Leardo Taylor

The following is the source-code of the TruTexts Ledger

Technical Document

TruText Database Driver Application

**Login Page**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace TruText\_Ledger

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

DataAccess.ConnectToDatabase();

}

private void btnLogin\_Click(object sender, RoutedEventArgs e)

{

string AccountName = txtAccountName.Text;

string AccountPassword = txtAccountPassword.Password;

int iAccount = 0;

iAccount = DataAccess.Login(AccountName, AccountPassword);

if (iAccount == 1)

{

MessageBox.Show("User name and password is correct!");

this.Hide();

DBView nextWindow = new DBView();

nextWindow.ShowDialog();

}

else

{

MessageBox.Show("The account and password combination does not exist!");

}

}

}

}

**Viewer Page**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace TruText\_Ledger

{

/// <summary>

/// Interaction logic for DBView.xaml

/// </summary>

public partial class DBView : Window

{

public DBView()

{

InitializeComponent();

DataAccess.SelectTable = "Customers";

DataAccess.SelectOperation = "Select a mode!";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

if (DataAccess.SelectPrivilege == "ADMIN")

{

Layout1();

}

else if (DataAccess.SelectPrivilege == "NORML")

{

Layout2();

}

else

{

DataAccess.SelectPrivilege = "NORML";

Layout2();

MessageBox.Show("There was an error idenitfying your SelectPrivilege! Default set to Normal User");

}

}

private void btnCust\_Click(object sender, RoutedEventArgs e)

{

DataAccess.SelectTable = "Customers";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnOrder\_Click(object sender, RoutedEventArgs e)

{

DataAccess.SelectTable = "Orders";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnBook\_Click(object sender, RoutedEventArgs e)

{

DataAccess.SelectTable = "Books";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnSupp\_Click(object sender, RoutedEventArgs e)

{

DataAccess.SelectTable = "Suppliers";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnEmpl\_Click(object sender, RoutedEventArgs e)

{

DataAccess.SelectTable = "Employees";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnRole\_Click(object sender, RoutedEventArgs e)

{

DataAccess.SelectTable = "Roles";

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnLogout\_Click(object sender, RoutedEventArgs e)

{

this.Hide();

MainWindow previousWindow = new MainWindow();

previousWindow.Show();

}

private void btnEdit\_Click(object sender, RoutedEventArgs e)

{

DataManipulationViewer nextWindow = new DataManipulationViewer();

nextWindow.ShowDialog();

}

private void btnRfsh\_Click(object sender, RoutedEventArgs e)

{

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("").DefaultView;

}

private void btnFlt1\_Click(object sender, RoutedEventArgs e)

{

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("MstSld").DefaultView;

}

private void btnFlt2\_Click(object sender, RoutedEventArgs e)

{

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("BstCst").DefaultView;

}

private void btnFlt3\_Click(object sender, RoutedEventArgs e)

{

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("MstSls").DefaultView;

}

private void btnFlt4\_Click(object sender, RoutedEventArgs e)

{

dgvTableView.ItemsSource = DataAccess.LoadToDataGrid("MstPrvd").DefaultView;

}

public void Layout1()

{

btnSupp.Visibility = Visibility.Visible;

btnEmpl.Visibility = Visibility.Visible;

btnRole.Visibility = Visibility.Visible;

btnEdit.Visibility = Visibility.Visible;

btnRfsh.Visibility = Visibility.Visible;

btnFlt1.Visibility = Visibility.Visible;

btnFlt2.Visibility = Visibility.Visible;

btnFlt3.Visibility = Visibility.Visible;

btnFlt4.Visibility = Visibility.Visible;

}

public void Layout2()

{

btnSupp.Visibility = Visibility.Collapsed;

btnEmpl.Visibility = Visibility.Collapsed;

btnRole.Visibility = Visibility.Collapsed;

btnEdit.Visibility = Visibility.Collapsed;

btnRfsh.Visibility = Visibility.Collapsed;

btnFlt1.Visibility = Visibility.Collapsed;

btnFlt2.Visibility = Visibility.Collapsed;

btnFlt3.Visibility = Visibility.Collapsed;

btnFlt4.Visibility = Visibility.Collapsed;

}

}

}

**Editor Page**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

namespace TruText\_Ledger

{

/// <summary>

/// Interaction logic for DataManipulationViewer.xaml

/// </summary>

public partial class DataManipulationViewer : Window

{

public DataManipulationViewer()

{

InitializeComponent();

ClearViewer();

}

private void btnSave\_Click(object sender, RoutedEventArgs e)

{

if (DataAccess.SelectOperation == "Insert")

{

switch(DataAccess.SelectTable)

{

case ("Customers"):

MessageBox.Show(DataAccess.ExecuteOperation(txt1.Text, txt2.Text, txt3.Text, txt4.Text, txt5.Text, txt6.Text, ""));

