

## 第13章 触发器

Microsoft SQL Server 提供两种主要机制来强制使用业务规则和数据完整性：约束和触发器。触发器可以提供更为复杂的数据完整性约束。

触发器是一种特殊的存储过程，它在特定语言事件发生时自动执行，通常用于实现强制业务规则和数据完整性。

本章的学习要点：

- \* 触发器的概念和特点
- \* 创建触发器

### 13.1 触发器概述

触发器与存储过程非常相似，触发器也是 SQL 语句集，它是通过事件进行触发而被执行的，不能用 EXECUTE 语句调用；而存储过程可以通过存储过程名字而被直接调用。当对某一表进行诸如 UPDATE、INSERT、DELETE 这些操作时，SQL Server 就会自动执行触发器所定义的 SQL 语句，从而确保对数据的处理必须符合由这些 SQL 语句所定义的规则。

#### 1. 触发器的类型

SQL Server 包括三种常规类型的触发器：DML 触发器、DDL 触发器和登录触发器。

##### 1) DDL 触发器

当服务器或数据库中发生数据定义语言 (DDL) 事件时将调用 DDL 触发器。

##### 2) 登录触发器

登录触发器将为响应 LOGON 事件而激发存储过程。与 SQL Server 实例建立用户会话时将引发此事件。

##### 3) DML 触发器

当数据库中发生数据操作语言 (DML) 事件时将调用 DML 触发器。DML 事件包括在指定表或视图中修改数据的 INSERT 语句、UPDATE 语句或 DELETE 语句。

在数据修改时，DML 触发器是强制业务规则的一种很有效的方法。一个表最多有三种不同类型的触发器，当 UPDATE、DELETE、INSERT 发生时分别使用一个触发器。

本节中主要介绍 DML 和 DDL 触发器。

#### 2. 触发器的功能

触发器可以用于实现以下功能：

##### 1) 强化约束 (Enforce restriction)

触发器的主要作用就是其能够实现由主键和外键所不能保证的复杂的参照完整性和数据的一致性。与 CHECK 约束不同，触发器可以引用其他表中的列，因此，触发器可实现更为复杂的约束。

##### 2) 跟踪变化 (Auditing changes)

触发器可以侦测数据库内的操作，从而不允许数据库中未经许可的特定更新和变化。

##### 3) 级联运行 (Cascaded operation)

触发器通过侦测数据库内的操作，能自动地级联影响到整个数据库的各项内容。如：A 表的触发器中包含对 B 表的数据操作（如插入、修改、删除），而该操作又导致 B 表上的触发器被触发。

##### 4) 存储过程的调用 (Stored procedure invocation)

为了响应数据库更新，触发器可以调用一个或多个存储过程，甚至可以调用外部存储过

程，从而在 DBMS 之外进行操作。

由此可见，触发器可以实现更高级形式的业务规则、复杂行为限制等功能。DML 触发器可以查询其他表，还可以包含复杂的 Transact-SQL 语句。

在 DBMS 中，将触发器和触发它的语句作为可在触发器内回滚的单个事务对待。如果检测到错误（例如，磁盘空间不足），则整个事务即自动回滚。

### 3. 与存储过程的区别

触发器与存储过程主要的区别在于触发器的运行方式。存储过程必须由用户、应用程序或者触发器来显式地调用并执行，而触发器是当特定事件出现的时候，自动执行或者激活的，与连接到数据库中的用户或者应用程序无关。

---

说	尽管触发器的功能强大，但是它们也可能对服务器的性能有影响。因此，要注意不要在触发器
明：	中放置太多的功能，因为它将降低响应速度，使用户等待的时间增加。

---

## 13.2 DML 触发器

### 13.2.1 DML 触发器的概述和作用

DML 触发器是当数据库服务器中发生数据操作语言（DML）事件时将触发 DML 触发器。DML 触发器可以查询其他表，还可以包含复杂的 T-SQL 语句。

DML 触发器在以下方面非常有用：

1. DML 触发器可通过数据库中的相关表实现级联更改。不过，通过参照完整性约束可以更有效地进行这些更改。
2. DML 触发器可以防止恶意或错误的 INSERT、UPDATE 以及 DELETE 操作，并强制执行比 CHECK 约束定义的限制更为复杂的其他限制。
3. DML 触发器可以评估数据修改前后表的状态，并根据该差异采取措施。
4. 一个表中的多个同类 DML 触发器（INSERT、UPDATE 或 DELETE）允许采取多个不同的操作来响应同一个修改语句。

