TU2983: Advanced Databases

Lab Notes 5:

Updating and Deleting Database Data Through Visual Basic .NET

Written by:

Hafiz Mohd Sarim

Centre of Artificial Intelligence Technology
Fakulti Teknologi dan Sains Maklumat
Universiti Kebangsaan Malaysia

Table of Contents

PART 1: Introduction

A. Lab Objectives

1. This TU2983 lab module aims to fulfil the following objectives:

   i. To teach students how to use SQL UPDATE to edit existing data stored in database tables from Visual Basic .NET forms.

   ii. To teach students how to use SQL DELETE to remove existing data stored in database tables from Visual Basic .NET forms.

2. The programming language used for the TU2983 lab modules is Visual Basic .NET, while database used to hold project data will be either of the following
   a. Microsoft Access 2007 or above as the portable database
   b. IBM DB2 Enterprise Server Edition v9.7 or above as the server database.

   Students will be instructed on the exact database to use for each lab exercise and lab assignment.

3. These lab notes were written using the "**Visual Studio 2019 Enterprise**" integrated development environment (IDE) for the **Visual Basic .NET** programming language. However, the steps demonstrated in these lab notes can also be used for all versions of Visual Basic .NET in Visual Studio IDEs starting from Visual Studio .NET 2002 and above.

B. Student Requirements for this Lab

1. All students must bring the following equipment for the lab:

   - **MANDATORY REQUIREMENT:** One USB hard drive / thumb drive pen drive (to save your work)

2. All students must have the following requisite knowledge:

   - **MANDATORY REQUIREMENT:** Using the Microsoft Windows operating system.
   - **MANDATORY REQUIREMENT:** Understand the fundamentals of programming and recognize the different control statements and data structures that are generally available in all programming languages.
   - **MANDATORY REQUIREMENT:** Writing Structured Query Language code, or SQL.
   - **MANDATORY REQUIREMENT:** Creating a sample database in Microsoft Access (see TU2983 Lab Notes 1).
   - **MANDATORY REQUIREMENT:** Building windows forms in Visual Basic .NET and connecting to a Microsoft Access database (see TU2983 Lab Notes 2).

PART 2: Editing Existing Data

A. Opening an Existing VB project

1. Start Visual Studio. In the Windows desktop, click **'Start' > 'Microsoft Visual Studio 2019'**.

2. We will use the Visual Basic .NET project you created in TU2983 Lab Notes 4. In the Visual Studio start page, click on **'prj_facultyrecords_a123456'**, which is listed under the **'Recent Projects'** list. Your project will open.

3. Alternatively, you can also browse to the project folder contained in your working folder, for example **'C:\a123456_facultyrecords\prj_facultyrecords_a123456'**, and double-click on the ***.sln** file (solution file), for example **'prj_facultyrecords_a123456.sln'**.

4. At minimum, verify that you have the following forms and modules:
   - A main menu form (**frm_mainmenu_a123456.vb**) [Lab Notes 2]
   - A module containing your global variables (**mod_globals_a123456**) [Lab Notes 3]
   - A connection string that has been cast as an **OleDb.OleDbConnection** data type, which has been declared as a global variable in your **Globals_Module** [Lab Notes 4] as shown below:

```
Module mod_globals_a123456

        Public myconnection as String = "Provider=Microsoft.ACE.OLEDB.12.0;Data
        Source=DB_FACULTYRECORDS_A123456.accdb;Persist Security Info=False;"

        Public myconnection2 as New OleDB.OleDbConnection(myconnection)

End Module
```

5. If you do not have this form, module, or connection string, please complete TU2983 Lab Notes 2, 3, and 4 before proceeding any further.

B. Creating a Editing Form

1. In the Solution Explorer list, add a new form called '**frm_updatecourses_a123456.vb**'.

2. Double-click on the 'My Project' item in the Solution Explorer list and change your **'Startup form'** to '**frm_updatecourses _a123456.vb**'.

