Visual Basic Database Connectivity

An Introductory Guide

Create the VB .Net Application

Create a blank VB .Net solution in your account and add a new project to the solution. This should automatically create a blank form for you. Also copy the **Authors.mdb** database file from the COMP315 library folder into your account.

Using Data Adapters and Data Sets

To manipulate data in an access database you need to use a data adapter to connect to the database and a data set to manipulate the data.

Creating the Data Adapter

To create a Data Adapter that contains the SQL statement you want to populate the dataset with, do the following:

Click on the 'Data' tab in the tool box and then double click on the **OleDbDataAdapter** control to add it into the form. The Data Adapter Configuration Wizard will automatically appear to help you configure the Data Adapter.

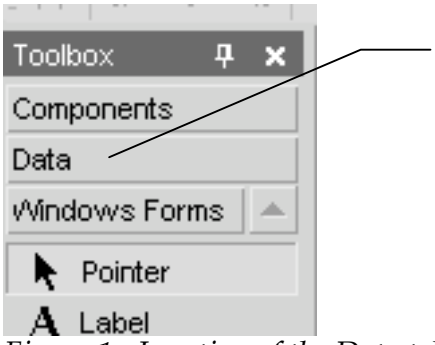

Figure 1. Location of the Data tab in the toolbox.

Click on **Next>** to move to the next screen. Now the Data connection to use box should be blank so click on the **New Connection** button to configure the new connection. Select the **Provider** tab and then select **Microsoft Jet 4.0 OLE DB Provider** and then click on **Next>**.

In section 1 where it asks for the database to use, click on the '**...**' button and then navigate to and select the **Authors.mdb** database and click on the **Open** button.. Leave section 2 as it is. Your screen should look similar to this:

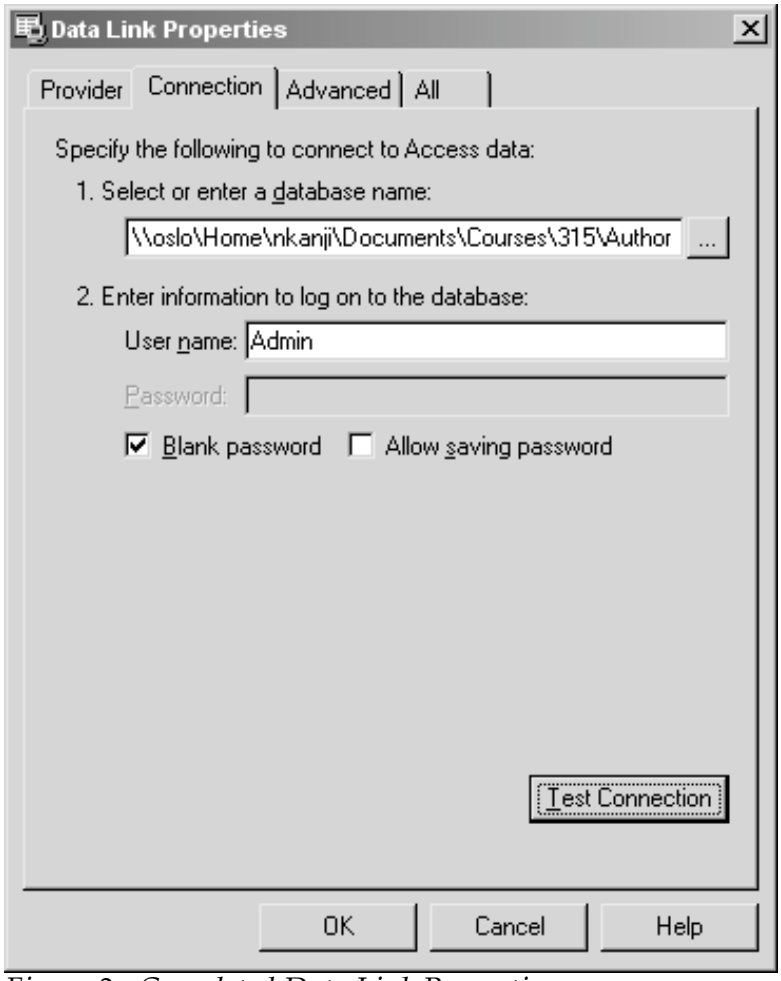

Figure 2. Completed Data Link Properties.

