Visual C# 下利用 ADO.NET 访问 SQL Server 技术

The Technique of Visiting SQL Server Using ADO.NET in Visual C#

刘秋香　张永胜　（济南山东师范大学信息管理学院　250014）

摘要：本文主要介绍了在 Visual C# 中如何利用 ADO.NET 访问 Microsoft SQL Server 2000 数据库。文中先对.NET 框架和 Visual C#.NET 作了简单的介绍，然后对 ADO.NET 作了较为详尽的介绍，最后示范了四种基本的 SQL Server 数据库操作：读取数据、写入数据、更新或修改数据、删除数据。

关键词：.NET 框架　Visual C#　ADO.NET　SQL Server

1　引言

随着应用程序开发的发展演变，越来越多的应用程序需要通过网络传递数据。为了允许在分布式、可缩放应用程序中实现数据集成，.NET 框架提供了一种名为 ADO.NET 的新一代数据访问技术。通过 ADO.NET 不仅能够使应用程序连接到数据源，检索、操纵和更新数据，而且能够使应用程序实现对非关系数据库格式数据的访问。本文主要介绍在 Visual C#.NET 中如何利用 ADO.NET 进行数据访问，在数据库的选用上，采用了微软公司的 SQL Server 2000。

2　ADO.NET 组件与对象模型

ADO.NET 是一组向.NET 程序员公开数据访问服务的类。组成 ADO.NET 的各个类被包含在 System.Data、System.Data.SqlClient、System.Data.OleDb、System.Data.Odbc 以及 System.Data.OracleClient 五个命名空间中，可以将其分为两大类型：提供者类和使用者类。提供者类完成将数据从数据源的读取和写入等实际操作，而当数据被读到存储介质后，由使用者类完成数据的访问和操作等功能。提供者类中包括了 Connection、Command、DataReader 和 DataAdapter 等，上述名称都是泛称，并非真的是定义于.NET 框架类库中的实际类，每一种.NET 数据提供程序都有这些类的自我版本，例如 SQL Server.NET 框架数据提供程序的 Connetcion 是通过 SqlConnection 类实现的，而在 OLE DB.NET 框架数据提供程序中则是通过 OleDbConnection 类实现的。使用者类中包括了 DataSet、DataTable、DataColumn、DataRow 和 DataRelation 等，其中 DataSet 类是 ADO.NET 的断开式结构的核心组件，用以实现独立于任何数据源的数据访问。

(1) Connection：用来建立与特定数据源的连接。SqlConnection 管理与 SQL Server 数据库的连接，用于连接 SQL Server 7.0 或更高版本的数据库，它经过了优化，使用自身的协议与 SQL Server 通信，因此具有良好的性能。对于 Microsoft SQL Server 6.5 版和较早版本，应当将用于 SQL Server 的 OLE DB 提供程序与 OLE DB .NET 框架数据提供程序一起使用。OleDbConnection 管理与可通过 OLE DB 访问的任何数据存储区的连接。连接的两个主要方法是 Open 和 Close。Open 方法联系数据源并建立一个打开的连接，Close 方法关闭连接。关闭连接是必要的，因为大多数数据源只支持有限数目的打开的连接，并且打开的连接占用宝贵的系统资源。如果正在使用数据适配器，则不必显式打开和关闭连接。当调用数据适配器的 Fill 或 Update 方法时，该方法将检查连接是否已打开，如果没有，适配器将打开连接，执行其逻辑，然后再关闭连接。这类方法仅当连接尚未打开时才自动打开和关闭连接；如果连接是打开的，则这些方法使用连接但并不关闭它。

(2) Command：用来对数据源执行 SQL 命令语句或存储过程，如进行数据插入、删除、修改等操作。可以使用 SqlCommand 直接操作 SQL Server 7.0 或更高版本数据库；通过 OleDbCommand 操作 OLE DB 数据源。

(3) DataReader：用来从数据源中读取数据行的只进数据流。包含在 DataReader 的数据是由数据库返回的只读、只能向下滚动的流信息，因此很适合应用在只需读取一次的数据。SqlDataReader 专门用来读取 SQL Server 的数据，OleDbDataReader 用来读取 OLE DB 的数据。当 DataReader 打开时，该 DataReader 将以独占方式使用 Connection，因此，每次使用完 DataReader 后，都应调用 Close 方法关闭 DataReader，否则，在初始 DataReader 关闭之前，将无法对 Connection 执行任何命令（包括创建另一个 DataReader）。所以，使用 DataReader 包括这样几步：生成 DataReader，从 DataReader 读数据，关闭 DataReader。

