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**](http://lib.csdn.net/base/14)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。

[**MySQL**](http://lib.csdn.net/base/14)触发器语法详解.  
㈠CREATE TRIGGER语法  
CREATE TRIGGER trigger\_name trigger\_time trigger\_event  
    ON tbl\_name FOR EACH ROW trigger\_stmt  
触发程序是与表有关的命名数据库对象，当表上出现特定事件时，将激活该对象。  
触发程序与命名为tbl\_name的表相关。tbl\_name必须引用永久性表。不能将触发程序与  
TEMPORARY表或视图关联起来。  
trigger\_time是触发程序的动作时间。它可以是BEFORE或AFTER，以指明触发程序是在激活它的语句之前或之后触发。  
trigger\_event指明了激活触发程序的语句的类型。trigger\_event可以是下述值之一：  
·         INSERT：将新行插入表时激活触发程序，例如，通过INSERT、LOAD DATA和REPLACE  
语句。  
·         UPDATE：更改某一行时激活触发程序，例如，通过UPDATE语句。  
·         DELETE：从表中删除某一行时激活触发程序，例如，通过DELETE和REPLACE语句。  
请注意，trigger\_event与以表操作方式激活触发程序的SQL语句并不很类似，这点很重要。例如，  
关于INSERT的BEFORE触发程序不仅能被INSERT语句激活，也能被LOAD DATA语句激活。  
可能会造成混淆的例子之一是INSERT INTO .. ON DUPLICATE UPDATE ...语法：BEFORE INSERT  
触发程序对于每一行将激活，后跟AFTER INSERT触发程序，或BEFORE UPDATE和AFTER   
UPDATE触发程序，具体情况取决于行上是否有重复键。  
对于具有相同触发程序动作时间和事件的给定表，不能有两个触发程序。例如，对于某一表，不  
能有两个BEFORE UPDATE触发程序。但可以有1个BEFORE UPDATE触发程序和1个BEFORE   
INSERT触发程序，或1个BEFORE UPDATE触发程序和1个AFTER UPDATE触发程序。  
trigger\_stmt是当触发程序激活时执行的语句。如果你打算执行多个语句，可使用BEGIN ... END复  
合语句结构。这样，就能使用存储子程序中允许的相同语句

㈡DROP TRIGGER语法  
DROP TRIGGER [schema\_name.]trigger\_name  
舍弃触发程序。方案名称（schema\_name）是可选的。如果省略了schema（方案），将从当前方  
案中舍弃触发程序。  
注释：从MySQL 5.0.10之前的MySQL版本升级到5.0.10或更高版本时（包括所有的MySQL 5.1版  
本），必须在升级之前舍弃所有的触发程序，并在随后重新创建它们，否则，在升级之后DROP   
TRIGGER不工作。  
DROP TRIGGER语句需要SUPER权限。

㈢使用触发程序  
在本节中，介绍了在MySQL 5.1中使用触发程序的方法，并介绍了在使用触发程序方面的限制。  
触发程序是与表有关的命名数据库对象，当表上出现特定事件时，将激活该对象。在某些触发程  
序的用法中，可用于检查插入到表中的值，或对更新涉及的值进行计算。  
触发程序与表相关，当对表执行INSERT、DELETE或UPDATE语句时，将激活触发程序。可以将  
触发程序设置为在执行语句之前或之后激活。例如，可以在从表中删除每一行之前，或在更新了  
每一行后激活触发程序。  
要想创建触发程序或舍弃触发程序，可使用CREATE TRIGGER或DROP TRIGGER语句

·         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  
（允许存储程序通过参数将数据返回触发程序）。  
·         触发程序不能使用以显式或隐式方式开始或结束事务的语句，如START TRANSACTION、  
COMMIT或ROLLBACK。  
使用OLD和NEW关键字，能够访问受触发程序影响的行中的列（OLD和NEW不区分大小写）。  
在INSERT触发程序中，仅能使用NEW.col\_name，没有旧行。在DELETE触发程序中，仅能使用  
OLD.col\_name，没有新行。在UPDATE触发程序中，可以使用OLD.col\_name来引用更新前的某一  
行的列，也能使用NEW.col\_name来引用更新后的行中的列。  
用OLD命名的列是只读的。你可以引用它，但不能更改它。对于用NEW命名的列，如果具有  
SELECT权限，可引用它。在BEFORE触发程序中，如果你具有UPDATE权限，可使用“SET NEW.  
col\_name = value”更改它的值。这意味着，你可以使用触发程序来更改将要插入到新行中的值，  
或用于更新行的值。  
在BEFORE触发程序中，AUTO\_INCREMENT列的NEW值为0，不是实际插入新记录时将自动生  
成的序列号。

通过使用BEGIN ... END结构，能够定义执行多条语句的触发程序。在BEGIN块中，还能使用存储  
子程序中允许的其他语法，如条件和循环等。但是，正如存储子程序那样，定义执行多条语句的  
触发程序时，如果使用mysql程序来输入触发程序，需要重新定义语句分隔符，以便能够在触发  
程序定义中使用字符“;”。在下面的示例中，演示了这些要点。在该示例中，定义了1个  
UPDATE触发程序，用于检查更新每一行时将使用的新值，并更改值，使之位于0～100的范围  
内。它必须是BEFORE触发程序，这是因为，需要在将值用于更新行之前对其进行检查：  
mysql> delimiter //  
mysql> CREATE TRIGGER upd\_check BEFORE UPDATE ON account  
    -> FOR EACH ROW  
    -> BEGIN  
    ->     IF NEW.amount < 0 THEN  
    ->         SET NEW.amount = 0;  
    ->     ELSEIF NEW.amount > 100 THEN  
    ->         SET NEW.amount = 100;  
    ->     END IF;  
    -> END;//  
mysql> delimiter ;  
较为简单的方法是，单独定义存储程序，然后使用简单的CALL语句从触发程序调用存储程序。  
如果你打算从数个触发程序内部调用相同的子程序，该方法也很有帮助。  
在触发程序的执行过程中，MySQL处理错误的方式如下：  
·         如果BEFORE触发程序失败，不执行相应行上的操作。  
·         仅当BEFORE触发程序（如果有的话）和行操作均已成功执行，才执行AFTER触发程序。  
·         如果在BEFORE或AFTER触发程序的执行过程中出现错误，将导致调用触发程序的整个语  
句的失败。  
·         对于事务性表，如果触发程序失败（以及由此导致的整个语句的失败），该语句所执行的  
所有更改将回滚。对于非事务性表，不能执行这类回滚，因而，即使语句失败，失败之前所作的  
任何更改依然有效。

例一：  
mysql> CREATE TABLE account (acct\_num INT, amount DECIMAL(10,2));  
mysql> CREATE TRIGGER ins\_sum BEFORE INSERT ON account  
    -> FOR EACH ROW SET @sum = @sum + NEW.amount;