实验报告　存储过程触发器和函数

Hollow Man

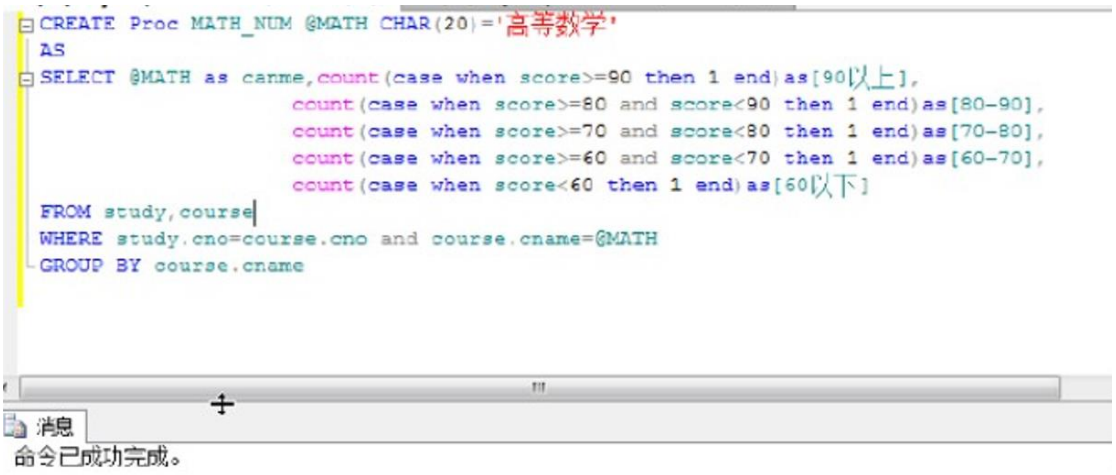
1. 存储过程

1)

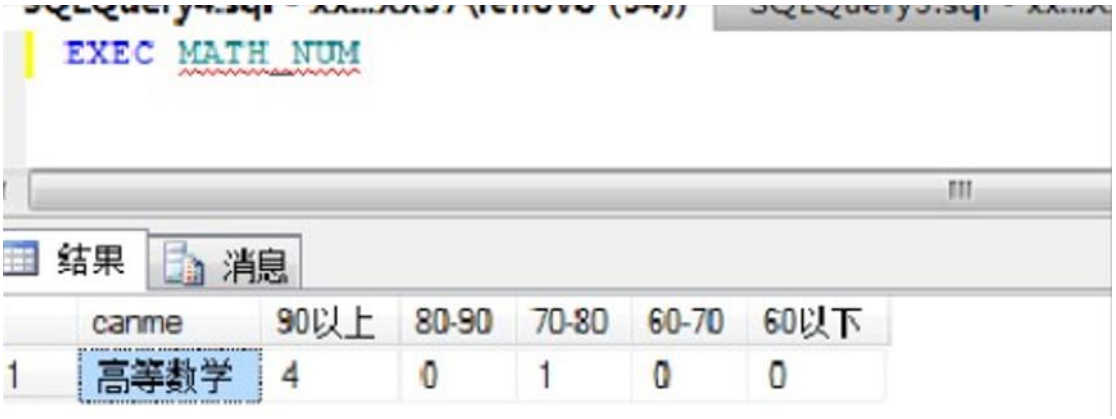
使用如下 SQL 命令:

```
CREATE Proc MATH_NUM @MATH CHAR(20 )= ' 高等数学 '
AS
SELECT @MATH as canme, count (case when score >= 90 then 1 end) as [90 以
上],
count (case when score >= 80 and score <90 then 1 end) as [80-90],
count (case when score >= 70 and score <80 then 1 end) as [70-80],
count (case when score >= 60 and score <70 then 1 end) as [60-70],
count (case when score <60 then 1 end) as [60 以下]
FROM study, course
WHERE study.cno =course.cno and course.cname =@MATH
GROUP BY course.cname
```

执行结果如下图所示:



运行结果如下图所示:

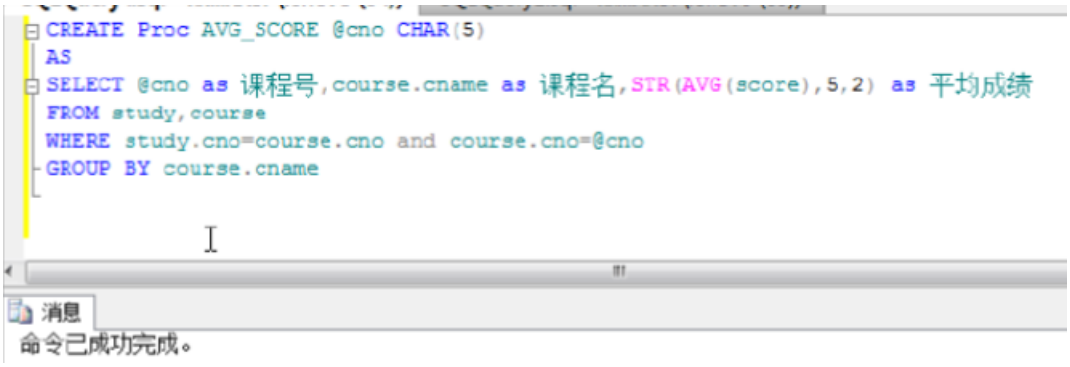


2)

使用如下 SQL 命令:

```
CREATE Proc AVG_SCORE @cno CHAR(5)
AS
SELECT @cno as 课程号, course.cname as 课程名 , STR(AVG(score), 5, 2) as 平均成绩
FROM study, course
WHERE study.cno =course.cno and course.cno =@cno
GROUP BY course.cname
```

执行结果如下图所示:



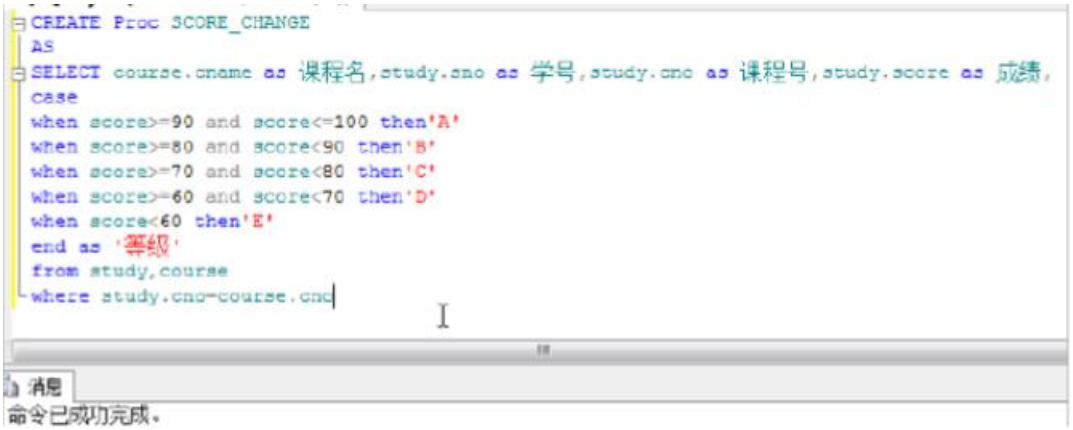
运行结果如下图所示:

3)

使用如下 SQL 命令:

```
CREATE Proc SCORE_CHANGE
AS
SELECT course.cname as 课程名 , study.sno as 学号 , study.cno as 课程号,
study.score as 成绩 ,
case
when score >= 90 and score <= 100 then 'A'
when score >= 80 and score <90 then 'B'
when score >= 70 and score <80 then 'C'
when score >= 60 and score <70 then 'D'
when score <60 then 'E'
end as '等级'
from study , course
where study.cno =course.cno
```

执行结果如下图所示:



运行结果如下图所示:

