

RE: Connection pooling

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-05/2617.html>

From: vishal subramaniam (*vishalsu_at_microsoft.com*)

Date: 05/12/04

Date: Wed, 12 May 2004 12:29:25 GMT

vishalsu@online.microsoft.com

RESOLUTION:

=====

Connection Pooling with SQL Server 2000 Analysis Services
Dennis Kennedy
Microsoft Corporation

Originally published May 2001, updated November 2002

Applies to:

Microsoft® SQL Server™ 2000 Analysis Services

Summary: Learn how to use the connection pooling objects included with the Microsoft XML for Analysis Provider to develop scalable client and Web applications for Microsoft SQL Server 2000 Analysis Services. (11 printed pages)

Contents

Introduction

Audience

Connection Pooling Objects

Using the Connection Pooling Objects

Requesting and Returning Connections

Balancing and Shrinking the Connection Pool

ADOConPool Object

OleDbConPool Object

Conclusion

Additional Information

Introduction

Resource management is an important consideration in the development of scalable client and Web-based applications. In the construction of a client application that might serve many concurrent users, the guideline for resource management is to allocate resources as late as possible and de-allocate resources as early as possible. The availability of resources,

such as memory, process threads, and network or database connections, relates directly to the performance and user satisfaction of a client application. Therefore, resource management becomes more and more important as the client application is scaled up and out.

By providing more control over resource management, connection pooling can reduce the impact of scalability. Connection pooling enables a client application to use a connection to a given resource from a pool of connections that do not need to be re-established for every use. After a connection has been created and placed in a connection pool, a client application can reuse that connection without performing the complete connection process.

Using a pooled connection can result in significant performance gains because client applications do not need to repeatedly establish and close a connection. The time required by this process can be particularly significant for client applications that use high latency resources, such as Internet or network connections. After the client application no longer needs a connection, the connection is simply returned to the connection pool.

In addition to performance gains, connection pooling enables a resource to be managed more effectively, without forcing the overhead of resource manage