ClearViewer();

break;

case ("Orders"):

MessageBox.Show(DataAccess.ExecuteOperation(txt1.Text, cmb2.Text, cmb3.Text, cmb4.Text, txt5.Text, cmb6.Text, ""));

ClearViewer();

break;

case ("Books"):

MessageBox.Show(DataAccess.ExecuteOperation(txt1.Text, txt2.Text, txt3.Text, txt4.Text, txt5.Text, txt6.Text, cmb7.Text));

ClearViewer();

break;

case ("Suppliers"):

MessageBox.Show(DataAccess.ExecuteOperation(txt1.Text, txt2.Text, txt3.Text, txt4.Text, txt5.Text, txt6.Text, txt7.Text));

ClearViewer();

break;

case ("Employees"):

MessageBox.Show(DataAccess.ExecuteOperation(txt1.Text, txt2.Text, txt3.Text, txt4.Text, cmb5.Text, txt6.Text, txt7.Text));

ClearViewer();

break;

case ("Roles"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, txt2.Text, txt3.Text, cmb4.Text, "", "", ""));

ClearViewer();

break;

}

}

else if (DataAccess.SelectOperation == "Update")

{

switch (DataAccess.SelectTable)

{

case ("Customers"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, cmb2.Text, cmb3.Text, cmb4.Text, cmb5.Text, cmb6.Text, ""));

ClearViewer();

break;

case ("Orders"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, cmb2.Text, cmb3.Text, cmb4.Text, cmb5.Text, cmb6.Text, ""));

ClearViewer();

break;

case ("Books"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, cmb2.Text, cmb3.Text, cmb4.Text, cmb5.Text, cmb6.Text, cmb7.Text));

ClearViewer();

break;

case ("Suppliers"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, cmb2.Text, cmb3.Text, cmb4.Text, cmb5.Text, cmb6.Text, cmb7.Text));

ClearViewer();

break;

case ("Employees"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, cmb2.Text, cmb3.Text, cmb4.Text, cmb5.Text, cmb6.Text, cmb7.Text));

ClearViewer();

break;

case ("Roles"):

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, cmb2.Text, cmb3.Text, cmb4.Text, "", "", ""));

ClearViewer();

break;

}

}

else if(DataAccess.SelectOperation == "Delete")

{

MessageBox.Show(DataAccess.ExecuteOperation(cmb1.Text, "", "", "", "", "", ""));

}

}

private void btnCancel\_Click(object sender, RoutedEventArgs e)

{

ClearViewer();

this.Hide();

}

private void btnInsert\_Click(object sender, RoutedEventArgs e)

{

ClearViewer();

DataAccess.SelectOperation = "Insert";

SetViewer();

}

private void btnUpdate\_Click(object sender, RoutedEventArgs e)

{

ClearViewer();

DataAccess.SelectOperation = "Update";

SetViewer();

}

private void btnDelete\_Click(object sender, RoutedEventArgs e)

{

ClearViewer();

DataAccess.SelectOperation = "Delete";

SetViewer();

}

private void cmb1\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

CMBSync();

}

private void SetViewer()

{

lblTitle.Text = DataAccess.SelectOperation;

txt1.Text = "";

txt2.Text = "";

txt3.Text = "";

txt4.Text = "";

txt5.Text = "";

txt6.Text = "";

txt7.Text = "";

switch (DataAccess.SelectTable)

{

case ("Customers"):

lbl1.Text = "Cell Number :";

lbl2.Text = "First Name :";

lbl3.Text = "Surname :";

lbl4.Text = "Physical Address:";

lbl5.Text = "City :";

lbl6.Text = "Units Bought:";

lbl7.Text = "";

PickLayout();

break;

case ("Orders"):

lbl1.Text = "Order Number:";

lbl2.Text = "Book Title :";

lbl3.Text = "Book Auther:";

lbl4.Text = "Book Publisher:";

lbl5.Text = "Quantity :";

lbl6.Text = "Supplier :";

lbl7.Text = "";

PickLayout();

break;

case ("Books"):

lbl1.Text = "Barcode :";

lbl2.Text = "Title :";

lbl3.Text = "Auther :";

lbl4.Text = "Publisher :";

lbl5.Text = "Stock :";

lbl6.Text = "UnitsSold :";

lbl7.Text = "Supplier :";

PickLayout();

break;

case ("Suppliers"):

lbl1.Text = "ID :";

lbl2.Text = "Company Name:";

lbl3.Text = "Address :";

lbl4.Text = "City :";

lbl5.Text = "Province :";

lbl6.Text = "Country :";

lbl7.Text = "UnitsAvailable:";

PickLayout();

break;

case ("Employees"):

lbl1.Text = "ID :";

lbl2.Text = "Name :";

lbl3.Text = "Surname :";

lbl4.Text = "Position :";

lbl5.Text = "IsStaff :";

lbl6.Text = "UnitsSold :";

lbl7.Text = "Salary :";

