实验6 存储过程、自定义函数和触发器

* **实验环境：**

操作系统：Windows 10 专业版

数据库管理系统：8.0.18 MySQL Community Server - GPL

客户端：MySQL Workbench 8.0 Community+VScode 1.71.2

数据库模式：Test6

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

课程表：Course(Cno,Cname,Cpno,Ccredit)

选课表：SC(Sno,Cno,Grade)

数据库的值：如下图所示



1. 创建一个带输入参数的存储过程，输入参数n执行存储过程得到n的阶乘。测试5！。

实验代码：

-- 如果存储过程已存在，则drop

drop procedure if exists fact;

-- 声明构建存储过程

DELIMITER //

create procedure fact(n int)

begin

    declare i int default 1;

    declare f int default 1;

    while i<=n do

        set f=f\*i; -- set用于赋值，且set不可省略(两个减号加一个空格可开启注释)

        set i=i+1;

    end while; /\*注意：end while后面要有分号\*/

    select f; -- 输出结果

end//

 -- 调用存储过程

call fact(5);

运行结果：



1. 创建一个简单的存储过程（SCInfo），要求结果输出选课信息，包括学号、姓名、课程号、课程名和成绩。

实验代码：

-- 如果存储过程已存在，则drop

drop procedure if exists SCInfo;

-- 声明构建存储过程

DELIMITER //

create procedure SCInfo()

Begin

select student.sno,sname,sc.cno,cname,grade

from student,sc,course

where student.sno = sc.sno and sc.cno = course.cno

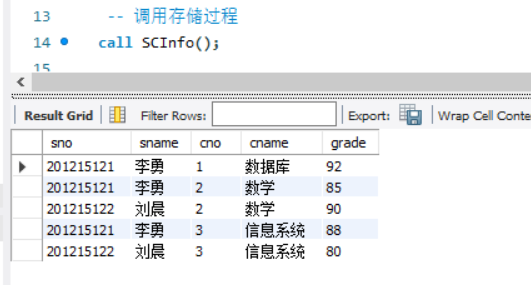
order by sc.cno;

end//

 -- 调用存储过程

call SCInfo();

运行结果：



3、创建一个带输入参数的存储过程，输入分数参数执行存储过程得到平均分大于该分数的学生统计成绩信息（包括学号、姓名、平均分、课程门数字段）。

实验代码：

-- 如果存储过程已存在，则drop

drop procedure if exists studentgrade;

-- 声明构建存储过程

DELIMITER //

create procedure studentgrade(in sgrade smallint)

Begin

select student.Sno,Sname,AVG(GRADE),COUNT(\*)

from student,sc

where student.sno = sc.sno

group by student.Sno

having avg(grade) > sgrade;

end//

 -- 调用存储过程

call studentgrade(85);

运行结果：



4、创建带两个输入参数和一个输出参数的存储过程，执行存储过程时，输入参数为分数段，输出参数为得到该分数段的人次数。

实验代码：

-- 如果存储过程已存在，则drop

drop procedure if exists SCNum;

-- 声明构建存储过程

DELIMITER //

create procedure SCNum(in sbegin int,in send int, inout SCNT int)

Begin

select count(\*) into SCNT from sc

where grade >= sbegin and grade < send;

end//

 -- 调用存储过程

set @SCNT = 0;

call SCNum(80,90,@SCNT);

select @SCNT;

运行结果：



5、创建一个自定义函数，函数返回值为全体学生人数，并写出测试语句以查看结果。

实验代码：

-- 如果函数已存在，则drop

drop function if exists num\_of\_students;

-- 声明构建函数

create function num\_of\_students()

returns smallint

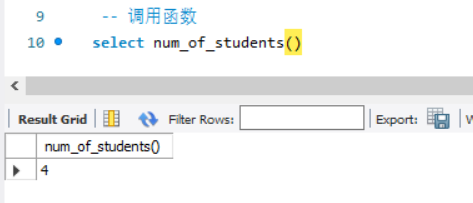
return

(select count(\*) from student);

 -- 调用函数

select num\_of\_students()

运行结果：



6、创建一个自定义函数，函数返回值为某个学生的平均成绩，并写出测试语句以查看结果。

实验代码：

-- 如果函数已存在，则drop

drop function if exists student\_avg\_grade;

-- 声明构建函数

create function student\_avg\_grade(ssno char(9))

returns float

return

(select avg(grade) from student,sc

where student.sno = sc.sno and student.sno = ssno);

-- 函数调用

select student\_avg\_grade('201215122');

运行结果：



7、在选课表上建立更新触发器，当对表SC的Grade属性进行修改时，若分数增加了10%则将此次操作记录到下面表中：SC\_U（Sno,Cno,Oldgrade,Newgrade），其中Oldgrade是修改前的分数，Newgrade是修改后的分数。写出测试语句以查看结果。

实验代码：

-- 如果表已存在，则drop

drop table if exists SC\_U;

-- 如果触发器已存在，则drop

drop trigger if exists ins\_rec;

-- 表恢复成初始值防止二次运行结果出问题

update sc set grade = 92 where sno = '201215121' and cno = '1';

update sc set grade = 85 where sno = '201215121' and cno = '2';

update sc set grade = 88 where sno = '201215122' and cno = '3';

-- 创建SC\_U表

create table SC\_U

(

    Sno CHAR(9),

    Cno CHAR(4),

    primary key(Sno,Cno),

    foreign key(Sno) references Student(Sno),

    foreign key(Cno) references Course(Cno),

    Oldgrade smallint,

    Newgrade smallint

);