### 13.2.2 DML 触发器分类

SQL Server 中的 DML 触发器按被触发器的时机可以分为以下 2 种类型。

#### 1. AFTER 触发器

又称后触发器。在执行了 INSERT、UPDATE 或 DELETE 语句操作之后执行 AFTER 触发器。如果仅指定 FOR 关键字，则 AFTER 为默认值。AFTER 触发器只能在表上指定，可以为任何一个 DML 操作定义多个 AFTER 触发器。

#### 2. INSTEAD OF 触发器

又称替代触发器。INSTEAD OF 触发器在数据变动之前被触发，代替引起触发器执行的 T-SQL 语句，即 INSTEAD OF 触发器执行时并不执行所定义的 INSERT、UPDATE 或 DELETE 操作，而仅执行触发器本身。INSTEAD OF 触发器可在表或视图上定义，用 INSTEAD OF 触发器可以屏蔽原来的 SQL 语句，而转向执行触发器内部的 SQL 语句。不能为表或视图的 DML 操作创建多余一个的 INSTEAD OF 触发器。

INSTEAD OF 触发器的主要优点是可以使不能更新的视图支持更新。基于多个基表的视图必须使用 INSTEAD OF 触发器来支持引用多个表中数据的插入、更新和删除操作。例如，通常不能在一个基于联接的视图上进行 DELETE 操作。然而，可以编写一个 INSTEAD

OF DELETE 触发器来实现删除。根据以上所述，表 13-1 对 AFTER 触发器和 INSTEAD OF 触发器的功能进行了比较。

表 13-1 AFTER 触发器和 INSTEAD OF 触发器的功能比较

	AFTER 触发器	INSTEAD OF 触发器
适用范围	表	表和视图
每个表或视图可包含触发器的数量	每个触发操作（UPDATE、DELETE 和 INSERT）可包含多个触发器	每个触发操作（UPDATE、DELETE 和 INSERT）包含至多一个触发器
执行	晚于： <ul style="list-style-type: none"><li>● 约束处理</li><li>● 声明性引用操作</li><li>● 创建 inserted 和 deleted 表</li><li>● 触发操作</li></ul>	早于： <ul style="list-style-type: none"><li>● 约束处理</li><li>● 替代：</li><li>● 触发操作</li></ul> 晚于： <ul style="list-style-type: none"><li>● 创建 inserted 和 deleted 表</li></ul>
执行顺序	可指定第一个和最后一个执行	不适用

13.2.3 与 DML 触发器相关的逻辑表

SQL Server 为每个 DML 触发器都创建了两特殊的表：Deleted 表和 Inserted 表。这是两个逻辑表，由系统来创建和维护，用户不能对他们进行修改，他们存放在内存而不是数据库中。这两张表中包含了在激发触发器的操作中插入或删除的所有记录，因此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会被删除。

Inserted 表用于存储 INSERT 和 UPDATE 语句所影响的行的副本。在插入或更新事务期间，新行将同时被添加到 Inserted 表和触发器表（即对其尝试执行了用户操作的表）。Inserted 表中的行是触发器表中的新行的副本。

Deleted 表用于存储 DELETE 和 UPDATE 语句所影响的行的副本。在执行 DELETE 或 UPDATE 语句的过程中，行从触发器表中删除，并传输到 Deleted 表中。Deleted 表和触发器表通常没有相同的行。

说明：	一个 UPDATE 事务可以看作先执行一个 DELETE 操作，再执行一个 INSERT 操作，旧的行首先被移动到 Deleted 表，然后新行同时插入触发器表和 Inserted 表。
-----	---

