Atitit mysql存储过程编写指南

[1. 定义变量 1](#_Toc31767)

[1.1. 变量名以@开头用户变量 会话变量 1](#_Toc997)

[1.2. 以declare关键字声明 存储过程变量 2](#_Toc18429)

[1.3. @是用户自定义变量,@@是系统定义变量 2](#_Toc24287)

[1.4. 变量赋值 使用set语句或select 2](#_Toc20393)

[1.5. 表记录赋值到变量 用set或into的方式来实现赋值 2](#_Toc45)

[1.6. 查询结果存储到变量 3](#_Toc15464)

[2. 参数与返回值 4](#_Toc23312)

[2.1. 设定参数与返回值 `obj\_id` int ，，返回类型 varchar(200) 4](#_Toc3835)

[2.2. 在语句中使用传入的obj\_id参数 4](#_Toc26051)

[2.3. 存储过程返回值使用out参数 4](#_Toc25411)

[3. other 5](#_Toc6376)

[3.1. 测试的时候，注意字符串参数要包含双引号 5](#_Toc4732)

[3.2. Sql参数拼接 CONCAT， 传入like的参数 5](#_Toc10088)

[3.3. Sql语句调用存储过程call 语句 5](#_Toc12316)

[3.4. 游标嵌套 handler 嵌套 独立begin快 6](#_Toc6732)

[4. 游标与循环 6](#_Toc32458)

[5. 异常处理机制 7](#_Toc6228)

[5.1. 动态SQL 8](#_Toc9751)

[6. 调试 8](#_Toc8028)

[6.1. 日志法 8](#_Toc30419)

[6.2. Print法 8](#_Toc26060)

[7. 调用外部程序 自定义事件机制模式 8](#_Toc15157)

[7.1. 临时禁用触发器 8](#_Toc3240)

[7.2. leave 关键字吧，此关键字可以模仿 return 的 9](#_Toc14650)

# 定义变量

**[MySQL](http://lib.csdn.net/base/14" \o "MySQL知识库" \t "http://blog.csdn.net/dyzhen/article/details/_blank)**存储过程的declare声明变量必须放在最前面

MySQL存储过程中，定义变量有两种方式：

## 变量名以@开头用户变量 会话变量

1、使用set或select直接赋值，变量名以@开头

例如:

set @var=**1**;

可以在一个会话的任何地方声明，作用域是整个会话，称为用户变量。

## 以declare关键字声明 存储过程变量

2、以declare关键字声明的变量，只能在存储过程中使用，称为存储过程变量，例如：

declare var1 int default **0**;

主要用在存储过程中，或者是给存储传参数中。

两者的区别是：

在调用存储过程时，以declare声明的变量都会被初始化为null。而会话变量（即@开头的变量）则不会被再初始化，在一个会话内，只须初始化一次，之后在会话内都是对上一次计算的结果，就相当于在是这个会话内的全局变量。

## @是用户自定义变量,@@是系统定义变量

## 变量赋值 使用set语句或select

## 表记录赋值到变量 用set或into的方式来实现赋值

方式 2

15

**set** @cnt = (**select** count(\*) **from** test\_tbl);

16

**select** @cnt;

17

18

--  方式 3

19

**select** count(\*) **into** @cnt1 **from** test\_tbl;

20

**select** @cnt1;

21

22

--  多个列的情况下似乎只能用 into 方式

23

**select** **max**(status), avg(status) **into** @**max**, @avg **from** test\_tbl;

24

**select** @**max**, @avg;

## 查询结果存储到变量

把查询结果赋值到变量，大部分情况下使用游标来完成，但是如果明确知道查询结果只有一行（例如统计记录的数量，某个字段求和等），其实可以使用set或into的方式来实现赋值。示例代码：

*多个列的情况下似乎只能用 into 方式*

23

select max(status), avg(status) into @max, @avg from test\_tbl;

24

select @max, @avg;

定义变量以及

declare namex VARCHAR(200);

select city.city\_name into namex from city where city\_id=obj\_id ;

# 参数与返回值

## 设定参数与返回值 `obj\_id` int ，，返回类型 varchar(200)

## 在语句中使用传入的obj\_id参数

直接使用。。

BEGIN

#Routine body goes here...

declare namex VARCHAR(200);

#call log( obj\_id);

select city.city\_name into namex from city where city\_id=obj\_id ;

RETURN namex;

END

## 存储过程返回值使用out参数

out参数 rzt

BEGIN

#Routine body goes here...

set rzt=9;

END

call ret(@rzt);

select @rzt as rzt

# other

## 测试的时候，注意字符串参数要包含双引号

Call xxxx()

使用gui根据测试sp的时候，只填写参数即可。

## Sql参数拼接 CONCAT， 传入like的参数

select \* from wxb\_customer where customer\_name like CONCAT('%',uname ,'%');

## Sql语句调用存储过程call 语句

call query1('mer')

## **游标嵌套** handler **嵌套 独立begin快**

在mysql中，每个begin end 块都是一个独立的scope区域，由于MySql中同一个error的事件只能定义一次，如果多定义的话在编译时会提示Duplicate handler declared in the same block。

多个begin款就可以了

mysql游标的用法及作用 - ```求知若渴，虚怀若谷``` - CSDN博客.html

# 游标与循环

CREATE DEFINER=`root`@`%` PROCEDURE `cursorT`()

BEGIN

#Routine body goes here...

