## 什么是触发器

触发器是与表有关的数据库对象，在满足定义条件时触发，并执行触发器中定义的语句集合。触发器的这种特性可以协助应用在数据库端确保数据的完整性。

举个例子，比如你现在有两个表【用户表】和【日志表】，当一个用户被创建的时候，就需要在日志表中插入创建的log日志，如果在不使用触发器的情况下，你需要编写程序语言逻辑才能实现，但是如果你定义了一个触发器，触发器的作用就是当你在用户表中插入一条数据的之后帮你在日志表中插入一条日志信息。当然触发器并不是只能进行插入操作，还能执行修改，删除。

## 创建触发器

CREATE TRIGGER trigger\_name trigger\_time trigger\_event ON tb\_name FOR EACH ROW trigger\_stmt

trigger\_name：触发器的名称

tirgger\_time：触发时机，为BEFORE或者AFTER

trigger\_event：触发事件，为INSERT、DELETE或者UPDATE

tb\_name：表示建立触发器的表明，就是在哪张表上建立触发器

trigger\_stmt：触发器的程序体，可以是一条SQL语句或者是用BEGIN和END包含的多条语句

所以可以说MySQL创建以下六种触发器：

BEFORE INSERT,BEFORE DELETE,BEFORE UPDATE

AFTER INSERT,AFTER DELETE,AFTER UPDATE

BEFORE和AFTER参数指定了触发执行的时间，在事件之前或是之后

FOR EACH ROW表示任何一条记录上的操作满足触发事件都会触发该触发器

**创建有多个执行语句的触发器**

CREATE TRIGGER 触发器名 BEFORE|AFTER 触发事件

ON 表名 FOR EACH ROW

BEGIN

执行语句列表

END

其中，BEGIN与END之间的执行语句列表参数表示需要执行的多个语句，不同语句用分号隔开

**tips：**一般情况下，mysql默认是以 ; 作为结束执行语句，与触发器中需要的分行起冲突

　　   为解决此问题可用DELIMITER，如：DELIMITER ||，可以将结束符号变成||

　　   当触发器创建完成后，可以用DELIMITER ;来将结束符号变成;

[复制代码](javascript:void(0);)

mysql> DELIMITER ||

mysql> CREATE TRIGGER demo BEFORE DELETE

-> ON users FOR EACH ROW

-> BEGIN

-> INSERT INTO logs VALUES(NOW());

-> INSERT INTO logs VALUES(NOW());

-> END

-> ||

Query OK, **0** rows affected (**0.06** sec)

mysql> DELIMITER ;

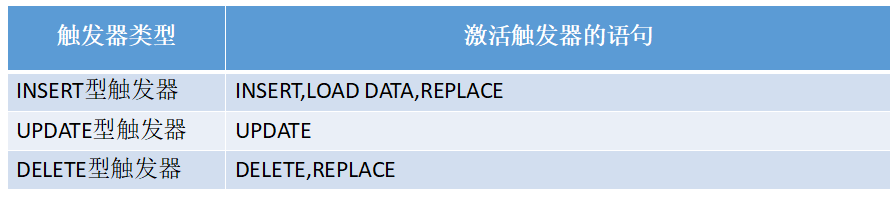
[复制代码](javascript:void(0);)

上面的语句中，开头将结束符号定义为||，中间定义一个触发器，一旦有满足条件的删除操作

就会执行BEGIN和END中的语句，接着使用||结束

最后使用DELIMITER ; 将结束符号还原

tigger\_event：

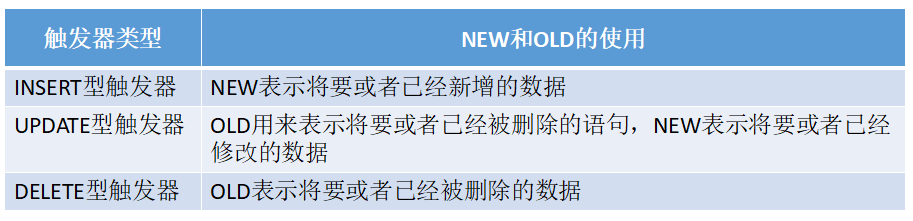


在BEGIN...END语句中也可以定义变量，但是只能在BEGIN...END内部使用：

DECLARE var\_name var\_type [DEFAULT value] #定义变量，可指定默认值

SET var\_name = value #给变量赋值

NEW和OLD的使用:



 根据以上的表格，可以使用一下格式来使用相应的数据：

NEW.columnname：新增行的某列数据

OLD.columnname：删除行的某列数据

## 实践操作

用户users表

CREATE TABLE `users` (

`id` int(**11**) unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(**255**) CHARACTER SET utf8mb4 DEFAULT NULL,

