**[SQL Server 中的嵌套事务与@@TranCount（转）](http://www.cnblogs.com/kingcat/archive/2011/12/23/2299823.html)**

    在处理事务的时候，一般都用RollBack Transaction来回滚，但是如果在嵌套事务中这样使用的话，就会出现错误。

在SqlServer里，嵌套事务的层次是由@@TranCount全局变量反映出来的。每一次Begin Transaction都会引起@@TranCount加1。而每一次Commit Transaction都会使@@TranCount减1，而RollBack Transaction会回滚所有的嵌套事务包括已经提交的事务和未提交的事务，而使@@TranCount置0。例如：  
Begin Transaction -- @@TranCount = 1  
         BeginTransaction -- @@TranCount = 2     
                  BeginTransaction -- @@TranCount = 3  
                  Commit Transaction -- @@TranCount = 2  
         Commit Transaction -- @@TranCount = 1  
Commit Transaction -- @@TranCount = 0

如果出现错误ROLLBACK TRANSACTION   
则：  
Begin Transaction -- @@TranCount = 1  
         BeginTransaction -- @@TranCount = 2     
                  BeginTransaction -- @@TranCount = 3  
                  ROLLBACK TRANSACTION  -- @@TranCount = 0  
         Commit Transaction -- @@TranCount = 0---出现错误  
Transaction count after EXECUTE indicates that a COMMIT or ROLLBACK TRANSACTION statement is missing. Previous count = 1, current count = 0.  
         如果被嵌套的事务中发生错误，最简单的方法应该是无论如何都先将它提交，同时返回错误码（一个正常情况不可能出现的代码 如 -1）让上一层事务来处理这个错误，从而使@@TranCount 减1。 这样外层事务在回滚或者提交的时候能够保证外层事务在开始的时候和结束的时候保持一致。由于里层事务返回了错误码，因此外层事务（最外层）可以回滚事务，这样里面已经提交的事务也可以被回滚而不会出现错误。

         在项目中应该会常常出现这样的情况，一个存储过程里面用了事务，但是不能保证它会被别的带有事务的存储过程调用，如果单独调用的话，出现错误可以直接回滚，但是如果是被别的带事务的存储过程调用的话，RollBack 就会出错了。因此需要一种机制来区分，建立一个临时的变量来区分是否嵌套，和嵌套的层数，如下：  
  
DECLARE @TranCounter INT;  
    SET @TranCounter = @@TRANCOUNT;  
    IF @TranCounter > 0  
        SAVE TRANSACTION ProcedureSave;  
    ELSE  
        BEGIN TRANSACTION;  
…………  
--事务内要执行的代码  
…………

IF @@ERROR<>0  
goto Error  
　Commit Transaction  
Commit Transaction  
--下面返回要返回的值0只是个例子  
Return 0

Error:  
     IF @TranCounter = 0  
       ROLLBACK TRANSACTION;  
    Else

     ROLLBACK TRANSACTION ProcedureSave;  
  
   Return @Error

posted @ 2011-12-23 17:51 老金 阅读(1507) 评论(0) [编辑](http://www.cnblogs.com/kingcat/admin/EditPosts.aspx?postid=2299823)

[**隐式事务（转）**](http://www.cnblogs.com/kingcat/archive/2011/12/23/2299817.html)

隐式事务模式，在这种模式中，SQL Server在没有事务存在的情况下会开始一个事务，但不会像在自动模式中那样自动执行COMMIT 或ROLLBACK 语句。隐式事务必须显式结束(既Commit 或者Rollback)。以下语句在没有事务时隐式开始一个事务。

|  |  |  |  |
| --- | --- | --- | --- |
| ALTER TABLE | GRANT | FETCH | DELETE |
| CREATE | REVOKE | INSERT | SELECT |
| DROP | OPEN | UPDATE | TRUNCATE TABLE |

       使用隐式事务

1.    启动SQL Server Management Studio并打开一个“新建查询”窗口。

2.    键入并执行以下语句来设置连接为隐式事务模式(此例的代码包含在示例文件UsingImplicitTransactions.sql中)：

 SET IMPLICIT\_TRANSACTIONS ON;

GO

3.    执行以下代码创建一个表检验是否已启动事务：

CREATE TABLE T1

     (i int PRIMARY KEY);

4.    用@@TRANCOUNT来测试是否已经打开一个事务。执行如下所示的SELECT语句：

SELECT @@TRANCOUNT AS [Transaction Count];

5.    结果是1，意思是当前连接已经打开了一个事务。0的意思是当前没有事务，一个大于1的数的意思是有嵌套事务(详见后文)。

6.    现在执行以下语句在表中插入一行并再次检查@@TRANCOUNT：

INSERT INTO T1 VALUES(5);

GO

SELECT @@TRANCOUNT AS [Transaction Count];

       @@TRANCOUNT的值仍然是1。由于已经有一个打开的事务，因此SQL Server没有开始一个新的事务。

7.    这个时候必须提交或者回滚事务，不然等到会话结束，表仍然是锁住，因为事务没有提交。现在执行以下语句回滚这个事务并再次检查@@TRANCOUNT。可以看出，在ROLLBACK TRAN 语句执行之后，@@TRANCOUNT 的值变成了0。

ROLLBACK TRAN

GO

SELECT @@TRANCOUNT AS [Transaction Count];

8.    尝试对表T1执行SELECT语句：

SELECT \* FROM T1;

9.    由于表不复存在，所以会得到一个错误信息。这个隐式事务起始于CREATE TABLE语句，并且ROLLBACK TRAN语句取消了第一个语句后所做的所有工作。

10.   执行以下代码关闭隐式事务：

SET IMPLICIT\_TRANSACTIONS OFF;