表 13-2 对 Deleted 表、Inserted 表在执行三种数据操作时记录的变化情况。

表 13-2 Deleted 表、Inserted 表在执行触发器时记录变化情况

T-SQL 语句	Deleted 表	Inserted 表
INSERT	空	新增加的记录
UPDATE	旧记录	新纪录
DELETE	删除的记录	空

## 13.2.4 创建 DML 触发器

创建 DML 触发器时需指定：触发器名称、定义触发器时所基于的表或视图、触发器被触发的时间、激活触发器的数据修改语句（INSERT、UPDATE 或 DELETE）和执行触发器操作的编程语句。其中，多个数据修改语句可激活同一个触发器。例如，触发器可由 INSERT 或 UPDATE 语句激活。

### 1. 语法：

```
CREATE TRIGGER trigger_name
ON { OBJECT NAME }
{ FOR → AFTER → INSTEAD OF }
{ [INSERT] [,] [UPDATE] [,] [DELETE] }
AS
{ sql_statement [ ...n ] }
```

### 2. 参数摘要与说明

- 1) trigger\_name：指定触发器名称
- 2) OBJECT NAME：要对其执行 DML 触发器的表或视图
- 3) { FOR → AFTER → INSTEAD OF }：指定触发器的类型，如果仅指定 FOR 关键字，则 AFTER 是默认值
- 4) {[INSERT][,][UPDATE][,][DELETE]}：指定激活触发器的数据修改语句，必须至少指定一项，在触发器定义中允许使用上述选项的任意顺序组合。
- 5) sql\_statement：指定触发器所指定的 T-SQL 语句。

---

说明： DML 触发器是根据数据修改语句来检查或更改数据，不向用户返回数据。

---

### 3. 创建触发器的注意事项

- 1) CREATE TRIGGER 语句必须是批处理中的第一个语句，该语句后面的所有其他语句被解释为 CREATE TRIGGER 语句定义的一部分。
- 2) 创建 DML 触发器的权限默认分配给表的所有者，且不能将该权限转给其他用户。
- 3) 触发器的名称必须遵循标识符的命名规则。
- 4) 只能在当前数据库中创建 DML 触发器，但不能对临时表或系统表创建 DML 触发器。
- 5) 使用 UPDATE 语句可以一次对多行数据进行修改，但不管修改了多少行数据，触发器都只触发一次。
- 6) 在执行修改语句过程中，触发器的执行是修改语句事务的一部分。因此，如果触发器执行不成功，整个事务将回滚。
- 7) 在 DML 触发器中不允许使用下列 Transact-SQL 语句：CREATE /ALTER/ DROP DATABASE、CREATE/DROP INDEX、DROP TABLE；用于执行以下操作的 ALTER TABLE：添加、修改或删除列、添加或删除 PRIMARY KEY 或 UNIQUE 约束。
- 8) 当使用约束、默认值就可以实现预定的数据完整性时，应优先考虑这些措施。因为触发器性能通常比较低。
- 9) TRUNCATE TABLE 语句类似于不带 WHERE 子句的 DELETE 语句（用于删除所有行），但它并不会触发 DELETE 触发器，因为 TRUNCATE TABLE 语句没有日志记录。

**例 13- 1:** 创建一个插入触发器。功能：在 Singer 表中添加一条记录时，触发器向客户端显示一条提示信息。语句如下：

```
create trigger trg_singer on Singer
after insert as raiserror('你向 Singer 表添加了一条新纪录',16,10)
```

为了验证该触发器的作用，可执行以下的插入语句：

```
insert into singer(singerid,name,nation)
values('GC004','刘欢','中国')
```

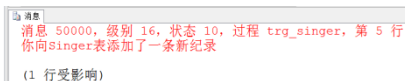


图 13-1 验证 trg\_singer 触发器

在向 singer 表插入新纪录后，显示了触发器中定义的提示信息。如图 13-1 所示。

**例 13-2：**在 Songs 表上创建一个 UPDATE 触发器，使得在对其做更改操作时，显示两张逻辑表 Inserted 和 Deleted 表中的内容。代码如下：

```
create trigger trg_song on songs
after update
as
--显示 deleted 表的内容
select * from deleted
--显示 inserted 表的内容
select * from inserted
```

