

Re: help – transaction control

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-07/3164.html>

From: Viviana (*anonymous_at_discussions.microsoft.com*)

Date: 07/14/04

Date: Wed, 14 Jul 2004 08:26:52 -0700

Andrew:

I have the same problem that Roger.

I try to do that you say and still hapening the message 266

Is there any solution for that ?? For example assigning a name for transactions ??

>-----*Original Message*-----

>*Roger,*

>

>*Yes transactions are nested but will always depend on the outer most*

>*transaction to successfully commit before any of the inner ones are*

>*committed. If a Rollback is issued anywhere along the way ALL transactions*

>*are immediately rolled back. This is be design and is how it has always*

>*worked. So that is why it is imperative that before you issue a Commit or*

>*Rollback you check the @@TRANCOUNT variable to see if there is an open tran*

>*and avoid these errors.*

>

>*IF @@TRANCOUNT > 0*

> *COMMIT TRAN*

>

>*or*

>

>*IF @@TRANCOUNT > 0*

> *ROLLBACK TRAN*

>

>

>

>--

>*Andrew J. Kelly SQL MVP*

>

>

```
>"roger" <xrsr@rogerware.com> wrote in message
>news:Xns9525D7095B374rsrrogerwarecom@216.148.227.77...
>>
>> I need to create (lots of) stored procedures that
>> operate transactionally, but may call each other or
>> be called as part of a larger operation.
>>
>> My understanding of the way that transactions nest
>> is that I should be able to have each SP
>> begin and commit a transaction within itself,
>> and if the SP happens to be called from some context
>> that has begun a transaction, then the SP would
participate
>> in that transaction context.
>> And that's exactly what I want.
>>
>> Like here's a generic sort of structure of an SP then:
>>
>> drop procedure trans_test
>> go
>>
>> create procedure trans_test
>> as
>> begin
>> begin tran
>> print 'trancount 1= ' + str(@@trancount)
>> -- do stuff...
>> if @@error <> 0
>> goto error -- something wrong, abort the
transaction
>> -- everything OK, commit the transaction, or
decrement trancount
>> commit tran
>> print 'trancount 2= ' + str(@@trancount)
>> return 0
>> error:
>> print 'error trancount 1= ' + str(@@trancount)
>> rollback tran
>> print 'error trancount 2= ' + str(@@trancount)
>> return -1
>> end
>> go
>>
>> begin tran
>> declare @status int
>> exec @status = trans_test
>> if @status = 0 begin
>> print 'commit 1'
>> commit tran
>> end
>>
```

>>
>> *And this seems to work right if trans_test does its
commit.*
>> *But, if instead it takes the goto error path and rolls
back*
>> *the transaction, then I get the following messages
printed via isqlw*
>>
>> *trancount 1= 2*
>> *error trancount 1= 2*
>> *Server: Msg 266, Level 16, State 2, Procedure
trans_test, Line 16*
>> *Transaction count after EXECUTE indicates that a
COMMIT or ROLLBACK*
>> *TRANSACTION statement is missing. Previous count = 1,
current count = 0.*
>> *error trancount 2= 0*
>>
>> *As far as the transaction handling goes, it does seem
>> that my SP is doing the right thing, but what's with
the messages?*
>>
>> *I understand that error handling is problematic with T-
SQL, but*
>> *I think I'm working within its capabilities here.*
>>
>> *Any ideas, or tips?*
>>
>> *Thanks*
>>
>>
>
>
>
>
>