

数据库课程实验日志

学号	201726010211	姓名	陈汉轩	专业年级班级	数媒 1701
实验日期	2020. 5. 29	实验项目	存储过程		

目录

数据库课程实验日志.....	1
一、实验目的.....	1
二、实验过程&错误.....	2
0. 准备数据表	2
1 存储过程实验	3
创建存储过程.....	3
查看存储过程.....	6
修改存储过程.....	6
删除存储过程.....	7
2 自定义函数实验	7
定义不带有参数函数并调用.....	8
定义带参数函数并调用.....	10
删除函数.....	11
3 游标实验	12
游标的处理过程.....	12
创建使用删除游标.....	13
三、实验重难点.....	13
四、实验心得体会.....	14

一、实验目的

掌握数据库 PL/SQL 编程语言，以及数据库存储过程的设计和使用方法。

掌握数据库 PL/SQL 编程语言以及数据库自定义函数的设计和使用方法。

掌握 PL/SQL 游标的设计、定义和使用方法，理解 PL/SQL 游标按行操作和 SQL 按结果集操作的区别和联系。

二、实验过程&错误

0. 准备数据表

表 7.1 sch 表结构

字段名	数据类型	主键	外键	非空	唯一	自增
id	INT(10)	是	否	是	是	否
name	VARCHAR(50)	否	否	是	否	否
class	VARCHAR(50)	否	否	是	否	否

表 7.2 sch 表的内容

id	name	class
1	李明	C1
2	小梅	C2

toor

dbexp6

运行

```
1 create table sch(  
2   id int(10) primary key,  
3   name varchar(50) not null,  
4   class varchar(50) not null);  
5  
6
```

信息

剖析

状态

```
create table sch(  
id int(10) primary key,  
name varchar(50) not null,  
class varchar(50) not null)  
> OK  
> 时间: 0.098s
```

对象 * 无标题 - 查询			
开始事务 文本 筛选 排序 导入			
id	name	class	score
1	李明	C1	90
2	小梅	C2	80

1 存储过程实验

创建存储过程

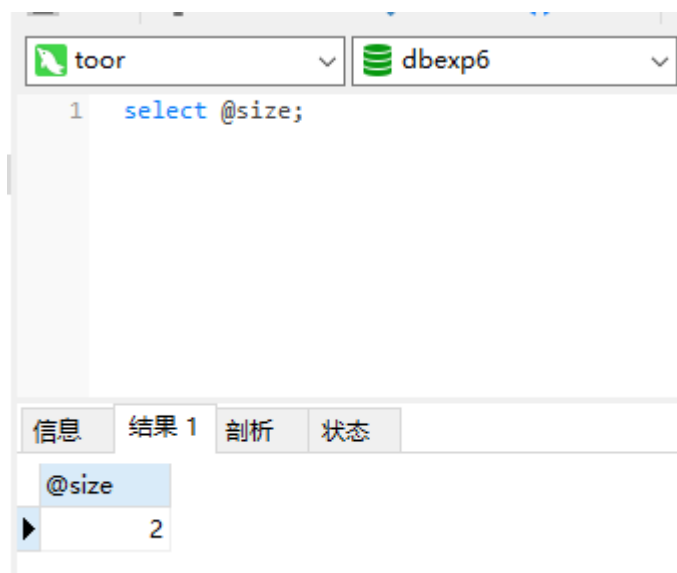
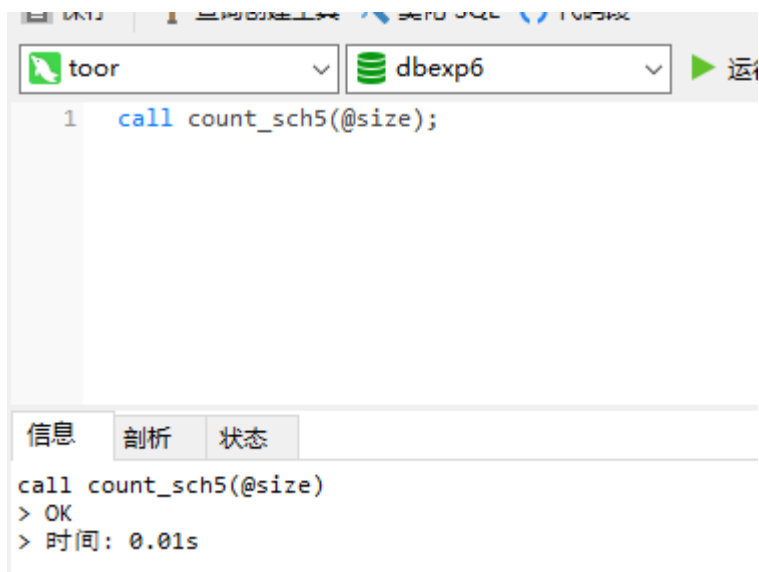
创建一个存储过程，用来统计表 sch 中的记录数。

```
toor dbexp6 运行
1 delimiter $$
2 create PROCEDURE count_sch5(out size int)
3 BEGIN
4 select count(*) into size from sch;
5 END
6 $$
7 delimiter;
8
```

