30 人，该生是否可以选修此门课程，给出结果提示。

```
BEGIN TRANSACTION
USE jiaoxuedb
GO
Declare @person_num tinyint, @cno char(5), @sno char(6)
Select @cno=cno From Course Where Cname=' 计算机网络'
Select @sno=sno From Student Where Sname=' 王一山'
Select @person_num = count(*) From SC Where Cno=@cno
If @person_num<30
Begin
    Insert into SC(Sno, Cno) Values (@sno, @cno)
    Commit /* 提交事务*/
    Print ' 王一山同学选修计算机网络课程注册成功!'
End
Else
Begin
    Rollback Transaction /* 回滚事务*/
    Print ' 选修计算机网络课程的人数已满，王一山同学不能再选修此课程!'
End
GO
```

带格式的：缩进：左 4 字符

实验 14 数据库备份和恢复

实验 14.1 完全数据库备份与简单恢复

2. 使用 T-SQL 执行完全数据库备份及其简单恢复

(1) 操作 1：对现有数据库执行完全备份 Fullbackup_jiaoxuedb2

```
USE jiaoxuedb
GO
BACKUP DATABASE jiaoxuedb
TO DISK='D:\备份数据库\Fullbackup_jiaoxuedb2'
WITH INIT
GO
```

(2) 操作 2：将张彬同学的名字改为：“张斌”

```
USE jiaoxuedb
UPDATE Student SET Sname=' 张斌 'WHERE Sname=' 张彬'
GO
```

(3) 操作 3：执行恢复，将数据库恢复到操作 2 之前的状态。

```
USE master
GO
RESTORE DATABASE jiaoxuedb
FROM DISK='D:\备份数据库\Fullbackup_jiaoxuedb2'
```

```
WITH RECOVERY
GO
```

实验 14.2 差异数据库备份与简单恢复

2. 使用 T-SQL 执行数据库差异备份及其恢复

(1) 操作 1: 创建数据库 jiaoxuedb 一个完全数据库备份 Fbackup_jiaoxuedb2。

```
USE jiaoxuedb
GO
BACKUP DATABASE jiaoxuedb
TO DISK='D:\备份数据库\Fbackup_jiaoxuedb2'
WITH INIT
GO
```

(2) 操作 2: 把表 SC 中学号为 001201 课程号为 02002 的记录删除。

```
USE jiaoxuedb
GO
DELETE FROM SC WHERE Sno='001201' and Cno='02002'
GO
```

(3) 操作 3: 进行差异备份当前数据库 Dbackup_jiaoxuedb2。

```
USE jiaoxuedb
GO
BACKUP DATABASE jiaoxuedb
TO DISK='D:\备份数据库\Dbackup_jiaoxuedb2'
WITH DIFFERENTIAL
GO
```

(4) 操作 4: 向表 SC 中插入记录: 学号: 001201、课号: 02002、成绩: 79

```
USE jiaoxuedb
GO
INSERT INTO SC VALUES('001201', '02002', 79)
GO
```

(5) 操作 5: 把数据库恢复到操作 2 完成后的状态。

```
USE master
GO
RESTORE DATABASE jiaoxuedb
FROM DISK='D:\备份数据库\Fbackup_jiaoxuedb2'
WITH NORECOVERY
GO
RESTORE DATABASE jiaoxuedb
FROM DISK='D:\备份数据库\Dbackup_jiaoxuedb2'
WITH FILE=2
GO
```

实验 15 数据的导入、导出

实验 15.2 bcp 实用程序

1. 使用 bcp 从源数据表导出数据至 Excel 表

```
bcp "SELECT * From jiaoxuedb.dbo.Teacher Order by Tno"
```

```
queryout e:\张小山数据库\teacher_Excel.xls -c -SMXM -Usa -Psa
```

2. 使用 bcp 从源[数据库表](#)导出数据至 txt 格式的文件

```
bcp "Select * From jiaoxuedb.dbo.Student"
```

```
queryout e:\张小山数据库\Student_txt.txt -T -c -SMXM\SQLEXPRESS
```

3. 使用 bcp 从 Excel 文件导入数据到[数据库表](#)

```
bcp jiaoxuedb.dbo.Course
```

```
in e:\张小山数据库\Course_Excel.xls -T -c -SMXM\SQLEXPRESS
```

实验 16 SQL Server 中对大对象数据的访问

实验 16.1 用普通方法访问大值类型数据

- (2) 设置 large value types out of row 选项为 OFF，在数据行中存储大值数据类型数据项。

```
sp_tableoption N'MyTable', 'large value types out of row', 'OFF'
```

- (4) 查询 resume 数据。

```
Select Sn, resume From Teacher Where Sn='王一凡'
```

实验 16.2 访问大值数据类型

- (1) 开启 large value types out of row 选项。

```
use jiaoxuedb
```

```
Go
```

```
sp_tableoption 'Teacher', 'large value types out of row', 'ON'
```

```
Go
```

- (2) 给 resume 插入数据。

为教师“许红霞”写入简历数据。

```
use jiaoxuedb
```

```
Go
```

```
/* 给许红霞的简历resume列输入值*/
```

```
UPDATE teacher
```

```
SET resume .WRITE ( '1993年月毕业于东北财经学院金融系，1993年7月任教于东北财经大学。
```

```
1995年9月就读于中国财经学院研究生，1998年10月获得硕士学位。', 0, null )
```

```
WHERE Tname = '许红霞'
```

```
Go
```

```
/*查询刚插入的resume值*/
```

```
Select resume From teacher Where Tname = '许红霞'
```

```
Go
```

- (3) 读取 resume 列数据。

```
Use jiaoxuedb
```

```
Go
```

```
select resume from teacher where Tname = '许红霞'  
Go
```

- (4) 修改 resume 列数据。

```
/*将数据列resume中的1995改为1996*/  
UPDATE teacher  
SET resume .WRITE ( '6', 39, 1 )  
WHERE Tname = '许红霞'  
GO  
/*将数据列resume中的1998改为2000*/  
UPDATE teacher  
SET resume .WRITE ( '2000', 56, 4 )  
WHERE Tname = '许红霞'  
GO  
/*查询刚修改的resume值*/  
Select resume From teacher Where Tname = '许红霞'  
Go
```