3. From your toolbox, add the following objects to the form:

   - On the top of the form, add a **Label**. Change its **(Name)** property to **'lbl_title'**. Change the **'Text'** property of this Label to **'Update Courses'**, and redesign the font of this label as you see fit.

   - Underneath **'lbl_title'**, add a **DataGridView** object. Change its **(Name)** property to **'grd_courses'**. Expand the size of this grid such that it fills two-thirds (2/3) of the form.

   - Underneath **'grd_courses'**, add four **TextBox** objects in a single row.

- Change the **(Name)** property the of the first **TextBox** to **'txt_code'**.
- Change the **(Name)** property the of the second **TextBox** to **'txt_name'**.
- Change the **(Name)** property the of the third **TextBox** to **'txt_credit'**.
- Change the **(Name)** property the of the fourth **TextBox** to **'txt_dept'**.
- Add additional Label objects to label each textbox accordingly.

- Underneath all the textboxes, add a **Button** object. Change its **(Name)** property to **'btn_update'**. Change the **'Text'** property of this Button to **'UPDATE'.**

4. Open the **Form_Load** event for this form, by double-clicking on any blank space in the '**frm_updatecourses_a123456.vb**' form.

5. Underneath the '**End Sub**' of the **Form_Load** event, write the following to create a sub-procedure called '**refresh_grid()**':

```
Private Sub refresh_grid()

        Dim mysql as String = "SELECT * FROM TBL_COURSES_A123456"

        Dim mydatatable as New DataTable

        Dim myreader as New OleDb.OleDbDataAdapter(mysql, myconnection)

        myreader.Fill(mydatatable)

        grd_courses.DataSource = mydatatable

End Sub
```

6. Underneath the '**End Sub**' of the '**refresh_grid()**' function call, write the following to create a sub-procedure called '**clear_fields()**':

```
Private Sub clear_fields()

        txt_code.Text = ""
        txt_name.Text = ""
        txt_credit.Text = ""
        txt_dept.Text = ""

End Sub
```

7. In the **Form_Load** event add the '**refresh_grid()**' function call:

```
Private Sub frm_insertcourses_a123456_Load(sender As System.Object, e as System.EventArgs)
Handles MyBase.Load

        refresh_grid()

End Sub
```

8. Save your project and run the program.

C. Capturing Primary Key Information

1. Just under the 'Public Class' declaration of your **frm_updatecourses_a123456.vb** form, before any existing events or sub-procedures, declare a variable that will be used to store the currently selected course code.

```
Public Class frm_updatecourses_a123456

        Dim current_code as String

        ...

End Class
```

2. Create a new sub-procedure at the bottom of your form, before the **End Class**, to capture the primary key information. The primary key is always displayed in the first column of your **grd_courses**, which is column '**0**':

```
Private Sub get_current_code()

    Dim current_row As Integer = grd_courses.CurrentRow.Index

    current_code = grd_courses(0, current_row).Value

End Sub
```

3. Call the **get_current_code()** sub-procedure just after the **refresh_grid()** call in your **Form_Load** event:

```
Private Sub frm_insertcourses_a123456_Load(sender As System.Object, e as System.EventArgs)
Handles MyBase.Load

        refresh_grid()

        get_current_code()

End Sub
```

4. In your form design view, double click on **grd_courses** to generate the code-snippet for its default event, **CellContentClick**.

5. Using the event list on the top of the main source code display (with the 'lightning' icon), change the event to **CellClick**, to generate its code-snippet.

6. Call the **get_current_code()** sub-procedure in this **CellClick** event as follows:

```
Private Sub grd_courses_CellClick(sender As Object, e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles grd_courses.CellClick

        get_current_code()

End Sub
```

7. Delete the code-snippet for the previous event, **CellContentClick**.

8. Add the following lines of code to your **get_current_code()** sub-procedure, to display the contents of the selected course in the text boxes:

```
Private Sub get_current_code()

    Dim current_row As Integer = grd_courses.CurrentRow.Index

    current_code = grd_courses(0, current_row).Value

    txt_code.Text = current_code
    txt_name.Text = grd_courses(1, current_row).Value
    txt_credit.Text = grd_courses(2, current_row).Value
    txt_dept.Text = grd_courses(3, current_row).Value

End Sub
```

9. Save your project and run the program.

10. You'll notice that when the form loads, and when you click on any cell in the grid, the contents of the text box will change to reflect the row you have clicked.