(4) DataAdapter：用来在数据源和数据集（DataSet）

之间交换数据，它可以隐藏和 Connection、Command 对象沟通的细节。它可以从数据源取出数据，还可以对底层数据存储体进行数据的添加、删除、修改或更新操作。在许多应用程序中，这意味着从数据源将数据读入数据集，然后从数据集将已更改数据写回数据源。

(5) DataSet：用来处理从数据源读出的数据，表示数据在内存中的缓存。DataSet 对象是支持 ADO.NET 的断开式、分布式数据方案的核心对象。它可以用于多个不同的数据源，用于 XML 数据，或用于管理应用程序本地的数据。无论数据源是什么，它都会提供一致的关系编程模型。它与现有数据源的交互通过 DataAdapter 来控制。使用 DataSet 的过程通常包括这样几步：生成数据库连接，存储查询到 DataAdapter，生成和填充 DataSet，读出数据并显示。

DataSet 是 ADO.NET 结构的主要组件，它是一个集合对象，可以包含多个数据表，以及表的约束、索引和关系。DataSet 类包含数据表 DataTable 类的 Tables 集合和 DataRelation 类的 Relations 集合。DataTable 类包含数据行 DataRow 类的 Rows 集合、DataColumn 类的 Columns 集合以及数据关系的 ChildRelations 和 ParentRelations 集合。

3 ADO.NET 访问数据库的两种不同方式

3.1 保持连接状态下使用数据命令执行数据库交互

最常见数据任务是从数据库检索数据并对数据进行某些操作，如读取数据、插入数据、更新或修改数据、删除数据等。若要在数据库中执行操作，应执行相应的 SQL 语句或存储过程。例如，要从数据库读取一组行，则创建一个数据命令并用 SQL Select 语句的文本或获取记录的存储过程的名称配置它。

如果想要获取这些行，可执行以下操作：

(1) 打开一个连接；

(2) 执行该命令引用的 SQL 语句或存储过程；

(3) 关闭连接。

3.2 使用数据适配器与数据集执行数据库交互

在许多情况下，在应用程序运行过程中使连接保持打开状态是不实用或是不切实际的。这样做可能会失去将打开连接的需求降至最低所带来的许多好处。如果数据库并未被大部分时间空闲的连接占用，它可以为更多用户提供服务。

使用 ADO.NET 可以临时存储从数据库检索的记录，然后使用该临时集，这便是数据集(DataSet)的概念。数据集是从数据源检索的记录的缓存。可以用与操作实际数据十分类似的方式操作数据集。这样操作时，将保持与数据库的不连接状态，使数据库可以自由执行其他任务。使用数据适配器可以实现从数据库获取数据和将数据写回数据库。

使用这种方式进行数据库访问的基本过程是：首先，使用提供者类中的对象连接所要访问的数据库，将数据从该数据库读到存储介质中，然后使用使用者类中的对象在非连接的模式下对数据进行相应的操作，操作完毕后，再使用提供者类中的对象将对数据的改动更新到数据库中。

基本步骤如下：

(1) 创建一个数据库链接；

(2) 请求一个记录集合；

(3) 将记录集合暂存到 DataSet；

(4) 如果需要，返回第 2 步(DataSet 可容纳多个数据表集合)；

(5) 在 DataSet 上作所需要的操作；

(6) 更新数据源。

数据库编程的内容十分丰富，但最为基本编程的也就是那么几点：连接数据库、得到需要的数据和针对数据记录的浏览、插入、修改、删除等操作。下面就着重探讨一下在 Visual C#.NET 中如何进行基本的数据库编程，包括如何浏览记录、插入记录、修改记录和删除记录等基本操作。

4 在 C# 中用 ADO.NET 进行数据库操作编程

数据库访问是应用程序中最普遍的部分，而随着 C# 和 ADO.NET 的引入，这种操作已变成相对来说比较容易的事情。下面将示范四种基础的数据库操作：读取数据、写入数据、更新或修改数据、删除数据，上述的操作都是基于 Microsoft SQL Server 2000 数据库。

首先，需用 SQL Server 2000 建立一个数据库 test。在使用 ADO.NET 之前，需先把有关的命名空间导入进来。即将下面的代码加入到要进行数据库操作的 C# 程序中，其具体位置应该在类声明之前。

```
using System;
using System.Data;
using System.Data.SqlClient; //导入 SQL Server 数据操作类所在的命名空间
```

