.NET Framework Class Library

**SqlCommand.ExecuteScalar Method**

Executes the query, and returns the first column of the first row in the result set returned by the query. Additional columns or rows are ignored.

**Namespace:**  [System.Data.SqlClient](http://msdn.microsoft.com/en-us/library/system.data.sqlclient.aspx)  
**Assembly:**  System.Data (in System.Data.dll)

http://i.msdn.microsoft.com/Global/Images/clear.gif Syntax

Visual Basic (Declaration)

Public Overrides Function ExecuteScalar As Object

Visual Basic (Usage)

Dim instance As SqlCommand

Dim returnValue As Object

returnValue = instance.ExecuteScalar()

C#

public override Object ExecuteScalar()

Visual C++

public:

virtual Object^ ExecuteScalar() override

JScript

public override function ExecuteScalar() : Object

**Return Value**

Type: [System..::.Object](http://msdn.microsoft.com/en-us/library/system.object.aspx)  
The first column of the first row in the result set, or a null reference (**Nothing** in Visual Basic) if the result set is empty. Returns a maximum of 2033 characters.

**Implements**

[IDbCommand..::.ExecuteScalar()()()](http://msdn.microsoft.com/en-us/library/system.data.idbcommand.executescalar.aspx)

http://i.msdn.microsoft.com/Global/Images/clear.gif Exceptions

|  |  |
| --- | --- |
| **Exception** | **Condition** |
| [SqlException](http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlexception.aspx) | An exception occurred while executing the command against a locked row. This exception is not generated when you are using Microsoft .NET Framework version 1.0. |

http://i.msdn.microsoft.com/Global/Images/clear.gif Remarks

Use the **ExecuteScalar** method to retrieve a single value (for example, an aggregate value) from a database. This requires less code than using the [ExecuteReader](http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlcommand.executereader.aspx) method, and then performing the operations that you need to generate the single value using the data returned by a [SqlDataReader](http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqldatareader.aspx).

A typical **ExecuteScalar** query can be formatted as in the following C# example:

[[http://i.msdn.microsoft.com/Global/Images/clear.gif](javascript:CopyCode('ctl00_mainContentContainer_ctl38other');)Copy Code](javascript:CopyCode('ctl00_mainContentContainer_ctl38other');)

cmd.CommandText = "SELECT COUNT(\*) FROM dbo.region";

Int32 count = (Int32) cmd.ExecuteScalar();

http://i.msdn.microsoft.com/Global/Images/clear.gif Examples

The following example creates a [SqlCommand](http://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlcommand.aspx) and then executes it using **ExecuteScalar**. The example is passed a string representing a new value to be inserted into a table, and a string to use to connect to the data source. The function returns the new **Identity** column value if a new row was inserted, 0 on failure.

Visual Basic

[[http://i.msdn.microsoft.com/Global/Images/clear.gif](javascript:CopyCode('ctl00_mainContentContainer_ctl46VisualBasic');)Copy Code](javascript:CopyCode('ctl00_mainContentContainer_ctl46VisualBasic');)

Public Function AddProductCategory( \_

ByVal newName As String, ByVal connString As String) As Integer

Dim newProdID As Int32 = 0

Dim sql As String = \_

"INSERT INTO Production.ProductCategory (Name) VALUES (@Name); " \_

& "SELECT CAST(scope\_identity() AS int);"

Using conn As New SqlConnection(connString)

Dim cmd As New SqlCommand(sql, conn)

cmd.Parameters.Add("@Name", SqlDbType.VarChar)

cmd.Parameters("@Name").Value = newName

Try

conn.Open()

newProdID = Convert.ToInt32(cmd.ExecuteScalar())

Catch ex As Exception

Console.WriteLine(ex.Message)

End Try

End Using

Return newProdID

End Function

C#

[[http://i.msdn.microsoft.com/Global/Images/clear.gif](javascript:CopyCode('ctl00_mainContentContainer_ctl47CSharp');)Copy Code](javascript:CopyCode('ctl00_mainContentContainer_ctl47CSharp');)

static public int AddProductCategory(string newName, string connString)

{

Int32 newProdID = 0;

string sql =

"INSERT INTO Production.ProductCategory (Name) VALUES (@Name); "

+ "SELECT CAST(scope\_identity() AS int)";

using (SqlConnection conn = new SqlConnection(connString))

{

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.Add("@Name", SqlDbType.VarChar);

cmd.Parameters["@name"].Value = newName;

try

{

conn.Open();

newProdID = (Int32)cmd.ExecuteScalar();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

return (int)newProdID;

}