

Re: Sql / Dot Net General Discussion

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.general/2006-10/msg00615.html>

- *From:* "Tom Dacon" <tdacon@xxxxxxxxxxxxxxxx>
 - *Date:* Thu, 19 Oct 2006 19:24:47 -0700
-

You know, this might be one of those "if what you have is a hammer, everything looks like a nail" situations.

If what you have is a bunch of DBA's doing the software architecture, they're going to be oriented towards doing the business logic with constraints, triggers, and stored procedures, because that's what they know. If what you have is a bunch of application programmers (C++, .Net languages, etc.) they're going to want to do the business logic in the middle tier, and use the data layer only for access and update and so forth, because that's what they know.

I could go on at considerable length, and probably will if this turns into an active thread, but for now...

Regards,
Tom Dacon
Dacon Software Consulting

"Doug" <dnlwhite@xxxxxxxx> wrote in message
<news:1161309863.849475.276630@xx>

Hi,

I wanted to start a general discussion more for getting some thoughts on what other people think/practice out there just to see how far (if at all) I'm off base on my own thoughts.

My primary experience is developing applications using VB or DotNet. I have some sql skills but they are limited. In a previous company our concept on SQL was that it was used for very simple work, (i.e. insert, update, delete, select, etc). The applications we wrote did the bulk of the work. We had very limited DTS's wrtten and stored procs were very small.

In the company I've been working at for the last year, we have two different mindsets on this issue (to be honest, the numbers of people who feel like I do have dwindled, we've lost some of our DotNet

Re: Sql / Dot Net General Discussion

developers in the last few months). A very large amount of work is being developed in SQL and even where our DotNet applications are concerned, I have seen some push for putting a lot of the work into the stored procs instead of having them be simplistic like I mentioned in the previous paragraph. I have seen stored procs called by dot net apps that call other stored procs, that call others, etc. Some of these procs are like miniature apps in of themselves.

I have a hard time wrapping my brain around why anyone would do this. I believe that this type of design is problematic for maintainence at the very least. But I would think it puts an unnecessary burden on SQL too. I just don't know how to prove it. When I've mentioned this, some of the feedback I get is that my concept would cause network traffic that is unnecessary (i.e. multiple stored proc calls, etc). Again, I am unsure how to test/verify such a claim.

I would think the best approach is to have your business logic stay in DotNet if you have a DotNet app. Obviously if you have a process that doesn't get put into an application at all, then I believe the logic should stay in SQL. I do question the necessity of having that type of work happen as often as I'm seeing it though. DotNet can be used to write pretty much any type of application that you would do in SQL.

I'm curious as to how other groups approach this issue? Any feedback at all – regardless of how it would be in regards to my own opinion on this subject would be much appreciated.