4.1 检索并显示数据

数据阅读器对象 DataReader 的功能是从数据源中读取只进且只读的数据流。因为每次在内存中始终只有一行，所以使用 DataReader 可提高应用程序的性能并减少系统开销。使用 SqlDataReader 对象检索并显示数据的代码如下：

```
string strConn = "Data Source = localhost;Integrated Security = SSPI;Initial Catalog = test";
//或者 string strConn = "server = localhost;Integrated Security = SSPI;database = test";
//或者 string strConn = "server = ServerName; User ID = userName;Password = thePassword;database = test");
//或者 string strConn = " server = ServerName; uid = userName;pwd = thePassword;database = test");
//? 上面给出了连接字符串的 4 种不同写法
string selectSQL = "SELECT * FROM 成绩表"; //"成绩表"是 test 数据库中的一个表
SqlConnection conn = new SqlConnection(strConn); //创建数据库连接
SqlCommand theCMD = new SqlCommand(selectSQL,conn); //创建一个 SqlCommand 对象
conn.Open(); //打开数据库
SqlDataReader theReader = theCMD.ExecuteReader(); //生成数据阅读器对象,从数据源中检索行
? ("学号" + "----" + "姓名" + "-----" + "数学");
while(accpReader.Read()) //循环遍历 SqlDataReader 中的每条记录,直到完成
{
    ? (accpReader.GetString(0) + "---" + accpReader.GetString(1) + "---" + accpReader.GetDecimal(2));
}
accpReader.Close(); //关闭数据阅读器
conn.Close(); //关闭连接
```

为简明起见,本文中用"?"表示输出命令,在 C# 中的实际代码应为 Console.WriteLine 或 Response.Write 等。

上面给出的是在保持连接的状态下,使用数据阅读器检索及显示数据。如果使用 SqlDataAdapter 对象和 DataSet 对象,则不必与数据库保持连接,代码如下:

```
string strConn = "server = localhost;Integrated Security = SSPI;database = test";
string selectSQL = "SELECT * FROM 成绩表";
SqlConnection conn = new SqlConnection(strConn);
SqlDataAdapter da = new SqlDataAdapter(selectSQL,conn);
//为简化书写,在后面的程序中将以上 4 行记为"代码段一"
DataSet theDS = new DataSet(); //创建数据集
da.Fill(theDS,"dsTable"); //填充数据集 theDS 中的 dsTable 表
? ("学号" + "----" + "姓名" + "-----" + "数学");
foreach(DataRow pRow in theDS.Tables["dsTable"].Rows)
{
    ? (pRow["学号"] + "-----" + pRow["姓名"] + "----" + pRow["数学"]);
}
```

几点说明:

(1) SqlCommand 特别提供了以下对 SQL Server 数据库执行命令的方法:ExecuteReader 执行返回行的命令,ExecuteNonQuery 执行 INSERT、DELELE、UPDATE 及 SET 语句等命令。

(2) SqlDataAdapter 是 DataSet 和 SQL Server 之间的桥接器。程序仅在填充(Fill)数据集或更新(Update)对数据集的修改时,才与数据库进行连接。

(3) 异常处理。作为优秀的程序员,应该将异常处理代码放在 try/catch/finally 结构中,确保能够控制所有异常情况。

```
try
{
    // 数据操作
}
catch(Exception e)
{
    //一些通常的异常处理
}
finally
{
    //释放 try 块中分配的资源
}
```

当确定应用程序应使用 DataReader 还是应使用 DataSet 时,应考虑应用程序所需的功能类型。DataSet 用于组合并关联来自多个源的数据,对数据执行大量的处理,而不需要与数据源保持打开的连接,从而将该连接释放给其他客户端使用。如果不需要 DataSet 所提供的功能,则可以使用 DataReader 以只进只读方式返回数据,从而提高应用程序的性能。因为这样能节省 DataSet 所使用的内存,并将省去创建 DataSet 并填充其内容所需的必要处理。

4.2 使用 SqlCommand 直接插入、更新和删除数据

插入、更新和删除操作可以用 SqlCommand 直接实现,通过调用 SQL 语句或存储过程并向其传递更新信息,可以将更新直接发送到数据库。下面通过几个例子予以说明。

下面的代码通过一个 SQL 命令插入一条记录:

```
string strConn = "server = localhost;Integrated Security = SSPI;database = test";
string sSQLCommand = " INSERT INTO 成绩表(学号,姓名,数学,语文) Values('98765','王杰',98,95)";
SqlConnection conn = new SqlConnection(strConn);
SqlCommand theCMD = new SqlCommand(sSQLCommand,conn); // 创建 SqlCommand 对象
conn.Open();
int nNoAffected = theCMD.ExecuteNonQuery(); //执行 Sql-
```

```
Command 命令
? ("RecordsAffected=" + nNoAffected.ToString());
conn.Close();
```

