

Re: ASP – FROM statement slows down connection to database

Source:

<http://www.tech-archive.net/Archive/Internet-Server/microsoft.public.inetserver.asp.db/2006-06/msg00001.html>

- *From:* "Bob Barrows [MVP]" <reb01501@xxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 5 Jun 2006 07:33:30 -0400
-

Wayne & Carr wrote:

Hello All;

```
=====
sLastParentCat = ""
set Bconn = server.createobject("adodb.connection")
Bconn.open "provider=microsoft.jet.oledb.4.0;data source=" &
server.MapPath("FastData\CFF_DB.mdb") & ""
i = 0
set Srs = server.CreateObject("adodb.recordset")
sSQL = "SELECT distinct A.*, B.CatID, B.Cat, " & _
" (SELECT COUNT(1) FROM Cats WHERE A.CatTypeID=Cats.CatTypeID)
AS ParentCount, " & _
" (SELECT COUNT(1) FROM Sierra WHERE B.CatID=Sierra.CatID) AS
ChildCount " & _
" FROM (CatTypes A LEFT JOIN Cats B ON A.CatTypeID =
B.CatTypeID) " & _
" LEFT JOIN Sierra C on B.CatID = C.CatID "
```

For purposes of debugging add these two lines:

```
Response.Write sSQL
Response.End
```

Run the page, copying the statement written to the browser window to the clipboard.

Open your database in Access, switch to the Queries tab, create a new query in Design View without choosing a table, switch to SQL View, paste the sql statement and run it. How long does it take to run?

I see several issues here:

1. Is the "distinct" keyword actually needed? Assuming that B.CatID is the primary key, this query is going to return distinct rows. Why use "distinct"? Run the query without it to see the difference in the results and in the time it takes to run

Re: ASP – FROM statement slows down connection to database

2. Do you really need all the fields from CatTypes? Avoid using selstar (Select *).
3. The subqueries can kill performance since they need to be run for every record returned by the query. For debugging purposes, remove them to see the difference. There should be a different way to get what you want.
4. I see no reason to include that last left join to the Sierra table. You don't seem to be using the fields from that table in your main query. It shouldn't have a big impact on performance, but remove that join anyway.
5. A real biggie: no WHERE clause. Why are you retrieving all the records from these tables? Are you planning to use all of them? If not, you are really wasting resources and bandwidth by retrieving all of them.

Bottom line: this query needs to be rewritten, but I don't know how to rewrite it without knowing the goal you are trying to achieve. I suspect something like this will be better:

Create and save a new query called "qParentCounts" with this sql:

```
SELECT CatTypeID, COUNT(*) As ParentCount
FROM Cats
GROUP BY CatTypeID
```

Create and save another new query called "qChildCounts" with this sql:

```
SELECT CatID, COUNT(*) As ChildCount
FROM Sierra
GROUP BY CatID
```

Now test this sql statement to see if it gives you what you want as well as performing better:

```
SELECT A.*, B.CatID, B.Cat, ParentCount,ChildCount
FROM ((CatTypes A LEFT JOIN Cats B ON A.CatTypeID = B.CatTypeID)
JOIN qParentCounts P ON A.CatTypeID=P.CatTypeID)
JOIN qChildCounts C ON B.CatID=C.CatID
```

This should make a difference.

Now, unless you are really planning to use all the records, add a WHERE clause to limit the records returned to the ones you are actually planning to use in your web page.

Now, let's look at this Open statement, which also has some issues:

```
Srs.Open sSQL, Bconn, 1, 3, 1
```

Let's start with the second argument, Bconn. Opening a recordset using a connection string (resulting in an implicit connection) rather than using an

Re: ASP – FROM statement slows down connection to database

explicit connection object is a bad idea. Probably nothing to do with this specific problem, but a bad idea nonetheless. Always create and open an explicit connection object instead of using connection strings in your recordset open statements. Use the connection object in all your dealings with the database. Failure to do so can disable connection pooling, resulting in excessive connections being opened to your database, which can kill your application's scalability and performance.

The third argument, 1, is attempting to open a keyset cursor. Why do you want a keyset? In an ASP application, you should not be intending to keep this recordset open long enough to care about changes made by other users. A keyset is overkill.

The 4th argument, 3, specifies an optimistic lock. Are you planning to edit the data in this cursor? Again, that's a bad idea in ASP. Data maintenance should be done via efficient SQL DML statements (INSERT, UPDATE and DELETE). Furthermore, given the left joins and subqueries, I rather doubt that this cursor will be updatable in any case. I strongly suggest that you open the default forwardonly, readonly cursor.

The last argument, 1, specifies adCmdText, which is actually the correct specification to be using in this case!

So, with these changes, your code should look like this:

```
Dim cn
Set cn=CreateObject("adodb.connection")
cn.Open Bconn
Set Srs=cn.Execute(sSQL,,1)
```

Here is some more reading info:

Using saved parameter queries:

<http://groups-beta.google.com/group/microsoft.public.inetserver.asp.db/msg/b3d322b882a604bd>

Using Command object to parameterize CommandText:

<http://groups-beta.google.com/group/microsoft.public.inetserver.asp.db/msg/72e36562fee7804e>

--

Microsoft MVP – ASP/ASP.NET

Please reply to the newsgroup. This email account is my spam trap so I don't check it very often. If you must reply off-line, then remove the "NO SPAM"

.