- (5) 在 resume 列追加数据。

```
use jiaoxuedb  
GO  
/*在 resume 列结尾追加数据 by setting @Offset to NULL.*/  
UPDATE teacher  
SET resume .WRITE ( '2005年获北京大学博士学位。', null, 0 ) WHERE Tname = '许红霞'  
,  
Go  
select resume from teacher where Tname = '许红霞'  
Go
```

- (6) 删除 resume 列部分数据。

```
use jiaoxuedb  
GO  
/*删除resume列部分数据*/  
UPDATE teacher  
SET resume .WRITE ( '', 19, 17 ) WHERE Tname = '许红霞'  
go  
/*查询已部分删除的resume值*/  
select resume from teacher where Tname = '许红霞'  
go
```

- (7) 删除 resume 列全部数据。

```
use jiaoxuedb  
GO  
/*删除resume所有数据*/  
UPDATE teacher  
SET resume .WRITE ( null, 0, 0 ) WHERE Tname = '许红霞'  
Go  
/*查询已全部删除的resume值*/
```

```
Select resume From teacher Where Tname = '许红霞'
Go
```

实验 17 在 VB 中采用 ADO 方法访问 SQL Server

实验 18 用 ASP 动态页面发布数据

1. ASP 程序设计

(1) 设计 global.asa 程序。

```
<!--Visual InterDev Generated - startspan===>
<!--METADATA TYPE="TypeLib" NAME="Microsoft ActiveX Data Objects 2.6 Library"
UUID="{00000206-0000-0010-8000-00AA006D2EA4}" VERSION="2.6"-->
<!--Visual InterDev Generated - endspring===>
<SCRIPT LANGUAGE=VBScript RUNAT=Server>
Sub Application_OnStart
End Sub

sub session_onstart
    set cn =server.CreateObject("adodb.connection") ‘创建一个连接数据库对象
    cn.Provider="MSDASQL"
    cn.Open "jiaoxuedb_odbc","sa","sa"
    session("cnn")=cn ‘保存连接对象变量
end sub

sub Session_OnEnd
    set cn=nothing ‘释放连接对象变量
end sub
</SCRIPT>
```

(2) 设计主页 Default.asp

```
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>
<SCRIPT ID=serverEventHandlersVBS LANGUAGE=vbscript>
<!--
function checkinput
    id=msgbox("真的要删除选中的记录吗?",vbYesNo)
    if id=6 then
        document.removeform.action="deletepage.asp"
        document.removeform.submit
    else
        document.removeform.focus
    end if
end function
function addnew_onclick()
```


[illegible]

[illegible]

(3) 设计修改记录页 Updatepage.asp

```
<% @ Language=VBScript %>
<HTML>
<HEAD>
<META name=VI60_defaultClientScript content=VBScript>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>
<% if request.querystring("update")="0" then
    Sno=Request.querystring("Sno")
    Cname=Request.querystring("Cname")
    set rs=server.CreateObject("ADODB.Recordset")
    strSQL="select Student.Sno,Sname,Cname,Score,Course.Cno " & _
        " From Student,Course,Sc " & _
        " Where Student.Sno=SC.Sno and Course.Cno=SC.Cno and " & _
        " Student.Sno='" & Sno & "' and Cname='" & Cname & "'"
    cn = session("cmn")
    rs.Open strSQL,cn
%>
<!--显示信息-->
<div align="center">
<FORM name=updateform id=form1 action="UpdatePage.asp?Sno=<%=Sno%>&amp;
    update=1" method="post" >
<center>
    <table border=0 cellpadding=1 cellspacing=1 width="40%" >&nbsp;
    <tbody>
    <tr bgcolor="#fbfce2">
        <td align=right width="25%"> 学号 </td>&nbsp;
        <td width=50%><p align="center"><%=rs.Fields(0).Value%></p></td></tr>
        <td align=right width="25%"> 姓名 </td>
        <td width=50%><p align="center"><%=rs.Fields(1).Value%></p></td></tr>
        <td align=right width="25%"> 课程名 </td>
        <td width=50%><p align="center"><%=rs.Fields(2).Value%></p></td></tr>
        <td align=right width="25%">成绩</td><td width=50%><p align="center">
            <input name="score" value="<%=rs.Fields(3).Value%>"></td></tr>
    </tbody></table>
```

```

<p><input type="submit" name="UpdatePage" value=" 修 改  "></p>
<%else
    '更新数据
    Sno=Request.Form("Sno")
    Cname=Request.Form("Cname")
    Score=Request.Form("Score")
    set rs=server.CreateObject("ADODB.Recordset")
    strSQL="select Cno From Course " & " Where Cname='" & cname & "'"
    cn = session("cnn")
    rs.Open strSQL,cn
    Cno=rs.fields(0)
    strSQL="update SC set Score='" & Score & "' where sno='" & _
        sno & "' and Cno='" & Cno & "'"
    cn=session("cnn")
    set cmd =server.CreateObject("adodb.command")
    cmd.CommandText=strSQL
    cmd.ActiveConnection=cn
    cmd.Execute nrecordsaffected
%> <script language=vbscript>
    document.location.href="default.asp"
</script>
<% end if %>
</FORM>
</BODY>
</HTML>

```

(4) 设计增加新记录页 Addpage.asp

```

<%@ Language=VBScript %>
<HTML>
<HEAD>
<META name=VI60_defaultClientScript content=VBScript>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
<SCRIPT LANGUAGE=vbscript>
</SCRIPT>
</HEAD>
<BODY>
<FORM name=addform action="addnewpage.asp" method="post" >
<table align=center border=0 width="40%" >
<tr bgcolor="yellow">
<td align=middle > 姓名 <br>
<INPUT name=Sname ID=SName size=15%></td>
<td align=middle > 课程名 <br>
<INPUT name=Cname ID=CName size=25%></td>
<td align=middle > 成绩 <br>
<INPUT name=Score ID=Score size=15%></td></tr>
<tr>
<td> </td>
<td align=middle><br><INPUT name=add type=submit value=" 提交  "></td>
</tr>