信息 剖析 状态

```
create PROCEDURE count_sch5(out size int)
BEGIN
select count(*) into size from sch;
END
> OK
> 时间: 0.023s
```

调用 count_sch5:



创建一个存储过程，通过调用存储函数的方法来获取表 sch 中的记录数和 sch 表中 id 的和。

toor

dbexp6

运行

停

```
1 delimiter $$
2 create PROCEDURE count_sch6(out s_a int, out s_id int)
3 BEGIN
4     select count(*) into s_a from sch;
5     select sum(id) as s_id from sch;
6 END
7 $$
8 delimiter;
```

信息

剖析

状态

```
create PROCEDURE count_sch6(out s_a int, out s_id int)
BEGIN
select count(*) into s_a from sch;
select sum(id) as s_id from sch;
END
> OK
> 时间: 0.015s
```

调用:

toor

dbexp6

运行

停

```
1 call count_sch6(@out1, @out2);
2 select @out1;
3 select @out2;
```

信息

结果 1

结果 2

结果 3

剖析

状态

s_id
3

查看存储过程

toor

dbexp6

运行

停止

解释

```
1 SHOW PROCEDURE STATUS LIKE "%count_sch6%";
2
```

信息

结果 1

剖析

状态

Db	Name	Type	Definer	Modified	Created	Security_type
dbexp6	count_sch6	PROCEDURE	root@localhost	2020-05-30 12:28:03	2020-05-30 12:28:03	DEFINER

修改存储过程

MySQL 中修改存储过程的语法格式如下：

```
ALTER PROCEDURE 存储过程名 [ 特征 ... ]
```

特征 指定了存储过程的特性，可能的取值有：

- CONTAINS SQL 表示子程序包含 SQL 语句，但不包含读或写数据的语句。
- NO SQL 表示子程序中不包含 SQL 语句。
- READS SQL DATA 表示子程序中包含读数据的语句。
- MODIFIES SQL DATA 表示子程序中包含写数据的语句。
- SQL SECURITY { DEFINER | INVOKER } 指明谁有权限来执行。
- DEFINER 表示只有定义者自己才能够执行。
- INVOKER 表示调用者可以执行。
- COMMENT 'string' 表示注释信息。

