**触发器基础知识**

**1. 概述**

官方解释：触发器是一种特殊的存储过程，它不能被显式地调用，而是在往表中插入记录﹑更新记录或者删除记录时被自动地激活。 所以触发器可以用来实现对表实施复杂的完整性约束。

通俗解释：触发器可以检测表的所有增、删、改操作。并可以根据编程者的逻辑在这些操作进行前或进行后进行一系列其他操作。触发器相当于对表进行监视的眼睛。

**2. 触发器的分类**

SQL Server提供了两种触发器:“Instead of” 和“After” 触发器。

一个表或视图的每一个修改动作(Insert、Update和Delete)都可以有一个“Instead of” 触发器，一个表的每个修改动作都可以有多个“After”触发器。

**2.1 “Instead of”触发器**

* “Instead of”触发器在执行真正“插入”之前被执行。又被称为替代触发器：不执行命令所定义的insert、 delete、update操作,而仅执行触发器中定义的代码。
* “Instead of”触发器会替代所要执行的SQL语句，言下之意就是所要执行SQL并不会“真正执行”。

上例中定义了“trigger学生\_Delete”触发器，该触发器从“delete”表中打印出所要删除的学生.在执行“delete”操作后，会发现“学号 = 4”的学生并未被删除， 原因在于“trigger学生Delete”替代了所要执行的“delete from 学生 where 学号 = 4”语句，而在“trigger学生\_Delete”中并未真正删除学生。

**2.2 “After”触发器**

* “After”触发器在Insert、Update或Deleted语句执行之后被触发。“After”触发器只能用于表。
* “After”触发器主要用于表在修改后（insert、update或delete操作之后），来修改其他表

**3. Inserted和Deleted表**

SQL Server为每个触发器都创建了两个专用表:Inserted表和Deleted表。

* 这两个表由系统来维护，它们存在于内存中而不是在数据库中，可以理解为一个虚拟的表。
* 这两个表的结构总是与被该触发器作用的表的结构相同。
* 触发器执行完成后，与该触发器相关的这两个表也被删除。
* Deleted表存放由于执行Delete或Update语句而要从表中删除的所有行。
* Inserted表存放由于执行Insert或Update语句而要向表中插入的所有行。

临时表为逻辑表，由系统来维护，用户不能对它们进行修改。其结构与源表 (激活触发器的表)结构相同，触发器执行完成后，该临时表会自动被删除。

|  |  |  |
| --- | --- | --- |
| **对表的操作** | **Inserted逻辑表** | **Deleted逻辑表** |
| 增加记录（insert） | 存放增加的记录 | 无 |
| 删除记录（delete） | 无 | 存放被删除的记录 |
| 修改记录（update） | 存放更新后的记录 | 存放更新前的记录 |

**4. 触发器的执行过程**

* 如果一个Insert﹑update或者delete语句违反了约束，那么这条SQL语句就没有执行成功，因此“After”触发器也不会被激活。
* “Instead of” 触发器可以取代激发它的操作来执行。它在Inserted表和Deleted表刚刚建立，其它任何操作还没有发生时被执行。因为“Instead of” 触发器在约束之前执行，所以它可以对约束进行一些预处理。