要验证触发器的作用，执行以下语句：

```
update songs set Composer='小柯'
where songid='S0104'
```

执行结果如图 13-2 所示，从本例可验证，在 UPDATE 操作时，Deleted 表中存放的是旧的记录，而 Inserted 表中存放的是新纪录。



SongID	Name	Lyricist	Composer	Lang
1	S0104	因为爱情	NULL	中文

SongID	Name	Lyricist	Composer	Lang
1	S0104	因为爱情	小柯	中文

图 13-2 显示 Deleted、Inserted 表的内容

**例 13-3：**创建一个触发器，在 Track 表上做插入操作时，判断 Circulation 是否小于 5，如果小于 5，则拒绝插入。

代码如下：

```
create trigger trg_circulation
on track
after insert
as
--判断拟插入行的 Circulation 是否小于
if (select Circulation from inserted)<5
begin
raiserror('Circulation 不能小于 5',16,1)
rollback
end
```

验证该触发器，输入以下代码：

```
insert into dbo.Track(songid,singerid,circulation)
values('S0001','GC002',4)
```

因为不符合插入条件，插入操作被终止。如图 13-3 所示。

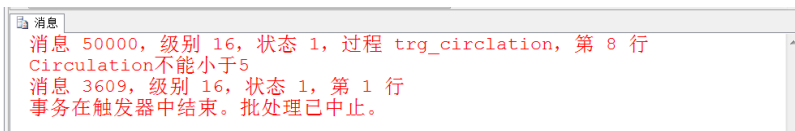


图 13-3 验证 trg\_circulation 触发器

如果执行

```
insert into dbo.Track(songid,singerid,circulation)
values('S0001','GC002',8)
```

则可以成功插入这条记录。

**例 13-4：** 如果用户希望删除 Songs 表中的一条记录，因为在 Songs 表和 Track 表之间有参照完整性约束，因此要求必须先删除子表（即 Track 表）的相关记录，再删除主表（即 Songs）表中的记录。

可以利用在 Songs 表上建立一个 INSTEAD OF 触发器，将 Songs 表上的删除操作替代为依次删除 Track 表和 Songs 中的相应记录。代码如下：

```
create trigger trg_delete on Songs
instead of delete
as
--删除 track 表中的相应记录
delete from track where songid in(select songid from deleted)
--删除 Songs 表中的相应记录
delete from songs where songid in(select songid from deleted)
```

当执行如下代码时，会触发 Instead of 触发器，从而执行触发器中的 T-SQL 语句。

```
delete from songs where songid='S0001'
```

**例 13-5：** 创建一个触发器，在修改 SC 表中的 grade 列时，判断平均成绩是否大于 80，如果大于 80，拒绝该修改操作。

代码如下：

```
CREATE TRIGGER trgUpdateSC ON dbo.SC
FOR UPDATE
AS
begin
    IF UPDATE (Grade)
    BEGIN
        --判断平均 grade 是否大于 80，如果大于 80，回滚
        if(select avg(Grade) from dbo.SC)>80
        begin
            print 'The average value of Grade cannot be more than 80'
            rollback
        end
    end
end
```

上例中，update(column)返回一个布尔值，指示是否对表或视图的指定列进行了 INSERT 或 UPDATE 尝试。可以在 T-SQL 的 INSERT 或 UPDATE 触发器主体中的任意位置使用 UPDATE()，以测试触发器是否应执行某些操作。

**例 13-6：** 创建一个触发器，不允许对 SC 表做删除操作。代码如下：

```
CREATE TRIGGER trgDeleteSC ON dbo.SC
FOR delete
AS
```

```
PRINT 'Deletion of SC is not allowed'
ROLLBACK TRANSACTION
```

**例 13-7：**假设在数据库中以下代码创建了视图 `vwSal`，该视图基于三张表。

```
create view vwSal
as SELECT student.sno,sname,sdept, Course.Cno,Cname,SC.Grade
FROM student INNER JOIN SC ON student.sno= SC.Sno
join Course on Course.Cno= SC.Cno
```