DECLARE $id VARCHAR(30);

DECLARE $余额 VARCHAR(30);

DECLARE 游标 CURSOR FOR select id,balance from fs\_book\_t limit 3;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET @完成=1;

set @完成=0;

OPEN 游标;

REPEAT

/\* this memberno must declare ,,not cant @ derfalut var mode\*/

FETCH 游标 INTO $id,$余额;

select $id,$余额;

-- print(11);

UNTIL @完成 END REPEAT;

CLOSE 游标;

select 'end';

END

fetch是获取游标当前指向的数据行，并将指针指向下一行，当游标已经指向最后一行时继续执行会造成游标溢出。  
使用loop循环游标时，他本身是不会监控是否到最后一条数据了，像下面代码这种写法，就会造成死循环；

read\_loop:loop

fetch cur into n,c; set total = total+c; end loop;

* 1
* 2
* 3
* 4

在MySql中，造成游标溢出时会引发mysql预定义的NOT FOUND错误，所以在上面使用下面的代码指定了当引发not found错误时定义一个continue 的事件，指定这个事件发生时修改done变量的值。

# 异常处理机制

Signal

## 动态SQL

Mysql 支持动态SQL的功能，

set @sqlStr='select \* from table where condition1 = ?'; prepare s1 for @sqlStr; *--如果有多个参数用逗号分隔* execute s1 using @condition1; *--手工释放，或者是 connection 关闭时， server 自动回收* deallocate prepare s1;

# 调试

## 日志法

## Print法

输出 没有print，使用select代替即可

# 调用外部程序 自定义事件机制模式

## 临时禁用触发器

存储过程,触发器debug不易.且触发器隐藏了实际执行的步骤,不易发现问题

mysql不能临时禁用,启用触发器,这点如果做数据迁移,修复会比较麻烦,需要临时drop触发器,可能影响到生产环境.  
由于存储过程,触发器不易测试,或者未做充分测试,一旦升级失败可能导致数据错误,因为已经先drop了存储过程或者触发器.  
不易分析存储过程,触发器的性能.不能通过慢查询日志去分析存储过程,触发器的具体执行情况.仅仅记录call procedure\_name();这样简单的信息;  
触发器可能导致死锁;  
以上只是列举一些问题,具体的使用过程中,mysql的存储过程和触发器离商业产品的距离还有很远

## leave 关键字吧，此关键字可以模仿 return 的

替代 return 的关键字呢？

    没有！

    像 exit, quit 之类的关键字全没有！

    怎么办？

    使用功能稍次一些的 leave 关键字吧，此关键字可以模仿 return 的行为。

作者:: 绰号:老哇的爪子claw of Eagle 偶像破坏者Iconoclast image-smasher

捕鸟王"Bird Catcher kok 虔诚者Pious 宗教信仰捍卫者 Defender Of the Faith. 卡拉卡拉红斗篷 Caracalla red cloak 万兽之王

简称：： Emir Attilax Akbar 埃米尔 阿提拉克斯 阿克巴

全名：：Emir Attilax Akbar bin Mahmud bin attila bin Solomon bin adam Al Rapanui 埃米尔 阿提拉克斯 阿克巴 本 马哈茂德 本 阿提拉 本 所罗门 本亚当 阿尔 拉帕努伊

常用名：艾提拉（艾龙）， EMAIL:1466519819@qq.com

头衔：uke总部o2o负责人，全球网格化项目创始人，

uke交友协会会长 uke捕猎协会会长 Emir Uke部落首席大酋长，

uke宗教与文化融合事务部部长， uke宗教改革委员会副主席

uke制度与重大会议委员会委员长，uke保安部首席大队长,uke制度检查委员会副会长，

uke 首席cto 软件部门总监 技术部副总监 研发部门总监主管 产品部副经理 项目部副经理 uke科技研究院院长 uke软件培训大师

uke波利尼西亚区大区连锁负责人 汤加王国区域负责人 uke克尔格伦群岛区连锁负责人，莱恩群岛区连锁负责人，uke布维岛和南乔治亚和南桑威奇群岛大区连锁负责人

Uke软件标准化协会理事长理事长 Uke 数据库与存储标准化协会副会长

uke终身教育学校副校长 Uke医院 与医学院方面的创始人

uec学院校长， uecip图像处理机器视觉专业系主任 uke文档检索专业系主任

Uke图像处理与机器视觉学院首席院长

Uke 户外运动协会理事长 度假村首席大村长 uke出版社编辑总编

转载请注明来源：attilax的专栏 ?http://blog.csdn.net/attilax

--Atiend v8