PickLayout();

break;

case ("Roles"):

lbl1.Text = "ID :";

lbl2.Text = "Account Name:";

lbl3.Text = "Account Password:";

lbl4.Text = "Privilege :";

lbl5.Text = "";

lbl6.Text = "";

lbl7.Text = "";

PickLayout();

break;

default:

MessageBox.Show("There was an error and the interface did not adjust to a table.");

ClearViewer();

break;

}

} // this method sets the static objects on the forms and sets their text

private void PickLayout()

{

lblTitle.Text = DataAccess.SelectOperation + " in " + DataAccess.SelectTable;

switch (DataAccess.SelectOperation)

{

case ("Insert"):

switch (DataAccess.SelectTable)

{

case ("Customers"):

Layout1();

break;

case ("Orders"):

Layout2();

break;

case ("Books"):

Layout3();

break;

case ("Suppliers"):

Layout4();

break;

case ("Employees"):

Layout5();

break;

case ("Roles"):

Layout6();

break;

}

DisplayDataSets();

break;

case ("Update"):

Layout7();

DisplayDataSets();

break;

case ("Delete"):

Layout8();

DisplayDataSets();

break;

}

} //this method picks on of the provided layout methods and affects whats visible on the form

private void ClearViewer()

{

DataAccess.SelectOperation = "Select a mode!";

lblTitle.Text = DataAccess.SelectOperation;

lbl1.Text = "";

lbl2.Text = "";

lbl3.Text = "";

lbl4.Text = "";

lbl5.Text = "";

lbl6.Text = "";

lbl7.Text = "";

lbl1.Visibility = Visibility.Collapsed;

lbl2.Visibility = Visibility.Collapsed;

lbl3.Visibility = Visibility.Collapsed;

lbl4.Visibility = Visibility.Collapsed;

lbl5.Visibility = Visibility.Collapsed;

lbl6.Visibility = Visibility.Collapsed;

lbl7.Visibility = Visibility.Collapsed;

txt1.Visibility = Visibility.Collapsed;

txt2.Visibility = Visibility.Collapsed;

txt3.Visibility = Visibility.Collapsed;

txt4.Visibility = Visibility.Collapsed;

txt5.Visibility = Visibility.Collapsed;

txt6.Visibility = Visibility.Collapsed;

txt7.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Collapsed;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

cmb1.SelectedIndex = -1;

cmb2.SelectedIndex = -1;

cmb3.SelectedIndex = -1;

cmb4.SelectedIndex = -1;

cmb5.SelectedIndex = -1;

cmb6.SelectedIndex = -1;

cmb7.SelectedIndex = -1;

cmb1.Text = "";

cmb2.Text = "";

cmb3.Text = "";

cmb4.Text = "";

cmb5.Text = "";

cmb6.Text = "";

cmb7.Text = "";

cmb1.ItemsSource = null;

cmb2.ItemsSource = null;

cmb3.ItemsSource = null;

cmb4.ItemsSource = null;

cmb5.ItemsSource = null;

cmb6.ItemsSource = null;

cmb7.ItemsSource = null;

cmb1.Items.Clear();

cmb2.Items.Clear();

cmb3.Items.Clear();

cmb4.Items.Clear();

cmb5.Items.Clear();

cmb6.Items.Clear();

cmb7.Items.Clear();

} //this method resets all objects on the form to default values

private void DisplayDataSets()

{

switch (DataAccess.SelectOperation)

{

case ("Insert"):

switch (DataAccess.SelectTable)

{

case ("Orders"):

CMB2("Books", "Title");

CMB3("Books", "Auther");

CMB4("Books", "Publisher");

CMB6("Suppliers", "Name");

break;

case ("Books"):

CMB7("Suppliers", "Name");

break;

case ("Employees"):

cmb5.ItemsSource = null;

cmb5.Items.Clear();

cmb5.Items.Add("True");

cmb5.Items.Add("False");

break;

case ("Roles"):

CMB1("Employees", "ID");

cmb4.ItemsSource = null;

cmb4.Items.Clear();

cmb4.Items.Add("ADMIN");

cmb4.Items.Add("NORML");

break;

}

break;

case ("Update"):

switch (DataAccess.SelectTable)

{

case ("Customers"):

CMB1(DataAccess.SelectTable, "CellNumber");

CMB2(DataAccess.SelectTable, "FirstName");

CMB3(DataAccess.SelectTable, "Surname");

CMB4(DataAccess.SelectTable, "PhysicalAddress");

CMB5(DataAccess.SelectTable, "City");

CMB6(DataAccess.SelectTable, "UnitsBought");

break;

case ("Orders"):

CMB1(DataAccess.SelectTable, "OrderNumber");

CMB2(DataAccess.SelectTable, "BookName");

CMB3(DataAccess.SelectTable, "BookAuther");

