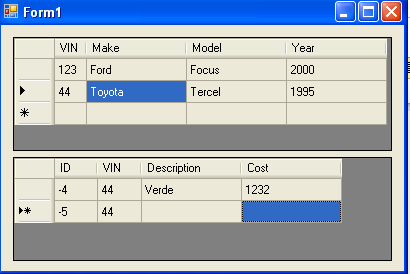
In the form include 2 DataGridViews

* dgVehicles.DataMember
* dgRepairs.DataSource



namespace VehicleRepairLab

{

public partial class Form1 : Form

{

private const string xsdFile = "VehiclesRepairs.xsd";

private const string xmlFile = "VehiclesRepairs.xml";

private DataSet ds;

private string cnString;

private SqlDataAdapter daVehicles;

private SqlDataAdapter daRepairs;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

cnString = ConfigurationManager

.ConnectionStrings["db"]

.ConnectionString;

InitializeDataAdapters();

PopulateDataSet();

dgVehicles.DataSource = ds;

dgVehicles.DataMember = "Vehicles";

dgRepairs.DataSource = ds;

dgRepairs.DataMember = "Vehicles.vehicles\_repairs";

}

private void PopulateDataSet()

{

if (File.Exists(xsdFile))

{

ds = new DataSet();

ds.ReadXmlSchema(xsdFile);

}

else

{

CreateSchema();

}

if (File.Exists(xmlFile))

{

ds.ReadXml(xmlFile, XmlReadMode.IgnoreSchema);

}

Synchronize();

}

private bool CheckConnectivity()

{

try

{

using(var cn = new SqlConnection(cnString))

{

cn.Open();

var version = cn.ServerVersion;

}

}

catch

{

return false;

}

return true;

}

private void CreateSchema()

{

ds = new DataSet("VehiclesRepairs");

var vehicles = ds.Tables.Add("Vehicles");

vehicles.Columns.Add("VIN", typeof(string));

vehicles.Columns.Add("Make", typeof(string));

vehicles.Columns.Add("Model", typeof(string));

vehicles.Columns.Add("Year", typeof(int));

vehicles.PrimaryKey = new DataColumn[] { vehicles.Columns["VIN"] };

var repairs = ds.Tables.Add("Repairs");

var pk = repairs.Columns.Add("ID", typeof(int));

pk.AutoIncrement = true;

pk.AutoIncrementSeed = -1;

pk.AutoIncrementStep = -1;

repairs.Columns.Add("VIN", typeof(string));

repairs.Columns.Add("Description", typeof(string));

repairs.Columns.Add("Cost", typeof(decimal));

repairs.PrimaryKey = new DataColumn[] { repairs.Columns["Id"] };

ds.Relations.Add(

"vehicles\_repairs",

vehicles.Columns["VIN"],

repairs.Columns["VIN"]);

ds.WriteXmlSchema(xsdFile);

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

Synchronize();

ds.WriteXml(xmlFile, XmlWriteMode.DiffGram);

}

private void gridError(object sender, DataGridViewDataErrorEventArgs e)

{

MessageBox.Show(e.Exception.Message);

}

public void InitializeDataAdapters()

{

//do vehicles with the SQL Command Builder;

daVehicles = new SqlDataAdapter("SELECT \* FROM Vehicles", cnString);

var bldVehicles = new SqlCommandBuilder(daVehicles);

//do repairs by creating all commands

var cn = new SqlConnection(cnString);

var cmdSelectRepairs = cn.CreateCommand();

var cmdUpdateRepairs = cn.CreateCommand();

var cmdDeleteRepairs = cn.CreateCommand();

var cmdInsertRepairs = cn.CreateCommand();

cmdSelectRepairs.CommandText =

"SELECT \* FROM Repairs";

cmdInsertRepairs.CommandText =

"INSERT Repairs(VIN,Description, Cost) "

+ " OUTPUT inserted.\* "

+ " VALUES( @VIN, @Description, @Cost); ";

cmdInsertRepairs.Parameters.Add("@VIN", SqlDbType.VarChar, 20, "VIN");

cmdInsertRepairs.Parameters.Add("@Description", SqlDbType.VarChar,

60, "Description");

cmdInsertRepairs.Parameters.Add("@Cost", SqlDbType.Money, 0, "Cost");

cmdUpdateRepairs.CommandText =

"UPDATE Repairs SET "

