Tutorial 3: Executing Database Commands

Overview

- Database commands are executed through Command objects, and are part of the managed providers.
- There are three Command classes: SqlCommand, OleDbCommand, and OdbcCommand.
- You use a Command object to execute a SQL SELECT, INSERT, UPDATE, or DELETE statement.
- A Command object communicates with the database using a Connection object

Executing a SELECT Statement Using the ExecuteReader() Method

- Let's take a look at an example that executes a SELECT statement using the ExecuteReader() method.
- This method returns the result set in a DataReader object, which you can then use to read the rows returned by the database.
- For example, the following code creates the required objects and executes a SELECT statement that retrieves the top five rows from the Customers table:

```
// create a OleDbConnection object to connect to the database
System.Data.OleDb.OleDbConnection myOleDbConnection =
      new System.Data.OleDb.OleDbConnection
      ("provider=Microsoft.Jet.OLEDB.4.0;" + "data source="
      + Page.Server.MapPath("App_Data\\northwind.mdb"));

// create a OleDbCommand object
System.Data.OleDb.OleDbCommand myOleDbCommand
      =myOleDbConnection.CreateCommand();

// set the CommandText property of the OleDbCommand object to
// the SELECT statement
myOleDbCommand.CommandText = "SELECT TOP 5 CustomerID, CompanyName,
      ContactName, Address FROM Customers ORDER BY CustomerID";

// open the database connection using the
// Open() method of the OleDbConnection object
myOleDbConnection.Open();

// create a OleDbDataReader object and call the ExecuteReader()
// method of the OleDbCommand object to run the SQL SELECT statement
System.Data.OleDb.OleDbDataReader myOleDbDataReader =
       myOleDbCommand.ExecuteReader();
```

- You'll notice that I didn't call the Open() method of the OleDbConnection object until just before calling the ExecuteReader() method of the OleDbCommand object. This is intentional. By opening the connection at the very last moment, you minimize time spent connected to the database and therefore conserve database resources.

- The result set returned by myOleDbCommand is stored in myOleDbDataReader. You then read the rows from myOleDbDataReader using the Read() method.
- This method returns the Boolean true value when there is another row to read, otherwise it returns false.
- You can read an individual column value in a row from myOleDbDataReader by passing the name of the column in square brackets.
- For example, to read the CustomerID column, you use myOleDbDataReader["CustomerID"].
- You can also specify the column you want to get by passing a numeric value in brackets. For example, myOleDbDataReader[0] also returns the CustomerID column value. 0 corresponds to the first column in the table, which in this example is the CustomerID column.
- You can use the Read() method in a while loop to read each row in turn, as shown in the following example:

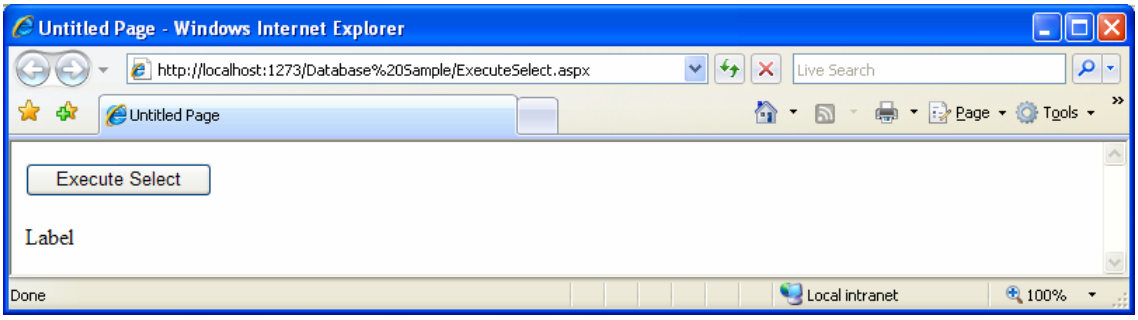
```
      // read the rows from the OleDbDataReader object using
      // the Read() method
      while (myOleDbDataReader.Read())
      {
            this.Label1.Text = "myOleDbDataReader[\" CustomerID\"] = "
                  + myOleDbDataReader["CustomerID"]
                  + "<br>"
                  + "myOleDbDataReader[\" CompanyName\"] = "
                  + myOleDbDataReader["CompanyName"]
                  + "<br>"
                  + "myOleDbDataReader[\" ContactName\"] = "
                  + myOleDbDataReader["ContactName"]
                  + "<br>"
                  + "myOleDbDataReader[\" Address\"] = "
                  + myOleDbDataReader["Address"]
                  + "<br><br>";
      }
```

