Mysql

delimiter $$　　#将语句的结束符号从分号;临时改为两个$$(可以是自定义)

默认的MySQL语句分隔符为分号 ; 。Mysql命令行实用程序也是 ; 作为语句分隔符。如果命令行实用程序要解释存储过程自身的 ; 字符，则他们最终不会成为存储过程的成分，这会使存储过程中的SQL出现句法错误

解决方法是临时更改命令实用程序的语句分隔符

触发器：

分析：

监视谁：ord

监视动作：insert

触发之间：之后（之前）

触发事件：update

创建语法：

delimiter //

Ceate trigger 触发器名称

Before / after （触发时间）

Insert / update /delete （监视动作）

On 表名 （监视对象）

For each row --行级触发器，mysql必须写。此处的含义是监视动作修改了多少行，后面的触发动作执行多少次。

Begin

Declare -- 声明变量

rnum int [default 18] ; 变量名 类型 可选的默认值 与建表的定义一样（python不用只需要赋值）。set 变量，给变量赋值；@变量名 ，再begin和end里面@可以不加。

Select num into rnum from goods where gid = new.gid;

If rnum < new.much then

Set new.much = rnum

End if;

Update goods set num = num –new.much where gid = new.gid; -- new表示插入的新行

……

sqlN

End//

查看触发器：

Show tiggers /G

删除触发器：

Drop trigger triggerName

存储过程： 类似于 python 中的函数。封装一段复杂sql代码,存储再数据库中，方便重复调用。

基本结构

DELIMITER // //定义新的语句分隔符为//

CREATE PROCEDURE productpricing([in/out/inout] 参数名 参数类型)

BEGIN

SELECT Avg(prod\_price) AS priceaverage FROM products;

END //

DELIMITER ; //改回原来的语句分隔符为 ;

除\符号外，任何字符都可以作为语句分隔符

使用productpricing存储过程：CALL productpricing(18，@name); 输出变量位置需要加一个变量来获取输出。如果是inout型变量，则先set @currage = 18，然后在把@currage传进存储过程。

查看已有的procedure： show procedure status;