+ " VIN=@VIN, Description=@Description, Cost=@Cost "

+ " WHERE ID=@OriginalID "

+ " AND VIN=@OriginalVIN "

+ " AND Description=@OriginalDescription "

+ " AND Cost=@OriginalCost";

cmdUpdateRepairs.Parameters.Add("@OriginalID", SqlDbType.Int, 0, "ID")

.SourceVersion = DataRowVersion.Original;

cmdUpdateRepairs.Parameters.Add("@VIN", SqlDbType.VarChar, 20, "VIN");

cmdUpdateRepairs.Parameters.Add("@OriginalVIN", SqlDbType.VarChar,

20, "VIN").SourceVersion = DataRowVersion.Original;

cmdUpdateRepairs.Parameters.Add("@Description", SqlDbType.VarChar,

60, "Description");

cmdUpdateRepairs.Parameters.Add("@OriginalDescription",

SqlDbType.VarChar, 20, "Description")

.SourceVersion = DataRowVersion.Original;

cmdUpdateRepairs.Parameters.Add("@Cost", SqlDbType.Money, 0, "Cost");

cmdUpdateRepairs.Parameters.Add("@OriginalCost", SqlDbType.Money,

0, "Cost").SourceVersion = DataRowVersion.Original;

cmdDeleteRepairs.CommandText =

"DELETE Repairs "

+ " WHERE ID=@OriginalID "

+ " AND VIN=@OriginalVIN "

+ " AND Description=@OriginalDescription "

+ " AND Cost=@OriginalCost";

cmdDeleteRepairs.Parameters.Add("@OriginalID", SqlDbType.Int, 0, "ID")

.SourceVersion = DataRowVersion.Original;

cmdDeleteRepairs.Parameters.Add("@OriginalVIN", SqlDbType.VarChar,

20, "VIN").SourceVersion = DataRowVersion.Original;

cmdDeleteRepairs.Parameters.Add("@OriginalDescription",

SqlDbType.VarChar, 20, "Description")

.SourceVersion = DataRowVersion.Original;

cmdDeleteRepairs.Parameters.Add("@OriginalCost", SqlDbType.Money,

0, "Cost").SourceVersion = DataRowVersion.Original;

daRepairs = new SqlDataAdapter(cmdSelectRepairs);

daRepairs.InsertCommand = cmdInsertRepairs;

daRepairs.UpdateCommand = cmdUpdateRepairs;

daRepairs.DeleteCommand = cmdDeleteRepairs;

}

private void CreateTablesIfNotExisting()

{

try

{

using (var cn = new SqlConnection(cnString))

using (var cmd = cn.CreateCommand())

{

cn.Open();

cmd.CommandText =

"IF NOT EXISTS ( "

+ " SELECT \* FROM sys.Tables WHERE NAME='Vehicles') "

+ " CREATE TABLE Vehicles( "

+ " VIN varchar(20) PRIMARY KEY, "

+ " Make varchar(20), "

+ " Model varchar(20), Year int)";

cmd.ExecuteNonQuery();

cmd.CommandText =

"IF NOT EXISTS ( "

+ " SELECT \* FROM sys.Tables WHERE NAME='Repairs') "

+ " CREATE TABLE Repairs( "

+ " ID int IDENTITY PRIMARY KEY, "

+ " VIN varchar(20), "

+ " Description varchar(60), "

+ " Cost money)";

cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void Synchronize()

{

if (CheckConnectivity())

{

CreateTablesIfNotExisting();

SyncData();

}

}

private void SyncData()

{

//send changes

using (var tran = new TransactionScope())

{

try

{

daVehicles.Update(ds, "Vehicles");

daRepairs.Update(ds, "Repairs");

ds.AcceptChanges();

tran.Complete();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

//retrieve updates

var tempVehicles = new DataTable();

daVehicles.Fill(tempVehicles);

ds.Tables["Vehicles"].Merge(tempVehicles);

//merge changes

var tempRepairs = new DataTable();

daRepairs.Fill(tempRepairs);

ds.Tables["Repairs"].Merge(tempRepairs);

}

private void dgVehicles\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}