Re: ADO.Net Connection Pooling Problem with Oracle

Source:
http://www.tech−archive.net/Archive/DotNet/microsoft.public.dotnet.framework.adonet/2005−04/msg00300.html

- *From*: "Bird" <mattsiemer@xxxxxxxxxxxxxxxx>
- *Date*: 5 Apr 2005 10:40:47 −0700

Matthew,

I do have a development environment, but why would I want to use OLEDB which I thought added another layer to the data access interface? I'm trying to solve the problem with the OracleClient.

When I turn off connection pooling, then the database gets overloaded with database connections as soon as people start accessing the application. I don't think .Net releases the connections until the garbage collector cleans them up. I've experimented with this and that's essentially what I saw. See the 2nd last paragraph of my original message.

Moving connOracle.close up to the try section of the try block does nothing different. The finally part of the try block always gets executed whether there is an exception or not unless the exception is within the code within the finally section.

This code is in production.

Can anyone give an example of a pooled database connection's life cycle? I haven't read anything that tells me what causes a connection to be put into the pool and when does a connection get taken from the pool. Is the connection immediately put into the pool upon executing the .close method of the connection object or does it wait until after the connection object is cleaned up by the garbage collector? Does .Net always get a pooled connection if one exists for the same connection string or are there exceptions? How does Oracle interact with .Net? Does .Net pool just one connection for each unique connection string or does it do something similar to what OLEDB did in ADO 2.5−2.6, where it creates n+nbr processors connections?

If .Net will not use connections that have been closed but have not been garbage collected, then what the hell good is pooling? Or should I be collecting garbage everytime I close a database connection? Isn't garbage collection an intensive process, hence the reason it runs in the background? Problem with calling System.GC.Collect() explicitly is

that I'm open multiple connections for each web page that's loaded. I have a data access component that returns datasets and closes the connections beforehand so I don't have to worry about closing connections on the calling side.

Thanks,
Bird

.

- ***Follow−Ups***:
  - ♦ ***Re: ADO.Net Connection Pooling Problem with Oracle***
    - ◊ *From:* Bird

- ***References***:
  - ♦ ***ADO.Net Connection Pooling Problem with Oracle***
    - ◊ *From:* Bird
  - ♦ ***RE: ADO.Net Connection Pooling Problem with Oracle***
    - ◊ *From:* Matthew Holton

- Prev by Date: ***Re: ADO.NET 2.0 − Question for Microsoft Employees***
- Next by Date: ***RE: memory leak in SqlDataAdapter.Fill method?***
- Previous by thread: ***RE: ADO.Net Connection Pooling Problem with Oracle***
- Next by thread: ***Re: ADO.Net Connection Pooling Problem with Oracle***
- Index(es):
  - ♦ ***Date***
  - ♦ ***Thread***