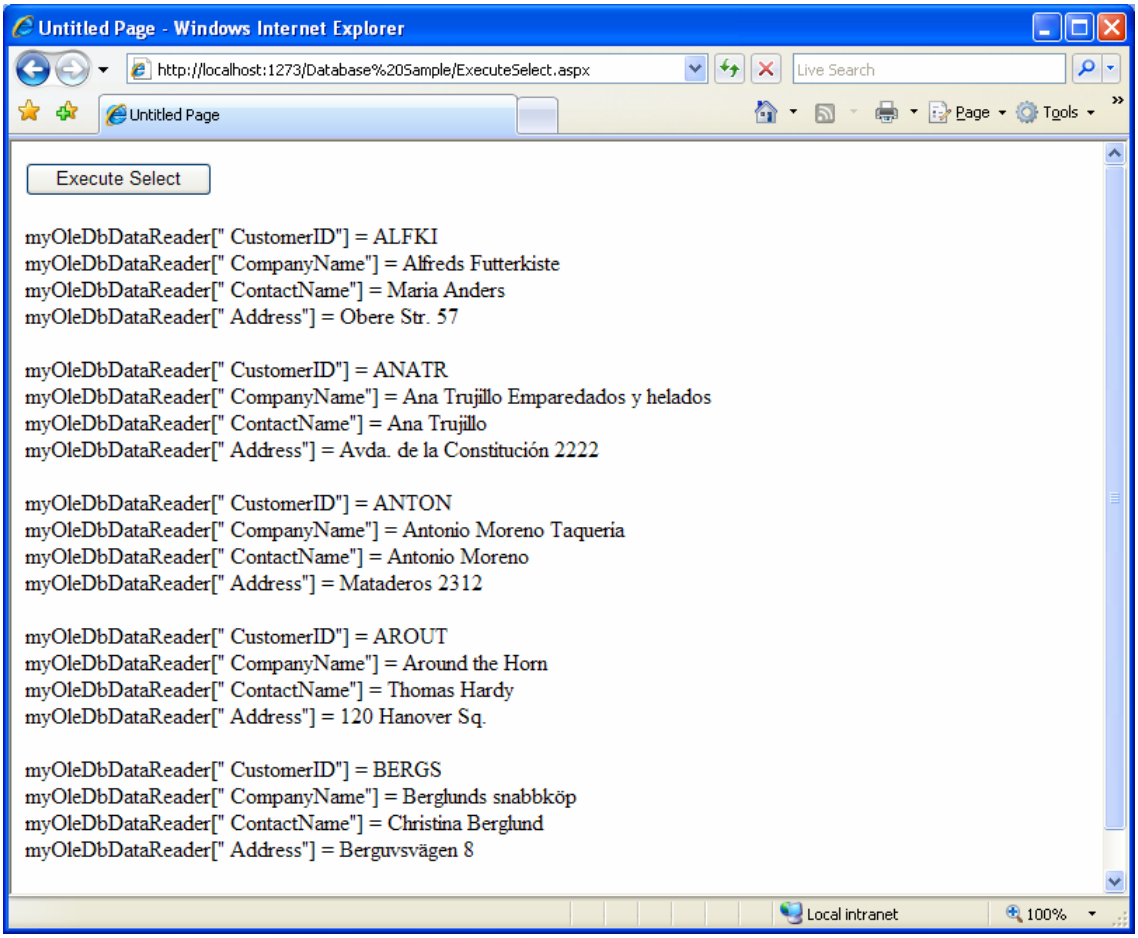
Program segment below illustrates a complete program that uses the code examples shown in this section.