D. Creating a General Purpose SQL Command Execution Sub-Procedure

1. Normally, at this point, you would add the data manipulation code to execute an SQL UPDATE command into the **'Button_Click'** event of **btn_update**.

2. However, since the SQL UPDATE command can only update one attribute value at a time, doing this means that you might need to repeat your code up to 3-times, because there are 3 non-key attributes that may have changed during editing.

3. A more efficient way of doing this is declare your data manipulation code as a global sub-procedures in your **'Globals_module'**. This global sub-procedure will serve as a general purpose SQL command execution sub-procedure that can be used anytime you need to run an SQL INSERT, UPDATE, or DELETE event.

4. Add the following code to your **'Globals_Module'** to declare this global sub-procedure:

```
Public Sub run_sql_command(thissql As String)

    Dim mywriter As New OleDb.OleDbCommand(thissql, myconnection2)

    Try

        mywriter.Connection.Open()
        mywriter.ExecuteNonQuery()
        mywriter.Connection.Close()

    Catch ex As Exception

        Beep()
         MsgBox("There is a mistake in the data you entered, as shown below" & vbCrLf & vbCrLf
         & ex.Message)

        mywriter.Connection.Close()

    End Try

  End Sub
```

E. Editing Data Using SQL UPDATE

1. Double-click on **btn_update** to generate its **'Button_Click'** event code snippet, and add the following code:

```
Private Sub btn_update_Click(sender As System.Object, e As System.EventArgs) Handles
btn_update.Click

        run_sql_command("UPDATE TBL_COURSES_A123456 SET FLD_COURSE_NAME='" &
        txt_name.Text & "' WHERE FLD_COURSE_CODE='" & current_code & "'")

        run_sql_command("UPDATE TBL_COURSES_A123456 SET FLD_CREDITS=" &
        txt_credit.Text & " WHERE FLD_COURSE_CODE='" & current_code & "'")

        run_sql_command("UPDATE TBL_COURSES_A123456 SET FLD_MANAGING_DEPT='" &
        txt_dept.Text & "' WHERE FLD_COURSE_CODE='" & current_code & "'")

        Beep()
        MsgBox("You have successfully updated the course """ & current_code & """.")

        refresh_grid()
        clear_fields()
        get_current_code()

End Sub
```

2. Note that for **txt_credit.Text** there are no enclosing double-quote marks (**""**) before or after this object property reference. This is because the '**fld_credits'** attribute that it corresponds to in the database tables is of a numeric (i.e. **Integer**) data type.

3. Alternatively you can retype the SQL UPDATE command above in a single function call as follows:

```
Private Sub btn_update_Click(sender As System.Object, e As System.EventArgs) Handles
btn_update.Click

        run_sql_command("UPDATE TBL_COURSES_A123456 SET FLD_COURSE_NAME='" &
        txt_name.Text & "',  FLD_CREDITS=" & txt_credit.Text & ", FLD_MANAGING_DEPT='" &
        txt_dept.Text & "' WHERE FLD_COURSE_CODE='" & current_code & "'")

        Beep()
        MsgBox("You have successfully updated the course """ & current_code & """.")

        refresh_grid()
        clear_fields()
        get_current_code()

End Sub
```

4. Save your project.

5. Run the program, and click on the course **'TU3983'**.

6. Change the following details only

   - **Course name = Undergraduate Project**
   - **Credits = 18**

7. When you click update, you will receive a message indicating that your update has been successful.

F. Protecting your Input Fields

1. In your form design view, click on the textbox '**txt_code'**, and change its '**ReadOnly**' property to '**True'**. This will prevent anyone from altering your existing primary keys during input, since the primary key is crucial for the successful update of existing data.