CMB4(DataAccess.SelectTable, "BookPublisher");

CMB5(DataAccess.SelectTable, "Quantity");

CMB6(DataAccess.SelectTable, "Supplier");

break;

case ("Books"):

CMB1(DataAccess.SelectTable, "Barcode");

CMB2(DataAccess.SelectTable, "Title");

CMB3(DataAccess.SelectTable, "Auther");

CMB4(DataAccess.SelectTable, "Publisher");

CMB5(DataAccess.SelectTable, "Stock");

CMB6(DataAccess.SelectTable, "UnitsSold");

CMB7(DataAccess.SelectTable, "Supplier");

break;

case ("Suppliers"):

CMB1(DataAccess.SelectTable, "ID");

CMB2(DataAccess.SelectTable, "Name");

CMB3(DataAccess.SelectTable, "Address");

CMB4(DataAccess.SelectTable, "City");

CMB5(DataAccess.SelectTable, "Province");

CMB6(DataAccess.SelectTable, "Country");

CMB7(DataAccess.SelectTable, "UnitsAvailable");

break;

case ("Employees"):

CMB1(DataAccess.SelectTable, "ID");

CMB2(DataAccess.SelectTable, "Name");

CMB3(DataAccess.SelectTable, "Surname");

CMB4(DataAccess.SelectTable, "Position");

cmb5.Items.Add("True");

cmb5.Items.Add("False");

CMB6(DataAccess.SelectTable, "UnitsSold");

CMB7(DataAccess.SelectTable, "Salary");

break;

case ("Roles"):

CMB1(DataAccess.SelectTable, "ID");

CMB2(DataAccess.SelectTable, "AccountName");

CMB3(DataAccess.SelectTable, "AccountPassword");

cmb4.Items.Add("ADMIN");

cmb4.Items.Add("NORML");

break;

}

break;

case ("Delete"):

switch (DataAccess.SelectTable)

{

case ("Customers"):

CMB1("Customers", "CellNumber");

break;

case ("Orders"):

CMB1("Orders", "OrderNumber");

break;

case ("Books"):

CMB1("Books", "Barcode");

break;

case ("Suppliers"):

CMB1("Suppliers", "ID");

break;

case ("Employees"):

CMB1("Employees", "ID");

break;

case ("Roles"):

CMB1("Roles", "ID");

break;

}

break;

}

} // this method displays information in the comboboxes

private void CMBSync()

{

int Column = cmb1.SelectedIndex;

if (Column == -1)

{

Column++;

}

if (DataAccess.SelectOperation == "Update")

{

cmb2.Text = DataAccess.LoadDataSets(DataAccess.SelectTable).Tables[0].Rows[cmb1.SelectedIndex][1].ToString();

cmb3.Text = DataAccess.LoadDataSets(DataAccess.SelectTable).Tables[0].Rows[cmb1.SelectedIndex][2].ToString();

cmb4.Text = DataAccess.LoadDataSets(DataAccess.SelectTable).Tables[0].Rows[cmb1.SelectedIndex][3].ToString();

if (DataAccess.SelectTable != "Roles")

{

cmb5.Text = DataAccess.LoadDataSets(DataAccess.SelectTable).Tables[0].Rows[cmb1.SelectedIndex][4].ToString();

cmb6.Text = DataAccess.LoadDataSets(DataAccess.SelectTable).Tables[0].Rows[cmb1.SelectedIndex][5].ToString();

if (DataAccess.SelectTable == "Suppliers" || DataAccess.SelectTable == "Books" || DataAccess.SelectTable == "Employees")

{

cmb7.Text = DataAccess.LoadDataSets(DataAccess.SelectTable).Tables[0].Rows[cmb1.SelectedIndex][6].ToString();

}

}

}

} //this method syncronizes the combo boxes during an update operation so they all display related information

private void CMB1(string table, string column)

{

cmb1.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb1.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb1.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

} //the following CMB methods set the item indexs of the comboboxes

private void CMB2(string table, string column)

{

cmb2.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb2.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb2.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

}

private void CMB3(string table, string column)

{

cmb3.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb3.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb3.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

}

private void CMB4(string table, string column)

{

cmb4.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb4.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb4.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

}

private void CMB5(string table, string column)

{

cmb5.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb5.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb5.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

}

private void CMB6(string table, string column)

{

cmb6.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb6.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb6.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

}

private void CMB7(string table, string column)

{

cmb7.ItemsSource = DataAccess.LoadDistinctDataSets(table, column).Tables[0].DefaultView;

cmb7.DisplayMemberPath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[column].ToString();

cmb7.SelectedValuePath = DataAccess.LoadDistinctDataSets(table, column).Tables[0].Columns[0].ToString();

