10

第十讲 存储过程、触发器和游标

教学内容

- **10.1 存储过程**
  - 存储过程介绍
  - 创建存储过程
  - 使用存储过程
- **10.2 触发器**
  - 触发器介绍
  - 创建触发器
  - 管理触发器
- **10.3 游标**
  - 游标介绍
  - 创建游标
  - 使用游标

10.1.1 存储过程概念

- 存储过程是将SQL语句放入一个集合里，然后直接调用存储过程来执行已经定义好的SQL语句集合
- 避免开发人员重复编写相同的SQL语句
- 存储过程还可以减少数据在数据库和应用服务器之间的传输，可以提高数据处理的效率。

存储过程的优点

- 存储过程中可以包含数据存取语句、流程控制语句、错误处理语句等，在使用上非常有弹性。
- 优点
  - 执行效率高
  - 统一的操作流程
  - 重用性、共享性和可移植性
  - 安全性高

10.1.2 创建存储过程

- CREATE [DEFINER = { user | CURRENT_USER }]
- PROCEDURE sp_name ([proc_parameter[,...]])
- [characteristic ...] routine_body
- proc_parameter: [IN | OUT | INOUT] param_name type
- characteristic: COMMENT 'string' | LANGUAGE SQL
- | [NOT] DETERMINISTIC | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA } | SQL SECURITY { DEFINER | INVOKER }
- routine_body:
- Valid SQL routine statement
- [begin_label:] BEGIN
- [statement_list]
- ……
- END [end_label]

存储过程中的关键语法

- 声明语句结束符:
  - **DELIMITER $$ 或 DELIMITER //**
- 声明存储过程:
  - **CREATE PROCEDURE demo_in_parameter(IN p_in int)**
- 存储过程开始和结束符号:
  - **BEGIN .... END**
- 变量赋值:
  - **SET @p_in=1**
- 变量定义:
  - **DECLARE l_int int unsigned default 4000000;**

10.1.3 存储过程的参数

- CREATE PROCEDURE 存储过程名([[IN |OUT |INOUT] 参数名 数据类型...])
- 三种参数类型
  - IN 输入参数：表示调用者向过程传入值（传入值可以是字面量或变量）
  - OUT 输出参数：表示过程向调用者传出值(可以返回多个值)（传出值只能是变量）
  - INOUT 输入输出参数：既表示调用者向过程传入值，又表示过程向调用者传出值（值只能是变量）
- 查看存储过程的定义
  - show create procedure proc_name;

in 输入参数

```
delimiter $$
create procedure in_param(in p_in int)
   begin
   select p_in;
   set p_in=2;
   select P_in;
   end $$
delimiter ;
set @p_in=1;
call in_param(@p_in);
```

out输出参数

```
delimiter //
create procedure out_param(out p_out int)
   begin
     select p_out;
     set p_out=2;
     select p_out;
   end //
delimiter ;
set @p_out=1;
call out_param(@p_out);
```

inout输入参数

```
delimiter $$
create procedure inout_param(inout p_inout int)
    begin
      select p_inout;
      set p_inout=2;
      select p_inout;
    end $$
delimiter ;
set @p_inout=1;
call inout_param(@p_inout);
```

inout输入参数

```
DELIMITER //
CREATE PROCEDURE sp_test(IN p_num INT,OUT p_result INT)
BEGIN
SET p_result = 1;
WHILE p_num>1 DO
  SET p_result = p_result * p_num ;
  SET p_num = p_num -1;
END WHILE;
END //
DELIMITER ;
CALL sp_test(5,@result);
SELECT @result;
```

求阶乘存储过程

- create procedure get_jc(in n int)
- begin
  - declare i int default 1;
  - declare k int default 1;
  - while i<=n do
  - set k=k*i;
  - set i=i+1;
  - end while;
  - select i;
  - select k;
- end;

Case存储过程

```
DELIMITER //
CREATE PROCEDURE proc3 (in parameter int)
 begin
    declare var int;
    set var=parameter+1;
    case var
    when 0 then
    insert into t values(17);
    when 1 then
    insert into t values(18);
    else
    insert into t values(19);
    end case;
end;   //
DELIMITER ;
```

While存储过程添加记录

```
DELIMITER //
CREATE PROCEDURE proc4()
    begin
    declare var int;
    set var=0;
    while var<6 do
        insert into t values(var);
        set var=var+1;
    end while;
    end;
    //
DELIMITER ;
```

学生存储过程

```
CREATE PROCEDURE sp_stud (IN p_num INT)
BEGIN
  CASE p_num
    WHEN 1 THEN
      SELECT * FROM StudInfo WHERE StudName  LIKE '王%';
    WHEN 2 THEN
      SELECT * FROM StudInfo WHERE StudName  LIKE '张%';
    ELSE
      SELECT * FROM StudInfo WHERE StudName  LIKE '李%';
END CASE;
END
```