```

```

        </table>
    </FORM>
</BODY>
</HTML>

```

(5) 设计完成新记录入库的程序页 AddNewpage.asp

```

<% @ Language=VBScript %>
<%=Request.form("Sno")%>
<%=Request.form("Sname")%>
<%=Request.form("Cname")%>
<%=Request.form("Score")%>
<%  cn = session("cnn")
'set cn = server.CreateObject("adodb.connection")
'cn.open "newbase_source","sa","sa"
Sname=request.form("Sname")
Cname=request.form("Cname")
Score=request.form("Score")
set rs=server.createobject("adodb.recordset")
strsql="select * from Student,SC,Course where SC.Cno=Course.Cno and " + _
        " Student.Sno=SC.Sno and Sname=" & Sname & _
        "" and Cname=" & Cname & ""
rs.open strsql,cn
if rs.eof = false then  %>
    <script language=vbscript>
        document.location.href="companyadd.asp"
        msgbox( "该生已选该课程！ " )
    </script>
<%  rs.close
set rs=nothing
else
    rs.close
    strsql="select Cno from Course where Cname=" & Cname & ""
    rs.open strsql,cn
    if rs.eof = true then  %>
        <script language=vbscript>
            document.location.href="addpage.asp"
            msgbox( "没有该课程！ " )
        </script>
<%  rs.close
else
    Cno=rs.fields(0)
    rs.close
    strsql="select Sno from Student where Sname=" & Sname & ""
    rs.open strsql,cn
    if rs.eof = true then  %>
        <script language=vbscript>
            document.location.href="addpage.asp"
            msgbox( "没有该学生！ " )
        </script>

```

```

<%      rs.close
else
    Sno=rs.fields(0)
    rs.close
    set rs=nothing
    if Score="" then
        strsql="insert into SC(Sno,Cno) values('" & _
            Sno & "','" & Cno & "')"
    else
        strsql="insert into SC(Sno,Cno,Score) values('" & _
            Sno & "','" & Cno & "','" & Score & "')"
    end if
    set cmd = server.CreateObject("adodb.command")
    cmd.commandtext=strsql
    cmd.activeconnection=cn
    cmd.execute recordsaffected
    response.redirect("default.asp")
end if
end if
end if %>

```

(6) 设计一个删除选择记录的 ASP 程序: Deletepage.asp.

```

<%@ Language=VBScript %>
<%  if Request.Form("Sno").Count <> 0 then
    strsql = "delete SC where "
    for i =1 to Request.Form("Sno").Count
        Sno=Left(Request.Form("Sno")(i),6)
        cname=right(Request.Form("Sno")(i),len(Request.Form("Sno")(i))-6)
        strsql = strsql & "(Sno='" & Sno & "' and Cno in " & _
            "(select Cno from course where cname='" & Cname & "'))"
        if i <> Request.Form("Sno").Count then
            strsql = strsql & " or "
        end if
    next
    myconn = session("cnn")
    set cmd = server.CreateObject("adodb.command")
    cmd.CommandText = strsql
    cmd.ActiveConnection = myconn
    cmd.Execute recordsaffected
end if
Response.redirect ("default.asp") %>

```

第 19 章 采用 ADO.NET 访问 SQL Server

实验 19.1 查询数据库

1. 通过 SqlCommand 对象实现单值查询。根据学生姓名和课程名查询成绩。查询学生 “王一山” 的 “数据库” 课程的成绩。

④ 设计查询页面 Default.aspx。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
    <head runat="server">
        <title>查询成绩</title>
    </head>
    <body>
        <form id="form1" runat="server" >
            <div >&nbsp;</div>
            <table align="center" style="width: 387px; height: 34px;">
                <tr align="center" valign="middle"><td>
                    <asp:Label ID="Label1" runat="server" Text="查 询 成 绩"
                        Font-Bold="True" Font-Names="黑体" Font-Size="X-Large" ForeColor="Red">
                    </asp:Label></td></tr>
                <tr>
                </tr>
            </table>
            <table align="center" style="width: 387px;">
                <tr>
                    <td style="width: 81px" align="right">
                        <asp:Label ID="Label2" runat="server" Text="姓 名" Width="61px"
                            ForeColor="Blue"></asp:Label></td>
                    <td style="width: 158px">
                        <asp:TextBox ID="TBSname" runat="server" Width="290px"></asp:TextBox></td>
                </tr>
                <tr>
                    <td align="right" style="width: 81px">
                        <asp:Label ID="Label3" runat="server" Text="课程名"
                            ForeColor="Blue"></asp:Label></td>
                    <td style="width: 158px">
                        <asp:TextBox ID="TBCname" runat="server" Width="291px"></asp:TextBox></td>
                </tr>
                <tr>
                    <td align="right" style="width: 81px">
                        <asp:Label ID="Label4" runat="server" Text="成 绩" Width="53px"
                            ForeColor="Blue"></asp:Label></td>
                    <td style="width: 158px">
                        <asp:Label ID="LaScore" runat="server" Width="297px" BackColor="#C0FFFF"
                            ForeColor="Fuchsia"></asp:Label></td>
                </tr>
            </table>
            <table align="center" style="width: 387px;">
                <tr align="center"><td style="height: 36px">
                    <asp:Button ID="BTQuery" runat="server" Text="查 询" OnClick="BTQuery_Click" /></td>
                </tr>
            </table>
```

```
</form>
</body>
</html>
```

⑤ 设计 Default.aspx 页面处理代码程序 Default.aspx.cs。

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void BTQuery_Click(object sender, EventArgs e)
    {
    }
}
```

⑥ 编写Default.aspx.cs代码。

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void BTQuery_Click(object sender, EventArgs e)
    {
        string connString;
        string SName, CName;

        //获取姓名和课程名
        SName=TBSname.Text;
        CName=TBCname.Text;
        //设置数据库连接串，使用系统认证
```

