

Re: LINQ Queries vs Stored Procs

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2008-01/msg00893.html>

- *From:* "Frans Bouma [C# MVP]" <perseus.usenetNOSPAM@xxxxxxxxxx>
 - *Date:* Wed, 09 Jan 2008 01:45:14 -0800
-

Paul Shapiro wrote:

I think the security issue is the big one. If ALL data access is via Stored Procedures, you can limit the user to only executing stored procedures, without allowing read permissions on any tables. If you have well-defined application roles, you can group users into role-based security groups and assign the minimally-necessary permissions on the stored procedures. Stored procedures eliminate most sql injection attacks, which are more possible with dynamic sql.

Injection attacks with dyn. sql are only possible if you don't use parameters. If you DO use parameters (and you should), your claim is void.

Also about the security aspects: it's not as you claim it to be: I can also specify security on tables via roles for dyn. sql. Also, if I need to return only a few columns based on user, I can also create a view for that.

The thing with 'security' and procs is that it isn't going to help: if I as a user in Marketing have to use your app which uses procs and I have to be able to delete a customer from the db, your proc will be: pr_DeleteCustomer @customerID

I can call that proc with whatever ID I want from query analyzer/whatever querying tool available.

Also, if you place business logic outside the DB, your application (which runs under the user I can use to call all your procs) is going to call more procs as it has to obtain more data (as processing takes place outside the db). I.o.w.: procs won't help here.

But this topic has been beaten to death a gazillion times :)

FB

Re: LINQ Queries vs Stored Procs

"Jon Skeet [C# MVP]" <skeet@xxxxxxxx> wrote in message
news:MPG.21edd90b3baba01179a@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx >Frank Calahan
<ichbineinhund@xxxxxxxx> wrote:

<snip>

Other than that, why should anyone would stop using SPs and start using LINQ to SQL queries? Can anyone provide a reference that refutes the guidance above from the Microsoft Patterns and Practices Group?

The discussion I've seen isn't nearly as unbalanced as you've made out. I won't go into all the details, but searching for "Frans Bouma" "stored procedures" will go a long way. Alternatively, look at the documentation for ORM projects (including the one Frans develops). They're obviously bias in the other direction.

I'm very surprised to see the patterns and practices group put out what looks like FUD. Let's see:

- o Execution plan optimization and caching: this is done for prepared statements too
- o Passing less information across the network: yeah, like the size of your SQL statement (excluding parameters which would have to be passed in either way) is likely to be in any way significant traffic
- o Putting responsibility into the hands of SQL experts: true, although there's no reason not to get your DBAs to review your LINQ queries too. In my experience, putting in this extra layer mostly results in a lack of flexibility: if I want to do anything even slightly different to what's already been done, you have to go to a lot more trouble with the "always use SPs" mantra.
- o Maintenance and security benefits: in certain situations I agree about security. Personally I'd rather maintain code than stored procs, although it's often easier to deploy a fixed stored proc once than update all clients, of course.
- o Countermeasure for SQL injection: not an issue for LINQ to SQL (or any decent ORM or use of parameterized SQL) in the first place

Now, which issue are you actually concerned about? If it's performance, I suggest you try it against your data, as that's what's going to be really important.

Re: LINQ Queries vs Stored Procs

In some cases stored procs certainly are the best way to go – but all of the processing can be expressed in a single query statement, I suspect you'll usually find that a tuned stored proc and a tuned LINQ to SQL query will be very similar in performance. The tuning of both is important, however. Also make sure when you do performance tests that you take account of the slight hit incurred the first time a particular query is executed in LINQ to SQL – don't just do one query and time that; time lots of them.

-- Jon Skeet – <skeet@xxxxxxxx>

<http://www.pobox.com/~skeet> Blog: <http://www.msmvps.com/jon.skeet>

World class .NET training in the UK: <http://iterativetraining.co.uk>

--

Lead developer of LLBLGen Pro, the productive O/R mapper for .NET

LLBLGen Pro website: <http://www.llblgen.com>

My .NET blog: <http://weblogs.asp.net/fbouma>

Microsoft MVP (C#)

.