存储过程示例

```
DELIMITER //
DROP PROCEDURE if exists P_Save_ClassInfo;
Create Procedure P_Save_ClassInfo
(in CID Varchar(10),in CName Varchar(50),in CDesc Varchar(100))
begin
  if Exists(Select * From ClassInfo Where ClassID=CID) then
     Update ClassInfo Set ClassName=CName,ClassDesc=CDesc
      Where ClassID=CID;
  Else
     Insert Into ClassInfo(ClassID,ClassName,ClassDesc) Values
(CID,CName,CDesc);
End if;
End  //
DELIMITER ;
```

存储过程示例

```
DELIMITER //
DROP PROCEDURE if exists P_Save_ClassInfo;
Create Procedure P_Save_ClassInfo
(in CID Varchar(10),in CName Varchar(50),in CDesc Varchar(100),out msg
varchar(20))
begin
  if Exists(Select * From ClassInfo Where ClassID=CID) then
     Update ClassInfo Set ClassName=CName,ClassDesc=CDesc
     Where ClassID=CID;
     set msg='修改成功';
  Else
     Insert Into ClassInfo(ClassID,ClassName,ClassDesc) Values (CID,CName,CDesc);
     set msg='添加成功';
End if;
End //
DELIMITER ;
call P_Save_ClassInfo('20181152','computer2018','good',@msg);
select @msg;
```

10.2 触发器

- 定义
  - 触发器(Trigger)是针对单一数据表所撰写的特殊存储过程，当该数据表发生Insert、Update或Delete时会自动被触发(执行)
- 功能
  - 检查所做的更改是否允许
  - 进行其他相关数据的更改动作
  - 发出更改或警告通知
  - 自定义错误信息
  - 更改原来所要进行的数据操作

触发器

- create trigger triggerName
- after/before insert/update/delete
- on 表名
- for each row
- begin
- 　sql语句; -- 触发器内容主体，每行用分号结尾
- end;

查看触发器

- 查看全部触发器
  - **show triggers;**
- 查看触发器的创建语句
  - **show create trigger 触发器名字;**
- 删除触发器
  - **drop trigger 触发器名字**

触发器中的虚拟表

- **new**
  - 存储要添加的记录
  - **INSERT**型触发器
- **old**
  - 存储被删除的记录
  - **DELETE**型触发器

学生选课触发器示例

教师任课表（TeacherCourseInfo）

字段名称	数据类型	字段长度	PK	字段描述	举例
TeacherNo	Varchar	15	Y	教师编号	HB001
CourseID	Varchar	15	Y	课程编号	GDSX01
LimitPersonCount	Int			限选人数	15
RemainPersonCount	Int			已选人数	2

选课结果表（ElectCourseResult）

字段名称	数据类型	字段长度	PK	字段描述	举例
StudNo	Varchar	15	Y	学生学号	20191152001
TeacherNo	Varchar	15	Y	教师编号	HB001
CourseID	Varchar	15	Y	课程编号	GDSX01

创建教师任课表（TeacherCourseInfo）

```
Create Table TeacherCourseInfo
(
    TeacherNo Varchar(15),
    CourseID Varchar(15),
    LimitPersonCount int,
    RemainPersonCount int,
    Constraint PK_T_C Primary key(TeacherNo,CourseID)
);
```

添加测试数据

- Insert Into TeacherCourseInfo values('HB001','GDSX01',15,0);
- Insert Into TeacherCourseInfo values('LL001','GDSX01',10,0);
- Insert Into TeacherCourseInfo values('HB001','GCSX01',10,0);
- Insert Into TeacherCourseInfo values('LL001','GCSX01',12,0);

创建选课结果表（ElectCourseResult）

- Create Table ElectCourseResult
- (
- StudNo varchar(15),
- TeacherNo varchar(15),
- CourseID varchar(15),
- Constraint PK_S_T_C Primary key(StudNo,TeacherNo,CourseID)
-);

创建学生选课触发器（Insert）

```
Create Trigger T_ElectCourseResult_Insert
After Insert  On ElectCourseResult
For each row
Begin
  Update TeacherCourseInfo T
  Set RemainPersonCount=RemainPersonCount+1
  Where T.TeacherNo=new.TeacherNo
                And T.CourseID=new.CourseID;
End;
```

创建学生选课触发器（Delete）

```
Create Trigger T_ElectCourseResult_Delete
After DELETE  On ElectCourseResult
For each row
Begin
  Update TeacherCourseInfo T
  Set RemainPersonCount=RemainPersonCount-1
  Where T.TeacherNo=old.TeacherNo
                And T.CourseID=old.CourseID;
End;
```