}

private void Layout1()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Visible;

lbl6.Visibility = Visibility.Visible;

lbl7.Visibility = Visibility.Collapsed;

txt1.Visibility = Visibility.Visible;

txt2.Visibility = Visibility.Visible;

txt3.Visibility = Visibility.Visible;

txt4.Visibility = Visibility.Visible;

txt5.Visibility = Visibility.Visible;

txt6.Visibility = Visibility.Visible;

txt7.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Collapsed;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

} //the following layout methods set the visibility of the form

private void Layout2()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Visible;

lbl6.Visibility = Visibility.Visible;

lbl7.Visibility = Visibility.Collapsed;

txt1.Visibility = Visibility.Visible;

txt2.Visibility = Visibility.Collapsed;

txt3.Visibility = Visibility.Collapsed;

txt4.Visibility = Visibility.Collapsed;

txt5.Visibility = Visibility.Visible;

txt6.Visibility = Visibility.Collapsed;

txt7.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Collapsed;

cmb2.Visibility = Visibility.Visible;

cmb3.Visibility = Visibility.Visible;

cmb4.Visibility = Visibility.Visible;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Visible;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

}

private void Layout3()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Visible;

lbl6.Visibility = Visibility.Visible;

lbl7.Visibility = Visibility.Visible;

txt1.Visibility = Visibility.Visible;

txt2.Visibility = Visibility.Visible;

txt3.Visibility = Visibility.Visible;

txt4.Visibility = Visibility.Visible;

txt5.Visibility = Visibility.Visible;

txt6.Visibility = Visibility.Visible;

txt7.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Collapsed;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Visible;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

}

private void Layout4()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Visible;

lbl6.Visibility = Visibility.Visible;

lbl7.Visibility = Visibility.Visible;

txt1.Visibility = Visibility.Visible;

txt2.Visibility = Visibility.Visible;

txt3.Visibility = Visibility.Visible;

txt4.Visibility = Visibility.Visible;

txt5.Visibility = Visibility.Visible;

txt6.Visibility = Visibility.Visible;

txt7.Visibility = Visibility.Visible;

cmb1.Visibility = Visibility.Collapsed;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

}

private void Layout5()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Visible;

lbl6.Visibility = Visibility.Visible;

lbl7.Visibility = Visibility.Visible;

txt1.Visibility = Visibility.Visible;

txt2.Visibility = Visibility.Visible;

txt3.Visibility = Visibility.Visible;

txt4.Visibility = Visibility.Visible;

txt5.Visibility = Visibility.Collapsed;

txt6.Visibility = Visibility.Visible;

txt7.Visibility = Visibility.Visible;

cmb1.Visibility = Visibility.Collapsed;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Visible;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

}

private void Layout6()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Collapsed;

lbl6.Visibility = Visibility.Collapsed;

lbl7.Visibility = Visibility.Collapsed;

txt1.Visibility = Visibility.Collapsed;

txt2.Visibility = Visibility.Visible;

txt3.Visibility = Visibility.Visible;

txt4.Visibility = Visibility.Collapsed;

txt5.Visibility = Visibility.Collapsed;

txt6.Visibility = Visibility.Collapsed;

txt7.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Visible;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Visible;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

}

private void Layout7()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Visible;

lbl3.Visibility = Visibility.Visible;

lbl4.Visibility = Visibility.Visible;

lbl5.Visibility = Visibility.Visible;

lbl6.Visibility = Visibility.Visible;

txt1.Visibility = Visibility.Collapsed;

txt2.Visibility = Visibility.Collapsed;

txt3.Visibility = Visibility.Collapsed;

txt4.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Visible;

cmb2.Visibility = Visibility.Visible;

cmb3.Visibility = Visibility.Visible;

cmb4.Visibility = Visibility.Visible;

cmb1.IsEditable = true;

cmb2.IsEditable = true;

cmb3.IsEditable = true;

cmb4.IsEditable = true;

if (DataAccess.SelectTable != "Roles")

{

txt5.Visibility = Visibility.Collapsed;

txt6.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Visible;

cmb6.Visibility = Visibility.Visible;

cmb5.IsEditable = true;

cmb6.IsEditable = true;

if (DataAccess.SelectTable == "Employees" || DataAccess.SelectTable == "Suppliers" || DataAccess.SelectTable == "Books")

{

lbl7.Visibility = Visibility.Visible;

txt7.Visibility = Visibility.Visible;

cmb7.Visibility = Visibility.Visible;

cmb7.IsEditable = true;

}

}

}

private void Layout8()