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class ExecuteSelect : System.Web.UI.Page
{
      protected void Page_Load(object sender, EventArgs e)
      {

      }
```

```
	protected void Button1_Click(object sender, EventArgs e)
	{
		// create a OleDbConnection object to connect to the database
		System.Data.OleDb.OleDbConnection myOleDbConnection = new
			System.Data.OleDb.OleDbConnection(
			"provider=Microsoft.Jet.OLEDB.4.0;" + "data source="
			+ Page.Server.MapPath("App_Data\\northwind.mdb"));

		// create a OleDbCommand object
		System.Data.OleDb.OleDbCommand myOleDbCommand =
			myOleDbConnection.CreateCommand();

		// set the CommandText property of the OleDbCommand object to
		// the SELECT statement
		myOleDbCommand.CommandText = "SELECT TOP 5 CustomerID,
			CompanyName, ContactName, Address FROM Customers ORDER BY
			CustomerID";

		// open the database connection using the
		// Open() method of the OleDbConnection object
		myOleDbConnection.Open();

		// create a OleDbDataReader object and call the ExecuteReader()
		// method of the OleDbCommand object to run the SQL SELECT
		// statement
		System.Data.OleDb.OleDbDataReader myOleDbDataReader =
			myOleDbCommand.ExecuteReader();

		this.Label1.Text = "";

		// read the rows from the OleDbDataReader object using
		// the Read() method
		while (myOleDbDataReader.Read())
		{
			this.Label1.Text = this.Label1.Text
				+ "myOleDbDataReader[\" CustomerID\"] = "
				+ myOleDbDataReader["CustomerID"]
				+ "<br>"
				+ "myOleDbDataReader[\" CompanyName\"] = "
				+ myOleDbDataReader["CompanyName"]
				+ "<br>"
				+ "myOleDbDataReader[\" ContactName\"] = "
				+ myOleDbDataReader["ContactName"]
				+ "<br>"
				+ "myOleDbDataReader[\" Address\"] = "
				+ myOleDbDataReader["Address"]
				+ "<br><br>";
		}

		// close the OleDbDataReader object using the Close() method
		myOleDbDataReader.Close();

		// close the OleDbConnection object using the Close() method
		myOleDbConnection.Close();
	}
}
```

- The output from this program is as follows:





Executing INSERT, UPDATE, and DELETE Statements Using the ExecuteNonQuery() Method

- You can use the ExecuteNonQuery() method of a Command object to execute any command that doesn't return a result set from the database.
- In this section, you'll learn how to use the ExecuteNonQuery() method to execute commands that modify information in the database.
- Let's take a look at an example that executes an INSERT statement using the ExecuteNonQuery() method. First, a Command object is needed:

```
System.Data.OleDb.OleDbCommand myOleDbCommand =
      myOleDbConnection.CreateCommand();
```

- Next, you set the CommandText property of your Command object to the INSERT statement.
- The following example sets the CommandText property of mySqlCommand to an INSERT statement that adds a row to the Customers table:

```
myOleDbCommand.CommandText = "INSERT INTO Customers (CustomerID,
      CompanyName) VALUES ('J2COM', 'Jason Price Corporation')";
```

- Finally, you execute the INSERT statement using the ExecuteNonQuery() method:

```
int numberOfRows = myOleDbCommand.ExecuteNonQuery();
```

- The ExecuteNonQuery() method returns an int value that indicates the number of rows affected by the command.
- In this example, the value returned is the number of rows added to the Customers table, which is 1 since one row was added by the INSERT statement.

- Let's take a look at an example that executes an UPDATE statement to modify the new row just added.
- The following code sets the CommandText property of myOleDbCommand to an UPDATE statement that modifies the CompanyName column of the new row, and then calls the ExecuteNonQuery() method to execute the UPDATE:

```
myOleDbCommand.CommandText = "UPDATE Customers SET CompanyName = 'New
      Company' WHERE CustomerID = 'J2COM'";

numberOfRows = myOleDbCommand.ExecuteNonQuery();
```

- The ExecuteNonQuery() method returns the number of rows modified by the UPDATE statement, which is 1 since one row was modified.

- Finally, let's take a look at an example that executes a DELETE statement to remove the new row:

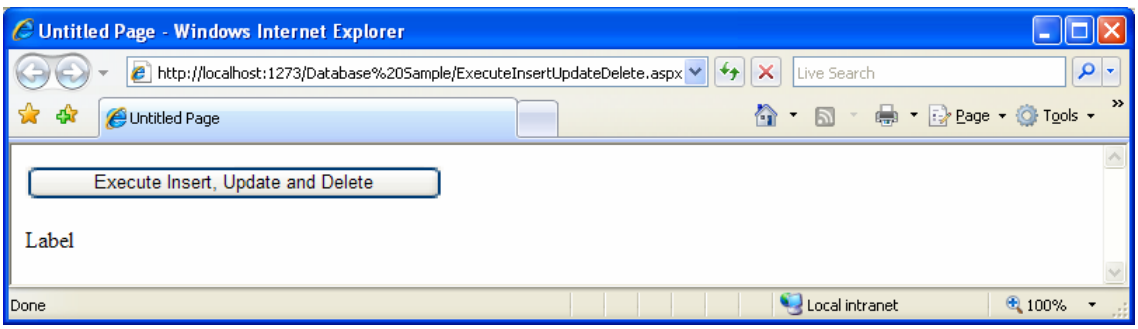
```
      myOleDbCommand.CommandText = "DELETE FROM Customers WHERE CustomerID
            = 'J2COM'";

