کار با trigger ها

Trigger ها نوع خاصی از برنامه ی ذخیره شده می باشند که می توانند برای عمل روی فعالیت یک جدول مانند INSERT, UPDATE, یا DELETE نوشته شوند. اگر زیاد از حد استفاده شوند، به طور بالقوه می توانند منجر به عملکردهای اجرایی مانند بلاک کردن (blockin) شوند و اگر به درستی نوشته نشوند ممکن است شما داده ای را از دست بدهید.

توضیحات

Trigger ها متداولا برای اجرای عملکردهای حسابرسی استفاده می شوند تا یکپارچگی جدول را در محل محدودیت های محلی مانند کلیدهای خارجی برقرار کنند و محدودیت ها را چک کرده و دیگر پردازش های post DML را اجرا کنند. Trigger ها در حوزه ی یک تراکنش (transaction) عمل می کنند، بنابراین اگر یک جدول آپدیت شود، آپدیت اتفاق افتاده و trigger برانگیخته می شود. در زمانیکه trigger در حال کار است، تراکنش اتفاق نمی افتد، تا زمانیکه trigger کامل شود (یا در صورت شکست به عقب برگردانده می شود.). اگر پردازش های زیادی در trigger در حال انجام باشد، قفل ها نیز حفظ می شوند تا زمانیکه trigger کامل شود. نکته مهم این است که: trigger ها طول زندگی تراکنش را افزایش می دهند. همچنین به خاطر حالت پنهان خود می توانند عیب یابی داده را مشکل و خسته کننده سازند.

استفاده از trigger برای انجام بررسی یکپارچگی احتمالا نظر خوبی نباشد، زیرا آنها طول زنگی تراکنش را افزایش می دهند. همچنین اگر یک نقص وجود داشته باشد، یک ROLLBACK باید روی هرکدام از داده های اصلاح شده اتفاق بیفتد هنگامی که برنامه منتظر تکمیل rollback می باشد، می تواند منجر به یک تنگنا در اجرا شود. در مقابل محدودیت های داخلی چک کردن های خود را قبل از هرگونه اصلاح انجام می دهند واین باعث می شود در صورت وجود نقص، ROLLBACK اتفاق نیفتد.

وقتی trigger ها برانگیخته می شوند، جدول های مجازی وجود دارند که مقادیر داده را قبل و بعد از اصلاح در خود حفظ می کنند. این جدول ها inserted و deleted نامیده می شوند. در هنگام دسترسی به این جدول های مجازی در داخل کد trigger، باید روی داده های آنها به عنوان یک مجموعه کار کنید. یک اشتباه متداول که بارها و بارها در کد trigger مشاهده کرده ام: یک trigger با فرض بر اینکه همیشه در زمان در یک ردیف مجزا کار می کند، نوشته می شود. این مودر درست نمی باشد.

در نمونه ی کد زیر یک آپدیت چند ردیفی اجرا می شود، اما trigger نوشته شده است، بنابراین آپدیت یک ردیف مجزا انتظار می رود.

```
use tempdb
go
create table t1 (id int primary key, t1_value varchar(50))
insert into t1 select 1, 'value1'
insert into t1 select 2, 'value2'
insert into t1 select 3, 'value3'
create table t2 (id int primary key, t2_value varchar(50))
```

```
insert into t2 select 1, NULL
insert into t2 select 2, NULL
insert into t2 select 3, NULL
go
create trigger update_t2 on t1
for update
as
begin
 set nocount on

 declare @id int, @t1_value varchar(50)

 select @id = id, @t1_value = t1_value from inserted

 update t2
 set t2_value = @t1_value
 where id = @id

end
go
update t1
set t1_value = cast(id as varchar(50))
go
```

با امتحان داده مشاهده می کنیم که trigger روی همه ی ردیف های آپدیت شده به درستی کار نمی کند.

	id	t1_value
1	1	1
2	2	2
3	3	3

	id	t2_value
1	1	1
2	2	NULL
3	3	NULL

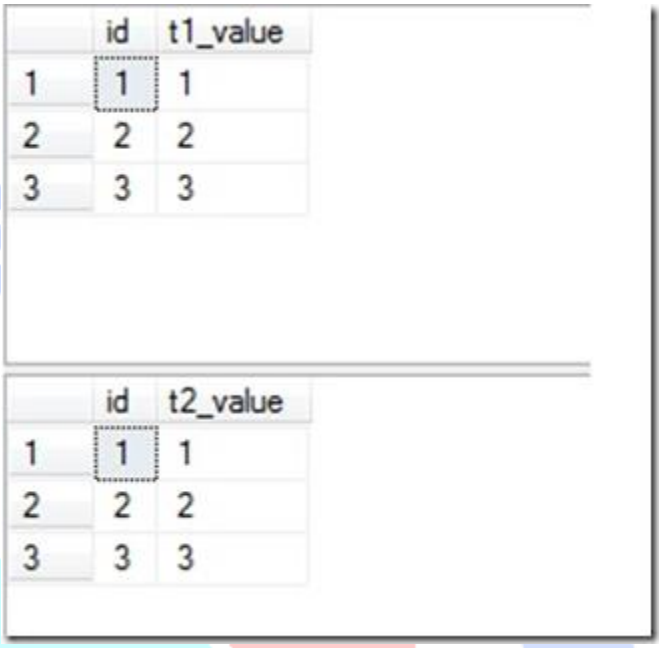
ممکن است با خود فکر کنید که این امر خوب است زیرا که فقط یک ردیف را در برنامه ی خود آپدیت کرده اید. اما اگر داده نیاز به آپدیت دستی از طریق Management Studio، یا یک برچسب patch و یا برچسب تبدیل داشته باشد، چه؟ ممکن است این نقص پنهان داده را متناقض نشان دهد.

همه ی trigger های خود را با فرض اینکه یک ردیف تحت تاثیر قرار خواهد گرفت، بنویسید. مانند بازنویسی trigger زیر:

```
alter trigger update_t2 on t1111111
for update
as
begin
 set nocount on

 update t2
 set t2_value = i.t1_value
 from inserted as i
 inner join t2 on t2.id = i.id

end
go
update t1
set t1_value = cast(id as varchar(50))
go
```



	id	t1_value
1	1	1
2	2	2
3	3	3

	id	t2_value
1	1	1
2	2	2
3	3	3

من تمایل دارم استفاده از trigger ها را به عملکردهای خیلی خاص تغییر دهم، زیرا مسائل عیب یابی داده با تعداد زیادی از آنها، می تواند اجرای بسیار سختی داشته باشد.