Click on the **Test Connection** button and you should get a message saying 'Test Connection Succeeded'. If you don't then check you have followed the instructions above. Click on **OK** to return to the Data Adapter Wizard and then click on the **Next>** button.

Make sure that **Use SQL statements** is selected and then click on the **Next>** button.

In the large text box type in the following SQL statement:

```
Select * from authors
```

Then click on the **Next>** button and then the **Finish** button. You may be asked if you want to include a password in the connection

string, so just click on **Don't Include Password**. You should now see a new Adapter object and a new Connection object.

Creating the Dataset

The next step is to create a Dataset class based on the query you specified above. Make sure that your Data Adapter object is selected and at the bottom of the properties pane click on the **Generate Dataset…** link and you should see the window below appear.

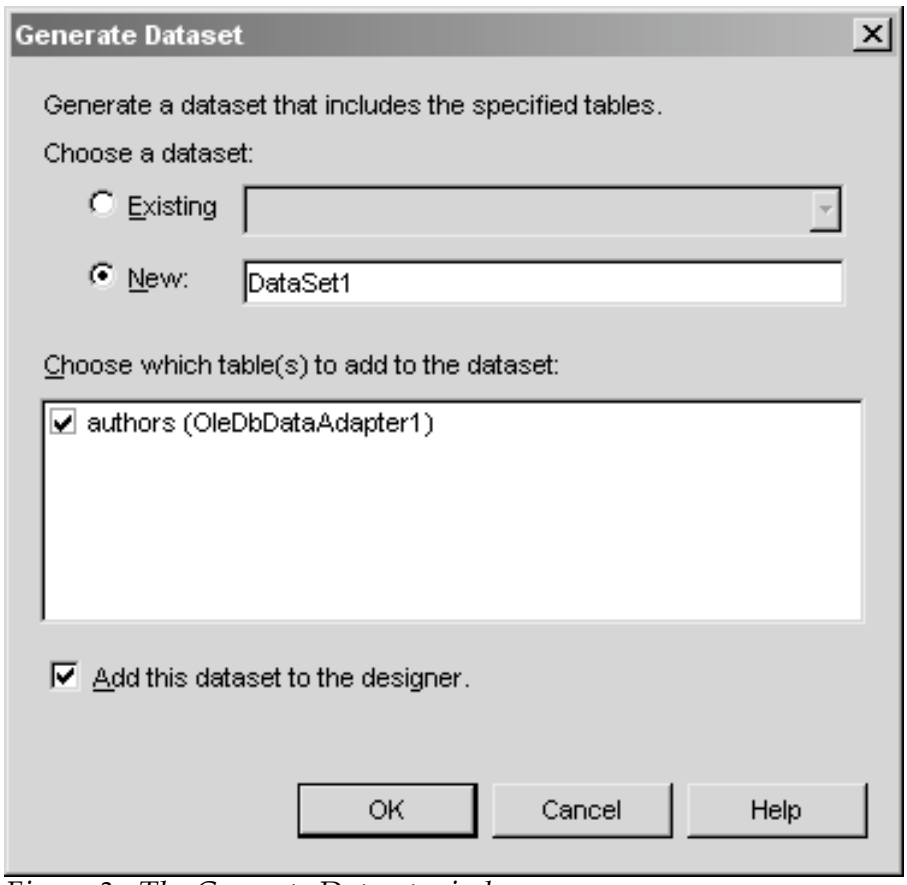

Figure 3. The Generate Dataset window.

Select 'New' data set rather than 'Existing' data set and give it the name **dsAuthors** and make sure the 'Add This Dataset to the Designer' option is selected. Click on the **OK** button to generate the authors dataset and you should see the Dataset object 'dsAuthors1' added to your project.

Adding a Data Grid Control

Now you need to add a data grid control to your form. Do this by selecting the Windows Forms tab in the tool box and then double click on the Data Grid control in the toolbox to add it to your form. Move it around and make it bigger.