下面修改存储过程 count_sch6 的定义，将读写权限改为 MODIFIES SQL DATA，并指明调用者可以执行

toor

dbexp6

运行

停止

解释

```
1 ALTER PROCEDURE count_sch6 MODIFIES SQL DATA SQL SECURITY INVOKER;  
2
```

信息

剖析

状态

ALTER PROCEDURE count_sch6 MODIFIES SQL DATA SQL SECURITY INVOKER
> OK
> 时间: 0.017s

删除存储过程

toor

dbexp6

运行

停止

解释

```
1 DROP PROCEDURE IF EXISTS count_sch5;  
2
```

信息

剖析

状态

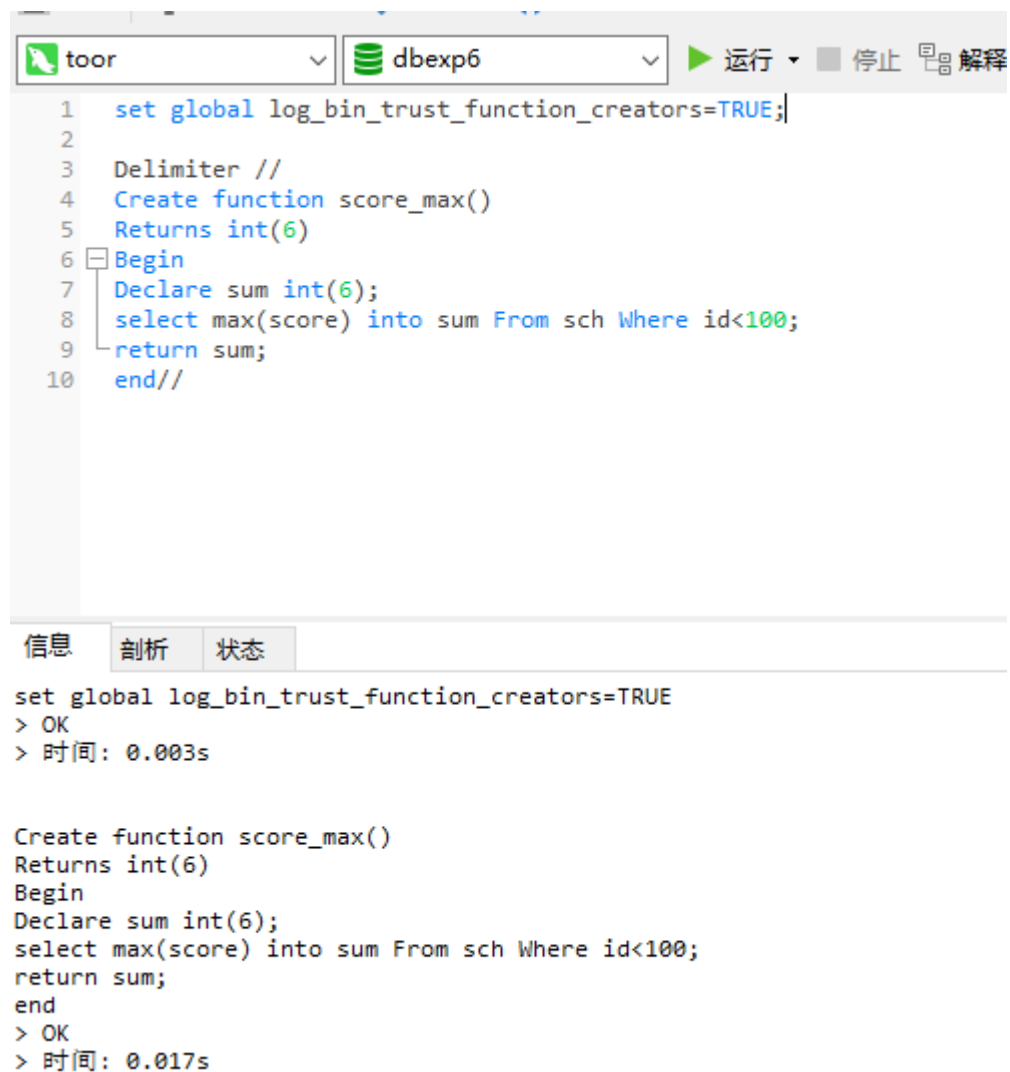
DROP PROCEDURE IF EXISTS count_sch5
> OK
> 时间: 0.023s

2 自定义函数实验

加上 score 列:

对象 * 无标题 - 查询			
开始事务 文本 筛选 排序 导入			
id	name	class	score
1	李明	C1	90
2	小梅	C2	80

定义不带有参数函数并调用



The screenshot shows a SQL IDE interface with a toolbar at the top containing icons for 'toor', 'dbexp6', '运行' (Run), '停止' (Stop), and '解释' (Explain). The main editor area contains the following SQL code:

```
1  set global log_bin_trust_function_creators=TRUE;
2
3  Delimiter //
4  Create function score_max()
5  Returns int(6)
6  Begin
7  Declare sum int(6);
8  select max(score) into sum From sch Where id<100;
9  return sum;
10 end//
```

Below the editor, there are three tabs: '信息' (Info), '剖析' (Analyze), and '状态' (Status). The '信息' tab is selected, showing the execution results of the SQL statements:

```
set global log_bin_trust_function_creators=TRUE
> OK
> 时间: 0.003s

Create function score_max()
Returns int(6)
Begin
Declare sum int(6);
select max(score) into sum From sch Where id<100;
return sum;
end
> OK
> 时间: 0.017s
```