此处并不需要从数据库中返回任何内容，所以用了 Command 对象的 ExecuteNonQuery( ) 方法。ExecuteNonQuery( ) 用于执行 SQL INSERT、UPDATE 和 DELETE 语句或存储过程，返回的值是受该命令影响的行数。

下面的代码通过一个 SQL 命令更新记录(其他代码类似于插入操作)：

```
string sSQLCommand = " UPDATE 成绩表 SET 语文 = 100 WHERE 姓名='吴娜";
```

UPDATE 命令指示了要被修改的表。如果在表格中有 3 个'吴娜'，就会返回 3。

下面的代码通过一个 SQL 命令删除记录(其他代码类似于插入操作)：

```
string sSQLCommand=" DELETE FROM 成绩表 WHERE 姓名='刘义'";
```

DELETE 命令显示要被删除的记录，这可能会是几条。如果表中有 2 个'刘义'就返回 2，这两条记录都会被删除。

4.3 用 DataSet 来插入、更新和删除数据库中的数据

如果想要在断开与数据源的连接时使用一组表和行，则应当使用数据集。

插入、更新和删除操作需要使用 Command、DataAdapter、DataSet 对象，下面通过例子具体说明。

下面的代码在数据库中插入一条记录：

```
//此处为"代码段一"，省略未写
string insertSQL="INSERT INTO 成绩表(学号,姓名,数学,语文) Values('98765', '王杰',98,95)";
da.InsertCommand=new SqlCommand(insertSQL,conn);
DataSet catDS=new DataSet();
da.Fill(catDS,"dsTable");
DataRow dRow=catDS.Tables["dsTable"].NewRow(); //创建一新行
catDS.Tables["dsTable"].Rows.Add(dRow); //将记录插入数据集
da.Update(catDS,"dsTable"); //更新回数据源
? ("王杰加入完成");
```

下面的代码在数据库中更新记录：

```
//此处为"代码段一"，省略未写
string updateSQL="UPDATE 成绩表 SET 语文=100 WHERE 姓名='吴娜";
da.UpdateCommand=new SqlCommand(updateSQL,conn);
DataSet theDS=new DataSet();
da.Fill(theDS,"dsTable");……
foreach(DataRow dRow in theDS.Tables["dsTable"].Rows)
dRow["语文"]=100; //将"语文"成绩改为 100 分
da.Update(theDS,"dsTable");
? ("吴娜成绩修改完成");
```

在数据库中删除记录的代码略。

在将 DataSet 或 DataTable 与 DataAdapter 和关系型数据源一起使用时，用 DataRow 的 Delete 方法移除行。Delete 方法只是在 DataSet 或 DataTable 中将行标记为 Deleted，而没有真正删除它。而 DataAdapter 在遇到标记为 Deleted 的行时，会执行其 DeleteCommand 以在数据源中删除该行。

对于数据访问编程，使用数据集并非总是最佳的解决方案。当尝试从数据集更改写入数据源时，可能出现错误，典型的错误是数据库中的记录在被读取到数据集之后已经更改。如果试图更新已更改的记录，则将导致并发冲突。

以上实例运用了新一代数据访问技术 ADO.NET，简要介绍了 Visual C# 中关于数据库的基本编程——浏览记录、插入记录、更新记录和删除记录。在数据库的选用上，采用了当前比较典型通用的数据库 SQL Server 2000。

5 结束语

针对数据库编程始终是程序设计语言的一个重要方面，ADO.NET 是迄今为止最新的一种数据库接口编程技术，它提供了完整的基于网络的数据访问服务，不仅支持传统的基于连接的数据访问，同时也为更适合于把数据返回到客户端应用程序的无连接的数据提供高性能的访问支持，必将得到越来越广泛的使用。

参考文献

1 孙三才、许薰尹，精通 C# 与 ASP.net 程序设计，中国铁道出版社，2003－1。

2 陈钟、刘强、张高等，C# 编程语言程序设计与开发，清华大学出版社，2003－9。

3 钱昆等，C# 实用编程技术，中国水利水电出版社，2001－9。

4 桂思强，C#/Visual Basic.net 与数据库程序设计，中国铁道出版社，2003－3。

5 王浩然，C# 行家设计手册，中国铁道出版社，2002－3。

6 张哲峰，高效掌握 ADO.NET[M]，清华大学出版社，2003－3。