```

connString = "Initial Catalog=jiaoxuedb;Data Source=mxm;Integrated Security=SSPI;";
SqlConnection Conn = new SqlConnection(connString);
SqlCommand QueryCommand = new SqlCommand("SELECT Score FROM Student,Course,SC " +
    "WHERE Student.Sno =Sc.Sno and Course.Cno=SC.Cno and " +
    "Sname=@SName and Cname = @CName", Conn);
    // Add the parameters for the SelectCommand.
QueryCommand.Parameters.Add("@SName", SqlDbType.Char, 8);
QueryCommand.Parameters.Add("@CName", SqlDbType.Char, 20);
QueryCommand.Parameters["@SName"].Value = SName;
QueryCommand.Parameters["@CName"].Value = CName;
Conn.Open();
//执行QueryCommand.ExecuteScalar方法查询成绩
try
{
    Int32 ScoreValue = (Int32)QueryCommand.ExecuteScalar();
    LaScore.Text =Convert.ToString(ScoreValue);
}
catch//错误处理
{
    LaScore.Text="没有成绩";
}
finally
{
    Conn.Close();
}
}
}

```

2. 通过创建 `SqlDataAdapter` 对象和 `DataTable` 对象实现多值查询。根据学生姓名查询该生所选课程。分别查询学生“张建国”所选的课程。

④ 设计查询页面 `Default.aspx`。

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
<title>多值查询</title>
</head>
<body>
<form id="form1" runat="server">
<div>
    &nbsp;  <table align=center style="width: 323px; text-align: center">
        <tr>
            <td style="width: 321px; height: 31px; text-align: center; color: red;"
                colspan="3" bgcolor="#ffffcc">查询学生选课情况</td>
        </tr></table>
    </div><table align=center style="text-align: center">
        <tr>
            <td style="width: 75px; height: 21px; text-align: right; color: blue;">
                姓名</td>

```



```

        <td style="width: 39px; height: 21px;" colspan="2">
            <asp:TextBox ID="TBSName" runat="server" Width="132px">
            </asp:TextBox></td>
        <td colspan="1" style="width: 95px; height: 21px">
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
                Text="查 询" /></td>
        </tr>
    </table>
    <table align=center style="width:323px">
    <tr><td style="height: 33px; text-align: center; color: blue; width: 324px;
        " colspan="3">
        <asp:Label ID="CourseID" runat="server" Text="课 程 名" BorderColor="DarkGray"
            BorderStyle="Double" Font-Size="Larger" ForeColor="Red" Height="29px"
            Width="314px"></asp:Label></td></tr>
    </table>
    <table align=center style="width: 323px">
    <tr><td style="width: 321px; height: 1px; text-align: center;" colspan="3">
        <asp:DataList ID="CourseList" runat="server" Width="314px" BorderStyle="Double"
            Height="30px" >
            <ItemTemplate>
                <asp:Label ID="CouresLabel" runat="server"
                    Text='<%# DataBinder.Eval(Container.DataItem, "Cname") %>'></asp:Label>
            </ItemTemplate>
            <SeparatorTemplate>
                -----
            </SeparatorTemplate>
        </asp:DataList></td>
    </tr>
    </table>
</form>
</body>
</html>

```

⑥ 编写Default.aspx.cs代码。

```

using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        CourseID.Visible = false;
    }
}

```

```

}
protected void QueryBT_Click(object sender, EventArgs e)
{
    string connString, Sname;
    CourseID.Visible = false;
    //设置数据库连接串, 使用SQL Server认证
    connString = "Persist Security Info=False;User ID=sa;Password=sa;Initial
        Catalog=jiaoxuedb;Server=mxm";
    Sname = TBSName.Text;
    // TODO: 在此处添加构造函数逻辑
    SqlConnection Conn = new SqlConnection(connString);
    //创建SqlCommand对象的命令对象: CourseCmd
    SqlCommand CourseCmd = new SqlCommand("SELECT Cname " +
        "FROM Student, SC, Course Where Student.sno=SC.sno and SC.cno=Course.cno" +
        " and Sname=@Sname", Conn);
    CourseCmd.Parameters.Add("@Sname", SqlDbType.Char, 8);
    CourseCmd.Parameters["@Sname"].Value = Sname;
    SqlDataAdapter CourseAdapt = new SqlDataAdapter();
    CourseAdapt.SelectCommand = CourseCmd;
    //创建DataSet对象, 并将其绑定到Datalist控件
    DataSet ds=new DataSet();
    try
    {
        Conn.Open();
        CourseAdapt.Fill(ds, "CourseTab");
        //创建一个数据表对象CourseTable
        DataTable CourseTable = ds.Tables["CourseTab"];
        //检索数据表中的行集合
        DataRow[] rows = CourseTable.Select();
        //是否有行数据返回.
        if(rows.Length != 0)//有行数据返回, 则用DataList显示数据行
        {
            CourseList.DataSource = CourseTable;
            CourseID.Text = "课 程 表";
            CourseID.Visible = true;
            CourseList.Visible = true;
            CourseList.DataBind();
        }
        else//没有行数据返回, 则显示: 抱歉, 没有此学生的选课信息!
        {
            CourseID.Text = "抱歉, 没有此学生的选课信息! ";
            CourseID.Visible = true;
            CourseList.Visible = false;
        }
    }
    catch
    {
        CourseID.Text = "抱歉, 没有此学生的选课信息! ";
        CourseID.Visible = true;
    }
    finally
        Conn.Close();
}

```



```

<td style="height: 2px; colspan="2">
    <asp:ListBox ID="NameLBox" runat="server" Style="border-right: black thin solid;
        border-top: black thin solid; border-left: black thin solid; border-bottom:
            black thin solid" Width="270px"></asp:ListBox></td>
<td style="width: 119px; height: 2px;"></td>
</tr>
<tr>
<td style="width: 55px; height: 28px;">
    <asp:Label ID="SumStudentLB" runat="server" BorderStyle="Solid" Font-Bold="True"
        Font-Size="20pt" ForeColor="#FF0033" Height="37px" Style="border-right: black 1px
            solid;border-top: black 1px solid; border-left: black 1px solid; border-bottom: black
                1px solid" Text="人 数" width="120px">
    </asp:Label></td>
<td style="width: 40px; height: 28px">
    <asp:TextBox ID="SumStudentTB" runat="server" BorderStyle="Solid" Height="35px"
        Style="border-right: black 1px solid;border-top: black 1px solid;
            border-left: black 1px solid; border-bottom: black 1px solid"
            Width="143px"></asp:TextBox></td>
<td style="width: 119px; height: 28px;"></td>
</tr>
</table>
</div>
</form>
</body>
</html>

