

Re: SQL Data Provider Performance Issues

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2007-08/msg00909.html>

- *From:* Greg <gmichaels@xxxxxxxxxxxxxxx>
 - *Date:* Tue, 07 Aug 2007 08:05:27 -0700
-

On Aug 7, 7:39 am, "Nicholas Paldino [.NET/C# MVP]"
<m...@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote:

Based on your other posts, it seems like it's not so much the individual fills, but rather, the fact that you have so many of them. Are the numbers you gave for one individual call to Fill the data set, or is it for all of them?

I believe you are on the right track when you say that it is the interaction of the app with the database, but you haven't given many details about that. I don't think it is the SqlDataAdapter in itself. Have you tried just executing the command, getting a reader, and then cycling through it (columns and rows) to see what the performance there is? Or, maybe just filling one dataset, and measuring the performance there?

—
– Nicholas Paldino [.NET/C# MVP]
– m...@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

"Greg" <gmicha...@xxxxxxxxxxxxxxx> wrote in message

<news:1186468407.360935.9580@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>

On Aug 6, 7:36 pm, "Nicholas Paldino [.NET/C# MVP]"
<m...@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote:

Greg,

It seems like something else is definitely going on here. I can't see the code you are using, so I can't tell what is going on on the .NET

Re: SQL Data Provider Performance Issues

side.

However, from what you have shown of the execution tree, it seems like

you

haven't optimized the ValidValues_HLAHighRes table for this query. It's

doing an index scan, which is not really what you want. You should

probably

index it so that an index seek will be performed. If you have a large

number of records in the ValidValues_HLAHighRes table, then that could be

impacting performance.

Of course, there is some overhead in populating a data set, but from

what you are mentioning, it doesn't seem like it should be that much.

Assuming a reasonable number of columns in the table, a few hundred rows

really shouldn't take that long.

--

- Nicholas Paldino [.NET/C# MVP]

- m...@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

"Greg" <gmicha...@xxxxxxxxxxxxxx> wrote in message

news:1186446086.255319.90070@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

I am trying to fill strongly typed datasets with data from a SQLServer DB. The data is used as a datasource for drop down lists. The data adapters are configured to perform only selects, no inserts, updates, or deletes. When I call the Fill method to load the dataset, CPU Usage goes to 100%. These simple queries take forever to complete. We are

Re: SQL Data Provider Performance Issues

only dealing with a few hundred rows. My windows form takes minutes to load because of the poor performance. The select statements require 2 inner joins to acquire the required data columns. Similar fills using selects with a single inner join perform OK. I tried running the SQL Profiler to analyze the problem. When I run the query from SQL Query Analyzer, SQL Profiler shows a duration of 30. When run from within the C# .net application duration jumps to 9000 or 300 times slower. Reads stay the same. SQL execution plan is the same when using Query analyzer versus the application. It appears the SQL Data Provider is the source of the poor performance. Here is the SQL Select that performs poorly:

```
SELECT H.HLAHighResID,
H.HighResName, H.HLALowResID,
L.LowResName
FROM ValidValues_HLAHighRes H
INNER JOIN ValidValues_HLACategories
C ON H.HLACategoryID =
C.HLACategoryID
INNER JOIN ValidValues_HLALowRes L
ON H.HLALowResID = L.HLALowResID
WHERE (C.CategoryName = 'A')
```

Execution Tree

```
-----
Nested Loops(Inner Join, OUTER
REFERENCES:([H].[HLALowResID]))
|--Nested Loops(Inner Join,
WHERE:([H].[HLACategoryID]=[C].
[HLACategoryID]))
| |--Index
Seek(OBJECT:([test].[dbo].[ValidValues_HLACategories].
[IX_ValidValues_HLACategories_1] AS
[C]), SEEK:([C].
[CategoryName]='A') ORDERED
FORWARD)
| |--Clustered Index
```

Re: SQL Data Provider Performance Issues

```
Scan(OBJECT:([test].[dbo].
[ValidValues_HLAHighRes].[PK_ValidValues_HLAHighRes]
AS [H]))
|--Clustered Index
Seek(OBJECT:([test].[dbo].[ValidValues_HLALowRes].
[PK_ValidValues_HLALowRes] AS [L]),
SEEK:([L].[HLALowResID]=[H].
[HLALowResID]) ORDERED FORWARD)
```

Here is the SQL Select that performs OK
even though duration doubles:

```
SELECT L.HLALowResID,
L.LowResName
FROM ValidValues_HLALowRes L
INNER JOIN ValidValues_HLACategories
C ON L.HLACategoryID =
C.HLACategoryID
WHERE (C.CategoryName = 'A') ORDER
BY L.LowResName
```

Execution Tree

```
-----
Sort(ORDER BY:([L].[LowResName]
ASC))
|--Nested Loops(Inner Join,
WHERE:([C].[HLACategoryID]=[L].
[HLACategoryID]))
|--Index
Seek(OBJECT:([test].[dbo].[ValidValues_HLACategories].
[IX_ValidValues_HLACategories_1] AS
[C]), SEEK:([C].
[CategoryName]='A') ORDERED
FORWARD)
|--Clustered Index
Scan(OBJECT:([test].[dbo].
[ValidValues_HLALowRes].[PK_ValidValues_HLALowRes]
AS [L]))
```

These are quite similar except for an extra
INNER JOIN. What can the
source of this poor performance be? Why does
CPU get pegged at 100% when

Re: SQL Data Provider Performance Issues

I run these selects from the application? I thought the SQL Data Provider was the preferred choice when accessing a SQL Server DB. Should I try an OLE Data Adapter?– Hide quoted text –

– Show quoted text –

The HLAHighRes table has 3600 rows, HLLowRes has 140 rows, And HLACategories has just 6 rows.

The queries run very fast from query analyzer but get terribly slow when called by the data adapter fill method.

CPU goes to 100%. In my DB experience it performs like it is doing full table scans with thousands of rows.

All the tables are keyed by an identity column hence the ID in the column name and have a primary clustered key index.

I tried adding an index on the combo of H.HLAHighResID, H.HighResName, H.HLLowResID

which was recommended by the profiler index analyzer. It really did not effect performance.

Seems to me there is some issue with the interaction of the app and DB.– Hide quoted text –

– Show quoted text –

40 of the datasets get filled when the form loads. The 1st 10 that use 1 INNER JOIN load OK.

As soon as it hits the next 10 that have 2 INNER Joins performance goes bad. Then the next 10 datasets have

the 1 INNER JOIN again and performance picks up. The last 10 datasets filled are back to 2 INNER JOINS and performance goes bad again.

I have a Profiler Trace showing this. I tried changing the order by putting the poor performers 1st. It did not matter.

The performance was still going from poor to good depending on the query and not the order.

.