# Exception handling and nested transactions

June 11th, 2009

I wanted to use a template for writing procedures that behave as intuitively as possible in regard to nested transactions. My goals were:

- The procedure template should wrap all the work done in the procedure in a transaction.
- The procedures should be able to call each other and the calee should nest its transaction inside the outer caller transaction.
- The procedure should only rollback its own work in case of exception, if possible.
- The caller should be able to resume and continue even if the calee rolled back its work.

My solution is to use a either a transactions or a savepoint, depending on the value of @@TRANCOUNT at procedure start. The procedures start a new transaction if no transaction is pending. Otherwise they simply create a savepoint. On exit the procedure commits the transaction they started (if they started one), otherwise they simply exit. On exception, if the transaction is not doomed, the procedure either rolls back (if it started the transaction), or rolls back to the savepoint it created (if calee already provided a transaction).

Because of the use of savepoints this template does not work in all situations, since there are cases like distributed transactions, that cannot be mixed with savepoints. But for the average run of the mill procedures, this template has saved me very well.

```
create procedure [usp_my_procedure_name]
as
begin
        set nocount on;
        declare @trancount int;
        set @trancount = @@trancount;
        begin try
                if @trancount = 0
                        begin transaction
                else
                        save transaction usp_my_procedure_name;

                -- Do the actual work here

lbexit:
                if @trancount = 0
                        commit;
        end try
        begin catch
                declare @error int, @message varchar(4000), @xstate int;
                select @error = ERROR_NUMBER(), @message = ERROR_MESSAGE(),
@xstate = XACT_STATE();
                if @xstate = -1
                        rollback;
                if @xstate = 1 and @trancount = 0
```

```
                rollback
            if @xstate = 1 and @trancount > 0
                rollback transaction usp_my_procedure_name;

            raiserror ('usp_my_procedure_name: %d: %s', 16, 1, @error,
@message) ;
        end catch
end
go
```