```

⑥ 编写Default.aspx.cs代码。

```

using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        NameLBox.Visible = false;
        SumStudentLB.Visible = false;
        SumStudentTB.Visible = false;
        NameLBox.Visible = false;
        Lab2.Visible = false;
        //清除列表框内容
        for (int i = 0; i < NameLBox.Items.Count;i++)
        {
            NameLBox.Items[i].Selected=true;

```

```

        NameLBox.Items.Remove(NameLBox.SelectedItem);
    }
}

protected void QueryBT_Click(object sender, EventArgs e)
{
    string Sname="";
    string connString="";

    //创建与数据库连接的对象：使用系统认证，从Web.Config中读取数据库连接串
    ConnectionStringSettings NameSettings;
    NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];
    if (NameSettings != null)
        connString = NameSettings.ConnectionString;//从Web.Config中获取连接串
    SqlConnection conn = new SqlConnection(connString);
    //创建生成DataReader对象的命令对象：CmdMsg
    SqlCommand CmdMsg = new SqlCommand("Select Sname From Student Where Sname like @Sname",
        conn);
    Sname = TBName.Text;
    CmdMsg.Parameters.AddWithValue("@Sname", Sname+"%");
    //创建DataReader对象
    SqlDataReader dr;
    try
    {
        conn.Open();
        dr = CmdMsg.ExecuteReader();
        int Sum = 0;
        while (dr.Read())
        {
            ++Sum;
            NameLBox.Items.Add(dr["Sname"]+"");
        }
        if (Sum == 0)
        {
            NameLBox.Visible = false;
            SumStudentLB.Visible = false;
            SumStudentTB.Visible = false;
            NameLBox.Visible = false;
            Lab2.Text = "没有该姓氏的学生!";//错误显示
            Lab2.Visible = true;
        }
        else
        {
            Lab2.Text = "  名  单  ";
            NameLBox.Visible = true;
            SumStudentLB.Visible = true;
            SumStudentTB.Visible = true;
            NameLBox.Visible = true;
            Lab2.Visible = true;
            SumStudentTB.Text = Sum.ToString();
            SumStudentTB.ReadOnly = true;
        }
    }
    catch

```

```

    {
        Lab2.Text = "系统错误：可能是连接数据库出错！";//错误显示
        Lab2.Visible = true;
    }
    Finally
    {
        dr.Close();
        conn.Close();
    }
}
}

```

⑦ 配置 Web.Config 文件。将数据库连接串代码插入 Web.Config 中，如下代码所示：

```

<?xml version="1.0"?>
<configuration>
    <appSettings></appSettings>
    <connectionStrings>
        <add name="SqlConnStr"
            connectionString="Database=jiaoxuedb;server=mxm;
                                Persist Security Info=false;Integrated Security=SSPI;"
            providerName="System.Data.SqlClient" />
    </connectionStrings>
    <system.web>
        <compilation debug="true"/>
        <authentication mode="Windows"/>
        <!--
            <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
                <error statusCode="403" redirect="NoAccess.htm" />
                <error statusCode="404" redirect="FileNotFound.htm" />
            </customErrors>
        -->
    </system.web>
</configuration>

```

实验 19.2 插入数据至数据库

1. 通过 SqlCommand 对象实现把数据插入数据库表中。插入一条记录：把一学生记录插入学生表中，学号：990109，姓名：钱力，性别：男。

④ 设计插入页面 Default.aspx。

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
    <head runat="server">
        <title>插入学生记录</title>
    </head>
    <body>
        <form id="form1" runat="server">
            <div>
                <table align=center style="vertical-align:
                    baseline; text-align: center">
                    <tr><td style="height: 33px;" colspan="2">

```



```

        SnoTB.Focus();
    }
    protected void InsertBT_Click(object sender, EventArgs e)
    {
        string connString = "";
        //创建与数据库连接的对象：使用系统认证，从Web.Config中读取数据库连接串
       ConnectionStringSettings NameSettings;
        NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];
        if (NameSettings != null)
            connString = NameSettings.ConnectionString;//从Web.Config中获取连接串
        SqlConnection conn = new SqlConnection(connString);
        //创建生成SqlCommand对象的命令对象：CmdMsg
        string SnoStr = SnoTB.Text;
        string SnameStr = SnameTB.Text;
        string SexStr = SexTB.Text;
        string StrCommand = "Insert Student (Sno, Sname, Sex) Values (@SnoStr, @SnameStr, @SexStr)";
        SqlCommand CmdMsg = new SqlCommand(StrCommand, conn);
        CmdMsg.Parameters.Add("@SnoStr", SqlDbType.Char, 6);
        CmdMsg.Parameters["@SnoStr"].Value=SnoStr;
        CmdMsg.Parameters.Add("@SnameStr", SqlDbType.Char, 8);
        CmdMsg.Parameters["@SnameStr"].Value=SnameStr;
        CmdMsg.Parameters.Add("@SexStr", SqlDbType.Char, 2);
        CmdMsg.Parameters["@SexStr"].Value=SexStr;
        //执行SqlCommand对象CmdMsg.
        try
        {
            conn.Open();
            int RecordsAffected = CmdMsg.ExecuteNonQuery();
            if (RecordsAffected == 1)
                MessageLB.Text = "插入成功! ";
            MessageLB.Visible = true;
        }
        catch
        {
            MessageLB.Text = "插入失败! 系统错误! ";//错误显示
            MessageLB.Visible = true;
        }
        finally
        {
            conn.Close();
        }
    }
}

```

⑦ 配置 Web.Config 文件。将数据库连接串代码插入 Web.Config 中，如图下所示：

```

<?xml version="1.0"?>
<configuration>
    <appSettings></appSettings>
    <connectionStrings>
        <add name="SqlConnStr"
            connectionString= "Database=jiaoxuedb;server=mxm;
                                Persist Security Info=false;Integrated Security=SSPI;"

```


2. 通过 `SqlDataAdapter` 对象把数据插入数据库表中。插入一条记录：把一课程记录插入课程表中，课程号：04001，课程名：信息存储与检索。

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>

④ 设计 Default.aspx 页面处理代码程序 Default.aspx.cs。

```
using System;  
using System.Data;  
using System.Data.SqlClient;  
using System.Configuration;
```