{

lbl1.Visibility = Visibility.Visible;

lbl2.Visibility = Visibility.Collapsed;

lbl3.Visibility = Visibility.Collapsed;

lbl4.Visibility = Visibility.Collapsed;

lbl5.Visibility = Visibility.Collapsed;

lbl6.Visibility = Visibility.Collapsed;

lbl7.Visibility = Visibility.Collapsed;

txt1.Visibility = Visibility.Collapsed;

txt2.Visibility = Visibility.Collapsed;

txt3.Visibility = Visibility.Collapsed;

txt4.Visibility = Visibility.Collapsed;

txt5.Visibility = Visibility.Collapsed;

txt6.Visibility = Visibility.Collapsed;

txt7.Visibility = Visibility.Collapsed;

cmb1.Visibility = Visibility.Visible;

cmb2.Visibility = Visibility.Collapsed;

cmb3.Visibility = Visibility.Collapsed;

cmb4.Visibility = Visibility.Collapsed;

cmb5.Visibility = Visibility.Collapsed;

cmb6.Visibility = Visibility.Collapsed;

cmb7.Visibility = Visibility.Collapsed;

cmb1.IsEditable = false;

cmb2.IsEditable = false;

cmb3.IsEditable = false;

cmb4.IsEditable = false;

cmb5.IsEditable = false;

cmb6.IsEditable = false;

cmb7.IsEditable = false;

}

}

}

**Data Access**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SQLite;

using System.Data;

namespace TruText\_Ledger

{

public class DataAccess

{

public static string DBCONN = @"DATA SOURCE = TruTexts\_Customers.db;"; //this is the connection string for where the database is which is TruText\_Ledger\TruText\_Ledger\bin\Debug

public static string SelectPrivilege { get; set; } //this is the current privilege of the user between Admin and Normal and can determine what user restrictions there are.

public static string SelectTable { get; set; }//this is the current table being worked on

public static string SelectOperation { get; set; }//this is the current operation being done between insert, update and delete.

public static void ConnectToDatabase()

{

using (SQLiteConnection TruTextsCustomers = new SQLiteConnection(DBCONN))

{

TruTextsCustomers.Open();

//6/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

string tableQueryCustomers = "CREATE TABLE IF NOT EXISTS" +

" Customers" +

"(" +

"CellNumeber NVARCHAR(2048) PRIMARY KEY NOT NULL," +

"FirstName NVARCHAR(2048) NULL," +

"Surname NVARCHAR(2048) NULL," +

"PhyscalAddress NVARCHAR(2048) NULL," +

"City NVARCHAR(2048) NULL," +

"UnitsBought INTEGER NULL" +

")";

SQLiteCommand createTableCustomers = new SQLiteCommand(tableQueryCustomers, TruTextsCustomers);

createTableCustomers.ExecuteReader();

//6/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

string tableQueryOrders = "CREATE TABLE IF NOT EXISTS" +

" Orders" +

"(" +

"OrderNumber NVARCHAR(2048) PRIMARY KEY NOT NULL," +

"BookName NVARCHAR(2048) NULL," +

"BookPublisher NVARCHAR(2048) NULL," +

"BookAuther NVARCHAR(2048) NULL," +

"Quantity INTEGER NULL ," +

"Supplier NVARCHAR(2048) NULL" +

")";

SQLiteCommand createTableCommandOrders = new SQLiteCommand(tableQueryOrders, TruTextsCustomers);

createTableCommandOrders.ExecuteReader();

//7/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

string tableCommandBooks = "CREATE TABLE IF NOT EXISTS" +

" Books" +

"(" +

"Barcode NVARCHAR(2048) PRIMARY KEY NOT NULL," +

"Title NVARCHAR(2048) NULL," +

"Auther NVARCHAR(2048) NULL," +

"Publisher NVARCHAR(2048) NULL," +

"Stock NVARCHAR(2048) NULL," +

"UnitsSold INTEGER NULL," +

"Supplier NVARCHAR(2048) NULL " +

")";

SQLiteCommand createTableBooks = new SQLiteCommand(tableCommandBooks, TruTextsCustomers);

createTableBooks.ExecuteReader();

//7/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

string tableQuerySuppliers = "CREATE TABLE IF NOT EXISTS" +

" Suppliers" +

"(" +

"ID NVARCHAR(2048) PRIMARY KEY NOT NULL," +

"Name NVARCHAR(2048) NULL," +

"Address NVARCHAR(2048) NULL," +

"City NVARCHAR(2048) NULL," +

"Province NVARCHAR(2048) NULL, " +

"Country NVARCHAR(2048) NULL," +

"UnitsAvailable INTEGER NULL" +

")";

SQLiteCommand createTableSuppliers = new SQLiteCommand(tableQuerySuppliers, TruTextsCustomers);

createTableSuppliers.ExecuteReader();

//7/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

string tableQueryEmployees = "CREATE TABLE IF NOT EXISTS" +

" Employees" +

"(" +

"ID NVARCHAR(2048) PRIMARY KEY NOT NULL," +

"Name NVARCHAR(2048) NULL," +

"Surname NVARCHAR(2048) NULL," +

"Position NVARCHAR(2048) NULL," +

"IsStaff NVARCHAR(2048) NULL," +

"UnitsSold INTEGER NULL," +

"Salary NVARCHAR(2048) NULL" +

")";