Go to the 'DataSource' property of the data grid and click on the down arrow and select **dsAuthors1**. For the 'DataMember' property select the **Authors** option. This has effectively bound the data grid control to your authors dataset.

Filling the Data Set With Information

Now run the application, what do you see? There is no data in the data grid! What you need to do is to fill the dataset with the appropriate data. Stop the application running and then add a button and name it appropriately and give it the caption **Display Data**. Now create a click event procedure for that button and in the click event procedure type in the following:

```
DsAuthors1.Clear()
OleDbDataAdapter1.Fill(DsAuthors1, "Authors")
```

The first statement clears the dataset of previous information and the second statement uses the fill method of the data adapter to get the new data from the database. The first parameter of the Fill method is the dataset to fill and the second parameter is the name of the table that you used when you set the DataMember property of the data grid control.

Now run your application again and when you click on your button you should see all the author information appear. If you click on the column names it will sort the data on that column. Pretty cool eh!

Updating Changes Made in the Data Grid

You can also change the information by clicking on a cell and then changing the data. Sort by last name and change the surname of 'Ann Dull' to 'Ann Exciting'. Close the application and run it again. What happened to the change you made to the surname? You should find that the surname is still 'Dull'. The reason why, is that the change was made to the data set but not to the database. The data set is like a temporary table in the computers memory so any changes to it doesn't affect the data in the actual table in the access database.

You need to use the Update method of the Data Adapter to save any changes made to the dataset to the database.

To do this first add another button the form and give it the caption **Save** and then create a click event procedure for the button. Then type in the following statement in the click event procedure:

```
OleDbDataAdapter1.Update(DsAuthors1)
```

This statement calls the Update method of the Data Adapter and you need to specify the dataset to update as the parameter.

Now run the application again and change the surname again and then click on the **Save** button. Close the application and run it again and you should see the change has been made to the database now. Try adding a new row at thee bottom of the data grid using your name and make up data for the other attributes. Be careful to use the correct format for data otherwise you may get an error if the format of what you enter does not match. Don't forget to click on 'Save' to make the change to the database.

Deleting Information using the Data Grid

Deleting a row from the table is also very easy. Just click in the area at the very beginning of the row before the first column to highlight the entire row. Now press the **Delete** key on your keyboard to delete the row and click on the **Save** button to save your changes to the database. In fact you don't need to click on save after every change, just make all the changes to want to make and then click on the Save button at the very end and all changes will be saved to the database. Too Easy!

Filtering the Data Set

Now lets try filtering the data so that we only see the data we are interested in. We will use a combo box to allow the user to enter a value for the State and then only show authors living in that state in the data grid. We will need a second data adapter to retrieve back all of the states in the database. Add another data adapter like you did for the first one and you should find that the database connection you used for the first one is already present so use that. When you type in your SQL statement, type in the following:

```
Select distinct state from authors
```

Then click on the **Advanced Options** button and clear the option for 'Generate Insert, Update and Delete statements' as shown below:

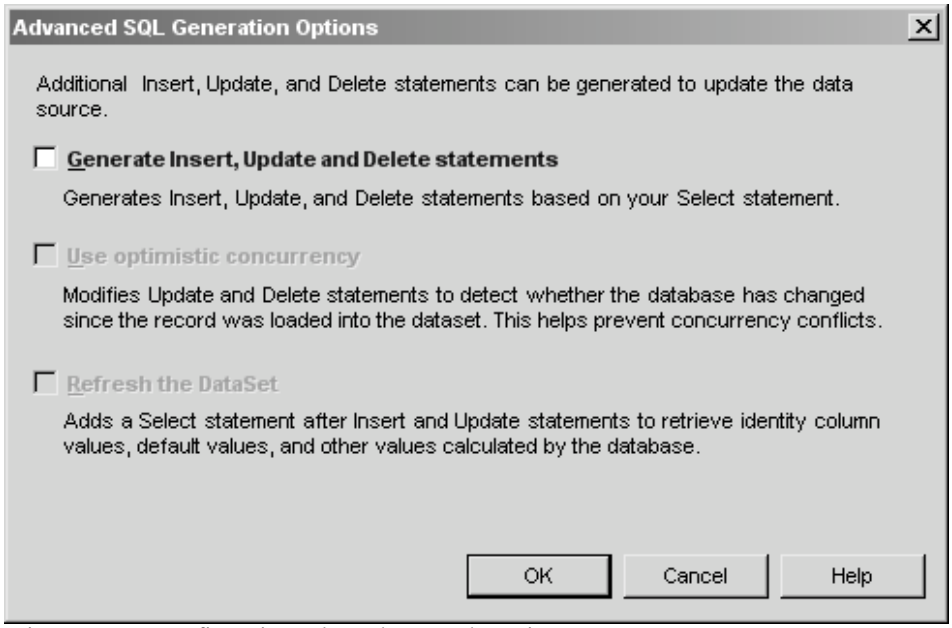

Figure 4. Configuring the advanced options.

Then click on **Next>** and **Finish** to create the second data adapter. At this point it would be better to rename this adapter so it easy to remember what the adapter is for so select the second data adapter object and in the properties pane change the name to **sdaState**. Now create a second data set with the name **dsState** from your sdaState data adapter..

Filling a Combo Box With Data From a Data Set

You are ready to create your combo box now so double click on the combo box control in the tool box to add it to your project and give it the name **cboStates**. Set the 'DataSource' property to **dsState1** and the 'DisplayMember' property to **Authors.State**, you will need to click the drop down arrow and then expand the Authors option to display the list of columns you can use. Set the 'DropDownStyle' property to **DropDownList** and set the 'ValueMember' property of the combo box to **Authors.State**. This is used so that the actual value is used when an item is selected in the list.

So we have the data set ready and the combo box ready, the last thing we need to do us to populate the combo box automatically when the application runs. To do this we need to create a Form Load event procedure which will execute the statements in the procedure when the form is loaded. Just double click on an empty area of the form to create the form load event procedure and then add the following statements:

```
DsState1.Clear()
sdaState.Fill(DsState1, "authors")
```

Now run your application and you should see in the dropdown list of the combo box all of the states in the authors table.

Changing the Select Command of the Data Adapter

What the application should do now is that when the user changes the value in the combo box the data grid should refresh with new information based on the State the user has chosen in the combo box.

Double click on your combo box to create the 'SelectedIndexChanged' event procedure. The procedure is executed whenever the value in the combo box changes. Type in the following statements in the event procedure:

```
OleDbDataAdapter1.SelectCommand.CommandText = ↲
"SELECT * FROM authors WHERE state = '" & ↲
cboState.Text & "'"

DsAuthors1.Clear()
SqlDataAdapter1.Fill(DsAuthors1, "authors")
```

The first statement sets the new query in the data adapter when the data set is generated. The value in the combo box is inserted into the query in the Where clause. Since the State value will be a string it needs single quotes around it. This statement should be on one line in your VB code. Then the data set is cleared and filled again.

Now run your application and when you select different states in the combo box it will automatically update the data grid showing authors in that state.

Using More Than One Filter

Now lets filter the data further by letting the user choose the state and whether the author has a contract or not. Add another combo box and click on the **…** button of the 'Items' property. Then in the text box type in **True** on one line and **False** on the next line and then click on **OK**. This sets the combo box with just these two items when the application is run.

Also set the 'DropDownStyle' property to **DropDownList**, this means that the user can't directly type something into the text box, they need to specifically select an item from the drop down list. A very good way of restricting the user to select from specific values.

Since we have two combo boxes it doesn't make sense to refresh the data when only one of them changes, rather, the user sets the two combo box values and then will click on a Display Data button.

Delete the code in the click event procedure for the Display Data button and then copy the code from the SelectedIndexChanged event procedure for the state combo box and paste it into the click event for the Display Data button. Now delete the SelectedIndexChanged event procedure.