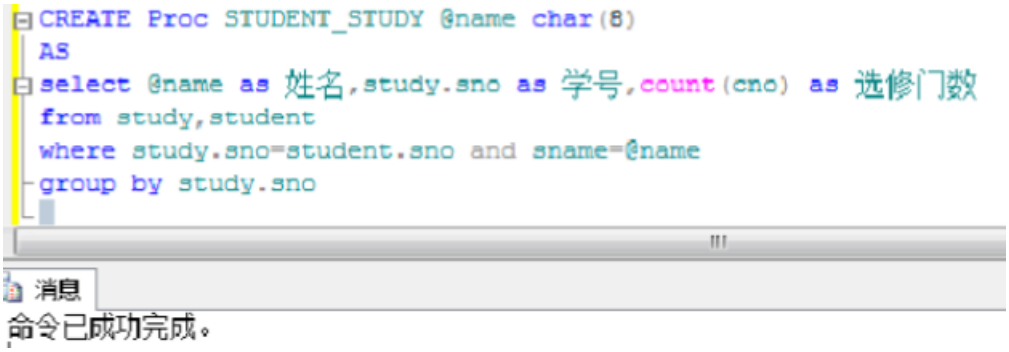
	课程名	学号	课程号	成绩	等级
1	高等数学	98501	C601	90	A
2	数据结构	98501	C602	90	A
3	操作系统	98501	C603	85	B
4	编译原理	98501	C604	87	B
5	高等数学	98502	C601	90	A
6	高等数学	98503	C601	75	C
7	数据结构	98503	C602	70	C
8	编译原理	98503	C604	56	E
9	高等数学	98504	C601	90	A
10	编译原理	98504	C604	85	B
11	高等数学	98505	C601	95	A
12	操作系统	98505	C603	80	B

4)

使用如下 SQL 命令:

```
CREATE Proc STUDENT_STUDY @name char (8)
AS
select @name as 姓名 , study.sno as 学号 , count (cno) as 选修门数
from study , student
where study.sno =student.sno and sname =@name
group by study.sno
```

执行结果如下图所示:



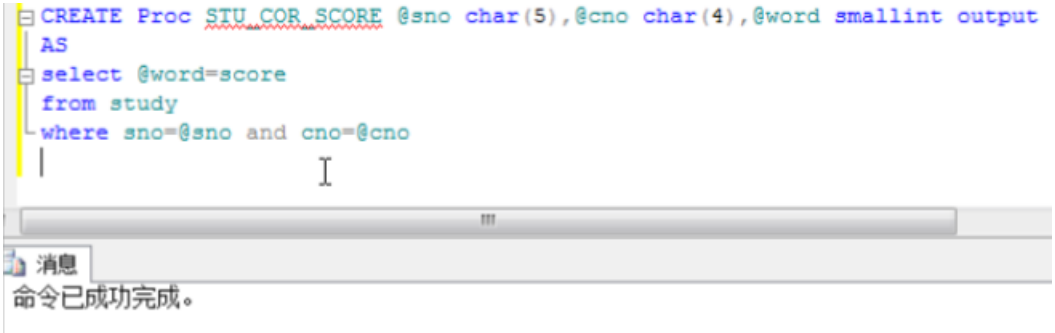
运行结果如下图所示:

5)

使用如下 SQL 命令:

```
CREATE Proc STU_COR_SCORE @sno char (5), @cno char (4), @word smallint
output
AS
select @word = score
from study
where sno = @sno and cno = @cno
```

执行结果如下图所示:



运行结果如下图所示:

```
declare @score smallint
execute STU_COR_SCORE 98601,C601,@score output
print ' 成绩是'+CONVERT(CHAR(2),@score)+'分'
```

消息

成绩是90分

2. 触发器

1)

使用如下 SQL 命令:

```
CREATE TRIGGER UPDATE_SCORE ON study
instead of update
as
declare @sno2 char ( 5), @cno2 char ( 4 ), @score1 smallint , @score2
smallint
select @sno2 =sno , @cno2 =cno , @score2 =score
from inserted
select @score1 =scorefrom deleted
if ( @score2 >= @score1 )
update study set score =@score2
where study.cno =@cno2 and study.sno =@sno2
go
```

执行结果如下图所示:

```
CREATE TRIGGER UPDATE_SCORE ON study
instead of update
as
declare @sno2 char(5),@cno2 char(4),@score1 smallint,@score2 smallint
select @sno2=sno,@cno2=cno,@score2=score
from inserted
select @score1=score
from deleted
if(@score2>=@score1)
update study set score=@score2
where study.cno=@cno2 and study.sno=@sno2
go
```

消息

命令已成功完成。

运行结果如下图所示:

可见在要求 sno=98604 cno=C604 score=85 改成 89 后，不能再改成 85 了

sno	cno	score
98601	C601	90
98601	C602	90
98601	C603	85
98601	C604	87
98602	C601	90
98603	C601	75
98603	C602	70
98603	C604	56
98604	C601	90
98604	C604	89
98605	C601	95
98605	C603	80
NULL	NULL	NULL

2)

