Re: Update statement performance decreases in stored proc

Source:
http://www.tech−archive.net/Archive/SQL−Server/microsoft.public.sqlserver.programming/2005−01/2792.html

---

From: Hugo Kornelis (*hugo_at_pe_NO_rFact.in_SPAM_fo*)
Date: 01/14/05

Date: Fri, 14 Jan 2005 16:28:15 +0100

On Wed, 12 Jan 2005 12:48:57 −0000, Jacco Schalkwijk wrote:

>Hugo,
>
>Have you had a look at the execution plan of the stored procedure? It might
>be that it is not using the index you create, because it is created after
>all the inserts happen on the table. Creating an index on a table will not
>automatically mark stored procedures that use that table for recompilation.
>You can try if moving the create index statement to a place before the
>inserts will make a difference in procedure execution time.

Hi Jacco,

I've spent over a day testing various combinations of indexes, created at different moments in the stored procedure. My findings are:

− If a nonclustered index is created after the data is entered in the table, it is not used in any queries. Apparently, no recompile is triggered. This is only true in stored procedures; if executed as a single batch from QA, the index will get used.

− If a clustered index is created after the data is entered in the table, it will be used in queries. Apparently, the building of a clustered index does trigger a recompile of the stored procedure.

− If a nonclustered index is created after the data is entered in the table, but the queries that might benefit from this index are moved to a new stored procedure that's called from this procedure, they will use the new index.

To my surprise, I also found that the supporting index didn't help much at all. Of all variations I tested with all code in one procedure, the quickest was the one using only the PK (clustered) and no supporting index. The only variant that was slightly faster than when I started with no PK and only the supporting index (clustered), then just before the

update to the main table (GenRegel), create a unique clustered index on
the column sorteercode WITH DROP_EXISTING. This version was (on average
after 3 executions) 0.37 seconds faster than the version with only the PK,
on a total execution time of almost 30 seconds for this procedure (which
is part of a larger process, that takes about 4 1/2 minutes). I decided
that this gain isn't worth the effort, so I'll just remove the supporting
index from my code and stick to using the primary key only.

Thanks for the help!

Best, Hugo

```
--
(Remove _NO_ and _SPAM_ to get my e-mail address)
```