

Re: Help with Stored Procedure

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-09/3109.html>

From: Louis Davidson (*dr_dontspamme_sql_at_hotmail.com*)

Date: 09/15/04

Date: Tue, 14 Sep 2004 22:42:48 -0400

Geez, relax. For starters I said probably shouldn't. And when I said maintenance routines, I did mean stuff like system stored procedures (even though most of the parameters you find will not be used in dynamic queries but rather to query system tables) which are not used over and over and over.

> *Talking of code generator. Isn't that just another form of dynamic generation/execution.*

No, it absolutely is not. All we care about is what happens when Joe User clicks the "List Orders" button on his browser. If we have to dynamically build the query, compile it, and optimize it, then, then this is less desirable than if the query was already compiled, optimized and ready to be executed at the moment requested. This needs to be obvious to anyone, since the precompiled object has already done the first few steps. It is not as easy, for sure. Sometimes it is not even realistic, but it is better when it can be done (and that is most of the time)

> *You still have to mine all schema to create the #proc.*
> *If you do not have access to the underlying table, the #proc that references these tables would not execute anyway.*

No one hopefully has said that good security is not possible using dynamic SQL, or temp procs, or whatever. Yes, it is possible. However, the problem is that with stored procedures security is far easier to implement at the database level (if you are so inclined! If you are implementing security in the middle tier then security is not a concern possibly, though it really should be)

Using the age old example of a person's salary. Say you want to give a user access to the aggregate salary costs for a company, but not to individuals. Very easy using procedures, very not easy using dynamic sql. You can write a procedure like:

```
create procedure GetAggregateSalary
as
```

microsoft.public.sqlserver.programming: Re: Help with Stored Procedure

```
select sum(salary) as totalSalary from employeeSalary
go
```

and bam, there you have it. Grant rights to this procedure, and not to the table and:

exec getAggregateSalary will return a value, select sum(salary) as totalSalary from employeeSalary will return an error stating you don't have rights. This gives you protection from peering eyes, but it also gives you protection from Cindy Lou Who (who has never been older than two) who want to run a query during the busiest part of the day. If she has access to the base table, whammo, she can write whatever query and bring down the server with blocks. If she executes a procedure for her report, you have boxed her in to a certain bit of code, which might even have a clause like:

```
--pseudocode, of course
if timeOfDay between '8:00' and '16:00'
begin
    raiserror 'No running the proc between 8 and 4 pm'
return
end
```

> *Everything has its place, dynamic or not.*

Yes, it does have its place. The question to ask is where is its place. Using dynamic SQL calls where multiple procedures will have marked improvements is plain lazyness. Is it so wrong? Not really. It is no worse than putting 5w-20 oil in your automobile that calls for 5w-30 to save a buck. Will it ever cause you real misery? Maybe not. But do it the best possible way the first time and your job will be a snap after that. If you cannot build a proper stored procedure that uses only compiled statements, punt and use dynamic. But just because you have a hammer doesn't mean everything is a mole.

--

```
-----
Louis Davidson (drsdl@hotmail.com)
Compass Technology Management
Pro SQL Server 2000 Database Design
http://www.apress.com/book/bookDisplay.html?bID=266
Note: Please reply to the newsgroups only unless you are
interested in consulting services. All other replies will be ignored :)
"oj" <nospam_ojngo@home.com> wrote in message
news:%23gS%23hDsmEHA.2340@TK2MSFTNGP11.phx.gbl...
> Excuse me. The sample code is simply a dynamic call. Have you to tried to
> compare YOUR dynamic SQL calls with #2.
>
> If such method is so bad, MS should not be using them for their system
> stored procedures.
>
> select distinct SPECIFIC_NAME,PARAMETER_NAME
> from master.INFORMATION_SCHEMA.PARAMETERS
> here PARAMETER_NAME like '%name%'
>
> Talking of code generator. Isnt' that just another form of dynamic
```

microsoft.public.sqlserver.programming: Re: Help with Stored Procedure

```
> generation/execution. You still have to mine all schema to create the
#proc.
> If you do not have access to the underlying table, the #proc that
references
> these tables would not execute anyway.
>
> Everything has its place, dynamic or not.
>
>
> "Louis Davidson" <dr_dontspamme_sql@hotmail.com> wrote in message
> news:eGOH21rmEHA.2588@TK2MSFTNGP12.phx.gbl...
> > Ok, then let's put it this way. You probably shouldn't (unless these
are
> > maintenance routines.) If you are going to write your procedures like
> this
> > (or even rebuilding them using .NET) you would be better off using
dynamic
> > SQL calls. You lose too many of the benefits of stored procedures
> > (security, performance, etc.)
> >
> > A far better plan would be to build a code generator and build one
> procedure
> > for each table you want to query. Then you get what you want, plus the
> > performance benefits. Bernie's almost there if he builds a procedure
> > temporarily, just make them permanent procedures and boom, you got it.
> >
> > --
>
> -----
> --
> > Louis Davidson (drsql@hotmail.com)
> > Compass Technology Management
> >
> > Pro SQL Server 2000 Database Design
> > http://www.apress.com/book/bookDisplay.html?bID=266
> >
> > Note: Please reply to the newsgroups only unless you are
> > interested in consulting services. All other replies will be ignored :)
> >
> > "oj" <nospam_ojngo@home.com> wrote in message
> > news:uVHCNGrmEHA.3988@tk2msftngp13.phx.gbl...
> > > Actually, you can. Here is a quick demo sp with 2 dynamic methods.
> > >
> > > e.g.
> > > create proc usp
> > > @tb sysname,
> > > @sd datetime,
> > > @ed datetime
> > > as
> > > declare @sql nvarchar(1000)
> > >
> > > --method #1
> > > set @sql='select top 1 * from '+quotename(@tb)+
> > > ' where OrderDate between ' + quotename(@sd,char(39)) +
> > > ' and ' +quotename(@ed,char(39))
> > >
> > > exec(@sql)
> > >
> > > --method #2
> > > set @sql='select top 2 * from '+quotename(@tb)+
> > > ' where OrderDate between @sd and @ed'
> > >
```

