Re: Under the hood of DataReader

Source: http://www.tech−archive.net/Archive/DotNet/microsoft.public.dotnet.general/2004−02/1455.html

---

From: Pablo Castro [MS] (*pablocas_at_online.microsoft.com*)
Date: 02/16/04

Date: Mon, 16 Feb 2004 11:22:28 −0800

Let me clarify this a little: first of all, implementation details for DataReader is provider−specific; we only provide guidelines of what would be the expected behavior.

In order to describe the details of buffering, I'll pick the SQL Server provider (SqlClient):

When you execute a SQL statement that returns rows, the server starts sending rows to the client over the network in packets[1]. The client on the other end consumes one packet at a time from the network. Every time you call Read() on the reader, we see if we still have a full row of data in the current 1−packet buffer; if we don't, we issue another network read and get another packet and obtain the rest of the data for the row[2]. If you don't call Read() for a while, we'll eventually have a packet on the client buffer and a packet or two ready to be sent in the server buffer; at that point, the server will stop producing data until we consume data from the client. This means that there is no need to buffer the rest of the data because it hasn't been sent by server yet.

Once you call Read() again, we discard the previous row and fetch the next one from the packet buffer or the network, as described above.

Bottom line is that SqlDataReader will never buffer more than 1 packet worth of data. You can even fine tune this further by using CommandBehavior.SequentialAccess, which will cause the provider to fetch one field at a time, instead of one row at a time.

[1] Default 8KB in size in 1.0/1.1, may (read will) change in future versions.
[2] The the row is wide enough (i.e. many columns or very large values) a single row may span many packets. In those cases, we may issue several network reads for a single Read() call on the reader.
[note] "network" here may actually be shared memory if client and server are in the same box, in that case there is no network, but the idea still applies.

I hope this clarifies the internals a little. Let me know if you'd like to dig into any other detail.

```
--
Pablo Castro
Program Manager - ADO.NET Team
Microsoft Corp.
This posting is provided "AS IS" with no warranties, and confers no rights.
"Cowboy (Gregory A. Beamer)" <NoSpamMgbworld@comcast.netNoSpamM> wrote in
message news:e43Zt4j8DHA.2656@TK2MSFTNGP11.phx.gbl...
> First, let's make sure we understand our terms.
> 1. Server is the database server
> 2. Client is the software getting data (the one that opened the
connection).
> This can be a fat client or a web app.
>
> The method of the data reader is simliar to any fast forward (firehose)
> cursor on the server side. The information is streamed directly from the
> database. On the client, the reader will require some form of buffer to
hold
> the data. It is a very short term buffer, as it is cleared when the
> connection closes (NOTE: Until GC, the memory is not necessarily cleared,
> but the data is no longer accessible).
>
> When the DataReader is read, each "row" is destroyed from the stream
> (buffer) as it is consumed. When closed, or the connection is closed, all
> "rows" are destroyed. There is no built in persistence of the data, ala a
> DataSet, so the data effectively dies if you do not store it in your app
> somehow.
>
> I know of a programmer who only uses DataReaders. He likes the perf.
Problem
> is his apps are much more complex and harder to maintain due to his
refusal
> to use a DataSet. I tend to err on the side of maintainability until I
find
> a perf problem. I would also caution anyone who decides that the ONLY way
to
> work with data is a DataReader, for performance reasons, as your app will
> likely be maintained by someone who is not as educated (and perhaps not as
> smart) as you.
>
> --
> Gregory A. Beamer
> MVP; MCP: +I, SE, SD, DBA
>
> **********************************************************************
> Think Outside the Box!
> **********************************************************************
> "Yasutaka Ito" <nobody@nonexistent.com> wrote in message
> news:e66x%23Zj8DHA.2480@TK2MSFTNGP12.phx.gbl...
> > Hi,
> >
> > My friend had a little confusion about the working of DataReader after
> > reading an article from MSDN. Following is a message from him...
> >
> > <!-- Message starts -->
> > I was going thru DataReader in ADO.NET in MSDN and there is a confusion
> > regarding the buffering
> > of data in case of DATAREADER. The link for MSDN JAN 2004 is -
> >
> >
>
ms-help://MS.MSDNQTR.2004JAN.1033/cpguide/html/cpconTheADONETDataReader.htm
> > At this link, they say that :
```

> > Results are returned as the query executes, and are stored in the
network
> > buffer on the client until you request them using the Read method of the
> > DataReader
> > That's in the starting part of the article. Later in the same article,
it
> is
> > said that DataReader provides unbuffered stream....
> >
> > The DataReader provides an unbuffered stream of data.................
> > DataReader is a good choice when retrieving large amounts of data
because
> > the data is not cached in memory.
> > So, there is no buffering in the primary memory going on but there is
some
> > network buffer. If there is, where is the network buffer − in the NIC or
> > some Router or other Gateway... ??
> > And whether at all, data in case of DataReader is buffered or not.. ??
> > <!−− Message ends −−>
> >
> > The way I understood it is as follows, but wanted to confirm it with you
> > foloks before I rant foolishly. Would appreciate further inputs or
> > corrections, which I most probably need.
> >
> > − data is streamed to the client machine and gets queued up until it is
> > processed or the data reader is closed
> > − application (that's making the request) is given the starting pointer
to
> > the data coming in
> > − data consumed by the application is destroyed as it is used (by
calling
> > the Read method), similar to electricity...
> > − term 'buffer' in the unbuffered means not getting buffered in the
> > application's memory space, as with DataSet
> >
> > thanks!
> > −Yasutaka
> >
> >
>
>