```

using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        CnoTB.Focus();
    }
    protected void InsertBT_Click(object sender, EventArgs e)
    {
        string connString = "";
        //创建与数据库连接的对象：使用系统认证，从Web.Config中读取数据库连接串
        ConnectionStringSettings NameSettings;
        NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];
        if (NameSettings != null)
            connString = NameSettings.ConnectionString; //从Web.Config中获取连接串
        SqlConnection conn = new SqlConnection(connString);
        string CnoStr = CnoTB.Text;
        string CnameStr = CnameTB.Text;
        //创建SqlDataAdapter对象，生成DataTable对象
        try
        {
            conn.Open();
            SqlDataAdapter Adapter = new SqlDataAdapter("select * From Course", conn);
            DataSet ds = new DataSet(); //创建数据集
            SqlCommandBuilder dbCB = new SqlCommandBuilder(Adapter);
            Adapter.Fill(ds, "Course");
            DataTable DBCourse = ds.Tables["Course"]; //创建生成数据表对象
            //创建新行
            DataRow dbRow = DBCourse.NewRow();
            //将数据存入该行
            dbRow["Cno"] = CnoStr;
            dbRow["Cname"] = CnameStr;
            //将该行加入表中
            DBCourse.Rows.Add(dbRow);
            //更新数据源
            Adapter.Update(ds, "Course");
            Response.Redirect("MessagePage.aspx?Cno=" + CnoStr + "&Cname=" + CnameStr +
"&MessStr= 插入成功！"); //切换新页
        }
        catch (Exception ee)
        {
            ee.Message.ToString();
            Response.Redirect("MessagePage.aspx?Cno=" + CnoStr + "&Cname=" + CnameStr
+ "&MessStr=Sorry, 插入失败！");
        }
    }
}

```

```

        finally
        {
            conn.Close();
        }
    }
}

```

⑤ 配置 Web.Config 文件。

```

<?xml version="1.0"?>
<configuration>
    <appSettings></appSettings>
    <connectionStrings>
        <add name="SqlConnStr"
            connectionString="Database=jiaoxuedb;server=mxm;
                Persist Security Info=false;Integrated Security=SSPI;"
            providerName="System.Data.SqlClient" />
    </connectionStrings>
    <system.web>
        <compilation debug="true"/>
        <authentication mode="Windows"/>
    </system.web>
</configuration>

```

⑥ 添加新页面 MessagePage.aspx，用于显示执行结果

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="MessagePage.aspx.cs" Inherits="MessagePage" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
    <head runat="server">
        <title>信息提示</title>
    </head>
    <body>
        <form id="form1" runat="server">
            <% Response.Write( Server.HtmlEncode(Request.QueryString["MessStr"]) + "<br>");%>
        </form>
    </body>
</html>

```

实验 19.3 更新数据库中的数据

1. 通过 SqlCommand 对象修改数据库中的数据：根据生姓名和课程名修改其成绩。执行该任务修改下面记录：姓名：王一山，课程名：数据库，成绩修改为：90。

③ 设计修改页面 Defaul.aspx。

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
    <head runat="server">
        <title>修改学生成绩</title>
    </head>

```



```

protected void Page_Load(object sender, EventArgs e)
{
    MessageLB.Visible = false;
    SnoTB.Focus();
}

protected void UpdateBT_Click(object sender, EventArgs e)
{
    string connString = "";
    //创建与数据库连接的对象：使用系统认证，从Web.Config中读取数据库连接串
   ConnectionStringSettings NameSettings;
    NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];
    if (NameSettings != null)
        connString = NameSettings.ConnectionString;//从Web.Config中获取连接串
    SqlConnection conn = new SqlConnection(connString);
    //创建生成SqlCommand对象的命令对象：CmdMsg
    string SnameStr = SnameTB.Text;
    string CnameStr = SnameTB.Text;
    int Score = ScoreTB.Text;
    string StrCommand = "Update SC Set Score=@Score) Where Student.Sno=SC.Sno and
SC.Cno=Course.Cno" + " and Sname=@SnameStr and Cname=@CnameStr Score =@Score ";
    SqlCommand CmdMsg = new SqlCommand(StrCommand, conn);
    CmdMsg.Parameters.Add("@Score", SqlDbType.Int, 2);
    CmdMsg.Parameters["@Score"].Value=Score;
    CmdMsg.Parameters.Add("@SnameStr", SqlDbType.Char, 8);
    CmdMsg.Parameters["@SnameStr"].Value=SnameStr;
    CmdMsg.Parameters.Add("@CnameStr", SqlDbType.Char, 20);
    CmdMsg.Parameters["@CnameStr"].Value = CnameStr;
    //执行SqlCommand对象CmdMsg.
    try
    {
        conn.Open();
        int RecordsAffected = CmdMsg.ExecuteNonQuery();
        if (RecordsAffected == 1)
            Response.Redirect("MessagePage.aspx?Sname=" + SnameStr + "&Cname=" +
CnameStr + "&MessStr= 修改成功! ");//切换新页
        MessageLB.Visible = true;
    }
    catch
    {
        Response.Redirect("MessagePage.aspx?Sname=" + SnameStr + "&Cname=" + CnameStr
+ "&MessStr= 修改成功! ");//切换新页
        MessageLB.Visible = true;
    }
    finally
    {
        conn.Close();
    }
}
}

```

```
<?xml version="1.0"?>
<configuration>
  <appSettings></appSettings>
  <connectionStrings>
    <add name="SqlConnStr"
          connectionString="Database=jiaoxuedb;server=mxm;
                           Persist Security Info=false;Integrated Security=SSPI;"
          providerName="System.Data.SqlClient" />
  </connectionStrings>
  <system.web>
    <compilation debug="true"/>
    <authentication mode="Windows"/>
  </system.web>
</configuration>
```

⑥ 添加新页面 MessagePage.aspx。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="MessagePage.aspx.cs" Inherits="MessagePage" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
    <head runat="server">
        <title>信息提示</title>
    </head>
    <body>
        <form id="form1" runat="server">
            <% Response.Write( Server.HtmlEncode(Request.QueryString["MessStr"]) + "<br>"); %>
        </form>
    </body>
</html>
```

2. 通过 `DataTable` 对象修改数据库中的数据：根据学生姓名和课程名修改其成绩。执行该任务修改下面记录：姓名：王一山，课程名：数据库，成绩修改为：85。

