CREATE TYPE [dbo].[CustomerType] AS TABLE(

      [Id] [int] NULL,

      [Name] [varchar](100) NULL,

      [Country] [varchar](50) NULL

)

GO

CREATE PROCEDURE [dbo].[Update\_Khoas]

@tbl KhoaType READONLY

AS

BEGIN

SET NOCOUNT ON;

--UPDATE EXISTING RECORDS

UPDATE KHOA

SET TenKhoa = c2.TenKhoa

FROM KHOA c1

INNER JOIN @tblKhoas c2

ON c1.ID = c2.ID

--INSERT NON-EXISTING RECORDS

INSERT INTO KHOA

SELECT TenKhoa

FROM @tblKhoas

WHERE ID NOT IN(SELECT ID FROM KHOA)

END

CREATE TYPE [dbo].[SinhVienType] AS TABLE(

[MaSV] [int] NULL,

[HoSV] [nvarchar](255) NULL,

[TenSV] [nvarchar](255) NULL,

[NgaySinh] [varchar](50) NULL,

[Lop] [varchar](50) NULL

)

GO

CREATE PROCEDURE [dbo].[sp\_CheckSV]

@tbl SinhVienType READONLY

AS

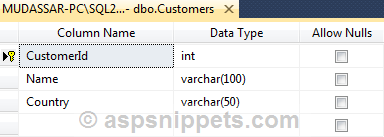
BEGIN

select \* from @tblKhoas c1 where NOT EXISTS(select c2.ID from LOP c2 where c1.Lop = c2.MaLop)

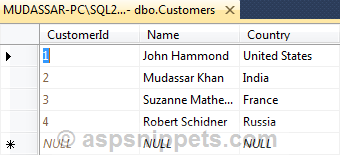
END

**Chi Tiết**

I have made use of the following table Customers with the schema as follows.



I have already inserted few records in the table.



Then I have created a User Defined Table Type in SQL Server using the following query

CREATE TYPE [dbo].[CustomerType] AS TABLE(

      [Id] [int] NULL,

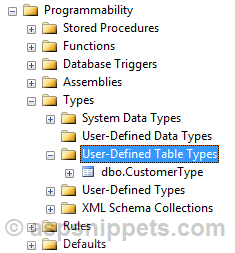
      [Name] [varchar](100) NULL,

      [Country] [varchar](50) NULL

)

GO

Once created the User Defined Type will be visible in the Object Explorer as shown below.



Finally the following stored procedure is created which will accept the DataTable as parameter and then will insert all records into the table that are not present in the table and the one that already exists will be updated.

SQL Server 2008 or higher versions

SQL Server 2008 came up with a nice function called MERGE, which allows to perform INSERT operation when records are not present and UPDATE when records are present in the table.

CREATE PROCEDURE [dbo].[Update\_Customers]

      @tblCustomers CustomerType READONLY

AS

BEGIN

      SET NOCOUNT ON;

      MERGE INTO Customers c1

      USING @tblCustomers c2

      ON c1.CustomerId=c2.Id

      WHEN MATCHED THEN

      UPDATE SET c1.Name = c2.Name

            ,c1.Country = c2.Country

      WHEN NOT MATCHED THEN

      INSERT VALUES(c2.Id, c2.Name, c2.Country);

END

SQL Server 2005 version

The below Stored Procedure can be used where in the SQL Server version 2005 where MERGE function is not supported. It first performs UPDATE using INNER JOIN and then INSERTS all records that are not present in the table.

CREATE PROCEDURE [dbo].[Update\_Customers]

      @tblCustomers CustomerType READONLY

AS

BEGIN

      SET NOCOUNT ON;

      --UPDATE EXISTING RECORDS

      UPDATE Customers

      SET Name = c2.Name

      ,Country = c2.Country

      FROM Customers c1

      INNER JOIN @tblCustomers c2

      ON c1.CustomerId = c2.Id

      --INSERT NON-EXISTING RECORDS

      INSERT INTO Customers

      SELECT Id, Name, Country

      FROM @tblCustomers

      WHERE Id NOT IN(SELECT CustomerId FROM Customers)

END

**Note**: The SQL for creating the Table and Stored Procedures is provided in the attached sample code.

**HTML Markup**

The HTML Markup consists of an ASP.Net GridView and a Button for inserting and updating bulk data from GridView to SQL Server Database table.

<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false">

<Columns>

    <asp:BoundField DataField="Id" HeaderText="Id" ItemStyle-Width="30" />

    <asp:BoundField DataField="Name" HeaderText="Name" ItemStyle-Width="150" />

    <asp:TemplateField>

        <ItemTemplate>

            <asp:TextBox ID="txtCountry" Text='<%# Eval("Country") %>' runat="server" />

        </ItemTemplate>

    </asp:TemplateField>

</Columns>

</asp:GridView>

<br />

<asp:Button Text="Bulk Update" OnClick="Bulk\_Update" runat="server" />

**Namespaces**

You will need to import the following namespaces.