      numberOfRows = myOleDbCommand.ExecuteNonQuery();
```

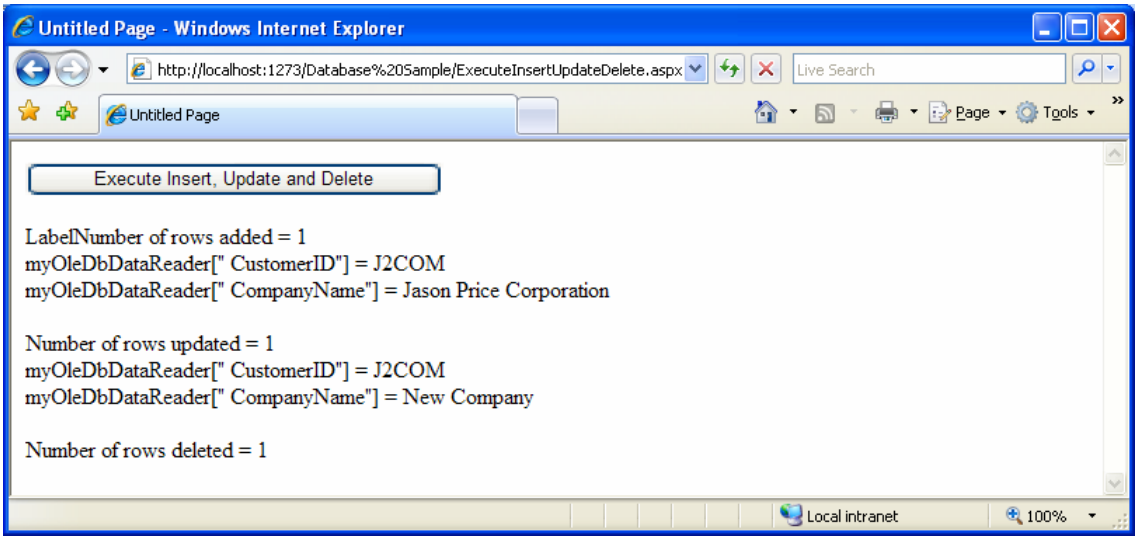
- ExecuteNonQuery() returns 1 again because only one row was removed by the DELETE statement.
- Program segment below illustrates the use of the ExecuteNonQuery() method to execute the INSERT, UPDATE, and DELETE statements shown in this section. This program features a procedure named DisplayRow() that retrieves and displays the details of a specified row from the Customers table. DisplayRow() is used in the program to show the result of the INSERT and UPDATE statements.

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class ExecuteInsertUpdateDelete : System.Web.UI.Page
{
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      private void DisplayRow(System.Data.OleDb.OleDbCommand
      myOleDbCommand, string CustomerID)
      {
            myOleDbCommand.CommandText = "SELECT CustomerID, CompanyName
                  FROM Customers WHERE CustomerID = '" + CustomerID + "'";

            System.Data.OleDb.OleDbDataReader myOleDbDataReader =
                  myOleDbCommand.ExecuteReader();

            while (myOleDbDataReader.Read())
            {
                  this.Label1.Text = this.Label1.Text
                        + "myOleDbDataReader[\" CustomerID\"] = "
                        + myOleDbDataReader["CustomerID"]
                        + "<br>"
                        + "myOleDbDataReader[\" CompanyName\"] = "
                        + myOleDbDataReader["CompanyName"]
                        + "<br><br>";
            }

            myOleDbDataReader.Close();
      }

      protected void Button1_Click(object sender, EventArgs e)
      {
            System.Data.OleDb.OleDbConnection myOleDbConnection = new
                  System.Data.OleDb.OleDbConnection(
```

```
                "provider=Microsoft.Jet.OLEDB.4.0;" + "data source="
                + Page.Server.MapPath("App_Data\\northwind.mdb"));

            // create a OleDbCommand object and set its Commandtext
            // property to an INSERT statement
            System.Data.OleDb.OleDbCommand myOleDbCommand =
                myOleDbConnection.CreateCommand();
            myOleDbCommand.CommandText = "INSERT INTO Customers
                (CustomerID, CompanyName) VALUES ('J2COM', 'Jason Price
                Corporation')";

            myOleDbConnection.Open();

            // call the ExecuteNonQuery() method of the OleDbCommand object
            // to run the INSERT statement
            int numberOfRows = myOleDbCommand.ExecuteNonQuery();
            this.Label1.Text = this.Label1.Text + "Number of rows added = "
                + numberOfRows + "<br>";
            DisplayRow(myOleDbCommand, "J2COM");

            // set the CommandText property of the OleDbCommand object to
            // an UPDATE statement
            myOleDbCommand.CommandText = "UPDATE Customers SET CompanyName
                = 'New Company' WHERE CustomerID = 'J2COM'";

            // call the ExecuteNonQuery() method of the OleDbCommand object
            // to run the UPDATE statement
            numberOfRows = myOleDbCommand.ExecuteNonQuery();
            this.Label1.Text = this.Label1.Text
                + "Number of rows updated = " + numberOfRows + "<br>";
            DisplayRow(myOleDbCommand, "J2COM");

            // set the CommandText property of the OleDbCommand object to
            // a DELETE statement
            myOleDbCommand.CommandText = "DELETE FROM Customers WHERE
                CustomerID = 'J2COM'";

            // call the ExecuteNonQuery() method of the OleDbCommand object
            // to run the DELETE statement
            numberOfRows = myOleDbCommand.ExecuteNonQuery();
            this.Label1.Text = this.Label1.Text
                + "Number of rows deleted = " + numberOfRows + "<br>";

            myOleDbConnection.Close();
        }
}
```

- The output from this program is as follows:




Untitled Page - Windows Internet Explorer
http://localhost:1273/Database%20Sample/ExecuteInsertUpdateDelete.aspx
Live Search
Untitled Page
Page
Tools
Execute Insert, Update and Delete
LabelNumber of rows added = 1
myOleDbDataReader[" CustomerID"] = J2COM
myOleDbDataReader[" CompanyName"] = Jason Price Corporation
Number of rows updated = 1
myOleDbDataReader[" CustomerID"] = J2COM
myOleDbDataReader[" CompanyName"] = New Company
Number of rows deleted = 1
Local intranet
100%