10.3 游标

- 可以把Cursor看成是一个用来保存“数据集”（多条记录）的对象
- 因SELECT语句查询的结果，是直接返回前端应用程序中，而无法在SQL程序中一笔一笔地处理
- 可以将挑选出来的结果先放入Cursor中，然后利用循环将每一条记录从Cursor中取出来处理

使用游标

- 声明
  - **declare 游标名 cursor for select_statement**
- 打开游标
  - **open 游标名**
- 从游标中取值
  - **fetch 游标名 into var1,var2[,...] --将取到的一行赋值给多个变量**
- 关闭游标
  - **close 游标名**

游标示例

```
create procedure Proc_cur_studinfo()
begin
  declare SID varchar(30) ;
  declare SName varchar(30);
  declare SGender varchar(10);
  declare have int default 1;
  declare cur_studinfo cursor for
     select studno,studname,studgender from studinfo;
  declare exit handler for NOT FOUND set have:= 0;
  open cur_studinfo;
  repeat
    fetch cur_studinfo into SID,SName,SGender;
    select SID,SName;
  until have = 0 end repeat;
  close cur_studinfo;
end;
```

学生成绩排名游标示例——创建平均分视图

```
Create View V_StudAvgScore
as
Select S.StudNo,StudName,
        Cast(Avg(StudScore) AS Decimal(4,1))
        As AvgScore
From StudInfo S,StudScoreInfo SI
Where S.StudNo=SI.StudNo
Group By S.StudNo,StudName;
```

学生成绩排名游标示例——存储过程、游标

```
Create Procedure P_GetStudQuene()
begin
Declare SNo varchar(20);
Declare SName varchar(20);
Declare i int default 1;
Declare Avg_Score decimal(4,1);
declare have int default 1;
Declare Cur_StudQuene Cursor For Select StudNo,StudName,AvgScore From
V_StudAvgScore Order By AvgScore desc;
declare exit handler for NOT FOUND set have:= 0;
Open Cur_StudQuene;
  repeat
   Fetch Cur_StudQuene into SNO,SName,Avg_Score;
   select SNO,SName,Avg_Score,i;
    set i=i+1;
  until have = 0 end repeat;
Close Cur_StudQuene;
end;
```

成绩同名处理示例

- Create Procedure P_GetStud_same_Quene()
- begin
- Declare SNo varchar(20);
- Declare SName varchar(20);
- Declare i int default 1;
- Declare j int default 1;
- Declare Avg_Score decimal(4,1);
- Declare prev_j int default 1;
- Declare prev_score decimal(4,1);
- declare have int default 1;
- Declare Cur_StudQuene Cursor For Select StudNo,StudName,AvgScore From V_StudAvgScore Order By AvgScore desc;
- declare exit handler for NOT FOUND set have:= 0;

成绩同名处理示例

```
Open Cur_StudQuene;
  repeat
   Fetch Cur_StudQuene into SNO,SName,Avg_Score;
   if avg_score=prev_score then
     set j=prev_j;
   else
     set j=i;
   end if;
   select SNO,SName,Avg_Score,j;
   set prev_j=j;
   set prev_score=avg_score;
   set i=i+1;
  until have = 0 end repeat;
Close Cur_StudQuene;
end;
```

成绩同名处理示例——方法二

```
DELIMITER //
DROP PROCEDURE IF EXISTS P_GetQuence;
Create Procedure P_GetQuence()
begin
   set @strsql= 'Drop Table if Exists Temp_Quence;';
   PREPARE drop_sql from @strsql;
   execute drop_sql;
   set @strsql=" Create Table Temp_Quence(
                        StudNo Varchar(15),
                        StudName Varchar(20),
                        AvgScore Numeric(4,1),
                        QuenceOne int auto_increment primary key,
                        QuenceTwo int null);";
    Prepare create_sql from @strsql;
    execute create_sql;
```

成绩同名处理示例——方法二

```
Insert Into Temp_Quence(StudNo,StudName,AvgScore)
    Select StudNo,StudName,AvgScore
    From V_StudAvgScore
    Order By AvgScore Desc;
     Update Temp_Quence,
     (Select AvgScore,Min(QuenceOne) MinQuence
     From Temp_Quence Group By AvgScore) A
      Set QuenceTwo=A.MinQuence
      Where Temp_Quence.AvgScore=A.AvgScore;
end
//
DELIMITER;
```

下次课内容

- 11.1 用户管理
- 11.2 权限管理
- 11.3 角色管理
- 11.4 批量用户管理