SQLiteCommand createTableEmployees = new SQLiteCommand(tableQueryEmployees, TruTextsCustomers);

createTableEmployees.ExecuteReader();

//6/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

string tableQueryRoles = "CREATE TABLE IF NOT EXISTS" +

" Roles" +

"(" +

"ID NVARCHAR(2048) PRIMARY KEY NOT NULL," +

"AccountName NVARCHAR(2048) NULL," +

"AccountPassword NVARCHAR(2048) NULL," +

"Privilege NVARCHAR(2048) NULL " +

")";

SQLiteCommand createTableRoles = new SQLiteCommand(tableQueryRoles, TruTextsCustomers);

createTableRoles.ExecuteReader();

///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

}

} //this method creates a new database if one does exist

public static int Login(string sName, string sPassword)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "SELECT \* FROM Roles WHERE AccountName ='" + sName + "' AND AccountPassword = '" + sPassword + "'" +

"AND Privilege = 'NORML';";

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

SQLiteDataReader DataReader = createCommand.ExecuteReader();

int iCount = 0;

while (DataReader.Read())

{

iCount++;

}

SelectPrivilege = "NORML";

if (iCount != 1)

{

sqlQuery = "SELECT \* FROM Roles WHERE AccountName ='" + sName + "' AND AccountPassword = '" + sPassword + "'" +

"AND Privilege = 'ADMIN';";

createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

DataReader = createCommand.ExecuteReader();

iCount = 0;

while (DataReader.Read())

{

iCount++;

}

SelectPrivilege = "ADMIN";

}

dbConn.Close();

return iCount;

}

} //this method logs the user in by comparing entered details with existing ones in the database and counting them

public static DataTable LoadToDataGrid(string FilterType)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "";

switch (FilterType)

{

default:

sqlQuery = "SELECT \* FROM " + SelectTable + " ";

break;

case ("MstSld"):

sqlQuery = "SELECT Title,UnitsSold FROM Books ORDER BY UnitsSold DESC ";

break;

case ("BstCst"):

sqlQuery = "SELECT FirstName,Surname,UnitsBought FROM Customers ORDER BY UnitsBought DESC";

break;

case ("MstSls"):

sqlQuery = "SELECT Name,Surname,UnitsSold FROM Employees ORDER BY UnitsSold DESC";

break;

case ("MstPrvd"):

sqlQuery = "SELECT Name,UnitsAvailable FROM Suppliers ORDER BY UnitsAvailable DESC";

break;

}

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

SQLiteDataAdapter AdaptData = new SQLiteDataAdapter(createCommand);

DataTable CurrentTable = new DataTable(SelectTable);

AdaptData.Fill(CurrentTable);

AdaptData.Update(CurrentTable);

dbConn.Close();

return CurrentTable;

}

} //this method loads the tables from the database to the program DataGrid to be viewed

public static DataSet LoadDistinctDataSets(string Table, string Column)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "SELECT DISTINCT "+ Column + " FROM '"+ Table + "'";

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

SQLiteDataAdapter AdaptData = new SQLiteDataAdapter(createCommand);

DataSet CurrentTable = new DataSet();

AdaptData.Fill(CurrentTable);

dbConn.Close();

return CurrentTable;

}

} //this method loads distinct entries in a column and puts it in comboboxe's itemsource to be used

public static DataSet LoadDataSets(string Table)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "SELECT \* FROM '" + Table + "'";

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

SQLiteDataAdapter AdaptData = new SQLiteDataAdapter(createCommand);

DataSet CurrentTable = new DataSet();

AdaptData.Fill(CurrentTable);

dbConn.Close();

return CurrentTable;

}

} //this method is used to update comboboxes text to show what row is selected by use of the primary column items

public static string ExecuteOperation(string s1, string s2, string s3, string s4, string s5, string s6, string s7)

{

switch (SelectOperation)

{

case ("Insert"):

InsertToTable(s1, s2, s3, s4, s5, s6, s7);

return "Your information has been Saved to the database!";

case ("Update"):

UpdateToTable(s1, s2, s3, s4, s5, s6, s7);

return "Your information has been Updated on the database!";

case ("Delete"):

DeleteToTable(s1, s2, s3, s4, s5, s6, s7);

return "Your information has been Deleted from the database!";

default:

return "Select a mode!";

}

} // this method is used to select what operation to preform and return messages for user input

public static void InsertToTable(string s1, string s2, string s3, string s4, string s5, string s6, string s7)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "INSERT INTO " + SelectTable + " ( "+ SelectColumns() + " ) " +

"VALUES ("+SelectValues(s1, s2, s3, s4, s5, s6, s7)+")";

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

dbConn.Close();

}

} //this method inserts to the database

public static void UpdateToTable(string s1, string s2, string s3, string s4, string s5, string s6, string s7)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "UPDATE '"+ SelectTable +"' " +