调用:

toor dbexp6 运行

```
1 SELECT score_max();|
2
```

信息 结果 1 剖析 状态

score_max()

▶	90
---	----

定义带参数函数并调用



The screenshot shows a database IDE interface. At the top, there are two dropdown menus: the first is labeled 'toor' and the second is labeled 'dbexp6'. To the right of these are two buttons: a green play button labeled '运行' (Run) and a grey square button labeled '停止' (Stop). Below the menus is a text area containing SQL code. The code is as follows:

```
1 Delimiter //
```

```
2 Create function avg_score( sname varchar(8))
```

```
3 Returns int
```

```
4 BEGIN
```

```
5     Declare a_score int;
```

```
6     select score INTO a_score from sch Where name=sname;
```

```
7 return a_score;
```

```
8 END//
```

Below the text area, there are two tabs: '信息' (Information) and '状态' (Status). The '信息' tab is selected, and it displays the following output:

```
Create function avg_score( sname varchar(8))
```

```
Returns int
```

```
BEGIN
```

```
    Declare a_score int;
```

```
    select score INTO a_score from sch Where name=sname;
```

```
return a_score;
```

```
END
```

```
> 1304 - FUNCTION avg_score already exists
```

```
> 时间: 0.001s
```

调用:

toor

dbexp6

运行

```
1 select avg_score("李明");
```

信息

结果 1

剖析

状态

avg_score("李明")	
▶	90

删除函数

toor

dbexp6

运行

```
1 drop FUNCTION avg_score;
```

信息

剖析

状态

```
drop FUNCTION avg_score
> OK
> 时间: 0.016s
```

3 游标实验

游标的处理过程

4 步

①声明游标 **declare**: 没有检索数据，只是定义要使用的 **select** 语句

②打开游标 **open**: 打开游标以供使用，用上一步定义的 **select** 语句把数据实际检索出来

③检索游标 **fetch**: 对于填有数据的游标，根据需要取出(检索)各行

④关闭游标 **close**: 在结束游标使用时，必须关闭游标

创建使用删除游标

toor

dbexp6

运行

停止

解释

```
1 CREATE PROCEDURE testEndHandle()
2 BEGIN
3     DECLARE done BOOLEAN DEFAULT 0;
4     DECLARE tmp_uid INT DEFAULT 0;
5     DECLARE tmp_uname VARCHAR(255) DEFAULT "default_name";
6     DECLARE tmp_uclass VARCHAR(255);
7     DECLARE tmp_uscore INT DEFAULT 0;
8
9     DECLARE t_index CURSOR FOR SELECT sch.id, sch.name, sch.class, sch.score FROM sch;
10    -- 写法一: DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=1;
11    -- DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
12    -- 写法二: DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET done = 1;
13    DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET done = 1;
14
15    OPEN t_index;
16    REPEAT
17    FETCH t_index INTO tmp_uid,tmp_uname,tmp_uclass,tmp_uscore;
18    IF done!=1 THEN
19        SELECT tmp_uid,tmp_uname,tmp_uclass,tmp_uscore;
20    END IF;
21 UNTIL DONE END REPEAT;
22 CLOSE t_index;
23
24 END;
25
26 call testEndHandle();
27
28 DROP PROCEDURE IF EXISTS testEndHandle;
```

信息	结果 1	结果 1 (2)	结果 1 (3)	结果 1 (4)	剖析	状态
	tmp_uid	tmp_uname	tmp_uclass	tmp_uscore		
	1	李明	C1	90		

三、实验重难点

实验重点：存储过程定义和运行。

实验难点：存储过程的参数传递方法。

实验重点：自定义函数的定义和运行。

实验难点：自定义函数的参数传递方法。

实验重点：游标定义和使用。

实验难点：游标类型。

四、实验心得体会

学习了存储过程定义、存储过程运行，存储过程更名，存储过程删除，存储过程的参数传递。

掌握 PL/SQL 编程语言和编程规范，规范设计存储过程。

学习了自定义函数定义、自定义函数运行，自定义函数更名，自定义函数删除，自定义函数的参数传递。掌握 PL/SQL 和编程规范，规范设计自定义函数。

学习了游标定义、游标使用。掌握各种类型游标的特点、区别与联系。