

## Re: sliced find in Linq

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2007-11/msg00059.html>

---

- *From:* "Frans Bouma [C# MVP]" <[perseus.usenetNOSPAM@xxxxxxxxxx](mailto:perseus.usenetNOSPAM@xxxxxxxxxx)>
  - *Date:* Thu, 01 Nov 2007 02:44:27 -0700
- 

Carl Daniel [VC++ MVP] wrote:

Frans Bouma [C# MVP] wrote:

Carl Daniel [VC++ MVP] wrote:

The only undesirable thing I see in the SQL above is that order by over all of the returned columns. If Customers is a large table, this will result in SQL server doing the sort in tempdb. Since the original query was unordered, I'd like the above SQL to simply order by the primary key, since that alone must define a unique ordering, which is all that's required for this to work. (Perhaps your table doesn't have a primary key though?)

The ROWNUMBER OVER statement requires an ordering, so normally you'd place either the ordering of the inner sql there or a new one (the PK field(s)).

Yes, of course it does. But in this case, the query as stated in C# didn't specify the ordering, so any ordering could have been used. It appears that LINQ chose to use the most expensive ordering possible, when it should use the least expensive when no ordering was specified in the original expression. Hence my question to Jon – did this table have a primary key? If it didn't (and no unique keys either), then this is the best LINQ could do, but if any unique keys exist, LINQ should use one of those – perhaps it does, but I can't tell from this example.

As I explained in another post in a different branch in this thread ;), paging or using TOP over a query without an ordering is

Re: sliced find in Linq

meaningless, as SELECT has by definition no specified order, it's undefined: it's not defined in which order the rows will be returned to you. So using TOP or a paging construct over such a set which isn't ordered by a specified ordering will return a rowset that's by definition undefined as you don't know the order of the set you page over / limit the resultset on. :)

A lot of developers have the perception that SELECT has always the same order, but that's not true if there's no ORDER BY clause: the RDBMS can re-order rows because of mempage constraints or other internal housekeeping.

FB

--

---

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#)

---