③ 设计修改页面 Default.aspx。

[illegible]


```

string CnameStr = CnameTB.Text;
int Score = Int32.Parse(ScoreTB.Text);
string StrSQL = "Select Sname,Cname,Score From Student,SC,Course Where
Student.Sno=SC.Sno"+ " and SC.Cno=Course.Cno and Sname=@SnameStr and name=@CnameStr";
//执行SqlCommand对象CmdMsg.
try
{
    conn.Open();
    //用SelectCommand属性创建数据集
    SqlDataAdapter dataAdapter = new SqlDataAdapter();
    SqlCommand SelCommand = new SqlCommand(StrSQL, conn);
    // Add the parameters for the SelectCommand.
    SelCommand.Parameters.Add("@SnameStr", SqlDbType.Char, 8);
    SelCommand.Parameters["@SnameStr"].Value = SnameStr;
    SelCommand.Parameters.Add("@CnameStr", SqlDbType.Char, 20);
    SelCommand.Parameters["@CnameStr"].Value = CnameStr;
    dataAdapter.SelectCommand = SelCommand;
    //用UpdateCommand属性修改数据源
    dataAdapter.UpdateCommand = new SqlCommand(
"UPDATE SC SET Score = @Score From Student,SC,Course WHERE Sname=@SnameStr and " +
" Cname=@CnameStr and Student.Sno=Sc.Sno and C.Cno=Course.Cno", conn);
    //设置UpdateCommand参数
    SqlParameter parameter1 = dataAdapter.UpdateCommand.Parameters.Add(
        "@SnameStr", SqlDbType.VarChar, 8, "SnameStr");
    parameter1.SourceColumn = "Sname";
    parameter1.SourceVersion = DataRowVersion.Current;//.Original;
    SqlParameter parameter2 = dataAdapter.UpdateCommand.Parameters.Add(
        "@CnameStr", SqlDbType.VarChar, 20, "CnameStr");
    parameter2.SourceColumn = "Cname";
    parameter2.SourceVersion = DataRowVersion.Original;
    SqlParameter parameter3 = dataAdapter.UpdateCommand.Parameters.Add(
        "@Score", SqlDbType.Int, 2, "Score");
    parameter3.SourceColumn = "Score";
    parameter3.SourceVersion = DataRowVersion.Current;// Original;
    DataSet dataSet = new DataSet();
    dataAdapter.Fill(dataSet, "SC");
    DataRow row = dataSet.Tables["SC"].Rows[0];//取行数据
    row["Score"] = Score;//修改成绩值
    dataAdapter.Update(dataSet, "SC");//修改数据源
    MessStr = " 修改成功! ";
}
catch(Exception err)
{
    MessStr = " 修改失败! ";
    Response.Redirect("MessagePage.aspx?MessStr=" +err.Message.ToString());
}
finally
{
    conn.Close();
    Response.Redirect("MessagePage.aspx?Sname=" + SnameStr + "&Cname=" + CnameStr

```



```

+ "&MessStr= "+MessStr); //切换信息显示页
    }
}
}

```

⑤ 配置 Web.Config 文件。

```

<?xml version="1.0"?>
<configuration>
  <appSettings></appSettings>
  <connectionStrings>
    <add name="SqlConnStr"
          connectionString="Database=jiaoxuedb;server=mxm;
                          Persist Security Info=false;Integrated Security=SSPI;"
          providerName="System.Data.SqlClient" />
  </connectionStrings>
  <system.web>
    <compilation debug="true"/>
    <authentication mode="Windows"/>
  </system.web>
</configuration>

```

⑥ 添加新页面 MessagePage.aspx，用于显示执行结果。

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="MessagePage.aspx.cs" Inherits="MessagePage" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
  <head runat="server">
    <title>信息提示</title>
  </head>
  <body>
    <form id="form1" runat="server">
      <% Response.Write( Server.HtmlEncode(Request.QueryString["MessStr"]) + "<br>"); %>
    </form>
  </body>
</html>

```

实验 19.4 删除数据库中的数据

1. 通过 SqlCommand 对象删除数据库中的数据：根据学生姓名删除该生记录。执行任务删除记录：姓名：许辉。

③ 设计删除页面 Defaul.aspx。

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
  <head runat="server">
    <title>删除学生信息</title>
  </head>
  <body>
    <form id="form1" runat="server">

```

[illegible]

④ 设计Default.aspx页面处理代码程序Default.aspx.cs。

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            SnameDropList.DataSource = DeleteStudent.DeleteClass.GetAllSname();
            SnameDropList.DataTextField = "Sname";
            SnameDropList.DataValueField = "Sname";
            SnameDropList.DataBind();
        }
    }
    protected void DeleteBT_Click(object sender, EventArgs e)
    {
        string SnameStr = SnameDropList.Text;
        DeleteStudent.DeleteClass.DeleteSelectSname(SnameStr);
        SnameDropList.DataSource = DeleteStudent.DeleteClass.GetAllSname();
    }
}
```

```

        SnameDropList.DataTextField = "Sname";
        SnameDropList.DataValueField = "Sname";
        SnameDropList.DataBind();
    }
}

```

⑤ 配置Web.Config文件。

```

<?xml version="1.0"?>
<configuration>
    <appSettings></appSettings>
    <connectionStrings>
        <add name="SqlConnStr"
            connectionString="Database=jiaoxuedb;server=mxm;
                                Persist Security Info=false;Integrated Security=SSPI;"
            providerName="System.Data.SqlClient" />
    </connectionStrings>
    <system.web>
        <compilation debug="true"/>
        <authentication mode="Windows"/>
    </system.web>
</configuration>

```

⑥ 创建新类 DeleteClass.cs。

```

using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
namespace DeleteStudent
{
    /// 定义类DeleteClass，及其2个方法。GetAllSname：获得学生姓名列表；
    ///DeleteSelectSname：删除学生信息
    public class DeleteClass
    {
        public DeleteClass()
        {
        }
        //获得所有学生姓名数据表，将其作为下拉列表框的数据源
        public static DataTable GetAllSname()
        {
            string connString = "";
            DataTable dt = new DataTable();
            //创建与数据库连接的对象：使用系统认证，从Web.Config中读取数据库连接串
            ConnectionStringSettings NameSettings;
            NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];

```