如果用户希望在该视图上做如下修改操作，

```
update vwSal set sdept='ma',Grade=60 where sno='01'
```

系统会报错：“消息 4405，级别 16，状态 1，第 1 行。视图或函数'vwSal' 不可更新，因为修改会影响多个基表。”在此情况下，可以考虑使用在视图上创建 `Instead of` 触发器，屏蔽对视图的更新操作，而转为对两张基表的更新操作。代码如下：

```
create trigger trgView on dbo.vwSal
instead of update
as
begin
--更新 student 表
update dbo.student set sdept=(select sdept from inserted)
where sno=(select sno from inserted)
--更新 SC 表
update dbo.SC set Grade=(select Grade from inserted)
where sno=(select sno from inserted)
end
```

## 13.3 DDL 触发器

### 13.3.1 DDL 触发器概述

DDL 触发器当服务器或者数据库中发生数据定义语言（DDL，CREATE、ALTER 和 DROP）事件时将被触发。如果要执行以下操作，可以使用 DDL 触发器：

1. 要防止对数据库架构进行某些更改。
2. 希望数据库中发生某种情况以响应数据库架构中的更改。
3. 要记录数据库架构中的更改或者事件。

### 13.3.2 创建 DDL 触发器

#### 1. 语法

```
CREATE TRIGGER trigger_name
ON { ALL SERVER → DATABASE }
[ WITH <ddl_trigger_option> [ ,...n ] ]
{ FOR → AFTER } { event_type → event_group } [ ,...n ]
AS { sql_statement [ ; ] [ ,...n ] }
```

#### 2. 参数说明：

除以下参数外，其余参数的说明同 DML 触发器。

1) DATABASE :将 DDL 触发器的作用域应用于当前数据库。如果指定了此参数，则只要当前数据库中出现 *event\_type* 或 *event\_group*，就会激发该触发器。

2) ALL SERVER :将 DDL 触发器的作用域应用于当前服务器。如果指定了此参数，则只要当前服务器中的任何位置上出现 *event\_type* 或 *event\_group*，就会激发该触发器。

3) *event\_type* : 执行之后将导致激发 DDL 触发器的 T-SQL 语言事件的名称。

4) *event\_group* : 预定义的 T-SQL 语言事件分组的名称。执行任何属于 *event\_group* 的 T-SQL 语言事件之后，都将激发 DDL 触发器。

**例 13- 8:** 为 “Music” 数据库创建 DDL 触发器，用于禁止对数据库中的表进行删除和修改操作。

```
CREATE TRIGGER trg_safe ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
PRINT 'You must disable Trigger " trg_safe" to drop or alter tables!'
ROLLBACK
```

当对表做删除操作时，触发该 DDL 触发器的结果如图 13- 4 所示。

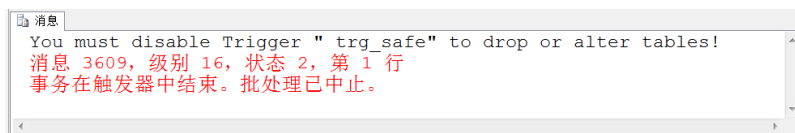


图 13- 4 DDL 触发器

---

说明:	DML 触发器创建成功后，可以在“对象资源管理器”中的“数据库”→“表”→“触发器”节点下找到对应的触发器。
	DDL 触发器创建成功后，可以在“对象资源管理器”中的“数据库”→“可编程性”→“数据库触发器”节点下找到对应的触发器。

---

## 13.4 管理触发器

本节介绍如何对已存在触发器进行管理，例如对触发器的查看、修改、删除等。

### 1. 查看触发器

可以把触发器看作是特殊的存储过程，因此所有适用于存储过程的管理方式都适用于触发器。可以使用像 *sp\_helptext*、*sp\_help* 和 *sp\_depends* 等系统存储过程来查看触发器的有关信息，也可以使用 *sp\_rename* 系统存储过程来重命名触发器。例如，使用 *sp\_helptext* 系统存储过程可以查看触发器的定义语句，如下所示：

```
exec sp_helptext trg_delete
```

执行结果如图 13- 5 所示。

### 2. 修改触发器



修改现有触发器的定义可以使用 ALTER TRIGGER 语句。具体语法格式如下所示：

```
ALTER TRIGGER trigger_name
ON { table → view }
{ { FOR → AFTER → INSTEAD OF }
{ [DELETE] [,] [INSERT] [,] [UPDATE] }
AS
{sql_statement}}
```