使用如下 SQL 命令:

```
CREATE TRIGGER DEL_STUDY ON study
instead of DELETE
AS
begin
declare @num int , @sno char (5), @cno char (4)
select @num=COUNT(*) from deleted
if @num=1begin
select @sno=sno, @cno=cno from deleted
delete from study where @sno =study.sno and @cno=study.cno
end
else print '一次不能删除多条记录'
end
```

执行结果如下图所示:

```
CREATE TRIGGER DEL_STUDY ON study
 instead of DELETE
 AS
begin
 declare @num int,@sno char(5),@cno char(4)
 select @num=COUNT(*)from deleted
if @num=1
begin
 select @sno=sno,@cno=cno from deleted
 delete from study where @sno=study.sno and @cno=study.cno
 end
 else print '一次不能删除多条记录'
 end
```

消息

命令已成功完成。

运行结果如下图所示:

```
delete from study
where sno='98605'
```

消息

一次不能删除多条记录

(2 行受影响)

3)

使用如下 SQL 命令:

```
CREATE TRIGGER INSERT_COR ON course
instead of insert
AS
declare @cno char(4), @cname char(20), @teacher char(8)
select @cno =cno , @cname=cname , @teacher =teacher from inserted
if (@teacher is null)
print '注意：任课教师不能为空！'
else
insert course values (@cno , @cname, @teacher)
```

执行结果如下图所示:

```
CREATE TRIGGER INSERT_COR ON course
instead of insert
AS
declare @cno char(4),@cname char(20),@teacher char(8)
select @cno=cno,@cname=cname,@teacher=teacher from inserted
if(@teacher is null)
print '注意：任课教师不能为空！'
else
insert course values(@cno,@cname,@teacher)
```

消息

命令已成功完成。

运行结果如下图所示:

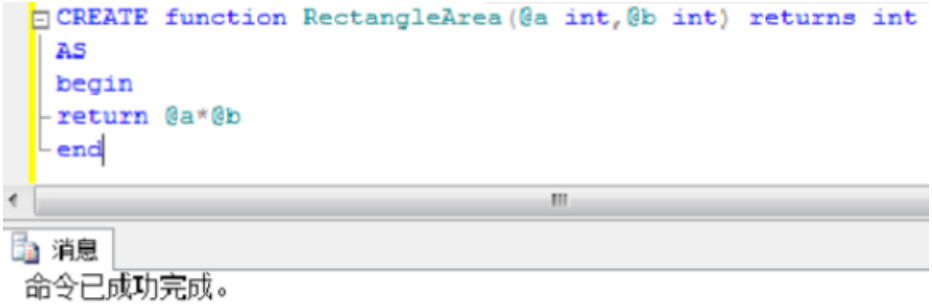
3. 函数

1)

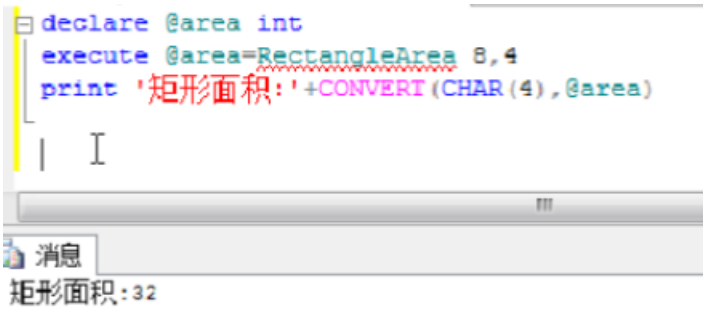
使用如下 SQL 命令:

```
CREATE function RectangleArea ( @a int , @b int ) returns int
AS
begin
return @a* @b
end
```

执行结果如下图所示:



运行结果如下图所示:



2)

使用如下 SQL 命令:

```
CREATE function STUDENT_TABLE () returns table
AS
return (
select student_course.cno 课程号 , course.cname 课程
名, COUNT(student_course.sno) 选修人数 ,
max (student_course.score) 最高分 , min (student_course.score ) 最低
分, AVG(student_course.score) 平均分
from student_course , course
where student_course.cno =course.cno
group by student_course.cno , course.cname
)
```

执行结果如下图所示:

运行结果如下图所示:

	课程号	课程名	选修人数	最高分	最低分	平均分
1	C1	数据库	6	95	55	79
2	C2	Java	2	99	95	97
3	C3	c++	2	55	55	55
4	C4	英语	3	98	79	88
5	C5	高等数学	6	99	10	64