```

        if (NameSettings != null)
        {
            connString = NameSettings.ConnectionString; //从Web.Config中获取连接串
            SqlConnection conn = new SqlConnection(connString);
            //创建生成SqlDataAdapter对象的命令对象: Adapter
            string StrSQL = "Select Sname From Student";
            conn.Open();
            SqlDataAdapter dataAdapter = new SqlDataAdapter(StrSQL, conn);
            dataAdapter.Fill(dt);
            conn.Close();
            return dt;
        }
    }
    //删除下拉列表框中选中的学生信息
    public static void DeleteSelectSname(string SnameStr)
    {
        string connString = "";
        DataTable dt = new DataTable();
        //创建与数据库连接的对象: 使用系统认证, 从Web.Config中读取数据库连接串
        ConnectionStringSettings NameSettings;
        NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];
        if (NameSettings != null)
        {
            connString = NameSettings.ConnectionString; //从Web.Config中获取连接串
            SqlConnection conn = new SqlConnection(connString);
            //创建生成SqlCommand对象的命令对象
            string StrSQL = "Delete From Student Where Sname=@SnameStr";
            conn.Open();
            SqlCommand DelCommand = new SqlCommand(StrSQL, conn);
            // Add the parameters for the DelCommand.
            DelCommand.Parameters.Add("@SnameStr", SqlDbType.Char, 8);
            DelCommand.Parameters["@SnameStr"].Value = SnameStr;
            DelCommand.ExecuteNonQuery();
            conn.Close();
        }
    }
}

```

2. 通过 SqlDataAdapter 对象删除数据库中的数据: 根据学生姓名和课程名删除该生该课程的记录。执行任务删除记录: 姓名: 张彬, 课程名: 计算机基础。

③ 设计删除页面Default.aspx。

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>删除学生选课信息</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <table align="center" style="vertical-align: baseline; text-align: center">
        <tr>

```

[illegible]

④ 设计Default.aspx页面处理代码程序Default.aspx.cs。

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void DeleteBT_Click(object sender, EventArgs e)
    {
        string connString = "", MessStr = "";
        //创建与数据库连接的对象：使用系统认证，从Web.Config中读取数据库连接串
        ConnectionStringSettings NameSettings;
        NameSettings = ConfigurationManager.ConnectionStrings["SqlConnStr"];
        if (NameSettings != null)
            connString = NameSettings.ConnectionString; //从Web.Config中获取连接串
        SqlConnection conn = new SqlConnection(connString);
        //创建生成SqlDataAdapter对象的命令对象：Adapter
        string SnameStr = SnameTB.Text;
```

```

string CnameStr = CnameTB.Text;
string StrSQL = "Select Sname,Cname From Student,SC,Course Where
Student.Sno=SC.Sno+ " and SC.Cno=Course.Cno and Sname=@SnameStr and Cname=@CnameStr";
//执行SqlCommand对象CmdMsg.
try
{
    conn.Open();
    //用SelectCommand属性创建数据集
    SqlDataAdapter dataAdapter = new SqlDataAdapter();
    SqlCommand SelCommand = new SqlCommand(StrSQL, conn);
    // Add the parameters for the SelectCommand.
    SelCommand.Parameters.Add("@SnameStr", SqlDbType.Char, 8);
    SelCommand.Parameters["@SnameStr"].Value = SnameStr;
    SelCommand.Parameters.Add("@CnameStr", SqlDbType.Char, 20);
    SelCommand.Parameters["@CnameStr"].Value = CnameStr;
    dataAdapter.SelectCommand = SelCommand;
    //用DeleteCommand属性删除数据源
    dataAdapter.DeleteCommand = new SqlCommand("Delete SC From Student,SC,Course "
+"WHERE Sname=@SnameStr and Cname=@CnameStr and Student.Sno=Sc.Sno and SC.Cno=Course.Cno",
    conn);
    //设置DeleteCommand参数
    SqlParameter parameter1 = dataAdapter.DeleteCommand.Parameters.Add(
        "@SnameStr", SqlDbType.VarChar, 8, "SnameStr");
    parameter1.SourceColumn = "Sname";
    parameter1.SourceVersion = DataRowVersion.Original;
    SqlParameter parameter2 = dataAdapter.DeleteCommand.Parameters.Add(
        "@CnameStr", SqlDbType.VarChar, 20, "CnameStr");
    parameter2.SourceColumn = "Cname";
    parameter2.SourceVersion = DataRowVersion.Original;
    DataSet dataSet = new DataSet();
    dataAdapter.Fill(dataSet, "SC");
    DataRow row = dataSet.Tables["SC"].Rows[0]; //取行数据
    row.Delete();
    dataAdapter.Update(dataSet, "SC");
    MessStr = " 删除成功! ";
}
catch(Exception err)
{
    MessStr = " 删除失败! ";
    Response.Redirect("MessagePage.aspx?MessStr=" +err.Message.ToString());
}
finally
{
    conn.Close();
    Response.Redirect("MessagePage.aspx?Sname=" + SnameStr + "&Cname=" + CnameStr +
"&MessStr=" +MessStr); //切换提示信息页
}
}
}

```

⑤ 配置Web.Config文件。

```
<?xml version="1.0"?>
<configuration>
  <appSettings></appSettings>
  <connectionStrings>
    <add name="SqlConnStr"
        connectionString="Database=jiaoxuedb;server=mxm;
                          Persist Security Info=false;Integrated Security=SSPI;"
        providerName="System.Data.SqlClient" />
  </connectionStrings>
  <system.web>
    <compilation debug="true"/>
    <authentication mode="Windows"/>
  </system.web>
</configuration>
```

⑥ 添加新页面 MessagePage.aspx。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="MessagePage.aspx.cs" Inherits="MessagePage" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
  <head runat="server">
    <title>信息提示</title>
  </head>
  <body>
    <form id="form1" runat="server">
      <% Response.Write( Server.HtmlEncode(Request.QueryString["MessStr"]) + "<br>"); %>
    </form>
  </body>
</html>
```

实验 20 数据库应用系统设计

1. 数据库：rc_base.mdf
2. 代码：见文件夹“第 20 章人才档案管理系统 VB”