Add the following to the click event procedure for the Display Data button so that it looks like this:

```
OleDbDataAdapter1.SelectCommand.CommandText = ↲
"SELECT * FROM authors WHERE state = '" & ↲
cboState.Text & "'"

If cboContract.Text = "True" Then
   OleDbDataAdapter1.SelectCommand.CommandText = ↲
   OleDbDataAdapter1.SelectCommand.CommandText & ↲
   " AND contract = -1"
Else
   OleDbDataAdapter1.SelectCommand.CommandText = ↲
    OleDbDataAdapter1.SelectCommand.CommandText & ↲
   " AND contract = 0"
End If

DsAuthors1.Clear()
OleDbDataAdapter1.Fill(DsAuthors1, "authors")
```

The If statement is needed because Access stores True as the value -1 and False as the value 0.

Now run and test the application with different state values and different contract values.

Making the Data Grid Read-Only

Sometimes you might not want to let the user make changes in the data grid and just use it for displaying information. To do this you can set the 'ReadOnly' property of the data grid to **True**. Do this for the data grid on your form. Now add another button to the form and give it the caption **Display Author**. Create a click event procedure for the button and we will write the statements necessary so that when the user click on an author id in the data grid and clicks on the Display Author button, it will display a new form to display information just for that author.

Adding a Separate Display Author Form

You will need to add a new form to the project first, do this by selecting **Add Windows Form…** from the **Project** menu and then click on the **OK** button. Change the 'Name' property of the form to **frmDisplayAuthor**. Add a button to the form and give it the caption **Close**.

Now go back to your main form with the data grid and in the click event procedure for the DisplayAuthor button type in the following statements:

```
Dim DisplayForm As New frmDisplayAuthor

DisplayForm.Show()
```

The first statement will create a new form object with the name DisplayForm based on the Display Author form we are designing. The second statement will display the form on the screen using the DisplayForm object we have just created.

In the click event procedure for the Close button in the Display Author form type in:

```
Me.Close()
```

The Me part of the statement means the current form and the Close method will close the current form and take you back to the previous form.

Now run your application and click on the **Display Author** button to show the second form and click on the **Close** button to take you back to the main form then close the application.

In your Display Author form create a new Ole data adapter and when asked for a query type in:

```
Select * from authors
```

Then create a new data set and call it **dsSingleAuthor** as we will use it to get information on a single author.

Now on the form create text boxes for each attribute in the authors table and give them appropriate names.

Now we need to bind each text box to a field in the table so that the information is displayed in the text box and any changes will be saved to the database.

Select the Author ID text box and then expand the **DataBindings** property. Then select the **Text** property and click on the down arrow to display the dropdown list. Now expand **dsSingleAuthor1** and then expand **Authors** to see all of the fields and then select **au_id**. The text box is now bound to the author id field in the database. Bind the other text boxes to the appropriate attributes in the data set.

Now go back to your main form and change the click event procedure for the Display Author button to look like this:

```
Dim DisplayForm As New frmDisplayAuthor

DisplayForm.OleDbDataAdapter1.SelectCommand.↲
CommandText = "select * from authors where ↲
au_id = '" & DataGrid1.Item(DataGrid1.CurrentCell) ↲
& "'"

DisplayForm.Show()
```

Since we are accessing the Data Adapter in the second form we need to put DisplayForm first before the name of the SQL Data Adapter. The part after thewhere clause means get the value in the cell that is currently selected in the data grid (assuming your data grid has the name DataGrid1). So this will set up the SQL statement in the Data Adapter to retrieve the information for the author who has the author id selected in the data grid.

The last thing to do is to fill the data adapter with the information for the author we want. Go to the Display Author form and create a form load event procedure to look like this:

```
DsSingleAuthor1.Clear()
OleDbDataAdapter1.Fill(DsSingleAuthor1, "authors")
```

So when the form loads it will fill the data set based on the SQL statement in the Data Adapter. Now run the application and select a state and contract status. Then click on an author id in the data grid and then click on the Display Author button.

The second form should appear with the authors id, surname and firstname. Click on the close button to go back to the main form. Try displaying different authors to make sure it is working properly.

Saving Changes Using the Bound Controls

Now we need to be able to make changes and save the changes to the database from the Display Author form. Can you think how to do this? Yes you are correct, we just need to have a Save button. Add a Save button to the Display Author form. Create a click event procedure for the Save button and type in the following statements:

```
DsSingleAuthor1.authors.Rows(0).EndEdit()
OleDbDataAdapter1.Update(DsSingleAuthor1)
```

The first statement tells the data set that you have finished editing the data in the data set and the second statement obviously updates the database with any changes.