**C#**

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

**VB.Net**

Imports System.Data

Imports System.Data.SqlClient

Imports System.Configuration

**Populating the GridView**

In this article I am populating the GridView using the following XML file in the Page Load event. The first 4 records are already present in the Table, the only difference is that the countries of some records are modified for illustrating the bulk update operation.

<?xmlversion="1.0"standalone="yes"?>

<Customers>

    <Customer>

        <Id>1</Id>

        <Name>John Hammond</Name>

        <Country>Brazil</Country>

    </Customer>

    <Customer>

        <Id>2</Id>

        <Name>Mudassar Khan</Name>

        <Country>India</Country>

    </Customer>

    <Customer>

        <Id>3</Id>

        <Name>Suzanne Mathews</Name>

        <Country>Germany</Country>

    </Customer>

    <Customer>

        <Id>4</Id>

        <Name>Robert Schidner</Name>

        <Country>Russia</Country>

    </Customer>

    <Customer>

        <Id>5</Id>

        <Name>Shen Ching</Name>

        <Country>China</Country>

    </Customer>

    <Customer>

        <Id>6</Id>

        <Name>Max Haynes</Name>

        <Country>Australia</Country>

    </Customer>

</Customers>

**C#**

protected void Page\_Load(object sender, EventArgs e)

{

    if (!this.IsPostBack)

    {

        DataSet ds = new DataSet();

        ds.ReadXml(Server.MapPath("~/Customers.xml"));

        GridView1.DataSource = ds.Tables[0];

        GridView1.DataBind();

    }

}

**VB.Net**

Protected Sub Page\_Load(sender As Object, e As EventArgs) Handles Me.Load

    If Not Me.IsPostBack Then

        Dim ds As New DataSet()

        ds.ReadXml(Server.MapPath("~/Customers.xml"))

        GridView1.DataSource = ds.Tables(0)

        GridView1.DataBind()

    End If

End Sub



**Bulk Insert records and Update existing rows if record exists**

The following event handler is executed on the click of the Button. Here first a DataTable is created with column schema same as that of the User Defined Table Type that was created and then a loop is executed of the GridView rows.

Inside the loop, the values of the cells of the rows are fetched and are inserted into the DataTable.

Finally the Stored Procedure is executed with the DataTable passed as parameter to it.

**C#**

protected void Bulk\_Update(object sender, EventArgs e)

{

    DataTable dt = new DataTable();

    dt.Columns.AddRange(new DataColumn[3] { new DataColumn("Id", typeof(int)),

                new DataColumn("Name", typeof(string)),

                new DataColumn("Country",typeof(string)) });

    foreach (GridViewRow row in GridView1.Rows)

    {

        int id = int.Parse(row.Cells[0].Text);

        string name = row.Cells[1].Text;

        string country = (row.FindControl("txtCountry") as TextBox).Text;

        dt.Rows.Add(id, name, country);

    }

    string constr = ConfigurationManager.ConnectionStrings["constr"].ConnectionString;

    using (SqlConnection con = new SqlConnection(constr))

    {

        using (SqlCommand cmd = new SqlCommand("Update\_Customers"))

        {

            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Connection = con;

            cmd.Parameters.AddWithValue("@tblCustomers", dt);

            con.Open();

            cmd.ExecuteNonQuery();

            con.Close();

        }

    }

}

**VB.Net**

Protected Sub Bulk\_Update(sender As Object, e As EventArgs)

    Dim dt As New DataTable()

    dt.Columns.AddRange(New DataColumn(2) {New DataColumn("Id", GetType(Integer)), \_

                                           New DataColumn("Name", GetType(String)), \_

                                           New DataColumn("Country", GetType(String))})

    For Each row As GridViewRow In GridView1.Rows

        Dim id As Integer = Integer.Parse(row.Cells(0).Text)

        Dim name As String = row.Cells(1).Text

        Dim country As String = TryCast(row.FindControl("txtCountry"), TextBox).Text

        dt.Rows.Add(id, name, country)

    Next

    Dim constr As String = ConfigurationManager.ConnectionStrings("constr").ConnectionString

    Using con As New SqlConnection(constr)

        Using cmd As New SqlCommand("Update\_Customers")

            cmd.CommandType = CommandType.StoredProcedure

            cmd.Connection = con

            cmd.Parameters.AddWithValue("@tblCustomers", dt)

            con.Open()

            cmd.ExecuteNonQuery()

            con.Close()

        End Using

    End Using

End Sub

**Screenshot**

The following screenshot displays the Customers table containing the updated as well as the inserted records.