"SET "+ SelectValues(s1, s2, s3, s4, s5, s6, s7) +";";

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

dbConn.Close();

}

} //this method updates to the database

public static void DeleteToTable(string s1, string s2, string s3, string s4, string s5, string s6, string s7)

{

using (SQLiteConnection dbConn = new SQLiteConnection(DBCONN))

{

dbConn.Open();

string sqlQuery = "DELETE FROM '" + SelectTable + "' " +

SelectValues(s1, s2, s3, s4, s5, s6, s7) + ";";

SQLiteCommand createCommand = new SQLiteCommand(sqlQuery, dbConn);

createCommand.ExecuteNonQuery();

dbConn.Close();

}

} //this method deletes in the database

public static string SelectColumns()

{

string sColumns = "";

switch (SelectTable)

{

case ("Customers"):

sColumns = "CellNumber, FirstName, Surname, PhysicalAddress, City, UnitsBought";

break;

case ("Orders"):

sColumns = "OrderNumber, BookName, BookPublisher, BookAuther, Quantity, Supplier";

break;

case ("Books"):

sColumns = "Barcode, Title, Auther, Publisher, Stock, UnitsSold, Supplier";

break;

case ("Suppliers"):

sColumns = "ID, Name, Address, City, Province, Counrty, UnitsAvailable";

break;

case ("Employees"):

sColumns = "ID, Name, Surname, Position, IsStaff, UnitsSold, Salary";

break;

case ("Roles"):

sColumns = "ID, AccountName, AccountPassword, Privilege";

break;

}

return sColumns;

} // this method selects what columns are going to be used in a operation method

public static string SelectValues(string s1, string s2, string s3, string s4, string s5, string s6, string s7)

{

string sValues = "";

switch (SelectOperation)

{

case ("Insert"):

if (SelectTable == "Roles")

{

sValues = "'" + s1 + "', '" + s2 + "', '" + s3 + "', '" + s4 + "'";

}

else if (SelectTable == "Orders" || SelectTable == "Customers")

{

sValues = "'" + s1 + "', '" + s2 + "', '" + s3 + "', '" + s4 + "', '" + s5 + "', '" + s6 + "'";

}

else

{

sValues = "'" + s1 + "', '" + s2 + "', '" + s3 + "', '" + s4 + "', '" + s5 + "', '" + s6 + "', '" + s7 + "'";

}

break;

case ("Update"):

switch (SelectTable)

{

case ("Customers"):

sValues = "FirstName = '" + s2 + "', Surname = '" + s3 + "', PhysicalAddress = '" + s4 + "', City = '" + s5 + "', UnitsBought = '" + s6 + "' " +

"WHERE CellNumber = '" + s1 + "'";

break;

case ("Orders"):

sValues = "BookName = '" + s2 + "', BookAuther = '" + s3 + "', BookPublisher = '" + s4 + "', Quantity = '" + s5 + "', Supplier = '" + s6 + "' " +

"WHERE OrderNumber = '" + s1 + "'";

break;

case ("Books"):

sValues = "Title = '" + s2 + "', Auther = '" + s3 + "', Publisher = '" + s4 + "', Stock = '" + s5 + "', UnitsSold'" + s6 + "', Supplier = '" + s7 + "' " +

"WHERE Barcode = '" + s1 + "'";

break;

case ("Suppliers"):

sValues = "Name = '" + s2 + "', Address = '" + s3 + "', City = '" + s4 + "', Province = '" + s5 + "', Country = '" + s6 + "', UnitsAvailable = '" + s7 + "' " +

"WHERE ID = '" + s1 + "'";

break;

case ("Employees"):

sValues = "Name = '" + s2 + "', Surname = '" + s3 + "', Position = '" + s4 + "', IsStaff = '" + s5 + "', UnitsSold = '" + s6 + "', Salary = '" + s7 + "' " +

"WHERE ID = '" + s1 + "'";

break;

case ("Roles"):

sValues = "AccountName = '" + s2 + "', AccountPassword = '" + s3 + "', Privilege = '" + s4 + "' " +

"WHERE ID = '" + s1 + "'";

break;

}

break;

case ("Delete"):

switch (SelectTable)

{

case ("Customers"):

sValues = "WHERE CellNumber = '" + s1 + "'";

break;

case ("Orders"):

sValues = "WHERE OrderNumber = '" + s1 + "'";

break;

case ("Books"):

sValues = "WHERE Barcode = '" + s1 + "'";

break;

case ("Suppliers"):

sValues = "WHERE ID = '" + s1 + "'";

break;

case ("Employees"):

sValues = "WHERE ID = '" + s1 + "'";

break;

case ("Roles"):

sValues = "WHERE ID = '" + s1 + "'";

break;

}

break;

}

return sValues;

} //this method concatinates string to be used in sql queries

}

}