Now run the application and display some records in the data grid, then click on an individual author id to select it and then click on your Display Author button. The second form should appear with all of the information displayed. You can now change any of the information and when you click on the Save button all of the changes should be saved back to the database.

Adding a New Author Using a Separate Form

Now you will learn how to add a new author into the data base using text boxes or other types of controls. You could do it through the data grid but it is not the best type of control for this as you can't use combo boxes or other types of controls to restrict the values the user can input.

Add a new form to the project and give it the name **frmNewAuthor**. Go back to your Display Author form and copy all of the text boxes and labels and paste them into the New Author form. That way you don't have to manually create them again and all the controls have the same name. Make sure that the controls are not bound to any fields in the database as you can't use bound controls when adding a new row into the database.

Create a new Data Adapter and use the following query when asked:

```
SELECT *
FROM authors
WHERE au_id = -1
```

The reason for the where clause is so that an empty data set it returned as we are only adding a new author and we don't want to retrieve other author information. Then create a new data set called **dsNewAuthor**.

Now create a new button on the form with the caption **Add Author**. The user will fill the text boxes with appropriate information and then click on this button when they are ready to add the new author.

Now create a click event procedure for this button and then type in the following code:

```
Dim AuthorRow As DataRow =
DsNewAuthor1.Tables("authors").NewRow

AuthorRow("au_id") = txtAuthorID.Text
AuthorRow("au_lname") = txtSurname.Text
AuthorRow("au_fname") = txtFirstname.Text
AuthorRow("phone") = txtPhone.Text
AuthorRow("address") = txtAddress.Text
AuthorRow("city") = txtCity.Text
AuthorRow("state") = txtState.Text
AuthorRow("zip") = txtZip.Text
If txtContract.Text = "True" Then
       AuthorRow("contract") = -1
Else
       AuthorRow("contract") = 0
End If

AuthorRow.EndEdit()
Try
       DsNewAuthor1.Tables("authors").Rows.Add ↲
       (AuthorRow)
       SqlDataAdapter1.Update(DsNewAuthor1, ↲
       "authors")
       DsNewAuthor1.AcceptChanges()
Catch objException As Exception
       MessageBox.Show(objException.Message)
End Try
```

In the Dim statement we create a new row based on the Authors table, this is separate from the data set. Then we populate each field in the row with the appropriate data from the text boxes. We then call the EndEdit method of the AuthorRow object to say that we have finished editing the row.

The Try..Catch..End Try structure is used to catch any errors which might occur otherwise the error may crash your application back to the operating system which is not a good thing. After the Try keyword we have the statements which we want to execute. In the Catch structure we use the objException object to catch the error and the statements in that section will display the error message in a message box window. This means the user is alerted to the error but the application does not crash. You should use the Try..Catch structure whenever trying to do something with the database like updated or filling a data set, etc so that your application won't crash.

The first statement in the Try section add our row to the dataset and the next statement updates the database and the third tells the dataset to accept the changes.

You should also have a close button on the form as well which closes the form.

Now on your main form create a button with the caption **New Author** and in the click event procedure type in the following:

```
Dim NewAuthorForm As New frmNewAuthor

NewAuthorForm.Show()
```

Now run your application and click on the New Author button. The form should appear and type in the following information:

```
Author id:  213-46-9999   *
Surname:    Kanji
Firstname:  Nilesh
Phone:      415 986-9999   *
Address:    111 Prancing Pony Rd
City:       Minas Tirith
State:      CA   *
Zip:        90210   *
Contract:   True   *
```

The values with stars means you must type them in exactly as stated otherwise you will get an error because the table has format checks on those attributes. Now click on the Add Author button and then click on the Close button.

On the main form get the data grid to display Authors with True for contract and you should see the new author in the table. You could change some of the text boxes on the New Author form to use combo boxes or other types of controls to make it easier for the user to use.

You should now have all of the information you need to build the application for the project. You can lots more information by searching the Microsoft website or just searching the web using Google.