修改触发器语句 ALTER TRIGGER 中各参数的含义与创建触发器 CREATE TRIGGER 时相同,这里不再重复说明。

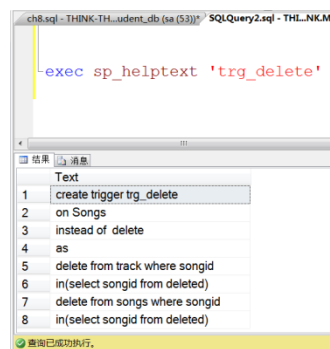


图 13-5 查看触发器信息

说明：一旦使用 WITH ENCRYPTION 对触发器加密,即使是数据库所有者也无法查看或者修改触发器。

### 3. 删除触发器

当不再需要某个触发器时,可以删除它。触发器删除时,触发器所在表中的数据不会因此改变。当某个表被删除时,该表上的所有触发器也会自动被删除。

使用 DROP TRIGGER 语句可以删除当前数据库中的一个或者多个触发器。例如删除触发器 “Trg\_delete”,就可以执行如下代码：

```
DROP TRIGGER Trg_delete
```

### 4. 禁用触发器

用户可以禁用、启用一个指定的触发器或者一个表的所有触发器。当禁用一个触发器后,它在表上的定义仍然存在。但是,当对表执行 INSERT、UPDATE 或者 DELETE 语句时,并不执行触发器的动作,直到重新启动触发器为止。

#### 1) 禁用对表的 DML 触发器

例如,使用语句禁用在数据库中 “Songs” 表创建的触发器 trg\_delete:

```
DISABLE TRIGGER trg_delete ON Songs
```

#### 2) 禁用对数据库的 DDL 触发器

下面的语句禁用一个数据库作用域的 DDL 触发器 trig\_DDL:

```
DISABLE TRIGGER trig_DDL ON DATABASE
```

#### 3) 禁用以同一作用域定义的所有触发器

以下示例禁用在服务器作用域中创建的所有 DDL 触发器:

```
DISABLE TRIGGER ALL ON ALL SERVER
```

禁用之后的启用操作,应该使用语句 ENABLE TRIGGER,该语句的参数与对应的禁用语句相同。

## 习题 13

1. 一个化妆品公司维护三张表：一张存储商标的产品,称为 BrandItems 表；一张存储本地制造的产品,称为 LocalItems 表；一张用于交易,称为 ItemsSold 表。当产品出售时,通过在表中加入一行把交易记录在 ItemsSold 表中。在出售物品时为更新产品的现有数量,需创建以下触发器中哪个？

- A. 用 ItemsSold 表的更新触发器来更新两张产品表。
- B. BrandItems 表和 LocalItems 表的更新触发器。

C. BrandItems 表和 LocalItems 表的插入触发器。

D. ItemsSold 表的插入触发器来更新两张产品表。

2. 考察关于销售表（sales table）的以下触发器：

Create trigger trgDelSales On Sales For delete As Rollback transaction

预测发出以下命令时的输出：Delete sales Where datepart(yy,tran\_dt.<1990。

A. 将从销售表中删除 1990 年中出售的项目的销售材料。

B. 将从销售表中不删除任何记录。.

C. 将从销售表中删除 1990 年和以前出售的项目的销售材料。

D. 将从销售表中删除 1990 年以前出售的项目的销售材料。

3. 简述触发器的主要用途和优缺点。

4. 简述触发器的分类和各种触发器的主要特点。

5. DML 触发器使用 Deleted 和 Inserted 临时表存储什么内容？

6. 编程题

SC 的结构为 SC（sno，cno，grade），在该结构上建立第 1），2）题的触发器：

1）创建限制更新数据的触发器，限制将 SC 表中不及格学生的成绩改为及格。

2）创建限制删除的触发器，限制删除 SC 表中成绩不及格学生的选课记录。

3）创建一个 update 触发器，功能：如果修改了 dbo.student 的 sage 字段，若 sage 的值不在 20-30 范围之内，则显示“年龄需在 20-30 之间”，并回滚。