1. **创建触发器**

**create trigger trigger\_name**

**on {table\_name|view\_name}**

**{After|Instead of} {insert|update|delete}**

**as 相应T-SQL语句**

**本章测试样表和Sql**

**业务场景：有两张表，分为为顾客表Customers和顾客订单表Orders，SQL语句分别如下：**

**创建Customes并初始化**

**CREATE TABLE Customers**

**(**

**CustID VARCHAR(50) NOT NULL, --顾客ID**

**CustName VARCHAR(50),--顾客姓名**

**CustCompany VARCHAR(50) --顾客公司**

**)**

**INSERT INTO Customers VALUES('SXN-DD-01','赵武','A')**

**INSERT INTO Customers VALUES('SXN-DD-02','刘杨','B')**

**INSERT INTO Customers VALUES('SXN-DD-03','张永为','C')**

**INSERT INTO Customers VALUES('SXN-DD-04','李龙飞','D')**

**INSERT INTO Customers VALUES('SXN-FF-01','邓华','E')**

**INSERT INTO Customers VALUES('SXN-HH-01','张涛明','F')**

**CREATE TABLE Orders**

**(**

**CustID VARCHAR(50) NOT NULL, --顾客ID**

**OrdetID VARCHAR(50) )--订单ID**

**INSERT INTO Orders VALUES('SXN-DD-01','SCCCCFFFFFSSOX002')**

**INSERT INTO Orders VALUES('SXN-DD-02','SCCCCFFFFFSSOX0X3')**

**INSERT INTO Orders VALUES('SXN-DD-03','CCCCFFFFFSSOX0X4')**

**INSERT INTO Orders VALUES('SXN-DD-04','CCCCFFFFFSSOX0X5')**

**INSERT INTO Orders VALUES('SXN-DD-05','SCCCCFFFFFSSOX0H6')**

**INSERT INTO Orders VALUES('SXN-DD-06','CCCCFFFFFSSOX0X7')**

**以上是我们常见的建表并插入数据的过程，下边我们把数据删除，体验触发器的作用。**

**Delete Customers**

**Delete Orders**

**体验instead of对表的insert监视**

**CREATE trigger trigger\_Customers**

**on Customers**

**instead of INSERT**

**as**

**declare @li\_count int**

**begin**

**select \* from inserted**

**select @li\_count=count(\*) from inserted**

**print @li\_count**

**end**

**INSERT INTO Customers VALUES('SXN-DD-01','赵武','A')**

**INSERT INTO Customers VALUES('SXN-DD-02','刘杨','B')**

**INSERT INTO Customers VALUES('SXN-DD-03','张永为','C')**

**INSERT INTO Customers VALUES('SXN-DD-04','李龙飞','D')**

**INSERT INTO Customers VALUES('SXN-FF-01','邓华','E')**

**INSERT INTO Customers VALUES('SXN-HH-01','张涛明','F')**

**INSERT INTO Customers VALUES('SXN-DD-01','赵武','A'),('SXN-DD-02','刘杨','B'),('SXN-DD-03','张永为','C'),('SXN-DD-04','李龙飞','D'),('SXN-FF-01','邓华','E'),('SXN-HH-01','张涛明','F')**

**体验以上两个语句在触发器里的不同反应。得出结论。**

**体验After对表的update监视**

**update Customers set custname='aaa' where custname='赵武'**

**Alter trigger trigger\_Customers**

**on Customers**

**After update**

**as**

**declare @li\_count int**

**begin**

**select \* from inserted**

**select @li\_count=count(\*) from inserted**

**print @li\_count**

**end**

**update Customers set custname='bbb' where custname='aaa'**

**update Customers set custname='bbb'**

**inserted和deleted表的存放的数据不一定是一条，也许是多条。**

**体验**after **对表的delete监视**

删除Orders表的纪录时，自动向Orders2表中插入被删除的纪录。

CREATE TABLE Orders2

(

CustID VARCHAR(50) NOT NULL, --顾客ID

OrdetID VARCHAR(50) )--订单ID

create trigger trigger\_delete

on Orders

after DELETE

As

begin

insert Orders2 select \* from deleted

end

以上实现了当对表Orders进行删除操作时，触发器自动将删除的内容插入到Orders2表中。这类应用适合用户操作日志管理。

1. **修改触发器：**

**alter trigger trigger\_name**

**on {table\_name|view\_name}**

**{After|Instead of} {insert|update|delete}**

**as 相应T-SQL语句**

1. **删除触发器：**

**drop trigger trigger\_name**

**8. 查看数据库中已有触发器：**

**8.1 查看数据库中所有触发器**

**select \* from sysobjects where xtype='TR'**

**8.2 查看单个触发器**

**exec sp\_helptext '触发器名'**

作业：只写一个触发器，对表**Customers 进行insert，delete,和update监视，无论哪种操作，都能提示用户进行了**

**何种操作。**