2. Click on the DataGridView '**grd_courses**', and change its '**AllowUsersToAddRows**' property to '**False**'. This will prevent users from selecting a non-data row in the grid.

3. Click on the DataGridView '**grd_courses**', and change its '**ReadOnly**' property to '**True'**. This will prevent users from altering the data while it is in the grid.

PART 3: Deleting Existing Data

A. Adding the Delete Controls

1. Underneath the 'btn_update' button, add a **Button** object. Change its **(Name)** property to **'btn_delete'**. Change the **'Text'** property of this Button to **'DELETE'.**

2. Double-click on **btn_delete** to generate its **'Button_Click'** event code snippet.

```
Private Sub btn_delete_Click(sender As System.Object, e As System.EventArgs) Handles
btn_delete.Click


End Sub
```

B. Capturing a MsgBox Response

1. We can use a MsgBox as an input box by changing the type of the message box to a "**YesNo**" type, and declaring a variable to capture the message box response.

2. Add the following lines of code to the **'Button_Click'** event for **btn_delete**.

```
Private Sub btn_delete_Click(sender As System.Object, e As System.EventArgs) Handles
btn_delete.Click

        Dim delete_confirmation = MsgBox("Are you sure you would like to delete the course """
        & current_code & """?", MsgBoxStyle.YesNo)

        If delete_confirmation = MsgBoxResult.Yes Then

                Beep()
                MsgBox("The course """ & current_code & """ has been successfully deleted.")

        End If

End Sub
```

3. The response message contained in the IF-THEN statement will only execute if the user clicked '**Yes'** in the MsgBox dialog.

C. Deleting Data Using SQL DELETE

1. Since we now have a global sub-procedure to execute all INSERT, UPDATE, and DELETE commands, we can call the same sub-procedure to execute our SQL DELETE command.

2. Add the highlighted lines of code below in between the IF-THEN statement of the '**Button_Click'** event from **btn_delete**:

```
Private Sub btn_delete_Click(sender As System.Object, e As System.EventArgs) Handles
btn_delete.Click

        Dim delete_confirmation = MsgBox("Are you sure you would like to delete the course """ &
        current_code & """?", MsgBoxStyle.YesNo)

        If delete_confirmation = MsgBoxResult.Yes Then

                run_sql_command("DELETE FROM TBL_COURSES_A123456 WHERE
                FLD_COURSE_CODE='" & current_code & "'")

                Beep()
                MsgBox("The course """ & current_code & """ has been successfully deleted.")

                refresh_grid()
                clear_fields()
                get_current_code()

        End If

End Sub
```

3. Save your project, and run the program.

4. Click on the course **'TU3983'** in the DataGridView '**grd_courses'**, and press the DELETE button. Click '**Yes'** when the message box appears.

5. The entire row for the course **'TU3983'** will be deleted when the DataGridView '**grd_courses'** refreshes.

EXERCISE 1: Create an Editing form for the 'Students' table (TBL_STUDENTS_A123456) and the 'Department' table (TBL_DEPARTMENT_A123456).

PART 4: Before Leaving the Lab

1. Copy your entire working folder, e.g **'c:\a123456_facultyrecords'**, to your USB drive. This folder will be used again for the next lab session.

2. Delete the existing **'c:\a123456_facultyrecords'** folder in your lab computer's **'C:\'** drive.

Additional Lab Exercises

1. Modify your Student Details form by adding buttons to UPDATE and DELETE existing data from the student table.

2. Create a new form to INSERT, UPDATE and DELETE data from the Department table, all in a single form.

3. Create a new form, or modify and existing form, to change or delete a student's picture. If the student's picture has been deleted, replace it with a default picture. ***IMPORTANT NOTE:*** *Visual Studio will prevent you from deleting a picture that has previously been opened in a form. This restriction is imposed by Visual Studio by design.* Therefore, come up with a way get around this restriction so that you can change a student's picture.