-- 创建触发器

DELIMITER //

CREATE TRIGGER ins\_rec

BEFORE UPDATE

ON sc

FOR EACH ROW

IF (new.grade > old.grade\*1.1) THEN

    insert into SC\_U

    values(old.sno, old.cno, old.grade, new.grade);

END IF //

update sc set grade = 98 where sno = '201215121' and cno = '1';

select \* from sc\_u;

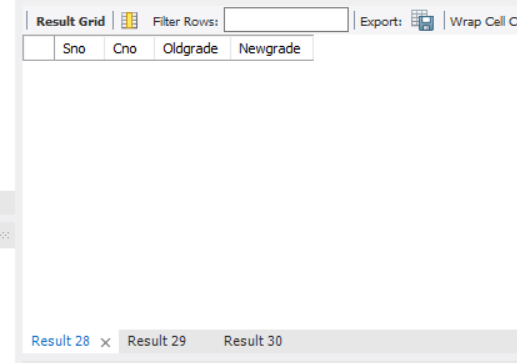
update sc set grade = 95 where sno = '201215121' and cno = '2';

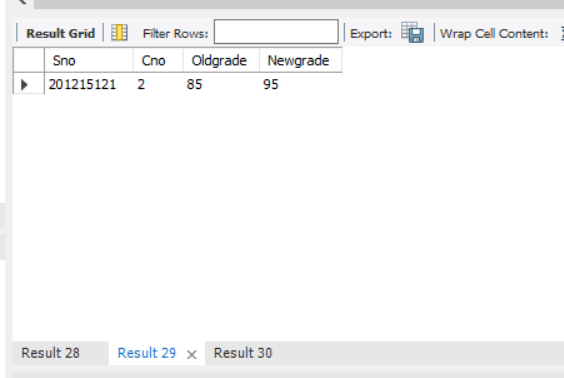
select \* from sc\_u;

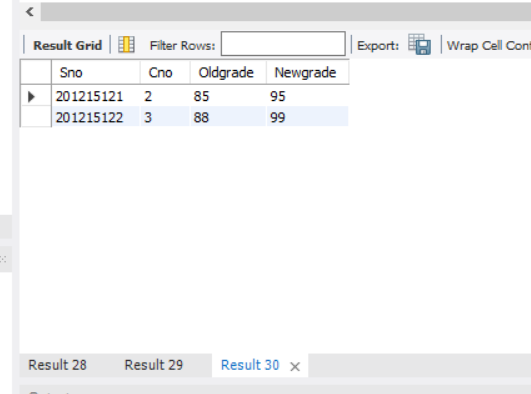
update sc set grade = 99 where sno = '201215122' and cno = '3';

select \* from sc\_u;

运行结果：







8、使用触发器实现检查约束，在向SC表插入记录时，grade字段的值或者为空，或者取值0~100，如果grade字段的值不满足要求，小于0则填入0，大于100则填入100。

实验代码：

-- 如果触发器已存在，则drop

drop trigger if exists set\_sc;

-- 创建触发器的过程

DELIMITER //

CREATE TRIGGER set\_sc

BEFORE insert

ON sc

FOR EACH ROW

IF new.grade is not null THEN

    if new.grade < 0 then

        set new.grade = 0;

    elseif new.grade > 100 then

        set new.grade = 100;

    end if;

END IF //

select \* from sc;

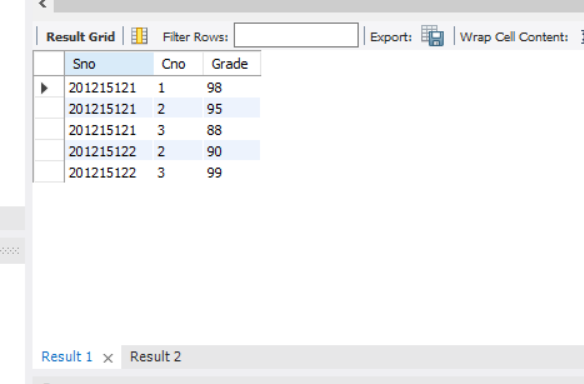
insert into sc values('201215123','5','-10');

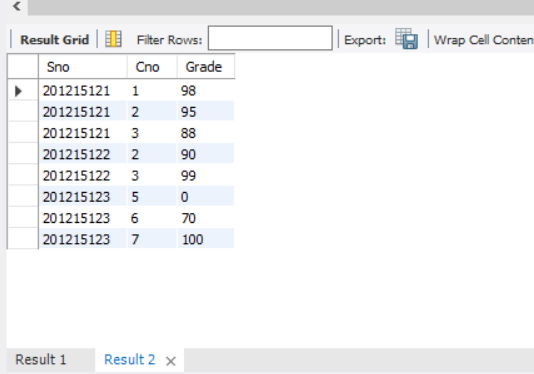
insert into sc values('201215123','6','70');

insert into sc values('201215123','7','110');

select \* from sc;

运行结果：





* 实验总结：

1.创建存储过程、函数、触发器实际上基本逻辑都是类似的，进行一系列的create，但是在创建的过程中就需要注意一些细节：比如我们想要把自己的代码供别人验证时就需要添加一些判断语句，就像这次实验一样每次创建前都检查是否exist，存在则drop，这样就不会出现别人验证出现error的问题。

2. 在使用if else语句也遇到了困难，要在末尾使用‘endif’才能构成一个完整的if语句。

3. 自定义终止符号“DELIMITER //”是个人查阅后选择的一种终止操作，由于sql语句默认的终止符号是 ‘;’，使用自定义符号解决了冲突问题。

4.异常交错问题仍需要多多注意，特别是最后两道题需要严谨去debug，更新了某数据想要二次检验一定要时刻保存原数据，防止每次运行结果都不一样，影响验证效果。