Re: FTS query performance on SQL 2005

Source:
http://www.tech−archive.net/Archive/SQL−Server/microsoft.public.sqlserver.fulltext/2006−11/msg00017.html

- *From*: Simon Sabin <SimonSabin@xxxxxxxxxxxxxxx>
- *Date*: Fri, 3 Nov 2006 23:35:19 +0000 (UTC)

Hello Hilary,

I've found that this greatly improved in sql 2005.
Simon Sabin
SQL Server MVP
http://sqlblogcasts.com/blogs/simons

> Note that it is not the SQL Server cache where the queries are cached,
> but rather the full−text catalog pages which are cached in the file
> system cache (which oddly enough turns out to be the most efficient
> caching mechanism not only for SQL FTS, and other Microsoft Search
> products, but also for Lucene).
>
> Here is an example:
>
> Issue this query
>
> Select * from myfulltexttable where contains(*,'test')
>
> and observe the length taken. Repeat the query and not the decrease in
> time due to caching then clear the procedure cache and data cache
>
> declare @int int
> select @int=db_id()
> DBCC FLUSHPROCINDB (@int)
> DBCC DROPCLEANBUFFERS
> try again and note the time taken is still better than the first time
> due to caching advantages in the file system:
>
> Select * from myfulltexttable where contains(*,'test')
>
> Stop and start msftesql and repeat the query, note the time. Again
> caching advantages.
>
> This posting is my own and doesn't necessarily represent
> RelevantNoise's positions, strategies or opinions.

"Simon Sabin" <SimonSabin@xxxxxxxxxxxxxx> wrote in message
news:62959f1a2a9338c8cc15dcf18530@xxxxxxxxxxxxxxxxxxxxxx

> Hello Bahama,
>
> Its caching, when the server restarts there is nothing in memory so
> everything has to be retrieved from disk. Once its been read from
> disk it stays in the cache until something pushes it out. If you are
> just doing contains table then all that needs to be cached is ids and
> they're quite small. so even if the id's are 20 bytes, 20 million
> rows is 400Mbytes.
>
> What version of SQL server are you using?
>
> Simon Sabin
> SQL Server MVP
> http://sqlblogcasts.com/blogs/simons
>
>> I have achieved a breakthrough in this performance testing,
>> whereby
>> now all of my queries return in under 1/2 second. All of my
>> testing
>> was being performed right after a server reboot in an attempt
>> to
>> eliminate any effects of cached query results. However, I
>> also found
>> that sqlservr.exe was using very little memory during this
>> time, and
>> during those long queries, the memory usage was steadily
>> increasing.
>> I also found that once sqlservr.exe settled out at around
>> 740MB of
>> memory usage, that all queries responded very quickly. So,
>> before
>> doing any testing, I rebooted the server, allowed all processes
>> to
>> fully startup, then issued a query that returned nearly all rows
>> in
>> the database, such as "select count(1) from
>> containstable(Item,
>> ItemText, 'is')", since the word "is" is in almost every row in
>> my
>> database. (BTW, I have blanked out my "noise file", since I
>> need all
>> those words to be searchable.) That query for "is" takes

several
minutes, but at the end of that query, my sqlservr.exe process has
reached about 740MB of memory usage. From that point on, the
sqlservr.exe process memory usage does not increase any more, and
all queries (including those that were taking several minutes) now
return in under 1/2 second.

Can anyone out there provide an under−the−covers explanation for the
behavior that I'm seeing? Is it that the full−text index (with 1.4
million unique keys) is being cached in memory, after which the
performance improves? If that's the case, why wouldn't that memory
be used by the msftesql.exe process instead of the sqlservr.exe
process?

Thanks for any insights.

.