`add\_time` int(**11**) DEFAULT NULL,

PRIMARY KEY (`id`),

KEY `name` (`name`(**250**)) USING BTREE

) ENGINE=MyISAM AUTO\_INCREMENT=**1000001** DEFAULT CHARSET=latin1;

日志logs表：

CREATE TABLE `logs` (

`Id` int(**11**) NOT NULL AUTO\_INCREMENT,

`log` varchar(**255**) DEFAULT NULL COMMENT '日志说明',

PRIMARY KEY (`Id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='日志表';

需求是：当在users中插入一条数据，就会在logs中生成一条日志信息。

创建触发器：

DELIMITER $

CREATE TRIGGER user\_log AFTER INSERT ON users FOR EACH ROW

BEGIN

DECLARE s1 VARCHAR(**40**) character set utf8;

DECLARE s2 VARCHAR(**20**) character set utf8;

SET s2 = " is created";

SET s1 = CONCAT(NEW.name,s2);

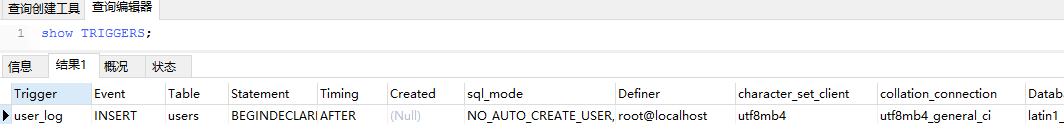
INSERT INTO logs(log) values(s1);

END $

DELIMITER ;

**查看触发器**

**SHOW TRIGGERS语句查看触发器信息**



drop trigger user\_log;#删除触发器

**在triggers表中查看触发器信息**

SELECT \* FROM information\_schema.triggers;

所有触发器信息都存储在information\_schema数据库下的triggers表中

**限制和注意事项**

触发器会有以下两种限制：

1.触发程序不能调用将数据返回客户端的存储程序，也不能使用采用CALL语句的动态SQL语句，但是允许存储程序通过参数将数据返回触发程序，也就是存储过程或者函数通过OUT或者INOUT类型的参数将数据返回触发器是可以的，但是不能调用直接返回数据的过程。

2.不能再触发器中使用以显示或隐式方式开始或结束事务的语句，如START TRANS-ACTION,COMMIT或ROLLBACK。

注意事项：MySQL的触发器是按照BEFORE触发器、行操作、AFTER触发器的顺序执行的，其中任何一步发生错误都不会继续执行剩下的操作，如果对事务表进行的操作，如果出现错误，那么将会被回滚，如果是对非事务表进行操作，那么就无法回滚了，数据可能会出错。

**总结**

触发器是基于行触发的，所以删除、新增或者修改操作可能都会激活触发器，所以不要编写过于复杂的触发器，也不要增加过得的触发器，这样会对数据的插入、修改或者删除带来比较严重的影响，同时也会带来可移植性差的后果，所以在设计触发器的时候一定要有所考虑。

触发器是一种特殊的存储过程，它在插入，删除或修改特定表中的数据时触发执行，它比数据库本身标准的功能有更精细和更复杂的数据控制能力。

数据库触发器有以下的作用：

1.安全性。可以基于数据库的值使用户具有操作数据库的某种权利。

  # 可以基于时间限制用户的操作，例如不允许下班后和节假日修改数据库数据。

  # 可以基于数据库中的数据限制用户的操作，例如不允许股票的价格的升幅一次超过10%。

2.审计。可以跟踪用户对数据库的操作。

  # 审计用户操作数据库的语句。

  # 把用户对数据库的更新写入审计表。

3.实现复杂的数据完整性规则

  # 实现非标准的数据完整性检查和约束。触发器可产生比规则更为复杂的限制。与规则不同，触发器可以引用列或数据库对象。例如，触发器可回退任何企图吃进超过自己保证金的期货。

  # 提供可变的缺省值。

4.实现复杂的非标准的数据库相关完整性规则。触发器可以对数据库中相关的表进行连环更新。例如，在auths表author\_code列上的删除触发器可导致相应删除在其它表中的与之匹配的行。

  # 在修改或删除时级联修改或删除其它表中的与之匹配的行。

  # 在修改或删除时把其它表中的与之匹配的行设成NULL值。

  # 在修改或删除时把其它表中的与之匹配的行级联设成缺省值。

  # 触发器能够拒绝或回退那些破坏相关完整性的变化，取消试图进行数据更新的事务。当插入一个与其主健不匹配的外部键时，这种触发器会起作用。例如，可以在books.author\_code 列上生成一个插入触发器，如果新值与auths.author\_code列中的某值不匹配时，插入被回退。

5.同步实时地复制表中的数据。

6.自动计算数据值，如果数据的值达到了一定的要求，则进行特定的处理。例如，如果公司的帐号上的资金低于5万元则立即给财务人员发送警告数据。